

Publication Number 87000064
Release 2.0
March 1983

9520 SOFTWARE DEVELOPMENT SYSTEM

USERS MANUAL

Gould Inc., Design & Test Systems Division
4600 Old Ironsides Drive
Santa Clara, CA 95051-1279
Telephone: (408) 988-6800
TWX/TELEX # 910-338-0509

Copyright © 1983. No part
of this publication may be
reproduced without written
permission from Gould, Inc.,
Design & Test Systems Division
Printed in U.S.A. 3/83

WARNING:

This equipment generates, uses and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. As temporarily permitted by regulation, it has not been tested for devices pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

The following procedures may help to alleviate the Radio or Television Interference Problems:

1. Reorient the antenna of the receiver receiving the interference.
2. Relocate the equipment causing the interference with respect to the receiver (move or change relative position).
3. Reconnect the equipment causing the interference into a different outlet so the receiver and the equipment are connected to different branch circuits.
4. Remove the equipment from the power source.

NOTE:

The user may find the following booklet prepared by the FCC helpful: "How to Identify and Resolve Radio-TV Interference Problems". This booklet is available from the U.S. Printing Office, Washington, D.C. 20402. Stock No. 004-000-00345-4.

This manual is designed to serve as a reference manual and as a guide to exercising the Gould Inc., Instruments Division 9520 Software Development System and its terminal the 9501. The manual contains introductory, installation and operational information and a description of system software. For specific examples and detailed explanations of the operating system software, the operator is referred to the MP/M User's Guide and the CP/M Handbook that are shipped with the system.

The material in this manual is up-to-date at the time of publication, but is subject to change without notice.

Copies of this publication and other Gould publications may be obtained from the Gould sales office or distributor serving your locality.

RELATED PUBLICATIONS

Cross Assemblers:

- o 6800/6802
- o 6801/6803
- o 6809
- o 8080/8085
- o 8048 Family
- o Z80
- o Z8001/Z8002
- o 8086

Terminal

- o Operators Manual Model 950, TeleVideo No. 300002-001, Rev B

Emulators:

- o 9508 MicroSystem Emulator
- o 9516 MicroSystem Integration Station

CP/M and MP/M:

- o The CP/M Handbook with MP/M by Rodney Zaks, Published by Sybex
- o CP/M User Guide by Thom Hogan, Published by Osborne/McGraw-Hill
- o MP/M Multiprogramming Monitor Control Program User's Guide, by Digital Research

PREFACE

If you require any assistance with this product, please call Gould Instruments Division Customer Service on the toll free, hot line numbers listed below.

National (800) 538-9320/9321

California (800) 662-9231

Chapter		Page
1	INTRODUCTION AND OVERVIEW	
	GENERAL DESCRIPTION.	1-1
	SYSTEM HARDWARE CONFIGURATION.	1-1
	Processor Board	1-3
	Disk Drives	1-3
	Power Supply.	1-3
	Display Terminal.	1-3
	SOFTWARE FEATURES.	1-5
	Overview of Software Resources.	1-5
	User Interface.	1-5
	Software Packaging.	1-6
	Operating System Environment.	1-6
	Operating System.	1-6
2	SPECIFICATIONS	
	INTRODUCTION	2-1
	9520 PHYSICAL CHARACTERISTICS.	2-1
	Specifications.	2-1
	9520 Controls and Connectors.	2-3
	9501 PHYSICAL CHARACTERISTICS.	2-6
	Specifications.	2-6
	9501 Controls and Connectors.	2-8
3	INSTALLATION AND CHECKOUT	
	INTRODUCTION	3-1
	UNPACKING AND INSPECTON.	3-1
	AC INPUT REQUIREMENTS.	3-1
	CONNECTING THE 9520 TO THE DISPLAY TERMINAL.	3-2
	INTERFACE SWITCH SETTINGS.	3-8
	Baud Rate Switch.	3-8
	Device Address Switch	3-8
	Disk Diagnostic Switch.	3-8
	POWER ON CHECK	3-10
	Initial StartUp of System	3-10
	Restarting System Operations.	3-19
	System Shutdown Operations.	3-19
	CHANGING AND HANDLING DISKS.	3-20
	Disk Drive Particulars.	3-20
	Inserting Disk in Drive	3-20
	Removing Disk from Drive.	3-21
	Care of Disk.	3-21

CONTENTS

Chapter		Page
4	OPERATION	
	INTRODUCTION.4-1
	TERMINAL KEYBOARD AND KEY FUNCTIONS4-1
	Data Entry Keys.4-1
	Program Control Character Keys4-2
	Text Control Character Keys.4-5
	SYSTEM INITIATION4-6
	System Initiation for MP/M4-6
	System Initiation for CP/M4-7
	ENTERING COMMANDS4-7
	Immediate Command Mode4-7
	Interactive Command Mode4-9
	RUNNING A PROGRAM4-11
	ABORTING AND ATTACHED PROGRAM4-11
	DETACHING FROM A RUNNING PROGRAM.4-11
	ATTACHING TO A DETACHED PROGRAM4-11
5	OPERATING SYSTEM AND UTILITIES	
	INTRODUCTION.5-1
	SOFTWARE ORGANIZATION5-1
	Software Support Functions5-2
	CONSOLE COMMANDS.5-4
	User Identification Commands.5-4
	USER5-5
	CONSOLE.5-6
	DSKRESET5-7
	File Manipulation Commands.5-8
	ERA/ERAQ5-9
	TYPE5-11
	DIR.5-12
	REN.5-13
	STAT5-14
	GENMOD5-15
	GENHEX5-16
	PRLCOM5-17
	System Operation Commands5-18
	PIP.5-19
	ASM.5-20
	SUBMIT5-21
	DUMP5-22
	LOAD5-23
	MPMSTAT.5-24
	SPOOL.5-25
	Program Operation Commands.5-26
	ED5-27
	DDT.5-28
	TOD.5-29
	SCHED.5-30
	ABORT.5-31

Chapter		Page
	SYSTEM UTILITIES.5-32
	FDISK.5-33
	BRATE.5-36
	CONVERT.5-38
	DOWNLOAD5-44
	UPLOAD5-45
	HCOM5-46
6	TEXT EDITOR, DYNAMIC DEBUGGER, AND RELOCATABLE DEBUGGER UTILITIES	
	TEXT EDITOR UTILITY PROGRAM6-1
	WordStar Key Functions6-1
	Invoking WordStar.6-3
	No-File Commands6-4
	Examples of No-File Commands6-7
	Help Levels.6-9
	DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER6-11
	Dynamic Debugger6-11
	Relocatable Debugger6-12
	Commands6-12
	Differences between CP/M and MP/M.6-13
	IMPLEMENTATION NOTES.6-21
7	ASSEMBLER AND LINKER UTILITIES	
	THE CROSS ASSEMBLER7-1
	THE LINKER.7-3
	Overview7-3
	Advice to the First Time Linker User7-7
	LINKER INVOCATION7-8
	File Name Conventions.7-8
	Simple Command Mode.7-9
	Interactive Command Mode7-11
	Indirect Command File Mode7-12
	LINKER COMMANDS7-13
	DEFINE7-14
	END.7-15
	LINK7-16
	LIST7-17
	LOAD7-18
	LOCATE7-19
	LOG.7-21
	MAP.7-22
	NOCONO7-23
	NOLOG.7-24
	NOMAP.7-25
	TRANSFER7-26
	LINKER EXECUTION.7-27
	Program Sections7-27
	The Default Section.7-28
	Memory Allocation of Sections.7-29
	Examples of Memory Allocation Types.7-31

CONTENTS

Chapter		Page
7	LINKER OUTPUT	7-34
	ERROR MESSAGES	7-35
	Command Processing Errors	7-35
	Errors during Linker Execution.	7-36
8	MP/M SYSTEM GENERATION	8-1
	INTRODUCTION	8-1
	GENSYS PROGRAM DESCRIPTION	8-1
	MODIFYING SYSTEM CONFIGURATION.	8-1
	BANK SWITCHED MEMORY CONFIGURATION	8-3
	Memory Segment Table.	8-5
9	SYSTEM DIAGNOSTIC PROGRAM	
	INTRODUCTION	9-1
	DIAGNOSTIC EXECUTION	9-1
	COMMANDS	9-2
10	HARD DISK SYSTEM	
	INTRODUCTION	10-1
	HARDWARE FEATURES.	10-1
	SPECIFICATIONS	10-2
	ORGANIZATION OF THE HARD DISK.	10-7
	INSTALLATION AND PREPARATION FOR USE	10-9
	Installation.	10-9
	Preparation for Use	10-10
	SOFTWARE FEATURES.	10-14
	Explanation of FLOPPYD.	10-15
	SELF TEST OPERATIONS	10-17
	DIAGNOSTICS.	10-24
	Diagnostic Operating System	10-24
	Hard Disk Drive Interface Switch.	10-24
	Diagnostic Tests.	10-26
	Help Command.	10-28
	Valid Diagnostic Commands	10-29
	Operator Intervention	10-33
	Diagnostic Tests and Error Reporting.	10-33
Appendix A	9520 SYSTEM COMMAND SUMMARY	
	ALPHABETICAL LISTING	A-2
	FUNCTIONAL LISTING	A-6
	User Identification Commands.	A-6
	File Manipulation Commands.	A-6
	System Operation Commands	A-7
	Program Operation Commands.	A-7
	System Utilities.	A-8
	MP/M Commands	A-8
	Gould Millennium Support Packages Commands.	A-8

		Page
Appendix B	Hexadecimal and Decimal Conversion	B-1
Appendix C	ASCII Character Set.	C-1
Appendix D	Schematic Diagrams	D-1

Figures

1-1	9520 Software Development System	1-2
1-2	Block Diagram of Processor Board	1-4
2-1	Front Panel of the 9520.	2-2
2-2	Rear Panel of the 9520	2-4
2-3	9501 Display Terminal.	2-7
2-3	9501 Display Terminal, Rear Panel Controls and Connectors.	2-8
2-5	Bit Assignments of the S1 BAUD RATE Switch Bank.	2-9
2-6	Bit Assignments of the S2 FUNCTION Switch Bank	2-10
3-1	Diagram of Development System Terminal RS232 Interface	3-3
3-2	Diagram of 9520 Host-Emulator RS232 Interface.	3-4
3-3	9520 Connected to a Console.	3-5
3-4	9520 Connected to a Printer and a Console.	3-5
3-5	9508 Emulator (Two Console) Connected to the 9520.	3-6
3-6	9508 Emulator (One Console) Connected to the 9520.	3-6
3-7	9516 Emulator Connected to the 9520.	3-7
3-8	Inserting Disk in Drive.	3-22
3-9	Disk Write Protect Tab	3-23
4-1	Typical Keyboard Arrangement	4-3
5-1	Development System Software Organization	5-2
5-2	Cross Support Interface with Remote Station.	5-3
5-3	Host to 9520 Communications Link Interface	5-50
7-1	Linking and Converting Object Files Generated by Cross Assembler	7-4
7-2	Linking Object Files Generated by Previous Link.	7-5
7-3	Linking Object Files Generated by Cross Assembler and Previous Link	7-6
10-1	Front Panel of the 9520 Hard Disk System	10-3
10-2	Rear Panel of the Hard Disk System	10-4

Tables

2-1	Host Interface Connector Configuration	2-11
2-2	Printer Connector Configuration.	2-11
3-1	Packaged Items for Standard 9520 System Configuration.	3-2
5-1	Conversions.	5-38
6-1	No-File Commands	6-5
10-1	Drive Interface Board Switch Settings.	10-25
10-2	Processor Board SW1 Settings	10-25

INTRODUCTION AND OVERVIEW

GENERAL DESCRIPTION

Gould Inc., Instruments Division 9520 Software Development System (figure 1-1) is a general purpose, low-cost, user-oriented microcomputer system designed to meet the needs of the programmer in a program development environment. The system provides all of the capabilities for developing, testing and debugging user programs. The user can furnish his own display terminal, or a display terminal console can be supplied with the system (the Gould 9501).

The 9520 is available in two disk configurations:

1. Two integral eight inch, single sided, double density floppy disk drives
2. A thin-line double sided, double density, floppy disk drive and a 10 megabyte hard disk

Two operating systems may be used with the 9520, MP/M™ or CP/M™. MP/M is a priority-driven, disk operating system that provides the user with a multi-tasking software environment. MP/M runs only on the dual floppy hardware configuration. CP/M, a single user operating system, can be used with either hardware configuration.

Both configurations are Z80A-based systems with 64K memory standard, and an additional 48K/256K of memory as an available option. Interface connections are three RS-232-C ports, one RS-422 port and an IEEE-488 port. These interfaces allow the 9520 to connect with a display terminal console, a printer, an emulator, as well as any other compatible peripherals.

Most information concerning the 9520 is true for both the dual floppy and hard disk systems. Information specific to the 9520 Hard Disk system is found in Chapter 10.

The 9520 is supplied with hardware and software features that allow easy interfacing with other members of the Gould 9500 family, e.g., the 9508 MicroSystem Emulator and 9516 MicroSystem Integration Station.

SYSTEM HARDWARE CONFIGURATION

The 9520 hardware consists of five basic functional units located on a single printer circuit board, two floppy disk drives, and a power supply.

™ Trademarks of Digital Research, Pacific Grove, CA 93950



Figure 1-1. 9520 Software Development System

INTRODUCTION AND OVERVIEW

Processor Board

The single printed circuit processor board in the software development system contains all the circuits necessary to perform the data manipulations required to develop software. Its five basic function units are: the Z80A central processing unit, a 64K byte dynamic RAM array, direct memory access (DMA) and floppy disk controllers, two dual asynchronous receiver/transmitters and a IEEE-488 port I/O controller. Each is represented in figure 1-2.

Disk Drives

The dual floppy disk drives for the 9520 Software Development System allow data to be formatted in single or double density on standard 8-inch floppy disks. The reading or writing of single or double density is selected by the user at format time. Single density programs such as those supplied by the CP/M user's group or other sources can be converted to double density by a simple copy utility, Peripheral Interchange Program (PIP).

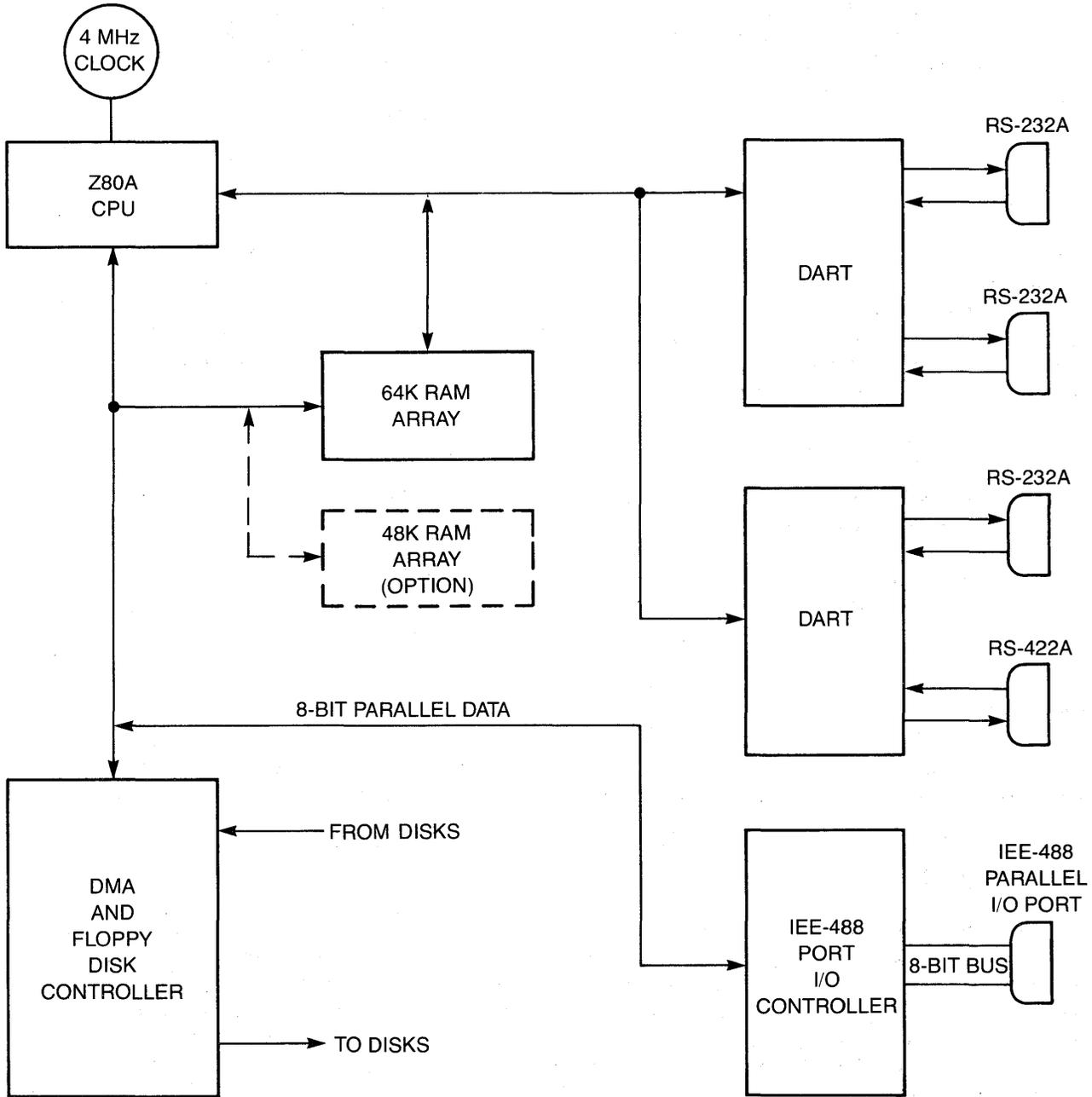
Drive Type	Disks	Recording Format
8" Single Side	SSDD	MFM
Double Density	SSSD	FM
8" Double Side	DDSD	MFM
Double Density	SSDD	MFM
	SSSD	FM

Power Supply

The power supply furnishes the +5 VDC and +12 VDC required by the processor board plus the +5 VDC and +24 VDC used by the disk drives.

Display Terminal

The display terminal may be a Gould 9501 Display Terminal Console supplied with the 9520 Software Development System or may be furnished by the user. The only prerequisite for a user supplied display terminal is that it must be able to display 80 character lines and be compatible with the RS-232C interface. The Gould supplied 9501 terminal provides enhanced operation through the use of preprogrammed function keys and screen features (e.g. half intensity, and xy addressing).



9520-014

Figure 1-2. Block Diagram of Processor Board

SOFTWARE FEATURES

This section describes the 9520 software features and provides an overview of the disk operating facilities.

Overview of Software Resources

The two operating systems, MP/M and CP/M, provide the following capabilities:

- o Creating and editing user source program files
- o Storing and managing of files in source and object format
- o Assembling and linking of object files for conversion into executable load modules
- o Checkout and dynamic debugging of source programs originating in the MP/M and CP/M environment
- o Uploading and downloading to transfer data between the 9520 Development System and a remote hardware/software in-circuit-emulator station
- o Uploading and downloading to transfer data between the 9520 and another computer system.

User Interface

The user communicates with the software development system by entering input data and commands at the display terminal keyboard. The commands are entered as a single-line character string. Users of the Gould 9501 may also enter commands by pressing function keys. The system responds by displaying data and messages at the terminal which prompt the user for additional input, as necessary, to enable processing to be completed.

The user programs originated at the 9520 Software Development System can be downloaded to a remote in-circuit emulator station (e.g. Gould's 9508 or 9516). The users program can then be executed to locate and correct programming errors and/or to detect and correct malfunctions in the microprocessor system circuits that are detected when undergoing hardware/software integration tests.

INTRODUCTION AND OVERVIEW

Software Packaging

The system programs reside on three standard floppy disks. Two of the disks contain the operating system program and one or more additional sets of programs. The remaining disk contains the system diagnostics.

- Disk #1 (SYSTEM).. Operating System and Utilities
- #2 (LANGUAGE TRANSLATOR).. Cross-Assembler, Linker
- #3 (DIAGNOSTIC).. System Diagnostic Monitor and Test Routines

The WordStar™ Text Editor Utility is a standard software feature provided with the operating system disk.

Separate disks are available for optional high-level programming languages (such as a PASCAL Compiler). The compilers are used to generate object code output for 8080, 8085 and Z80 microprocessor-based systems.

Operating System Environment

The development system software consists of the following sets of programs:

- o Operating System
- o System Utilities
- o Language Translator
- o System Diagnostics

Operating System. The operating system is based on a memory resident Executive that provides a variety of services to the system and user programs. The Executive provides all of the interaction between the hardware and software, and between the system and the user. All queries to the input/output devices are processed through the Executive. The Executive includes interrupt handling, I/O processing, supervisor call processing, user communications and disk file management routines.

The MP/M operating system is easily reconfigured with optional user memory segment sizes, resident process modules, relocatable modules, and number of terminals. The description of the system generation procedure for MP/M is given in chapter 8. The CP/M operating system requires no system generation.

System Utilities. The utility programs provide various file management functions such as editing, renaming, deleting, copying and displaying existing files. The utilities also include programs that allow the user to access and change certain system parameters and submit a series of commands from a file for batch processing. There is a format utility which allows the user to format, duplicate or copy disks. The communications utility allows the user to transfer data between the 9520 and another device.

™ Trademark of MicroPro International Corporation, San Rafael, CA 94901

Language Translator. The language translator utility is a relocatable macro-assembler which translates the target microprocessor assembly language programs into object code which is executable in the users remote microprocessor system. The following microprocessors are supported via the cross-assemblers:

- o Z80A
- o 8048/49/41/21/22 (Absolute)
- o 8080/85
- o 6800/01/02/03/09
- o Z8001/2
- o 9900 Family
- o 8086

Each cross-assembler program is described in a separate publication supplement listed in the preface. Each cross-assembler translator utility accepts an assembly language program written in the source code of the supported microprocessor.

The disk containing the cross-assembler utility also includes the linker and formatter/downloader software. The linker utility links relocatable object files into an executable load module by resolving all external references to the file while building the load module.

System Diagnostics. The 9520 System Diagnostic program provides the user with a comprehensive set of programs to diagnose and test system components. The diagnostic monitor and test routines are described in chapter 9.

INTRODUCTION

This chapter describes the physical characteristics of the 9520 and its terminal, the 9501. Also included are diagrams and descriptions of the controls and connectors of the two instruments.

9520 PHYSICAL CHARACTERISTICS**Specifications**

Dimensions

Height - 13 5/16 inches (33.3325 cm)

Width - 19 3/16 inches (48.4475 cm)

Depth - 24 1/8 inches (61.085 cm)

Weight - 74 lbs (33.6 kg)

Environmental Limits	Operating	Storage
Ambient Temperatures =	40° to 104°F (4.4° to 40°C)	-8°F to 117°F (-22.2° to 47.2°C)
Relative Humidity =	20% to 80%	1% to 95%
Maximum Wet Bulb =	78°F (25°C)	No Condensation

AC Power Requirements

50/60 Hz \pm 0.5 Hz

100/115 \pm 10% VAC

200/230 \pm 10% VAC

Fuses for Rated Voltage

Voltage	Power Supply and Fan (F1)	Disk Drives (F2)
110/115 VAC	3AG, 5 Amp	3AG, 1.0 Amp
200/230	3AG, 2.5 Amp	3AG, 0.75 Amp

SPECIFICATIONS

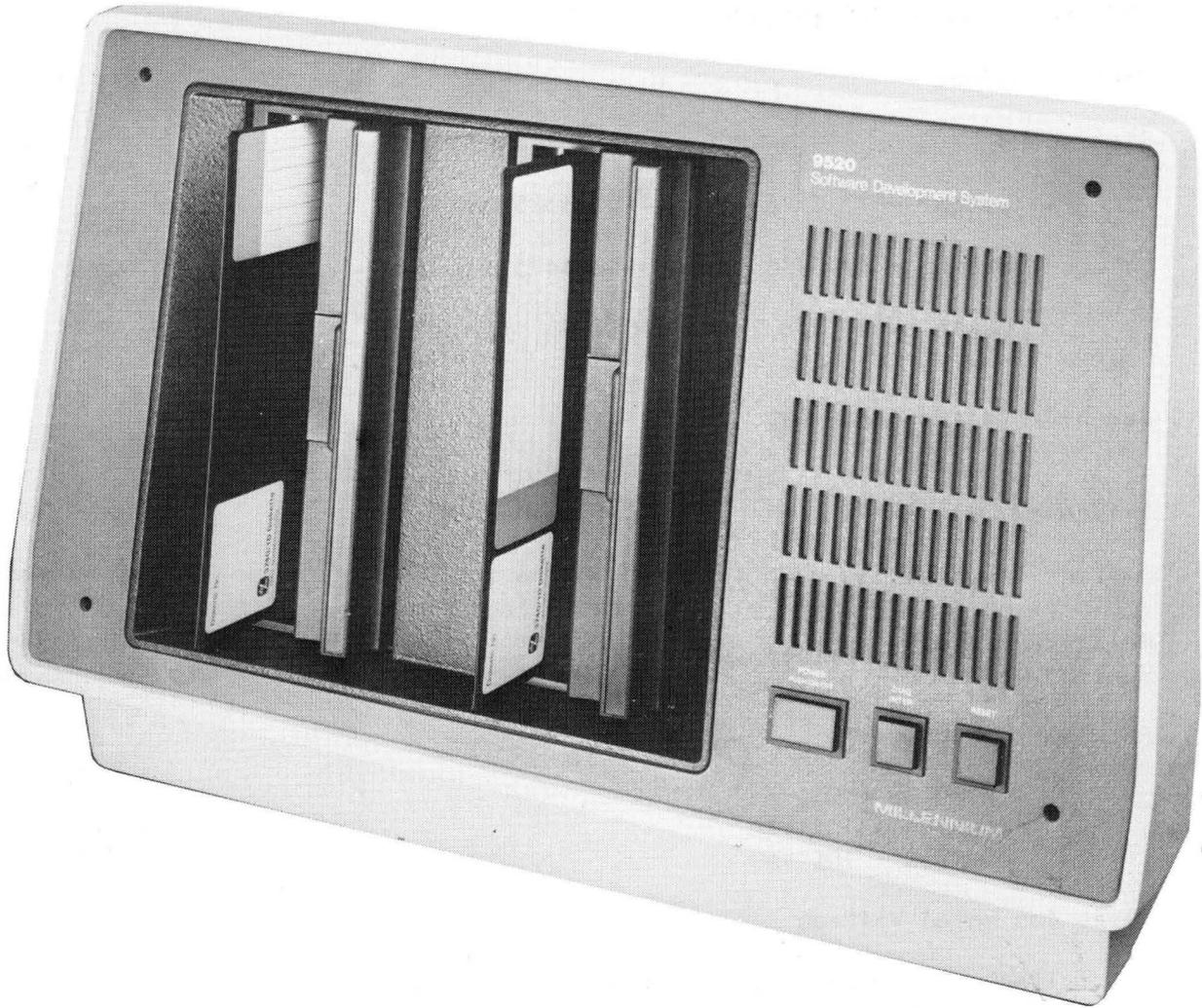


Figure 2-1. Front Panel of 9520

9520 Controls and Connectors

The following controls are located on the front panel of the 9520. (Refer to figure 2-1.)

**POWER
PUSH ON/OFF** The power on/off switch is illuminated (red) when AC power is applied to the Software Development System. When the push-switch is illuminated, this indicates not only the presence of AC voltage but also the presence of +5 VDC.

RESET When activated, the RESET push switch reinitializes the internal microprocessor and forces the system to jump to the on-board self-test program.

The following are the rear panel ports and switches. (See figure 2-2.)

J10-CONSOLE Standard RS-232C, D-type, 25-pin (DCE configuration) female connector. Connector J10 is generally connected to the Display Terminal.

J11-RS-232-1 Standard RS-232C, D-type, 25-pin (DCE or DTE configuration option) female connector. Connector J11 is generally connected to the software/hardware debug system.

J12-RS-232-2 Standard RS-232C, D-type, 25-pin (DCE configuration) female connector. Connector J12 is generally connected to the printer.

J13-RS-422 Standard RS-422A, D-type, 37-pin female connector. Connector J13 is generally connected to a unit with serial, high-speed input/output transfer requirements. (Corresponds to RS-449 configuration.)

J14-IEEE 488 This connector is a standard, 24-pin, female, high-speed parallel transfer I/O port. The port provides eight-data/address lines, eight control lines and eight signal and frame ground lines.

J15 POWER ON Standard AC, male, connector for power input.

SPECIFICATIONS

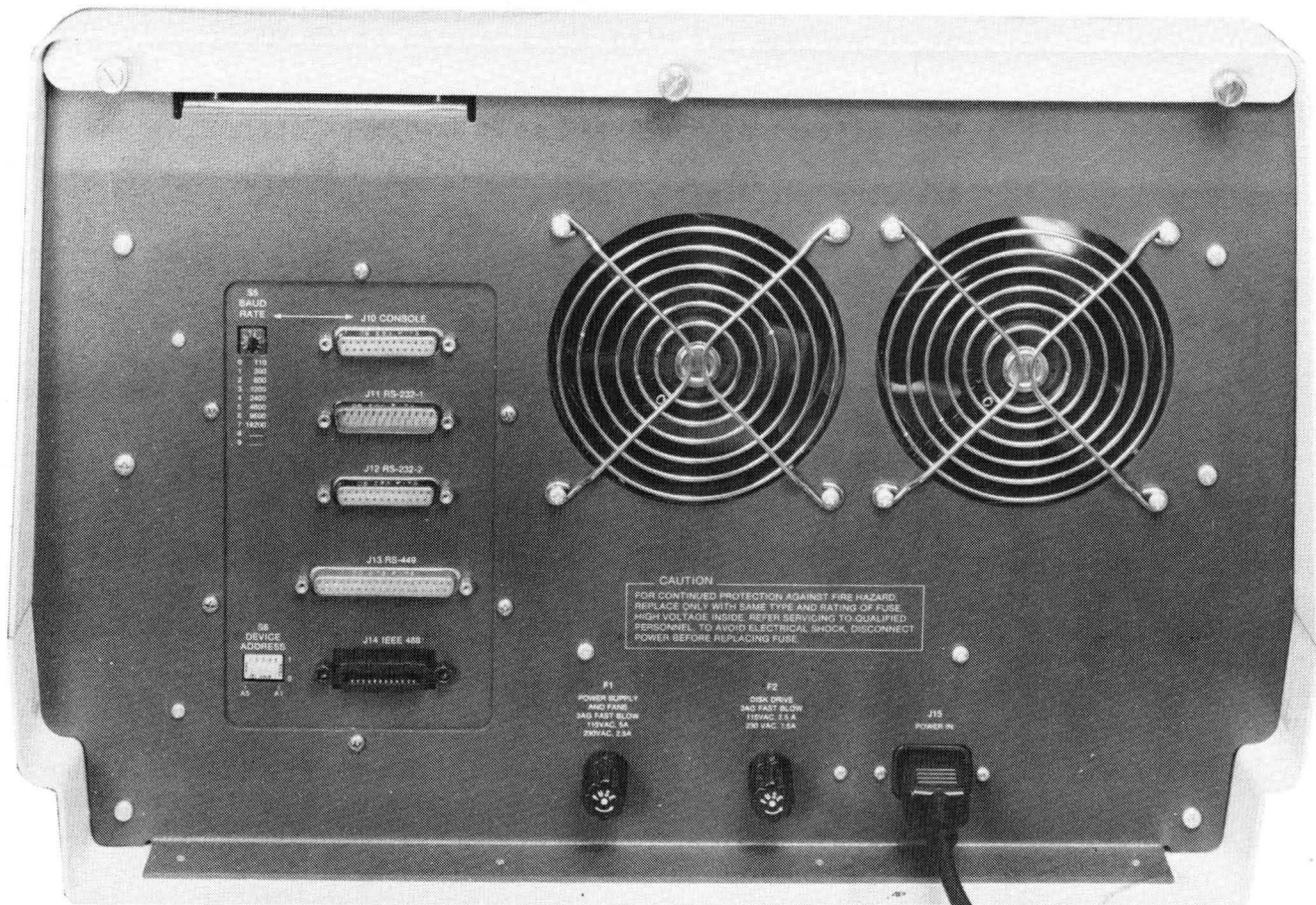


Figure 2-2. Rear Panel of 9520

S5 BAUD RATE S5 is an eight-position rotary switch used to select the desired baud rate. The switch position is read by the system during initialization and can be changed each time the system is reset. A baud rate different from the switch position may be selected by using the BRATE utility. A function key may be programmed to execute the BRATE utility program.

Positions:

0	110
1	300
2	600
3	1200
4	2400
5	4800
6	9600
7	19200
8	----
9	----

(default setting when shipped with a 9501 Terminal)

S6 DEVICE ADDRESS The DEVICE ADDRESS switch is a five-position DIP switch that controls addressing of the peripherals connected to the IEEE-488 (GPIB interface) port.

Cooling. System cooling is provided by two AC operated fans, mounted at the rear of the enclosure, that draw cool air through the disk drives, over the power supply and across the printed circuit board. The enclosure must be covered before AC power is applied. Operation without the top cover will defeat the cooling function of the fan and may cause damage to the power supply and/or the printed circuit cards.

Power Supply Specifications. The 9520 power supply provides the four DC operating voltages necessary for the operation of the processor board and disk drives. The specifications for the power supply are as follows:

AC Input

90 to 130 volts or 180 to 260 volts AC
 47 to 63 Hz, single-phase
 (Power Supply input is adjusted at Gould in accordance with user requirements.)

DC Outputs

<u>Voltage</u>	<u>Current</u>
+5 volts	20A max.
+12 volts	5A max.
-12 volts	3A max.
+24 volts	3.5A max.

Maximum Output Power 150 Watts

Regulation Characteristics

Line: $\pm 0.1\%$ for a full line change of 90-130 VAC or 180-260 VAC.
 Load: $\pm 0.1\%$ for a 100% load change.

SPECIFICATIONS

9501 PHYSICAL CHARACTERISTICS

The 9501 Display Terminal, figure 2-3, consists of a microprocessor-based intelligent CRT console and a separate ASCII keyboard unit that provides 96 alphanumeric and 32 control characters. The keyboard communicates with the CRT console via a cable connected to the connector panel at the rear of the console. (See figure 2-4.) The specifications and other connectors and controls are outlined in the following pages.

Specifications

Display Cabinet Dimensions

Height - 14.0 inches (35.6 Cm)
Width - 16 1/2 inches (41.9 Cm)
Depth - 14 1/4 inches (36.2 Cm)

Keyboard Dimensions

Height - 3.0 inches (7.6 Cm)
Width - 16 1/2 inches (41.9 Cm)
Depth - 7 1/2 inches (19.0 Cm)

Cabinet Weight - 30 lbs (13.6 kilograms)
Keyboard Weight - 4.5 lbs (2.3 kilograms)

Environmental Limits	Operating	Storage
Ambient Temperature	32°F to 122°F (0°C to 50°C)	-40°F to 149°F (-40°C to 65°C)
Relative Humidity	10% to 95% (Non Condensing)	N/A (No Restriction)

AC Power Requirements

50/60 Hz \pm 0.5 Hz
100/115 \pm 10% VAC
200/230 \pm 10% VAC

Fuses for Rated Voltage

100/115 VAC	3AG, 1 Amp
200/230 VAC	3AG, 0.5 Amp SLO-BLO

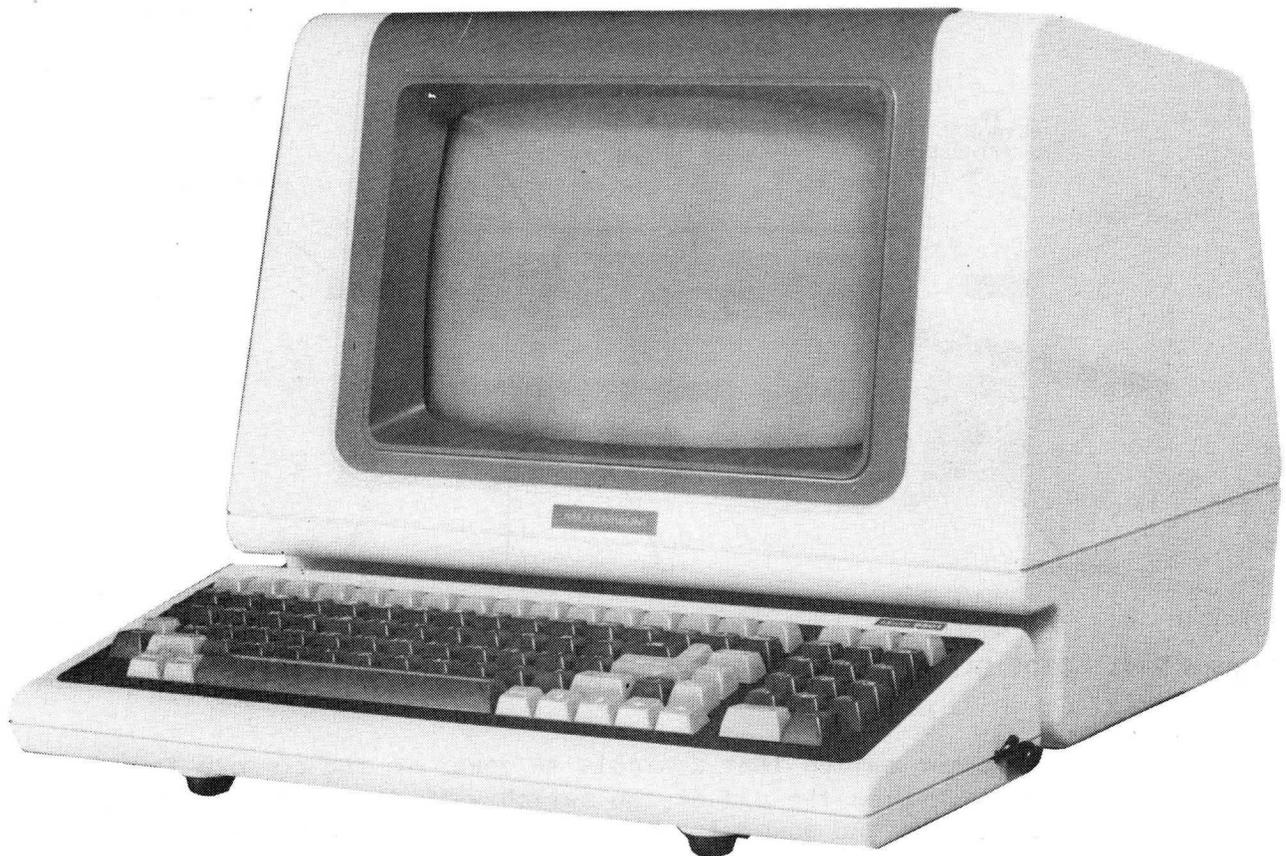


Figure 2-3. 9501 Display Terminal

9501 Controls and Connectors

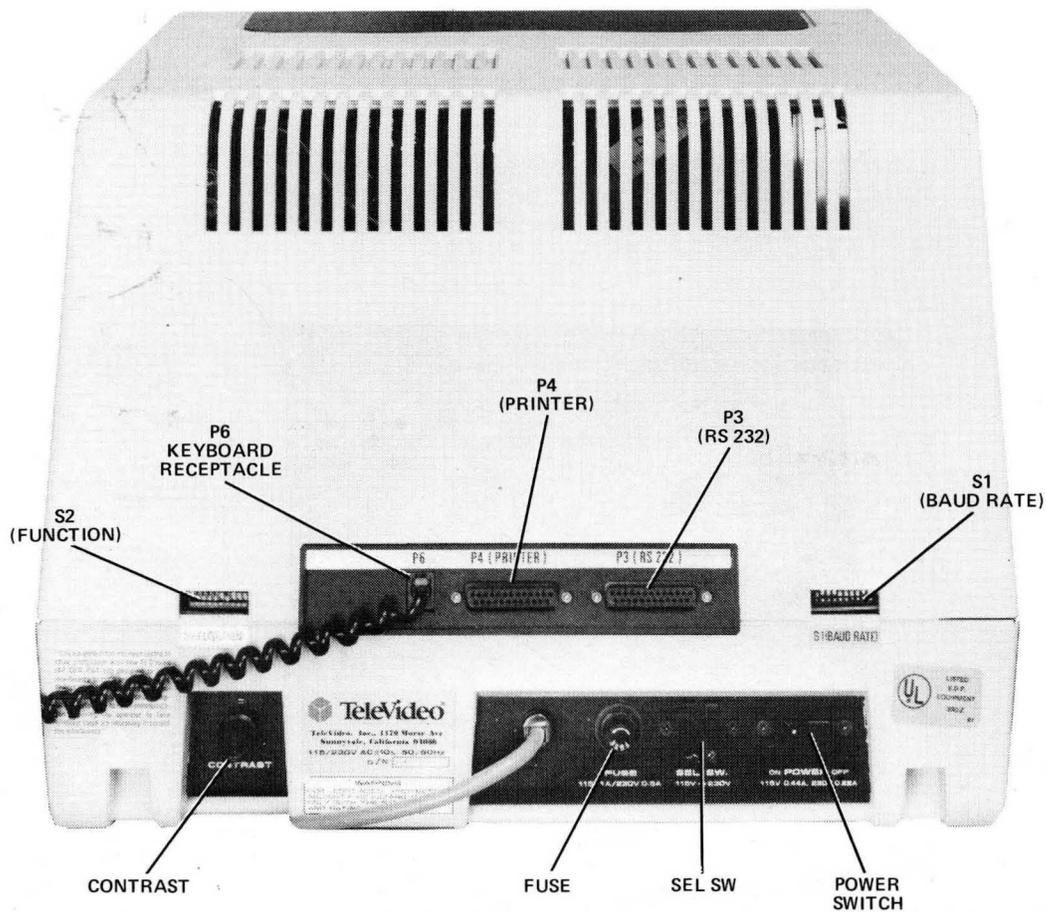
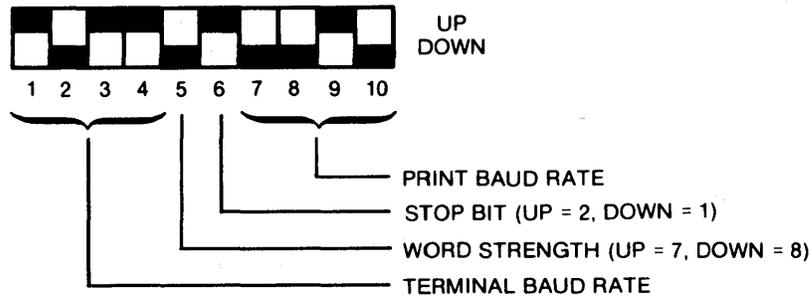


Figure 2-4. 9501 Display Terminal, Rear Panel Controls and Connectors

- POWER Switch A rocker switch that controls AC power to the console terminal. When the end of the switch with the white dot is depressed, AC power is applied to the unit; depressing the unmarked end of the switch removes power. One second after the power is applied, an internal beeper will sound, indicating the presence of AC power.
- SEL SW
115/230 VAC The position of this slide switch must coincide with the primary AC source voltage. The left position of the switch is for operation from a 115-VAC source; the right position is for a 230-VAC source. The switch blocking strip prevents inadvertent changing of the switch position.
- CONTRAST Switch A rotary potentiometer control that determines the white-to-black intensity of the visible display.

S1 BAUD RATE
Switch Bank

A microswitch bank with 10 switches that determine the input baud rate to the console terminal, the output baud rate from the display terminal, number of stop bits and word length. The following illustration (figure 2-5) shows the bit allocations and the baud rate selection bit combinations.



SWITCHES				BAUD RATE
7	8	9	10	Printer
1	2	3	4	Terminal
0	0	0	0	9600
1	0	0	0	50
0	1	0	0	75
1	1	0	0	110
0	0	1	0	135
1	0	1	0	150
0	1	1	0	300
1	1	1	0	600
0	0	0	1	1200
1	0	0	1	1800
0	1	0	1	2400
1	1	0	1	3600
0	0	1	1	4800
1	0	1	1	7200
0	1	1	1	9600
1	1	1	1	19200

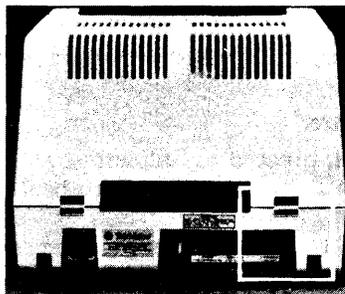


Figure 2-5. Bit Assignments of the S1 BAUD RATE Switch Bank.

S2 FUNCTION
Switch Bank

A microswitch bank with 10 switches that select operational functions of the display terminal. The following illustration (figure 2-6) shows the bit allocations, and bit combinations for transmission mode selection and parity selection.

SPECIFICATIONS

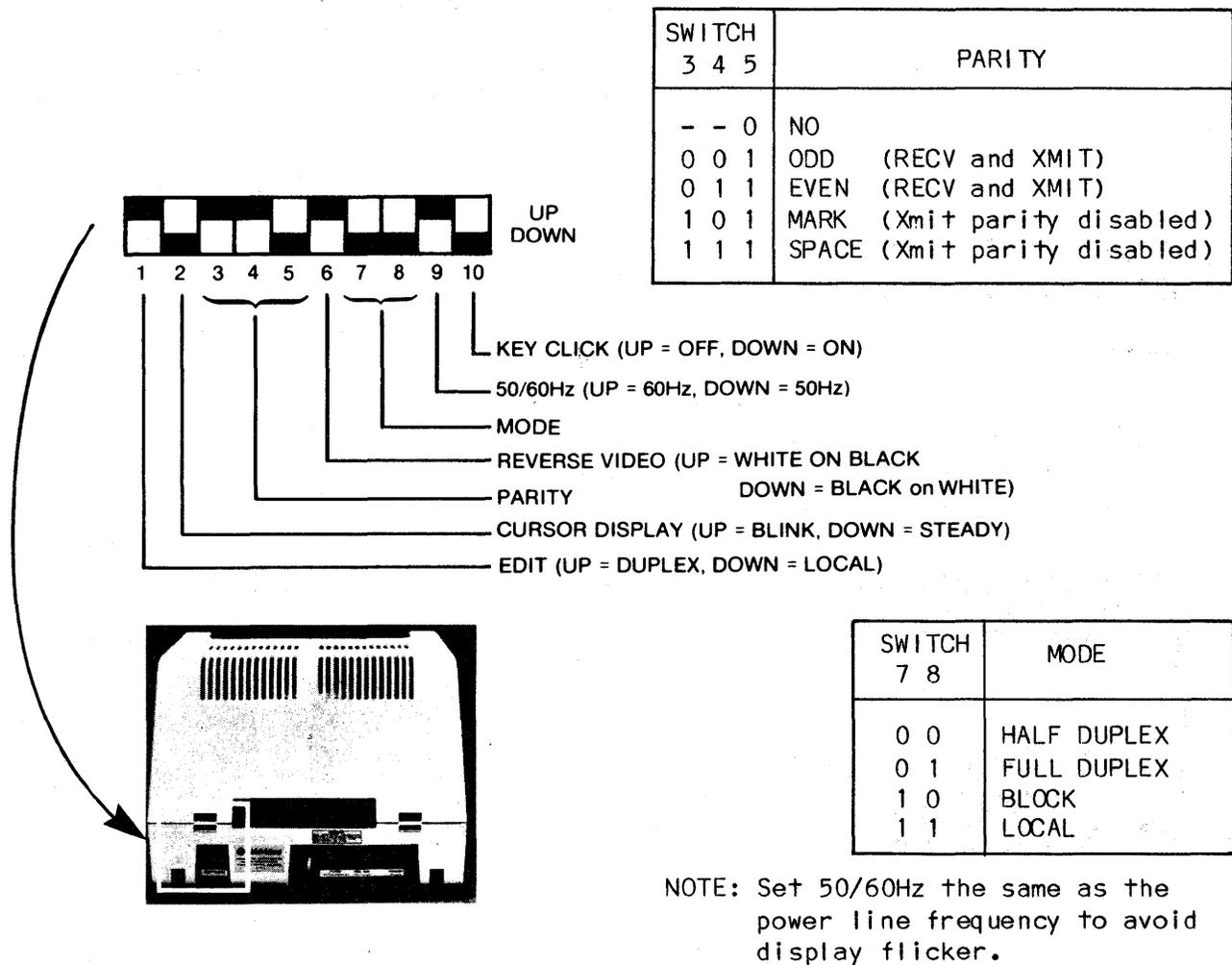


Figure 2-6. Bit Assignments of the S2 FUNCTION Switch Bank.

FUSE 115 VIA/
230 0.5 A

The FUSE holder contains the fuse that protects the internal circuits.

```

* * * * *
*           CAUTION           *
* * * * *
* The intent of the fuse should never *
* be ignored, nor should any attempt be *
* made to defeat the intention of this *
* protective device, as damage to the *
* equipment may occur.         *
* * * * *
    
```

P3 (RS-232) This is a 25-pin, female, RS-232C connector port that is generally connected to the host source. The internal configuration of the connector is shown in table 2-1 and also described in the manufacturer's manual.

Table 2-1. Host Interface Connector Configuration

PIN No.	SIGNAL NAME
1	Frame Ground
2	Transmit Data Output
3	Receive Data Input
4	Request To Send Output
5	Clear To Send Input
6	Data Set Ready Input (opt.)
7	Signal Ground
8	Carrier Detect Input
20	Data Terminal Ready Output
9	20 mA source (+12V, no load)
14	20 mA source (+12V, no load)
10	Detected current loop data
25	Current Loop +, Transmit*
13	Current Loop -, Transmit*
12	Current Loop +, Receive*
24	Current Loop -, Receive*

P4 (PRINTER) This is a 25-pin, female, RS-232C connector port that may be connected to an optional printer. The internal configuration of the connector is shown in table 2-2.

Table 2-2. Printer Connector Configuration

PIN No.	SIGNAL NAME
1	Protect Ground
2	Transmit Data
3	Receive Data
4	Request To Send
5	Clear To Send
6	Data Set Ready
7	Signal Ground
8	Data Carrier Connect
20	Data Terminal Ready

SPECIFICATIONS

Power Cord

The power cord applies the AC power to the 9501 Display Terminal. One end of the power cord is permanently affixed to the 9501. The free end is a standard 3-prong connector. If the equipment is used for international applications, remove the U.S.-style connector from the power cord and install a connector that matches the local power receptacle. Power cord wires are color-coded as follows:

green = earth ground,
black = primary power (hot),
white = primary power return (neutral).

P6

The P6 receptacle accepts the expansion cord from the keyboard. The receptacle is configured as a snap-type modular receptacle.

INSTALLATION AND CHECKOUT

INTRODUCTION

This chapter describes the procedure for installing the 9520 Software Development System components at the user's site. The user may install his own display terminal or the optional 9501 Display Terminal that is available for use with the equipment. Installation involves inspecting and setting up the components, connecting the Display Terminal Console to the 9520 Software Development System, and conducting the system power on check to verify the equipment is ready for operation.

UNPACKING AND INSPECTION

All of the hardware and software items required to install and operate the equipment at the users site are shipped in packaged units. External cables are included for connecting the display terminal to the 9520 Software Development System. All items shipped for the standard system configuration are described in table 3-1.

After unpacking the equipment, inspect for scratches, dents or other damage that might have occurred during shipping. Refer to the shipping papers to verify that all the components are present.

If the equipment is damaged, do not operate the equipment. File a claim with the shipping firm immediately, and notify Gould Inc., Instruments Division Customer Service department at once. Gould will arrange for repair or replacement of the equipment without waiting for settlement of the claim against the carrier.

If the equipment must be returned to Gould, attach a tag showing the owner, address, serial number, and a description of the failure. The original shipping carton and packing material should be reused with the RMA (Returned Material Authorization) number prominently displayed. An RMA number must be obtained by calling Customer Service on the toll-free, hot-line numbers listed in the preface.

Gould Technical Support Representatives and Customer Engineers are available to provide consultation and assistance on request.

AC INPUT REQUIREMENTS

The 9520 Software Development System is wired by the manufacturer to accept AC power input for 100V/115V or 200V/230V at 50 or 60 Hz. Before installing the equipment, check the power specification label on the back panel to ensure AC input requirements for the equipment coincide with the facility supply level. If a discrepancy is noted, do not attempt to make adjustments. Contact the Gould Customer Service Representative.

INSTALLATION AND CHECKOUT

Table 3-1. Package Items for Standard 9520 System Configuration

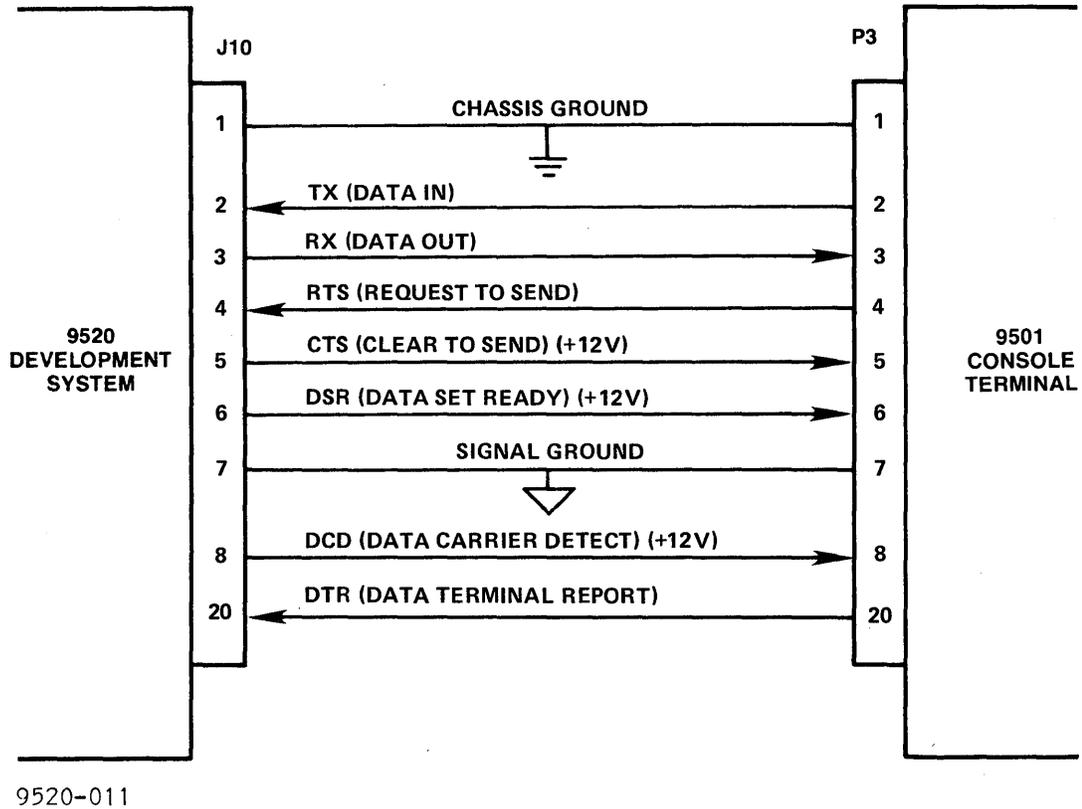
QTY	ITEM DESCRIPTION
1	9520 Software Development System Unit
1	AC Power Cord
1	RS-232 Interface Signal Cable
1 (optional)	9501 Display Terminal Keyboard Unit (Televideo Model #950)
3	Standard Disks containing the following software: a) Operating System Software b) Language Translator Software c) System Diagnostic Software
1	Document Package containing the following manuals: a) MP/M User's Guide b) CP/M Handbook c) 9520 Development System Users Manual d) 9501 Display Terminal Operators Manual (provided with option)

CONNECTING THE 9520 TO THE DISPLAY TERMINAL

When the 9520 is unpacked, the system must then be connected to a display terminal and to a remote emulator if desired. If the 9520 is operated together with the Gould 9501 Display Terminal and either a 9508 or 9516 emulator, the physical interconnections, program interfaces, and operating protocol are all fully compatible. All units can be installed, checked out, and operated as shipped from the factory.

If another terminal is to be used, the console terminal can be any 80 character-per-line CRT/Keyboard or ASR-33 type send/receive terminal. The console terminal is connected to the J10 TERMINAL on the back of the development system either by the standard 10 foot cable or a user-fabricated RS-232 cable. The development system console terminal interface is RS-232C, using a standard 25-pin D-type connector, as illustrated in figure 3-1. A minimum of three wires must be used to implement the interface, with the other five maintained in the states shown in the figure. All remaining lines of the 25-pin connector can be left open or floating.

The following illustrations show the interface connections necessary to operate the 9520 in varied configurations.



Minimum 3-wire Interface

PIN NO.

(J10 on 9520)

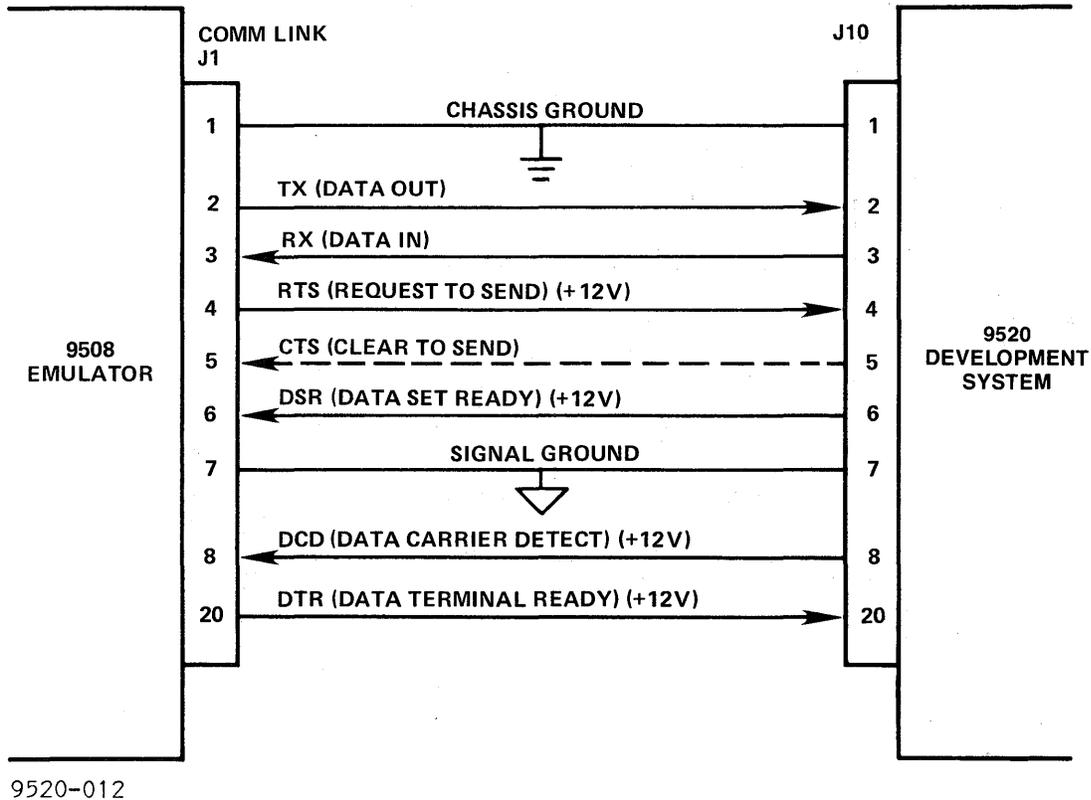
0 FUNCTION

- 2 Transmitted data (TX) - data from console terminal to the development system
- 3 Received Data (RX) - data from development system to console terminal.
- 7 Logic ground

If connected, pins 4 and 20 must be maintained at +12 V or floating; pins 5, 6, and 8 must be allowed to remain at +12 V; connection to pin 1 is optional; all remaining pins on the 25-pin connector are not connected in the emulator.

Figure 3-1. Diagram of Development System Console Terminal RS-232 Interface

INSTALLATION AND CHECKOUT



Minimum 3-wire Interface

PIN NO.

(J2 on 9508)

FUNCTION

- 2 Transmitted data (TX) - data from console terminal to the development system
- 3 Received Data (RX) - data from development system to console terminal.
- 7 Logic ground

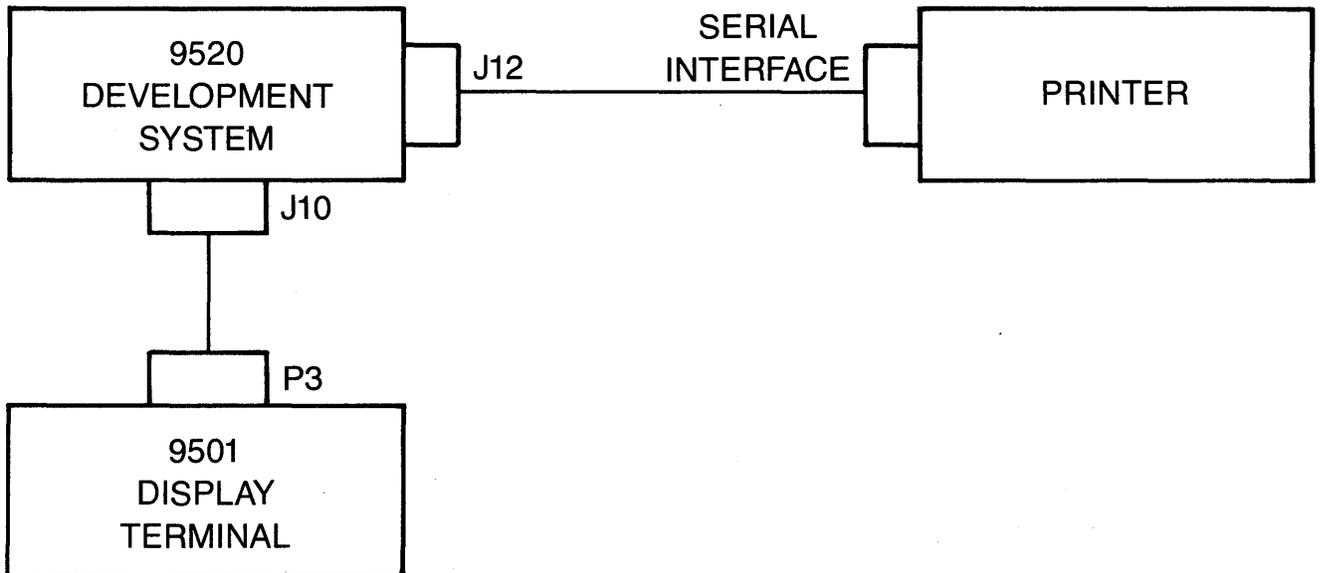
If connected, pins 6 and 8 must be maintained at +12 V or floating; pins 4 and 20 must be allowed to remain at +12 V; connection to pin 1 is optional; all remaining pins on the 25-pin connector are not connected in the emulator.

Figure 3-2. Diagram of 9520 Host-Emulator RS-232 Interface



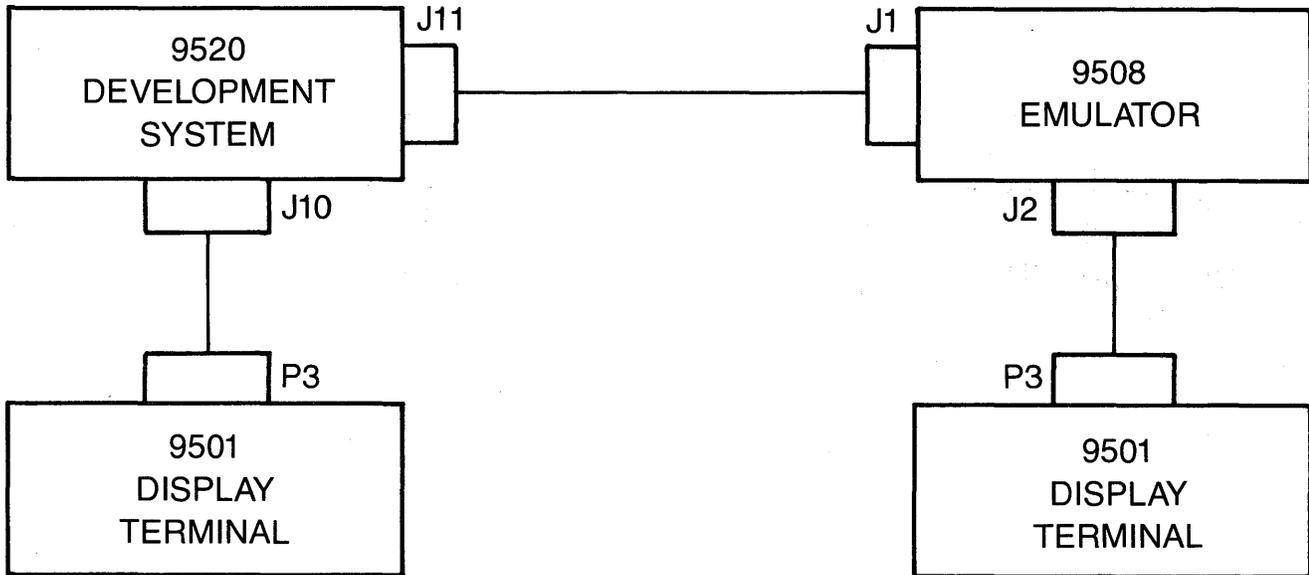
9520-001

Figure 3-3. 9520 Connected to a Console



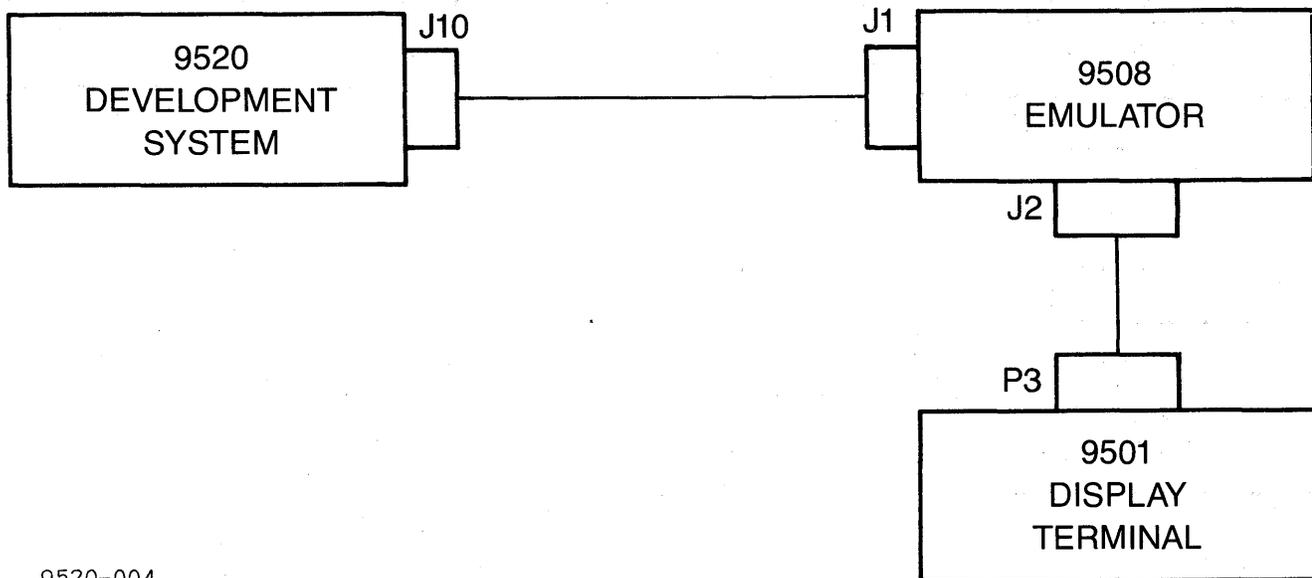
9520-002

Figure 3-4. 9520 Connected to a Printer (with a serial interface) and a Console



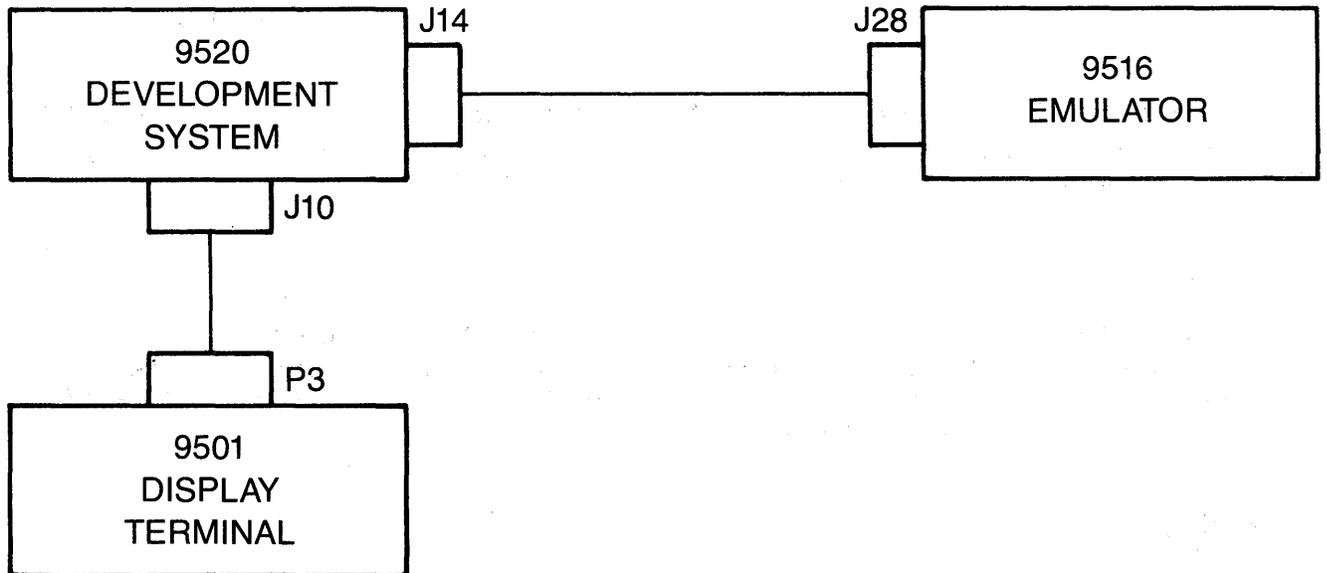
9520-003

Figure 3-5. 9508 Emulator (Two console) Connected to the 9520 (MP/M only)



9520-004

Figure 3-6. 9508 Emulator (one console) connected to the 9520



9520-005

Figure 3-7. 9516 Emulator Connected to the 9520

INSTALLATION AND CHECKOUT

INTERFACE SWITCH SETTINGS

After the interface connections have been made, the interface switches must be set. The development system employs three hardware interface switches, SW1, S5 and S6, that adapt the system to the user's operating requirements. Two of the switches, S5 (Baud Rate) and S6 (Device Address), are located at the rear of the development system chassis. The Disk Diagnostic Switch (SW-1) is located on the processor printed circuit board inside the Development System chassis.

The functions of the switches are described below, and the settings are listed on page 3-9.

Baud Rate Switch

The Baud Rate Switch (S5) is an eight-position, rotary switch. Each position of the switch selects a specified baud rate within the range of 110 through 19,200 as indicated by the label and associated calibration marks for each switch position. The switch position selected is the default baud rate used at initialization.

All other baud rates can be selected after the system is initialized. The BRATE utility program, described in chapter 5, permits a user to examine and modify the baud rate or status line usage for the assigned port.

Device Address Switch

The Device Address switch (S6) is a five-position DIP switch used to assign the hexadecimal I/O device address for the IEEE-488 parallel port at connector J14. The user can position the bit switches to select one out of a possible 32 combinations for the address assignment.

Disk Diagnostic Switch

The Disk Diagnostic Switch (SW1) is a four-position DIP switch used to select the type of display terminal to be interfaced with the system and to enable the Disk Alignment Diagnostic program.

NOTE: The positioning of SW1 to enable the Disk Alignment Diagnostic is not required for installation.

The switch positions SW1-1 through SW1-4 must be set as follows to interface the display terminal to the 9520:

Switch SW1-1

- OFF - Selects booting options
- ON - Selects disk alignment and disk checks

When SW1-1 is OFF:

SW1-2 has no effect

SW1-3 selects whether system identification bytes will be checked during booting

SW1-4 selects the terminal interface during booting

<u>SW1-1</u>	<u>SW1-2</u>	<u>SW1-3</u>	<u>SW1-4</u>	<u>FUNCTION</u>
OFF	OFF	X	X	PRESENTLY NOT USED
OFF	ON	X	X	PRESENTLY NOT USED
OFF	X	OFF	X	BOOT, CHECK SYSTEM IDENTIFICATION BYTES
OFF	X	ON	X	BOOT, DO NOT CHECK SYSTEM IDENTIFICATION BYTES
OFF	X	X	OFF	BOOT, TELEVIDEO 9501 TERMINAL
OFF	X	X	ON	BOOT, ANY OTHER TERMINAL TYPE
ON	X	X	X	ENABLE DISK ALIGNMENT AID PROGRAM
ON	OFF	OFF	OFF	SEEK TRACK 00
ON	OFF	OFF	ON	ALTERNATELY SEEK TRACKS 00 AND 01
ON	OFF	ON	OFF	READ TRACK 1
ON	OFF	ON	ON	READ TRACK 37
ON	ON	OFF	OFF	READ TRACK 38
ON	ON	OFF	ON	READ TRACK 39
ON	ON	ON	OFF	READ TRACK 76
ON	ON	ON	ON	WRITE TRACK 76
OFF	OFF	ON	ON	9520 HARD DISK
				9501 TERMINAL
OFF	OFF	ON	OFF	9520 HARD DISK
				NOT A 9501 TERMINAL

INSTALLATION AND CHECKOUT

POWER ON CHECK

The power on check is performed by the system to verify the operational readiness of hardware components whenever the system is powered up or restarted. The power on check is implemented by a self-test, bootstrap program permanently stored in the EPROM firmware on the printed circuit board.

The 9520 Boot PROM diagnostic is automatically initiated when the POWER switch is turned ON to power up the system from a cold start, or when the RESET switch is pressed. The Self Test diagnostic performs an operational check of on-board circuit functions. The status of the test is presented to the operator by means of eight LED indicators located on the printed circuit board and by a message displayed on the terminal screen.

```
* * * * *
*           CAUTION           *
*           *                 *
*   The disk should not be engaged *
* with the disk drive during the initial *
* power-up sequence as damage to the disk *
* may result.                   *
*                               *
* * * * *
```

Initial Start Up of System

If the 9520 Software Development System in use has a serial number of 339 or below, use the following procedure to power up the system from a cold start:

1. Turn on the POWER switch at the display terminal. The physical location of this switch will vary with different terminals, and may be located at the front, rear or side of the terminal chassis.

The terminal response when power is applied will also vary with different terminals. A typical response is as follows:

- a. The terminal bell will sound within 1 second to indicate power is on.
 - b. After 10 to 15 seconds, the cursor will appear in the upper left-hand corner of the screen.
2. Verify the disk is not mounted on the disk drives and press the POWER switch at the 9520 development system control panel.
 3. Observe the terminal display which presents the following message line to indicate the Boot PROM diagnostic is running.

9520 SELF TEST VERS 2.0 C0

The diagnostic routine runs for approximately three seconds. A total of eight circuit functions are checked by the program. The displayed message line spells out the word C-O-M-P-L-E-T-E as each circuit passes its test. In addition, there is a set of LEDs located on the top edge of the printed circuit board. All of the LEDs are illuminated initially when the system is powered up or reset. As the diagnostic steps through each circuit under test, the validity of each circuit function is verified, and an LED is extinguished. The diagnostic then advances to test the next circuit in sequence.

The following circuit functions are tested in sequence with the corresponding LED and display message providing status of the test results:

<u>DISPLAY MESSAGE</u>	<u>TEST SEQUENCE LED #</u>	<u>ON-BOARD CIRCUIT TESTED</u>
	1	Baud Rate Generator and Interrupt Controller (Z80-CTC Counter/Timer)
CO	2	Serial RS-232 I/O Port Dual Asynchronous Receiver/Transmitter (DART 2)
COM	3	Serial RS-232 I/O Port Dual Asynchronous Receiver/Transmitter (DART 1)
COMP	4	Floppy Disk Controller
COMPL	5	Parallel IEEE-488 I/O Port Controller
COMPLE	6	RAM Memory Locations 8000H - FFFFH
COMPLET	7	RAM Memory Locations 0000H - 7FFFH
COMPLETE	8	DMA Controller

4. If no malfunction of the hardware is detected by the diagnostic, the following message is displayed on the screen to notify the operator that the system disk can be installed and loaded into memory to generate the system parameters and begin system operations:

9520 SELF TEST VERS 2.XX COMPLETE

INSTALLATION AND CHECKOUT

Resolving Start Up Problems. If a malfunction is detected by the self test diagnostic, the LED associated with the failed circuit remains illuminated. The displayed message line is not completed, and the program will halt on the error. One of the following error messages is displayed at the terminal screen to notify the operator that a malfunction is present at startup:

<u>ERROR MESSAGE</u>	<u>LED #</u>	<u>DISPLAY CHARACTER</u>
CKSUM ERROR NO STACK ERROR TIMER ERROR	1	
RDR/PUN ERROR CONSOLE ERROR	2	CO
RS422 ERROR REM10 ERROR	3	M
INVALID INT ERROR	4	P
	5	L
RAM 8XXXH ERROR RAM CXXXH ERROR	6	E
RAM 0XXXH ERROR RAM 4XXXH ERROR	7	T
DMA TST BAD DATA UNEXPECTED DMA INT UNEXPECTED IEEE-488 INT UNEXPECTED 60 Hz INT	8	E

The presence of a malfunction makes it necessary to turn off AC power at the system and remove the top cover from the 9520 chassis to examine the LEDs. Turn on AC power to repeat the self-test diagnostic and check the LEDs and the display. The LED associated with the failed task will be illuminated along with corresponding LEDs for any remaining tests that were interrupted. The cause of the malfunction will be indicated by the displayed error message and the missing characters in the C-O-M-P-L-E-T-E string.

Initial Start Up of System (cont'd)

If the serial number of the 9520 System in use is 340 or greater, the following checkout procedure should be used.

The boot PROM diagnostic routine runs for approximately eight seconds. A total of 46 circuit functions are checked by the program. Each circuit function is coupled with an LED which remains illuminated or is extinguished to provide status to the operator. The displayed message line spells out the word MILLENNIUM, then erases it and spells out the word COMPLETE as each circuit function passes its test.

If no malfunction of the hardware is detected by the diagnostic, the following message is displayed on the terminal screen to notify the operator that the system disk can be installed and loaded into memory to generate the system parameters and begin system operations:

9520 SELF TEST VERS 2. COMPLETE

If a malfunction is detected, the displayed message is not completed, the LED associated with the test remains illuminated, and the program halts on the error. An error message is displayed.

The following pages list the possible malfunctions that may occur. In the left column is the displayed message line. At the far right is a listing of error messages that may appear on the screen with the incomplete message line. The middle two columns are information that does not appear on the screen, but may be used as a troubleshooting guide to detect a malfunction that occurs during initial start up. This guide uses the following codes:

- b = Known bug in code
- f = Fatal error; processing stops after error message.
- n = Non-fatal error. Processing continues after error message.
- i = Low memory interrupt may occur during this test.
 - f Parity FATAL Error
 - f Invalid Interrupt FATAL Error
 - f Reader/Punch FATAL Error
 - f RS-449 FATAL Error
 - f Remote I/O FATAL Error
- l = high memory interrupt may occur during this test.
 - n Parity Error
 - n Unexpected GPIB Interrupt
 - n Unexpected 60 Hz Interrupt
 - n Unexpected DMA Interrupt
 - f Check W-5 jumper: DMA interrupt failure FATAL Error

INSTALLATION AND CHECKOUT

<RAM ERROR REPORT FORMAT> : There are two possible errors. The first is a RAM memory error itself. The second is a failure while attempting to change the contents of the mapping RAM during the test. A RAM error can cause a mapping RAM error message in the following format:

Map x RAM Error Address = aaaa is = yy S/B = ss
Bank x RAM Error Address = aaaa is yy S/B = ss
Non-zero Mapping RAM failure yy S/B ss

where:

aaaa = four hexadecimal digits of address
x = one hex digit for map (0 to 3)
ss = two hex digits of anticipated data
yy = two hex digits of actual data read
S/B = should be

These codes, which can be found in columns 2 and 3 (Test Description and Flags), may help locate the source of a hardware malfunction.

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
		1 Verify correct PROM checksum		
		2 Initialize: timer chip, interrupt controller, and console UART channel Clear screen for 9501 terminal.		
<lf>		3 Initialize remote channel		
<cr>		4 Initialize RS-449 channel		
M		5 Initialize reader/punch channel		
Mi		6 Indicates PROM checksum failure	fi	CHECK-SUM FATAL Error
		7 Initialize Hard disk controller		
Mil		8 Initialize mapped memory expansion board		
Mill		9 Set interrupt mode 2 and page pointer	i	
Mille		10 Verify stack area	fi	NO STACK FATAL Error
Millen		11 Verify timer interrupts	fi	Check W-5 jumper FATAL Error or
Millenn			fi	Timer Interrupt FATAL Error not both
		12 Verify timer interval	i	Timer Error

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
Millenni	1 off	13 Test reader/punch channel interrupts	fi	
Millenniu		14 Disable console interrupts	fi	
Millennium	2 off	15 Send hardware configuration sign on message: a. Send carriage return with no line feed to erase "Millennium" b. "Mapped memory:" or "Banked Memory:" or "" c. "Hard Disk :" if present d. "9520 " e. "Self-Test Version 2.95"		
		16 Test RS-449 UART channel interrupts	fi	
C		17 Test Remote UART channel interrupts	fi	
Co	3 off	18 Test Floppy disk drive controller interrupts	fi	Floppy Controller FATAL Error
Com	4 off	19 Test General Purpose Interface Bus controller	i	GPIB Controller Error
Comp	5 off	20 Memory test 1: map 0 8000 to ffff hex	fi	<see RAM error report>
Compl	6 off	21 Move test code to high memory		
Comple		22 run memory test 2: map 0 0 to 7fff hex	fi	<see RAM error report>
Comple	7 off			

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
Complete		23 Verify map hardware maps to unique RAM segments	fi	ss Maps to yy <cr> MAP verify FATAL Error
Completed		24 Enable parity interrupt hardware		
Completed!<cr><lf>		25 Test Direct Memory Access controller. Send carriage return and line feed to terminate first message line	fi	Check W-5 jumper: DMA Interrupt Failure FATAL
	8 off		fi	DMA Test Bad Data FATAL Error
1		26 Memory test 3 is 48k: map 1 0 to bfff hex	I	<see RAM error report>
12		27 Memory test 4 is 16k: map 1 c000 to ffff hex	I	<see RAM error report>
123		28 Memory test 5 is 4k: map 2 8000 to 8fff hex Mapped at 1000 hex	I	<see RAM error report>
1234		29 Memory test 6 is 4k: map 3 8000 to 8fff hex Mapped at 2000 hex	I	<see RAM error report>
12345		30 Memory test 7 is 60k: map 2 9000 to ffff hex and map 2 0 to 7fff hex	I	<see RAM error report>
123456		31 Memory test 8 is 60k: map 3 9000 to ffff hex and map 3 0 to 7fff hex	I	<see RAM error report>

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
		32 Report memory configuration as tested. Only one line is displayed.		
> 64k Bytes :		a. No expansion memory in system or defective hardware		
1>112K bytes :		b. Banked memory board or partially defective map board		
12>128K bytes :		c. Mapped board with only 64K RAM stuffed		
123>128K bytes :		d. failed 28 and 29		
1234>128K bytes :		e. failed 30		
1235>192K bytes :		f. Mapped board with 2 64K banks stuffed		
12345>192K bytes :		g. Mapped board with 3 banks stuffed but failed test 31		
123456>256K bytes :		h. Mapped board fully stuffed		
		33 If hard disk is in system perform controller tests. This is added on same line as expansion RAM test results. "Hard Disk Sub-system"		
R		34 Run hard disk controller self test	hl	Hard Disk Controller Self-Test Error
Re		35 Run hard disk controller RAM self test	hl	Hard Disk Controller RAM Self-Test Error
Rea		36 Verify valid hard disk drive select switch setting	hl	Hard Disk Switch Encode Error
Read		37 Configure hard disk controller for "installed" drive	hl	Hard Disk Controller Parameter Error
		38 Verify hard disk is ready	hl	Hard Disk Drive Not Ready

At this point, rather than run system diagnostics, turn off AC power and check the printed circuit board connections to ensure the contacts are firmly seated in the back panel socket. Also, check the board interface harness and cable connectors to ensure all connections are tight, as these connections can become loose if the equipment has been subjected to rough handling. Turn on AC power to repeat the diagnostic with board connections secured. Verify the test is completed without malfunction, and that the message is displayed on the terminal screen to indicate the system can be placed in operation.

If the malfunction is still present, it will be necessary to load the System Diagnostics disk to isolate and correct the problem. Contact the Gould Customer Service Representative via the telephone numbers listed in the preface.

Restarting System Operations

```
* * * * *
*           CAUTION           *
*                               *
* Activation of the system reset switch *
* will terminate the current process. *
* Although it may be necessary to press *
* the reset button to unlock the system,*
* loss of data can occur.           *
* * * * *
```

System operations can be restarted by pressing the RESET pushbutton switch located on the control panel. Operation of this switch terminates the current process operation and places the system in a reset state. This state causes the system to self test, load CP/M or MP/M, and wait until the next command entry is issued from the terminal keyboard.

System Shutdown Operations

The user should wait for the current process operation to be completed, so that the system is waiting for an input command from the keyboard. Next, open the disk drive and remove the disk. Press the POWER switches on the development system control panel and at the display terminal to remove AC power from both units.

Removing Disk from Drive

The disk can be removed from the disk drive with all power on and the drive spindle rotating. Any read or write operation in process should be terminated before removing the disk from the drive. Likewise, any data in memory that is to be saved should be written to the disk before it is removed. Use the following procedure (see figure 3-8):

1. Verify all read/write operations have been completed at the selected drive (i.e. the LEDs on the latch handle should be turned off).
2. Press the latch bar to open the access door at the selected drive. The access door is spring-loaded to swing open and eject the disk from the spindle when the latch bar is pressed.
3. Pull the disk from the drive to complete the removal procedure.

Care of Disk

The floppy disk is a flexible disk enclosed in a permanent jacket. The interior of the jacket is lined with a wiping material that cleans the disk of foreign matter. The disk should be stored in an envelope when it is removed from the drive.

The following special precautions should be used to protect the disk during handling and storing:

1. Return the disk to its storage envelope whenever it is removed from the disk drive.
2. Keep disk away from magnetic fields and from ferromagnetic materials which might become magnetized. Strong magnetic fields can distort recorded data on the disk.
3. Replace storage envelopes when they become worn, cracked or distorted. Envelopes are designed to protect the disk.
4. Do not write on the plastic jacket with a lead pencil or ball-point pen. Use a felt tip pen.
5. Heat and contamination from a carelessly dropped tobacco ash can damage the disk.
6. Do not expose disk to heat or sunlight.
7. Do not touch or attempt to clean the disk face. Abrasions may cause loss of stored data.

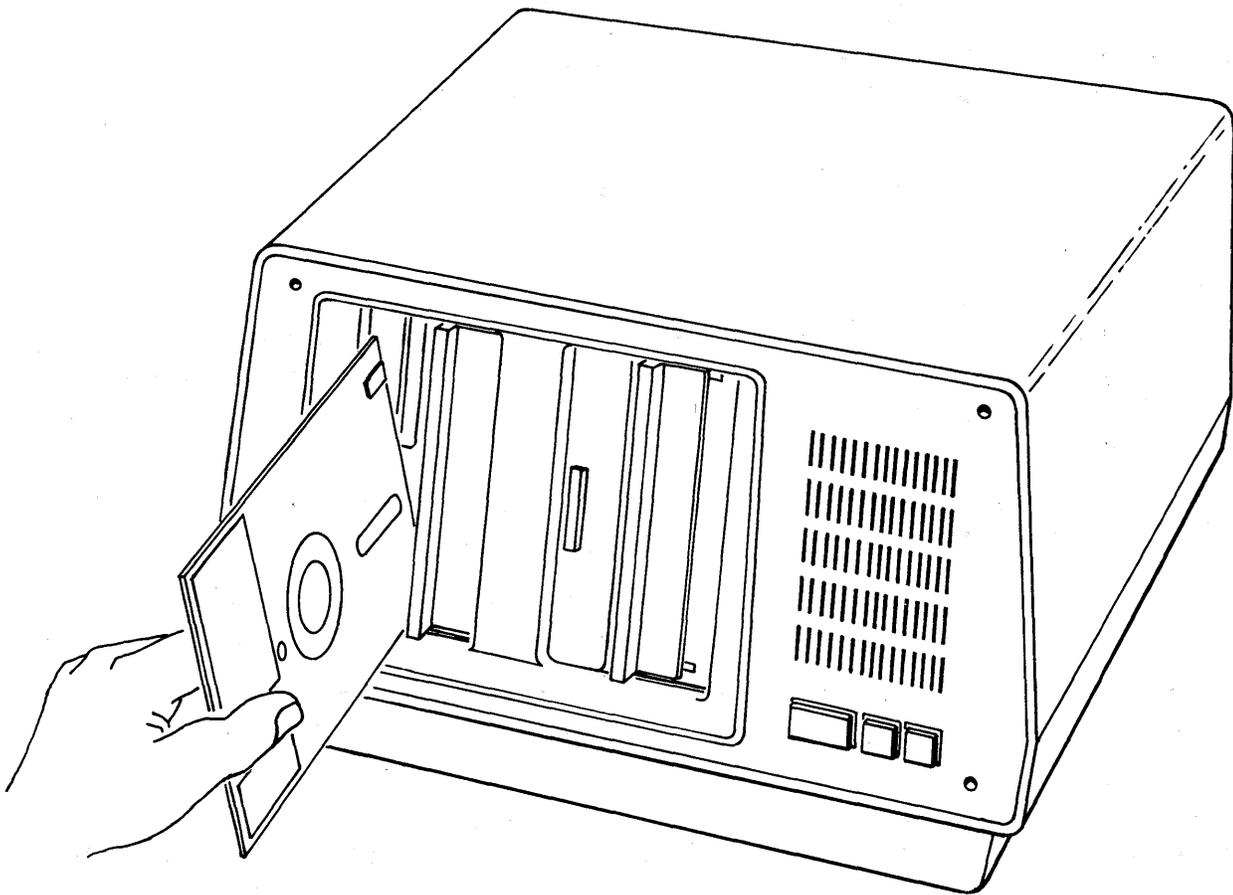
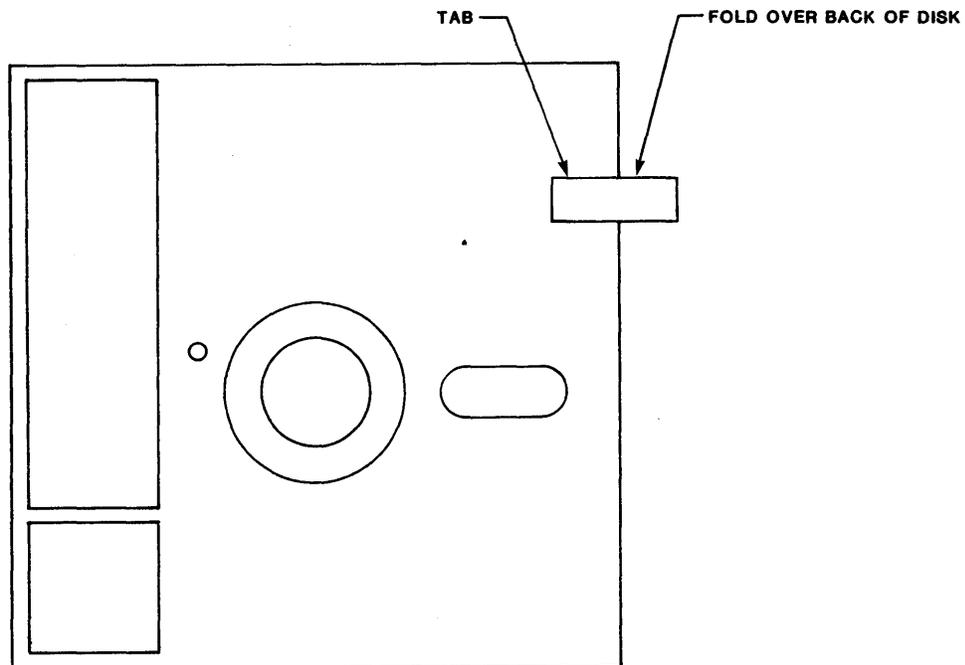


Figure 3-8. Inserting Disk in Drive

Write Protect Feature. When a disk contains master programs, text or other data intended for read-only purposes, it is important to prevent inadvertent writing to the disk that would destroy the data. The write protect feature of the disk, which inhibits writing to the disk, is enabled by the small hole located near the outside edge of the jacket. (See figure 3-9.) When the hole is open, writing is inhibited.

By covering the hole in the disk, writing is enabled. The hole is covered by placing a paper tab over the front of the hole and folding the tab over the edge of the jacket to cover the rear of the hole. The tab can be removed from the hole at anytime to restore the write protect condition.



9520-006

Figure 3-9. Disk Write Protect Tab

INTRODUCTION

This chapter discusses the following information needed to operate the 9520 Software Development System: the design and function of the terminal keyboard, the initiation of the system, the command modes, and the running of programs.

TERMINAL KEYBOARD AND KEY FUNCTIONS

The terminal keyboard is arranged like that of a typewriter, with the exception of a number of additional keys. The placement of the keys and the labels on the keys will differ according to the type of keyboard used. A typical keyboard arrangement showing all essential keys for using the 9520 Software Development System is shown in figure 4-1. These keys can be grouped into three functional categories: data entry keys, program control character keys, and text control character keys.

Data Entry Keys

Data entry keys are used to provide user input to, and interaction with the system. After keying in the desired input, the RETURN key must be pressed; only then does the system accept the input. Control character inputs, however, do not require a carriage return.

The data entry keys are similar to a standard typewriter keyboard. These keys provide 96 ASCII characters including upper and lower case English alphanumeric and special characters. This section of the keyboard also includes the control keys for DEL, CTRL, ESC and RETURN functions.

The DEL key is used to delete the last character that was typed at the keyboard. The DEL key may also be labeled DELETE, RUB or RUBOUT and may, or may not, be shared with the underscore key. It may, or may not, require SHIFT to activate.

The RETURN key may be labeled CARRIAGE RETURN or ENTER and causes the cursor to be moved to column 1 of the current line.

The space bar is for entering spaces. Unlike a typewriter, the space bar cannot be used to move across characters already displayed on the screen.

The CTRL key is used like a shift key to enter alphabetic control characters. To type a control character, hold the CTRL key down while typing the letter. In this manual, a caret (^) character is used in front of the letter to indicate a control character. For example, ^D indicates control-D, which is typed by holding down the CTRL key and typing a D; B means Control-B, etc. Control characters are used to enter commands for program control and text control during data entry. It is not necessary to use the RETURN key after entering a control character.

OPERATION

A number of additional keys may be present for use in data entry which include the following:

BACKSPACE: Same as control character, H which is used for backspacing the cursor on a line.

TAB: Same as control character, I which is used with the Text Editor Utility for tabbing.

LINEFEED: Same as control character, J which is used to terminate the current input and cause the cursor to move down one line on the screen.

REPEAT: Used to automatically send the same character continuously. Some keyboards will repeat any character whose key is held down 1/2 second. Other keyboards require the REPEAT key to be held down while another key is pressed to send the character string.

CURSOR DIRECTION: Consists of four keys with arrows to indicate the four directions of cursor motion. Pressing the key causes the cursor to move in the direction indicated by the arrow.

Cursor motion can also be initiated by control characters used in the WordStar Text Editor Utility as follows:

^D: Moves cursor to the right -- to the next character in a line.

^S: Moves cursor to the left -- to the previous character in a line.

^E: Moves cursor up to preceding line on screen.

^X: Moves cursor down to next line on screen.

Program Control Character Keys

The program control characters provide limited user control over the program operation processes, such as:

- o Aborting a Program
- o Detaching from a Running Program
- o Terminating the Current Input
- o Deleting and Inserting a Line
- o Obtaining Exclusive use of a Listing (printer) Device
- o Stopping the Display Output Before Continuing with Execution



Figure 4-1. Typical Keyboard Arrangement

OPERATION

The program control characters also permit limited editing functions of a line entry while typing in a command line at the keyboard:

The following control characters are used in the MP/M or CP/M system:

MP/M, CP/M

CONTROL CHARACTER

DESCRIPTION OF CONTROL FUNCTION

C	Abort program in process and terminate execution
D	Detach from a running program (MP/M only)
E	Physical end of line
H	Delete the last character typed at keyboard and backspace one character position
J	Terminate current input (same as line feed)
M	Terminate input (same as carriage return)
P	Copy all remaining keyboard outputs to the list (printer) device. Output is sent to both the list device and display device until the next $\wedge P$ is typed. For MP/M, if the list device is not available, a PRINTER BUSY status message displayed on the screen.
Q	Obtain ownership of the PRINTER BUSY status message (MP/M only). Attaching the printer via this command prevents other terminal consoles from gaining access to the printer by issuing $\wedge P$, $\wedge Q$, PIP, or SPOOLER commands. The printer is thus owned by the console which issued the $\wedge Q$ command until another $\wedge P$ or $\wedge Q$ command is entered to release the printer. The $\wedge Q$ command should only be used when a program is executed that does not get exclusive control of the printer before writing to it. If the printer is not available, a PRINTER BUSY message is displayed on the screen.
R	Retype current command line to provide a clean line following character deletion with rubouts
S	Temporarily stops the display motion. The program execution and display motion will resume after any character is typed on the keyboard. $\wedge S$ is used to halt the display motion on high speed displays so that the operator can view a segment of the output data before continuing.
U	Remove current line after new line
X	Delete entire line typed at keyboard and backspace to the beginning of the current line
Z	End input from keyboard

Text Control Character Keys

The WordStar™ Text Editor Utility provides an array of text control characters that are fully described in chapter 6 of the WordStar manual.

These text control characters are used in conjunction with the CTRL key to issue a specific command to control the processing for text inputs, such as, creating a new document file, editing an existing document file, manipulating the display, and listing of document files. Text control functions involve the following categories of commands that can be executed by the text control characters:

1. Cursor Motion -- Forward, Backward, Upward and Downward directions
2. Scrolling -- Single line, full screen, continuous, up/down and fast/slow motion
3. Text Entry -- Insertion, tabbing and tabulating
4. Text Deletion on a Page -- Individual characters, words, lines and blocks of data
5. Saving and Deleting -- Updated files, original files and old files
6. On-Screen Text Formatting -- Set margins, line spacing, tabs and paragraphs
7. Find and Replace -- Global File, Word, Phrase, Character Strings
8. Place Markers -- Set mark locations in a file for future reference
9. Block Commands -- Moving paragraphs, sentences, copying and deleting blocks
10. Document or non-document files
11. Help Commands -- Display learning aids and reference user information on screen
12. Miscellaneous Commands -- Repeat a command, interrupt a command/printing and display file directory

OPERATION

SYSTEM INITIATION

System Initiation for MP/M

1. Follow the procedures in chapter 3 for initial start up of the system.
2. Insert the system disk in either of the disk drives.
 - a. Press the latch bar to open the drive access door at the selected drive.
 - b. Position the disk as shown in figure 3-8 and push the disk forward until a click is heard.
 - c. Close the drive access door to lock the disk on the drive spindle.
3. Press the RESET switch on the 9520 Development System control panel to start up system operations and load the operating system from the disk.
 - a. The system will perform the self-test diagnostic and then load the bootstrap loader program from the disk, which in turn loads the operating system from the disk.
 - b. After a few seconds, loading is complete, and the following message appears on the display screen:

```
MP/M  
XA>
```

where: XA shown in prompt:

X = User code and console assignment number. For example, console #0 is initialized to user #0 for a single user/console system configuration and would be presented as:

```
OA>
```

Variations for assigning multiple users and consoles to the system are described in section 1.3 of the MP/M Users Guide.

A = Default for the disk drive (A or B) that is currently logged to the console and can be changed at any console as described in section 1.3 of the MP/M Users Guide.

The specified default disk must contain the utility files, such as, DIR, REN, ERA, etc.

NOTE: The reset procedure need not be performed when the 9520 SELF TEST VERS 2.0 COMPLETE is displayed.

5. After the initial loading of the MP/M operating system is complete, perform the System Generation Procedure described in chapter 8 to select the operating system parameters. This will place the system in the Command Interpreter State, so that the system is ready to accept and interpret commands issued from a local console keyboard or process UPLOAD/ DOWNLOAD commands that are issued from a remote emulator/debug station.

System Initiation for CP/M

1. Follow steps 1, 2 and 3 above.
2. Press the RESET switch on the 9520 Development System control panel to start up system operations and load the operating system from the disk.
 - a. The system performs the self test diagnostic and then loads the operating system from the disk.
 - b. After a few seconds, loading is completed and the following message appears on the display screen:

```
XXK CP/M  
A>
```

where: A = the disk drive that is currently logged on.

ENTERING COMMANDS

A command may be keyed as an explicit entry (for the Immediate Command Mode) by specifying all of the parameters in the command prior to execution. A command may also be keyed as an implicit entry (for the Interactive Command Mode) so that the user can interact with, and respond to queries issued by the system to provide parameter inputs during the execution. Both types of command entries are described, and examples are given below.

Immediate Command Mode

The Immediate Command Mode allows entry of one-line commands. The parameters included in the line entry identify operators and operands that are required to complete the program execution. The user will not be prompted for additional input.

OPERATION

Syntax for an Immediate command is as follows:

<command> <parameter-1>...<parameter-n> <cr>

where:

- <command> -- identifies a 9520 system utility name, an MP/M or CP/M system utility name, or a user program name (e.g., utility name could be BRATE)
- <parameter-1> thru <parameter-n> -- defines parameters required to run the program. Each parameter may consist of a file name, a function, a device name, an indicator or an assigned value. (e.g., function parameter #1 could be I/O device: PRINTER, parameter #2 could be arithmetic indicator: = parameter #3 could be value for Baud Rate)
- <cr> -- indicates a carriage return is required to complete the command entry

Entering Immediate Command

The Immediate Command can be keyed by using the following guidelines:

1. Use data entry keys to key in characters.
2. Corrections can be made by erasing (using the DELETE, RUB, RUBOUT, BACKSPACE or control^H character key) each of the characters up to, and including the first erroneous character in the command, and then re-entering them correctly. The control^X character may be used to erase the entire line and to start a new line entry.
3. Enter parameters to define operators and operands for the program.
4. Press the RETURN key to initiate execution of the selected program command.

EXAMPLE: Use of single-line Immediate Command entry to call out the Baud Rate Utility and assign a new Baud Rate value for I/O device.

Syntax: <BRATE> <I/O Device> <=> <BAUD Rate> <cr>

(5) carriage return,
execute command

(4) BAUD Rate Value assigned = 300

(3) Parameter for arithmetic indicator

(2) Name of device to be changed = PRINTER

(1) Name of Utility = BRATE

Keyboard Entry: BRATE PRINTER=300 <cr>

Interactive Command Mode

The Interactive Command Mode allows the user to call out a utility and respond to prompts issued by the system after the program begins execution. Each of the prompts will query the user to provide additional input information so that execution can be resumed and/or completed.

The following types of input information are requested by the system prompts. Each is followed by a carriage return:

- o Parameter #1 (device name, file name, type of function)
- o Parameter #2 (Indicator, Operator)
- o Parameter #3 (Value, address location operand)

Syntax for the Interactive Command is as follows:

```

Step 1. Call out program:           <command> <cr>
Step 2. Respond to prompt,
      enter 1st parameter:         <parameter #1> <cr>
Step 3. Respond to prompt,
      enter 2nd parameter:         <parameter #2> <cr>
      .
      .
      .
Step 4. Respond to prompt,
      enter last parameter:        <parameter #n> <cr>

```

Where each entry step performs the following functions:

- (1) <command> <cr> -- identifies a 9520 System Utility name, an MP/M or CP/M System Utility, or a user program name that begins executing when carriage return is entered.
- (2) <parameter #1> <cr> -- prompts the user to input specific information to satisfy program processing requirements and resume execution.
- (3) <parameter #2> <cr> -- prompts the user to input additional information to satisfy program processing requirements and resume execution.
- (4) <parameter #n> <cr> -- prompts the user to input final information so that processing will be completed.

OPERATION

Entering Interactive Command

The Interactive Command can be keyed by making the entry in this manner:

1. Use data entry keys to key in characters.
2. Make corrections at any time before pressing the RETURN key.
3. Corrections are made in the same manner described for Immediate Command Entry. The C character may be used to abort the program at any time.
4. Enter each parameter as requested by the various prompts to define specific operators and operands for the program. Press the RETURN key after each parameter entry to resume or complete the execution.

EXAMPLE: Use the multiple-line, Interactive Command Entry to call out the Baud Rate Utility and assign a new Baud Rate value for I/O device:

<u>SYSTEM PROMPT</u>		<u>SYNTAX: USER ENTRY</u>	
(1) OA>	<BRATE>	<cr>	carriage return, begin execution
			Name of Utility, BRATE
(2) I/O DEVICE:>	<PRINTER>	<cr>	carriage return, resume execution
			I/O device selected
(3) BAUD RATE:>	<300>	<cr>	carriage return, complete execution
			Baud Rate Value assigned

Where each entry step performs the following functions:

- (1) <command> <cr> -- identifies a 9520 System Utility name, an MP/M or CP/M System Utility, or a user program name that begins executing when carriage return is entered.
- (2) <parameter #1> <cr> -- prompts the user to input specific information to satisfy program processing requirements and resume execution.
- (3) <parameter #2> <cr> -- prompts the user to input additional information to satisfy program processing requirements and resume execution.
- (4) <parameter #n> <cr> -- prompts the user to input final information so that processing will be completed.

RUNNING A PROGRAM

Program execution is initiated by typing the Program Name followed by a carriage return <cr>. Some programs contain one or more parameters which follow the program name on the line entry. The programs provided with MP/M are described in sections 1.4 and 1.5 of the MP/M Users Guide, and in Chapter 6 of The CP/M Handbook.

ABORTING AN ATTACHED PROGRAM

Under CP/M or MP/M a program can be aborted by keying a control C (\wedge C) character at the console. The \wedge C terminates execution of the program that was initiated (and thereby attached) by the console. Under MP/M a program can be detached from its terminal; under CP/M it cannot be detached. A detached program cannot be aborted with a \wedge C. A detached program must first be attached and then aborted. A running program may also be aborted using the ABORT command as described in section 1.5 of the MP/M Users Guide.

DETACHING FROM A RUNNING PROGRAM

Detaching from a running program may be invoked by keying a control D (D) character at the console. In order to detach a program using the D character, the executing program must be performing a check console status to observe the detach request.

ATTACHING TO A DETACHED PROGRAM

A detached program (i.e., the program is not owned by a console) may be attached to a console by keying: ATTACH, followed by the program name. A program may only be re-attached to the console from which it was detached.

OPERATING SYSTEM AND UTILITIES

INTRODUCTION

Software for the 9520 Development System is based on the CP/M Operating System or MP/M Operating System. MP/M and CP/M incorporate command formats, conventions, syntax and file structures that support the following types of programming capabilities:

- o Utility programs to download and upload executable load modules between the 9520 Development System and a remote debug/emulator station and to transfer files to and from other devices
- o A floppy disk utility to format, duplicate and copy information on disks
- o I/O Baud Rate utility to display and set baud rate values for the 9520 I/O ports
- o A text editor to prepare and manipulate the user's source program files.
- o A variety of assemblers that translate source program files into target object files for different types of microprocessors
- o A linker to link object programs together to produce the executable load module.

The 9520 Software is contained on three disks:

- (1) System Disk
- (2) Language Translator Disk*
- (3) System Diagnostic Disk

*NOTE: Separate Language Translator disks are available for the various assemblers and cross assemblers used to support the preparation of object files for different types of microprocessors.

SOFTWARE ORGANIZATION

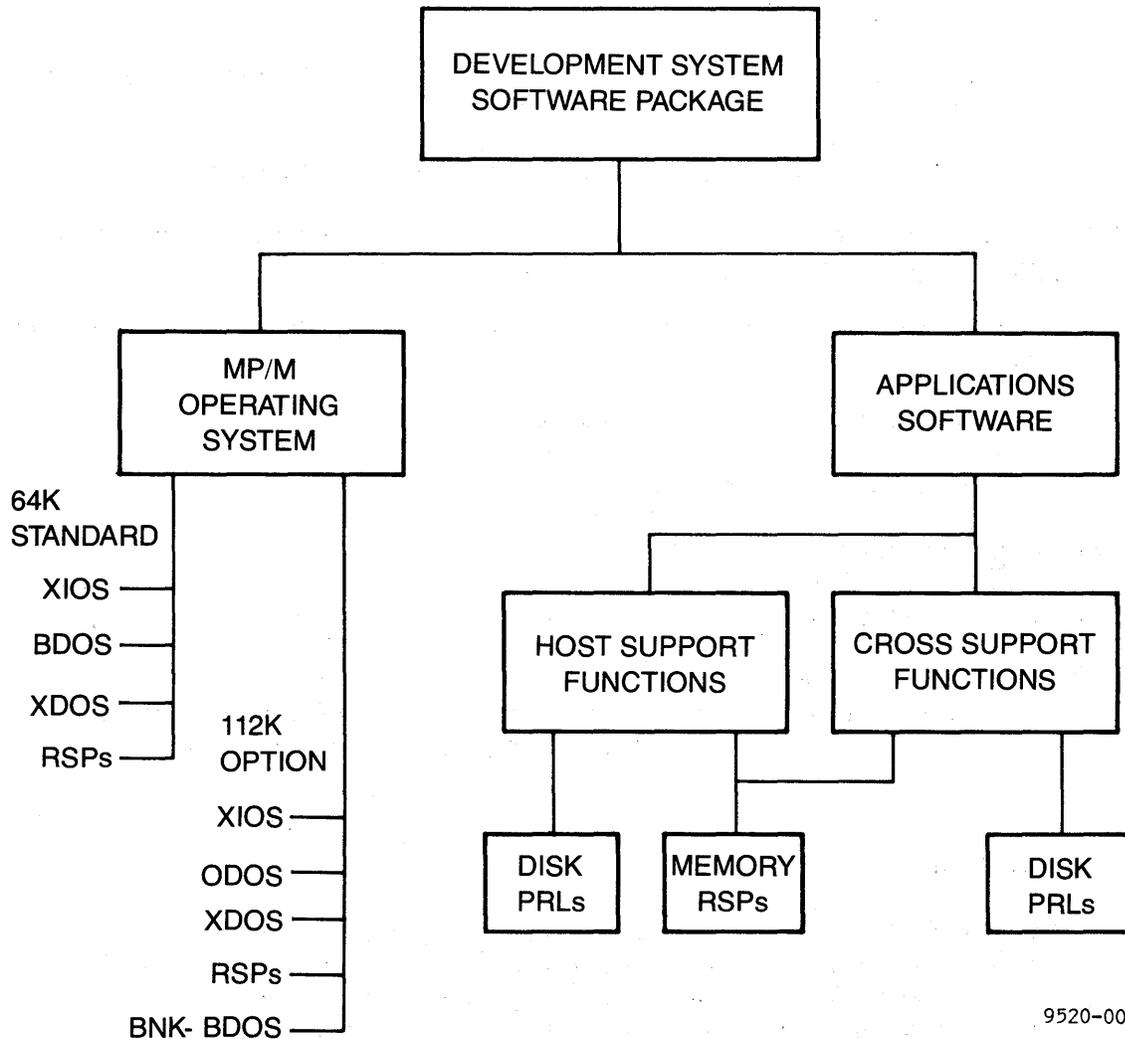
The 9520 Development system software is organized around the operating system and applications software as shown in figure 5-1. The operating system is described in the MP/M Users Guide or the CP/M Handbook. The applications software can be divided into two functional groups: (1) Host Support Functions, and (2) Cross Support Functions.

OPERATING SYSTEM AND UTILITIES

Software Support Functions

Host support functions contain all program files that are made available to the 9520. Cross support functions contain programs and specific files that will produce target code for a remote debug/emulator.

Both the host support and cross support software is further divided into memory and disk resident programs. This distinction (memory/disk) is merely to differentiate between programs permanently resident in the system memory from those accessed from the disk.



9520-007

Figure 5-1. Development System Software Organization

For MP/M, programs loaded as part of the operating system are classified as resident system processes, RSP. Programs that are transient and loaded on page boundaries are classified as page relocatable, PRL. Programs that are transient and are loaded at a fixed address are classified as command programs (or files), COM. RSPs and PRLs associated with host and cross support functions are shown in figure 5-2.

For CP/M, the transient programs are classified as command programs, COM.

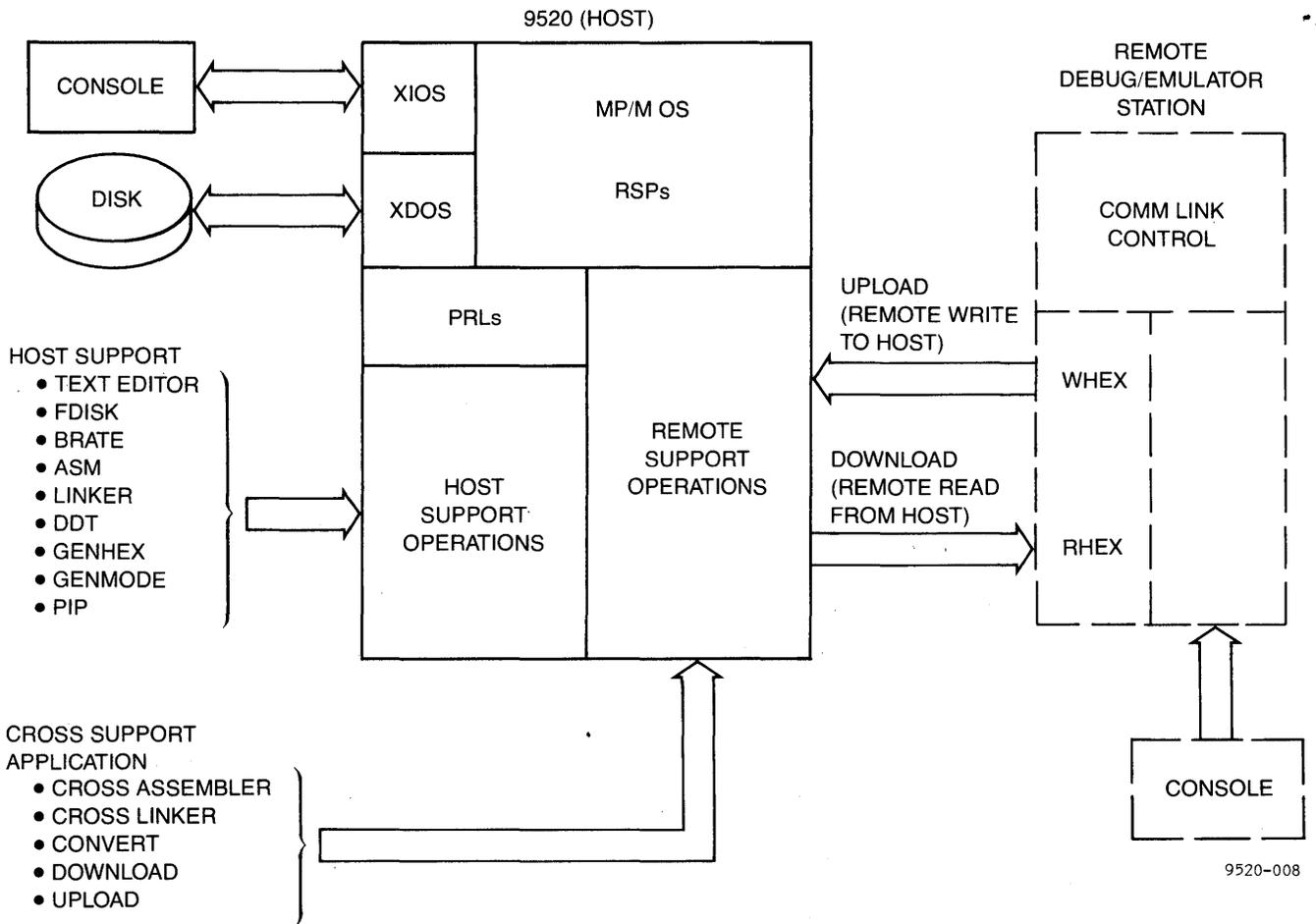


Figure 5-2. Cross Support Interface with Remote Station

OPERATING SYSTEM AND UTILITIES

CONSOLE COMMANDS

This section describes the commonly used console commands that are issued by the operator to communicate with the software. For MP/M, the commands available at the console are established by resident system process (RSP) programs that are included by the user during the system generation or by disk resident files. (See chapter 8 for details of Operating System generation for MP/M.) Disk resident files may be either page relocatable (PRL) modules or absolute (COM) modules. For CP/M, disk resident files are (COM) files.

The MP/M system does not use a system-defined, or built-in command structure that contains a fixed number of commands. Each available command coincides with a specific RSP program that is included by the user during the system generation process or by a file on a disk where the file name is the same as the command and the file type is PRL or COM. For CP/M, the file type must be COM. These commands can be categorized into four groups as follows:

1. User Identification Commands
2. File Manipulation Commands
3. System Operation Commands
4. Program Operation Commands

A listing and short description of individual commands associated with each category are included in the following pages. The user is referred in each case to the appropriate page(s) in the MP/M Users Guide and The CP/M Handbook for a complete explanation and examples.

User Identification Commands

The user identification commands permit a user to perform the following functions:

- o Display and/or set user area number - USER
- o Display user console number - CONSOLE
- o Reset Disk Drives - DSKRESET

USER

Command Name: USER

Operating System: MP/M

Function: The GET/SET USER AREA is used to display the current user area as well as to set a user area value.

To Display User Code: Entering the USER command, followed by <cr> will display the current user area. (Observe that the current user area is always displayed in the prompt.)

Syntax: 0A> USER <cr>

To Set the User Area in a Specified Number: Entering the command, USER, followed in sequence by a space, user area and then a <cr> will set the user area to the specified user area number. The example which follows assigns a user area value of 2.

Syntax: 0A> USER 2 <cr>

References: MP/M User's Guide, page 8
CP/M Handbook, pages 96, 270

OPERATING SYSTEM AND UTILITIES

CONSOLE

Command Name: CONSOLE

Operating System: MP/M

Function: The CONSOLE command is used to display the console number at the location where the command is being entered. The console number thus displayed, allows the user when examining system status to determine which processes are detached from consoles.

Syntax: OA> CONSOLE <cr>

Response: Console = 1

References: MP/M Users Guide, page 8
CP/M Handbook, page 224

DSKRESET

Command Name: DSKRESET

Operating System: MP/M

Function: The DSKRESET (Disk Reset) command is used to allow the operator to change disks. If there are open files on any of the drives to be reset, the disk reset is not allowed. The cause of the rejection is displayed in an error message. Open files (i.e., in process of being written to the disk) will lose their updated information if they are not closed prior to the disk reset command.

To test all drives: Entering the DSKRESET command followed by a <cr> allows all drives to be reset:

Syntax: 0A> DSKRESET <cr>

Response: Confirm reset disk (Y/N)?

To reset a specific drive: Entering the DSKRESET command, followed in sequence by a space, the drive descriptor and a <cr> allows a specific drive to be reset.

Syntax: 0A> DSKRESET B: <cr>

References: MP/M Users Guide, page 8
 CP/M Handbook, 229

OPERATING SYSTEM AND UTILITIES

FILE MANIPULATION COMMANDS

The File Manipulation Commands allow the user to perform the following operations on system files:

- o Erase a File - ERA/ERAQ
- o Display Contents of Source File - TYPE
- o List all Filenames - DIR
- o Change Filenames - REN
- o Provide Status of File Storage - STAT
- o Produce PRL Type Files from Hex Type Files - GENMOD
- o Produce Hexadecimal Type Files from COM files - GENHEX
- o Produce COM Type Files from PRL Type Files - PRLCOM
- o Produce PRL Type Files from COM Type Files - GENHEX and GENMOD

The parameter entered in a command to reference the filename consists of two parts, primary filename and an extension. The primary filename is a 1-8 character, user-assigned name for the file. The extension is a 1-3 character name for the file type that may be assigned by the user or, under certain circumstances, a utility assigned by default.

A period (.) is used to separate the primary filename from the extension, and is used in a command in the following forms to specify the type of processing search that will be performed on files:

Where: Variations of the parameter entry are:

- *.* = Process all system files (primary filenames and extension parameters do not define a specific filename or file type.)
- *.ASM = Process only those files assigned with the ASM file type descriptor.
- W.ASM = Process only the file referenced by the W primary filename and ASM file type descriptors.
- W.* = Process only those files assigned the W primary filename descriptor.

OPERATING SYSTEM AND UTILITIES

ERA/ERAQ

Command Name: ERA/ERAQ

Operating System: ERA - MP/M and CP/M
ERAQ - MP/M

File Type: .COM or .PRL in MP/M
.COM in CP/M

Function: Two commands, ERA (erase) and ERAQ (erase Query), are used to delete files. The ERA command is an immediate entry that allows the user to delete a specified file or all files. The ERAQ command is an interactive entry that initially displays a list of specified files in the system. The user is prompted by a query to respond with Y or N (for yes or no confirmation) to delete each file contained in the list.

To delete a specific file: Entering the ERA command followed by the filename and extension; and <cr> will delete the named file.

Example: OA>ERA FASTPROB.TXT <cr>

To delete all files of a specific type: Entering the ERA followed by an asterisk, *, in the primary filename position and a specific type as an extension will delete all files of a specific type.

Example: OA>ERA *.SRC <cr>

To delete all files: A reference to a file may be unambiguous or ambiguous. The presence of a question mark in a file name allows the substitution of any character where performing a directory search. The unambiguous reference

MILLDEV.SYS

is only satisfied by the directory entry

MILLDEV.SYS

The ambiguous reference

MILL???.SYS

is satisfied by the directory entry

MILLUSA.SYS
MILLDEV.SYS

but not

MILLFAST.SYS

OPERATING SYSTEM AND UTILITIES

ERA/ERAQ (Cont'd)

To display specific files and delete from list: Entering the ERAQ command followed by the specified filename parameters allows the user to delete only those files that match the filename reference.

Syntax: OA> ERAQ *.LST <cr>

References: MP/M Users Guide, page 9
CP/M Handbook, pages 232-235

TYPE

Command Name: TYPE

Operating System: CP/M and MP/M

Function: The TYPE command displays the contents of a specified ASCII file on the screen. The user can specify the number of lines of data to be displayed on a page. The TYPE command automatically expands tabs at every eighth-column location.

For MP/M, a file parameter following the TYPE command is required. The TYPE command has an optional pause mode to halt the display after the specified number of lines appear on the screen. The pause mode is set by entering the letter P, followed by two decimal digits (to indicate line count) after the filename extension entry. The pause mode will cause the display to halt until the <cr> key is pressed. The additional lines will then be displayed, until the end of file is reached.

In the following example, the primary filename, DUMP, distinguishes a category name of source files. The extension filename, ASM, distinguishes the type of file for a particular category.

NOTE: The Peripheral Interface Command (PIP) may also be used to display source files. The description of the PIP command is described under System Operation Commands.

Syntax: OA> TYPE DUMP.ASM P23 <cr>

References: MP/M Users Guide, page 9
CP/M Handbook, page 269

OPERATING SYSTEM AND UTILITIES

DIR

Command Name: DIR

Operating System: MP/M and CP/M

Function: The DIR (directory) command causes a list of the filenames stored in a disk to be displayed at the terminal. The command can be invoked to list all filenames stored on the currently assigned disk drive, all filenames stored on a specific disk drive, or list particular filenames stored on a specific disk drive. An error message is displayed if a requested filename cannot be found on the addressed disk drive.

The parameter field entered in the command string to identify the disk drive name (e.g., A or B) must be followed by a colon (:). If the drive name is not specified in the command, the system will address the last drive that was assigned.

To list all filenames on the currently assigned disk drive: Entering the command, DIR, followed by a <cr> will display all filenames on the currently assigned drive.

Syntax: OA> DIR <cr>

To list filenames stored on a specific disk drive: Entering the command, DIR, followed in sequence by the parameter for a specified disk drive and a <cr>, will cause the specified disk drive to be addressed and a search to be conducted to list all filenames stored on the disk.

Syntax: OA> DIR B: <cr>

To list particular filenames on specified disk: Entering the command, DIR, followed in sequence by the parameter for a specific disk drive, the two parameters for primary filename and/or extension, and a <cr>, will cause the specified file to be displayed.

Syntax: OA> DIR B: * .ASM <cr>

References: MP/M Users Guide, page 10
CP/M Handbook, page 227

REN

Command Name: REN

Function: The REN (rename) command allows the user to change the name of files stored on the disk. It is assumed that the currently assigned disk contains the old filename to be changed to a new filename.

To change existing filename, MYFILE.ASM to read YY.ZZ:

Syntax: OA> REN YY.ZZ= MYFILE.ASM <cr>

References: MP/M Users Guide, pages 10, 42
CP/M Handbook, page 252

OPERATING SYSTEM AND UTILITIES

STAT

Command Name: STAT

Operating System: MP/M and CP/M

Function: The basic function of the STAT (status) command is to provide statistical information about file storage and device assignments. The STAT command is invoked as follows:

Syntax: OA> STAT [command line] <cr>

To display status of the active drive:

Enter STAT

OA> STAT

A: R/W, Space:24K

where: The active drive is A.

R/W indicates the disk in drive A can be read from or written to.
Space left on the disk for data entry is 24K.

To display status of a drive other than the active drive:

Enter STAT "non-active drive name"

OA> STAT B:

Bytes remaining on B: 400K

where: The active drive is A.

B is the non active drive name.

To display the status of a file:

Enter STAT "full filename"

OA> STAT RASP.BAK

References: MP/M Users Guide, page 11
CP/M Handbook, pages 259-261

GENMOD

Command Name: GENMOD

Operating System: MP/M

Function: The GENMOD command accepts a file containing two concatenated files of type HEX which are offset from each other by 0100H bytes, and produces files of type PRL (page relocatable module). The GENMOD command is invoked in the following form:

Syntax: OA> GENMOD B: PROG.HEX PROG.PRL \$bbbb <cr>

References: MP/M Users Guide, page 11
CP/M Handbook, page 237

OPERATING SYSTEM AND UTILITIES

GENHEX

Command Name: GENHEX

Operating System: MP/M

Function: The GENHEX command is used to produce a file of type HEX from a file of type COM. This capability allows the user to generate HEX files for GENMOD input. The GENHEX command has two parameters which consist of the COM filename and the offset value of the HEX file.

Syntax: OA> GENHEX primary filename.COM 100 <cr>

References: MP/M Users Guide, page 11
CP/M Handbook, page 236

PRLCOM

Command Name: PRLCOM

Operating System: MP/M

Function: The PRLCOM command accepts a file of type PRL (page relocatable module) and produces a file of type COM (absolute). If the destination COM file already exists, a query is made to determine if the file should be deleted before continuing the command processing.

Syntax: OA> PRLCOM d:(filename) .PRL d:(filename) .COM <cr>

References: MP/M Users Guide, page 11
CP/M Handbook, page 251

OPERATING SYSTEM AND UTILITIES

System Operation Commands

The system operation commands permit the user to perform the following functions:

- o Transfer data files from one peripheral I/O device to another via the peripheral interchange program command - PIP
- o Assemble the users program and store the result on disk - ASM
- o Select a file of commands for automatic batch processing - SUBMIT
- o Display the contents of a specified disk file on the screen in hexadecimal form - DUMP
- o Load a specified disk file with hexadecimal machine code and produce a memory image file which can be executed - LOAD
- o Display run-time status of the MP/M Operating System via the system status command - MPMSTAT
- o Transfer ASCII text files to a list device - SPOOL

PIP

Command Name: PIP

Operating System: MP/M and CP/M

Function: The PIP (peripheral interchange program) command allows the user to initiate data transfer operations between a disk file and other peripheral devices. The PIP command may be invoked by specifying the Interactive Command mode or Immediate Command mode.

To invoke PIP in Interactive Mode: Entering the command PIP followed by a <cr> will call out the program.

Syntax: OA> PIP <cr>

To invoke PIP in Immediate Mode: Entering the command PIP, followed in sequence by the command line entry and a <cr> will call out the program and execute the command without the need for user interaction. The program automatically terminates after execution is completed.

Syntax: OA> PIP (command line) <cr>

References: MP/M Users Guide, page 10
CP/M Handbook, pages 247-250

OPERATING SYSTEM AND UTILITIES

ASM

Command Name: ASM

Operating System: MP/M and CP/M

Function: The ASM (assembler) command allows the user to assemble a specified program on a disk. The MP/M assembler is invoked as follows:

Syntax: OA> ASM (filename) (flags) <cr>

References: MP/M Users Guide, page 10
CP/M Handbook, page 221

SUBMIT

Command Name: SUBMIT

Operating System: MP/M and CP/M

Function: The SUBMIT command allows a user to combine several commands into a single file for automatic batch processing. The SUBMIT function creates a file of substituted commands with the name, \$\$\$SUB. The SUBMIT command is invoked as follows:

Syntax: OA> SUBMIT (filename) (parameter #1 ... parameter #n) <cr>

References: MP/M Users Guide, page 10
CP/M Handbook, page 263

OPERATING SYSTEM AND UTILITIES

DUMP

Command Name: DUMP

Operating System: MP/M and CP/M

Function: The DUMP command displays the contents of the specified disk file at the console in hexadecimal form. The contents are listed 16-bytes at a time, with the absolute byte address listed in hexadecimal to the left of each line.

Syntax: OA> DUMP (filename) <cr>

References: MP/M Users Guide, page 11
CP/M Handbook, page 230

LOAD

Command Name: LOAD

Operating System: MP/M and CP/M

Function: The LOAD command reads the specified filename, which is assumed to contain hexadecimal Intel Hex format machine code and produces a memory-image file that can be subsequently executed. The LOAD command is invoked as follows:

Syntax: OA> LOAD (filename) <cr>

References: MP/M Users Guide, page 11
CP/M Handbook, page 241

OPERATING SYSTEM AND UTILITIES

MPMSTAT

Command Name: MPMSTAT

Operating System: MP/M

Function: The MPMSTAT (MP/M System Status) command allows the user to display the run-time status of the MP/M Operating System. The MPMSTAT command is invoked as follows:

Syntax: OA> MPMSTAT <cr>

References: MP/M Users Guide, pages 13-15
CP/M Handbook, pages 103-105, 212-215, 246

SPOOL

Command Name: SPOOL

Operating System: MP/M

Function: The SPOOL command allows the user to transfer (spool) ASCII text files to the list device. Multiple filenames may be specified in the command tail. The spooler expands tabs (ctl-1 characters), assuming tab positions are set at every eighth column.

The spooler queue can be purged at any time by using the STOPSPLR command.

The SPOOL command is invoked as follows:

Syntax: OA> SPOOL filename, [filename,...] <cr>

The non-resident version of the spooler (SPOOL.PRL) differs in its operation from the SPOOL.RSP. SPOOL.PRL uses all of the memory available in the memory segment in which it is running for buffer space. SPOOL.PRL displays a message indicating its status and then detaches from the console. Also, it may be aborted from a console other than the initiator only by specifying the console number of the initiator as a parameter of the STOPSPLR command.

The STOPSPLR command is keyed:

Syntax: OA> STOPSPLR 0 cr

References: MP/M Users Guide, page 15
CP/M Handbook, page 257

OPERATING SYSTEM AND UTILITIES

Program Operation Commands

The program operation commands allow the user to perform the following functions:

- o Invoke the MP/M Text Editor Utility - ED
- o Invoke the Dynamic Debugger Utility - DDT
- o Examine and Set the System Date and Time Parameter - TOD
- o Schedule Programs for Execution - SCHED
- o Abort a Running Program - ABORT

ED

Command Name: ED

Operating System: MP/M and CP/M

Function: The ED (editor) command allows the user to create and edit ASCII text files.

The alternate text editor program, WordStar, described in Chapter 6, may be used to provide an extended ASCII text processing capability for 9520 Software Development System applications.

The ED command is invoked as follows:

Syntax: OA> ED <cr>

Reference: CP/M Handbook, pages 24, 28, 146, 231

OPERATING SYSTEM AND UTILITIES

DDT

Command Name: DDT

Operating System: MP/M and CP/M

Function: The DDT (dynamic debugging tool) command loads and executes the debugger. The DDT is described in detail in chapter 6 of this manual.

The DDT command is invoked as follows:

Syntax: OA> DDT <cr>

References: MP/M Users Guide, pages 12, 148
CP/M Handbook, pages 86, 225
CP/M Dynamic Debugging Tool (DDT) User's Guide

Command Name: TOD

Operating System: MP/M

Function: The TOD (time of day) command allows the user to read and set the date and time.

To Display the Current Date and Time:

Syntax: OA> TOD <cr>

To Set the Date and Time:

Syntax: OA> TOD [date] [time] <cr>

References: MP/M Users Guide, pages 15, 79
CP/M Handbook, page 267

OPERATING SYSTEM AND UTILITIES

SCHED

Command Name: SCHED

Operating System: MP/M

Function: The SCHED (scheduler) command allows the user to schedule a program for execution. Entering 'SCHED' followed by a date, time and command line will cause the command line to be executed when the specified date and time is reached.

Syntax: OA> SCHED date time filename <cr>

References: MP/M Users Guide, page 16
CP/M Handbook, page 100

ABORT

Command Name: ABORT

Operating System: MP/M

Function: The ABORT command allows the user to abort a running program. The program to be aborted is entered as a parameter in the ABORT command.

Syntax: OA> ABORT processname <cr>

A program that is initiated from another console may only be aborted by including its console number as a parameter of the ABORT command. The console entry is as follows:

Syntax: OA> ABORT filename (console #) <cr>

OPERATING SYSTEM AND UTILITIES

SYSTEM UTILITIES

System Utilities is a collective name for the programs that provide various system operations:

- o Formatting disks - FDISK
- o Setting baud rates - BRATE
- o Converting absolute object files to a suitable format for downloading to a remote emulator/debugger - CONVERT
- o Downloading a program from the 9520 Software Development System to a remote emulator - DOWNLOAD
- o Uploading a program from a remote emulator to the 9520 Software Development System - UPLOAD
- o Transferring files from the 9520 to another device - HCOM

Each of the utility commands can be entered in either the immediate or interactive mode. On the following pages both modes are discussed for each utility.

Utility Name: FDISK

Function: The FDISK (floppy disk) utility allows a user to perform the following functions:

- o Format a floppy disk
- o Duplicate a floppy disk
- o Copy the system tracks from one disk onto another disk

NOTE: Under no circumstances should any copy of the FDISK utility be used on the hard disk system. The hard disk system has its own utility FLOPPYD to format and duplicate floppy disks. (See Chapter 10.)

The FDISK utility may be invoked using the interactive mode or the immediate mode.

To invoke the FDISK utility using the interactive mode:

Syntax: OA> FDISK <cr>

System Response: FDISK will prompt the user with the following menu:

F) ormat? D) uplicate? C) opy System? Q) uit?

The user should respond by typing the character F for formatting the disk, D for duplicating the disk, C for copying the system tracks, and Q for aborting the FDISK program.

Upon reception of a valid character, FDISK will prompt the user according to the response entered. For the F character response, FDISK will prompt with:

Disk to format - A, B, C, or D?

The user may choose the disk to format by striking the letter A for the disk in drive A, B for the disk in drive B.

FDISK will then respond with:

D) ouble Density? S) ingle Density?

OPERATING SYSTEM AND UTILITIES

FDISK (Cont'd)

The user may choose double density with the D character or single density with the S character. Upon receiving either one of these characters, FDISK formats the appropriate floppy disk and again responds with the prompt:

F) ormat? D) uplicate? C) opy System? Q) uit?

When the character D is chosen from the prompt issued by FDISK, the utility again prompts with:

Source disk - A, B, C or D?

The user may choose the disk to be duplicated by striking the letter A for the disk in drive A or B for the disk in drive B.

FDISK then prompts with:

Destination disk - A, B, C, or D?

The user may choose the new disk that receives the duplicated data by striking the letter A for the disk in drive A, or B for the disk in drive B.

FDISK then duplicates the destination disk with the contents of the source disk and responds with the menu prompt again. It is not necessary for the destination disk to have been formatted.

When the C character is chosen from the prompt issued by FDISK, the utility performs the same action as that taken for the duplicate function with one exception. Only the first two tracks of the source disk are duplicated onto the first two tracks of the destination disk.

To invoke the FDISK utility using the immediate command mode:

Syntax: OA> FDISK function flag #1 Flag#2 <cr>

where: Function means :

FORMAT
DUPE
CPYSYS

and FLAG #1 means:

DRIVE = for the FORMAT indicator.
FROM = for the DUPE and CPYSYS indicators.

and FLAG #2 means:

DENSE = for the FORMAT indicator.
TO = for the DUPE and CPYSYS indicators.

FDISK (Cont'd)

These parameters to FDISK cause the utility to execute without prompting the user, and may appear in any sequence.

There may be 6 variations of the command line for the FORMAT indicator. For example:

FDISK	FORMAT	DRIVE = B	DENSE = D
FDISK	FORMAT	DENSE = D	DRIVE = B
FDISK	DENSE = D	FORMAT	DRIVE = B
FDISK	DENSE = D	DRIVE = B	FORMAT
FDISK	DRIVE = B	FORMAT	DENSE = D

A, B, C or D characters must follow the equal symbol (=) for the DRIVE, FROM, and TO indicators and S, or D characters must follow the = sign for the DENSE indicator. The function names cannot be abbreviated and TO = and FROM = must be used with the DUPE and CPYSYS functions.

WARNING: Whenever the FDISK utility is used to FORMAT or DUPLICATE a disk, no other USER or PROCESS should be running on the system.

WARNING: Formatting or Duplicating will destroy the existing contents of the destination disk.

OPERATING SYSTEM AND UTILITIES

BRATE

Utility Name: BRATE

Function: The BRATE (baud rate) utility program allows the user to display and set baud rates for the 9520 Development System I/O devices. These devices include RS-422 I/O, Remote I/O, Reader/Punch I/O and Console I/O communication ports. The BRATE setting selected overrides the system initialization Baud Rate set by switch block S5, the Baud Rate switches on the back of the 9520.

The BRATE utility may be invoked using the interactive mode or the immediate mode.

To invoke the BRATE utility using the interactive mode:

Example: 0A> BRATE <cr>

System Response: The BRATE program then displays the current baud rate status for each I/O device and prompts the user as follows:

I/O device:

The user may respond with the ASCII description of the I/O device and enter a <cr> after each response:

```
RS449 <cr>
REMOTE <cr>
PRINTER <cr>
CONSOLE <cr>
```

The baud rate program will then prompts the user for the baud rate desired:

BAUD RATE

The user may respond by entering the following values for the baud rate.

BRATE (Cont'd)

where: valid baud rate values are:

110
134.5
150
300
600
1200
2400
4800
9600
19200
38400
56000
76800
187500
375000
750000

NOTE: An error message is displayed if a selected baud rate is invalid for a particular I/O device.

A control C character may be entered to abort the baud rate program at any time.

To invoke the BRATE Utility using the immediate mode:

All information is provided in the parameters of the command string, and immediate execution occurs to complete the processing.

Syntax: 0A> BRATE <I/O device> = <Baud Rate> <cr>

where: I/O device is the ASCII description for one of the following I/O ports:

RS449
REMOTE
PRINTER
CONSOLE

The equal symbol (=) is a delimiter

Baud Rate is one of the values shown in the preceding listing of valid baud rate values.

CONVERT

Command Name: CONVERT

Function: To enhance the universality of the 9520, Gould has installed a CONVERT utility program in the 9520 to run under the CP/M or MP/M operating system. The CONVERT utility enables the user to create a new file that will contain reformatted data from a specified file type. The following table displays the available conversions.

Table 5-1. Conversions

		T	O	
F		BIN	HEX	TEK
R	BIN	*	X	X
O	HEX	X	*	X
M	OBJ	X	X	X
	TEK	X	X	*

Legend: BIN = Millennium Binary Format Code
HEX = Intel Hexadecimal Format Code
TEK = Tektronix Hexadecimal Format Code
OBJ = Gould Cross Assembler Object File
X = Allowable Conversions
* = Not applicable

To invoke the Convert utility:

The CONVERT utility is invoked by entering the following instructions at the console terminal. The syntax is

```
CONVERT [d:] Fn1.Ft1 [d:] Fn2.Ft2 <cr>  
[drive:]input primary filename.file type extension
```

where:

[d:] = the disk drive A, B, C, or D. The default is the drive currently logged on.

Fn1 = the actual primary file name of the file that will contain the converted data.

Ft1, Ft2 = file type
one of the conversion extensions (e.g.,
.BIN, .HEX, .OBJ, or .TEK).
(The file type .OBJ can only be used for the input file.)

CONVERT (Cont'd)

Fn2 = the primary file name of the file containing the data to be converted.

space = delimiter. Also an equals sign "=" can be used as a delimiter between the two file-ids.

The Convert utility program flow.

1. Verify specified input and output file type extension.
2. Open specified input file ([d:]Fn2.Ft2), erase working output file ([d:]Fn1.\$\$\$), and open working output file ([d:]Fn1.\$\$\$).
3. Input data from specified input file ([d:]Fn2.Ft2), convert the data and output converted data to the working output file ([d:]Fn1.\$\$\$).
4. Close specified input file ([d:]Fn2.Ft2) and close working output file ([d:]Fn1.\$\$\$).
5. If errors or warnings occur during the conversion, proceed to step 6. If not, erase specified output file ([d:]Fn1.Ft1), rename working output file ([d:]Fn1.\$\$\$) to specified output file ([d:]Fn1.Ft1), output to the console "FILE CONVERTED, NO ERRORS", and return control to the operating system.
6. Output to the console "FILE CONVERTED, REMAINS ".\$\$\$"," and return control to the operating system.

In the event conversion is not accomplished without errors, the CONVERT utility causes (an) error message(s) to be displayed. There are two types: warning error messages and fatal error messages. In the event the user needs to abort the convert utility for any reason, the user must enter a control C at the keyboard.

Warning Error Messages. When the CONVERT utility encounters any impropriety in the input file that is not a fatal error, a warning message is displayed and file conversion continues. The warning messages, as they are displayed on the console terminal, are as follows:

1. ERROR: CHECKSUM DOES NOT MATCH
CHECKSUM IN FILE: XX, CHECKSUM COMPUTED: XX
RECORD LOAD ADDRESS: XXXX

(XX=Hexadecimal value) (XXXX=Hexadecimal address value)
2. ERROR: INPUT MODULE IS NOT ABSOLUTE, IT CONTAINS RELOCATION INFO
RECORD LOAD ADDRESS: XXXX
3. ERROR: INPUT FILE HAS SHORT BLOCK
RECORD LOAD ADDRESS: XXXX

OPERATING SYSTEM AND UTILITIES

CONVERT (cont'd)

4. ERROR: INPUT MODULE HAS UNRESOLVED EXTERNAL REFERENCE
RECORD LOAD ADDRESS: XXXX
5. ERROR: UNDEFINED ERROR
RECORD LOAD ADDRESS: XXXX
6. ERROR: INTEL RECORD MARK (":") EXPECTED,
FILE CONTAINS: XXX-----XXX
XXX-----XXX
RECORD LOAD ADDRESS: XXXX

NOTE: For ERROR 6, XXX-----XXX (to 80 column width) represents the data read from the file until the next record mark was encountered. When converting Millennium Binary files, unprintable binary data is represented by a period ("."). In addition, spaces between data are also represented by a period (".").

7. NOTICE: INTEL RECORD TYPE 02 (EXTENDED ADDRESS RECORD) ENCOUNTERED
RECORD LOAD ADDRESS: XXXX
8. WARNING: INTEL RECORD TYPE 02 CONTAINS RECORD LENGTH OF XX
RECORD LOAD ADDRESS: XXXX
9. WARNING: EOF RECORD TYPE CONTAINED A NON-ZERO RECORD LENGTH OF XX
RECORD LOAD ADDRESS: XXXX
10. NOTICE: INTEL RECORD TYPE 03 (START ADDRESS RECORD) ENCOUNTERED
RECORD LOAD ADDRESS: XXXX
11. WARNING: INTEL RECORD TYPE 03 CONTAINS A RECORD LENGTH OF XX
RECORD LOAD ADDRESS: XXXX
12. ERROR: UNEVEN NUMBER OF BYTES TO CONVERT TO BINARY
RECORD LOAD ADDRESS: XXXX
13. ERROR: BAD RECORD TYPE (MRLLDADRT)
RECORD LOAD ADDRESS: XXXX

where - M=record mark (":") =INTELHEX record mark)
RL=record length
LDAD=load address
RT=record type

14. ERROR: TEK RECORD MARK ("/" or " ") EXPECTED
FILE CONTAINS: XXX-----XXX
XXX-----XXX
RECORD LOAD ADDRESS: XXXX
(see ERROR 6 for definition of XXX----XXX for ERRORS 14,15,16,17.)

CONVERT (Cont'd)

15. ERROR: BAD EXTENDED-TEK RECORD TYPE: /XX
FILE CONTAINS: XX-----XXX
XXX-----XXX
RECORD LOAD ADDRESS XXXX
16. ERROR: BAD EXTENDED-TEK SEGMENT ADDRESS: BC0000RTSSSS
FILE CONTAINS: XXX-----XXX
XXX-----XXX
RECORD LOAD ADDRESS: XXXX

where - BC=byte count
0000=load address
RT=record type
SSSS=segment or offset address

17. ERROR: BAD EXTENDED-TEK OFFSET ADDRESS: BC0000RTSSSS
FILE CONTAINS: XXX-----XXX
XXX-----XXX
RECORD LOAD ADDRESS: XXXX
(BC0000RTSSSS-same as ERROR 16)
18. NOTICE: EOF RECORD PROCESSED, BUT INPUT FILE STILL CONTAINS
UNPROCESSED DATA
19. ERROR: EOF REACHED WITHOUT PROCESSING EOF RECORD

Fatal Error Messages. The following messages are output to the console display terminal when the CONVERT utility is aborted and program control is returned to the operating system.

1. ERROR: INVALID PARAMETER(S)
2. ERROR: INVALID INPUT FILE TYPE
3. ERROR: INVALID OUTPUT FILE TYPE

NOTE: ERROR 1, 2 or 3 is displayed when the .file-type-extension is not one of the previously mentioned file type extension(s), or when the file type extension(s) entered do not correspond to the types of conversions enumerated in the conversion table.

4. ERROR: OUTPUT ERROR
5. ERROR: CANNOT OPEN FILE
6. ERROR: INVALID BLOCK OR RECORD TYPE

OPERATING SYSTEM AND UTILITIES

CONVERT (cont'd)

7. ERROR: FILE NOT OPEN
8. ERROR: CANNOT CLOSE FILE
9. ERROR: CANNOT RENAME FILE

If fatal error messages 4 through 9 occur, the following entry finds the error locations:

```
Convert fn1.ft1 fn2.ft2 D <CR>
```

The system responds with the following display:

```
FATAL ERROR nn fn.ft record=rr  
(error message)
```

where: nn = the error number
Fn = the primary file name
Ft = the file type. For input errors, it will be the file type of input file. For output errors, it will be the file type of the output file.
rr = the physical record number in the File Control Block (FCB).

Examples. The following are examples of the use of the CONVERT utility program:

File type usage

BIN	A file of type BIN will download to the Gould 9508. Application software is provided that allows files of other types to be downloaded to a 9508.
TEK	Tektronix Hexadecimal Format
OBJ	The output of all Gould cross-assemblers is an object file of type OBJ.
HEX	Intel Hexadecimal Format

CONVERT (cont'd)

Examples:

1. Convert the output of a cross-assembler or MLINK to Millennium Binary format:

CONVERT Z80S10.BIN = Z80SER.OBJ

Now the file Z80S10.BIN can be downloaded to a 9508 MicroSystem Emulator.

2. Convert the output of a cross-assembler to Intel Hexadecimal format. This allows a COM or PRL file to be built from the output of a cross-assembler:

CONVERT TAPEHDLR.HEX = TAPEHDLR.OBJ

OPERATING SYSTEM AND UTILITIES

DOWNLOAD

Utility Name: DOWNLOAD

Function: This utility program downloads a program from a file on the 9520 Software Development System to the 9508 MicroSystem Emulator or 9516 Microsystem Integration Station so it can be debugged. The input file must be in Millennium Binary format. The 9508 or 9516 link port must be connected to a console port on the 9520.

The DOWNLOAD utility is invoked in the following manner:

Example: 0A> DOWNLOAD [d:] fn [.ft] cr

where:

d: is the disk drive letter. A, B, C, or D. The default is the current logged on drive.

fn: is the primary filename of the input file.

ft: is the file type extension. The default is BIN.

UPLOAD

Utility Name: UPLOAD

Function: The UPLOAD utility is used to upload a program from the 9508 MicroSystem Emulator or the 9516 Microsystem Integration Station to a disk file on the 9520 Software Development System. It assumes the link from the 9508 or 9516 is connected to a CONSOLE PORT on the 9520.

The syntax for the UPLOAD command is

UPLOAD [d:] fn [.ft] <cr>

where:

d: is the disk drive letter: A, B, C, or D. The default is the current logged-on disk.

fn: is the primary filename of the output file.

ft: is the file type extension. The default is BIN.

NOTE: This program uses: 1) Start-Synchronization Handshake
2) ACK/NAK Protocol

The format of the output file is always Millennium Binary.

OPERATING SYSTEM AND UTILITIES

HCOM

INTRODUCTION

The host communications utility (HCOM) for the 9520 Software Development System provides the capability to transfer ASCII files between the 9520 and other devices. It may be executed under either MP/M or CP/M. The transfer is made over the RS-232 link. HCOM responds to and manages the RS-232 control lines DTR and DSR. In addition, two software protocols (ACK/NAK and XON/XOFF) are supported.

The transferred files are composed of 7 or 8-bit ASCII characters in variable length records. Each record is normally terminated by a user-selectable end-of-record sequence. The entire file may be terminated by user selectable end-of-file sequence.

OVERVIEW

Several steps are required to prepare for and perform a file transfer:

1. The type of connection must be determined: to a terminal (console) port on another computer, or to a non-terminal port on another computer or device.
2. The physical link must be established. This requires knowledge of the signals originating from and expected by both the 9520 and the other device. A special cable or a breakout box may be needed to make the correct connections.
3. Each of the options controlling a file transfer must be considered and correctly selected. The option selections depend on the operating parameters of the interface, the structure of the transferred file, the protocol used, and the type of connection. The options may also depend on the direction of the file transfer.
4. After the options for a particular transfer have been selected, they may be saved in an option file and quickly loaded in a subsequent invocation of HCOM.
5. Files may now be transferred.

INVOKING HCOM

The host communications utility may be invoked in either of two forms:

```
OA>HCOM
```

or:

```
OA> HCOM file-id
```

HCOM

In the first form the HCOM options are initialized with their default values. In the second form the specified options file is loaded to initialize the HCOM options. The file-id may include a logical disk designation and a file type.

HCOM MENU

HCOM presents its menu. A menu item (display) may be selected by typing the appropriate letter and pressing the RETURN key. The console screen is then filled with the selected display. Some displays are purely informational; others control options. Those that control options offer information about the options as well as indicate their current values.

The operator may alter the values of the options in response to the display's queries. Entering a blank line in response to any query leaves the value of the option unaltered. The menu is presented again after the operator has responded to all queries in a display. Other displays control file actions. In these displays, a blank line immediately returns the operator to the menu. The remaining menu items allow the operator to request the menu to be presented again, or to exit HCOM.

An example of the menu and display selection is shown below.

File Transfer Utility - Asynchronous Communications Interface - Version 1.0

Menu:

- A General Information regarding protocol
- B RS-232 link information
- C DTR/DSR information
- D Interface parameters options
- E End-of-record sequence option
- F End-of-file sequence option
- G Abort sequence option
- H ACK/NAK options
- I XON/XOFF option
- J Terminal mode options
- K Save or load selected options to or from a file
- L Transfer files from interface to disk
- M Transfer files from interface to disk
- N Display this menu
- O Exit file transfer utility

In response to any query, the new options (blank or comma separated) must be entered with a carriage return. A blank line leaves the options unaltered. Striking CTL-C as the first character returns control to the operating system.

Enter desired menu item letter (Q <letter> - skips informative text): A <cr>

OPERATING SYSTEM AND UTILITIES

HCOM

HCOM Display

If the operator selects A for the desired menu item, the following display is presented:

General Information Regarding Protocol

The File Transfer Utility is used to transfer ASCII files between the 9520 and other devices over an RS232 type link. The other device may be a computer, a PROM programmer or anything else capable of meeting the interface and protocol requirements.

Data is transferred in variable length records terminated by an operator selectable, end-of-record sequence. The maximum record length is 255 characters (excluding the end-of-record sequence).

Various methods of synchronizing and acknowledging record transfers are available (DTR/DSR, ACK/NAK, and XON/XOFF). Any or all of these protocols may be selected, but at sufficiently low transfer rates (2400 or less) none are required. The DTR/DSR protocol is always selected, but may be disregarded provided certain hardware considerations are met. (See DTR/DSR Information.)

File transfers are terminated normally by an operator selectable, end-of-file sequence. An abnormal termination is signalled by the transfer of an operator selectable, abort sequence.

Enter blank line to return to menu: <cr>

Quiet Option

If the operator wishes to suppress the informative text of a display, the quiet option should be used. This is done by typing an upper or lower case "q" followed by a blank or comma and then the desired menu item letter. If the selected display controls options, their current values are presented before the display's queries. The menu is not presented again on return from a display selected with the quiet option. Only a reminder of how to request the menu is displayed.

Operating Modes

HCOM operates in three modes. It is in entry mode whenever it is waiting for the operator to type something in response to a query. The operating system provides a means to abort HCOM, while in entry mode, by typing control-C as the first character of a response.

A Control-T entered as the first character of a response puts HCOM in terminal mode after the RETURN key is pressed. When terminal mode is exited the remainder of the line (following the control-T) is taken as the response to the query, unless it is blank; in which case, the query is repeated.

Finally, HCOM is in transfer mode when a file transfer is taking place.

By stepping through the menu the user can complete all the steps necessary to transfer files. The remainder of this section is organized in the order of the menu displays. The displays are not be shown in this text because the information displayed would be redundant.

STEP 1: INITIAL CONNECTION DECISION

The first decision facing the user is how the 9520 is to be connected to the other device. The other device may treat the 9520 either as a terminal or a non-terminal. Most computer system provide utilities to display ASCII text files at a terminal and accept input from a terminal to be stored in a file. The 9520 can be connected to these systems as a terminal and make use of these utilities to transfer files. Also, some computer systems provide special utilities that support communications through alternate I/O ports. The 9520 can be connected to these as a non-terminal. And of course, other equipment may be terminal peripherals themselves. With these, the 9520 must not be considered a terminal.

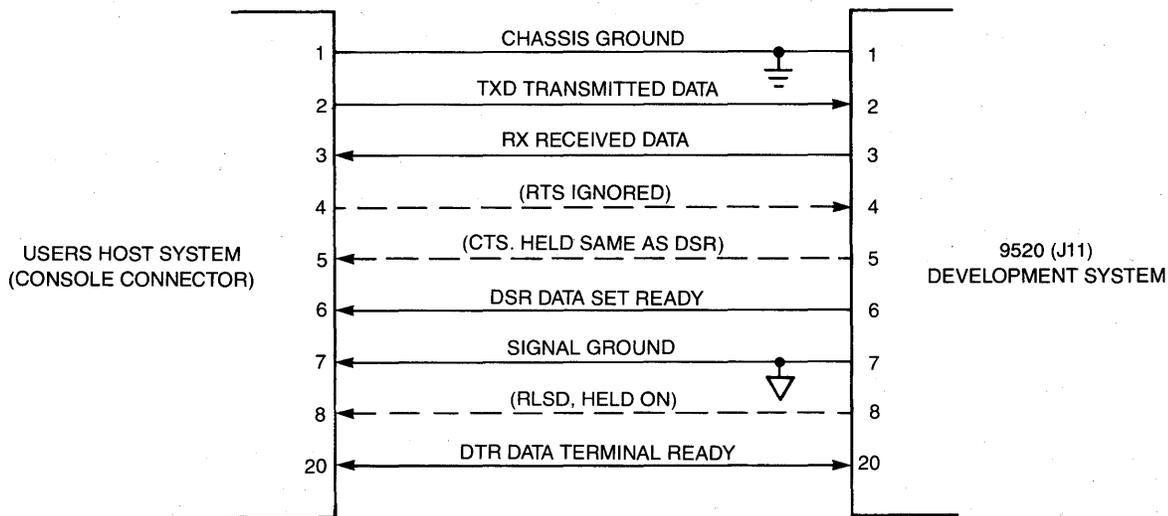
OPERATING SYSTEM AND UTILITIES

HCOM

ESTABLISHING THE PHYSICAL LINK

RS-232 link information (Display B)

The interface to the 9520 involves an RS-232 link. HCOM handles its port in a fashion similar to Data Communications Equipment (DCE). The ability of the other device to manage and respond to these lines must be determined. It may be necessary to make a special cable or use a breakout box, to cross connect certain lines. The pins connected, the lines assigned to them, the directions of the signals and their uses are indicated below:



9520-016

Figure 5-3. Host to 9520 Communications Link Interface

DTR/DSR Information (Display C)

The interface DTR line controls transmission from the 9520. The DSR line controls transmission from the other device. Of the other two commonly used control signals, RTS is ignored and CTS is held the same as DSR.

The 9520 will not transmit unless the other device indicates it is ready to receive by holding the DTR line in an active state. Similarly, the 9520 will hold the DSR line in an active state only when it is ready to receive. Thus, these lines provide data flow control if the other device is capable of managing and responding to them.

If the other device cannot manage and respond to these lines, flow control can be accomplished with the XON/XOFF protocol. If neither XON/XOFF nor DTR/DSR can be used for flow control, then the ACK/NAK protocol can be used for flow control as well as for error retransmission. Furthermore, the DTR line must be held in an active state. This is done automatically in the 9520 hardware if the other device is not connected to the DTR line. If the other device is connected to the DTR line, that device must be responsible for holding the line in an active state.

STEP 3: OPTIONS SELECTION

Having established the physical link, the operator must consider and select each of the options controlling a file transfer. The option selections depend on the operating parameters of the interface, the structure of the transferred file, the protocol used, and the type of connection. The options may also depend on the direction of the file transfer.

Interface Parameters Options (Display D)

It is necessary to inform HCOM of the interface parameters. There are four: the size of each data character (either 7 or 8-bit ASCII), parity, the number of stop bits transmitted with each character, and the transmission rate (in bits/second). In response to the query, the user may enter a list of up to four letters, separated by blanks (spaces) or a comma. The letters correspond to one of the options selected for each parameter and may be typed in any order. Provided there is no entry error, any new options entered are selected and the user is returned to the menu when the RETURN key is pressed.

End-of Record Sequence Option (Display E)

The ASCII character data transferred by HCOM is organized into variable length records terminated by an end-of-record sequence. The ASCII text files stored on the 9520 disks have variable length records terminated by a carriage-return/line-feed sequence. The purpose of the end-of-record sequence option is to provide the translation between the sequence used on the 9520 file system and that being generated or expected by the other device. In some cases it may be necessary to have an empty end-of-record sequence: with fixed length records, for example, or with variable length records that contain a count rather than a terminator. Such files may not be acceptable to many of the 9520 standard utilities, but HCOM may be able to transfer them for special purposes.

There may be up to 16 characters in the end-of-record sequence. Each is entered, in order and blank or comma separated, as a two digit hexadecimal number representing its ASCII codes. An empty sequence is selected by entering "N" or "n" alone on the line.

The HCOM record buffer may contain up to 255 characters. If the record read from disk fills the buffer before a carriage-return/line-feed is encountered, the record is sent to the other device without an end-of-record sequence. Likewise, if the buffer fills with characters from the other device before an end-of-record sequence is detected, the record is written to the 9520 file without a carriage-return/line-feed. This occurs when the end-of-record sequence is empty, unless the entire file is less than 255 characters.

OPERATING SYSTEM AND UTILITIES

HCOM

If the last record received by HCOM is not properly terminated with an end-of-record sequence, it is discarded. This action ensures that when the 9520 is connected as a terminal, the prompt that is normally sent to a terminal after a transfer is not written to the 9520 file. Also when the end-of-record sequence is empty, a last partial record containing valid data may be discarded. In this case, loss of data may be avoided by having the other device send 254 characters to ensure all valid data is written to the file. A text editor on the 9520 may then be used to remove any bad characters that may be in the file.

In Display E, the operator may enter a new end-of-record sequence in hex bytes.

End-of-File Sequence Option (Display F)

Normally, the end of the transfer is signalled by an end-of-file sequence. The ASCII text files stored on the 9520 disks use SUB (1A hex) for this purpose, when the last block of a file is a partial block. The purpose of Display F is to provide the translation between the character used on the 9520 file system and whatever sequence is being generated or expected by the other device. In some cases, it may be best or even necessary to have an empty end-of-file sequence.

The end-of-file sequence may contain up to 16 characters. It is entered in the same manner as the end-of-record sequence.

Abort Sequence Option (Display G)

An abort sequence may be received from the other device, or sent to it if the operator forces an abort from the 9520 console. When an abort occurs, HCOM immediately halts the transfer and discards any characters subsequently received from the other device. If the transfer is from the other device, the portion of the transfer received prior to the abort is written to the file.

The abort sequence may contain up to 16 characters. It is entered in the same manner as the end-of-record sequence.

ACK/NAK Options (Display H)

The ACK/NAK software protocol requires the receiver to respond, when a record is sent to it, with either a positive (ACK) or negative (NAK) acknowledgement. The sequences used for acknowledgements are operator selectable. If this protocol option is selected, HCOM expects the appropriate sequences when sending and generates them when receiving. When sending, HCOM sends a record and then waits until an acknowledgement is received. If it is positive, then HCOM sends the next record. If it is negative, then HCOM sends the same record again. When receiving, HCOM sends an acknowledgement when it receives a record. If there were no errors in the record (e.g., parity or overrun), then a positive acknowledgement is sent. If any errors were detected, a negative acknowledgement is sent.

Although the primary purpose of this protocol is for error correction using retransmission of a word, it can also be used for flow control on a record-by-record basis. Since the sender cannot send another record until it receives a positive or negative acknowledgment for the last record, the receiver can dispose of the data received and prepare for the next record before sending a positive acknowledgment.

A record is normally terminated by an end-of-record sequence. However, for the purposes of the ACK/NAK protocol, a full record buffer (255 characters) is also considered a record. Therefore, ACK/NAK should only be used with a non-empty end-of-record sequence; and none of the transferred records should contain more than 254 characters, excluding the end-of-record sequence. Display H allows the user to enable the ACK/NAK option.

XON/XOFF Option (Display I)

The XON/XOFF software protocol allows the receiver to suspend and resume the transfer. When the HCOM receive buffer becomes nearly full, it sends an XOFF character (13 hex) to the other device as a signal that the other device should stop sending. While the transfer is thus suspended, HCOM continues processing the characters it has already received. Then, when the receive buffer becomes nearly empty, it sends an XON character (11 hex) to inform the other device it may resume sending. HCOM also responds to incoming XOFF and XON characters when this protocol option is selected. When it receives an XOFF, it sends at most two more characters before stopping, and sends no more until after it receives an XON. Display I allows the user to enable the XON/XOFF option.

Terminal Mode Option (Display J)

The terminal mode options are selected to control HCOM actions when the 9520 is connected to the other device as a terminal. These options are only in effect when HCOM is in the terminal mode. They have no effect during transfer mode. This menu allows the user to enable the automatic terminal mode, choose terminal mode operation (half or full duplex), select the receive/echo filter list, select the terminal mode exit character and choose a transmitted exit sequence.

When the automatic terminal mode option is enabled, HCOM immediately enters terminal mode after a transfer file name is entered. Then when terminal mode is exited, HCOM is ready to begin the transfer immediately.

This option is useful when HCOM must be prepared to respond very quickly after leaving terminal mode. The terminal mode operation option determines whether or not HCOM echoes input characters locally while in terminal mode. It will echo locally when HALF-DUPLEX is selected. When FULL-DUPLEX is selected, HCOM assumes that the other device will echo all characters sent to it.

OPERATING SYSTEM AND UTILITIES

HCOM

The "receive/echo filter" is a list of up to 32 characters that are not sent to the 9520 console during terminal mode. This is important because many characters, notably control characters (01-1F hex), can adversely alter the state of the 9520 console. If any of the characters in the list is received, it is filtered out. The "filter" also affects echo characters, both locally echoed and those echoed by the other device. Thus, it is possible to send control characters to the other device during terminal mode without affecting the 9520 console.

The character used to exit terminal mode is operator selectable. It should be chosen so as not to conflict with characters that must be sent to the other device during terminal mode. When this character is typed, HCOM returns to the mode it was in (either entry or transfer mode) when terminal mode was entered. The exit character is not sent to the other device. If it is necessary to send characters to the other device when terminal mode is exited, the characters should be specified in the transmitted exit sequence, which may contain up to 16 characters. When the selected exit character is typed, the transmitted exit sequence is sent to the other device just before terminal mode is exited.

STEP 4: SAVING AND LOADING OPTION FILES

When the options have been correctly selected for a certain type of transfer, they may be saved in a file on the 9520 disks. When HCOM is used for another transfer of the same type (i.e., to or from a particular device), the file may be loaded and options will not have to be selected again. Options may be loaded by using Display K or by specifying them in the HCOM invocation line.

Save or Load Selected Options To or From a File (Display K)

All of the options selected (interface parameters, end-of-record sequence, end-of-file sequence, abort sequence, ACK/NAK, XON/XOFF and terminal mode) may be saved in a file that can later be loaded in the HCOM utility.

An options file can be saved by responding to the queries in Display K. The user can save and/or load this file by specifying its name.

STEP 5: TRANSFERRING DATA FILES

HCOM is in transfer mode when it is transferring files either to or from the other device. As each logical record is detected by HCOM, a period is written to the 9520 console. After a multiple of 64 periods has been written, the total number is also displayed in decimal.

HCOM

When the transfer is terminated, a message is written to the 9520 console. This message contains the name of the 9520 file, the type of termination (either complete or aborted), and the number of characters and disk records in the 9520 file. Another transfer may then be performed.

Error Messages

In addition, during a transfer, error messages may be displayed. Before any error message is written, the decimal number of logical records is written. Therefore, the number of the record in which the error occurred is one greater than the decimal number preceding the message.

There are three types of errors:

1. Character errors can occur on each character transferred. These are, for example, errors in parity detection, framing errors, overrun errors and interrupt buffer overflow. The number of the character, within the record, on which the error occurred is also displayed with the error message.
2. Record errors occur in conjunction with a complete logical record. For example, the reception of an abort sequence, is a record error.
3. Console overrides, while not strictly errors, are displayed in the same manner as errors.

A console override is a character struck at the 9520 console while HCOM is in transfer mode. All but three characters are ignored:

1. A control-T places HCOM in terminal mode. If the other device is in the process of transferring necessary data to the 9520, this data is sent to the 9520 console rather than written to the file; data is lost. When terminal mode is exited, HCOM resumes the transfer.
2. An escape struck while in transfer mode aborts the transfer. If a non-empty abort sequence has been selected, it is immediately sent to the other device. No subsequent data is sent by HCOM, and all subsequently received data is discarded. The 9520 file is closed. If the transfer was from the other device, the data transferred prior to the abort will be in the file.
3. A control-Z from the console forces HCOM to act as if it had detected the end of the file. If the transfer is to the other device, the end-of-file sequence is sent, provided it is non-empty; no other data is sent. If the transfer is from the other device, any further data is discarded. The 9520 file is closed. If the transfer was from the other device, the data transferred prior to the forced end of file will be in the file.

OPERATING SYSTEM AND UTILITIES

HCOM

Transfer Files from Disk to Interface (Display L)

When this menu display is invoked, the operator is queried for the name of a file. This is the file on the 9520 disk system that is read and sent to the other device. A blank line in response to this query causes HCOM to return to the menu. Only one file may be sent at a time; but after each transfer, the operator is queried for another.

When a file name is entered, transfer mode is entered, the file is opened, and records are read from it and transferred to the interface. A period is written to the console after each record is successfully transferred. Periodically, the number of records that have been read is displayed in decimal.

The transfer is normally terminated when the end of the file is encountered. The end-of-file sequence is then transferred to the interface, and the file is closed. The operator may force this condition by striking CTL-Z (1A hex) at the 9520 console; any part of the file remaining will not be transferred.

An abnormal termination occurs when the abort sequence is detected from the interface or when the operator strikes ESC (1B hex) at the 9520 console. This halts the file transfer; and in the latter case, the abort sequence is transferred to the interface, as well. Operator intervention at the other device might be required if it is unable to respond to the abort sequence.

Transfer Files from Interface to Disk (Display M)

In this display, the operator is queried for the name of a file. This is the file created on the 9520 disk system and to which the data received from the other device are written. When a file name is entered, transfer mode is entered, the file is created, and records are transferred from the interface and written to that file. A period is written to the console after each record is successfully transferred. Periodically, the number of records transferred is displayed in decimal.

Termination of this type of transfer is exactly the same as when transferring files from disk to interface.

Display this Menu (Display N)

If the operator selects this display, the entire File Transfer Utility Menu is displayed.

Exit File Transfer Utility (Display 0)

Selecting "0" exits the file transfer Utility and the 9520 returns with the prompt A>.

EXAMPLES

The following two examples give a detailed description of the use of HCOM. Each example is presented in a step by step manner, as the user might step through the displays that have just been listed.

Mention of specific products not produced or sold by Gould, should not be construed as a recommendation of their merits over other similar products. They have been chosen as examples of actual transfers merely to illustrate different types of HCOM usage, and how a user, faced with a similar task, might proceed.

Example #1 INTELLEC SERIES 11, Model 230, running ISIS-11, V3.4

In this example, an Intel hex format file was transferred from the Intellec to the 9520. Intel hex format is a means of specifying object files using ASCII characters. Each record contains an introductory character, a record length, an address, a record type, memory data and a checksum. All items except the introductory character are represented in hexadecimal digits. The 9520 has utilities for manipulating and executing Intel hex format files.

Initial Connection Decision. The Intellec had an alternate port (TTY) that was connected to a line printer. This printer utilized the type of RS-232 communications required by HCOM. The user learned that the Intellec had utilities for copying ASCII text files to its TTY port. Since the intended transfer was only from the Intellec to the 9520, these capabilities were deemed sufficient. The 9520 was then connected as a non-terminal to the TTY port on the Intellec.

RS-232 Link Information. The Intellec used the line printer attached to its TTY port in the same manner as the 9520 used that printer. The user concluded the Intellec TTY port was performing like DCE, as does the printer port on the 9520. To connect the 9520 to the TTY port, an RS-232 breakout box was used to connect the 9520 TXD pin to the Intellec RXD pin. Since the data were only transferred from the Intellec to the 9520, the other data line was unnecessary and not connected. The user examined the cable connecting the Intellec TTY port to the line printer and learned that the printer DTR pin was connected to the Intellec CTS pin. Since the printer managed the DTR line correctly, the 9520 DSR pin was connected to the Intellec CTS pin. Signal ground was also connected. The connections used were:

<u>9520 pin</u>	to	<u>Intellec pin</u>
(TXD) 2	- - - -	3 (RXD)
(DSR) 6	- - - -	5 (CTS)
7	- common -	7

OPERATING SYSTEM AND UTILITIES

HCOM

DTR/DSR Information. The Intellec TTY port correctly responded to a control line signal. For this reason, the 9520 DSR pin was connected as explained previously. Since the data were only transferred from the Intellec to the 9520, the other control line (9520 DTR to Intellec RTS) was unnecessary and therefore not connected.

Interface Parameters Options. Three of the Intellec interface parameters had to be: 8 bit data size, no parity, and one stop bit. The rate for the printer was 9600 baud. The Intellec TTY port was set to this rate, however, by a special program. The user decided instead to use the port's default rate for the Intel hex format file transfer. The name of the TTY port suggested that its default rate might be 110 baud. This rate was tried and it worked. Since only the rate differed from the default initialization parameters, its new selection was entered. The other options were left unaltered.

Interface parameters options:

Size: B - 8 bits Parity: D - None Stop Bits: H - 1 Rate: P - 9600

Enter new options: J <cr>

End-of-Record Sequence Option. The Intellec sent a carriage-return/line-feed sequence at the end of each line. Since this is the default initialization end of record sequence, no new sequence was entered.

End of record sequence: 0D 0A

Enter new end-of-record sequence

(hex bytes or N if none): <cr>

End-of-File Sequence Option. The Intellec sent no characters to signal the end of a file that had been copied to its TTY port. The user also knew the default initialization end-of-file sequence (1A hex) was unlikely to occur in the file to be transferred. If the default end-of-file sequence had occurred in the file, it could have caused HCOM to close the file prematurely, thereby losing data. In order to prevent this, the user selected an empty end-of-file sequence.

End-of-file sequence: 1A

Enter new end-of-file sequence

(hex bytes or N if none): N <cr>

HCOM

Abort Sequence Option. The Intellec would not respond to an abort sequence received by its TTY port, nor would it send such a sequence to signal a problem in sending the file. Therefore, the user selected an empty abort sequence.

Abort sequence: 1B

Enter new abort sequence
(hex bytes of N if none): N <cr>

ACK/NAK Options. Since the Intellec was capable of responding to a control line protocol, no software protocol was needed. The ACK/NAK protocol was left in its default (disabled) initialization state.

ACK/NAK option: DISABLED
ACK sequence: 06
NAK sequence: 15

Enable or disable ACK/NAK option (E or D): <cr>
Enter new ACK sequence
(hex bytes): <cr>
Enter new NAK sequence
(hex bytes): <cr>

XON/XOFF Option. Since the Intellec was capable of responding to a control line protocol, no software protocol was needed. The XON/XOFF protocol was left in its default (disabled) initialization state.

XON/XOFF option: DISABLED
Enable or disable option (E or D): <cr>

Terminal Mode Options. Since the 9520 was not connected to the Intellec as a terminal, none of the terminal mode options was altered from its default initialization state.

Automatic terminal mode option: DISABLED
Terminal mode operation: FULL-DUPLEX
"Receive/echo filter" list: NONE
Terminal mode exit character: 1C
Transmitted exit sequence: NONE

Enable or disable automatic terminal mode option (E or D): <cr>
Enter new terminal mode operation (F or H): <cr>
Enter new "receive/echo filter" list
(hex bytes or N if none): <cr>
Enter new exit character (hex byte): <cr>
Enter new transmitted exit sequence
(hex bytes or N if none): <cr>

OPERATING SYSTEM AND UTILITIES

HCOM

Saving and Loading Options Files. After the options were selected, they were saved in an options file.

Save or load selected options to or from a file

(S or L or blank line to return to menu): S <cr>

Enter name of options file to be SAVED (blank line will return to menu without performing action): FROMINT.OPT <cr>

Transfer Files from Disk to Interface. Since the transfer was only from the Intellec to the 9520, this display was not used in this example.

Transfer Files from Interface to Disk. The transfer of the Intel hex format file from the Intellec to the 9520 was performed. First, display M was selected and the name of the file to be created on the 9520 was entered in response to the transfer query. Then the following command was entered at the Intellec console:

```
-COPY :F1:INTEL.HEX TO :T0:
```

This command caused the named file to be copied to the Intellec TTY port (and hence, to the 9520). When the copy completion message had been written at the Intellec console, the user knew that the transfer was complete. At that time a control-Z was typed at the 9520 console to inform HCOM that no more data would be sent, (i.e., forced an end-of-file condition since no end-of-file sequence was sent by the Intellec.)

Enter name of file to be transferred from interface (blank line will return to menu without performing action): INTEL.HEX <cr>

FTU STATUS: Transfer Mode

```
.....64
.....128
.....192
.....256
.....320
.....384
.....448
.....512
.....576
.....640
.....704
.....768
.....832
.....892
```

CONSOLE OVERRIDE: End-of-file forced

INTEL.HEX transfer complete - 39356 bytes transferred - 308 disk records

FTU STATUS: Entry Mode

Example #2 UNIX Time Sharing System

In this example, text files were transferred in both directions between a computer running the UNIX system and the 9520. Communication with the UNIX system was through a modem connection over a dial-up telephone line.

Initial Connection Decision. The only available connection to the UNIX system was through a modem. The user learned that utilities existed for both writing a text file to a terminal and accepting input from a terminal to be stored in a text file. Therefore, text file transfers both to and from the UNIX system were possible. The 9520 was necessarily connected to the modem as a terminal.

RS-232 Link Information. The modem performed like DCE. A breakout box was used to cross connect the data lines. The modem required an active DTR signal from the connected terminal. The 9520 DSR pin was connected to the modem DTR pin for this purpose. The modem was also found to hold its CTS pin active when it was ready. This pin was connected to the 9520 DTR pin. The connections used were:

<u>9520 pin</u>		<u>Modem pin</u>
(TXD) 2	- - - -	3 (RXD)
(RXD) 3	- - - -	2 (TXD)
(DSR) 6	- - - -	20 (DTR)
7	- common -	7
(DTR) 20	- - - -	5 (CTS)

DTR/DSR Information. Since the transfer speed and synchronization of the UNIX system could not be managed with the control lines over a modem connection, the control line connections were only required for correct operation of the modem.

Interface Parameters Options. The interface parameters options for the UNIX system were determined from the settings of the terminal being used. They were found to be: 8 bit data size, no parity, one stop bit, and a 1200 baud rate. Only the rate differed from the default values.

Interface parameters options:

Size: B - 8 bits Parity: D - None Stop Bits: H - 1 Rate: P - 9600

Enter new options: M <cr>

OPERATING SYSTEM AND UTILITIES

HCOM

End-of-Record Sequence Option. The utility that accepted characters from a UNIX system terminal and stored them in a text file expected a carriage-return after each line. That character was selected as the end-of-record sequence for the text file transfer from the 9520 to the UNIX system.

End-of-record sequence: 0D 0A

Enter new end-of-record sequence
(hex bytes or N if none): D <cr>

The utility for writing text files to a UNIX system terminal sent a carriage-return/line-feed after each line. Those two characters were selected as the end-of-record sequence for the text file transfer from the UNIX system to the 9520.

End-of-record sequence: 0D

Enter new end-of-record sequence
(hex bytes or N if none): D A <cr>

End-of-File Sequence Option. For the text file transfer from the 9520 to the UNIX system, the UNIX system utility expected to receive an EOT (04 hex) to signal the end of input from a terminal. That character was selected as the end-of-file-sequence.

End-of-file sequence: 1A

Enter new end-of-file sequence
(hex bytes or N if none): 4 <cr>

For the text file transfer from the UNIX to the 9520, the UNIX system utility was incapable of sending a sequence to signal the end of the transfer. When the options were being selected for this transfer, an empty end-of-file sequence was selected.

End-of-file sequence: 04

Enter new end-of-file sequence
(hex bytes or N if none): N <cr>

HCOM

Abort Sequence Option. The UNIX system would respond to an ETX (03 hex) at any time by aborting the currently executing program, but would send no abort sequence. An ETX was selected as the abort sequence for the test file transfers in both directions.

Abort Sequence: 1B

Enter new abort sequence
(hex bytes of N if none): 3 <cr>

ACK/NAK Options. The ACK/NAK protocol was left in its default (disabled) initialization state for the text file transfers in both directions.

ACK/NAK option: DISABLED
ACK sequence: 06
NAK sequence: 15

Enable or disable ACK/NAK option (E or D): <cr>
Enter new ACK sequence
(hex bytes): <cr>
Enter new NAK sequence
(hex bytes): <cr>

XON/XOFF Option. The UNIX system used an XON/XOFF protocol for communicating with terminals. Since the 9520 was connected as a terminal, the XON/XOFF protocol was enabled for the text file transfers in both directions.

XON/XOFF option: DISABLED

Enabled or disabled option (E or D): E <cr>

Terminal Mode Options. For the text file transfer from the 9520 to the UNIX system, HCOM would not need to respond quickly. The automatic terminal option was left in its default (disabled) initialization state. The UNIX system operated with full-duplex terminals so the operation option was also left with its default initialization. The UNIX would send no characters that would adversely affect the 9520 console while HCOM was being used in terminal mode. Therefore, the receive/echo filter list was left empty. An EOT (04 hex) was selected as the terminal mode exit character since this was more like the UNIX system usage than the default exit character. Finally, the transmitted exit sequence was left empty since none was needed for this transfer.

OPERATING SYSTEM AND UTILITIES

HCOM

Automatic terminal mode option: DISABLED
Terminal mode operation: FULL-DUPLEX
"Receive/echo filter" list: NONE
Terminal mode exit character: 1C
Transmitted exit sequence: NONE

Enable or disable automatic terminal mode option (E or D): <cr>
Enter new terminal mode operation (F or H): <cr>
Enter new "receive/echo filter" list
(hex bytes or N if none): <cr>
Enter new exit character (hex byte): 4 <cr>
Enter new transmitted exit sequence
(hex bytes or N if none): <cr>

HCOM would have to respond quickly when the UNIX system utility was invoked to transfer the text file to the 9520. Therefore, the automatic terminal mode option was enabled. The next three options were left as they had been previously selected. A carriage-return was selected for the transmitted exit sequence option. When the UNIX system utility invocation line was typed, the exit character (EOT) caused, nearly simultaneously, the execution of the utility and the exit from terminal mode. In this manner, HCOM would be ready quickly to receive the first characters sent from the UNIX system.

Automatic terminal mode option: DISABLED
Terminal mode operation: FULL-DUPLEX
"Receive/echo filter" list: NONE
Terminal mode exit character: 04
Transmitted exit sequence: NONE

Enable or disable automatic terminal mode option (E or D): E <cr>
Enter new terminal mode operation (F or H): <cr>
Enter new "receive/echo filter" list
(hex bytes or N if none): <cr>
Enter new exit character (hex byte): <cr>
Enter new transmitted exit sequence
(hex bytes of N if none): 0D <cr>

Saving and Loading Options Files. The options for the text file transfer from the 9520 to the UNIX system were selected and then saved in an options file.

Save or load selected options to or from a file
(S or L or blank line to return to menu): S <cr>
Enter name of options file to be SAVED (blank line will return to menu
without performing action): TOUNIX.OPT <cr>

HCOM

After the previous options had been saved, the options for the text file transfer from the UNIX system to the 9520 were selected and then saved in another options file.

Save or load selected options to or from a file
(S or L or blank line to return to menu): S <cr>
Enter name of options file to be SAVED (blank line will return to menu
without performing action): FROMUNIX.OPT <cr>

Transfer Files from Disk to Interface. The text file transfer from the UNIX system was performed before this transfer. After editing that file to remove the spurious blank line, HCOM was invoked with the command:

OA>HCOM TOUNIX.OPT

This invocation initialized the options as they had been previously selected for the text file transfer from the 9520 to the UNIX system. Display L was selected and a control-T followed by a carriage-return was entered in response to the transfer query. This placed HCOM in terminal mode. Next, a carriage return was struck to obtain the UNIX system prompt. Assured by the prompt that HCOM was communicating properly with the UNIX system, the command line for the UNIX system utility to accept input from the terminal (9520) and store it in a file was typed and entered with a carriage-return. After waiting a few seconds to ensure that the utility was ready to accept characters, terminal mode was exited by striking the selected exit character (control-D). The transfer was then performed with the same file that had been previously transferred from the UNIX system so it could be compared with the original.

Enter the name of file to be transferred to interface (blank line will return to menu without performing action): < T> <cr>
FTU STATUS: Terminal Mode

>cp /dev/tty textfile

FTU STATUS: Entry Mode
Enter name of file to be transferred to interface (blank line will return to menu without performing action): TEXTFILE <cr>
FTU STATUS: Transfer Mode

.....64
.....78

TEXTFILE. transfer complete - 3040 bytes transferred - 234 disk records
FTU STATUS: Entry Mode

Enter name of file to be transferred to interface (blank line will return to menu without performing action): <cr>

OPERATING SYSTEM AND UTILITIES

HCOM

Transfer Files from Interface to Disk. The options were correct for the transfer of a text file from the UNIX system to the 9520, and this transfer was performed first. Display M was selected and the name of the file to be created on the 9520 system was entered in response to the transfer query. HCOM replied by automatically entering terminal mode. Next, a carriage return was struck to get the UNIX system prompt. Assured by the prompt that HCOM was communicating properly with the UNIX system, the command line for the UNIX system utility that wrote text files to a terminal was typed. Instead of terminating the command line with a carriage return, however, the exit character (control-D) was struck. This caused the transmitted exit sequence (carriage-return) to be sent to the UNIX system, thereby causing the command line to be executed. At the same time, terminal mode was exited and HCOM was immediately ready to accept characters from the UNIX system. When it was determined that the transfer was finished, a control-Z was struck at the 9520 console. It was necessary to force an end-of-file condition because the UNIX system utility used sent no character or sequence of characters to signal the end of the transfer.

```
Enter name of file to be transferred from interface (blank line will return
to menu without performing action): TEXTFILE <cr>
FTU STATUS: Terminal Mode
```

```
>cp textfile /dev/tty
```

```
FTU STATUS: Transfer Mode
```

```
.....64
.....79
```

```
CONSOLE OVERRIDE: End-of-file forced
```

```
TEXTFILE. transfer complete - 3042 bytes transferred - 24 disk records
```

```
FTU STATUS: Entry Mode
```

```
Enter name of file to be transferred from interface (blank line will return
to menu without performing action): <cr>
```

Because the transmitted exit sequence (carriage-return) had been echoed by the UNIX system (as carriage-return/line-feed) after HCOM exited terminal mode, the file on the 9520 contained a first blank line that the file on the UNIX system did not contain. HCOM was exited and a text editor was used to remove the blank line from the 9520 file to make it identical to the UNIX system file.

TEXT EDITOR, DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER UTILITY PROGRAMS

TEXT EDITOR UTILITY PROGRAM

The text editor utility for the 9520 Software Development System is WordStar. In conjunction with the operating system (CP/M or MP/M), WordStar provides the user with a versatile text editor utility for developing, editing, and storing programs.

To invoke WordStar, the operating system must be installed (booted) into the 9520. The operating system indicates its readiness by displaying the prompt A> or B> on the display terminal. The prompt is given after "booting" at system turn-on, after exiting from WordStar, or after completing an operating system command.

WordStar Key Functions

WordStar is customized by Gould to provide the user of the 9501 terminal a more convenient interface with WordStar. The 9501 keytops are changed to reflect this improvement. The WordStar functions are designated by the logos on the fronts of the keys. Functions that formerly required the depression of several keys can now be accessed by pressing a single key. Because these functions can still be accessed by depression of two or three keys, the multi-key method of access is shown in parentheses (with the CONTROL key abbreviated as CNTL) in the key function listing that follows:

Key #1		No function.
Key #2	WD (CNTL A)	Cursor Left Word. Moves the cursor back to the beginning of a word.
Key #3	WD (CNTL F)	Cursor Right Word. Moves the cursor to the beginning of the next word.
Key #4	SCRL (CNTL Z)	Scroll Up Line. Scrolls the file display up one line.
Key #5	SCRL (CNTL W)	Scroll Down Line. Scrolls the file display down one line.
Key #6	SCRN (CNTL C)	Scroll Up Screenful. Moves the display up by a screenful (approximately 3/4 the number of displayed lines).
Key #7	SCRN (CNTL R)	Scroll Down Screenful. Moves the display down a screenful.

TEXT EDITOR, DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER UTILITY PROGRAMS

Key #8 (CNTL T)	DEL WD	Delete Word Right. Deletes the word containing the cursor, and the following spaces.
Key #9 (CNTL B)	REFORM	Forms existing text, in a manner similar to entry by word wrap, to the end of the paragraph.
Key #10 (CNTL L)	AGAIN	Find/Replace Again. Causes the most recently used Find command or Replace command to be repeated as though it were entered again with the identical response to all questions.
Key #11 (CNTL QC)	EOF	Cursor End of File. Causes the cursor to move to a position just after the last character of the document.
Key #12 (CNTL QR)	BOF	Cursor beginning of file. Causes the cursor to move to a position at the beginning of the file.
Key #13 (CNTL V)	INS CH	Insertion On/Off. When WordStar is first invoked, Insertion is on, allowing typed characters to be inserted into the current line. The first depression of this key disables insertion, and characters typed over existing characters replace these existing characters. The second depression of this key again enables insertion.
Key #14 (CNTL G)	DEL CH	Delete Character Right. Deletes the character at the cursor position.
Key #15 (CNTL N)	INS LN	Insert Hard Carriage Return. Inserts a carriage return immediately following the cursor. The cursor remains where it is, and the text to the right of, or below it, moves down.
Key #16 (CNTL Y)	DEL LN	Delete Line. Deletes the entire document line in which the cursor appears.

The logos on the tops of the keys listed below are applicable only when using the Gould 9516 MicroSystem Integration Station.

HELP	MSG
TOP MENU	CMD LINE
MOVE	START
F1-F4 and F5-F8	STEP
LOG	FIELD SELECT

In addition, the cursor control keys $\uparrow, \downarrow, \rightarrow, \leftarrow$ output the characters CNTL-X CNTL-E CNTL-S CNTL-D respectively.

Invoking WordStar

Once the system prompt A> has been obtained, there are four methods of invoking WordStar:

1. Basic Method: To start WordStar with no file being edited. A copyright message appears for several seconds; then the no-file menu is displayed.
Example:
 A>WS <cr>
This basic method is sufficient for initial use.
2. To go directly to editing a document: Type WS, a space, and the name of the file, including disk drive and type as appropriate. WordStar will proceed to edit this file as though the "edit a document" command had been given from the no-file menu.
Example:
 A>WS LETTER.DOC <cr>
 A>WS B:ABC.XYZ <cr>
3. To go directly to editing a document file with the new file on a different disk drive: Type WS, a space, the name of the file to be edited, another space, and the DRIVE NAME and colon only of the disk drive to receive the edited version of the document. Do not type anything after the destination drive name and colon.
Example:
 A>WS A:BOOK.DOC B: <cr>

This method is for revising extremely long files, when the new file must be placed on a different disk because of disk space limitations.

The preceding example says to edit file BOOK.DOC on the disk in drive A and place the new version on file BOOK.DOC on drive B. When the save is completed, the file on drive A will have been renamed to BOOK.BAK. If a "save and continue edit" command is given, the continuing edit will edit BOOK.DOC from drive B onto drive A; each successive "save and continue edit" will alternate drives.

This method is for revising extremely long files, when the new file must be placed on a different disk because of disk space limitations.
4. To edit a non-document file: Type WS <cr>, and then type N at the no-file menu.

NOTE: If, when WordStar is invoked, the following two line message appears:

You are trying to run an unINSTALLED WordStar.
Please run INSTALL first.

then WordStar has not yet been installed to work with the terminal and printer. Refer to the WordStar User's Guide for installation instructions.

No-File Menu

WordStar displays the "no file menu" when it is started without a file name, or whenever editing of a file is terminated. The words "editing no file" are displayed at the top of the screen, and a "menu" of commands that may be entered is shown. Below the menu, if the file directory display is ON, WordStar displays the directory (the names of all files on the disk) of the logged drive.

To invoke one of the functions shown on the no-file menu, enter the single letter shown for that function. The letter may be entered in upper or lower case, or with the CTRL key depressed. Unrecognized characters are ignored. No RETURN or other key is used after the command letter. When a command is entered, the letter is displayed in the upper left hand corner of the screen and further action is taken depending on the command.

Table 6-1 lists the No-File Commands.

Table 6-1. No-File Commands

Command Letter	Function	Description
D	EDIT A DOCUMENT	Asks for file name, then initiates editing of the specified file. The file specified may be an existing file or a new file. To place the new version of the file on a different drive: enter the file name, a space, and the destination drive name followed by a colon.
N	EDIT A NON-DOCUMENT	Same as D except file is edited as a "nondocument", without dynamic pagination and with different defaults.
X	EXIT TO SYSTEM	Exit to CP/M or whatever operating system is in use. Used when the operator no longer requires WordStar and wishes to use a system command. The system prompt A> is displayed next.
H	SET HELP LEVEL	Asks for the new "help level" (0 to 3), which determines the degree of menu display and other prompting supplied by WordStar. An explanation of the help levels is displayed, unless the help level is already 0.
Y	DELETE	Asks for file name and then erases file. Performs the same function as the MP/M ERASE console command.
L	CHANGE LOGGED DISK DRIVE	Displays the name of the current logged disk drive and allows selection of a new logged disk drive. Used to allow display of directory of a different drive.
F	FILE DIRECTORY DISPLAY OFF/ON	Controls display of file directory (names of all files on disk in logged disk drive). First F command turns directory display off, second F turns directory display on again, etc. To display directory of a different drive, change logged disk with L command.
P	PRINT A FILE STOP PRINT CONTINUE PRINT	The P command has three possible effects, depending on whether printing is inactive, a file is being printed, or printing is suspended. The P line in the menu changes as appropriate.

Table 6-1. No-File Commands (cont'd)

Command Letter	Function	Description
M	MERGE-PRINT A FILE	In order to use the Merge-Print feature of WordStar, the MERGPRIN.OVR file must be present on the disk in drive A. The M command allows merging data from a data file into text at print time. If MERGPRIN.OVR is not present, an error message is displayed.
R	RUN A PROGRAM	The R command allows a program to be run without exiting from WordStar. For example, the amount of disk space could be checked by using the MP/M program STAT.COM. After R is entered, the following prompt is displayed: COMMAND? Enter the name of the program to be run and press RETURN. The program name may be followed by file name(s) or other arguments to be used by the program where appropriate.
O	COPY A	The O command allows the user to make a copy of a specified file FILE without having to use the MP/M program PIP.COM. The user may copy files from or to different disks as long as both disks are on-line at the same time. When O is entered, the following prompts are displayed: NAME OF FILE TO COPY FROM? NAME OF FILE TO COPY TO ? Enter the name of the file to be copied and press RETURN, followed by the name of the file where the copy is to be stored. Specify the disk drive with a drive letter and colon (A:,B:, etc.) preceding the filename.
E	RENAME	The E command allows the user to change the name of a file. E functions like the MP/M command REN.) Enter the name of the file to be renamed followed by the new name in response to the appropriate prompts: Enter NAME OF FILE TO RENAME? NEW NAME?

Examples of No-File Commands

D Command. With the no-file menu on the screen, the user types a D to invoke editing of a file. Wordstar then displays an explanation and a request to enter the file name.

The desired file name is typed, followed by a carriage return. The form of a file name is summarized on the screen as a reminder. For example,

```
LETTER.DOC <cr>
```

to edit file LETTER.DOC on the logged drive, or

```
B:LETTER.DOC <cr>
```

to edit the file LETTER.DOC on the disk in drive B.

To edit file BOOK.DOC on drive A and place the new version of BOOKDOC on drive B, type:

```
A:BOOK.DOC B: <cr>
```

After the carriage return, WordStar reads the file into memory. The user may then enter text into the document and/or use WordStar editing commands. If the file does not exist, NEW FILE is displayed for several seconds. If NEW FILE appears when the user intends to edit an existing file, he should check the spelling of the file name entered and verify that the proper disk has been inserted. In this situation, it is necessary to exit to the no-file menu by entering KQ.

In the D display (and also in the screen display of many other WordStar commands) the two lines

S=delete character	Y=delete entry	F= File directory
D=restore character	R=Restore entry	U=cancel command

list the control characters that may be used to correct typing errors and for other purposes. These characters may be used at any time while typing the answer, before pressing RETURN. The two display lines appear only at "help levels" 2 and 3, and do not appear until about two seconds have elapsed without a keystroke:

Use Control-S	to delete characters one at a time
Control-D	to restore characters one at a time
Control-Y	to erase the entire answer entered
Control-R	to restore the entire answer or previous answer
Control-F	to invoke the file directory display or to turn the the file directory on and off alternately
Control-U	to cancel the D command and return to the no-file menu

TEXT EDITOR, DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER UTILITY PROGRAMS

Y Command. Y is typed to initiate deletion of a file. The user enters the name of the file to delete and follows it by RETURN. The form of a file name and the use of control characters to correct typing errors is the same as for the D command. After the file is erased, the no-file menu reappears on the screen, and the user may enter another command. The Y command may be canceled by entering control U, or by pressing the RETURN only.

X Command. The X command is used to exit to the operating system. When an X is typed at the no-file menu, the prompt (A>) appears at the bottom of the screen.

F Command. The F command turns the file directory display off and on. The first F entered turns the directory display off, the next F restores it, etc. No additional information need be entered. No screen changes take place except that when directory display is off, the no-file menu item for F changes to read

F=File directory on (OFF)

L Command. The L command allows changing the logged disk drive. To log a different drive, the user types the letter (A, B, C, D to identify the drive), a colon, and RETURN. To leave the logged drive unchanged, the user types control-U, or just presses RETURN.

E Command. The E Command allows the user to rename files without having to exit from WordStar. This command performs the same function as the CP/M REN command. The user may rename a file on another drive by specifying the drive before the name of the file is entered (e.g., B:FILENAME.TXT). The NEW NAME? prompt is displayed after the name of the file has been entered.

R Command. The R command allows the user to run a different program without exiting from WordStar. This command is especially useful for determining the amount of available disk space by running the CP/M program STAT.COM. The user enters the name of the program to be run and presses RETURN. Only executable programs (file type .COM) should be specified. An attempt to run a non-executable file may result in an error message, or may lock up the system. When the program has completed, the following prompt is displayed:

Hit any key to return to WordStar:

This allows the user to view any results displayed by the program before returning to the WordStar no-file menu.

The R command will handle any CP/M console command (CCP command) command except the resident commands (TYPE, DIR, ERA, REN, and SAVE). File names or other arguments may follow the program name, as in CP/M commands. For example:

```
STAT LETTER.DOC
```

shows the size of file LETTER.DOC on the logged drive. Asterisks and question marks can be used to form "wild card" file names, as in CP/M console commands. For example:

```
STAT B:#.DOC
```

shows the size of all files of type .DOC on the disk in drive B. (#'s and '?'s are not allowed in file names entered in other WordStar commands.)

In order to use the R command, the user must have WS.COM (or other name as specified during INSTALLation) on the disk in drive A or the current logged drive.

O Command. The O command provides a way to copy files without exiting from WordStar.

If the name of an existing file is entered as the file to copy to, WordStar displays the prompt:

```
FILE d:name.typ EXISTS -- OVERWRITE? (Y/N):
```

Pressing Y to proceed with the copy destroys the present contents of the copy-to file. Pressing any other key will cause the NAME OF FILE TO COPY TO? question to be reasked. Pressing RETURN or U aborts the copy command.

The user may copy files from and/or to drives other than the logged drive by specifying a drive before the file name (e.g. B:FILENAME.TXT). The exact file-name to be copied must be entered. Asterisks (*) or question marks may not be used. The user may use -'s in the file names (with this or any other WordStar command) if soft-hyphen entry is OFF or by entering the - as *P-.

Help Levels

The Help Level setting controls the amount of explanatory material automatically displayed by WordStar. The setting determines whether and how much of the screen is to be used to display a "menu" of command keys that may be entered.

The Help Level is initially set to 3, the most helpful level. As the user gains experience with WordStar, he will want to reduce the help level in order to have more of the screen available for file display.

The Help Level is changed with the H command on the no-file menu, or with the JH command while editing a file. Either command displays an explanation of help levels and current help level, and requests a new help level.

TEXT EDITOR, DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER UTILITY PROGRAMS

Unlike the "FILE NAME?" questions asked by the D and Y commands, this question takes a single-key response; no RETURN is needed. Pressing a key other than 0, 1, 2, or 3 leaves the help level unchanged.

The difference between help levels 3, 2, and 1 is evident primarily when editing a file. Level 0 differs from the higher levels in that extra explanations associated with several commands are skipped. For example, the control characters that may be used while answering any question whose prompt ends in a question mark (S,D,Y,R,U) are displayed above such question only at help level 2 or 3. The control characters nevertheless work at all help levels.

In addition to the information automatically displayed by the help level, WordStar has explicit commands that display information on various subjects. For example, the command J D (entered while editing) invokes a sequence of screen displays describing the print directives.

Note: If the Wordstar editor is used to create a source file, the user must use the non-document mode. The document mode must not be used because it uses bit 7 of characters for "flags" for itself; thus the source file would contain non-ASCII characters. The user must also terminate the last line of a source file with a carriage return. If not, the cross assemblers will not recognize that line.

Additional information on WordStar can be obtained in the WordStar User's Guide.

DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER™

Dynamic Debugger

The Dynamic Debugger Tool (DDT) program allows dynamic interactive testing and debugging of programs generated in the MP/M and CP/M environments. The debugger is initiated by typing one of the following commands:

```
DDT
DDT filename.HEX
DDT filename.COM
DDT filename.PRL      (MP/M only)
```

where "filename" is the name of the program to be loaded and tested. In all cases, the DDT program is brought into main memory and resides directly below the Operating System. (Refer to the CP/M Interface Guide for standard memory organization.) The BDOS starting address, which is located in the address field of the JMP instruction at location 5H, is altered to reflect the reduced transient program area size.

The second and third forms of the DDT command shown above perform actions similar to the first, except there is a subsequent automatic load of the specified HEX or COM file. The action is identical to the sequence of commands

```
DDT
Ifilename.HEX or Ifilename.COM
R
```

where the I and R commands set up and read the specified program to test. (See the explanation of the I and R commands below for exact details.)

Upon initiation, DDT prints a sign-on message in the format

```
nnK DDT-s VER m.m
```

where nn is the memory size (which must match the CP/M system being used), s is the hardware system which is assumed, and m.m is the revision number.

Relocatable Debugger

For MP/M, in addition to the non-relocatable debugging tool (DDT.COM), there is a relocatable tool (RDT.COM). The valid commands for RDT and DDT are the same.

Commands

Following the sign on message, DDT prompts the operator with the character "-" and waits for input commands from the console. The operator can type any of several single character commands, terminated by a carriage return to execute the command. Each line of input can be line-edited using the standard operating system controls:

rubout	remove the last character typed
ctl-U or	remove the entire line, ready for re-typing
ctl-X	
ctl-C	system reboot

Any command can be up to 32 characters in length; an automatic carriage return is inserted as the 33rd character. The first character determines the command type

A	enter assembly language mnemonics with operands
B	allows update of bitmap of page relocatable file (MP/M)
D	display memory in hexadecimal and ASCII
F	fill memory with constant data
G	begin execution with optional breakpoints
I	set up a standard input file control block
L	list memory using assembler mnemonics
M	move a memory segment from source to destination
N	relocate a page relocatable file (MP/M)
R	read program for subsequent testing
S	substitute memory values
T	trace program execution
U	untraced program monitoring
V	compute the parameter to follow the W (Write Disk) Command (MP/M)
X	examine and optionally alter the CPU state
W	write a patched program to disk (MP/M)

In some cases, the command character is followed by zero, one, two, or three hexadecimal values that are separated by commas or single blank characters. All DDT numeric output is in hexadecimal form. In all cases, the commands are not executed until the carriage return is typed at the end of the command.

Differences between CP/M and MP/M

The operation of DDT under CP/M differs slightly from the operation under MP/M. The differences occur in two areas:

1. The types of input files
2. The manner of saving patched programs

DDT for CP/M can accept two formats for input. HEX files are in Intel Hexadecimal format, and COM files are an absolute memory image. DDT for MP/M not only accepts these two formats, but also accepts a Page-Relocatable Format. This format is similar to the COM format except there is a bit map at the end of the file which indicates how to relocate the program. Thus, under MP/M, the "N" command must be used to relocate a PRL file after loading it into memory. Likewise, after patching, the "B" command is used to update the bit map.

The second major area of difference is the method for saving programs that have been patched. Under CP/M, this is done by returning to the command level by

```
Control - C
G 0   Go to location 0 - JMP BIOS
```

and then by using the resident CP/M SAVE command to save the patched program.

```
SAVE n filename.COM
```

where n is the number of pages (256-byte blocks) to be saved on disk. The number of blocks can be determined by taking the high-order byte of the top load address and converting this number to decimal. For example, if the highest address in the transient program area is 1234H, then the number of pages is 12H, or 18 in decimal. Thus, the operator could type a control-C during the debug run, returning to the Console Processor level, followed by

```
SAVE 18 X.COM
```

The memory image is saved as X.COM on the disk, and can be executed directly by typing the name X. If further testing is required, the memory image can be recalled by typing

```
DDT X.COM
```

which reloads the previously saved program from location 100H through page 18 (12FFH). The machine state is not a part of the COM file, and thus the program must be restarted from the beginning in order to test it properly.

TEXT EDITOR, DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER UTILITY PROGRAMS

Under MP/M, the DDT user would use the "U" command to calculate the number of "blocks" to save, use the "I" command to name the output file, and then use the "W" command to write the file and specify the number "blocks" to write.

The individual commands are detailed in the following paragraphs. In the case of each command, the operator must wait for the prompt character (-) before entering the command.

In the explanation of each command, the command letter is shown. In some cases, the numbers are separated by commas, and the numbers are represented by lower case letters. These numbers are always assumed to be in a hexadecimal radix, and from one to four digits in length. (Longer numbers will be automatically truncated to the right.)

Many of the commands operate on a "CPU state" which corresponds to the program under test. The CPU state holds the registers of the program being debugged, and initially contains zeroes for all registers and flags except for the program counter (P) and stack pointer (S), which default to 100H. The program counter is subsequently set to the starting address given in the last record of a HEX file, if a file of this form is loaded.

A (Assemble) Command. DDT allows inline assembly language to be inserted into the current memory image using the A command, which takes the form

As

where s is a hexadecimal starting address for the inline assembly. DDT prompts the console with the address of the next instruction to fill, and reads the console, looking for assembly language mnemonics followed by register references and operands in absolute hexadecimal form. Each successive load address is printed before reading the console. The A command terminates when the first empty line is input from the console.

B (Bitmap Bit Set/Reset) Command. (MP/M only) The BITMAP BIT SET/RESET command enables the user to update the bitmap of a page relocatable file. To edit a PRL file the user reads the file in, makes changes to the code, and then determines the bytes that need relocation (e.g., the high order address bytes of jump instructions). The 'B' command is then used to update the bit map. There are two parameters specified: the address to be modified, followed by a zero or a one. A value of one specifies bit setting.

D (Display) Command. The D command allows the operator to view the contents of memory in hexadecimal and ASCII formats. The forms are:

D
Ds
Ds,f

In the first case, memory is displayed from the current display address (initially 100H) and continues for 16 display lines. Each display line takes the form shown below

aaaa bb cccccccccccccccc

where aaaa is the display address in hexadecimal, and bb represents data present in memory starting at aaaa. The ASCII characters starting at aaaa are given to the right (represented by the sequence of c's), where non-graphic characters are printed as a period (.) symbol. Note that both upper and lower case letters are displayed, and thus will appear as upper case symbols on a console device that supports only upper case. Each display line gives the values of 16 bytes of data, except that the first line displayed is truncated so the next line begins at an address which is a multiple of 16.

The second form of the D command shown above is similar to the first, except the display address is first set to address s. The third form causes the display to continue from address s through address f. In all cases, the display address is set to the first address not displayed in this command, so a continuing display can be accomplished by issuing successive D commands with no explicit addresses. Excessively long displays can be aborted by pushing the rubout key.

F (Fill) Command. The F command takes the form

Fs,f,c

where s is the starting address, f is the final address, and c is a hexadecimal byte constant. The effect is as follows: DDT stores the constant c at address s, increments the value of s and tests against f. If s exceeds f, the operation terminates. Otherwise the operation is repeated. Thus, the fill command can be used to set a memory block to a specific constant value.

G (Go) Command. Program execution is started using the G command, with up to two optional breakpoint addresses. The G command takes one of these forms:

G
Gs
Gs,b
Gs,b,c
G,b
G,b,c

The first form starts execution of the program under test at the current value of the program counter in the current machine state, with no breakpoints set. The second form is similar to the first except the program counter in the current machine state is set to address s before execution begins. The third form is the same as the second, except program execution stops when address b is encountered. (b must be in the area of the program under test.) The instruction at location b is not executed when the breakpoint is encountered. The fourth form sets two breakpoints, one at b and the other at c. Encountering either breakpoint causes execution to stop, and both breakpoints are subsequently cleared.

Execution continues in real-time from the starting address to the next breakpoint. That is, there is no intervention between the starting address and the break address by DDT. Upon encountering a breakpoint, DDT stops execution and types

*d

where d is the stop address. The machine state can be examined at this point using the X (Examine) command. At the beginning of the G command, the operator must specify breakpoints that are different from the program counter address. For example, if the current program counter is 1234H, then the commands

G,1234
and
G400,400

both produce an immediate breakpoint; no instructions are executed.

I (Input) Command. The I command prepares a file control block so the operator can use the R command to read a file from disk into memory. It sets up the default file control block (FCB) at location 5CH with the unambiguous file name and extension specified by the user. The form of the I command is

lfilename
or
lfilename.filetype

If the second form is used, and the filetype is either HEX or COM, then subsequent R commands can be used to read the pure binary or hex format machine code. (See the R command for further details.)

L (List) Command. The L command is used to list assembly language mnemonics in a particular program region. The forms are:

L
Ls
Ls,f

The first command lists twelve lines of disassembled machine code from the current list address. The second form sets the list address to s, and then lists twelve lines of code. The last form lists disassembled code from s through address f. In all three cases, the list address is set to the next unlisted location in preparation for a subsequent L command. Upon encountering an execution breakpoint, the list address is set to the current value of the program counter. (See the G and T commands.) Again, long typeouts can be aborted using the rubout key during the list process.

M (Move) Command. The M command allows block movement of program or data areas from one location of memory to another. The form is

Ms,f,d

where s is the start address of the move, f is the final address of the move, and d is the destination address. Data is first moved from s to d, and both addresses are incremented. If s exceeds f, then the move operation stops; otherwise the move operation is repeated.

N (Normalize) Command. (MP/M only) The purpose of the NORMALIZE command is to relocate a page relocatable file that has been read into memory by the debugger. To debug a PRL program the user reads it in with the 'R' command and then uses the 'N' command to relocate it within the memory segment the debugger is executing.

TEXT EDITOR, DYNAMIC DEBUGGER AND RELOCATABLE DEBUGGER UTILITY PROGRAMS

R (Read) Command. The R command is used in conjunction with the I command to read COM and HEX files from the disk into the transient program area in preparation for the debug run. Under MP/M, PRL files can also be read. The forms are

```
R
Rb
```

where b is an optional bias address which is added to each program or data address as it is loaded. The load operation must not overwrite any of the system parameters from 000H through 0FFH (i.e., the first page of memory). If b is omitted, then b=0000 is assumed. The R command requires a previous I command, specifying the name of a HEX, COM or PRL file. The load address for each record is obtained from each individual HEX record, while an assumed load address of 100H is taken for COM files. For PRL files, an offset of 100H in the current memory segment is assumed. Note that any number of R commands can be issued following the I command to reread the program under test, assuming the tested program does not destroy the default area at 5CH. Further, any file specified with the filetype "COM" or "PRL" is assumed to contain machine code in pure binary form, and all others are assumed to contain machine code in Intel hex format.

Recall that the command

```
DDT filename.filetype
```

which initiates the DDT program is equivalent to the commands

```
DDT
-lfilename.filetype
-R
```

Whenever the R command is issued, DDT responds with either the error indicator "?" (file cannot be opened, or a checksum error occurred in a HEX file), or with a load message taking the form

```
NEXT PC
nnnn pppp
```

where nnnn is the next address following the loaded program, and pppp is the assumed program counter (100H for COM files, or taken from the last record if a HEX file is specified).

S (Set) Command. The S command allows memory locations to be examined and optionally altered. The form of the command is

```
Ss
```

where s is the hexadecimal starting address for examination and alteration of memory. DDT responds with a numeric prompt, giving the memory location and the data currently held in the memory location. If the operator types a carriage return, the data is not altered. If a byte value is typed, then the value is

stored at the prompted address. In either case, DDT continues to prompt with successive addresses and values until either a period (.) or an invalid input value is detected.

T (Trace) Command. The T command allows selective tracing of program execution for 1 to 65535 program steps. The forms are:

T
Tn

In the first case, the CPU state is displayed, and the next program step is executed. The program terminates immediately, with the termination address displayed as

*hhhh

where hhhh is the next address to execute. The display address (used in the D command) is set to the value of H and L, and the list address (used in the L command) is set to hhhh. The CPU state at program termination can then be examined using the X command.

The second form of the T command is similar to the first, except execution is traced for n steps (n is a hexadecimal value) before a program breakpoint occurs. A breakpoint can be forced in the trace mode by typing a rubout character. The CPU state is displayed before each program step is taken in trace mode. The format of the display is the same as described in the X command.

Note that program tracing is discontinued at the interface to MP/M or CP/M, and resumes after return from the operating system to the program under test. MP/M or CP/M functions that access I/O devices such as the disk drive, run in real-time, thus avoiding I/O timing problems. Programs running in trace mode execute approximately 500 times slower than real time since DDT gets control after each user instruction is executed. Interrupt processing routines can be traced, but commands that use the breakpoint facility (G, T, and U) accomplish the break using an RST 7 instruction. Therefore, the tested program cannot use this interrupt location. Furthermore, the trace command enables interrupts. This can be a problem if a program requires that interrupts be disabled. The operator should use the rubout key to get control back to DDT during trace, rather than executing a RST 7. This ensures the trace for the current instruction is completed before interruption.

U (Untrace) Command. The U command is identical to the T command except intermediate program steps are not displayed. The untrace mode allows from 1 to 65535 (0FFFFH) steps to be executed in monitored mode, and is used principally to retain control of an executing program while it reaches steady state conditions. All conditions of the T command apply to the U command.

V (Value Command). (MP/M only) This command facilitates the use of the WRITE DISK command by computing the parameter to follow the 'W.' A single parameter immediately follows the 'V' which is the NEXT location following the last byte to be written to disk. For example, if the last address of a program is 1234H, then the command would be V1234 <cr>.

X (Examine) Command. The X command allows selective display and alteration of the current CPU state for the program under test. The forms are

X
Xr

where r is one of the 8080 CPU registers

C	Carry Flag	(0/1)
Z	Zero Flag	(0/1)
M	Minus Flag	(0/1)
E	Even Parity Flag	(0/1)
I	Interdigit Carry	(0/1)
A	Accumulator	(0-FF)
B	BC Register pair	(0-FFFF)
D	DE Register pair	(0-FFFF)
H	HL Register pair	(0-FFFF)
S	Stack Pointer	(0-FFFF)
P	Program Counter	(0-FFFF)

In the first case, the CPU register state is displayed in the format

CfZfMfEfIf A=bb B=dddd H=dddd S=dddd P=dddd inst

where f is a 0 or 1 flag value, bb is a byte value, and dddd is a double byte quantity corresponding to the register pair. The "inst" field contains the disassembled instruction which occurs at the location addressed by the CPU state's program counter.

The second form allows display and optional alteration of register values, where r is one of the registers given above (C,Z,M,E,I,A,B,D,H,S, or P). In each case, the flag or register value is first displayed at the console. The DDT program then accepts input from the console. If a carriage return is typed, then the flag or register value is not altered. If a value in the proper range is typed, then the flag or register value is altered. Note that BC, DE and HL are displayed as register pairs. Thus, the operator types the entire register pair when B,C, or the BC pair is altered.

W (Write Disk) Command. (MP/M only) The WRITE DISK command provides the capability to write a patched program to disk. A single parameter immediately follows the 'W' which is the number of sectors (128 bytes/sector) to be written. This parameter is entered in hexadecimal.

IMPLEMENTATION NOTES

The organization of DDT allows certain non-essential portions to be overlaid in order to gain a larger transient program area for debugging large programs. The DDT program consists of two parts: the DDT nucleus and the assembler/disassembler module. Under CP/M, the DDT nucleus is loaded over the Console Command Processor; and although loaded with the DDT nucleus, the assembler/disassembler can be overlaid unless used to assemble or disassemble.

In particular, the BDOS address at location 6H (address field of the JMP instruction at location 5H) is modified by DDT to address the base location of the DDT nucleus which, in turn, contains a JMP instruction to the BDOS. Thus, programs that use this address field to size memory see the logical end of memory at the base of the DDT nucleus rather than the base of the BDOS.

The assembler/disassembler module resides directly below the DDT nucleus in the transient program area. If the A, L, T, or X commands are used during the debugging process, the DDT program again alters the address field at 6H to include this module, further reducing the logical end of memory. If a program loads beyond the beginning of the assembler/disassembler module, the A and L commands are lost, and the trace and display (T and X) commands list the "inst" field of the display in hexadecimal, rather than as a decoded instruction.

ASSEMBLER AND LINKER UTILITIES

THE CROSS ASSEMBLER

The Cross Assembler is a 9520 Software Development System utility that assembles assembly language source programs into object modules for execution (after suitable link operation and downloading has been completed) at the remote 9508 MicroSystem Emulator or 9516 MicroSystem Integration Station.

A separate assembler utility is used for each microprocessor type. In some cases, one assembler is used to support a family of processor types, e.g. 8048 Family. Assembler directives in the source program are used to specify the particular microprocessor family member. The details of the assembler directives can be found in the assembler manuals. A separate user manual and language translator disk is provided for each assembler/cross-assembler. The available assembler manuals are listed in the preface.

The assembler is invoked in the following manner:

Entry: (assembler specifier) (primary filename) [flags] <cr>

where:

(assembler specifier) is defined by the following entry:

A8080 - The 8080/8085 assembler

A6800 - The 6800/ 6802 assembler

A6801 - The 6801/ 6803 assembler

A6809 - The 6809 assembler

A8048 - The 8048, 8049, 8021, 8022, and 8041 assembler

AZ80 - The Z80 assembler

A9900 - The TI 9900 family assembler

(primary filename parameter) is used by the command processor to specify the name of the source file (.SRC), the object file (.OBJ) and the list file (.LST). The assembler appends the appropriate file type.

Note the file type extensions are different from the MP/M and CP/M file type names. This is necessary to avoid mixing the Gould Millennium object files with those of Digital Research. The object file formats are not compatible. A blank must separate all fields in the command.

ASSEMBLER AND LINKER UTILITIES

(flags) is an optional field used to specify flags to be applied to source, object, list, and spooled console output files.

A, B, C, D - Logical drive designators

Z - No file is to be produced (applies only to object and list files)

T - Applies only to the list file and means use the system printer instead of file

X - Applied only to the list file and means use the console instead of a file

Errors in the parameter list cause "INVALID OPTIONS" to be displayed. The assembler then aborts and returns control to the operating system.

The assembly command allows one source file to be specified. At times the user may want to concatenate several source files and submit them as one source program. This is accomplished by building a file that has INCLUDE statements. For example, assume the user has three source files (B:FILEA.SRC, A:FILEB.SRC, FILEC.SRC) and wants them treated as one source program by the assembler. This is accomplished by creating a file, e.g. SOURCE.SRC that contains the following statements:

```
ORG 100H
INCLUDE "B:FILEA"
INCLUDE "A:FILE"
INCLUDE "FILEC"
END
```

In the assemble command, the name SOURCE is given for the source filename. The assembler opens the file SOURCE.SRC, finds the first INCLUDE statement, opens B:FILEA.SRC and processes it, finds the second INCLUDE, then processes A:FILEB.SRC. The third INCLUDE statement specifies FILEC.SRC. The default logical drive designator A is prefixed, making it A:FILEC.SRC.

NOTE: Include files cannot be nested. Therefore, one include file cannot call another include file. The end directive should not be used in any of the include files.

THE LINKER

Overview

The linker is a 9520 Software Development System utility program. It produces an executable load module by linking relocatable object modules and/or absolute object modules produced by the 9520 cross assemblers. The linker supports the unique qualities of each of the microprocessors supported by the 9520. The linker's outward appearance and its method of operation remain the same regardless of the cross assembler used.

The object modules output from the Assembler includes Text Blocks, Relocation Blocks, and Global Symbol Directory Blocks. Text Blocks from an independently assembled program section include three types of information:

1. Constants and machine instructions whose values are independent of their position in memory
2. Addresses or address constants whose values are relative to the starting location (base) of a section
3. Global references to other object modules whose values cannot be determined until all sections are assigned memory locations.

Relocation Blocks contain information necessary to update and relocate bytes of program text. Global Symbol Directory Blocks define global symbols and sections.

To create a load file (absolute object file), the linker performs three specific functions for each module:

1. Allocates memory space for each section of the load file
2. Establishes a reference table of global symbols
3. Relocates address-dependent locations to correspond to allocated space, when necessary.

In addition, the linker generates a listing that indicates where sections are allocated and states the values of all global symbols. If local symbols have been included in any object module, it also states the values of those symbols.

The linker uses as input object files generated by a cross assembler or by a previous link. Figures 7-1, 7-2 and 7-3 are block diagrams representing the operation of the linker with different types of input.

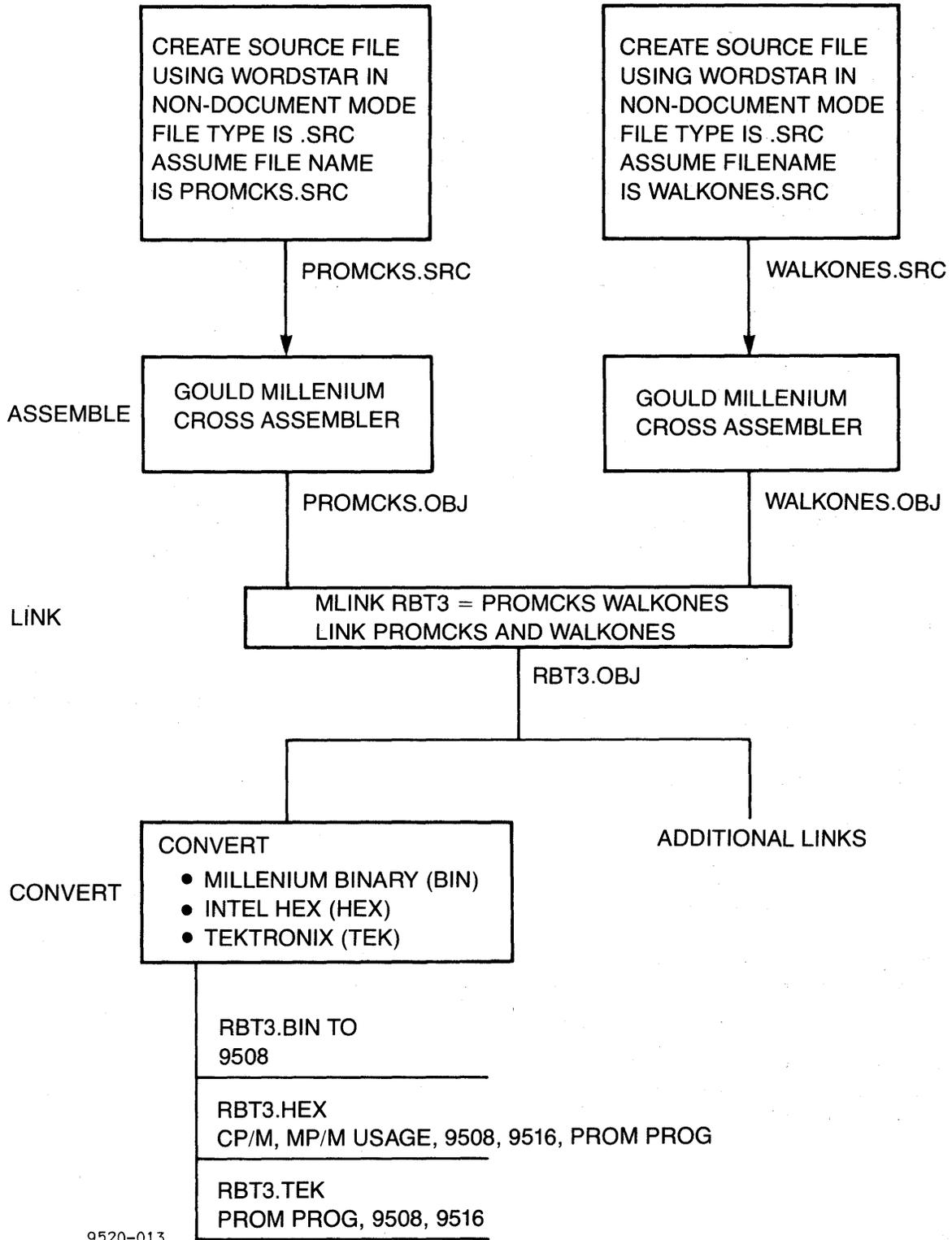
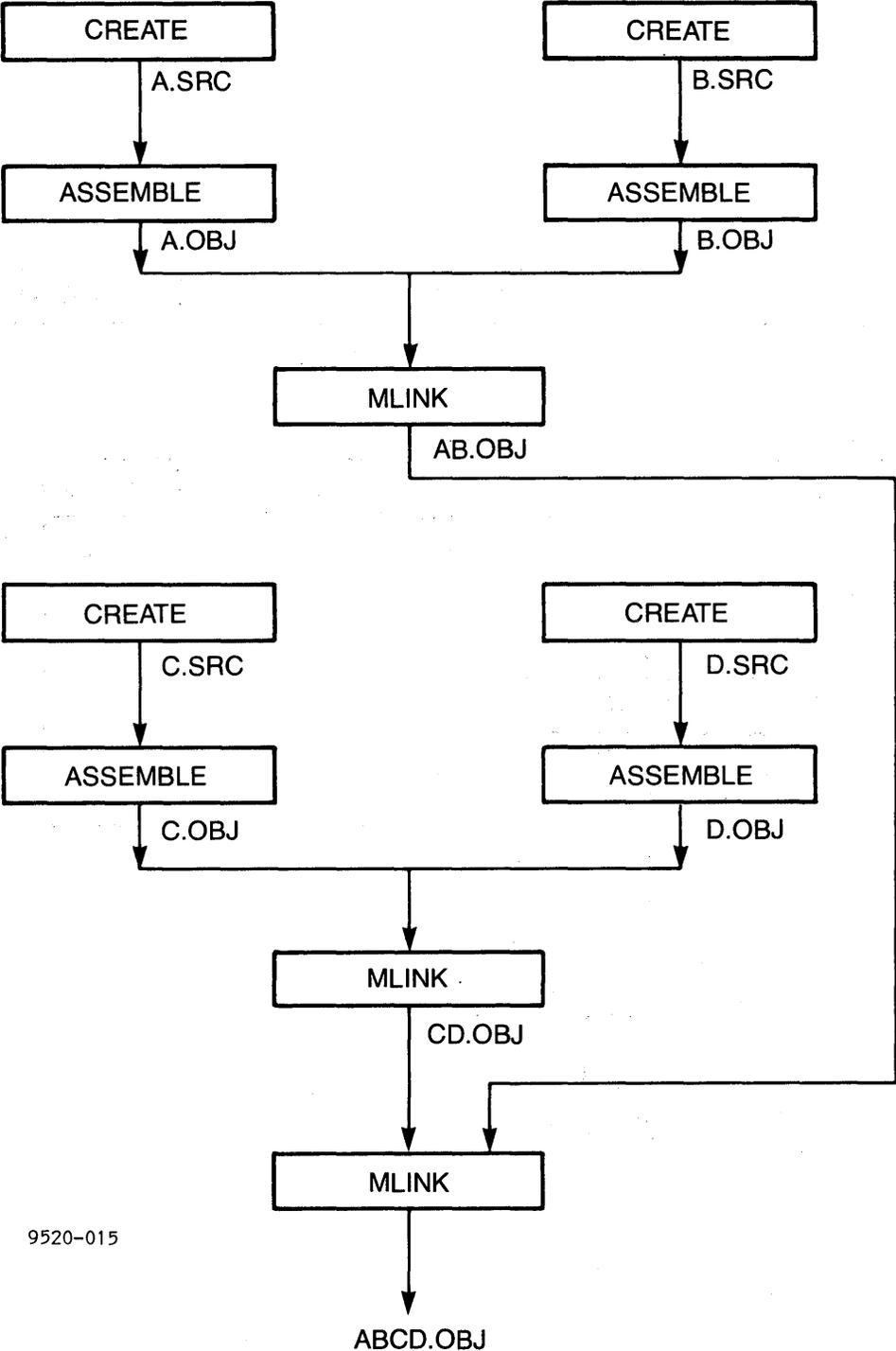
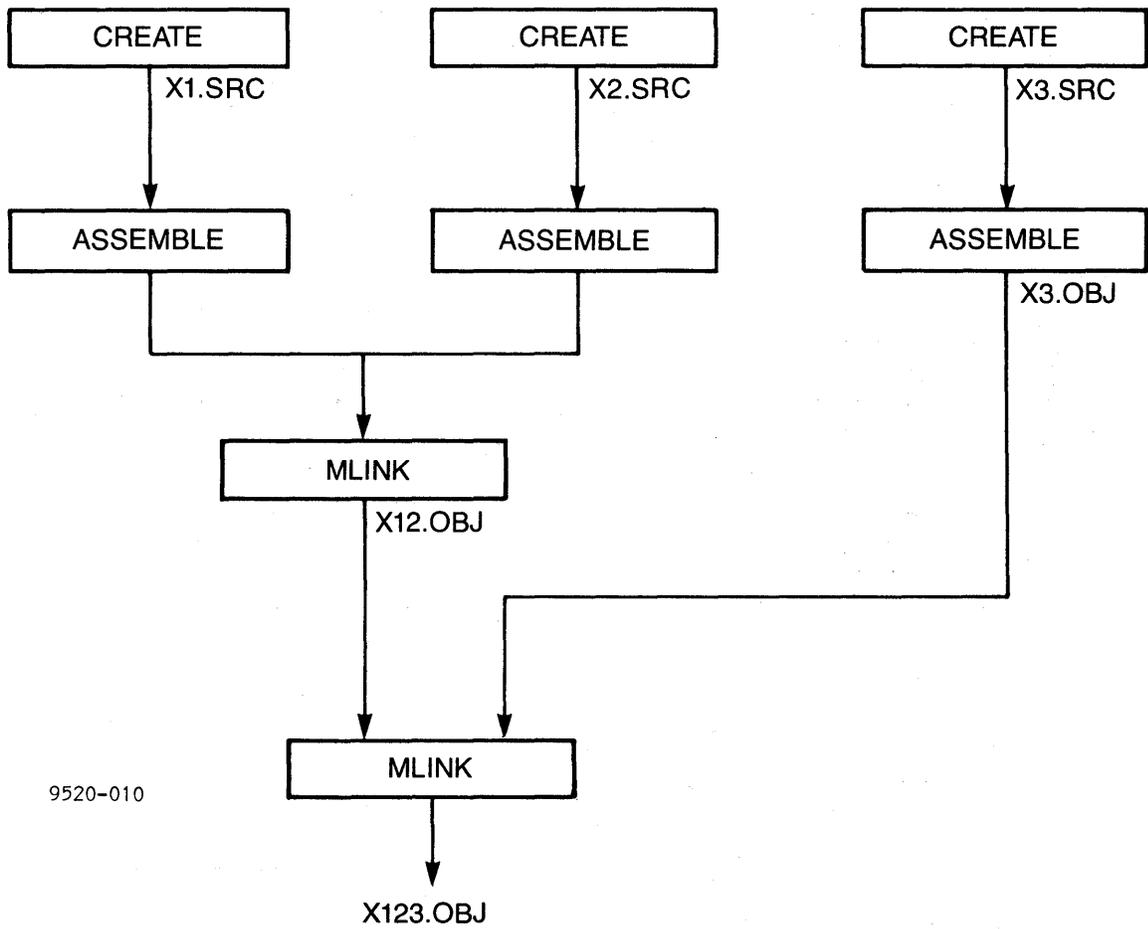


Figure 7-1. Linking and Converting Object Files Generated by Cross Assembler



9520-015

Figure 7-2. Linking Object Files Generated by Previous Link



9520-010

Figure 7-3. Linking Object Files Generated by Cross Assembler and a Previous Link

This section describes how to use the linker utility program, MLINK. It is divided into these sections:

- o LINKER INVOCATION
- o LINKER EXECUTION
- o LINKER COMMANDS
- o LINKER OUTPUT
- o ERROR MESSAGES

Advice to the First Time Linker User

Before invoking the linker it is recommended that the user do the following:

1. Study the linking diagrams, figures 7-1 through 7-3, carefully. MLINK is designed to work with object files produced by Gould Millennium cross assemblers and will not work with object files created by other vendor's cross assemblers or compilers.
2. When the linking process is understood, scan the remainder of this chapter, noting in particular the three forms of linker invocation.
3. When using the linker for the first time, use the Simple Command Mode.
4. Make sure that, when linking, the currently logged on disk contains all the linker routines, including COM files and overlay files (OVR). If all the routines are not present on the disk, the linking process will abort.

ASSEMBLER AND LINKER UTILITIES

LINKER INVOCATION

The linker may be invoked by one of three methods:

1. Simple Linker Command
2. Interactive Command Mode
3. Indirect Command Mode

The Simple Command Mode is used when the user wishes to link two or fewer object files and does not wish to control the manner in which the linker performs the linking and locating or the contents of the map/log file. This method requires the user to enter filenames only. All other parameters are set to reasonable default values.

If the user wants to link more than two object files, or control the process of linking and locating, or the contents of the log/map file, then he must use the Interactive or the Indirect Mode. In the Interactive mode, the linker accepts commands from the console; whereas in the Indirect mode, the linker accepts commands from a file.

In the Interactive Command mode, the linker causes the program to issue prompts to the user so that he supplies the required input. The Indirect Command mode requires the user to specify a file that contains linker commands.

File Name Conventions

Extensive use is made of file types in the linker commands. A complete file-id has three parts:

1. Logical drive designator: The first character, followed by a colon (:), specifies a logical drive. The letters A, B, C, and D specify logical drives. If a logical drive is not specified, the system defaults to the current logged on drive.
2. Filename: A filename has a maximum of eight (8) characters. (See the MP/M User's Guide or the CP/M Handbook for details.)
3. File extension (type): The file extension is preceded by a period (.) and is specified with zero to three characters. The file types used in linker commands are:
 - a. OBJ - Means object file. Object files are produced by the assemblers and the linker.
 - b. MAP - Means a list file. This file will contain map and/or log information.
 - c. IND - Contains linker commands. This file is used only with the Indirect Command Mode.

Simple Command Mode

The simple form of invoking the MLINK command is:

```
0A> MLINK loadfile=linkfile1 linkfile2 >F1 F2 Z
```

where:

1. MLINK invokes the linker
2. loadfile specifies a logical disk drive (optional) and filename that is used to create file-ids for the load file and the listing file.

The loadfile-id is created by appending the file type .OBJ.

The listing file-id is created by appending the file type .MAP.

The "loadfile" name cannot be the same as a "linkfile" name.
3. linkfile specifies a logical disk drive (optional) and filename that is used to create the file-id of an input file. The file-id is created by appending the file type .OBJ.
4. F1 specifies a flag for the load file. Valid characters are:
A through P - logical drive designators
Z - do not produce a loadfile
5. F2 specifies a flag for the listing file. See valid characters under F1. "X" causes the list file to be displayed on the console. "T" causes the output to go to the printer.
6. Z indicates that normal console output is suppressed. Error messages indicating disk file open, close, or write errors are not suppressed.

The F1 and F2 flags override any logical disk that was specified as part of the "loadfile" parameter.

ASSEMBLER AND LINKER UTILITIES

Simple Command Mode (cont'd)

```
* * * * *
*           CAUTION           *
*                               *
*   Since the output of the linker is   *
*   an OBJ file, the user should make   *
*   sure the loadfile is different from *
*   the "linkfile" name. Otherwise, the *
*   linker will write over an input file.*
*                               *
* * * * *
```

Example:

```
OA> MLINK A:ABS1=A:MAIN A:PUTCHR >AA
```

Assuming that A:MAIN.OBJ contains the byte-relocatable section %MAIN and A:PUTCHR.OBJ contains the byte-relocatable section %PUTCHR, then the output file A:ABS.OBJ will contain an absolute section "ABS." The section %MAIN will be located at the lowest address followed by the section %PUTCHR, since sections with the same relocation type are linked by name in ascending ASCII sequence.

Interactive Command Mode

The Interactive Command Mode is invoked as follows:

```
0A> MLINK <cr>
```

System Response: The linker responds with a prompt character (*), to indicate that linker commands will be accepted. Each command is terminated with a carriage return. Commands are accepted until the END command is received. The END command directs the linker to discontinue command entry and to begin processing the object files. See Linker Commands for a list of legal commands. Both upper and lower case are accepted.

Example:

```
A> MLINK
*LOG
*MAP
*LIST HAMMER
*LOCATE Z8001A, RANGE(2000,6FFF),PAGE
*LINK Z8001A,Z8001B
*LOAD B:Z8001
*END
```

Two object files A:Z8001A.OBJ and A:Z8001B.OBJ are linked together. The section Z8001A will be allocated by the linker to memory in the range 2000-6FFF and starts on a page boundary. Note in the LOCATE command Z8001A is a section name and in the LINK command Z8001A is a File name. Section names and file names do not have to be the same. The loadfile-id is B:Z8001.OBJ. The listfile-id is A:HAMMERMAMP.MAP.

Assume that A:Z8001A.OBJ contains a section named Z8001A.

ASSEMBLER AND LINKER UTILITIES

Indirect Command File Mode

The Indirect Command File Mode is invoked as follows:

```
OA> MLINK @indirect
```

where: `indirect` is the logical disk drive (optional) and a filename of a file containing linker commands. The total file-id is constructed by appending a file type of `.IND`.

System Response: The linker opens the indirect file and reads commands from the file until an end-of-file or an END command is encountered. End-of-file or END directs the linker to begin processing the object modules. If errors have been generated, the linker aborts with the message:

```
ERROR IN INDIRECT FILE, LINK ABORTED
```

See Linker Commands for a list of legal commands.

Example:

```
A> MLINK @B:INTELCHP
```

Where the file `B:INTELCHP.IND` contains:

```
LOG
MAP
LIST T:
LINK 8086A,8086B,8086C
LOCATE 8086A, BASE(7200)
LOCATE 8086B, RANGE(3000,BFFF),INPAGE
LOCATE 8086C,PAGE
LOAD 86CHIPS
END
```

Assume that the following sections are contained in these files:

File	Section
A:8086A.OBJ	8086A
A:8086B.OBJ	8086B
A:8086C.OBJ	8086C

Three object files `A:8086A.OBJ`, `A:8086B.OBJ`, `A:8086C.OBJ` are linked together, forming the file `A:86CHIPS.OBJ`. The section 8086A must start at location 7200. The section 8086B will be allocated in the range 3000-BFFF and will not cross a page boundary. The section 8086C will start on a page boundary. The loadfile-id is `A:86CHIPSOBJ`. The list file is output to the printer.

LINKER COMMANDS

The commands on the following pages may be used in invoking the Interactive or Indirect Command File Modes.

DEFINE

END

LINK

LIST

LOAD

LOCATE

LOG

MAP

NOCONO

NOLOG

NOMAP

TRANSFER

ASSEMBLER AND LINKER UTILITIES

DEFINE

COMMAND NAME: DEFINE

Command DEFINE symbol=value ,symbol=value ...
Syntax:

Input symbol A global symbol
Parameters: value A hexadecimal constant

Explanation

The DEFINE command assigns values to selected global symbols. Each symbol is entered into the global symbol table and assigned the corresponding value. Even if the global symbol was previously defined (by an object module), the value specified in a DEFINE command replaces the already-defined value.

Examples

```
DEFINE ABC=4000, DEF=1FFF,
```

This DEFINE command gives values to the global symbols ABC and DEF.

END

COMMAND NAME: END

Command END

Syntax:

Explanation

The END command signals the end of the command sequence. This command is entered to start the linking process after the user has completed entering all other linker commands.

This command must be used in interactive invocation, but can be omitted for command file invocation. If END is omitted in command file invocation, the linker begins the linking process when the end of the command file is reached.

Errors in a command file invocation terminate command entry mode. If errors are detected, an appropriate message is issued and control is returned to the system console.

ASSEMBLER AND LINKER UTILITIES

LINK

COMMAND NAME: LINK

Command LINK d: object-file, d: object-file ...
Syntax:

Input d: A disk drive letter. If the drive is omitted,
Parameters: the currently logged on disk drive is used.

 object-file The filename of an object file to be linked. No
 file type is specified. A file type .OBJ is
 appended.

Explanation

The LINK command designates the input object files that make up the program.

More than one LINK command can be specified in a sequence of linker commands. Successive LINK commands specify additional object files. For example, the command "LINK X, Y, Z" is identical in function to the command "LINK X" followed by commands "LINK Y" and "LINK Z".

Examples

LINK A:RASP

This command selects the object file A:RASP.OBJ on drive A to be linked.

LINK B:RASP, JAN

Assuming the logged-on drive is "A," this would cause the files B:RASP.OBJ and A:JAN.OBJ to be included in the link.

ASSEMBLER AND LINKER UTILITIES

LOAD

COMMAND NAME: LOAD

Command LOAD d: Filename
Syntax:

Input filename The filename of the output load file.
Parameters:

- d: The disk drive on which the load file resides. If the drive is omitted, the current active drive is used.

The file-id is constructed by appending the file type .OBJ to the logical disk drive and filename.

Explanation

The LOAD command designates the output disk file that receives the linked program. After linking, the file designated by the linker LOAD command may be input to the convert utility program.

Examples

LOAD B:RESULT

This LOAD command designates B:RESULT.OBJ to receive the linked program.

ASSEMBLER AND LINKER UTILITIES

LOCATE (Cont'd)

Examples (Cont'd)

LOCATE ASECT, PAGE

This linker command redefines ASECT to be page-boundary relocatable. The linker attempts to place the first address of ASECT at a page boundary. A warning message is displayed if ASECT was not defined to be page-relocatable at assembly time.

LOCATE ASECT, RANGE(0000,7FFF), BYTE

This linker command designates that ASECT will be placed somewhere in the lower 32K of memory, between 0000 and 7FFFH, and redefines ASECT to be byte-relocatable.

LOG

COMMAND NAME: LOG

Command LOG
Syntax:

Explanation:

The LOG command causes all subsequent linker commands to be recorded (logged) in the linker listing file.

The NOLOG command restores the default setting: commands are not recorded in the linker listing file.

ASSEMBLER AND LINKER UTILITIES

MAP

COMMAND NAME: MAP

Command

Syntax: MAP

Explanation

The MAP command causes the map to be included in the linker listing file. Refer to the description of the map in the Linker Output section in this chapter.

The NOMAP command restores the default setting. The map is not included in the linker listing file.

NOCONO

COMMAND NAME: NOCONO

Command

Syntax: NOCONO

Explanation

The NOCONO command causes all normal console output to be suppressed. In the Interactive and Indirect modes, this action is deferred until the user gives the END command. Messages that indicate a disk file open, write or close error are not suppressed.

ASSEMBLER AND LINKER UTILITIES

NOLOG

COMMAND NAME: NOLOG

Command

Syntax: NOLOG

Explanation:

The NOLOG command disables the recording (logging) of linker commands in the linker listing file. Refer to the LOG command description for further information.

NOMAP

COMMAND NAME: NOMAP

Command

Syntax: NOMAP

Explanation:

The NOMAP command restores the default map setting. The map is not included in the linker listing file. Refer to the description of the MAP command for further information.

ASSEMBLER AND LINKER UTILITIES

TRANSFER

COMMAND NAME: TRANSFER

Command TRANSFER global-symbol
Syntax: value

Input global-symbol A symbol appearing in the global symbol table.
Parameters:
 value A hexadecimal value that must begin with a digit
 (0 to 9).

Explanation:

The TRANSFER command defines the load file transfer address. The transfer address designates the address of the first instruction to be executed when the program is run. This address is displayed when the program is loaded into memory.

The transfer address can either be a fixed value (given as a hexadecimal address) or a global symbol. If a global symbol is designated, the transfer address is taken from the symbol's value after linking.

The transfer address may have been given

Examples:

TRANSFER 7000

This linker command designates address 7000 (hexadecimal) as the location of the first instruction to be executed.

TRANSFER ENTRY

This command designates the value of the global symbol ENTRY as the transfer address. When linking is completed, the value of ENTRY is taken from the global symbol table and designated as the transfer address.

LINKER EXECUTION

Program Sections

A section is a collection of object code that has been assembled with the same location counter. An object code module is a file containing object code and consists of one or more sections. These sections are treated separately by the linker. No limit is placed on the number of sections per link, but no more than 255 sections or globals may exist in any one object module.

A section has five attributes that provide the linker with information regarding memory allocation and where to link the section. These attributes are section name, section type, size, relocation type, and memory location.

1. Name: A section has a name consisting of up to eight characters, assigned by the section directives, SECTION, RESERVE, or COMMON at assembly time. The name must be a valid identifier. The section name is entered into the linker's symbol table and is a valid external symbol. If no SECTION, RESERVE, or COMMON directive appears in the source file, a default section name is created by appending seven characters from the filename to the character "%".
2. Section Type: The section type may be either Section, Reserve, or Common. The specification is made through use of the SECTION, RESERVE, or COMMON directive at assembly time. If no SECTION, RESERVE, or COMMON directive is used, the default section type is SECTION.

For Section Type = SECTION, each Section name must be unique. Multiple Sections with the same name will be flagged as errors, and only the first one will be linked.

Reserve sections with the same name are concatenated by the linker. The length of a Reserve section in a load module is the sum of all Reserve sections with the same name.

Common sections with the same name are allocated the same space in memory. The length of the linked Common is that of the largest Common section.

3. Size: The size of each section in an object module is determined at assembly time. Section size is the number of memory bytes that the section may occupy.
4. Relocation Type: A section may be absolute (non-relocatable), byte-relocatable, page-boundary-relocatable, or inpage-relocatable. The default relocation type is byte-relocatable.

ASSEMBLER AND LINKER UTILITIES

- a. Absolute: An absolute section is not relocated by the linker. Memory locations in an absolute section where code has been generated, or where locations have been explicitly reserved by the Assembler BLOCK directive, are not allocated to any relocatable section at link time. However, if two or more absolute sections have code at the same address, the contents of those memory locations after linking are undefined. These memory conflicts, if they occur, are noted on the linker memory map.
 - b. Byte-relocatable: A byte-relocatable section can be placed anywhere in memory.
 - c. Page-relocatable: A page-relocatable section begins on a page boundary (an integral multiple of the page length). The definition of page boundary is processor dependent; however, most processors have page boundaries every 100H bytes.
 - d. Inpage-relocatable: An inpage-relocatable section is placed entirely within a microprocessor page. The length of the page is microprocessor-specific. Page length for each microprocessor is given in the corresponding assembler cross-reference manual. If an inpage-relocatable section exceeds one page in length, the linker displays an error, redefines the relocation type of the section to be page-relocatable, and continues the linking process.
5. Memory Location: At link time the user may specify a relocatable section location, in the form of either a base address or an address range where the section may be placed. The default range for a relocatable section is the entire address space of the microprocessor. If the user elects not to specify a location for a section, the linker locates the section. An absolute section cannot be moved at link time.

The Default Section

If no SECTION directive is entered before assembly, the entire module is considered to be a byte-relocatable section which is constructed from the source file name. For example, if the name of the source file were MAINPROG.SRC, the default section name would be %MAINPRO.

Memory Allocation of Sections

The linker allocates memory in the following sequence:

1. Absolute sections.
2. Based sections. Based means a program section starting location has been specified by a LOCATE command.
3. Ranged page-relocatable section*. Ranged means the user has explicitly declared a RANGE (starting address, ending address) with the LOCATE command at link time.
4. Ranged inpage-relocatable sections*.
5. Ranged byte-relocatable sections*.
6. Page boundary-relocatable section*.
7. Inpage-relocatable section.
8. Byte-relocatable sections.

*Range was declared at link time.

Sections having the same allocation type are linked by name in ascending ASCII sequence. See the following page for examples.

ASSEMBLER AND LINKER UTILITIES

Memory Allocation of Sections (cont'd)

Examples

File-id	Source Contents
CMNHAND1.SRC	SECTION CMNDHAND,ABSOLUTE ORG 4000H . . END
CRACK1.SRC	SECTION CRACKER,ABSOLUTE ORG 5000H . . END
ABLE.SRC	SECTION AAA . . END
BAKER.SRC	SECTION BBB . . END
CHAIN.IND	MAP LINK CRACK1,BAKER LINK CMNHAND1,ABLE LOCATE AAA,BASE(9000) LOCATE BBB,BASE(C000) LOAD HOOKED END

Assemble the source code in

```
CMNHAND1.SRC  
CRACK1.SRC  
ABLE.SRC  
BAKER.SRC
```

Note that BAKER.SRC contains two sections - BBB and CCC.

Link the object files with the command

```
MLINK @CHAIN
```

The order of linking is

ABSOLUTE sections CMNDHAND and CRACKER are linked in the order of their alphanumeric names at the locations specified by the ORG directive.

BASE sections AAA and BBB are linked.

Byte-relocatable section CCC is allocated the lowest available memory space.

Examples of Memory Allocation Types

1. Absolute

```
Source-code      SECTION  PARTX,ABSOLUTE
                  ORG      1000H
```

where: PARTX is the section name. The "ABSOLUTE" and "ORG 1000H" statements force this section to absolute and are located at 1000H.

Note that the link directive LOCATE is not used with the section name PARTX.

2. Based

```
Source-code      SECTION  XYZ
```

where: XYZ is the section name and relocation options are absent

```
Link directive   LOCATE XYZ, BASE(7000)
```

3. Ranged page-relocatable

```
source-code      SECTION  MNP,PAGE
```

where: MNP is the section name and relocation options are absent

```
Link directive   LOCATE MNP, RANGE (4000-7FFF)
```

Examples of Memory Allocation Types (cont'd)

4. Ranged inpage-relocatable

Method one:

Source-code SECTION PARTS

where: PARTS is the section name and relocation options are absent

Link directive LOCATE PARTS, RANGE (0000-4FFF),INPAGE

Method two:

Source code SECTION NAME1,INPAGE

where: NAME1 is the section name and INPAGE is the relocation option

Link directive LOCATE NAME1, RANGE(0000-4FFF)

5. Ranged byte-relocatable

Method one:

Source code SECTION MILL

where: MILL is the section name and relocation options are absent

Link directive LOCATE MILL, RANGE(4000-7FFF),BYTE

Method two:

Source-code SECTION MILL

where: MILL is the section name

Link directive LOCATE MILL, RANGE(4000-7FFF)

Examples of Memory Allocation Types (Cont'd)

6. Page-relocatable

Method one:

Source code SECTION SORT

where: SORT is the section name and relocation options are absent

Link directive LOCATE SORT,PAGE

Method two:

Source code: SECTION SORT,PAGE

where: SORT is the section name
PAGE is the relocation option

Link directive LOCATE is not used with the section name SORT

7. Inpage-relocatable

Method one:

Source code SECTION CH19520

where: CH19520 is the section name and relocation options are absent

Link directive LOCATE CH19520,INPAGE

Method two:

Source code SECTION CH19520,INPAGE

where: CH19520 is the section name
INPAGE is the relocation option

Link directive LOCATE is not used with the section name SORT

8. Byte-relocatable

Source-code SECTION EIGHT

where: EIGHT is the section name and the absence of relocation
options defaults to relocation type BYTE

Link directive LOCATE EIGHT,BYTE

ASSEMBLER AND LINKER UTILITIES

LINKER OUTPUT

Output from the linker consists of the linker listing file and the load file.

Linker Listing File

The listing file may be output either to a flexible disk file, to the console, line printer, or it may be omitted.

The following information may be included in a linker output listing:

<u>Output Content Listed</u>	<u>Indirect/Interactive Command File Invoked</u>	<u>Simple Command Mode Invoked</u>
Linker Commands	If LOG was specified	Not applicable
Global Symbol List	Yes	Yes
Internal Symbol List	If MAP was specified	Yes
Map	If MAP was specified	Yes
Linker Statistics	Yes	Yes
Error Message	If necessary	Yes

Global Symbol List. A global symbol list is an alphabetical list of all global symbols (sections and symbols) and their assigned values. These global symbols include those defined with the GLOBAL directive, as well as the names of SECTION, COMMON, and RESERVE. If a symbol is undefined, its value field contains asterisks.

Internal/Local Symbol List. The internal or local symbol list contains all symbols in the source file and their actual values. The list consists of three parts:

1. Scalars.
2. Alphabetical list of labels for each section.
3. Alphabetical list of labels for each unbound global.

If there are no labels for a section or global, then no list for that section or global is output. The internal symbol list is displayed only if the DBG parameter was entered with the LIST directive in the source file before assembly.

Map. The map consists of two parts: a module map, and a memory map. The map is included in the listing file only if the linker MAP command has been specified. A module map lists all modules linked into the load file. The module map contains information about sections and global symbols defined in each object module.

The module map lists all linked modules. An alphabetical list of sections and entry points (globals defined within each section) is included for each module. If no sections were linked in a module, no room for a section exists, or a section is empty, an appropriate message is included in the module map.

A memory map is an ordered listing of the memory allocated to sections. The list starts with the lowest allocated address and continues to the highest allocated address. For every address range, each section name and its attributes are given.

Any address conflict (two or more sections assigned to the same address) is noted by an asterisk (*) following the address range in which the conflict occurs.

Linker Statistics. The linker statistics give the number of errors, undefined symbols, modules, sections, and the transfer address.

Example:

```
NO ERRORS      NO UNDEFINED SYMBOLS
3 MODULES     6 SECTIONS
TRANSFER ADDRESS IS 3E4A
```

The transfer address identifies the program starting location.

Load File

The primary output from linker processing is the load file. A load file is a subset of the linker input object modules with all references and relocation resolved. It has the same record format as the object files produced by the assembler except there is no relocation information or global symbol information. The load file is an absolute object file.

ERROR MESSAGES

Command Processing Errors

The following error messages appears if errors are made in the process of entering the linker commands.

EXTRANEIOUS INFORMATION IGNORED

Extra characters are on a command line that only requires an instruction (e.g., LOG, NOLOG, MAP). The Linker performs the appropriate action for the command, ignoring extra characters on the line.

ILLEGAL COMMAND

The command was not recognized.

ASSEMBLER AND LINKER UTILITIES

ERROR MESSAGES (cont'd)

SYNTAX ERROR

Statement syntax is invalid. This error occurs when a command is incorrectly formed. For example, if unmatched parentheses are found in the LOCATE command, or an operand is missing after the equals sign in the DEFINE command.

INDIRECT FILE DEPTH EXCEEDED

A filename command was found during processing of an indirect command file. The command is ignored.

INVALID FILENAME

The file in a LIST, LOAD, or LINK command contains illegal file characters. (See the MP/M User's Guide or the CP/M Handbook for filename details.)

NOTE: Processing of the command line ceases when an invalid filename is encountered. All files up to the invalid filename, in the case of the LINK command, are added to the list of files to be linked.

INVALID RANGE SPECIFIED

The range (starting address through ending address) in the LOCATE command is invalid. The ending address must be greater than the starting address.

PRINTER BUSY

If the listing file is being directed to the printer, and the linker cannot get exclusive control of the printer under MP/M, this message is displayed. The linker will continue to try to get exclusive control.

Errors During Linker Execution

Three classes of errors can be generated during linker execution:

1. WARNING (W): A problem may exist, but the linked program can probably be executed.
2. ERRORS (E): The linked program probably will not execute properly.
3. FATAL ERRORS (F): The error directly affects the linker's execution. The linker closes all files and returns control to the operating system.

All errors cause message to be output to the LOG and LIST file or device. A fatal error will be output to the console even if NOLOG was specified.

ERROR MESSAGES

SECTION name CHANGED FROM INPAGE TO (BYTE) RELOCATABLE
(PAGE)

Section length is greater than the page size of the microprocessor. This could occur if several inpage reserve sections were linked together and their total size exceeded the page size of the microprocessor. A section declared to be inpage relocatable, in a LOCATE command, will generate this error if the section exceeds microprocessor page size. If section size exceeds available page size, relocation will then be to a byte boundary.

TRANSFER ADDRESS UNDEFINED

No transfer address was specified to the linker either through the TRANSFER command or by specifying "END (expression)" during assembly. When no transfer address is specified, the linker creates transfer address 0.

TRANSFER ADDRESS MULTIPLY DEFINED IN MODULE name FILENAME

The module has attempted to redefine the transfer address previously specified by a linked module or by the TRANSFER command. The linker uses the first encountered transfer address to generate a transfer address for the load module. If no transfer address is specified, a transfer address of 0 is generated.

RELOCATION TYPE OF SECTION name MULTIPLY DEFINED IN MODULE name FILEname

An attempt was made to redefine the section relocation type (byte, page, inpage, or absolute). This occurs when the LOCATE command defined a relocation type differing from that specified at assembly time. The error also occurs when relocation attributes of a COMMON or RESERVE section differ between modules. The linker uses the first encountered relocation attribute to define the section.

TRUNCATION ERROR AT nnnn IN MODULE name FILEname

The relocated value computer for LO byte relocation is too large or too small to fit into one byte.

MACHINE REDEFINED FROM microprocessor IN MODULE name FILEname

The current input module has been generated for a different microprocessor than the previous object modules. Differences between microprocessor definitions may cause incompatibilities during linking (e.g., page length, alignment, etc.).

IMPLICIT REORIGIN TO 0 IN SECTION name IN MODULE name FILEname

The linker processed an object module where code in an absolute Section wrapped around from location FFFFH to 0.

ASSEMBLER AND LINKER UTILITIES

ERROR MESSAGES (Cont'd)

SECTION name CHANGED FROM PAGE TO BYTE RELOCATABLE

Either: The section was declared to be page-relocatable and the linker does not support paging for that microprocessor, or there was insufficient room for a paged section in available memory. The linker will attempt to allocate memory for the Section on a Byte Relocatable Boundary.

ATTEMPT TO REDEFINE FILE TYPE FOR filename

Filename was specified twice, once as an object file and once as a library file. The linker uses the first file type specified.

These error messages are classified ERRORS:

NO ROOM IN RANGE nnnn-nnnn FOR SECTION name

The section length is greater than available contiguous memory in range nnnn-nnnn of allocated section memory.

Symbol name MULTIPLY DEFINED IN MODULE name FILEname

Indicates an attempt was made to redefine a global symbol or section. This error occurs when two modules both define a global of the same name or when two sections have the same name. Code section names must be unique. In the event of multiply defined sections, the linker will only include the first one in the load file.

UNRESOLVED REFERENCE AT nnnn MODULE name FILEname

A reference to an undefined global or section was specified at this point in the object code. This occurs when a global is used in one module but was never defined. The unresolved reference is filled with zeros in the load file.

SECTION name EXCEEDS MAXIMUM SIZE

Section length is greater than the address space of the microprocessor. The section is not included in the load file. This error may occur when a Reserve is too long.

ERROR MESSAGES (Cont'd)

The FATAL ERRORS are the following:

F. LINKER INTERNAL ERROR AT nnnn

An error occurred in the linker. Try linking again. If this error persists, carefully document the incident and submit an Software Performance Report to Gould.

F. INVALID OBJECT CODE FORMAT FOR FILE name
LOCATION = nnnn

The information in file is not valid input object format. Make certain that all files to be linked have been assembled. Location is the internal linker address where the object file error was detected.

F. UNABLE TO ASSIGN file or device name

A filename specified as an input object module does not exist, or file/device is unavailable.

F. MEMORY FULL

Linker memory is totally allocated and linking has been terminated. The total number of globals, sections, or object modules must be reduced in order to link in the available memory.

F. (LIST FILE)
(LOAD FILE)
(CONSOLE) I/O ERROR #nn
(COMMAND FILE)
(OBJECT FILE)

This error indicates the linker was unable to read to or write from the specified file or device.

INTRODUCTION

This chapter describes how to modify the MP/M operating system configuration to meet the specific needs of a user's application environment.

The need for generating a new system configuration arises when the current space allocated for memory segment bases does not accommodate the size of the user's program. This situation could result in a possible overlap of memory partitioning for the transient program area. Also, the system operating requirements, (e.g., the need for additional resident programs, or the add-on of optional extended memory capability) would make it necessary to change parameters and generate a new system configuration.

NOTE: Before performing the steps in this chapter, Gould recommends that the user duplicate the system disk provided with the 9520. After the 9520 has been powered-up (refer to chapter 3), insert the system disk into the disk drive and press the 9520 front panel RESET switch to reinitialize the 9520 so the software on the disk is pulled into memory. When the prompt OA> is displayed on the display terminal, insert a disk in the unused disk drive, enter the command FDISK <cr>, and the disk duplication procedure will be invoked.

GENSYS PROGRAM DESCRIPTION

The MP/M system generation process employs the GENSYS program, which is invoked by the GENSYS command. The program is interactive and displays a table of the current system parameters. The user is prompted to provide inputs that modify the existing system parameters and establish a new system configuration.

The GENSYS program automatically builds a system program file. The file contains the assigned parameters and writes this program to the system disk so the information can be loaded into memory whenever the system is started up from a power-on or reset state.

The system configuration thus established by the GENSYS program remains effective until another GENSYS is performed by the user.

MODIFYING SYSTEM CONFIGURATION

GENSYS is invoked by typing

```
GENSYS <cr>
```

The system will respond by displaying the following table (Example #1), which reflects the current parameters assigned for the system when the last GENSYS was performed. The user is prompted to respond to each line item and to enter the appropriate parameters.

MP/M SYSTEM GENERATION

Example #1, System without Bank Switched Memory

```
0A> GENSYS

MP/M 1.1 System Generation
=====

Top page of memory = ff or 0
Number of consoles = 2 or 1
Breakpoint RST #   = 7
Add system call user stacks Y/N? y
Z80 CPU Y/N? y
Bank switched memory Y/N? n
Memory segment bases, (ff terminates list)
:<00>
:<90>
:<ff>
Select Resident System Processes: (Y/N)
ABORT   ? n
SPOOL   ? n
MPMSTAT ? y
SCHED   ? y
```

The description of each line item response entered during system generation is as follows:

(1) Top Page of Memory

Two hexadecimal ASCII digits are to be entered giving the top page of memory. A value of 0 can be entered; in this case the MP/M loader will determine the size of memory at load time by finding the top page of RAM.

(2) Number of Consoles

Each console specified requires 256-bytes of memory. The 9520 Software Development System supports up to two consoles.

(3) Breakpoint RST

The breakpoint restart number to be used by the SID (Symbolic Instruction Debugger) and DDT (Dynamic Debugging Tool) debuggers is specified. Restart 7 is for the 9520.

(4) Add System Call User Stacks (Y/N)?

If the user desires to execute CP/M *.COM files, his response should be Y. The Y response forces a stack switch with each system call from a user program. MP/M requires more stack space than CP/M.

(5) Bank Switched Memory (Y/N)?

If the user's system does not have bank switched memory, he should respond with an N. Otherwise a Y response and additional responses will be required.

(6) Memory Segment Bases

Memory segmentation is defined by the entries that are made. Care must be taken in the entry of memory bases because all entries must be made with successively higher bases.

(7) Select Resident System Processes (RSPs)

A directory search is made for all files of type RSP. Each file found is listed and included in the generated system file if the user responds with a Y.

where: ABORT: Allows the user to abort a running program

SPOOL: Allows the user to spool ASCII text files to the list device

MPMSTAT: Allows the user to display the run-time status of the operating system

SCHED: Allows the user to schedule a program for execution

After the last user response is typed, the GENSYS program produces the MPM.SYS file, which contains the new parameters assigned for the system configuration. The MPM.SYS file is then written to the system disk and is booted into memory whenever the system is started up from a power-on or reset state. The MPMLDR command can be used to load the new GENSYS program into memory; the system response is as follows:

```
0A> MPMLDR
Millennium serial number 1-236-000000
```

```
MP/M 1.1 Loader
=====
Number of Consoles = 2
Breakpoint RST #   = 7
Z80 CPU
Top of memory      = FFFFH
```

```
Memory Segment Table
SYSTEM DAT F00H 0100H
CONSOLE DAT FD00H 0200H
USERSYS STK FC00H 0100H
XIOS SPR F000H 0100H
BDOS SPR DC00H 1400H
XDOS SPR BD00H 1F00H
MPMSTAT RSP B000H 0D00H
Sched RSP AB00H 0500H
- - - - -
Memseg Usr 9000H 1B00H
Memseg Usr 0000H 9000H
```

MP/M SYSTEM GENERATION

BANK SWITCHED MEMORY CONFIGURATION

Bank switched memory is the condition in which the add-on, 48K expansion memory bank is available to operate in parallel with the base memory bank. The user can specify the segmenting of transient spaces in both memory banks.

When the system is configured with the Bank Switched Memory and Banked BDOS (Basic Operating System) File manager, the user is prompted to provide additional inputs to satisfy these configuration requirements.

This procedure requires an initial GENSYS and MPMLDR execution to determine the exact size of the operating system, followed by a second GENSYS.

The system displays the following table (Example #2) so the user can specify the bank switched memory requirements:

Example #2, System Generation with Bank Switched Memory

```
0A> GENSYS
```

```
MP/M System Generation
```

```
=====
```

```
Top page of memory = ff
Number of consoles = 2 .
Breakpoint RST # = 7
Add system call user stacks (Y/N)? y
Z80 CPU (Y/N)? y
Bank switched memory (Y/N)? y
Banked BDOS file manager (Y/N)? y
Enter memory segment table: (ff terminates list)
Base,size,attrib,bank = 0,90,0,0
Base,size,attrib,bank = 90,0A,0,0
Base,size,attrib,bank = 0,C0,0,1
Base,size,attrib,bank = ff
Select Resident System Processes: (Y/N)
ABORT ? n
SPOOL ? n
MPMSTAT ? y
SCHED ? y
```

The description of each line item response entered for Bank Switched Memory during system generation is as follows:

(1) Bank Switched Memory

The response is Y if the system hardware is configured with the add-on, 48K expansion memory.

(2) Bank Switched BDOS File Manager

A Y response includes the bank switched BDOS. This provides an additional 0C00H bytes of common area for some resident system processes (RSP). The banked BDOS is slower than the non-banked because file control blocks (FCB) must be copied from the bank of the calling program to common and then back again each time a BDOS disk function is invoked.

(3) Memory Segment Table

When bank switched memory has been specified, the user is prompted to enter base, size, attributes, and bank for each memory segment. Extreme care must be taken when making these entries as there is no error checking done by GENSYs regarding this function. The first entry made determines the bank in which the banked BDOS is to reside. Also, the bank specified in the first entry is the bank switched in at the time the MPMLDR is executed.

The attribute byte is normally defined as 00. The bank byte value is hardware dependent and is 0 for bank 0 and 1 for bank 1.

Next, the MPMLDR is executed in order to obtain the base address of the operating system.

Example #3, Base Address Assignment for Bank Switched Memory

```
0A>MPMLDR
Millennium serial number 1-236-000000
```

```
MP/M 1.1 Loader
=====
```

```
Number of consoles = 2
Breakpoint RST #   = 7
Z80 CPU
Banked BDOS file manager
Top of memory      = FFFFH
```

```
Memory Segment Table:
SYSTEM  DAT  FF00H  0100H
CONSOLE DAT  FD00H  0200H
USERSYS STK  FC00H  0100H
XIOS    SPR  F000H  0C00H
BDOS    SPR  E800H  0800H
XDOS    SPR  C900H  1F00H
MPMSTAT RSP  BC00H  0D00H
Sched   RSP  B700H  0500H
BNKBDOS SPR  A900H  0E00H
-----
```

```
Memseg  Usr  0000H  C000H  Bank 01H
Memseg  Usr  9000H  0A00H  Bank 00H
Memseg  Usr  0000H  9000H  Bank 00H
```

MP/M SYSTEM GENERATION

Using the information obtained from the initial GENSYS and MPMLDR execution, the following GENSYS and MPMLDR can be executed to reflect the new bank switched memory configuration:

0A> GENSYS

MP/M 1.1 System Generation

=====

Top page of memory = ff
Number of consoles = 2
Breakpoint RST # = 6
Add system call user stacks Y/N? y
Z80 CPU Y/N? y
Bank switched memory Y/N? y
Banked BDOS file manager Y/N? y
Enter memory segment table: (ff terminates list)
Base,size,attrib,bank = 0,9F,0,0
Base,size,attrib,bank = 9F,0A,0,0
Base,size,attrib,bank = 0,C0,0,1
Select Resident System Processes: (Y/N)
ABORT ? n
SPOOL ? n
MPMSTAT ? y
SCHED ? y

0A> MPMLDR

Millennium serial number 1-236-000000

MP/M 1.1 Loader

=====

Number of consoles = 2
Breakpoint RST # = 7
Z80 CPU
Banked BDOS file manager
Top of memory = FFFFH

Memory Segment Table

SYSTEM	DAT	FF00H	0100H
CONSOLE	DAT	FD00H	0200H
USERSYS	STK	FC00H	0100H
XIOS	SPR	F000H	0C00H
XDOS	SPR	C900H	1F00H
MPMSTAT	RSP	BC00H	0D00H
Sched	RSP	B700H	0500H
BNKBDOS	SPR	A900H	0E00H

Memseg Usr 0000H C000H Bank 01H
Memseg Usr 9F00H 0A00H Bank 00H
Memseg Usr 0000H 9F00H Bank 00H

SYSTEM DIAGNOSTIC PROGRAM

INTRODUCTION

A system diagnostic program is provided in addition to the boot PROM, self-test feature permanently installed in the 9520 Software Development System. This chapter describes the functions of the program.

The Diagnostic Disk contains software that thoroughly tests all functions of the system. In the event a subassembly fails the diagnostic test, the diagnostic program displays an error message that identifies the cause of failure.

DIAGNOSTIC EXECUTION

The user follows this procedure to execute the diagnostic program.

1. Insert the Diagnostic Disk in disk drive A or B. The contents of Diagnostic Disk are loaded into the 9520 System by entering the command, DIAG, on the console. After the diagnostic is loaded and initialized, it logs on to the display terminal and displays:

```
9520 DIAGNOSTIC VERSION 1.0
```

and then the prompt

```
0A>
```

2. The user can now communicate with the monitor and direct the flow of events by using the commands provided to invoke the various test routines. All commands have the generalized format:

```
xx<=  
xxdd...d<=  
xxdd...d:xxddd:xx<=
```

where:

```
xx = Command mnemonic  
dd = Individual data field commands  
< = Carriage return
```

The format xx<= is used when the user requires additional information to implement the command (i.e., the command SD<= displays all tests available for execution).

SYSTEM DIAGNOSTIC PROGRAM

3. The user may temporarily suspend message output (or test execution whenever a message is encountered) by pressing the console SPACE BAR. Testing and message display can be resumed by pressing the SPACE BAR again. The user may terminate test execution by entering x on the console keyboard. This command causes the diagnostic to return control to the diagnostic monitor program. The prompt OA> is displayed on the display terminal.

NOTE: There may be a delay before the monitor prompt appears after entering the x command.

```
* * * * *
* WARNING: Some of the diagnostic tests *
* write patterns on the disks. It is *
* advisable to have scratch disks *
* inserted when running these tests. *
* * * * *
```

A safe procedure for running any of the diagnostic tests is to load the CP/M or MP/M operating system, remove all disks, insert the Diagnostic Disk, and enter the command DIAG.

COMMANDS

HP (Help) Command. The help command causes a menu of valid diagnostic monitor commands to be displayed. The help command format is:

HP <=

Table 11-1 lists the response to the HP command, as displayed on the display terminal.

Table 9-1. Valid Diagnostic Commands

<u>MNEMONIC</u>	<u>DEFINITION</u>
HP	Help-Display Commands
WS	Warm Start
SD	Select Diagnostic
DR	Drive Select
LT	Set Loop-on-Test (S) Option
SM	Set No Messages Option
LE	Loop on Error
SO	Set Operator Intervention Tests Option
HE	Set Halt on Error Option
TL	Set Long Test Option
DP	Display Pass and Error Counts
CO	Continue Testing
OA> (Prompt)	

WS Command. The warm start command re-initializes the diagnostic to the default conditions with all options reset, and Disk Drives A and B selected for testing. All PASS and ERROR COUNT fields are cleared to zero.

The command format is:

WS <=

Terminal display response is:

9520 DIAGNOSTIC VERSION 1.0

0A>

SD (SELECT DIAGNOSTIC Test) Command. This command allows the operator to either queue a test (or tests) for execution or have a display of all possible tests.

The command format to display test only is:

SD <=

The display terminal responds with

ID	TEST
01-----	RAM TEST-FIXED PATTERNS
02-----	RAM TEST-ADDRESS PATTERN
03-----	SERIAL I/O PORT 1 TEST
04-----	SERIAL I/O PORT 2 TEST
05-----	SERIAL I/O PORT 3 TEST
06-----	60 Hz TIMER TEST
07-----	DISC RESTORE TEST
08-----	DISC SEEK TEST
09-----	DISC FIXED PATTERNS TEST
10-----	DISC TRACK AND SECTOR DATA TEST
11-----	DISC RANDOM TRACK-SECTOR-DATE TEST
12-----	DISC INTERRUPT NOT READY-READY TEST (S0 option must be set)
13-----	DISC INTERRUPT READY-NOT READY TEST (S0 option must be set)
14-----	DISC WRITE PROTECT TEST (S0 option must be set)

0A>

NOTE: The S0 command option must be enabled to perform tests 12, 13, and 14.

When the SD command is followed by an asterisk (*), the diagnostic monitor selects and executes all available tests.

When the SD command is followed by nn, the user selects the test (nn) for execution.

The command SD nn1 nn2 nn3 nn5 nn4...nnn <= means any number and/or sequence of tests may be queued for execution.

NOTE: If the SD command is part of a multiple command line string, it must be the last command in the line.

SYSTEM DIAGNOSTIC PROGRAM

DR (DRIVE SELECT) Command. This command allows the operator to select which disk drive(s) will be used for testing.

The command formats are:

DR A <=

Selects Drive A

"DRIVE A SELECTED"
OA>

is displayed.

DR B <=

Selects Drive B

"DRIVE B SELECTED"
OA>

is displayed.

DR * <=

Selects both Drive A and B

"DRIVE A AND B SELECTED"
OA>

is displayed.

To the following commands, the display terminal responds with the prompt OA>.

LT (LOOP on TEST(s) Command. The LT command sets the loop-on-test(s) options and causes the test(s) to be executed continuously until the operator suspends execution or an error is encountered if the Halt-on-Error option is set.

The command format is:

LT <=

SM (SUSPEND MESSAGES) Command. The SM command suspends all messages except the Diagnostic Monitor messages. This command is intended to be used in conjunction with the loop-on-test or loop-on-error options for troubleshooting purposes.

The command format is:

SM <=

LE (LOOP-ON-ERROR) Command. The LE command causes the test selected to loop on an error condition (when encountered) without changing any parameters that caused the error. This option is valid only in tests 01, 02, 03, 04, and 05.

The command format is:

LE <=

SO (SET OPERATOR intervention test(s)) Command. This command allows the execution of tests requiring operator intervention within the test(s) selected. The SO command must be set for tests 12, 13, and 14.

The command format is:

SO <=

HE (HALT on ERROR option) Command. The HE command causes the selected test execution to halt when an error is encountered. The test execution continues when the operator types " <= " on console.

The command format is:

HE <=

TL (LONG TEST option) Command. When the TL command is set, the DISC FIXED PATTERN tests are executed in long form by testing all tracks.

The command format is:

TL <=

SYSTEM DIAGNOSTIC PROGRAM

DP (DISPLAY PASS and ERROR COUNT Command. This command allows the operator to have the PASS and ERROR COUNT tally displayed on the terminal.

The command format is:

DP <=

The display terminal responds with:

```
<TEST> <PASS> <T E> <DRV A> <DRV B> <S E> <RNF E> <CRC E> <DATA E>
TEST PASSES ERRORS DRVA- DRVA- SEEK RNF- CRC- DATAL- DATAM - HARDE
01- - - 0000---00
02- - - 0000 00
03- - - 0000 00
04- - - 0000 00
05- - - 0000 00
06- - - 0000 00
07- - - 0000---00----00- - - 00- - - 00- - - - - 00
08- - - 0000---00----00- - - 00- - - 00- - - - - 00
09- - - 0000---00----00- - - 00- - - 00- -00- - - 00- - - 00- - - 00- - - -00
10- - - 0000---00----00- - - 00- - - 00- - 00- - - 00- - - 00- - - 00- - - -00
11- - - 0000---00----00- - - 00- - - 00- - 00- - - 00- - - 00- - - 00- - - -00
```

0A>

The label definitions are:

TEST	= Test ID Number
PASSES	= Total Number of Passes per Test
ERRORS	= Total Errors
DRVA	= Total Errors, Drive A only
DRVB	= Total Errors, Drive B only
SEEK	= Seek Errors
RNF	= Record-Not-Found Error
CRC	= CRC Error
DATAL	= Data Lost Error
DATAM	= Data Mis-Match Error
HARDE	= Hard Disk Errors

All counters are in Hex.

All error counters terminate count at OFF (hex).

All pass counters terminate count at OFFF (hex).

C0 (CONTINUE Execution of Suspended Test) Command - The C0 command causes the monitor to resume execution of a selected test that was terminated (suspended) by the user implementing the "X" (EXIT) function.

The command format is:

C0 <=

Test ID 01 RAM TEST - Fixed Pattern

A write/read/verify operation is performed on all available memory using these fixed patterns and a "Walking 1's" pattern:

00
FF (Hex)
55 (Hex)
AA (Hex)

If the system contains the optional expansion memory option, it is also tested:

Test ID 02 RAM TEST - Address Pattern

All available memory is written with data equal to the address HI BYTE LOW BYTE. After a period of delay, the memory is read and verified to ensure refresh is functional. Expansion memory option is tested also.

Test ID 03 SERIAL I/O Port 1 Test

Serial Port 1 is tested by enabling the wrap-back function on the 9520 control board and by writing the fixed patterns as described in Test 01 (RAM TEST). The Serial port is then read and verified. The baud rate is tested by measuring the time between character.

This test is performed using the following baud rates:

75, 110, 134.5, 150, 300, 600, 1.2k, 2.4k, 4.8k, 9.6k, 19.2k, 38.4k,
56k, 76.8k

Test ID 04 SERIAL I/O Port 2 Test

This test is the same as Port 1 with exception of different baud rates of:

75, 110, 134.5, 150, 300, 600, 1.2k, 2.4k, 4.8k, 9.6k

Test ID 05 SERIAL I/O Port 3 Test

Port 3 is tested in the same manner as Port 2.

Test ID 06 60 Hz TIMER Test

A software timing measurement is performed on the 60 Hz timer interrupt.

Disk Drive/Controller Tests

NOTE: All Disk Tests are performed on Drive A or Drive B, or both, as specified by operator input. Refer to the "DR" command.

SYSTEM DIAGNOSTIC PROGRAM

Test ID 07 DISC RESTORE Test

The floppy controller is issued a RESTORE/VERIFY command, and the controller status and track register is checked to ensure track 0 was reached and the track record was read correctly.

Test ID 08 DISC SEEK Test

Using the STEP-IN and STEP-OUT commands, the SEEK operation is performed from TRACK 0 to TRACK 76, sequentially, with a verify performed at each Track. The test then performs SEEKS in the following sequences:

TRACK 76 to 1
TRACK 1 to 75
TRACK 75 to 2, etc.
TRACK 38 to 39

Test ID 09 DISC FIXED PATTERNS Write/Read

All tracks and sectors are written and then verified with the following fixed patterns: 00; FF; 55; AA.

In the long test, the tracks are written sequentially from 0 to 76.

Test ID 10 DISC WRITE-READ TRACK/SECTOR Data Test

All tracks/sectors are written with a 2-byte "word" equal to the track and sector "address". All tracks are written sequentially from Track 0 to Track 76. After all tracks/sectors are written, the disk is read and verified.

Test ID 11 DISC RANDOM TRACK/SECTOR Data Test

This test performs a Write/Read/Verify operation to a random Track/Sector selection with Random data. This sequence is performed 256 times per test pass.

Test ID 12 DISC INTERRUPT on NOT READ/READY Test

This is an operator intervention test which verifies that an interrupt is generated when a drive goes from a NOT READY to READY state. This should occur whenever the operator places a disk in the drive-under-test and closes the drive door.

Test ID 13 DISC INTERRUPT on READY/NOT READY Test

In this test, the interrupt tested is that which is generated when the operator opens the drive-under-test door and causes the NOT READY condition (interrupt).

Test ID 14 DISC WRITE PROTECT Test

This operator intervention test requires that a write-protected disk be placed into the drive-under-test. An attempted WRITE SECTOR is performed, and a Write-Protected Status is generated. The same sector is then read to verify that the sector (data) was not written to disk.

Error Messages

RAM Tests

*** BANK x ADDRESS = xxxx DATA SB = xx IS = xx

*** RAM PARITY ERROR

SERIAL I/O Tests

*** NO SERIAL I/O RECEIVE DATA INTERRUPT
*** RECEIVE CHARACTER NOT AVAILABLE IN ALLOWED TIME
*** RECEIVE CHARACTER EARLY
*** TRANSMITTER BUFFER NOT EMPTY
*** RECEIVE DATA S/B xx IS xx
*** BAUD RATE xx (See Note)
*** SERIAL I/O PORT STATUS = xx

NOTE: Message displayed with any Serial I/O Test error.

60 Hz Timer Test

*** 60 Hz TIMER IS TOO SLOW
*** 60 Hz TIMER IS TOO FAST
*** NO 60 Hz TIMER INTERRUPT

DISC Tests

For each disk error, the controller command which produced the error is displayed. Possible command messages are:

*** RESTORE COMMAND
*** SEEK COMMAND
*** STEP-IN COMMAND
*** STEP-OUT COMMAND
*** WRITE MULTIPLE SECTOR COMMAND
*** READ MULTIPLE SECTOR COMMAND
*** WRITE SECTOR COMMAND
*** READ SECTOR COMMAND

SYSTEM DIAGNOSTIC PROGRAM

Additional possible error messages are:

*** DISC CONTROLLER STATUS = xx
*** TRACK UNDER TEST = xx (See Note 1)
*** TRACK REGISTER = xx
*** SECTOR xx DATA = xx SB = xx
*** SECTOR BYTE xx (See Note 2)
*** DATA WAS WRITTEN ON PROTECTED DISC

NOTE 1: Displayed with all Disc Errors.

NOTE 2: Displayed with all Disc (Data) Errors.

General Error Messages:

*** UNEXPECTED DISC CONTROLLER INTERRUPT
*** UNEXPECTED DMA INTERRUPT
*** UNEXPECTED GPIB INTERRUPT
*** INVALID TEST ID ***
*** INVALID COMMAND ***

INTRODUCTION

This chapter describes the 9520 Hard Disk Software Development System. The functional and operational differences between the Hard Disk and floppy disk based 9520 are included here. The chapter is divided into the following sections:

- o HARDWARE FEATURES
- o SPECIFICATIONS
- o INSTALLATION AND PREPARATION FOR USE
- o SELF TEST OPERATIONS
- o SOFTWARE FEATURES
- o SYSTEM DIAGNOSTICS

HARDWARE FEATURES

The Hard Disk system is available in two configurations: an eight inch thin-line double sided floppy disk drive and a 10 Megabyte Winchester disk drive. Other hardware features include: a processor board and external I/O board, signal cables, a hard disk I/O board, hard disk controller board and a DC power supply.

Processor Board. The printed circuit board, with a Z80A microprocessor, contains all circuits necessary to perform the data manipulations required to develop software. It is interfaced to the external I/O board which has line drivers and receivers for two way communication with the display terminal and for uploading and downloading software to a debugging system: 2 RS232, 1 RS422 and 1 IEEE 488 ports. The board has a self test feature, which will be discussed in the Self-Test Operations section.

Signal Cables. Five signal cables within the system provide the interface connections between the logic boards, the floppy disk drive and the fixed Winchester disk drive.

Hard Disk I/O Board. This board provides the electrical interface between the Z80 I/O bus on the processor board and the disk controller. It is a two layer printed circuit board mounted behind the Winchester disk drive.

Hard Disk Controller. The controller is a Xebec 5.15 inch Winchester disk controller. It provides the complete data and control interface for the hard disk. Control/status and data bytes are passed to the controller through the hard disk I/O Board, which interfaces to the processor board.

HARD DISK SYSTEM

Power Supply. The power supply furnished the +5 VDC and the +12 VDC required by the logic boards and disk drives.

SPECIFICATIONS

The physical characteristics of the hard disk system are summarized below.

Dimensions

Height - 13 5/16 inches (33.33 Cm)

Width - 19 3/16 inches (48.45 Cm)

Depth - 24 1/8 inches (61.09 Cm)

Weight - 68 pounds (30.9 Kg)

Environmental Limits

Temperature: Operating 40 to 115 degrees F (4.4 to 46 degrees C)
Non-Operating -40 to 140 degrees F (-40 to 60 degrees C)

Relative Humidity: 10 to 80% Non Condensing
Maximum Wet Bulb 78.8 degrees F (26 degrees C)

AC Power Requirements

90-130 VAC, 48-63Hz
180-260 VAC, 48-63Hz

Cooling. System cooling is provided by two AC operated fans, mounted at the rear of the enclosure, that draw cool air through the disk drives, over the power supply and across the printed circuit board. The enclosure cover must be replaced before AC power is applied. Operation without the top cover will defeat the cooling function of the fan and may cause damage to the power supply and/or printed circuit cards.

Controls and Connectors. The front and rear panels of the 9520 Hard Disk System are pictured in figures 10-1 and 10-2. Refer to these illustrations to locate the controls and connectors.

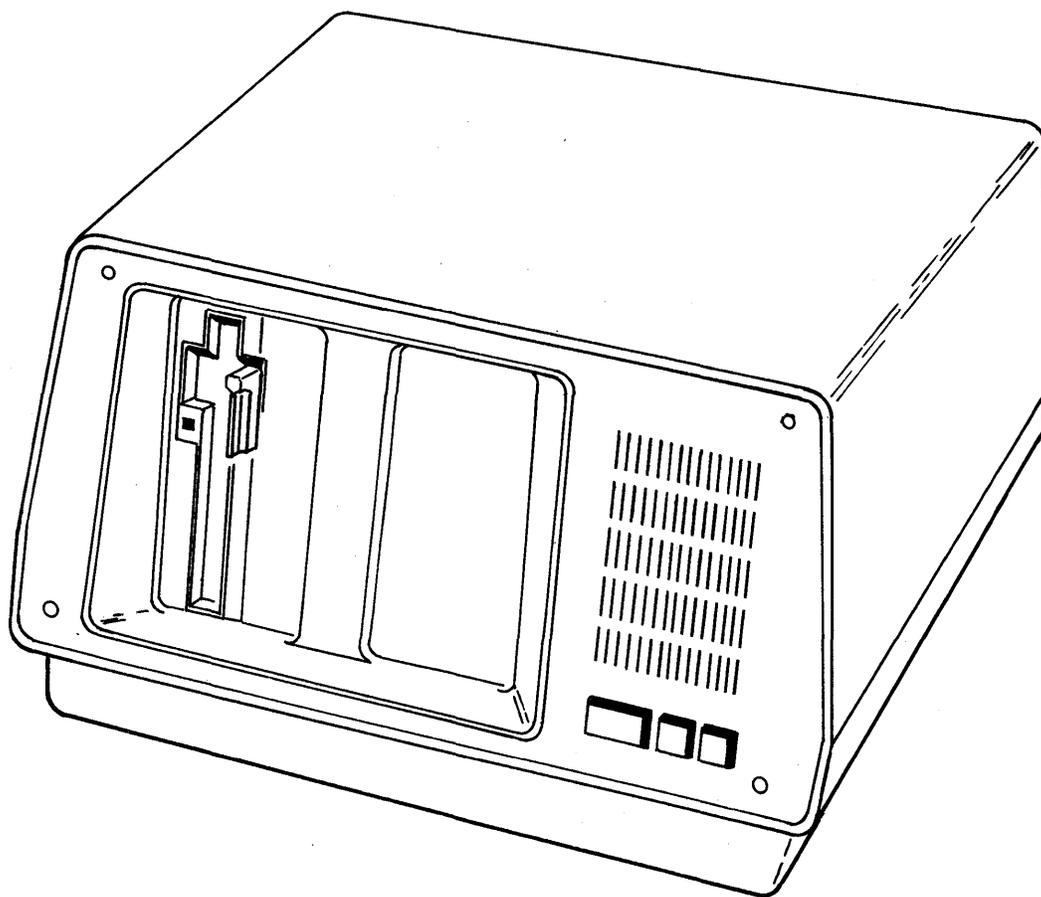


Figure 10-1. Front Panel of the 9520 Hard Disk System

HARD DISK SYSTEM

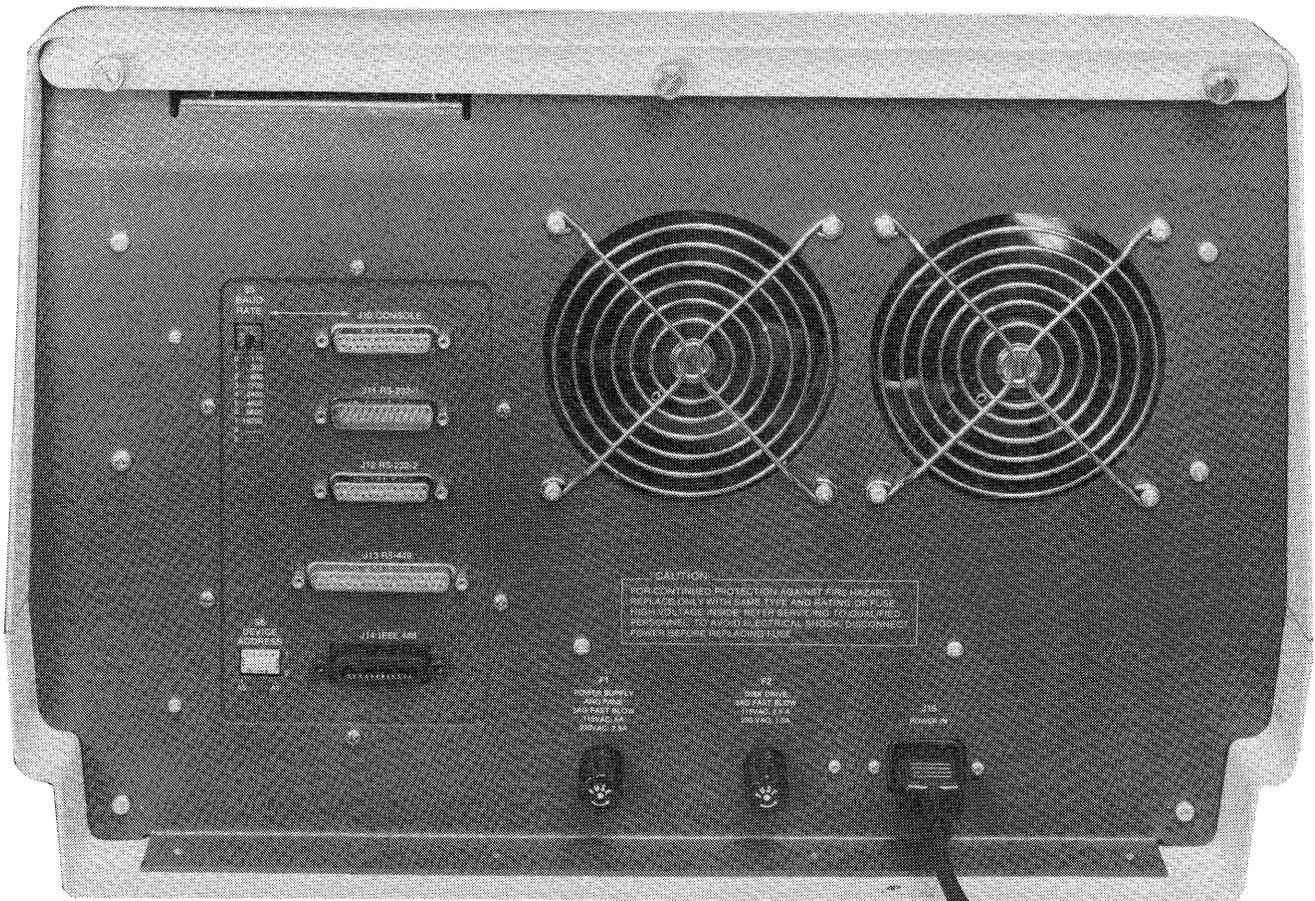


Figure 10-2. Rear Panel of the 9520 Hard Disk System

Controls and Connectors. The following controls are located on the front panel of the 9520 Hard Disk System (see figure 10-1):

POWER PUSH ON/OFF	The power on/off switch is illuminated (red) when AC power is applied to the Software Development System. When the push-switch is illuminated, this indicates not only the presence of AC voltage but also the presence of +5 VDC.
RESET	When activated, the RESET push switch reinitializes the internal microprocessor and forces the system to jump to the on-board self-test program.

The following are the rear panel ports and switches (see figure 10-2):

J10-CONSOLE	Standard RS-232C, D-type, 25-pin (DCE configuration) female connector. Connector J10 is generally connected to the Display Terminal.
J11-RS-232-1	Standard RS-232C, D-type, 25-pin (DCE or DTE configuration option) female connector. Connector J11 is generally connected to the software/hardware debug system.
J12-RS-232-2	Standard RS-232C, D-type, 25-pin (DCE configuration) female connector. Connector J12 is generally connected to the printer.
J13-RS-422	Standard RS-422A, D-type, 37-pin female connector. Connector J13 is generally connected to a unit with serial, high-speed input/output transfer requirements. (Corresponds to RS-449 configuration.)
J14-IEEE 488	This connector is a standard, 24-pin, female, high-speed parallel transfer I/O port. The port provides eight-data/address lines, eight control lines and eight signal and frame ground lines.

HARD DISK SYSTEM

- J15 POWER ON Standard AC, male, connector for power input.
- S5 BAUD RATE S5 is an eight-position rotary switch used to select the desired baud rate. The switch position is read by the system during initialization and can be changed each time the system is reset. A baud rate different from the switch position may be selected by using the BRATE utility. A function key may be programmed to execute the BRATE utility program.
- Positions:
- | | |
|---|-------|
| 0 | 110 |
| 1 | 300 |
| 2 | 600 |
| 3 | 1200 |
| 4 | 2400 |
| 5 | 4800 |
| 6 | 9600 |
| 7 | 19200 |
| 8 | ----- |
| 9 | ----- |
- (default setting when shipped with a 9501 Terminal)
- S6 DEVICE ADDRESS The DEVICE ADDRESS switch is a five-position DIP switch that controls addressing of the peripherals connected to the IEEE-488 (GPIB interface) port.

Power Supply Specifications. The 9520 power supply provides the four DC operating voltages necessary for the operation of the processor board and the disk drives. The specifications for the power supply are as follows:

AC Input

90 to 130 volts or 180 to 260 volts AC
47 to 63 Hz, single-phase
(Power Supply input is adjusted at Gould in accordance with user requirements.)

DC Outputs

<u>Voltage</u>	<u>Current</u>
+5 volts	20A max.
+12 volts	5A max.
-12 volts	3A max.
+24 volts	3.5A max.

Maximum Output Power 150 Watts

Regulation Characteristics

Line: $\pm 0.1\%$ for a full line change of 90-130 VAC or 180-260 VAC.
Load: $\pm 0.1\%$ for a 100% load change.

ORGANIZATION OF THE HARD DISK

This section defines some terms which are used later and also gives an overview of hard disks and the way in which they are viewed by CP/M. A complete understanding of this section is not necessary for the operation of the hard disk system, although it is helpful.

The hard disk can be thought of in several ways. Tracks are concentric circles, with the outermost track being Track 0. The hard disk has one or more platters, each with two surfaces. Each surface has a Track 0, Track 1, ..., etc. Cylinder 0 consists of Track 0 for all of the surfaces. Likewise, Cylinder-N consists of the collection of all Track-N's for all of the surfaces. Each track consists of 32 sectors, with each sector consisting of 256 bytes. When the hard disk format program is executed, it asks the operator to input any known bad spots on the disk. The format program accepts this information in terms of cylinders and heads, where a head corresponds to a surface.

However, the Operating System (CP/M) does not view the disk in this manner. CP/M considers all disks to be composed of tracks, with each track composed of 128 byte sectors. In order to avoid confusion, they are referred to as C-Tracks and C-Sectors.

Assuming the hard disk has 153 Cylinders, 4 surfaces, and 32 sectors (256 bytes/sector), then CP/M assumes the disk has 612 C-Tracks with each track having 64 C-Sectors. The relationship will be:

HARD DISK			CP/M	
CYLINDER	HEAD	SECTOR	C-TRACK	C-SECTOR
0	0	0	0	1 2
0	0	1	0	3 4
0	0	31	0	63 64
0	1	0	1	1 2
1	0	0	4	1 2

The first two C-tracks are allocated to the operating system. If one of these tracks is bad, then the system cannot be moved to the hard disk, and therefore, cannot be booted from the hard disk. However, the operating system can still be booted from the floppy disk. Thus, the hard disk is still usable for data files.

HARD DISK SYSTEM

Normally, the four C-tracks following the operating system tracks are used for the disk directory. (These C-Tracks are referred to as the Primary Directory Tracks.) However, if one of the primary directory tracks is bad, then the format utility searches until it finds four contiguous good C-tracks, which become the directory tracks. The first file in the directory is called badblkxx.sys, where xx represents two characters and is the binary value 0000H. The format utility creates this file, and it is used by the operating system to locate the first C-track of the directory. This enables the hard disk to be used, even if one of the primary directory tracks is bad. Likewise, if a directory track goes bad during use, it is possible to reformat the disk and rebuild it using backup files from floppy disks. It is important to keep backup files for all important files on the hard disk for this purpose.

CP/M allocates space to files in terms of blocks. Each block size is 2K. Since there are 8K bytes per C-track, there are 4 blocks or clusters per C-track. Each directory entry is used to allocate from 0 to 8 blocks. If a file's size is between 0 and 16K bytes, then only one directory is used. If it is greater than 16K, but less than 32K, then two entries are used; the second entry is an extension of the first.

Logical Disks

CP/M Version 2.2 restricts the logical disk size and the largest file on a disk to an 8 MB limit. Thus, if a physical disk is larger than 8 MB, it must be split into two or more logical disks, where each logical disk is less than or equal to 8 MB. The floppy and hard disks are organized in the following manner:

PHYSICAL HARD DISK SIZE	LOGICAL DISK AND SIZE	LOGICAL DISK AND SIZE
10 MB	A - 8 MB	C - 2 MB

The floppy disk is always logical disk B. The 10 MB hard disk is split into two logical disks; disk-A is 8 MB, and disk-C is 2 MB.

The directory for Disk-A has four tracks and room for 1024 directory entries. If Disk-C exists, its directory has two tracks and room for 512 entries.

INSTALLATION AND PREPARATION FOR USE

In order to prepare the system for normal use, it is necessary to:

- 1) Unpack the system
- 2) Connect the system
- 3) Verify the settings of the Dip Switches
- 4) Boot the Operating System from the floppy disk
- 5) Format and verify the hard disk
- 6) Move the operating system from the floppy disk to the hard disk.
- 7) Boot the operating system from the hard disk
- 8) Move the system files from the floppy disk to the hard disk.

See Chapter 3 for details concerning unpacking and connecting the system.

Installation

Verifying Dip Switches. There are two sets of Dip switches. The first set is located on the printed circuit board labeled Main Processor. This PCB is located on the right side when facing the front of the 9520. The four switches are set as follows:

SW1-1	SW1-2	SW1-3	SW1-4	MEANING
OFF	OFF	ON	ON	9520 HARD DISK. 9501 TERMINAL
OFF	OFF	ON	OFF	9520 HARD DISK. NOT A 9501

SW1-3 is used by the Boot PROM to determine where the Operating System is located. If the switch is OFF, it boots from the floppy disk. If the switch is ON, it attempts to boot from the hard disk; if it detects a failure, it boots from the floppy disk.

The second set of DIP switches are located on the PCB which is labeled 9520 Hard Disk I/O. This PCB is a small board located behind the disk drives. This Dip switch is also called SW1 and has eight switches. Switches 8 through 4 must be OFF, and the disk size is encoded using SW1-1,2,3:

DISK SIZE	3	2	1
10 MB	ON	OFF	ON

HARD DISK SYSTEM

Preparation for Use

The following sample terminal session describes how to boot the operating system, format the disk, move the operating system to the hard disk, and move the system files to the hard disk.

Booting the Operating System. Once the system has been unpacked, inspected and connected, turn on the terminal and the 9520 and follow these instructions:

Remove the Write Protect tab on the Distribution Floppy Disk, insert it into the floppy drive, and press the RESET button. This message is displayed by the Boot PROM:

```
Hard Disk : 9520 Self Test Version v.vv Completed!  
> 64 k Bytes : Hard Disk Sub-system Ready:12345  
Hard Disk Controller Boot Error  
SERIAL NUMBER s-sss-ssss  
9520, 64k CP/M vers 2.25
```

B>

where:

```
v.vv          = the version level of the Boot PROM  
s-sss-ssss   = the serial number of the user's copy of CP/M  
12345        = the number of times the Boot Prom tried to  
              boot the operating system from the hard disk  
              before declaring the Hard Disk Controller Boot  
              Error  
B>           = the floppy drive from which the system was booted
```

Formatting the Disk. Once the operating system has been loaded from the floppy disk, the FORMATHD utility can be executed to format the hard disk. Before executing this utility, look on the back panel of the 9520 or on the hard disk drive (inside the 9520) for a label which identifies any known bad spots on the hard disk. A bad spot is identified by Cylinder and Head numbers. This information is requested by the format utility.

Note that when the format utility is expecting a single character reply, such as Y or N, the reply is not followed by a carriage return, <cr>. When the format utility is expecting a number consisting of one or more digits, the number is terminated by a carriage return.

1. The user executes the format utility by giving the command `FORMATHD` as in the following example:

```
B>FORMATHD <CR>

Hard Disk Format Program. Vx.x
Formatting Disk A 8 MB
WARNING!
Format will destroy all data on the hard disk
Continue? (Y or N)Y
Are there any known bad Cylinders and Heads? (Y or N)Y
CYLINDER = 2 <CR>
HEAD = 3 <CR>
ANY MORE ENTRIES (Y/N)?N
Formatting Disk A
12345678901234567890123456789012
.....
.....
.....
.....

Formatting Disk C 2 MB
12345678901234567890123456789012
.....
.....
End of Hard Disk Format Program
```

Note the format program responded with the message:

```
Hard Disk Format Program. Vx.x
Formatting Disk A zMB
WARNING!
Format will destroy all data on the hard disk
Continue? (Y or N)
```

Where `x.x` = the version number of the format utility
`z` = the size of the disk it expects to format

If the user types `Y` or `y`, then the format program continues; otherwise, it aborts.

2. If the user elects to continue, then the format utility requests a list of any known bad spots on the disk. This list is on a label located on the rear panel of the 9520. Any bad spot located anywhere on the physical disk is reported at this time. If there are two logical disks, the user does not have to calculate the logical disk that corresponds to the bad spot. All bad spots are reported at this time. The utility asks:

Are there any known bad cylinders and heads? (Y or N)

HARD DISK SYSTEM

3. If the user answers y or Y, the format utility prompts for the Cylinder first, and then for the Head:

CYLINDER =

HEAD =

It then asks: ANY MORE ENTRIES (Y/N)?

4. If the user types y or Y, it prompts for cylinder and head. This process continues until the user types n or N. The system displays:

Formatting Disk A

5. The format routine then formats and verifies the operating system C-tracks. If an error is detected, it outputs the standard error message described below, followed by the message:

Unable to format operating system tracks
Do not move operating system to Hard Disk

This means the operating system cannot be moved to the hard disk, and the user must always boot off the floppy disk. SW1-3 on the PCB labeled 9520 Hard Disk I/O is set to OFF to tell the Boot PROM to boot from the floppy disk.

6. The format utility attempts to find 4 contiguous good C-tracks for the A-disk directory. If it cannot find four contiguous tracks, it outputs the message:

Cannot format directory. Disk is unusable.

7. The utility writes a pattern of E5H to all bytes in the directory. If an error is detected, then the message:

Error in writing directory track

is displayed, and the format utility aborts.

8. After the directory tracks have been prepared, the format utility begins to format and verify the remaining C-tracks. At this time, it displays the header:

12345678901234567890123456789012

For each 8 C-tracks, it displays a period (.) on the console. For each 256 C-tracks, it sends a carriage return, line feed to the console.

9. If there is a second logical disk, the format routine displays:

```
Formatting Disk C 2 MB
12345678901234567890123456789012
```

.....

10. When the format utility has finished successfully, it displays:

```
End of Hard Disk Format Program
```

At this point, it turns on the boot PROM and reboots the system.

11. For each error detected, the format utility displays a standard error message:

```
ERROR AT CYLINDER = cccc HEAD=hh
```

Where cccc is the cylinder number, hh is the head number.

CAUTION

All of the utilities distributed by Digital Research or Gould recognize the User Group, the Read/Only and Sys flags. However, the user is cautioned against using a non-standard erase file utility because it may not obey these flags and will inadvertently delete some or all of the badblk.sys files, thus making the disk unusable or causing bad blocks to be allocated to files.

The format program creates one or more files of the form badblkxx.sys, where xx represents two characters and is a binary number between 0 and FFFFH. The first file in the directory has a binary value of 0, and is used by the operating system during initialization to locate the directory tracks and initialize the disk parameter block. In addition, this file and files of this form are used to allocate any C-tracks identified by the user as being bad, or discovered by the format utility to be bad. Thus, it is vital that these files are not deleted by the user. In order to make it almost impossible for this to occur, the following steps have been taken:

- a. The file(s) have been placed in User 15 directory and marked as SYS files so that they will not be displayed by the DIR utility.
- b. The file(s) have been marked Read/Only so that the ERA utility will not delete them.
- c. Lower case characters have been used in the file name and the file type. Because all characters in the command line are converted to upper case, any utility that accepts a file name from the command line will be looking for a file whose name and type are in upper case; therefore, they cannot find a file with lower case characters in the file-id.

HARD DISK SYSTEM

12. The format routine can be aborted by rebooting the system using the RESET push-button on the front of the 9520.
13. Once the hard disk is formatted, the user moves the operating system onto track 0 and 1 of the hard disk by using the SYSGEN command.

Example:

```
A>SYSGEN <CR>
SYSGEN VER X.X
SOURCE DRIVE NAME (OR RETURN TO SKIP) B
SOURCE ON B, THEN RETURN <CR>
FUNCTION COMPLETE
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) A
DESTINATION ON A, THEN TYPE RETURN <CR>
FUNCTION COMPLETE
```

Now depress the RESET switch to perform a cold boot. The operating system should be loaded from the hard disk and the default disk should now be A instead of B:

```
Hard Disk : 9520 Self Test Version v.vv Completed!
> 64 k Bytes : Hard Disk Sub-system Ready:
SERIAL NUMBER s-sss-ssss
9520, 64k CP/M vers 2.25
```

A>

14. At this point, the system files can be copied from the floppy disks to the hard disk using the peripheral interface program (PIP) command:

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

SOFTWARE FEATURES

With the exception of the FDISK utility, all of the commands documented in the Chapter 5, 6, and 7 can be used on the hard disk system. The presence of the hard disk should be transparent to the user. Under no circumstances should any copy of the FDISK utility be used on the hard disk system. The hard disk system has its own utility, FLOPPYD, to format and duplicate floppy disks.

Explanation of FLOPPYD

The FLOPPYD utility is used to format, copy the operating system from one floppy disk to another, or to duplicate floppy disks. It can format the following types of disks:

```

Double Sided
  Double Density
  Single Density
Single Sided
  Double Density
  Single Density
  
```

The Copysys function is used to copy the outer two tracks from one system disk to another. Note this function can also be performed by the SYSGEN command.

The duplicate function is used to duplicate a floppy disk. This is accomplished by reading tracks from the source floppy disk and then writing the corresponding tracks on the destination disk. Since the system only has one disk drive, the duplicate function reads two tracks at a time and requests the user to exchange source and destination disks. After writing to the destination disk, it requests the user to exchange the disks again. Because a floppy disk has 77 tracks and two are copied at a time, this cycle is repeated 39 times.

It is much easier to duplicate a floppy disk by formatting the destination disk, copying the contents of the source floppy disk to an empty directory on the hard disk, and then copying those files from the hard disk to the destination floppy. The duplicate function of the FLOPPYD utility should only be used when the hard disk is not operational.

In the sample terminal session which follows, the FLOPPYD utility is used to format single and double sided disks, to copy the system from one disk to another, and to duplicate a disk. Note the user can return to the first level query by responding to any question with a Control-X. The user can abort by typing a Control-C to any query:

```

A>FLOPPYD

Ver 2.01
WARNING! This is a stand alone program.
Do not allow any other process to execute simultaneously
Reboot system if you abort with Control-C

F)ormat? D)uplicate? C)opysys? Q)uit?
F
Type of diskette S)ingle-sided? D)ouble-sided?
S
D)ouble density? S)ingle density?
D
  
```

HARD DISK SYSTEM

Put diskette to format into drive "B:", then press space bar

ffffffffffffffff 16 tracks formatted
ffffffffffffffff 32 tracks formatted
ffffffffffffffff 48 tracks formatted
ffffffffffffffff 64 tracks formatted
ffffffffffffffff 77 tracks formatted

F)ormat? D)uplicate? C)opysys? Q)uit?

F

Type of diskette S)ingle-sided? D)ouble-sided?

D

D)ouble density? S)ingle density?

D

Put diskette to format into drive "B:", then press space bar

ffffffffffffffff 16 tracks formatted
ffffffffffffffff 32 tracks formatted
ffffffffffffffff 48 tracks formatted
ffffffffffffffff 64 tracks formatted
ffffffffffffffff 77 tracks formatted

F)ormat? D)uplicate? C)opysys? Q)uit?

C

Type of diskette S)ingle-sided? D)ouble-sided?

D

NOTICE: Source and destination disk drives are the same.

Verify data via read back and compare after write? (Y/N) Y

Put source diskette into drive "B:", then press space bar

rr

Put destination diskette into drive "B:", then press space bar

ww

F)ormat? D)uplicate? C)opysys? Q)uit?

D

Type of diskette S)ingle-sided? D)ouble-sided?

D

NOTICE: Source and destination disk drives are the same.

Verify data via read back and compare after write? (Y/N) N

Put source diskette into drive "B:", then press space bar

rr

Put destination diskette into drive "B:", then press space bar

ww

Put source diskette into drive "B:", then press space bar

rr

Put destination diskette into drive "B:", then press space bar

ww

Put source diskette into drive "B:", then press space bar

.....

F)ormat? D)uplicate? C)opysys? Q)uit?

Q

NOTICE: Processing for current loading of FDISK is done,
press reset or any keyboard character to cause a cold boot.

WARNING: Depending upon your system's configuration,
this may clear your screen.

Millennium

Hard Disk : 9520 Self Test Version 2.96 Completed!

> 64 k Bytes : Hard Disk Sub-system Ready:

SERIAL NUMBER 2-236-00000

9520, 64k CP/M vers 2.25

SELF TEST OPERATIONS

At start up, a diagnostic Self-Test routine runs for approximately eight seconds. A total of 46 circuit functions are checked by the program. Each circuit function is coupled with an LED that provides status to the operator. The displayed message line spells out the word MILLENNIUM, then erases it and spells out the word C-O-M-P-L-E-T-E as each circuit function passes its test.

If no malfunction of the hardware is detected by the diagnostic, the following message is displayed at the terminal screen to notify the operator a system processor board malfunction is present at startup:

```
Hard Disk : 9520 Self Test Version v.vv Completed!
```

```
> 64 k Bytes : Hard Disk Sub-system Ready:12345
```

```
SERIAL NUMBER s-sss-ssss
```

```
9520, 64 k CP/M vers 2.25
```

Note the v.vv represents the version level of the Boot PROM and s-sss-ssss represents the serial number assigned to the users copy of CP/M.

If a malfunction is detected the displayed message is not completed, the LED associated with the test remains illuminated, and the program halts on the error. An error message is displayed.

The following pages list the possible malfunctions that may occur. In the left column is the displayed message line. At the far right is a listing of error messages that may appear on the screen with the incomplete message line. The middle two columns are information that does not appear on the screen, but may be used as a troubleshooting guide to detect a malfunction that occurs during initial start up.

HARD DISK SYSTEM

The coding used in this section is the following:

b = Known bug in code

f = Fatal error; processing stops after error message.

n = Non-fatal error. Processing continues after error message.

i = Low memory interrupt may occur during this test.

f Parity FATAL Error

f Invalid Interrupt FATAL Error

f Reader/Punch FATAL Error

f RS-449 FATAL Error

f Remote I/O FATAL Error

l = high memory interrupt may occur during this test.

n Parity Error

n Unexpected GPIB Interrupt

n Unexpected 60 Hz Interrupt

n Unexpected DMA Interrupt

f Check W-5 jumper: DMA interrupt failure FATAL Error

h = Time out loops are incorporated into the hard disk routines. If the test fails due to the time out interval being exceeded, an additional line of error message is issued.

n Select Hard Disk Time Out Error

n Request Hard Disk Time Out Error

<RAM ERROR REPORT FORMAT> There are two possible errors: First is the RAM memory error itself. Second is a failure while attempting to change the contents of the mapping RAM during the test. A RAM error can cause a mapping RAM error message:

Map x RAM Error Address = aaaa is = yy S/B = ss

Bank x RAM Error Address = aaaa is = yy S/B = ss

non-zero Mapping RAM failure yy S/B ss

where:

aaaa = Four hexadecimal digits of address

x = one hex digit for map (0 to 3) or bank (0 or 2) number

ss = two hex digits of anticipated data

yy = two hex digits of actual data read

S/B = should be

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
		1 Verify correct PROM checksum		
		2 Initialize: timer chip, interrupt controller, and console UART channel Clear screen for 9501 terminal.		
<lf>		3 Initialize remote channel		
<cr>		4 Initialize RS-449 channel		
M		5 Initialize reader/punch channel		
Mi		6 Indicates PROM checksum failure	fi	CHECK-SUM FATAL Error
		7 Initialize Hard disk controller		
Mil		8 Initialize mapped memory expansion board		
Mill		9 Set interrupt mode 2 and page pointer	i	
Mille		10 Verify stack area	fi	NO STACK FATAL Error
Millen		11 Verify timer interrupts	fi	Check W-5 jumper FATAL Error or
Millenn			fi	Timer Interrupt FATAL Error not both
		12 Verify timer interval	i	Timer Error

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
Millenni	1 off	13 Test reader/punch channel interrupts	fi	
Millenniu		14 Disable console interrupts	fi	
Millennium	2 off	15 Send hardware configuration sign on message: a. Send carriage return with no line feed to erase "Millennium" b. "Mapped memory:" or "Banked Memory:" or "" c. "Hard Disk :" if present d. "9520 " e. "Self-Test Version 2.95"		
		16 Test RS-449 UART channel interrupts	fi	
C		17 Test Remote UART channel interrupts	fi	
Co	3 off	18 Test Floppy disk drive controller interrupts	fi	Floppy Controller FATAL Error
Com	4 off	19 Test General Purpose Interface Bus controller	i	GPiB Controller Error
Comp	5 off	20 Memory test 1: map 0 8000 to ffff hex	fi	<see RAM error report>
Compl	6 off	21 Move test code to high memory		
Comple		22 run memory test 2: map 0 0 to 7fff hex	fi	<see RAM error report>
Comple	7 off			

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
Complete		23 Verify map hardware maps to unique RAM segments	fi	ss Maps to yy <cr> MAP verify FATAL Error
Completed		24 Enable parity interrupt hardware		
Completed!<cr><lf>		25 Test Direct Memory Access controller. Send carriage return and line feed to terminate first message line	fi	Check W-5 jumper: DMA Interrupt Failure FATAL
	8 off		fi	DMA Test Bad Data FATAL Error
1		26 Memory test 3 is 48k: map 1 0 to bfff hex	l	<see RAM error report>
12		27 Memory test 4 is 16k: map 1 c000 to ffff hex	l	<see RAM error report>
123		28 Memory test 5 is 4k: map 2 8000 to 8fff hex Mapped at 1000 hex	l	<see RAM error report>
1234		29 Memory test 6 is 4k: map 3 8000 to 8fff hex Mapped at 2000 hex	l	<see RAM error report>
12345		30 Memory test 7 is 60k: map 2 9000 to ffff hex and map 2 0 to 7fff hex	l	<see RAM error report>
123456		31 Memory test 8 is 60k: map 3 9000 to ffff hex and map 3 0 to 7fff hex	l	<see RAM error report>

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
		32 Report memory configuration as tested. Only one line is displayed.		
> 64k Bytes :		a. No expansion memory in system or defective hardware		
1>112K bytes :		b. Banked memory board or partially defective map board		
12>128K bytes :		c. Mapped board with only 64K RAM stuffed		
123>128K bytes :		d. failed 28 and 29		
1234>128K bytes :		e. failed 30		
1235>192K bytes :		f. Mapped board with 2 64K banks stuffed		
12345>192K bytes :		g. Mapped board with 3 banks stuffed but failed test 31		
123456>256K bytes :		h. Mapped board fully stuffed		
		33 If hard disk is in system perform controller tests. This is added on same line as expansion RAM test results. "Hard Disk Sub-system"		
		34 Run hard disk controller self test	hl	Hard Disk Controller Self-Test Error
R		35 Run hard disk controller RAM self test	hl	Hard Disk Controller RAM Self-Test Error
Re		36 Verify valid hard disk drive select switch setting	hl	Hard Disk Switch Encode Error
Rea		37 Configure hard disk controller for "installed" drive	hl	Hard Disk Controller Parameter Error
Read		38 Verify hard disk is ready	hl	Hard Disk Drive Not Ready

STATUS	LED	TEST DESCRIPTION	FLAG	ERROR MESSAGES
Ready		39 Test Hard Disk interrupts	hl	Hard Disk Interrupt Error
Ready:		40 Examine SW1-3 On = boot from hard disk. Go to 37. Off = boot from floppy drive. Go to 43.		
		41 Read cold boot loader from track 0 sect 0 of hard disk drive.	hl	Hard Disk Controller Boot Error
1		42 Examine data loaded to insure it is Gould Instruments Division universal boot loader. If so begin execution and start CPM execution. If not send retry number to console, wait 5 seconds and retry. Attempt 5 retries then error.	hl	Hard Disk Boot Error
12				
123				
1234				
12345				
<cr><lf>				
		43 Examine SW1-1 switch setting. On = to perform Floppy drive alignment routine. Off = try to perform floppy disk boot. Go to 44.		
		44 Sequentially examine drive A then B for a drive ready condition. When found go to 45.		
		45 Load one sector from track 0 sector 1 into memory.		
		46 Examine data loaded to ensure it is either standard Gould Instruments Division cold boot loader or Universal cold boot loader. If successful begin execution of loader.	fi	Not a system disk FATAL Error
<cr><lf>				

HARD DISK SYSTEM

DIAGNOSTICS

The diagnostic package provides the user with a method whereby the proper operation of the Hard Disk Software Development System can be verified when a problem with the system hardware is suspected.

Provisions are made in the diagnostic to test the CPU board, including the serial I/O and optional memory banks. Also, the floppy disk drive, hard disk drive and their associated interfaces are tested.

Disk diagnostics are destructive to data on the media; therefore, any data stored on the hard disk should be saved on a floppy disk before the hard disk diagnostics are run. Also, a formatted floppy disk with no files recorded on the media is required for the floppy disk drive tests.

Diagnostic Operating System

The diagnostic operating system provides the operator an interface with the diagnostic that enables the operator to configure the diagnostic to the options installed on the 9520 Hard Disk Development System. It allows the operator to call up individual tests for specific hardware functions, or the operator may call up multiple tests to check the entire system. The diagnostic has the capability to halt when an error condition is detected, at which time the operator may continue testing or abort the test. Also, in certain tests, the diagnostic is capable of looping on an error condition. This feature is available on the RAM tests, and is intended to allow the operator to use an oscilloscope in conjunction with the diagnostic to troubleshoot the CPU card. Normally with this procedure, an option is set to suppress messages from the terminal display, which provides a more stable sync on the oscilloscope. All commands that support these features are described in detail later in this chapter.

Hard Disk Drive Interface Switch

Before running the diagnostic package or CP/M operating system, the 9520 Hard Disk Development System has to be configured according to the manufacturer and model of hard disk drive installed. The system must also know whether to attempt booting from the hard disk.

The hard disk interface board, located directly behind the hard disk drive, contains an eight-pole, two position switch (SW1) used to match the drive-type installed in the system to the system. Switch positions 1-4 through 1-8 are presently unused and must remain in the off position in order to prevent a switch encode error message from the boot PROM.

The following table describes the various drive options available by using different settings of switch positions 1-1 through 1-3.

Table 10-1. Drive Interface Board Switch Settings

SW1 SWITCH POSITIONS			DRIVE MANUFACTURER AND MODEL	
3	2	1		
OFF	OFF	OFF	SEAGATE	ST406
OFF	OFF	ON	SEAGATE	ST506
OFF	ON	OFF	TANDON	TM602S
OFF	ON	ON	AMPEX	PYX157
ON	OFF	OFF	TANDON	TM603S
ON	OFF	ON	SEAGATE	ST412
ON	ON	OFF	AMPEX	PYX1513
ON	ON	ON	TANDON	TM603SE

Processor Board Switch. The Processor Board Switch (SW1) is a four-pole, two position DIP switch used to select various functions available on the Processor Board. The following table describes the functions that can be selected by using different settings of the SW1 switches on the processor board.

Table 10-2. Processor Board SW1 Settings

SWITCH NUMBER				SETTING SELECTION
1-1	1-2	1-3	1-4	
OFF*				BOOT SYSTEM
ON				FACTORY ALIGNMENT AID
	OFF*			SYSTEM OPERATION
	ON			FACTORY ALIGNMENT AID
		OFF		NON-HARD DISK SYSTEM
		ON		HARD DISK SYSTEM
			OFF*	9501 TERMINAL
			ON	NON-9501 TERMINAL

*Denotes Normal Switch Setting

HARD DISK SYSTEM

Diagnostic Tests

The diagnostic operating system is booted up by typing D9520HD. The diagnostic operating system is loaded, and a sign or message is displayed followed by the operating system prompt:

```
9520HD DIAGNOSTIC - VER X.XX
```

```
OA>
```

At this time, the operator may enter any desired diagnostic command. Commands follow a common format of a two character command I.D. followed by an optional data field. Not all commands use the optional data field. Commands that use the data field indicate the various options available when the data field is left open. This is intended to be used when the operator is unsure of the options available, and serves as a self help function. Commands not using the optional data field are implemented without further display, and return with the prompt. Any test in progress may be paused by pressing the space bar on the console. This causes a message to be displayed:

```
>>PAUSED - PRESS SPACE TO RESUME <<
```

After the space bar has been pressed again, operation resumes, and the following message is displayed:

```
>>
```

A test may be aborted at any time by entering x on the system console. To restart the test, use the command C0.

Note all hard disk and floppy disk diagnostic tests are destructive to data stored on the media. The operator should copy all hard disk files to a floppy disk before running the hard disk diagnostics. Before any hard disk test is run, a message is displayed to warn the operator to back-up the files from the hard disk:

```
>> WARNING << BACK-UP HARD DISK FILES BEFORE TESTING !  
>> PAUSED - PRESS SPACE TO RESUME <<
```

The test is not started until the space bar is pressed. If the hard disk files need to be copied, the CP/M operating system is loaded by pressing the front panel RESET switch on the 9520 Hard Disk Development System.

The diagnostic operating system asks the operator if the hard disk needs formatting prior to running a hard disk test:

```
>> FORMAT HARD DISK? (Y/N)
```

NOTE: This formatting is not the same as the formatting done by FORMATHD. This formatting cannot be used with CP/M.

If the operator enters "Y", the hard disk is formatted and the following message is displayed:

FORMATTING HARD DISK

The diagnostics for the hard disk use an interleave factor of 15, which is the same interleave factor used by the CP/M operating system. The format is not verified at this time; however, the format can be verified by running the FORMAT & CHECK FORMAT test. If the operator entered N when asked if the disk needs formatting, then no further format messages are displayed, unless they are related to the test being run.

If the operator formatted the hard disk with the command FD before attempting to run a hard disk test, then no prompting for the hard disk format occurs when the test is run.

Next, if the operator has not entered any hard disk bad track information into the bad track table by using the BT command, then the operator is asked to enter bad track data:

```
>> ENTER HARD DISK BAD TRACKS/SECTORS
    HIT <RETURN> ON BAD CYLINDER OF NO ENTRY.
    BAD CYLINDER =
```

If the operator does not wish to enter bad track information, then the carriage return must be pressed. Otherwise, the bad cylinder identification must be entered in hexadecimal format followed by a carriage return. Then the following message is displayed:

```
    HEAD NUMBER =
```

The bad identification head in hexadecimal format must be entered followed by a carriage return. Then the following message is displayed:

```
    BAD CYLINDER =
```

If the operator does not wish to enter any further bad track information, the carriage return must be pressed. Otherwise, the bad cylinder identification must be entered in hexadecimal format followed by a carriage return. Then the operator is prompted through each entry until the operator exits from the bad cylinder entry with a carriage return.

When the bad track table function is called up with the command BT, any entries in the bad track table are displayed prior to asking the operator if any entries are required.

HARD DISK SYSTEM

At the beginning of each test, the test number and test title are displayed. Normally, at the end of each test, a pass count and error tally table is displayed:

```
TEST PASSES ERRORS FLOPPY HARD SEEK RNF CRC DATAL DATAM HARD SIDE0 SIDE1
01---0000---00
02---0000---00
03---0000---00
04---0000---00
05---0000---00
06---0000---00
07---0000---00-----00-----00-----00-----00-----00
08---0000---00-----00-----00-----00-----00-----00-----00
09---0000---00-----00-----00---00---00---00---00---00---00---00
10---0000---00-----00-----00---00---00---00---00---00---00---00
11---0000---00-----00-----00---00---00---00---00---00---00---00
15---0000---00-----00   SEEK RNF ECC DATAL DATAM FATAL
16---0000---00-----00
17---0000---00-----00
18---0000---00-----00
19---0000---00-----00
20---0000---00-----00
21---0000---00-----00---00
22---0000---00-----00---00---00---00---00---00---00
23---0000---00-----00---00---00---00---00---00---00
```

This table can also be displayed by entering the command DP on the system console. The pass/error tally table display can be suppressed during test by entering the command SP on the system console.

Help Command (HP)

This command aids the operator in learning the diagnostic commands. When HP is entered on the system console, a table of all available commands and a brief description appended to the command is displayed.

Valid Diagnostic Commands

HP	HELP-DISPLAY COMMANDS	WS	WARM START
SD	SELECT DIAGNOSTIC	CO	CONTINUE TESTING
SM	SET NO MESSAGES	N*	RESTORE MESSAGES
LE	LOOP ON ERROR	E*	CANCEL LOOP ON ERROR
HE	HALT ON ERROR	H*	CANCEL HALT ON ERROR
SO	ALLOW OPERATOR TESTS	O*	DISALLOW OPERATOR TESTS
TL	SET LONG TEST	T*	CANCEL LONG TEST
SS	SET FLOPPY SIZE	SB	SET NUMBER OF RAM BANKS
DP	DISPLAY ERROR TABLE	BT	ENTER HARD DISK BAD TRACKS
LT	LOOP ON TEST	L*	CANCEL LOOP ON TEST
RD	RESET HARD DISK	FD	FORMAT HARD DISK
IM	INITIALIZE MAP RAM	DC	DISPLAY DURING TEST
SP	SUPPRESS ERROR TABLE		

Select Diagnostic (SD). This command allows the operator to display all tests available to be run, or to start all or any test. If the SD command is entered with the data field left open, all available tests are displayed on the console:

CPU BOARD TEST

01	RAM DATA LINES	02	RAM ADDRESS LINES
03	SERIAL IO PORT 1	04	SERIAL IO PORT 2
05	SERIAL IO PORT 3	06	60 HZ TIMER

FLOPPY DISK DRIVE TEST

07	RESTORE TEST	08	DISK SEEK TEST
09	FIXED PATTERNS	10	DISK TRACK AND SECTOR DATA
11	RANDOM TRACK-SECTOR DATA	12	INTERRUPT ON READY (01)
13	INTERRUPT ON NOT READY (01)	14	WRITE PROTECT (01)

HARD DISK DRIVE TEST

15	CONTROLLER SELF TEST	16	CONTROLLER RAM TEST
17	CONTROLLER DRIVE TEST	18	DRIVE READY TEST
19	RECALIBRATE TEST	20	FORMAT & CHECK FORMAT TEST
21	SEEK TEST	22	SEQUENTIAL WRITE/READ TEST
23	RANDOM WRITE/READ TEST	*	RUN ALL TEST

(01 INDICATES OPERATOR INTERVENTION REQUIRED)

HARD DISK SYSTEM

If the operator wishes to run all available tests, SD* is entered, and all tests are run sequentially. To run one test, SD should be entered followed by the test number (SD01 for test 01). If multiple tests are to be run, SD should be entered followed by the number for each test to be run (SD01020315 runs tests 01, 02, 03, and 15). No delimiters should be used between entries.

LOOP ON TEST (LT)

This command causes the operating system to restart the test currently selected after the test has been completed. If all tests are being run, then after the last test is completed, the first test is restarted. This is also true for running any multiple tests. This option is selected by entering LT and canceled by entering L* or executing a warm start.

SET NO MESSAGES (SM)

This command prevents error messages from being displayed on the system console. This is intended for use when the operator is using an oscilloscope to troubleshoot the system hardware, and delays created by the message I/O routines make it difficult to get a steady sync. This option is most useful when looping on an error condition. This option is selected by entering SM and can be canceled by entering N* or executing a warm start.

LOOP ON ERROR (LE)

This option causes the operating system to exercise the system hardware when an error condition is detected by continuously sending the same stimulus that caused the error condition. This option is useful in capturing intermittent system problems, and provides a stable stimulus for using an oscilloscope to locate a system hardware failure when messages are suppressed. This option is selected by entering LE and can be canceled by entering E* or executing a warm start.

HALT ON ERROR (HE)

This option is used to cause execution of the test to be stopped when an error is detected. When execution stops, the operator may abort the test by entering X or may resume testing by pressing the carriage return. This option is selected by entering HE and can be canceled by entering H* or executing a warm start.

SET LONG TEST (TL)

This option is used on the floppy disk drive and hard disk drive tests. Normally, on power up or warm start, this option is not set. When this option is not set, sequential testing of the disk drives tests only the inner, middle and outer tracks. When this option is set, all tracks are tested sequentially. The option is selected by entering TL and can be canceled by entering T* or executing a warm start.

RESET HARD DISK (RD)

This command causes a reset pulse to be sent to the hard disk controller and initializes the hard disk I/O interface card. The command does not initialize the hard disk drive parameters. The command is invoked by entering RD.

INITIALIZE MAP RAM (IM)

This command causes the map RAM on the extended memory board to be mapped to unity, or only bank 0 is mapped. This command has no effect on systems that have no extended memory option installed. The command is provided in the event the operator aborts the RAM test, during that portion of the test where RAM is mapped. The test is not able to reinitialize the mapped RAM after the abort. If the operator aborts a RAM test, the abort must be followed by this command. To initialize the map RAM, enter IM.

DISPLAY DURING TEST (DC)

This command is available only in the hard disk tests. These tests can take a considerable period of time to execute, and the operator may have no indication that the system is executing. Therefore, if this command is invoked, the current cylinder, head and sector under test is continuously displayed on the console. This command is invoked by entering DC and is only cleared via the warm start command.

WARM START (WS)

This command causes a restart of the diagnostic operating system. The pass error tally table is cleared, and all configuration commands revert to their default values. The diagnostic assumes that three banks of 64K RAM are available in extended memory for systems that have an extended memory card option installed. Also, the floppy disk is double sided, and the current disk is not a scratch disk. The hard disk is assumed to be in an unformatted state. The hard disk parameters are initialized according to the switch settings on the hard disk I/O card. The bad track table is cleared, and all options that have been previously set are cleared. The command is invoked by entering WS.

SET NUMBER OF EXTENDED MEMORY BANKS (SB)

This command does not affect testing on systems that do not have the optional extended memory card. For systems that do have the extended memory card, the diagnostic assumes that three 64K banks are available on the extended memory card. If the system under test has only one 64K bank, this command allows the number of extended memory banks to be tested to be set to one bank. This command is invoked by entering SB and causes the following messages to be displayed.

HARD DISK SYSTEM

When one bank is selected before entering SB, the following is displayed:

```
>> ONE EXTENDED MEMORY BANK SELECTED

  WHICH EXTENDED MEMORY OPTION IS INSTALLED?
  0 = 1 EXTENDED MEMORY BANK
  1 = 3 EXTENDED MEMORY BANK
  X = ABORT FUNCTION
  SELECT OPTION =
```

When three banks are selected before entering SB, the following is displayed:

```
>> THREE EXTENDED MEMORY BANKS SELECTED

  WHICH EXTENDED MEMORY OPTION IS INSTALLED?
  0 = 1 EXTENDED MEMORY BANK
  1 = 3 EXTENDED MEMORY BANK
  X = ABORT FUNCTION
  SELECT OPTION =
```

The operator must enter the appropriate number, or abort the function.

SELECT NUMBER OF SIDES ON FLOPPY DISK (SS)

This command allows the operator to test the floppy disk interface with a single sided disk. When SS is entered on the console, the current number of sides to be tested are displayed followed by a menu for changing the number of sides. The command is invoked by entering SS.

When one side is selected:

```
>> NUMBER OF SIDES ON FLOPPY = 01

  SET NUMBER OF SIDES
  0 = SINGLE SIDED DISK
  1 = DOUBLE SIDED DISK
  X = ABORT FUNCTION
  SELECT OPTION =
```

When two sides are selected:

```
>> NUMBER OF SIDES ON FLOPPY = 02

  SET NUMBER OF SIDES
  0 = SINGLE SIDED DISK
  1 = DOUBLE SIDED DISK
  X = ABORT FUNCTION
  SELECT OPTION =
```

The operator must enter the appropriate number, or abort the function.

Operator Intervention (OI)

Some tests require operator intervention in order for the test to be run. These tests are associated with the floppy disk drive, and require that the door to the drive be opened, closed, or that a write protected disk should be installed. Normally these tests are disabled, but can be enabled by entering OI. Tests that require the option are shown in the test display called up by the SD (select diagnostic) command, and are indicated by OI. This option can be cleared by entering O* or executing a warm start command.

Diagnostic Tests and Error Reporting

This section of the diagnostic specification describes in detail how each test executes, how errors are determined, and the format in which errors are reported. All error messages are preceded by a double asterisk (**). All information messages are preceded by a double right arrow (>>).

RAM TEST - DATA LINES

This test is a nondestructive data line test. The test starts at location zero in internal memory (RAM not on an extended memory card), and continues through OFFFF hex. Data is read and saved for each location. Then data from a pattern table is written to the location under test, and read back from that location. The original data is then restored to the location under test. Then a miscompare results, an error message is displayed to indicate the failed address in bank 0, the data written, and the data read back. Data patterns used to test the data lines are 00, 0FF, 0AA, 55 and a rotating logical 1. After all locations in internal RAM are tested, the diagnostic determines whether an extended memory card is installed by reading an I/O port. If the diagnostic determines that no extended memory is present, a message is displayed to indicate this, and the test ends. If an extended memory card is found, a data line test is run on the map RAM located on the extended memory card. All locations in map RAM are tested except for the location which maps the internal RAM (bank 0) to the bus for the address range where the current test being executed resides. If a miscompare results from a write/read/compare operation on the map RAM, the map RAM has been tested, the RAM on the extended memory card is tested by mapping 1K byte blocks of memory into address 8000 hex, one block at a time. Each block has each address in that block tested with a write/read/compare operation using the same data patterns as above. This is done for every 1K, byte block in a 64K bank. If the system has been configured for more than one extended memory bank, the test is executed on each bank. Any errors detected by a miscompare causes the bank number, address, data written, and data read back to be reported in an error message. Once the RAM test has been completed, the mapped RAM is mapped back to unity (bank 0 on the bus).

Error messages for this test are:

```
** BANK 0 ADDRESS XXXX WAS YY S/B ZZ  
** BANK 1 ADDRESS XXXX WAS YY S/B ZZ  
** BANK 2 ADDRESS XXXX WAS YY S/B ZZ  
** BANK 3 ADDRESS XXXX WAS YY S/B ZZ
```

HARD DISK SYSTEM

Where XXXX = the failed address
YY = the data read
ZZ = the data written

NOTE: Bank 0 refers to RAM on the CPU card. All other banks reside on the extended memory card.

RAM TEST - ADDRESS LINES

This test checks the integrity of the RAM address lines on the internal RAM (bank 0), and when an extended memory card is installed, the address lines on the mapped RAM, and the additional extended memory banks are tested. In bank 0, RAM from zero to 8FFF is cleared to zero, and then data 0AA hex is written to an address with a single address line asserted (0001 hex to start with). All RAM from address 0000 to 1FFF hex is searched for data 0AA hex, excluding the address where the data were written. If the data is found, the address where data was written to, address where data was found, and data written are reported in an error message. Then RAM is cleared, and the process is repeated with the next most significant address line being the only address line asserted. This write and search process is repeated until all address lines have been tested. If an extended memory card is installed, the mapped RAM address lines are tested, and each memory bank's address lines are also tested. The mapped RAM is tested by a similar process as before, with the exception that only nibble wide data of 0A hex is used to be mapped to the system. After the map RAM has been tested, the extended memory banks are tested. This is done by mapping a 1K byte block at address 8000 hex, and then repeating the address line test on the address range of 8000 hex to 87FF hex.

Error messages that may occur in this test are:

** BANK 0 WROTE TO XXXX FOUND AT ADDRESS YYYY DATA ZZ

And if extended memory is present.....

** WROTE TO MAP RAM ADDRESS XX FOUND DATA ZZ HEX AT MAP RAM ADDRESS YY

** BANK 1 WROTE TO XXXX FOUND AT ADDRESS YYYY DATA ZZ

** BANK 2 WROTE TO XXXX FOUND AT ADDRESS YYYY DATA ZZ

** BANK 3 WROTE TO XXXX FOUND AT ADDRESS YYYY DATA ZZ

Where XXXX or XX = address written to.
YYYY or YY = address where data found.
ZZ = data.

SERIAL I/O PORT 1

The diagnostic, serial I/O port 1 is tested by enabling the data wrap back hardware on the CPU board. The serial I/O port is configured for the desired baud rate; a software timer is then loaded, a character is sent to the I/O port to be transmitted, and the timer is started. When the timer times out, no character should be available at the receive data register. If the character is received, an error message reports the character was received early, and the current selected baud rate is also displayed. Next, another software timer is loaded and started. When the second timer times out, the serial I/O port is checked to determine data has been received. If no character has been received, an error message is reported to indicate the character is late, along with the current baud rate. If the data sent does not compare with data received, an error message reports data sent and received. If the serial I/O port does not indicate data has been transmitted, an error message is reported. If the receive buffer full does not interrupt the processor, an error message is reported.

Error messages for this test are:

- ** NO SERIAL IO RECEIVE DATA INTERRUPT
- ** RECEIVE CHARACTER NOT AVAILABLE IN ALLOWED TIME
- ** TRANSMITTER BUFFER NOT EMPTY
- ** RECEIVE DATA S/B XX IS YY
- ** BAUD RATE NNNNNN
- ** SERIAL IO PORT STATUS ZZ

Where XX = data sent
 YY = data received
 NNNNNN = baud rate
 ZZ = serial I/O port status

The following baud rates are used for this test:

75	110	134.5	150
300	600	1200	2400
4800	9600		

HARD DISK SYSTEM

SERIAL I/O PORT 2

The serial I/O port 2 is tested by enabling the data wrap back hardware on the CPU board. The serial I/O port is configured for the desired baud rate; a software timer is then loaded, a character is sent to the I/O port to be transmitted, and the timer is started. When the timer times out, no character should be available at the receive data register. If the character is received, an error message reports the character was received early, and the current selected baud rate is also displayed. Next, another software timer is loaded and started. When the second timer times out, the serial I/O port is checked to determine data has been received. If no character has been received, an error message is reported to indicate the character is late, along with the current baud rate. If data sent does not compare with data received, an error message reports data sent and received. If the serial I/O port does not indicate data has been transmitted, an error message is reported. If the receive buffer full does not interrupt the processor, an error message is reported.

Error messages for this test are:

```
** NO SERIAL IO RECEIVE DATA INTERRUPT
** RECEIVE CHARACTER NOT AVAILABLE IN ALLOWED TIME
** TRANSMITTER BUFFER NOT EMPTY
** RECEIVE DATA S/B XX IS YY
** BAUD RATE NNNNNN
** SERIAL IO PORT STATUS ZZ
```

Where XX = data sent
 YY = data received
 NNNNNN = baud rate
 ZZ = serial I/O port status

The following baud rates are used for this test:

75	110	134.5	150
300	600	1200	2400
4800	9600		

SERIAL I/O PORT 3

The serial I/O port 3 is tested by enabling the data wrap back hardware on the CPU board. The serial I/O port is configured for the desired baud rate, a software timer is then loaded, a character is sent to the I/O port to be transmitted, and the timer is started. When the timer times out, no character should be available at the receive data register. If the character is received, an error message reports the character was received early, and the current selected baud rate is also displayed. Next, another software timer is loaded and started. When the second timer times out, the serial I/O port is checked to determine data has been received. If no character has been received, an error message is reported to indicate the character is late, along with the current baud rate. If the data sent does not compare with data received, an error message reports the data sent and received. If the serial I/O port does not indicate data has been transmitted, an error message is reported. If the receive buffer full does not interrupt the processor, an error message is reported.

Error messages for this test are:

```

** NO SERIAL IO RECEIVE DATA INTERRUPT
** RECEIVE CHARACTER NOT AVAILABLE IN ALLOWED TIME
** TRANSMITTER BUFFER NOT EMPTY
** RECEIVE DATA S/B XX IS YY
** BAUD RATE NNNNNN
** SERIAL IO PORT STATUS ZZ

```

Where XX = data sent
 YY = data received
 NNNNNN = baud rate
 ZZ = serial I/O port status

The following baud rates are used for this test:

75	110	134.5	150
300	600	1200	2400
4800	9600		

HARD DISK SYSTEM

60 HZ TIMER TEST

This test checks the ability of the 60 hertz timer to interrupt the CPU, and that the 60 hertz timer runs at the correct speed. The 60 hertz timer is initialized and started simultaneously with a software timer. The 60 hertz timer should interrupt the CPU before the software timer times out. If the software timer times out, an error message is displayed to indicate no 60 hertz software timer occurred. If the interrupt does occur, a second software timer is started, and the CPU waits for another 60 hertz timer interrupt. When the second interrupt occurs, the software timer is stopped, and its contents are examined to determine if the 60 hertz timer interval was correct. If the contents of the software timer are lower than expected, an error message is displayed to indicate the 60 hertz timer ran too fast. If the contents of the software timer are higher than expected, an error message is displayed to indicate. Error messages for this test are:

```
** NO 60 HZ INTERRUPT
** 60 HZ TIMER TOO SLOW
** 60 HZ TIMER TOO FAST
```

FLOPPY DISK RESTORE TEST

This test checks that the floppy disk and the floppy disk controller respond properly to a disk restore command. This command causes the drive to seek to track 00. The CPU check the drive is not busy. If it is busy, a software timer is run, and the drive is checked again for busy status. If the drive is still busy, an error message is displayed to indicate the drive is not ready. If the drive is not busy, the restore command is sent to the controller, and the CPU waits for drive not busy. Then the controller status is then checked to determine an error was not detected. If an error was detected, the status bits are polled to determine what type of error occurred. The error type is displayed with the command that generated the error. This test is repeated for each side of the floppy disk unless a single sided disk has been inserted, and the drive is specified as a single sided drive.

FLOPPY DISK SEEK TEST

This test checks that the floppy disk drive can execute steps at a four millisecond step rate from track to track. The CPU checks that the drive is not busy, and then a step in or step out command is issued with a three millisecond delay provided by the controller, and an additional one millisecond delay provided by a software routine. The drive is sequentially stepped from track 0 to 76, and from 76 back to 0. A spiral seek test is then performed in which the drive is stepped from track 0 to 76, 76 to 1, 1 to 75 etc., to track 38. If an error occurs, the track the drive was being stepped to and the track that the drive is currently on is displayed in an error message. Also, the error status, and the command that caused the error are displayed. Error messages for this test are:

```
** STEP IN COMMAND
** STEP OUT COMMAND
   SENT TO TRACK = XX   CURRENTLY ON TRACK = YY
```

FLOPPY DISK WRITE/READ FIXED PATTERNS TEST

This test has two modes of operation. The first mode is the default mode, which verifies that the floppy disk drive can write and read the disk on tracks 00, 38, and 76. The second mode is invoked by the long test option, and verifies the floppy disk drive can write and read each track from 00 through 76. Data is written to each sector of each track to be tested. Then each sector of each track to be tested is read, and data read is compared to data written. If a miscompare results, an error message is displayed to indicate the track, sector, data written and data read. Data patterns for this test are 00, 0FF, 0AA and 055 hex. This test is repeated on each side of the disk unless the drive has been configured as a single sided drive.

FLOPPY DISK WRITE/READ TRACK-SECTOR DATA TEST

This test is similar to the fixed patterns test in the long test mode, except the disk is written with a two byte word, equal to all tracks and sectors written. A seek is issued to track 76, and all tracks and sectors are written through track 00. A seek is then issued back to track 76, and each track and sector through track 00 is read and compared with data written to that track and sector. If a miscompare results, the track, sector, data written and data read is displayed in an error message. This test is repeated on each side of the disk unless the drive has been configured as a single sided drive.

FLOPPY DISK WRITE/READ RANDOM TRACK-SECTOR DATA TEST

This test also executes a write/read verify to the floppy disk, except the track under test, sector under test, and test data are all generated by a random number generator. Error reporting is identical as in the previous two tests.

FLOPPY DISK INTERRUPT ON READY TEST

This test checks to determine an interrupt is generated when a disk is inserted into the floppy disk drive, generating a drive ready status. The drive door is opened at the start of this test, and the diagnostic prompts the operator to insert the disk, and close the door. The operator must press the carriage return after the disk has been inserted and the door has been closed. If the interrupt is not detected when the disk is inserted, an error message reports no interrupt occurred on drive ready. This test cannot be run unless the operator intervention option has been invoked.

FLOPPY DISK WRITE PROTECT TEST

This test verifies a write protected disk cannot be written to, and a write protect status is generated. A write protected disk is inserted into the drive, and an attempt to write a sector is made. The status is checked to determine a write protect status is indicated. If the status does not appear, an error message is reported to the console. Next, a read of the sector is performed, and if data attempted to be written on the sector is found, an error message is displayed to indicate a write occurred to a write protected disk. This test can be run only if the operator intervention option has been invoked.

HARD DISK SYSTEM

FLOPPY DISK DRIVE ERROR MESSAGES

The following error and status messages are associated with the floppy disk drive diagnostic tests:

```
** NO DISK CONTROLLER INTERRUPT
** DISK CONTROLLER BUSY
** DRIVE NOT READY
** UNEXPECTED DISK CONTROLLER INTERRUPT
** RESTORE COMMAND
** READ ADDRESS COMMAND
** STEP IN COMMAND
** STEP OUT COMMAND
** READ SECTOR COMMAND
** WRITE SECTOR COMMAND
** SEEK COMMAND
** READ MULTIPLE SECTOR COMMAND
** WRITE MULTIPLE SECTOR COMMAND
** FORCED INTERRUPT READY TO NOT READY COMMAND
** FORCED INTERRUPT NOT READY TO READY COMMAND
    DRIVE 0 UNDER TEST
    DRIVE 1 UNDER TEST
    TESTING SIDE 0
    TESTING SIDE 1
** SECTOR UNDER TEST = XX
** TRACK UNDER TEST = XX
** TRACK REGISTER = XX
>> CAUTION - INSERT SCRATCH DISKETTE
>> CAUTION - INSERT WRITE PROTECTED DISKETTE
** DISK CONTROLLER STATUS = XX
** SEEK TO TRACK XX CURRENTLY ON XX
```

HARD DISK CONTROLLER SELF TEST

This test sends a command to the hard disk controller to perform a self test. The diagnostic then waits for a not busy status from the hard disk controller, and then examines the controller status byte to determine if any errors occurred. This command causes the controller to perform a self test on the internal processor, data buffers, ECC circuitry, and the checksum of the program memory. If the controller self test fails, an error message indicates the error along with the status the controller sends back:

```
** CONTROLLER SELF TEST FAILED - STATUS = XX
```

HARD DISK CONTROLLER RAM TEST

This test sends a command to the hard disk controller to perform a test on the RAM resident on the controller board. The diagnostic then waits for a not busy status from the controller, at which time the status from the controller is examined to determine whether an error occurred. This command performs a data pattern test on the RAM buffers. If an error occurred, the controller status is reported in an error message indicating the controller RAM test has failed:

** CONTROLLER RAM TEST FAILED - STATUS = XX

HARD DISK CONTROLLER DRIVE TEST

This test sends a command to the hard disk controller to perform a test on the drive. This command tests the drive to controller interface. The controller sends a recalibrate and seek command to the hard disk drive, and verifies sector 0 of all the tracks on the disk. The disk must have been previously formatted to run this test. After the command is issued, the diagnostic waits for a not busy status from the controller, at which time the status is checked for an error. If an error is detected, an error message will report the failure and the controller status:

** CONTROLLER DRIVE TEST FAILED - STATUS = XX

HARD DISK DRIVE READY TEST

This test verifies the hard disk drive is ready by sending a "test drive ready" command to the hard disk controller. The controller returns an error status if the drive is not ready. If a not ready status is detected, an error message is displayed to indicate the failure and status:

** DRIVE NOT READY - STATUS = XX

HARD DISK DRIVE RECALIBRATE TEST

This test issues a recalibrate command to the hard disk controller, which causes the read/write arm on the hard disk drive to be positioned over cylinder 0. The controller returns with an error status if the recalibrate function was not successful. If an error is detected, an error message reports the failure and the status:

** RECALIBRATE ERROR - STATUS = XX

HARD DISK SYSTEM

HARD DISK FORMAT & CHECK FORMAT TEST

This test prompts the operator as to whether or not the hard disk needs to be formatted if the hard disk has not been formatted under the diagnostic before. If the operator chooses to format the disk, a format operation is performed. A verify format is then performed on the disk. The verify format function only verifies the track I.D. and interleave. If an error is detected, an error message is displayed to indicate a format error, failed cylinder, failed sector, failed read/write head and controller status:

** FORMAT ERROR - CYLINDER XXXX SECTOR XX HEAD XX - STATUS XX

If the display current option is invoked, each cylinder, sector and read/write head is displayed as the disk format is verified:

TESTING CYLINDER XXXX SECTOR XX HEAD XX

HARD DISK SEEK TEST

Four individual seek tests are actually performed. The first seek test is to track 0000. The second seek test sequentially seeks from track 0000 to the maximum track available. The third seek test sequentially seeks from the maximum track available to track 0000. And finally, the last seek test is a spiral seek test similar to the spiral seek test performed on the floppy disk drive. This last seek test seeks to track 0000, then the maximum track, then track 0001, then the maximum track minus one, etc., until the center track is reached. The drive must be formatted before this test can be run. After each individual seek operation, the controller status is examined to determine whether an error occurred during the seek. If an error occurred, a message reports a seek error, the failed cylinder, sector, head and controller status:

** SEEK ERROR TO CYLINDER XXXX SECTOR XX HEAD XX - STATUS XX

If the display current option is invoked during this test, the cylinder, sector, and write/read head are displayed for each seek:

HARD DISK SEQUENTIAL WRITE/READ TEST

This test sequentially writes, reads and verifies each sector and track with a fixed data pattern. Each sector is tested with two fixed data patterns of AA55 hex, and 55AA hex. Starting at track 0000 and sector 00, a fixed data pattern is first written to the disk, and then read back. A compare is made between data written to the disk, and data read back. If a miscompare is detected, the cylinder, sector, head, data written to the disk and data read from the disk are displayed in an error message.

** CYLINDER XXXX SECTOR XX HEAD XX DATA WAS XX DATA S/B XX

An exception to the error display is if the cylinder, sector, and head are found in the bad track table, the error is not displayed. If an error occurs on a cylinder, track and head not found in the bad track table, these parameters are entered into the bad track table automatically after the error has been displayed. After all sectors on the current track are tested, the next track is tested until all tracks and sectors have been verified. If the display current option has been invoked, the logical address is displayed for each new track and sector under test.

HARD DISK RANDOM WRITE/READ TEST

This test verifies 1000 random tracks and sectors using random data. Starting at a random track and sector, a random data pattern is first written to the disk, and then read back. A compare is made between data written to the disk, and data read back. If a miscompare is detected, the cylinder, sector, head, data written to the disk and the data read from the disk are displayed in an error message:

** CYLINDER XXXX SECTOR XX HEAD XX DATA WAS XX DATA S/B XX

An exception to the error display is if the cylinder, sector, and head are found in the bad track table, the error is not displayed. If an error occurs on a cylinder, track and head not found in the bad track table, these parameters are entered into the bad track table automatically after the error has been displayed. After the current track and sector is tested, the next random track and sector is tested, until 1000 tracks and sectors have been verified. If the display current option has been invoked, the logical address is displayed for each new track and sector under test:

HARD DISK ERROR AND STATUS MESSAGES

** ATTEMPTING TO WRITE COMMAND - I/O IN WRONG MODE - STATUS XX
(this error indicates a problem sending commands to the controller)

** STATUS - XX
(indicates status byte from controller)

** BAD CYLINDER TABLE FULL
(indicates the bad cylinder table can except no more entries)

** WRITE SECTOR FUNCTION ERROR
(indicates the controller detected an error when writing a sector)

** READ SECTOR FUNCTION ERROR
(indicates the controller detected an error when reading a sector)

** CHECK TRACK FORMAT FUNCTION ERROR
(indicates the controller detected an error when checking format)

9520 SYSTEM COMMAND SUMMARY

The following is a list of 9520 System Commands, with syntax, both in alphabetical order, as well as in functional grouping. A detailed description of most commands is in chapter 5. Each command is entered in response to the system prompt >, and ends with a carriage return. The system prompt is preceded by a drive identifier CP/M and MP/M (e.g. A>) and a console number when using MP/M (e.g. 0A>). The following symbols and conventions are used to describe command syntax:

{ } are used to enclose a required parameter, meaning that if it is not entered, the system will respond either with an error message or attempt to execute.

[] are used to enclose an optional parameter. If a parameter is not entered the system will invoke a fixed default value, use the last entered by the operator, or simply fail to perform the function represented by that parameter.

{ T1 }
 T2 } are stacked parameters, indicating that one of the parameters must be
 Both } chosen.

9520 SYSTEM COMMAND SUMMARY

ALPHABETICAL LISTING

COMMAND NAME

SYNTAX

ABORT {ABORT programname}

ABORT {ABORT programname} {console number}

A cross-assembler

A { 8085
8048
6800
6801
6809
Z80 } [drive:] {filename}

[A
B
.
.
.
D] [A
B
.
.
D
Z] [A
B
.
.
D
T
X
Z] [A
B] [] [] []

ASM

ASM [drive:] {filename}

[.A
.B
.
.
.
.P] [A
B
.
.
.
P
Z] [A
B
.
.
.
P
X
Z] [] []

ATTACH {filename}

BRATE {device=brate}

CONSOLE

CONVERT drive: { output-filename
.BIN
.HEX
.TEK }

drive: { input-filename
.BIN
.HEX
.OBJ
.TEK }

ALPHABETICAL LISTING (Cont'd)

<u>COMMAND NAME</u>	<u>SYNTAX</u>
DIAG	DIAG CO DP DR drive HE HP LE LT SD testno 01 testno 02 ... testno 14 SM SO TL WS
DDT	DDT [drive:] {filename}
DIR	DIR [filename] [filematch]
DSKRESET	DSKRESET [drive]
DOWNLOAD	DOWNLOAD [drive:] {filename}
DUMP	DUMP [drive:] {filename} [.filetype]
ED	ED [drive:] {filename} .filetype
ERA	ERA {file-id} {filematch}
ERAQ	ERAQ {file-id} {filematch}
FDISK	FDISK F(ormat) D(uplicate) C(opy system) Q(uit)
GENHEX	GENHEX program-name.COM offset
GENMOD	GENMOD primary-file-name.hex primary-file-name.prl [\$additional-memory]
GENSYS	GENSYS
LOAD	LOAD filename

ALPHABETICAL LISTING

COMMAND NAME

SYNTAX

MLINK filename=filename filename ... $\left[\begin{array}{l} >A \\ B \\ \cdot \\ \cdot \\ \cdot \\ Z \end{array} \left[\begin{array}{l} A \\ B \\ \\ \\ P \\ Z \end{array} \right] \right]$

MLINK LOG MAP
 NOLOG NOMAP LOAD LOCATE
 TRANSFER END

MLINK @primary-filename.IND

MPMSTAT MPMSTAT

PIP $\left\{ \begin{array}{l} \text{drive:} \\ \text{destination-file} \\ \text{drive:destination-file} \end{array} \right\} = \left\{ \begin{array}{l} \text{drive:} \\ \text{source-file} \\ \text{drive:source file} \end{array} \right\} \text{parameter}$

$\left[\begin{array}{l} \text{drive:} \\ \text{source-file} \\ \text{drive:source-file} \end{array} \right] \text{ [parameters] ... }$

PRLCOM {primary-filename.PRL}{primary-filename.COM}

REN {new-file-id}={old-file-id}

SCHED {mm/dd/yy hh:mm}{Filename}

SPOOL file-id ,file-id ...

STAT drive: filename R/O
 R/W

STAT DEV:

STAT drive:DSK:

STAT drive:=R/O

STAT logical-device:=physical-device:

STAT USR:

STAT VAL:

ALPHABETICAL LISTING (Cont'd)

<u>COMMAND NAME</u>	<u>SYNTAX</u>
STOPSPLP	STOPSPLP {terminal-number}
SUBMIT	SUBMIT {filename.SUB} P1 P2 ...
TOD	TOD {mm/dd/yy hh:mm:ss}
TYPE	TYPE file-id
UPLOAD	UPLOAD [drive:] {filename.BIN}
USER	USER [user-number]

FUNCTIONAL LISTING

User Identification Commands

CONSOLE	CONSOLE	Displays the console ID number at the location where the command is entered.
DSKRESET	DSKRESET	Allows the operator to change disks.
USER	USER [user-number]	Displays the current user code and permits the user to set a user code (number).

File Manipulation Commands

DIR	DIR	Lists all files found that match the filename or filematch.
ERA	ERA	Deletes a file or a group of files.
ERAQ	ERAQ	Selectively deletes files by prompting user for Yes/No.
GENHEX	GENHEX	Produces a file of type HEX from a file of type COM.
GENMOD	GENMOD	Produces a file of type PRL (Page Relocatable) from two concatenated files of type HEX.
PRLCOM	PRLCOM	Produces a file of type COM (which is not relocatable) from a file of type PRL (page relocatable).
REN	REN	Allows the user to rename a file.
TYPE	TYPE	Displays the contents of a specified ASCII file on the screen.

FUNCTION LISTING (Cont'd)

System Operation Commands

ASM	A)sm	Assembles the users program.
DUMP	D)ump	Displays the contents of a specified disk file on the console device in hexadecimal format.
LOAD	LOAD	Loads a specified disk file containing hexadecimal code and produces an executable memory image file.
MPMSTAT	MPMSTAT	Displays the current status of the MP/M operating system.
PIP	PIP	Transfers data files from one peripheral I/O device to another; inter-disk transfers are permitted.
SPOOL	SPOOL	Queues files in preparation for output usually to a printer.
STAT	S)tatus	Displays status and/or changes device assignments.
STOPSPLP	STOPSPLP	Detaches the spooler from a specified terminal.
SUBMIT	SUBMIT	Selects a file of commands for automatic batch processing.

Program Operation Commands

ABORT	ABORT	Allows aborting of a running program.
DDT	DDT	Invokes the dynamic debugger utility.
ED	ED)it	Invokes the MP/M test editor utility.
SCHEDULE	SCHEDULE	Schedules a program to run at a particular time and date.
TOD	TOD	Examines and sets the system date and time parameter.

9520 SYSTEM COMMAND SUMMARY

FUNCTIONAL LISTING (Cont'd)

System Utilities

BRATE	BRATE	Allows the baud rate to be displayed and set.
CONVERT	CONVERT	Converts Millennium Binary files to Intel Hexadecimal, Tektronix object code, or Tektronix Hexadecimal code. The conversions are interchangeable.
DOWNLOAD	DOWNLOAD	Downloads a 9520 development disk based program to a 9508 MicroSystem Emulator.
FDISK	FDISK	Allows a user to format, duplicate, or copy a disk.
UPLOAD	UPLOAD	Uploads a program from the 9508 MicroSystem Emulator to a disk file on the 9520 Software Development System System.

MPM Commands

ATTACH	ATTACH	Attaches a previously detached program to a terminal.
GENSYS	GENSYS	Automates the generation of a system by prompting the user.

Gould Millennium Support Packages Commands

A	cross-assembler	Assembles a source language program into an object program for either execution or linking.
DIAG	DIAG	Executes the Gould Millennium diagnostic program. The program is supplied on a disk.
MLINK	MLINK	Links relocatable object modules into a single executable module.

 HEXADECIMAL AND DECIMAL CONVERSION

ASCII CODE CONVERSION TABLE

		HEXADECIMAL							
		MOST SIGNIFICANT CHARACTER							
		0	1	2	3	4	5	6	7
LEAST SIGNIFICANT CHARACTER	0	NUL	DLE	SP	0	@	P	'	p
	1	SOH	DC1	!	1	A	Q	a	q
	2	STX	DC2	"	2	B	R	b	r
	3	ETX	DC3	#	3	C	S	c	s
	4	EOT	DC4	\$	4	D	T	d	t
	5	ENQ	NAK	%	5	E	U	e	u
	6	ACK	SYN	&	6	F	V	f	v
	7	BEL	ETB	'	7	G	W	g	w
	8	BS	CAN	(8	H	X	h	x
	9	HT	EM)	9	I	Y	i	y
	A	LF	SUB	*	:	J	Z	j	z
	B	VT	ESC	+	;	K	[k	}
	C	FF	FS	,	<	L	\	l	~
	D	CR	GS	-	=	M]	m	}
	E	SO	RS	.	>	N	^	n	~
	F	SI	US	/	?	O	_	o	DEL

EXAMPLES

W = 57

H = 48

a = 61

t = 74

@ = 40

NUL = 00

DEL = 7F

HEXADECIMAL AND DECIMAL CONVERSION

Decimal-Hexadecimal-Binary Equivalents 0-255₁₀

Decimal	Hexadecimal	Binary 8-bit Code									
0	00	0000 0000	64	40	0100 0000	128	80	1000 0000	192	C0	1100 0000
1	01	0000 0001	65	41	0100 0001	129	81	1000 0001	193	C1	1100 0001
2	02	0000 0010	66	42	0100 0010	130	82	1000 0010	194	C2	1100 0010
3	03	0000 0011	67	43	0100 0011	131	83	1000 0011	195	C3	1100 0011
4	04	0000 0100	68	44	0100 0100	132	84	1000 0100	196	C4	1100 0100
5	05	0000 0101	69	45	0100 0101	133	85	1000 0101	197	C5	1100 0101
6	06	0000 0110	70	46	0100 0110	134	86	1000 0110	198	C6	1100 0110
7	07	0000 0111	71	47	0100 0111	135	87	1000 0111	199	C7	1100 0111
8	08	0000 1000	72	48	0100 1000	136	88	1000 1000	200	C8	1100 1000
9	09	0000 1001	73	49	0100 1001	137	89	1000 1001	201	C9	1100 1001
10	0A	0000 1010	74	4A	0100 1010	138	8A	1000 1010	202	CA	1100 1010
11	0B	0000 1011	75	4B	0100 1011	139	8B	1000 1011	203	CB	1100 1011
12	0C	0000 1100	76	4C	0100 1100	140	8C	1000 1100	204	CC	1100 1100
13	0D	0000 1101	77	4D	0100 1101	141	8D	1000 1101	205	CD	1100 1101
14	0E	0000 1110	78	4E	0100 1110	142	8E	1000 1110	206	CE	1100 1110
15	0F	0000 1111	79	4F	0100 1111	143	8F	1000 1111	207	CF	1100 1111
16	10	0001 0000	80	50	0101 0000	144	90	1001 0000	208	D0	1101 0000
17	11	0001 0001	81	51	0101 0001	145	91	1001 0001	209	D1	1101 0001
18	12	0001 0010	82	52	0101 0010	146	92	1001 0010	210	D2	1101 0010
19	13	0001 0011	83	53	0101 0011	147	93	1001 0011	211	D3	1101 0011
20	14	0001 0100	84	54	0101 0100	148	94	1001 0100	212	D4	1101 0100
21	15	0001 0101	85	55	0101 0101	149	95	1001 0101	213	D5	1101 0101
22	16	0001 0110	86	56	0101 0110	150	96	1001 0110	214	D6	1101 0110
23	17	0001 0111	87	57	0101 0111	151	97	1001 0111	215	D7	1101 0111
24	18	0001 1000	88	58	0101 1000	152	98	1001 1000	216	D8	1101 1000
25	19	0001 1001	89	59	0101 1001	153	99	1001 1001	217	D9	1101 1001
26	1A	0001 1010	90	5A	0101 1010	154	9A	1001 1010	218	DA	1101 1010
27	1B	0001 1011	91	5B	0101 1011	155	9B	1001 1011	219	DB	1101 1011
28	1C	0001 1100	92	5C	0101 1100	156	9C	1001 1100	220	DC	1101 1100
29	1D	0001 1101	93	5D	0101 1101	157	9D	1001 1101	221	DD	1101 1101
30	1E	0001 1110	94	5E	0101 1110	158	9E	1001 1110	222	DE	1101 1110
31	1F	0001 1111	95	5F	0101 1111	159	9F	1001 1111	223	DF	1101 1111
32	20	0010 0000	96	60	0110 0000	160	A0	1010 0000	224	E0	1110 0000
33	21	0010 0001	97	61	0110 0001	161	A1	1010 0001	225	E1	1110 0001
34	22	0010 0010	98	62	0110 0010	162	A2	1010 0010	226	E2	1110 0010
35	23	0010 0011	99	63	0110 0011	163	A3	1010 0011	227	E3	1110 0011
36	24	0010 0100	100	64	0110 0100	164	A4	1010 0100	228	E4	1110 0100
37	25	0010 0101	101	65	0110 0101	165	A5	1010 0101	229	E5	1110 0101
38	26	0010 0110	102	66	0110 0110	166	A6	1010 0110	230	E6	1110 0110
39	27	0010 0111	103	67	0110 0111	167	A7	1010 0111	231	E7	1110 0111
40	28	0010 1000	104	68	0110 1000	168	A8	1010 1000	232	E8	1110 1000
41	29	0010 1001	105	69	0110 1001	169	A9	1010 1001	233	E9	1110 1001
42	2A	0010 1010	106	6A	0110 1010	170	AA	1010 1010	234	EA	1110 1010
43	2B	0010 1011	107	6B	0110 1011	171	AB	1010 1011	235	EB	1110 1011
44	2C	0010 1100	108	6C	0110 1100	172	AC	1010 1100	236	EC	1110 1100
45	2D	0010 1101	109	6D	0110 1101	173	AD	1010 1101	237	ED	1110 1101
46	2E	0010 1110	110	6E	0110 1110	174	AE	1010 1110	238	EE	1110 1110
47	2F	0010 1111	111	6F	0110 1111	175	AF	1010 1111	239	EF	1110 1111
48	30	0011 0000	112	70	0111 0000	176	B0	1011 0000	240	F0	1111 0000
49	31	0011 0001	113	71	0111 0001	177	B1	1011 0001	241	F1	1111 0001
50	32	0011 0010	114	72	0111 0010	178	B2	1011 0010	242	F2	1111 0010
51	33	0011 0011	115	73	0111 0011	179	B3	1011 0011	243	F3	1111 0011
52	34	0011 0100	116	74	0111 0100	180	B4	1011 0100	244	F4	1111 0100
53	35	0011 0101	117	75	0111 0101	181	B5	1011 0101	245	F5	1111 0101
54	36	0011 0110	118	76	0111 0110	182	B6	1011 0110	246	F6	1111 0110
55	37	0011 0111	119	77	0111 0111	183	B7	1011 0111	247	F7	1111 0111
56	38	0011 1000	120	78	0111 1000	184	B8	1011 1000	248	F8	1111 1000
57	39	0011 1001	121	79	0111 1001	185	B9	1011 1001	249	F9	1111 1001
58	3A	0011 1010	122	7A	0111 1010	186	BA	1011 1010	250	FA	1111 1010
59	3B	0011 1011	123	7B	0111 1011	187	BB	1011 1011	251	FB	1111 1011
60	3C	0011 1100	124	7C	0111 1100	188	BC	1011 1100	252	FC	1111 1100
61	3D	0011 1101	125	7D	0111 1101	189	BD	1011 1101	253	FD	1111 1101
62	3E	0011 1110	126	7E	0111 1110	190	BE	1011 1110	254	FE	1111 1110
63	3F	0011 1111	127	7F	0111 1111	191	BF	1011 1111	255	FF	1111 1111

ASCII CHARACTER SET

ASCII CHARACTER SET

HEX	CHAR.	HEX	CHAR.	HEX	CHAR.	HEX	CHAR.
00	NUL	20	SP	40	@	60	
01	SOH	21	!	41	A	61	a
02	STX	22	"	42	B	62	b
03	ETX	23	#	43	C	63	c
04	EOT	24	\$	44	D	64	d
05	ENO	25	%	45	E	65	e
06	ACK	26	&	46	F	66	f
07	BEL	27	'	47	G	67	g
08	BS	28	(48	H	68	h
09	HT	29)	49	I	69	i
0A	LF	2A	*	4A	J	6A	j
0B	VT	2B	+	4B	K	6B	k
0C	FF	2C	,	4C	L	6C	l
0D	CR	2D	-	4D	M	6D	m
0E	SO	2E	.	4E	N	6E	n
0F	SI	2F	/	4F	O	6F	o
10	DLE	30	0	50	P	70	p
11	DC1	31	1	51	Q	71	q
12	DC2	32	2	52	R	72	r
13	DC3	33	3	53	S	73	s
14	DC4	34	4	54	T	74	t
15	NAK	35	5	55	J	75	u
16	SYN	36	6	56	V	76	v
17	ETB	37	7	57	W	77	w
18	CAN	38	8	58	X	78	x
19	EM	39	9	59	Y	79	y
1A	SUB	3A	:	5A	Z	7A	z
1B	ESC	3B	;	5B	[7B	{
1C	FS	3C	<	5C	\	7C	
1D	GS	3D	=	5D]	7D	}
1E	RS	3E	>	5E	^	7E	~
1F	US	3F	?	5F	-	7F	DEL

DEFINITIONS FOR FORMATTING AND CONTROL CHARACTERS

CHARACTER	DEFINITION	CHARACTER	DEFINITION
NUL	NULL	SO	SHIFT OUT
SOH	START OF READING; ALSO START OF MESSAGE	SI	SHIFT IN
STX	START OF TEXT; ALSO EOA, END OF ADDRESS	DLE	DATA LINK ESCAPE
ETX	END OF TEXT; ALSO EOM, END OF MESSAGE	DC1	DEVICE CONTROL 1
EOT	END OF TRANSMISSION (END)	DC2	DEVICE CONTROL 2
ENQ	ENQUIRY (ENQRY): ALSO WRU	DC3	DEVICE CONTROL 3
ACK	ACKNOWLEDGE. ALSO RU	DC4	DEVICE CONTROL 4
BEL	RINGS THE BELL	NAK	NEGATIVE ACKNOWLEDGE
BS	BACKSPACE	SYN	SYNCHRONOUS IDLE (SYNC)
HT	HORIZONTAL TAB	ETB	END OF TRANSMISSION BLOCK
LF	LINE FEED OR LINE SPACE (NEW LINE): ADVANCES PAPER TO NEXT LINE	CAN	CANCEL (CANCL)
VT	BEGINNING OF LINE	EM	END OF MEDIUM
FF	VERTICAL TAB (VTAB)	SUB	SUBSTITUTE
CR	FORM FEED TO TOP OF NEXT PAGE (PAGE)	ESC	ESCAPE. PREFIX
	CARRIAGE RETURN	FS	FILE SEPARATOR
		GS	GROUP SEPARATOR
		RS	RECORD SEPARATOR
		US	UNIT SEPARATOR
		SP	SPACE

SCHEMATIC DIAGRAMS

This appendix contains schematic diagrams for the 9520 Software Development System. The diagrams are subject to change and therefore are to be used only for reference purposes. The following is a list of the drawings:

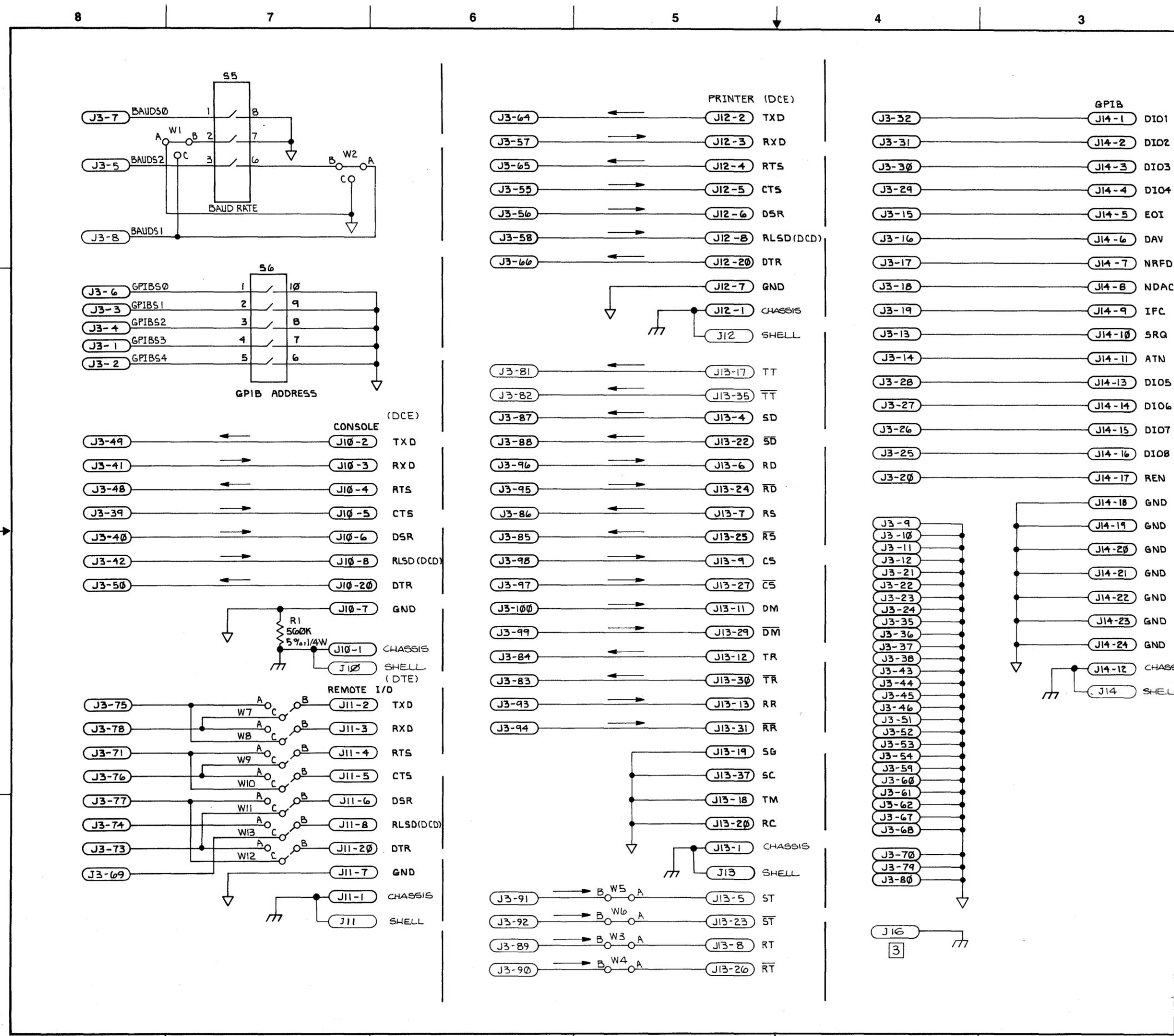
NAME	DRAWING NUMBER
9520 Connector Panel Schematic	12000227
Processor Schematic	12000228
9520 Memory Expansion Schematic	12000266
9520 Hard Disk I/O Schematic	12000276
Map/RAM Schematic	12000279
9520 Hard Disk Wiring Schematic	73000005

DWG. NO. 12000227		SH 1	REV C	1
REVISIONS				
ZONE	REV.	DESCRIPTION	DATE	APPROVED
	01	PROTOTYPE		
	02			
	50	PILOT RELEASE/ECO 2493	3/10/81	
	A	PROD. RELEASE/ECO 2565	5/22/81	
	B	REV PER ECO# 2643	8/27/81	
	C	ADD CHASSIS CONNECTION TO A.C. HARNESS PER ECO# 3284.	1/18/82	

- NOTES: UNLESS OTHERWISE SPECIFIED.
- CONN J3 REPRESENTS PIN FOR PIN P3 ON SCHEMATIC 12000228.
 - REMIO IS DTE WHEN W7 THRU 13 IS AB. REMIO IS DCE WHEN W7 THRU 13 IS BC.
 - J16 MATES WITH P16 ON AC HARNESS.

	W14				
	J1				
	S4				
	S3				
	S2				
	S1				
	J9				
	J8				
	J7				
	R1	J6			
	W14	J5			
	S6	J4			
	J14	J2			

13000227	9520					
NEXT ASSY	USED ON	LAST USED	NOT USED	TYPE	REF DES	QTY
FIRST APPLICATION		REF DESIGNATIONS		SPARE GATES		
Millennium Systems Inc.						
SCHEMATIC						
CONN. PANEL 9520						
APPROVALS		DATE				
DRAWN B. MARTIN		11-21-80				
CHECKED [Signature]		[Signature]				
ENGR						
SIZE D	DWG. NO. 12000227		REV C		SHEET 1 OF 1	

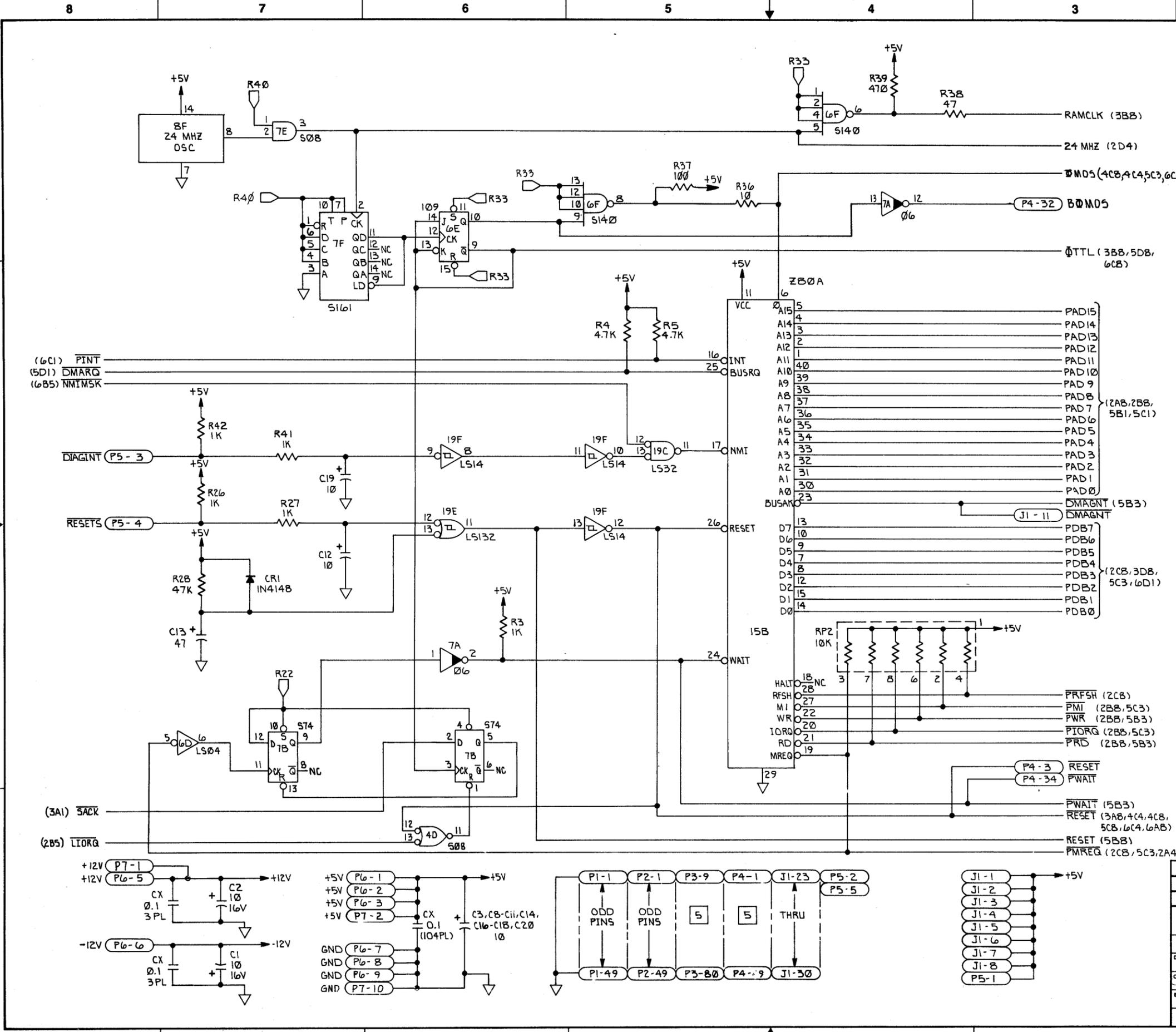


DWG. NO. 12000228		REV. K	
REVISIONS			
ZONE	REV.	DESCRIPTION	DATE
	0?	PROTOTYPE	
	50	PILOT RELEASE, ECO 2491	
	A	PER ECO # 2567	5/21/81
	B	PER ECO # 2657	11/18/81
	C	PER ECO # 2708	12/11/81
	D	PER ECO # 2730	1/11/82
	E	PER ECO # 2818	3/11/82
	F	MODIFY FOR 9520 HD PER ECO 3218	6/24/82
	G	INCREASE POWER-UP RESET BY CHANGING C19 TO 47uF PER ECO # 3315	1/20/83
	H	INVERT P4 CLOCK FOR H.D. PER ECO 3335	1/14/83
	J	PER ECO # 3398	9/8/82
	K	PER ECO 3403	4/14/82

- NOTES: UNLESS OTHERWISE SPECIFIED.
1. RESISTANCE VALUES ARE IN OHMS, 1/4W, 5%.
 2. CAPACITANCE VALUES ARE IN MICROFARADS.
 3. SYMBOL \square INDICATES A COMMON +5V PULL-UP RESISTOR CONNECTED TO MORE THAN ONE I.C. DEVICE. VALUE 2.2K
 4. I.C.'S ARE 7400 FAMILY.
- 5 THE FOLLOWING CONNECTOR PINS ARE TIED TO GROUND.
P3: 9-12, 21-24, 35-38, 43-46, 51-54, 59-62, 67, 68, 70, 79-80
P4: 1, 5-49 ODD PINS ONLY
- 6 P7 WILL BE INTERCONNECTED TO ATTACHED DATA SEPARATOR PCB. REFERENCE SCHEMATIC 12000263.
- 7 WB SHORTED FOR STANDARD SS DRIVE. WB OPEN FOR THIN LINE.

REF	DESIGNATION	TYPE	QTY
	26LS31	1G	2
	9602	21E	1
	B1LS97	22G	1
	B1LS97	5C	4
	LS109	20E	1
	S140	6F	1
	1489	1C	1
	1489	1D	1
	LS132	19E	3
	LS32	19C	3
SW1	LS32	5F	1
W10	LS14	19F	3
TP6	LS10	4F	2
J1	S08	4D	1
P7	S08	7E	1
CR2	06	22D	2
DS8	06	7A	4
C30	LS04	20C	4
RP6	S02	5E	2
R46	LS02	20B	3

15000212	9520	R46	LS02	20B	3	
NEXT ASSY	USED ON	LAST USED	NOT USED	TYPE	REF DES	QTY
FIRST APPLICATION		REF DESIGNATIONS		SPARE GATES		
APPROVALS		DATE		DRAWN		
T. TAKEDA		11-3-80		CHECKED		
ENGR		SIZE D		DWG. NO. 12000228		
		REV. K		SHEET 1 OF 6		



D

C

B

A

D

C

B

A

8

7

6

5

4

3

1

8

7

6

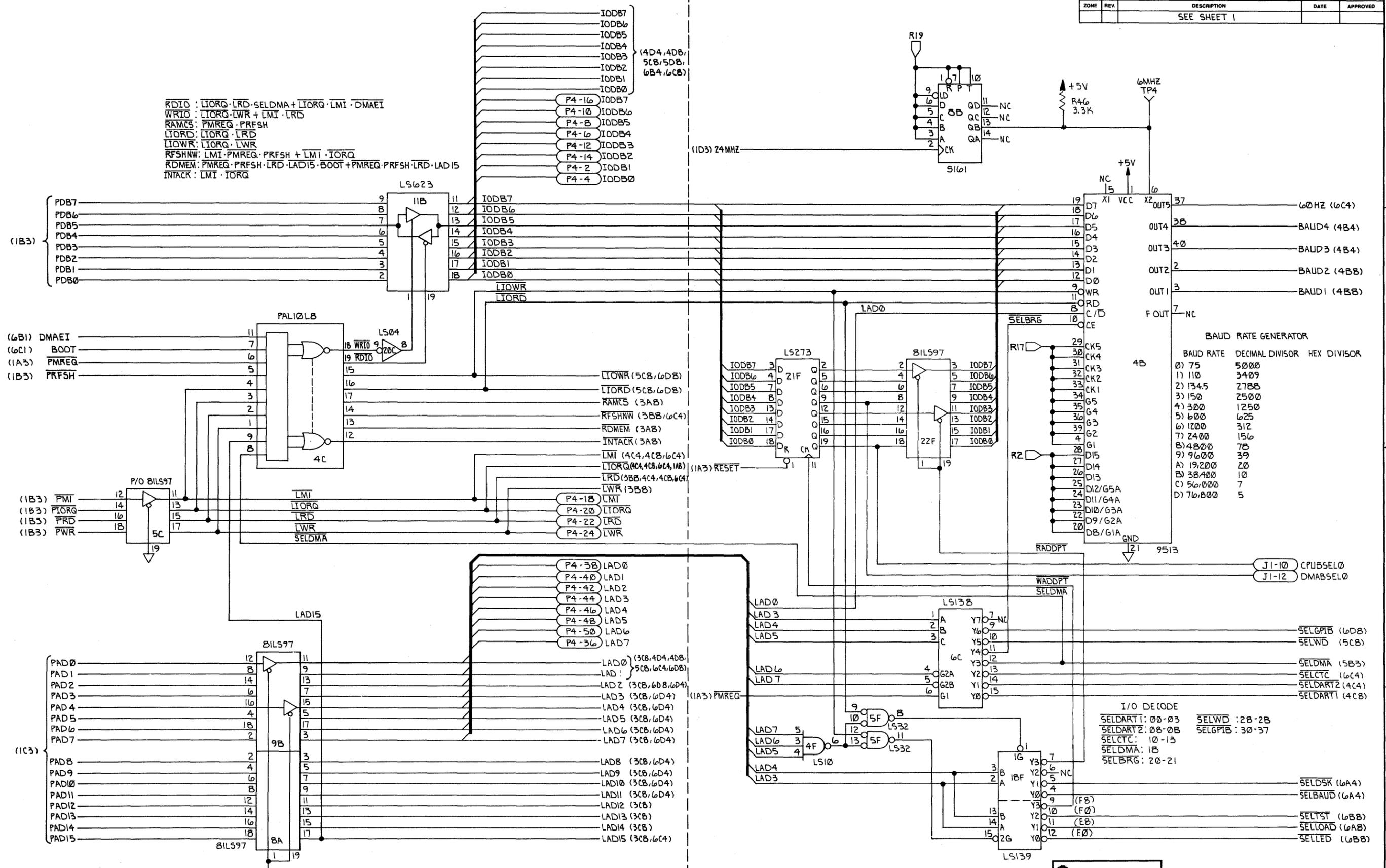
5

4

3

1

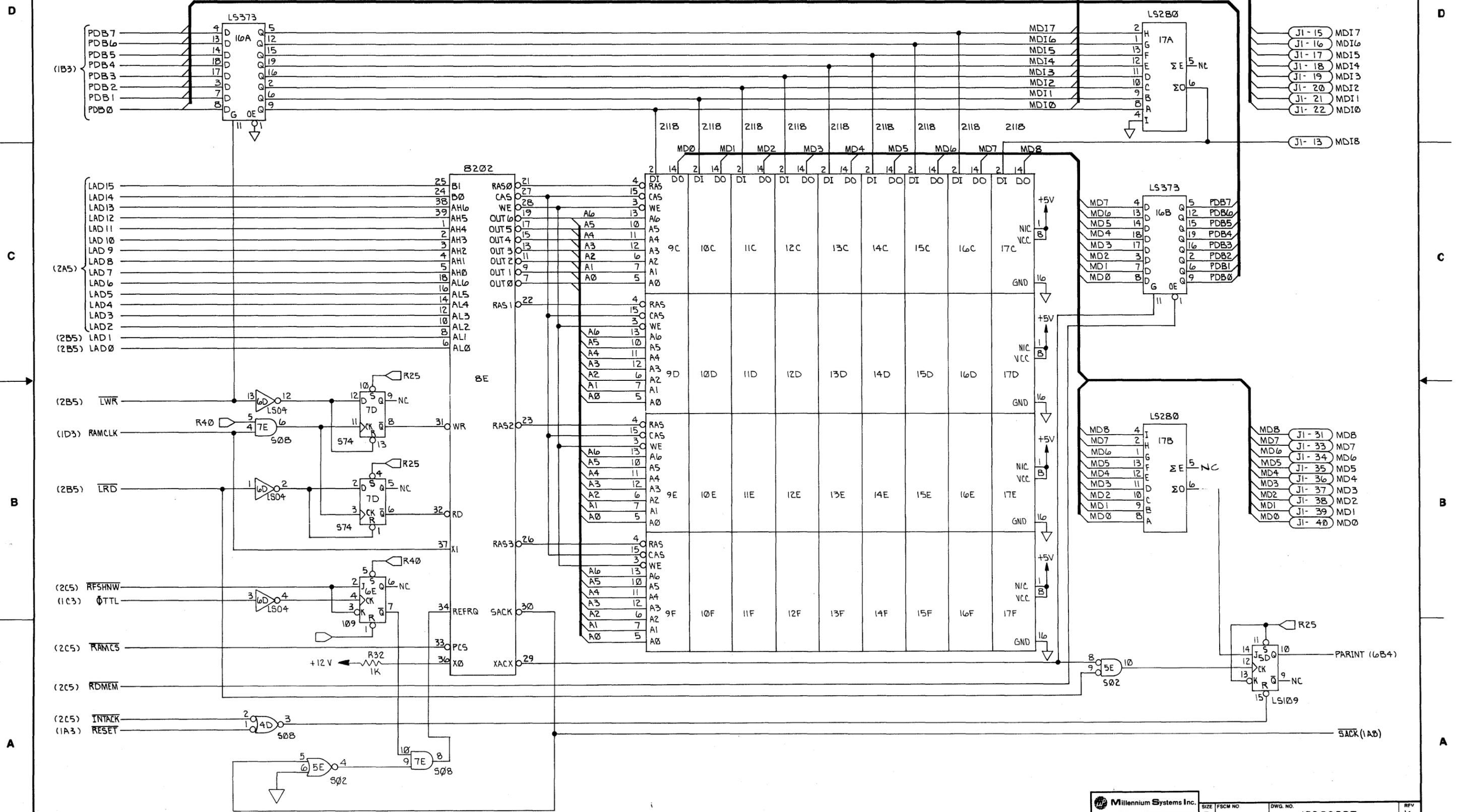
RDIO : LIORQ · LRD · SELDMA + LIORQ · LMI · DMAEI
 WRIO : LIORQ · LWR + LMI · LRD
 RAMCS : PMREQ · PRFSH
 LTORQ : LIORQ · LRD
 LIOWR : LIORQ · LWR
 RFSHNV : LMI · PMREQ · PRFSH + LMI · LIORQ
 RDMEM : PMREQ · PRFSH · LRD · LAD15 · BOOT + PMREQ · PRFSH · LRD · LAD15
 INTACK : LMI · LIORQ



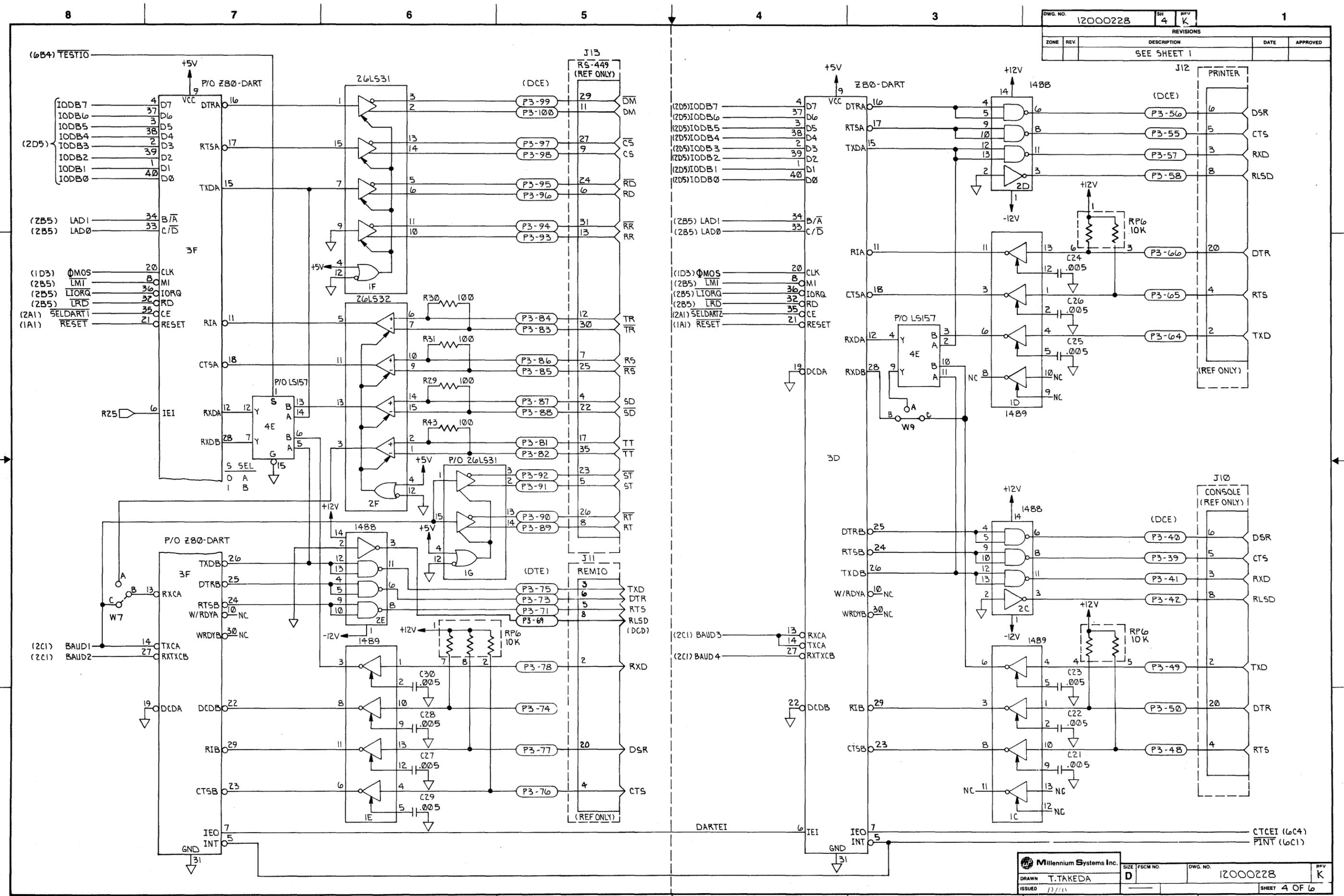
BAUD RATE GENERATOR

BAUD RATE	DECIMAL DIVISOR	HEX DIVISOR
0) 75	5000	
1) 110	3409	
2) 134.5	2788	
3) 150	2500	
4) 300	1250	
5) 600	625	
6) 1200	312	
7) 2400	156	
8) 4800	78	
9) 9600	39	
A) 19200	20	
B) 38400	10	
C) 56000	7	
D) 76800	5	

I/O DECODE
 SELDART1: 00-03 SELWD : 28-2B
 SELDART2: 08-0B SELGPIB : 30-37
 SELCTC : 10-13
 SELDMA : 18
 SELBRG : 20-21

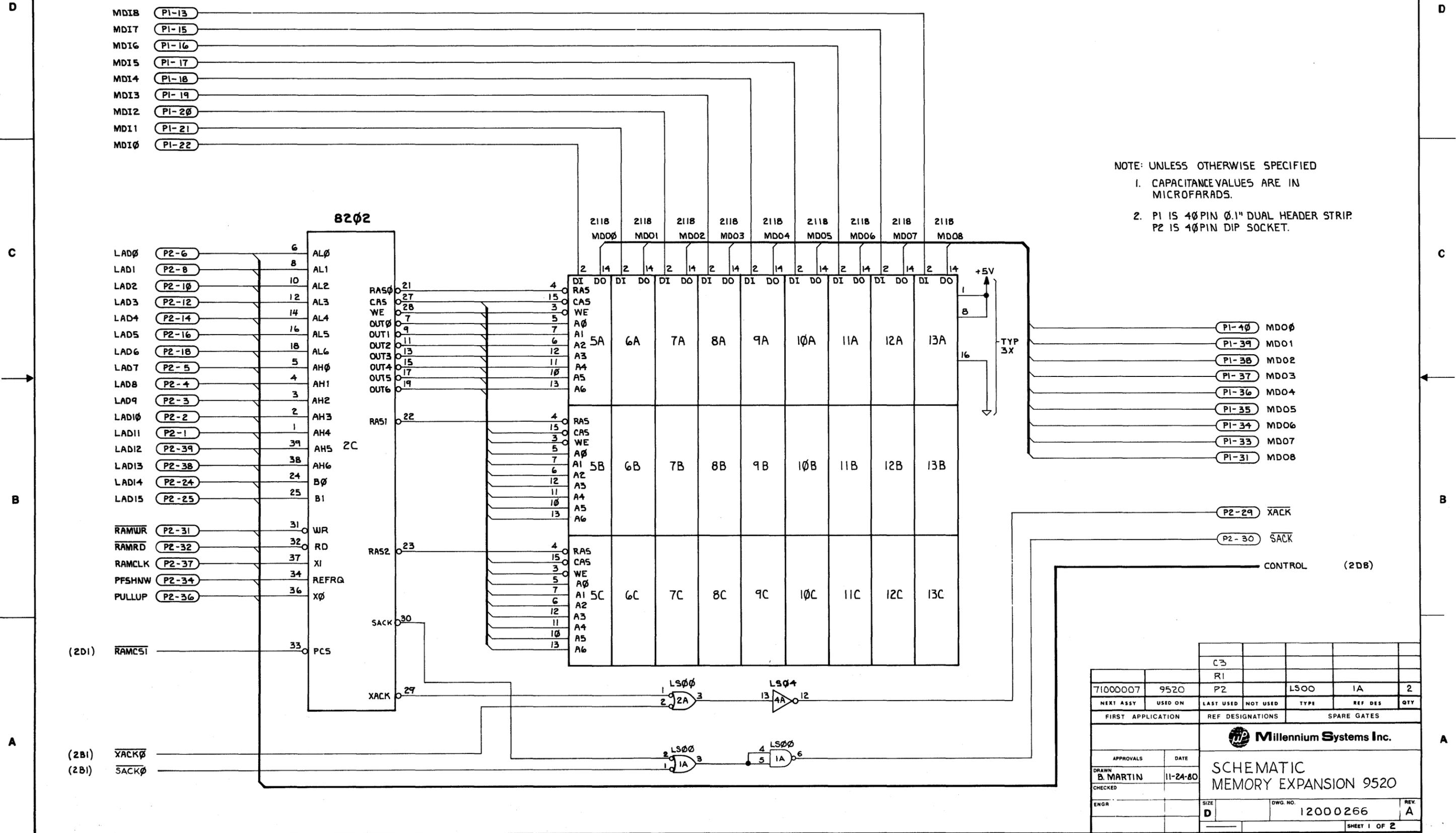


DWG. NO. 12000228		SH 4	REV K
REVISIONS			
ZONE	REV.	DESCRIPTION	DATE
		SEE SHEET 1	



Millennium Systems Inc.		SIZE D	FSCM NO.	DWG. NO. 12000228	REV K
DRAWN T. TAKEDA	ISSUED 12/11/88			SHEET 4 OF 6	

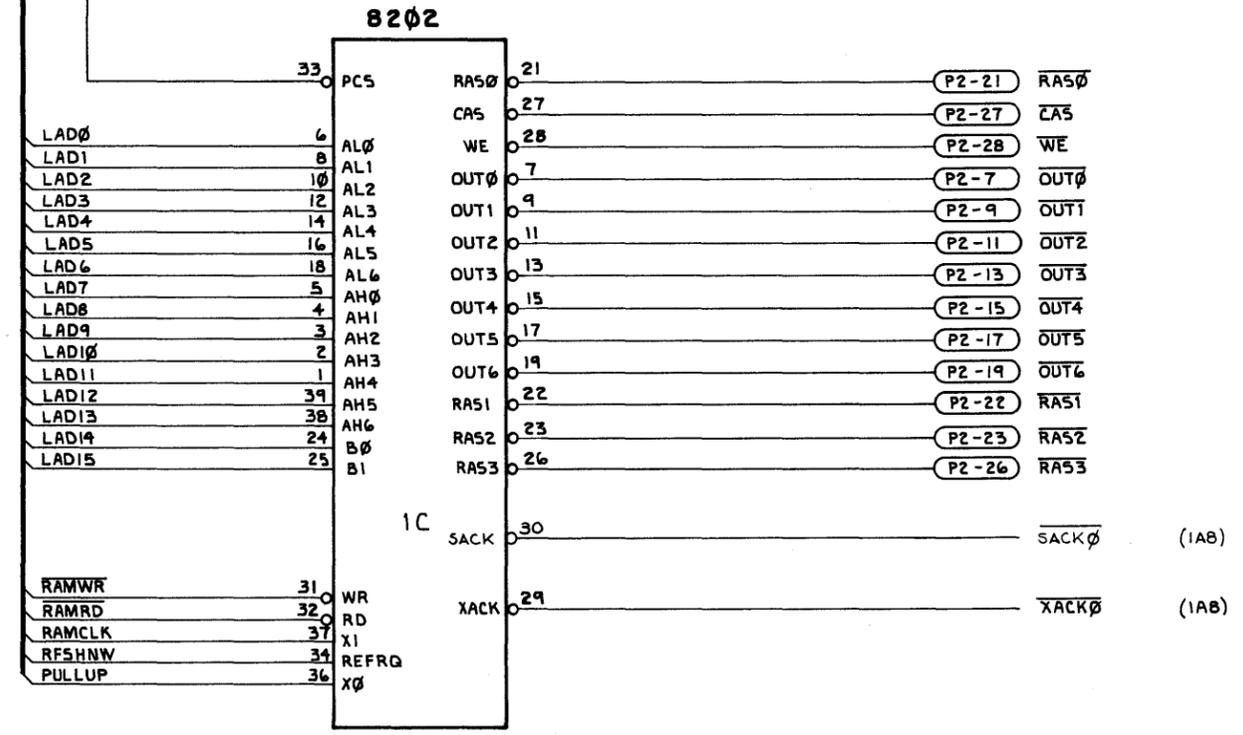
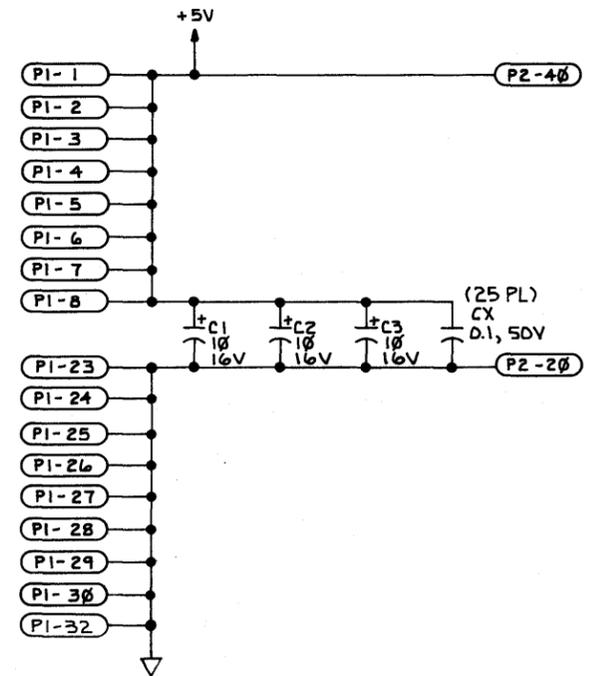
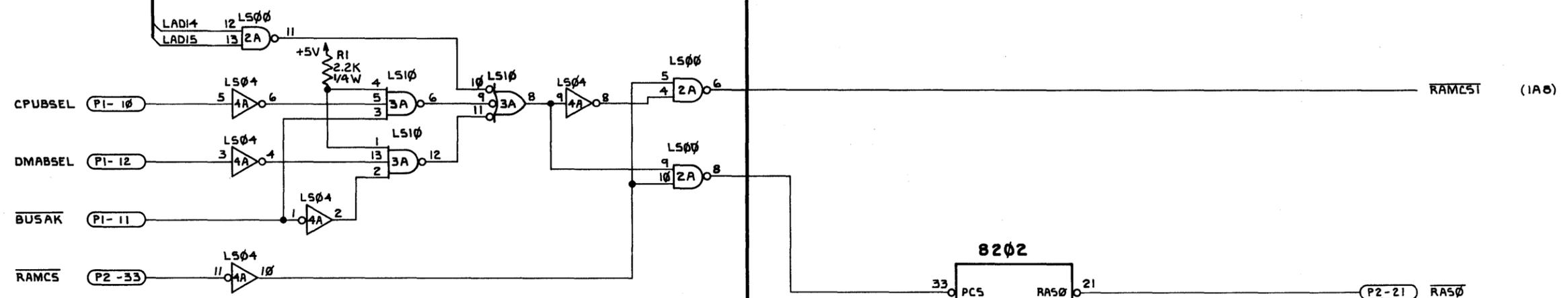
DWG. NO. 12000266		SH 1	REV A	1	
REVISIONS					
ZONE	REV	DESCRIPTION	DATE	APPROVED	
	A	PROD RELEASE ECO 2817	11-24-80		



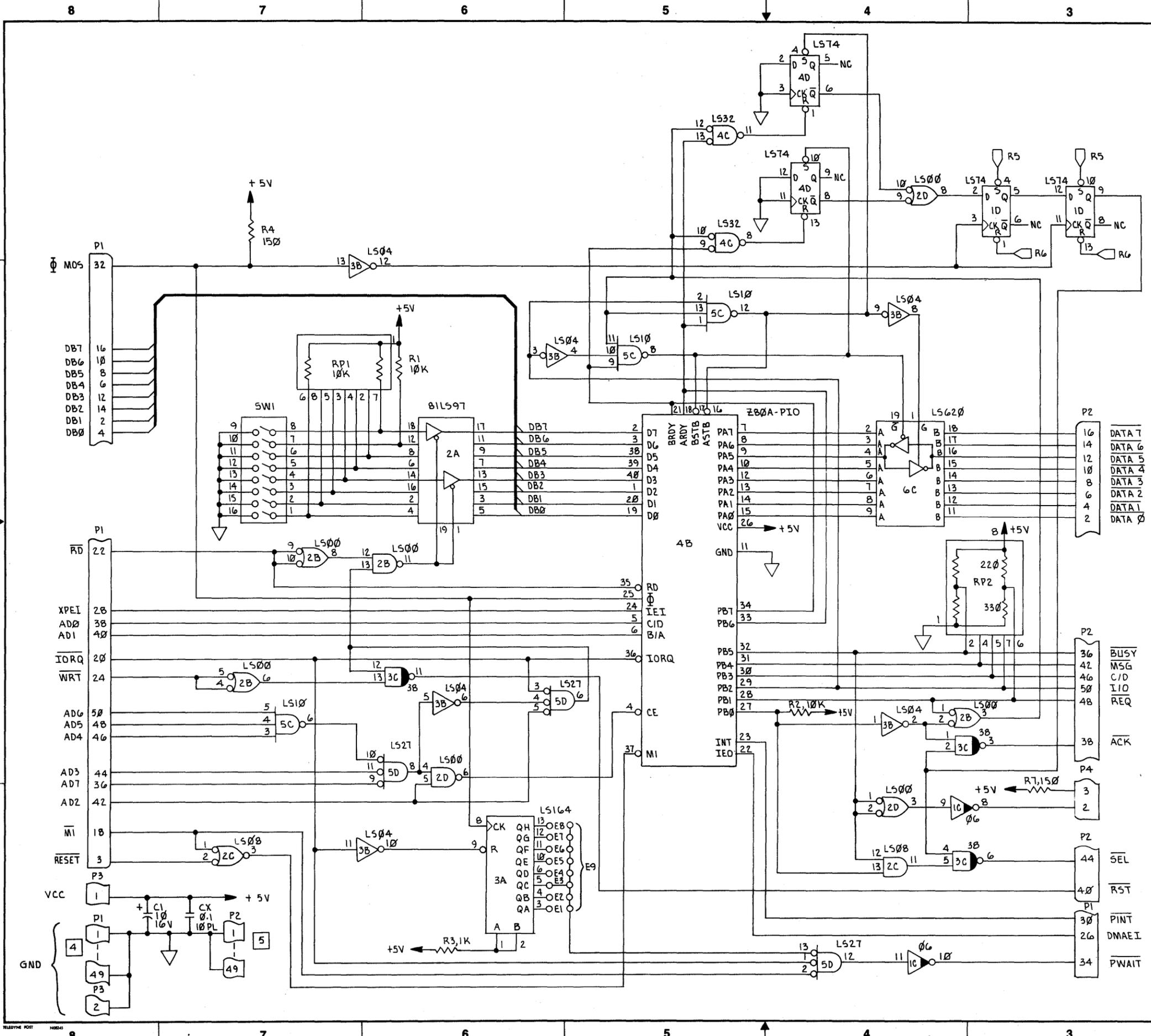
NOTE: UNLESS OTHERWISE SPECIFIED
 1. CAPACITANCE VALUES ARE IN MICROFARADS.
 2. P1 IS 40 PIN 0.1" DUAL HEADER STRIP.
 P2 IS 40 PIN DIP SOCKET.

71000007		9520		C3		R1		P2		LS00		1A		2	
NEXT ASSY	USED ON	LAST USED	NOT USED	TYPE	REF DES	QTY	FIRST APPLICATION		REF DESIGNATIONS		SPARE GATES				
APPROVALS		DATE													
DRAWN B. MARTIN		11-24-80		Millennium Systems Inc. SCHEMATIC MEMORY EXPANSION 9520											
CHECKED				SIZE D		DWG. NO. 12000266		REV A							
ENGR										SHEET 1 OF 2					

(1B1) CONTROL



DWG. NO.	12000276	REV	1	REV	A
REVISIONS					
ZONE	REV	DESCRIPTION	DATE	APPROVED	
50		PILOT RELEASE PER ECO 3153	11-11-82		
A		PROJ REL PER ECO N° 3476 SW	11-11-82		

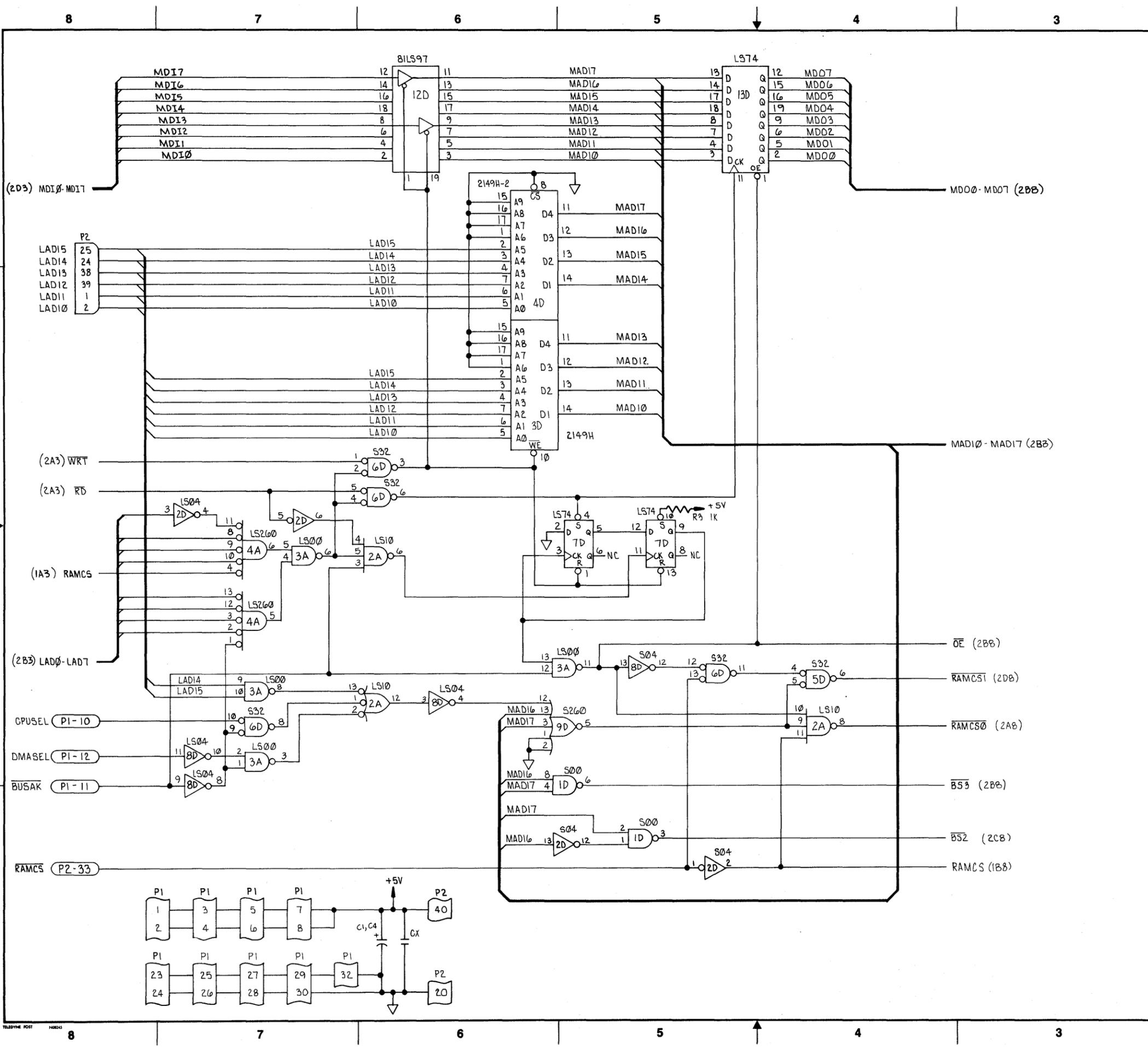


- NOTES: UNLESS OTHERWISE SPECIFIED
1. RESISTANCE VALUES ARE IN OHMS, 1/4 W, 5%
 2. CAPACITANCE VALUES ARE IN MICROFARADS.
 3. SYMBOL INDICATES A COMMON +5V PULL-UP RESISTOR CONNECTED TO MORE THAN ONE I.C. DEVICE AND ARE 1K.
 4. ODD NUMBERED CONNECTOR PINS 1,5 THRU 49 ON CONNECTOR P1 ARE CONNECTED TO GROUND.
 5. ODD NUMBERED CONNECTOR PINS 1 THRU 49 ON CONNECTOR P2 ARE CONNECTED TO GROUND.

I/O DECODE		
RST	OUT	70-73HEX
SWI	IN	70-73HEX
CE(Z80-PIO)	IN/OUT	74-77HEX

		SWI			
		P4	3B	3C	1
		RP2	L532	4C	2
		R7	L508	2C	2
		CE	L1	0C	4
NEXT ASSY	USED ON	LAST USED	NOT USED	TYPE	REF DES
FIRST APPLICATION	REF DESIGNATIONS	SPARE GATES			
Millennium Systems Inc.					
SCHEMATIC, HARD DISK I/O, 9520					
APPROVALS	DATE	SIZE	DWG. NO.	REV.	
DRAWN <i>[Signature]</i>	2-17-82	D	12000276	A	
CHECKED					
ENGR					
SHEET 1 OF 1					

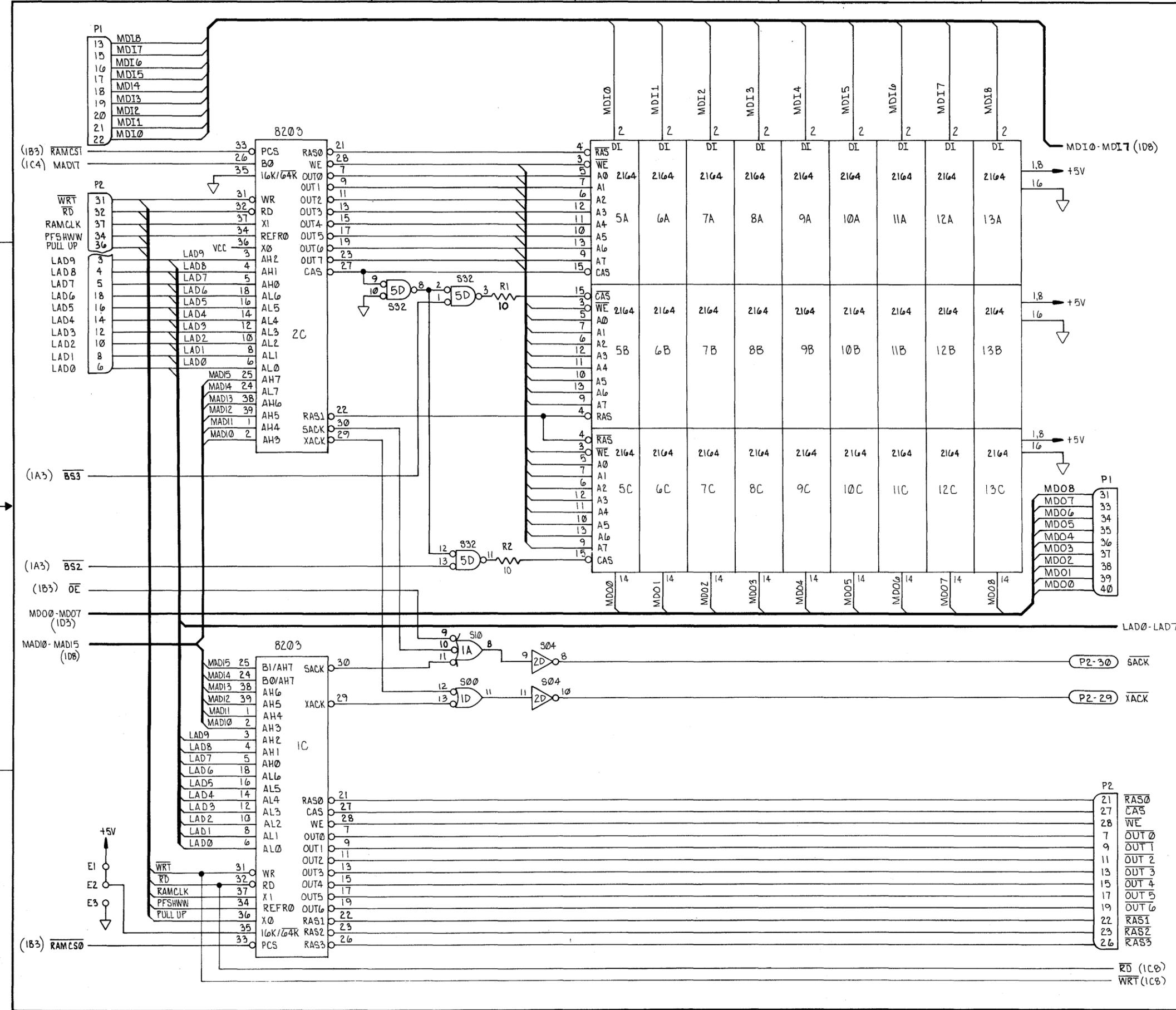
DWG. NO. 12000279		SH 1	REV. A	1	
REVISIONS					
ZONE	REV.	DESCRIPTION	DATE	APPROVED	
	01	PROTOTYPE			
	50	PILOT RELEASE PER ECO 3257			
	51	PER ECO* 3343			
	A	PRODUCTION RELEASE PER ECO 3399			



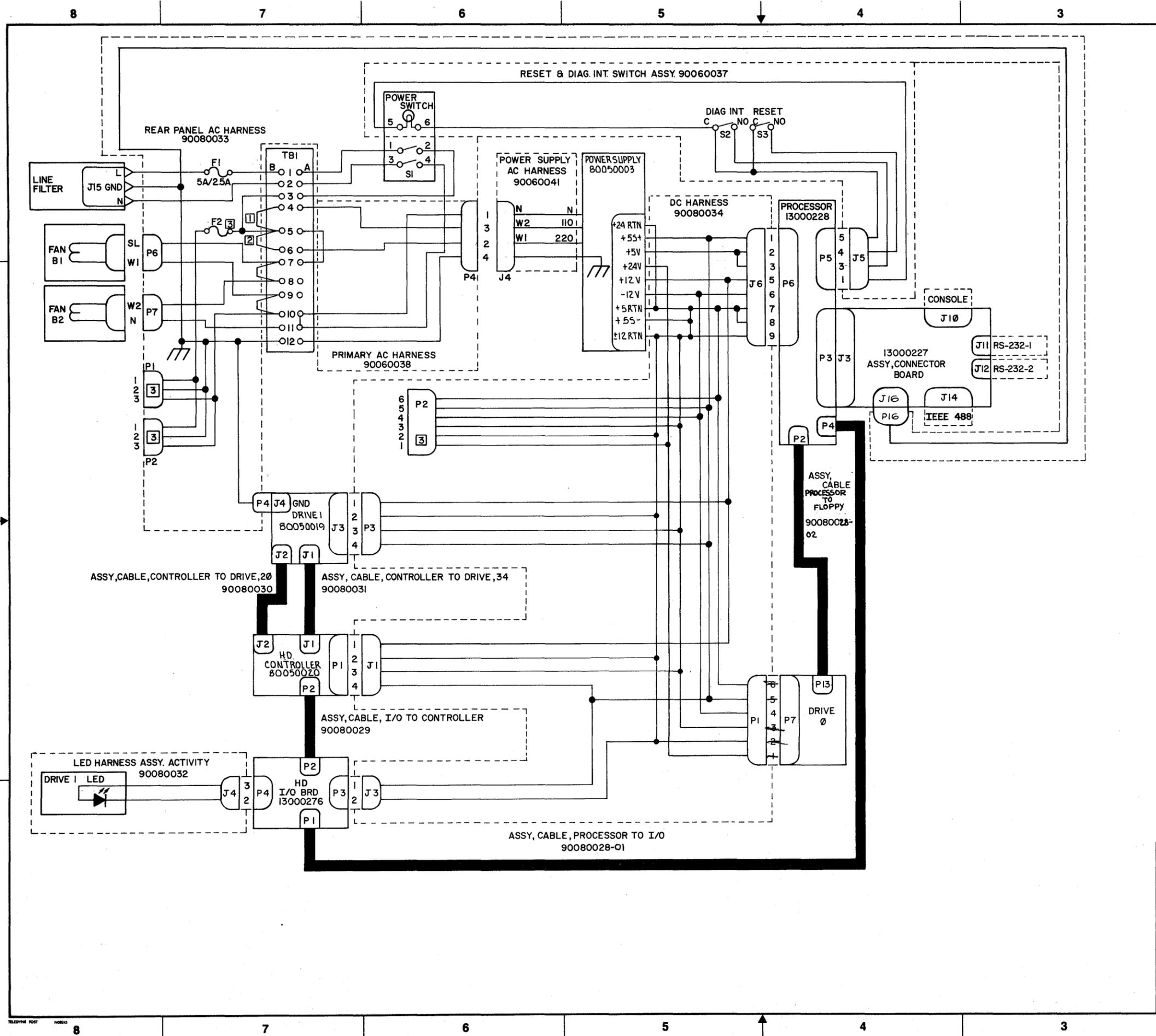
- NOTES: UNLESS OTHERWISE SPECIFIED.
1. RESISTANCE VALUES ARE IN OHMS, 1/4W, 5%.
 2. CAPACITANCE VALUES ARE IN MICROFARADS.
 3. SYMBOL  INDICATES A COMMON +5V PULL-UP RESISTOR CONNECTED TO MORE THAN ONE I.C. DEVICE.
 4. I.C.'S ARE 7400 FAMILY.

R3	74S260	9D	1
P2	74LS10	1A	2
E3	74LS04	8D	2
C4	74LS00	1D	1

NEXT ASSY	USED ON	LAST USED	NOT USED	TYPE	REF DES	QTY
FIRST APPLICATION		REF DESIGNATIONS		SPARE GATES		
Millennium Systems Inc.						
APPROVALS		DATE				
DRAWN: J. PINOCCHIO		4-20-82				
CHECKED: [Signature]		4-21-82				
ENGR: [Signature]						
SIZE D	DWG. NO. 12000279	REV. A		SHEET 1 OF 2		



DWG. NO. 73000005		SH 1	REV C	1	
REVISIONS					
ZONE	REV.	DESCRIPTION	DATE	APPROVED	
	01	PROTOTYPE			
	50	PILOT RELEASE PER ECO # 3246	11-18-82		
	A	ADD CHASSIS GND FOR I/O CONN BRD PER ECO #3284	1/14/83	gls/ps	
	B	CORRECT LED POLARITY PER ECO 3402	1/17/83	gls/ps	
	C	REVISED PER ECO # 3470JWC	11-19-82		



NOTES: (UNLESS OTHERWISE SPECIFIED)

- 1 JUMPER INSTALLED FOR 110V. OPERATION ONLY.
- 2 JUMPER INSTALLED FOR 220V. OPERATION ONLY.
- 3 NOT USED. F2 & CONNECTORS ARE FOR USE ON 9520 ONLY.

NEXT ASSY	USED ON	LAST USED	NOT USED	TYPE	REF DES	QTY
FIRST APPLICATION	REF DESIGNATIONS		SPARE GATES			
Millennium Systems Inc.						
APPROVALS			DATE			
DRAWN <i>Sandy Sirois</i>			3-30-82			
CHECKED <i>[Signature]</i>			1/1/83			
ENGR <i>[Signature]</i>			1/1/83			
SIZE D			DWG. NO. 73000005		REV. C	
SHEET 1 OF 1						

SCHMATIC, WIRING,
SDS .9520 HD

DOCUMENT REVIEW FORM

9520 SOFTWARE DEVELOPMENT SYSTEM
USERS MANUAL

Publication No. 87000064
March 1983
Release 2.0

Dear Customer:

Your comments concerning this document help us to produce better documentation for you.

GENERAL COMMENTS

- | | |
|--|---|
| <input type="checkbox"/> Easy to read? | <input type="checkbox"/> Complete? |
| <input type="checkbox"/> Well organized? | <input type="checkbox"/> Well illustrated? |
| <input type="checkbox"/> Accurate? | <input type="checkbox"/> Suitable for your needs? |

SPECIFIC COMMENTS AND CORRECTIONS

Reference	Page
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

Please send your comments to:

GOULD INC., DESIGN AND TEST SYSTEMS DIVISION
4600 Old Ironsides Drive
Santa Clara, California 95050-1279
ATTN: Publications

PRODUCT ENHANCEMENT REQUEST

1. DATE

2. CUSTOMER INFORMATION

NAME OF COMPANY

NAME OF EMPLOYEE

ADDRESS

CITY/STATE/ZIP CODE

PHONE # _____ EXT. _____

3. PRODUCT NAME (9508, 9520, 9516, 9580, ETC.)

4. DESCRIPTION OF ENHANCEMENT

5. DESCRIPTION OF HOW ENHANCEMENT WOULD BE USED

6. HOW LONG BEFORE YOU MUST HAVE THIS ENHANCEMENT?

NOW

6 MONTHS _____

1 YEAR _____

2 YEARS _____

7. WHAT SHOULD BE THE COST OF THIS ENHANCEMENT?