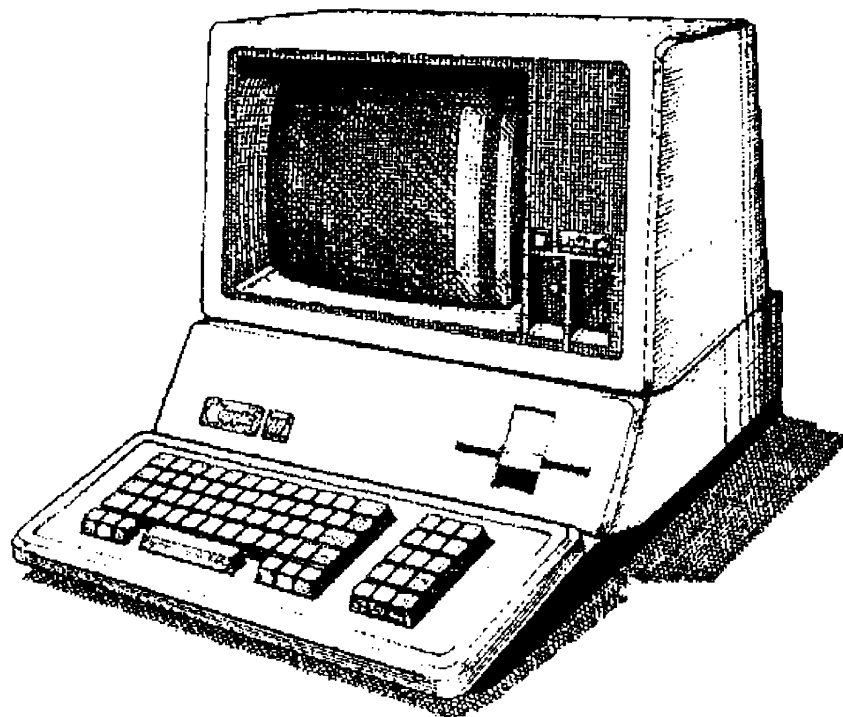




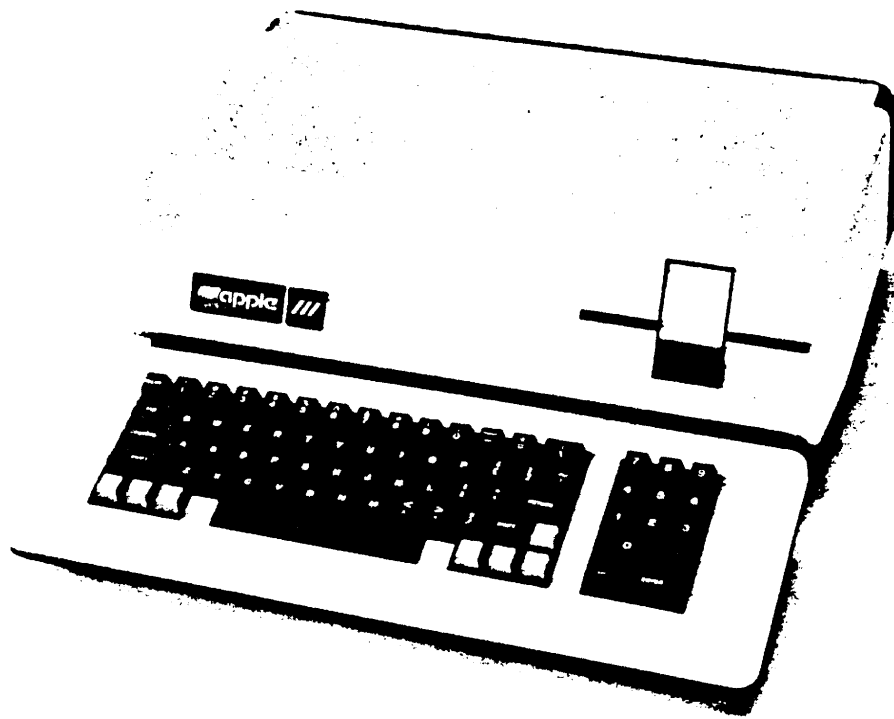
Apple /// Computer Information

Apple /// Service Reference Manual



Theory of Operation • Servicing Information

Written by Apple Computer • 1982



APPLE III SERVICE REFERENCE MANUAL



To the Reader,

This manual was developed for all the A/// Service Technicians at our Level II Regional Service Center. The intent of this book is to help you understand and repair the Apple ///. The book is partitioned into two sections: Theory of Operation and Servicing Information. There is sufficient information in this manual so that an inexperienced technician can be productive in a short time. This manual should help you understand and appreciate the Apple ///.

In Appreciation:

Although many people have helped me with this manual I wish to particularly thank the following people for their contribution to this manual:

Bill Holman
Wendell Sanders
Mike Fallon
Rick Hoiberg
Ed Goodwin
Sandy Sanford
Peter Quinn

Thanks!

Bob Cummings
Bob Cummings

APPLE III REFERENCE MANUAL

If you are interested in how the Apple /// works, or if you want to trouble shoot your machine, you should have this manual. As you can see from the table of contents (shown below), this comprehensive manual covers virtually every facet of the Apple ///. Like new condition, binder, 17 chapters, 460 pages (8-1/2 x 11), two diagnostic diskettes.....\$150.

TABLE OF CONTENTS

VIDEO DISPLAY LOGIC

- Display Modes
- Introduction
- 40 Character Apple II
- 40 Character Apple III
- 80 Character Black & White Apple III
- 80 Character Black & White Hires
- Medium Resolution 16 Color Graphics Apple III
- Super Hires Apple III
- Apple III Hires
- Super Hires Apple III
- Video Appendix

INPUT/OUTPUT

- Description
- Interface Control Signals
- Interrupts
- Addressing the I/O
- The Input Operation
- The Output Operation
- System Timing
- the A/// Joystick

THE KEYBOARD

- The Keyboard
- Reading the Keyboard
- Keyboard Codes
- The Apple II Emulation Mode
- Electronic Circuit Description
- The Repeat Function
- The Reset Function
- Keyboard Light

POWER SUPPLY

- The Apple /// Power Supply
- The Basic Switching Power Supply
- How it works
- Detailed Hardware Description
- Schematic
- Parts List
- Component Layout

A// EMULATION

- A// Emulation Restrictions
- The Color Video Connector
- The High-Resolution Graphics (Hi-Res) Mode
- The Speaker
- The Cassette Interface
- Input/Output Special Locations
- A/D Selection
- Analog Inputs
- Strobe Output
- Autostart ROM/Monitor ROM

LIST OF TABLES & ILLUSTRATIONS

- The System Monitor
- Built-in I/O Locations
- Peripheral I/O
- The Joystick Ports
- Peripheral Connector Pinout
- Peripheral Connector Signal Description
- ROM Listings

SCHEMATIC DIAGRAMS

- Annotated Schematic Diagrams
- DISK SUBSYSTEM
- Theory of Operation
- Disk Conditioning Circuit
- Analog Card

SERVICING INFORMATION

- TESTING & TROUBLESHOOTING
- Apple /// Final Test Procedure
- 16 Sector Disk /// Final Test (1000T)
- 5 Volt Memory Board Ram Troubleshooting Procedure
- The Apple /// Troubleshooting Flowchart (Component)
- PARTS LAYOUT AND PARTS LIST
- Apple /// IC Parts by Location
- Apple /// Indented Bill of Materials
- WIRE LIST
- Wire List

MODULE REPLACEMENT PROCEDURES & ASSEMBLY DRAWINGS

- Apple /// Module Replacement Procedures
- Apple /// Dealer Service Diagnostics Reference
- Apple /// Assembly Drawings

GENERAL APPENDIX

- Apple /// System Overview
- Apple /// System Monitor
- Apple /// Logic Signal Source
- Main Logic Board Circuit Function Areas
- How to Read Prom (Rom) Logic Expressions
- Prom (ROM) Logic Expressions
- Ascii Conversion Tables
- Apple /// Video Logic Block Diagram
- Hires Mode Page 1, 8/M, 280 X 192
- Hires Mode Page 2, 8/M, 280 X 192
- 280 X 192 Color Hires Mode Page 1
- 280 X 192 Color Hires Mode Page 2
- Super Hires Mode Page 1
- Super Hires Mode Page 2
- Ahires Test Page 1

AND MORE!

THEORY OF OPERATION

- INTRODUCTION
- General Description
- Simplified Functional Description
- The Main Logic Board
- The Memory Board
- The Keyboard PCB
- The Disk Drive
- The Apple /// Power Supply

MEMORY & MEMORY ADDRESSING

- Introduction to the Apple /// Memory
- Simplified Memory Logic
- Memory Addressing: Block Flow
- The Processor
- Memory Address Multiplexer
- RAS/CAS Decode
- Alternate Stack
- Memory & Memory Addressing Appendices

THE VERSATILE INTERFACE ADAPTER

- General
- VIA 6522 Pin Descriptions
- VIA Functional Description
- Interrupt Operation
- 6522 VIA Environment and Control
- VIA (FFDX)
- VIA (FFEX)
- VIA Appendices

THE ACIA

- The 6551 Asynchronous Communications Interface Adapter
- Status Register
- Command Register
- Control Register
- Simple Serial Port

SYSTEM CLOCKS & TIMING

- Main Clock CL44
- Frequency Divider
- "Q" Timing
- HPE's Freeze
- AR, PREIM, & CHM
- RAS (Row Address Strobe[Select])
- Video Horizontal & Vertical State Counters
- Horizontal Section
- Vertical Section



TABLE OF CONTENTS

<u>SECTION I.</u>	<u>THEORY OF OPERATION</u>	<u>PAGE</u>
CHAPTER 1	<u>INTRODUCTION</u>	
	General Description	1.1
	Simplified Functional Description	1.4
	The Main Logic Board	1.4
	The Memory Board	1.6
	The Keyboard PCB	1.7
	The Disk Drive	1.7
	The Apple /// Power Supply	1.7
CHAPTER 2	<u>MEMORY & MEMORY ADDRESSING</u>	
	Introduction to the Apple /// Memory	2.1
	Simplified Memory Logic	2.1
	Memory Addressing: Block Flow	2.4
	The Processor	2.4
	Memory Address Multiplexer	2.6
	RAS/CAS Decode	2.6
	Alternate Stack	2.10
	Memory & Memory Addressing Appendices	2.12
CHAPTER 3	<u>THE VERSATILE INTERFACE ADAPTER</u>	
	General	3.1
	VIA 6522 Pin Descriptions	3.1
	VIA Functional Description	3.6
	Interrupt Operation	3.8
	6522 VIA Environment and Control	3.10
	VIA (FFDX)	3.12
	VIA (FFEX)	3.15
	VIA Appendices	3.18
CHAPTER 4	<u>THE ACIA</u>	
	The 6551 Asynchronous Communications Interface Adapter	4.1
	Status Register	4.3
	Command Register	4.4
	Control Register	4.5
	Simple Serial Port	4.6
CHAPTER 5	<u>SYSTEM CLOCKS & TIMING</u>	
	Main Clock Cl4M	5.1
	Frequency Divider	5.1
	"Q" Timing	5.1
	HPE* Freeze	5.5
	Ax, PRE1M, & ClM	5.5
	RAS (Row Address Strobe[Select])	5.5
	Video Horizontal & Vertical State Counters	5.7
	Horizontal Section	5.7
	Vertical Section	5.10



CHAPTER 6	<u>VIDEO DISPLAY LOGIC</u>	
	Display Modes	6.1
	Introduction	6.2
	40 Character Apple][6.2
	40 Character Apple ///	6.3
	80 Character Black & White Apple ///	6.3
	Black & White Hires	6.4
	Medium Resolution 16 Color Graphics Apple ///	6.4
	Super Hires Apple ///	6.5
	Apple /// Hires	6.5
	Super Hires Apple ///	6.7
	Video Appendix	6.9
CHAPTER 7	<u>INPUT/OUTPUT</u>	
	Description	7.1
	Interface Control Signals	7.1
	Interrupts	7.2
	Addressing the I/O	7.2
	The Input Operation	7.3
	The Output Operation	7.3
	System Timing	7.3
	the A/// Joystick	7.5
CHAPTER 8	<u>THE KEYBOARD</u>	
	The Keyboard	8.1
	Reading the Keyboard	8.1
	Keyboard Codes	8.5
	The Apple][Emulation Mode	8.7
	Electronic Circuit Description	8.7
	The Repeat Function	8.8
	The Reset Function	8.8
	Keyboard Light	8.8
CHAPTER 9	<u>POWER SUPPLY</u>	
	The Apple /// Power Supply	9.1
	The Basic Switching Power Supply	9.1
	How it works!	9.3
	Detailed Hardware Description	9.6
	Schematic	9.9
	Parts List	9.10
	Component Layout	9.13
CHAPTER 10	<u>A][EMULATION</u>	
	A][Emulation Restrictions	10.1
	The Color Video Connector	10.2
	The High-Resolution Graphics (Hi-Res) Mode	10.3
	The Speaker	10.3
	The Cassette Interface	10.3
	Input/Output Special Locations	10.4
	A/D Selection	10.4
	Analog Inputs	10.5
	Strobe Output	10.5
	Autostart ROM/Monitor ROM	10.5



	The System Monitor	10.6
	Built-In I/O Locations	10.8
	Peripheral I/O	10.10
	The Joystick Ports	10.11
	Peripheral Connector Pinout	10.14
	Peripheral Connector Signal Description	10.15
	ROM Listings	10.18
CHAPTER 11	<u>SCHEMATIC DIAGRAMS</u>	
	Annotated Schematic Diagrams	11.1
CHAPTER 12	<u>DISK SUBSYSTEM</u>	
	Theory Of Operation	12.1
	Disk Conditioning Circuit	12.1
	Analog Card	12.4
<u>SECTION II.</u>	<u>SERVICING INFORMATION</u>	
CHAPTER 13	<u>TESTING & TROUBLESHOOTING</u>	
	Apple /// Final Test Procedure	13.1
	16 Sector Disk /// Final Test (1000T)	13.9
	Apple /// Troubleshooting (Module Level)	13.11
	5 Volt Memory Board Ram Troubleshooting Procedure	13.16
	The Apple /// Troubleshooting Flowchart (Component)	13.28
CHAPTER 14	<u>PARTS LAYOUT AND PARTS LIST</u>	
	Apple /// IC Parts by Location	14.1
	Apple /// Indented Bill of Materials	14.4
CHAPTER 15	<u>WIRE LIST</u>	
	Wire List	15.1
CHAPTER 16	<u>MODULE REPLACEMENT PROCEDURES & ASSEMBLY DRAWINGS</u>	
	Apple /// Module Replacement Procedures	16.1
	Apple /// Dealer Service Diagnostics Reference	16.25
	Apple /// Assembly Drawings	16.32
CHAPTER 17	<u>GENERAL APPENDIX</u>	
	Apple /// System Overview	17.1
	Apple /// System Monitor	17.3
	Apple /// Logic Signal Source	17.5
	Main Logic Board Circuit Function Areas	17.9
	How to Read Prom (Rom) Logic Expressions	17.10
	Prom (ROM) Logic Expressions	17.11
	Ascii Conversion Tables	17.19



LIST OF TABLES & ILLUSTRATIONS

AIII System Block Diagram (Illustration)	1.2
AIII System Block Diagram - Detailed (Illustration)	1.3
System Functional Block Diagram	1.5
The 12 Volt Memory Board (128K Configuration)	1.6
The 5 Volt Memory Board (128K Configuration)	1.7
The Apple /// Main Logic Board (Module)	1.8
The Apple /// Detailed System Functional Block Diagram	1.9
The Apple /// Memory Map	2.2
Simplified Memory Address Block Diagram	2.3
Block Diagram Memory Address Logic	2.5
RAS/CAS Decode Logic Block Diagram	2.7
Indirect Addressing (Listing)	2.8
Memory & Memory Addressing Appendix	2.12
Apple /// Memory Map	2.13
Memory Map Space Allocations	2.17
Address Logic Truth Table	2.18
The Apple /// Memory Board (12V)	2.19
The Apple /// Memory Board (5V)	2.20
The 5 Volt Memory Board (256K)	2.21
How to READ PROM (ROM) Logic Expressions	2.22
RAM Logic Expression	2.23-2.28
MPU Registers	2.29
Block Diagram of the 6522 VIA	3.2
Addressing 6522 VIA Internal Register	3.2
VIA FFDX Internal Register Summary	3.11
VIA FFEX Internal Register Summary	3.14
VIA Appendices	3.18
The ACIA Block Diagram	4.2
ACIA Control Register	4.11
ACIA Command Register	4.11
ACIA Status Register	4.12
ACIA Pin Configuration	4.12
Main Clock 14Mhz Circuit	5.2
Frequency Divider Circuit	5.2
A/// System Timing Diagrams	5.3-5.6
1 to 2 Mhz Gearshift	5.8
Video State Counter	5.8
Video Scan Decode ROM	5.9
Video Appendix	6.9
Apple /// Video Logic Block Diagram	6.10
Hires Mode Page 1, B/W, 280 X 192	6.11
Hires Mode Page 2, B/W, 280 X 192	6.12
280 X 192 Color Hires Mode Page 1	6.13
280 X 192 Color Hires Mode Page 2	6.14
Super Hires Mode Page 1	6.15
Super Hires Mode Page 2	6.16
Ahires Test Page 1	6.17

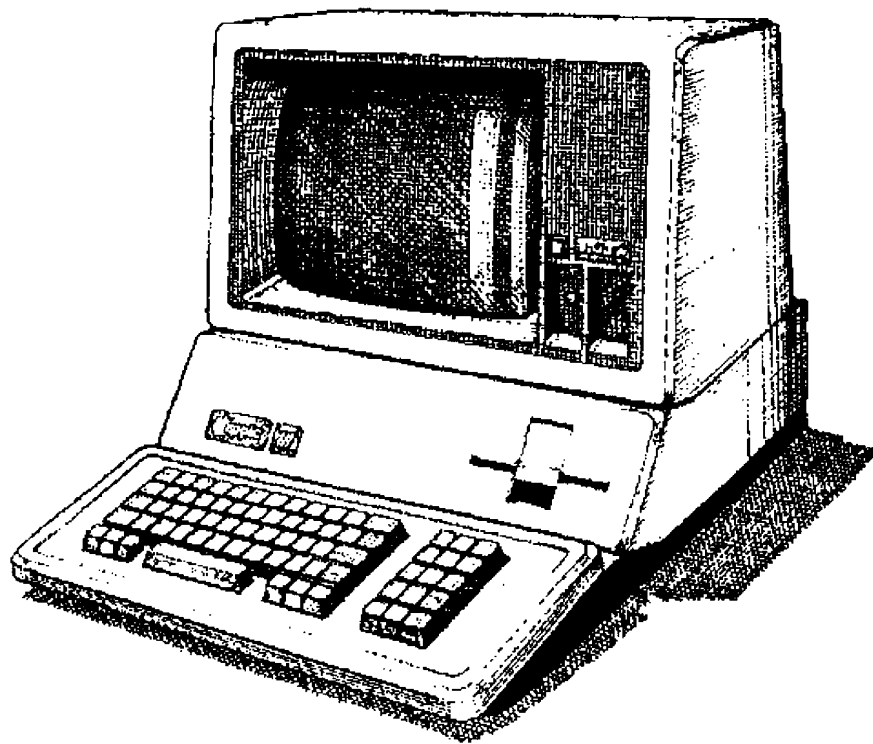


Ahires Test Page 2	6.18
Color Bar & Gray Scale Test	6.19
Apple][Text Mode Page 1	6.20
Apple][Text Mode Page 2	6.21
Sara 40 Column Text Mode Test	6.22
Sara 80 Column Text Mode Test	6.23
Video Mode Truth Tables	6.24
Video Circuit Schematic	6.25
Video Output types	6.26
Video Output Type (Circuit diagrams)	6.27
Video ROM Logic Expressions	6.29
Video ROM Circuit truth table representation	6.30
Color Video Connector Description	6.31
I/O System Timing Diagram	7.4
Peripheral Connector Pinout	7.7
Peripheral Connector Signal Description	7.8
Pin Signal Assignment	8.9
Keys & Associated ASCII Codes	8.10
Keyboard Circuit Schematic	8.12
Reset Circuit Schematic	8.13
A/// Power Supply Parts Layout	9.2
Switching Power Supply Block Diagram	9.4
Switching Power Supply Circuit Block Diagram	9.5
A/// Power Supply Schematic Diagram	9.8
A/// Power Supply Parts List	9.10
A/// Power Supply Parts Layout	9.13
A/// Schematic Diagrams	11.1-11.12
Disk Conditioning Circuit Schematic	12.2
Analog Card A/// Circuit Schematic	12.5
A/// Internal Disk Assembly Drawing	12.8
Install Disk Drive Illustration	12.9
Apple /// Disk Enables	12.10
Main Logic Board IC Designators	14.3
Apple /// Logic Signal Source	17.5
Main Logic Board Circuit Function Areas	17.9
How to Read Prom (Rom) Logic Expressions	17.10
Prom (ROM) Logic Expressions	17.11
Ascii Conversion Tables	17.19



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 1 • Introduction

Written by Apple Computer • 1982



INTRODUCTION

GENERAL DESCRIPTION

The Apple /// is a personal computer for the professional. It has the capabilities to run very involved programs since it can have up to 256K of RAM. The overall unit has been designed to incorporate the best features and options that make it a complete personal computer. The Apple][emulation mode allows users to run most Apple][software. However, minor modifications may be required for some Apple][programs or other peripheral devices.

The base system has a full ASCII keyboard which includes a 13-key numeric key pad with two special function keys. There are four cursor control keys. The A/// has two special repeat features: 1) each key repeats when held depressed, and 2) a high-speed repeat is activated with the Solid Apple key. It's typewriter style keyboard is sculptured for maximum typing speed and accuracy.

The Apple /// has a built-in disk drive (140K bytes) and controller which is capable of supporting three additional external drives without additional interfacing. One interesting feature is that the two drives may be on at the same time. This increases the disk-to-disk transfer effectiveness. The Apple /// can also be used with "Profile"- Apple's 5 Megabyte hard disk for mass data storage.

A built-in RS-232 port, located on the back panel, allows you to connect the Apple /// to letter quality printers, high-speed data collection devices, modems, and other serial input/output devices using RS-232-C protocol. The A/// has two joystick ports for games, sophisticated cursor control, or silentype operation.

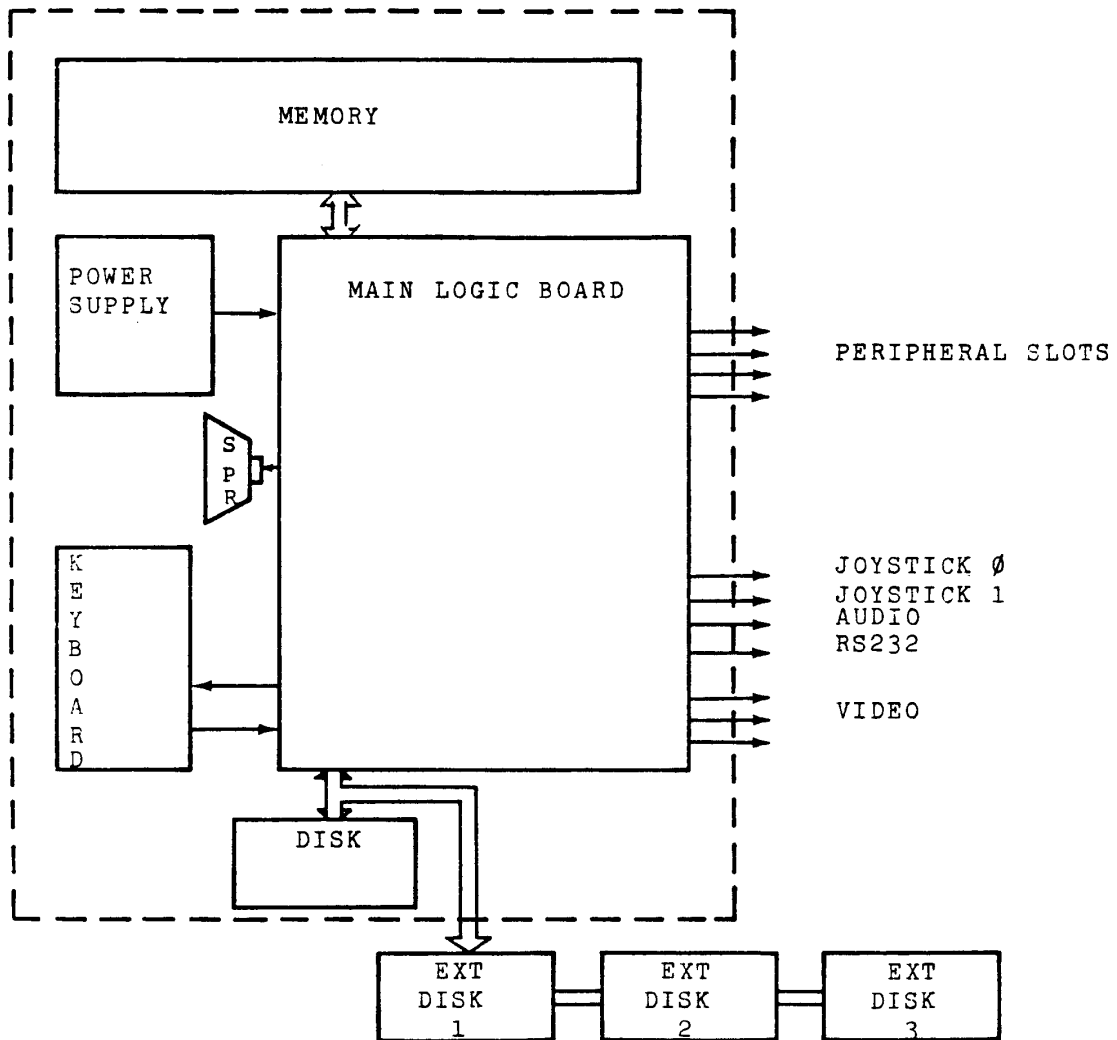
The Apple /// has 8 different modes of video operation. B/W Text in 40 and 80 column, a 40 column 16 color text mode (where the foreground and background of each character can be defined). The Apple][graphics modes are duplicated and there are three more graphics modes: a super black and white Hi-Res, 16 color Hi-Res, and 16 color medium resolution graphics. The eighth mode is actually a utilization of the color text mode where the user defines the character image and builds video images with these "character sets". Since the video character generator is RAM, not ROM, as in the Apple][, it provides the user with the capability of defining character sets to display whatever the user wants. Three video outputs are provided at the back panel; these are black and white, NTSC color composite, and RGB video for exceptional color purity and resolution.

The Apple ///'s Central Processing Unit (CPU) can be "interrupted" by peripheral devices whenever they require CPU control. Alternatively the CPU can poll the devices to determine which needs attention, thereby minimizing the software required for peripheral control.

There are more features in the Apple /// such as, a built-in clock/calender, a hardware beeper to simplify programming, a six-level D/A converter for more

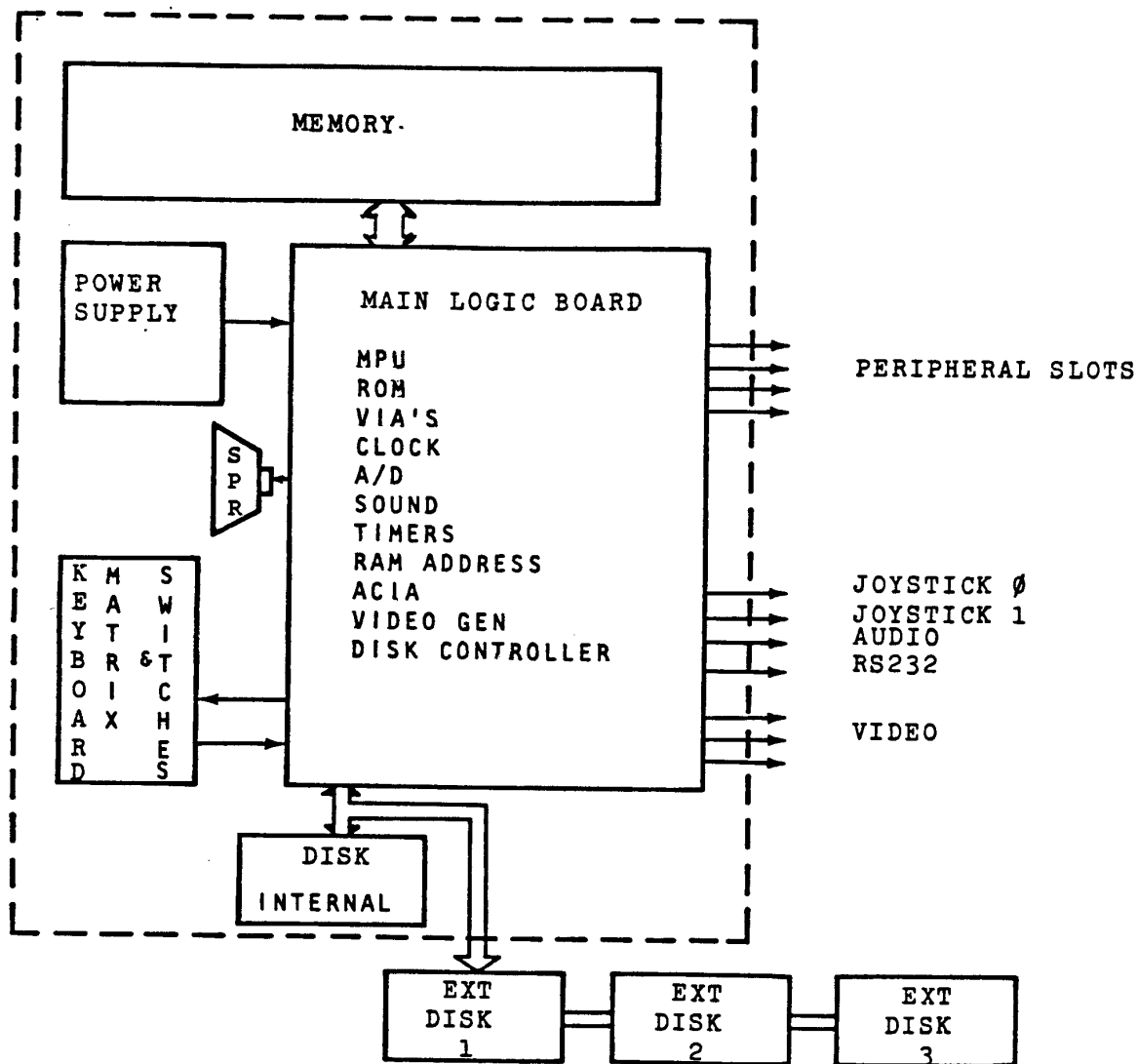


A III SYSTEM BLOCK DIAGRAM



1.2

A III SYSTEM BLOCK DIAGRAM



DEFINITIONS

- MPU - MICROPROCESSOR
- ROM - READ ONLY MEMORY
- VIA - VERSATILE INTERFACE ADAPTER: A SPECIAL PURPOSE PROCESSOR
- ACIA - ASYNCHRONOUS COMMUNICATIONS INTERFACE ADAPTER PROVIDES RS232 COMMUNICATIONS CAPABILITY,
- A/D - ANALOG TO DIGITAL CONVERTER

FIG 1.1



complex tone generation, and the duplication of the speaker function of the Apple][.

As you can see there are many onboard features that would fully load an Apple][, yet the Apple /// has four expansion slots for additional user interfacing. As you read this document and learn how it works, you will appreciate its capabilities, design, and usefulness.

SIMPLIFIED FUNCTIONAL DESCRIPTION

The Apple /// is not an easy machine to understand. It has been designed to emulate the Apple][and has done many operations in a different manner, while adding many enhancements which contribute to its complexity. To understand the system structure it is best to start building functional blocks and gain an understanding of each separately, and then comprehensively.

There are five major parts (modules) to the Apple ///. These parts are:

- 1) Main Logic PCB - this board functions primarily as a processor and device controller. Many functions are integrated into the board, including the disk controller.
- 2) Memory PCB - this board stores data/programs temporarily (until power is removed).
- 3) Keyboard PCB - this is the primary input device provided to the user.
- 4) Disk Drive - Mass storage device for storing data.
- 5) Power Supply - provides the voltages and regulation required to keep everything else working.

THE MAIN LOGIC BOARD

The Main Logic Board is easily identified by its large size and mass quantities of integrated circuits (IC's).

Referring to the block diagram of Figure 1.1 we encounter the microprocessor (MPU), Boot/Monitor ROM, address decode/select circuitry, the Versatile Interface (VIA) containing the bank switch register and the sound register, the VIA containing the environmental and zero page registers, the Asynchronous Communications Interface Adapter (ACIA), analog to digital circuit (joystick inputs), the expansion I/O slots, disk controller, keyboard encoder, video generator, RAM, RAM address circuits, and the system and video timing circuits. Whooo, now you see why it's so big!

Figure 1.1 shows the Apple /// in it's simplest form and presents its expanded I/O capability. On the other hand, the System Functional Block Diagram displays the system in more detail and presents a sophisticated system using some highly unique designs. Some definitions have been provided for some of the terms used in the block diagram.

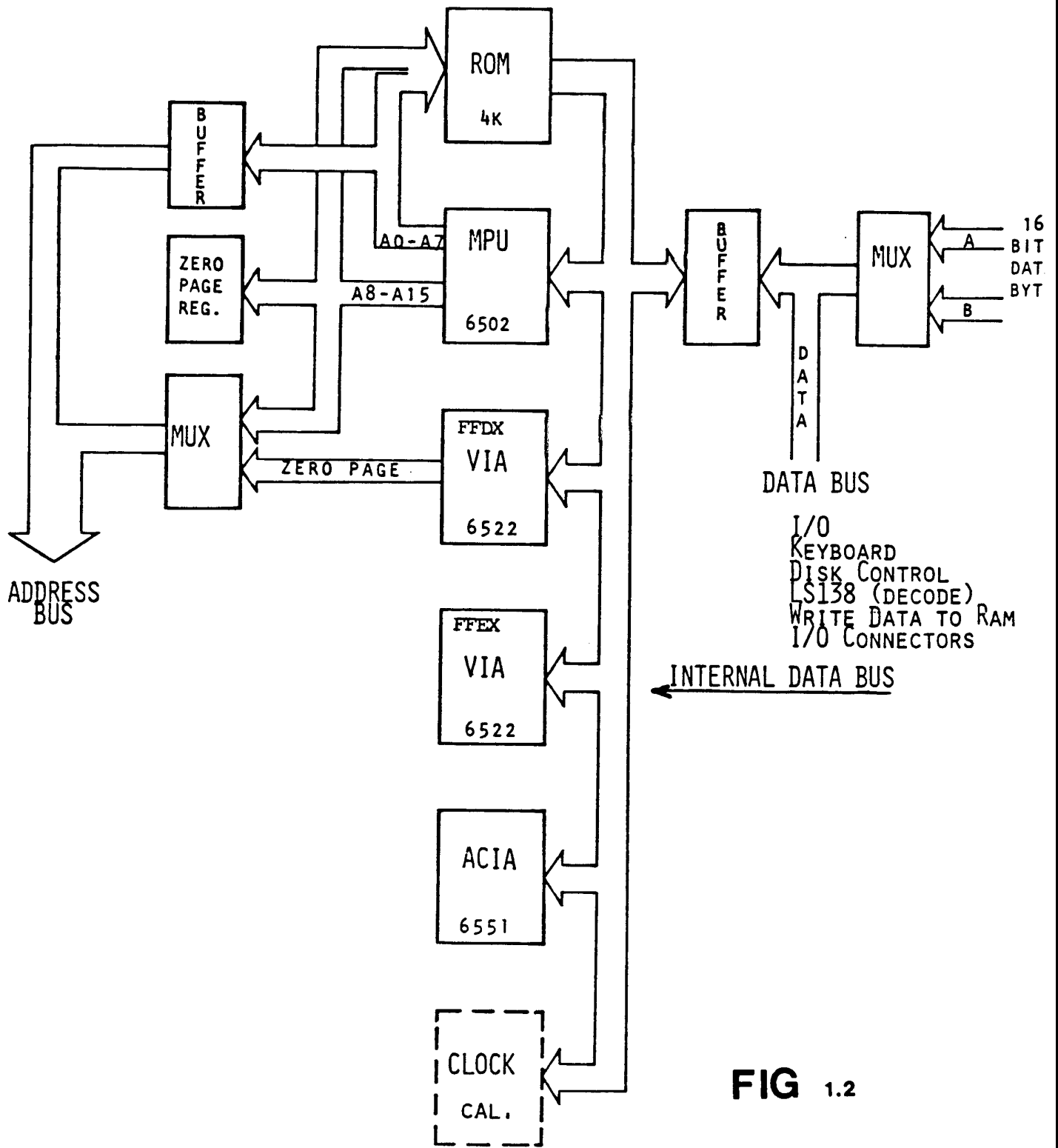


FIG 1.2

1.5



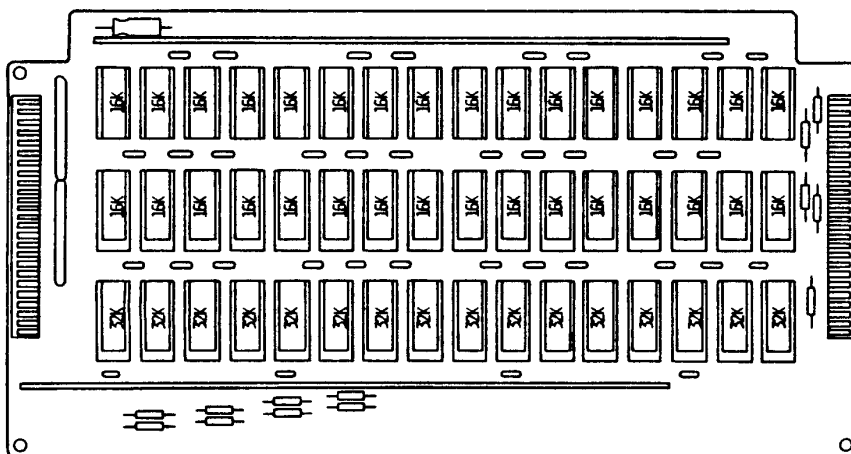
It is best to think of the Apple /// processor as more than just the MPU chip. The processor/controller is actually comprised of many components. The most significant of these are the MPU and the two VIA's. Because of the system's complexity and memory size, the MPU must have extended addressing. Extended addressing is accomplished thru bank switches, environmental register, and a zero page register.

The Apple /// system is interrupt driven. In order to efficiently use processing time, only those devices that allow programs access to the processor are serviced. In fact the processor can even totally mask (disable) the reset key.

THE MEMORY BOARD

The other PCB in the Apple /// is the memory board. It is mounted on the Main Logic Board by two rows of pin connectors. There are two distinct types of memory boards. The early memory board version is commonly referred to as the 12V Memory Board. Below is an illustration of this board.

THE 12 VOLT MEMORY BOARD (128K CONFIGURATION)

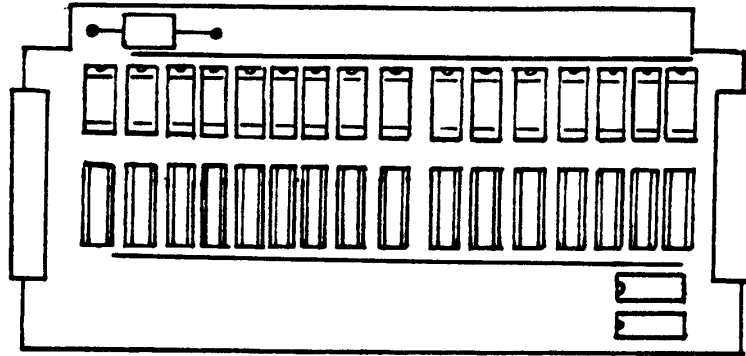


With the correct Main Logic Board configuration, this board can have up to 128K RAM (without modification). The board uses 16K and 32K RAM chips.

The latest memory board version is called the 5 Volt Memory Board. This board, illustrated in the accompanying pages, can be configured for 128K or 256K RAM. The 5 Volt Memory Board, however, requires the correct Main Logic Board configuration.



THE 5 VOLT MEMORY BOARD (128K CONFIGURATION)



THE KEYBOARD PCB

This has to be one of the nicest keyboards around. The sculptured keys are a delight to touch. The Apple /// does not have an on board keyboard encoder - the keyboard encoder is on the Main Logic Board. The keyboard is basically a matrix of switches. The keyboard is connected by means of a 26 pin ribbon cable to the Main Logic Board.

THE DISK DRIVE

The disk drive is similar to the Disk][. The major difference between the Disk][and the Apple /// drive are the door, the bezel, and the Analog Card. Disk switch detection circuitry has been added to the Disk /// Analog Card. Through daisy-chaining you can have up to three (3) external disk drives.

THE APPLE /// POWER SUPPLY

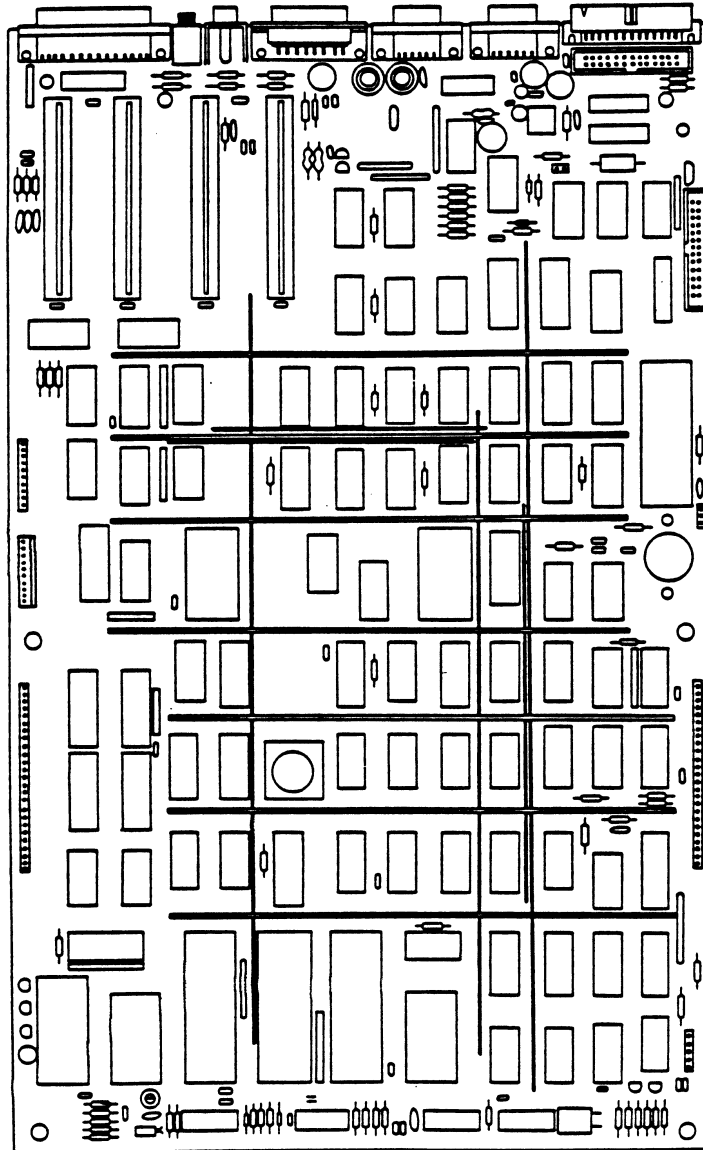
The power supply, accessible from the bottom of the Apple ///, is housed in the casting. It is a "switching type" power supply that supplies the following voltages:

- o +5.0 VDC
- o +11.8 VDC
- o -5.0 VDC
- o -12.0 VDC

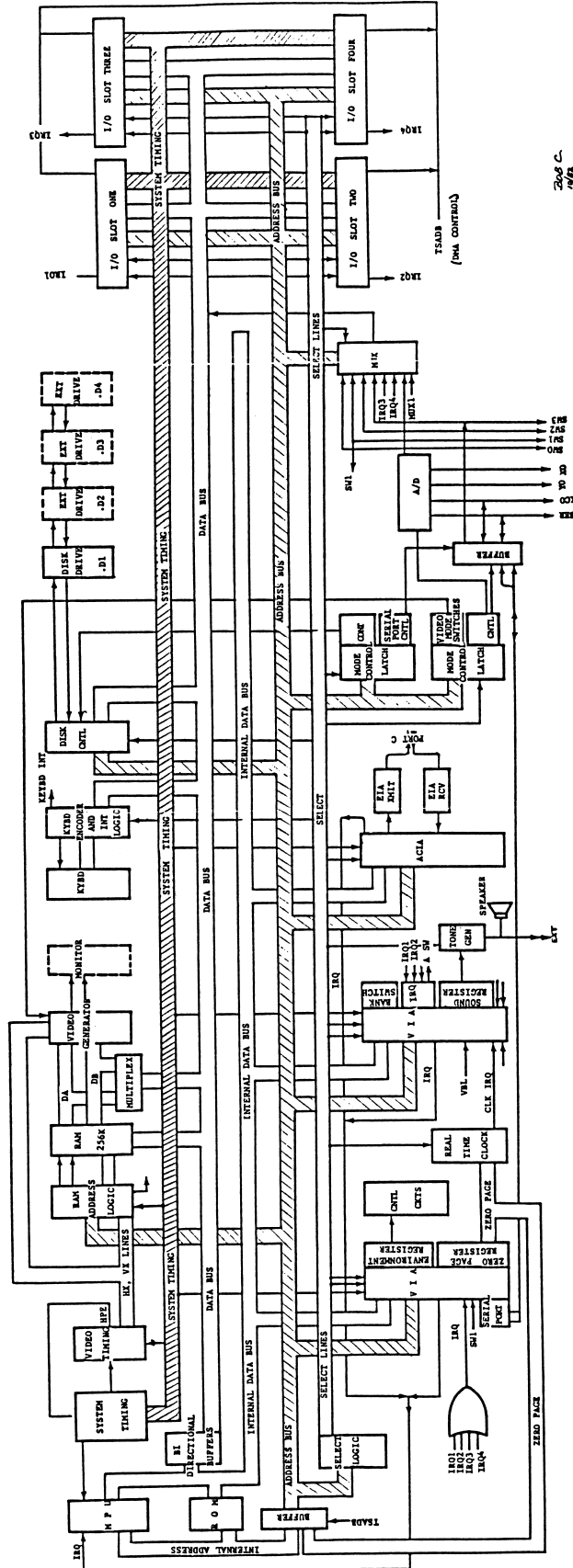
yet, it consumes less power than a 100 Watt light bulb. The power supply also has several protection features, ie. overvoltage protection.



Now that we have taken a tour of the contents of the Apple /// let us begin to learn the inner workings.



THE APPLE /// MAIN LOGIC BOARD (MODULE)

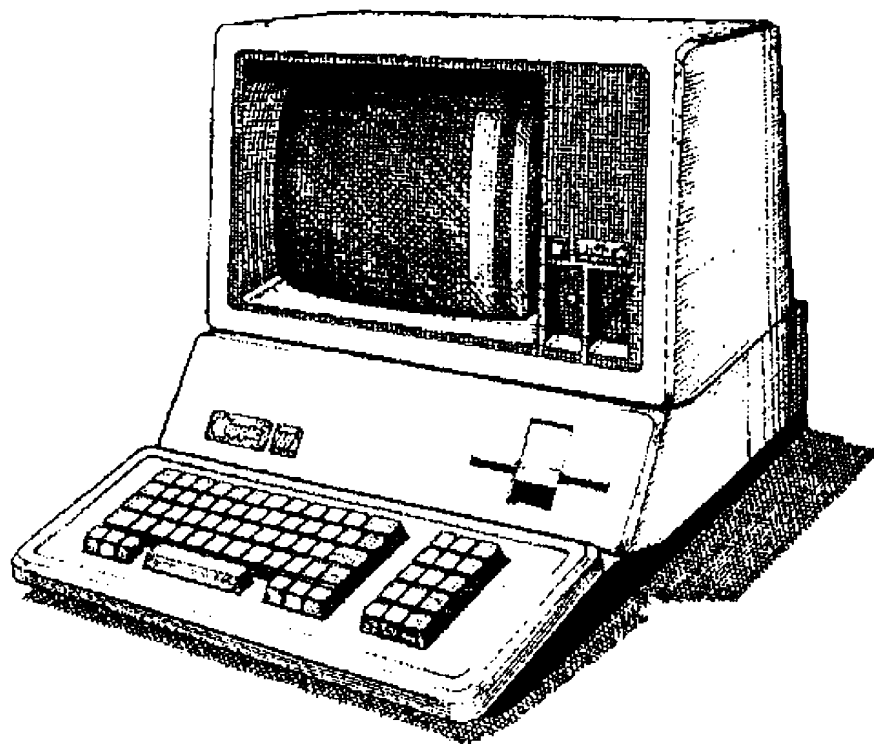


2408 C.
1/78



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 2 • Memory & Memory Addressing

Written by Apple Computer • 1982



MEMORY & MEMORY ADDRESSING

INTRODUCTION TO THE APPLE /// MEMORY

In looking at the Apple /// and its memory we are immediately posed with the problem of how the 6502 processor can handle 128K of RAM, 4K of ROM, and a heavy array of internal and external I/O devices. The Apple /// does indeed do just that, and, further, has the hardware capability of controlling an additional 128K of memory (for a total of 256K).

This "magic" is accomplished by Bank Switching technology. At any one time, the processor can directly address 65K locations. With the addition of the Bank Switch Register, an extended addressing register, the program can call up different banks of 32K RAM space. With other software switches, ROM and all I/O locations can be replaced with RAM. [See Figure 2.1]

The first 8K of memory, from locations 0000 to 1FFF, are fixed. The 32K memory from locations 2000 to 9FFF are electrically switchable. The Apple /// can choose any of 15 banks to place in this area at any one time. The maximum amount of storage on the Apple ///, therefore, is equal to 15 Banks x 32K per bank + 32K fixed storage. By comparison, a 128K system would have three banks, a 256K system would have seven, etc.

In addition, the area in the fixed bank from location C000 to CFFF can be switched from RAM memory to I/O space for the slots. The area from F000 to FFFF is also switchable. When the machine is turned on, this area is ROM containing the startup program. This program runs a quick system check then loads SOS in from the internal drive. SOS then switches this area back to RAM.

The extended addressing mode allows any two adjacent banks, N and N+1, to be addressed as a contiguous 64K RAM space. Addresses 0000 to 7FFF are mapped into bank N while addresses 8000 to FFFF are mapped into bank N+1.

The Apple /// has the capability for variable Zero Page locations and Alternate Stack locations. All these features give the Apple /// great flexibility. They also provide means for very large application programs, or applications that need large amounts of RAM space for data crunching.

SIMPLIFIED MEMORY LOGIC

If we simplify the memory and memory address logic we get three basic elements:

- o the processor
- o the RAM address circuits
- o the RAM array



Memory Map

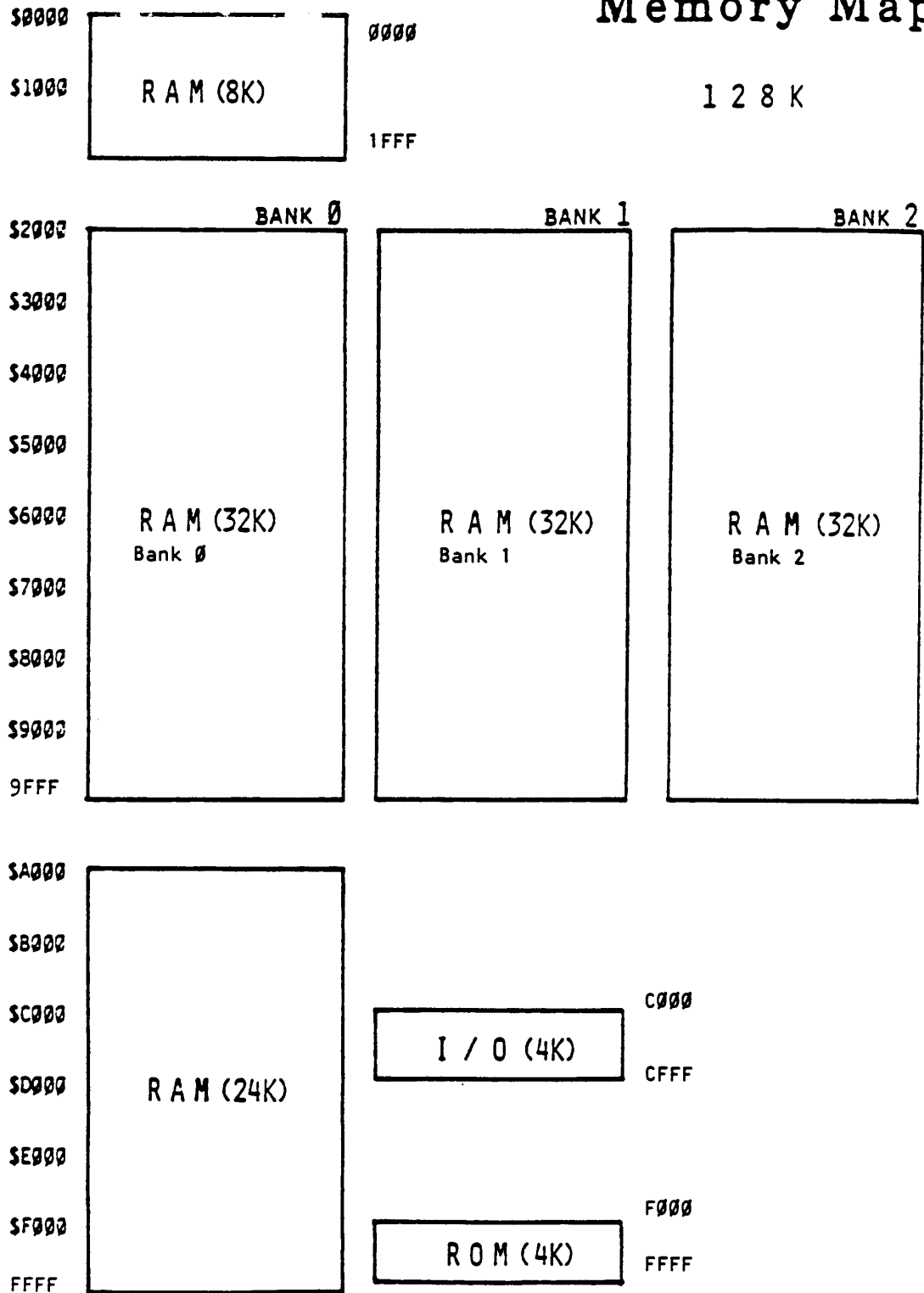


FIG 2.1

2.2

SIMPLIFIED MEMORY ADDRESS BLOCK DIAGRAM

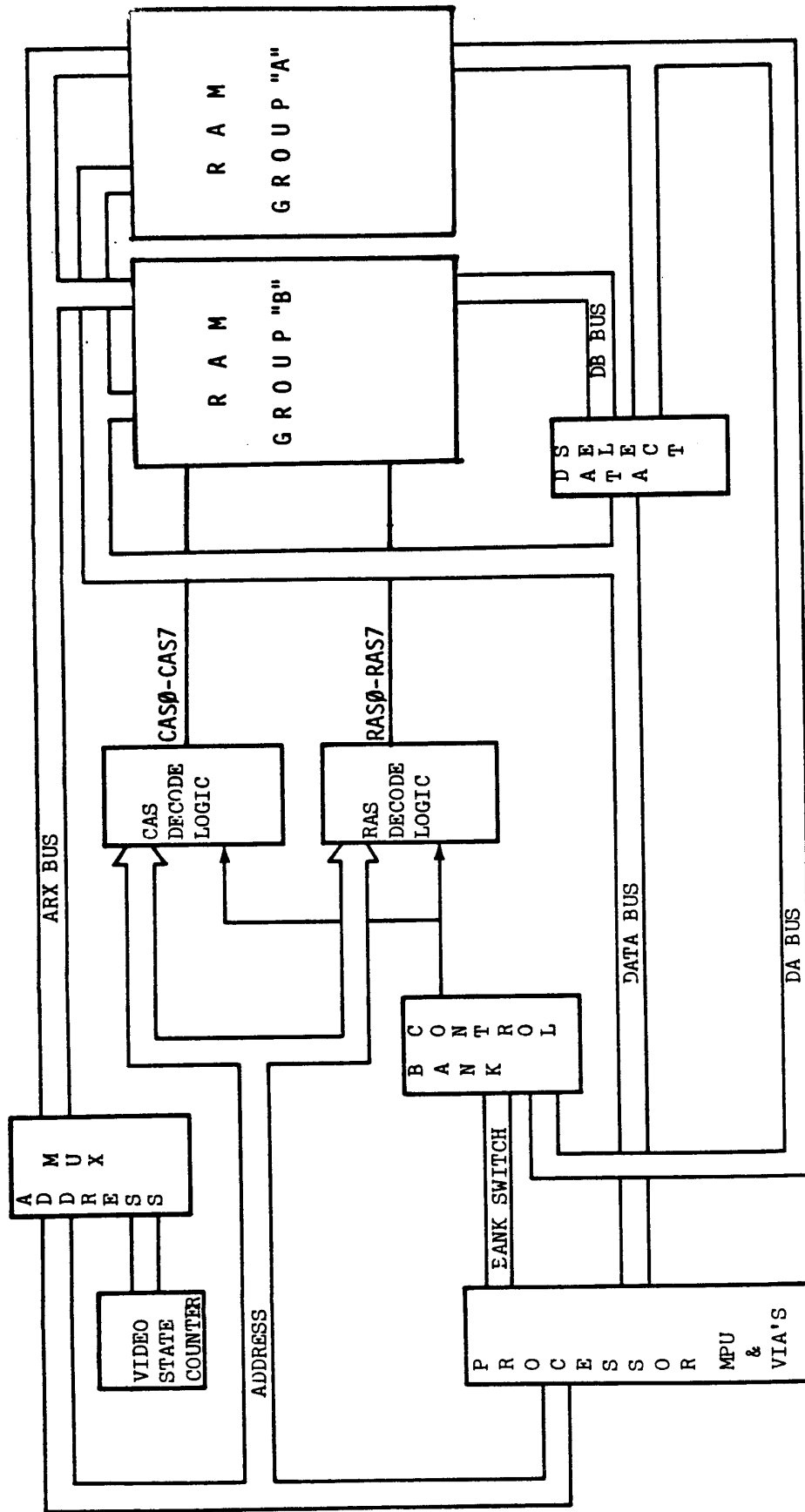


FIG 2.2



In this discussion, the processor is more than just the microprocessor chip; it also contains various external registers and control ROMs which enable the extended addressing and bank switch modes. [Refer to Figure 2.2.]

Very simply, the processor presents an address for memory cycle, which is multiplexed into the address bus, ARX, and is decoded to develop the RAS (Row Address Strobe) and CAS (Column Address Strobe) to the RAM array (RAM group A & B). The direction of the data is controlled by Read/Write*. The selection of which RAM group is being gated to the data bus is controlled by the CAS decode circuits.

The display is memory mapped, the Screen time-shares the RAM on the opposite phase of the processor clock. Both the screen and the processor are running at a 1MHz rate, which means that the RAM is running at a 2MHz rate. One new feature of the Apple /// is that the processor is able to make use of the other "phase" while the screen is off. In other words, any time the screen is off, the processor may run at a full 2MHz rate.

MEMORY ADDRESSING: BLOCK FLOW

As more detail is added, it is possible to see the basic elements of the complete memory system (For now, we are not considering any to the hardware or I/O).

- o In the memory addressing logic of Figure 2.3, the processor now shows the MPU and the two registers for expanded addressing capacity:
 - the bank address register
 - zero page register
- o The address circuit is comprised of two sections:
 - the address multiplexer
 - the RAS/CAS decoder
- o The RAM array is expanded to show the eight RAMs.

It should be noted that this diagram depicts a 128K system (with a 12V Memory board), and that each of the RAMs shown actually represents eight RAM chips, one for each bit. Each RAM contains 16K bytes. The dotted lines indicate the row of chips that contain the 32K RAM chips. These are actually two 16K RAMs which reside on the one IC package.

THE PROCESSOR

The MPU is isolated from the rest of the memory by various buffers, muxes, and registers. The mux switches in the Zero Page register whenever the MPU is attempting to reference the zero page.

BLOCK DIAGRAM MEMORY ADDRESS LOGIC

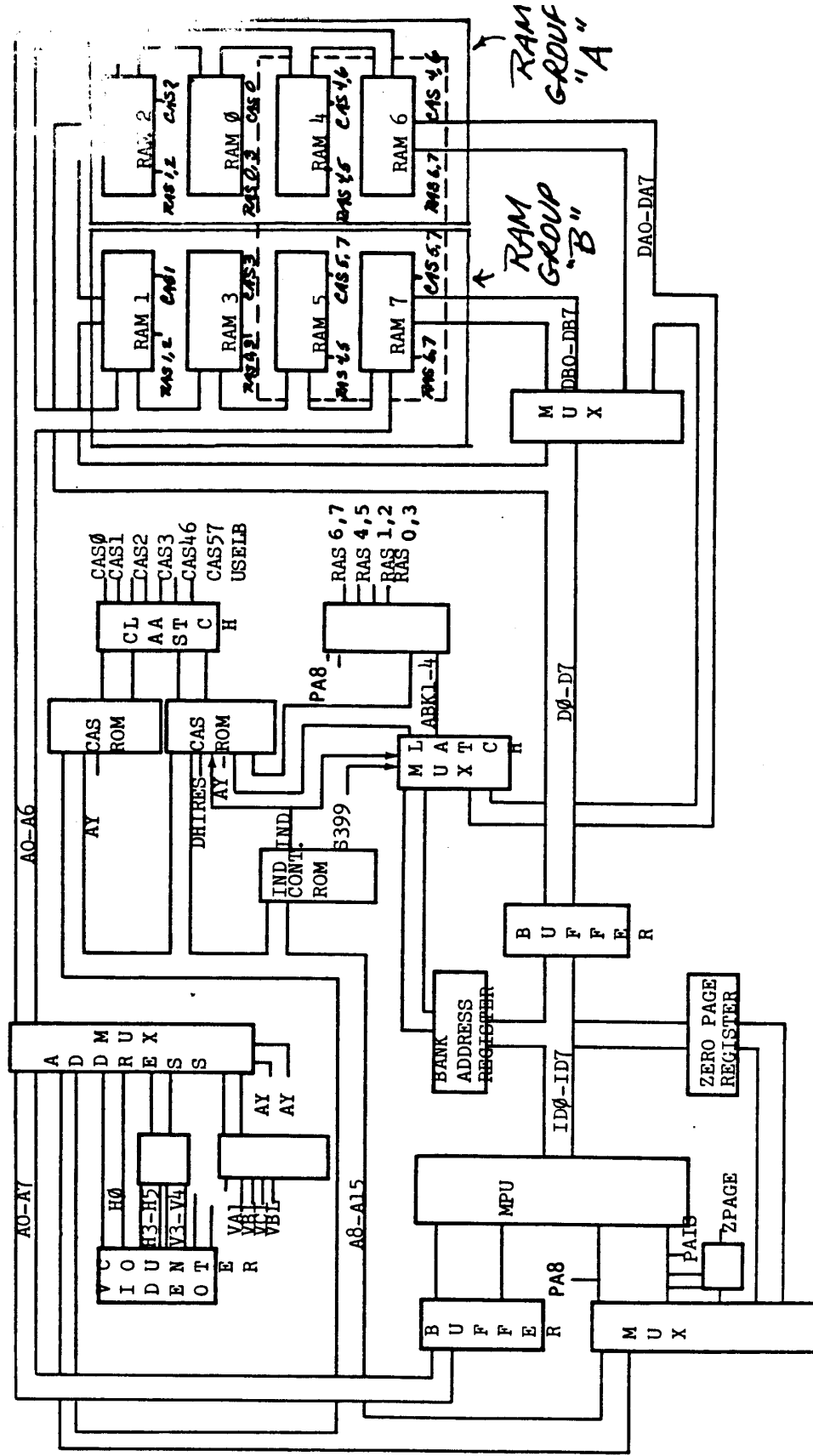


FIG 2.3



- o The register may contain the true zero page or may be set to any of 255 other values, under program control.
- o The zero page register resides in the VIA and is accessed at FF00. (On the Main Logic Board this is the VIA at location B6.)
- o The Bank Register is located in the other VIA and is accessed at FFEF (at location B5). Zero page selection is independent of bank selection.

MEMORY ADDRESS MULTIPLEXER

The Memory Address Mux provides the four sets of addresses to the RAM array. They are:

- o MPU RAS
- o Video RAS
- o MPU CAS
- o Video CAS

These are time multiplexed by the four states determined by AX* line which is held at a steady state, allowing the processor full access to the RAM.

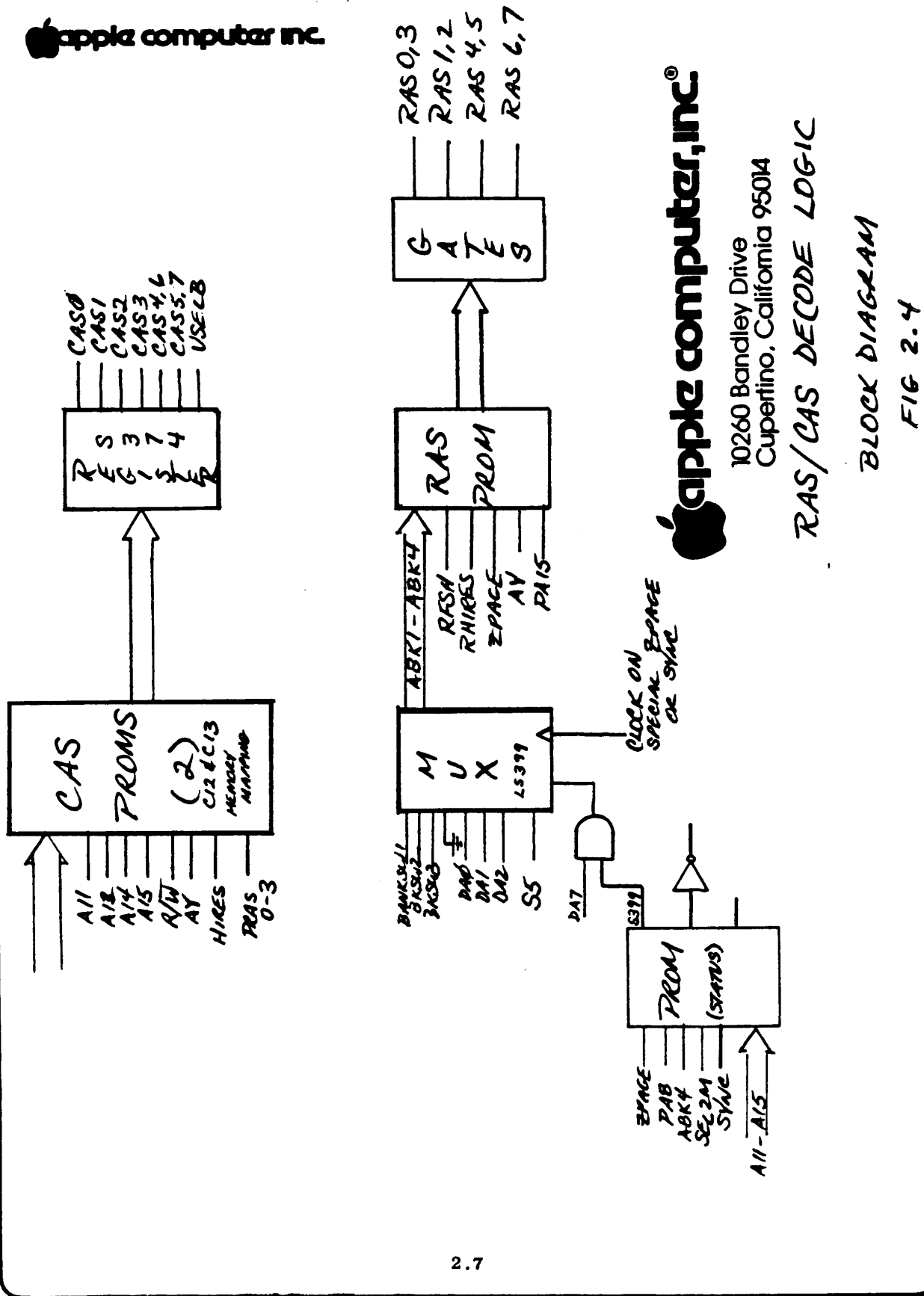
The Video addresses are much the same as in the Apple][. There is a minor difference in the Summing Circuit, but the technique of condensing undisplayed addresses is the same. There is an additional consideration in the Apple ///, which has a feature requiring additional control of the Video lines. This new feature is called slow scrolling of the screen.

Slow scrolling is accomplished in the Video Mux ROM by the arithmetic offset of the VA, VB, and VC lines. This offset causes characters to be fetched from memory in advance of where the screen actually thinks it is. The character array on the screen shifts up the number of dots determined by the binary weight of the VBX lines. The processor, by monitoring the Vertical Blanking, can then step the VBX lines and scroll the screen by moving in new lines at the bottom, removing the top line, and placing it at the bottom, thus rolling the display.

RAS/CAS DECODE

The RAS/CAS decode circuit is made from four ROMs, a latch, and a latching mux. The basic inputs to the circuit are:

- o the Address Bus
- o the Bank Switches
- o the Zero Page Select



10260 Bandy Drive
Cupertino, California 95014

RAS/CAS DECODE LOGIC

BLOCK DIAGRAM

FIG 2.4



INDIRECT ADDRESSING

LDA (Z PAGE), Y

Y REG = 0

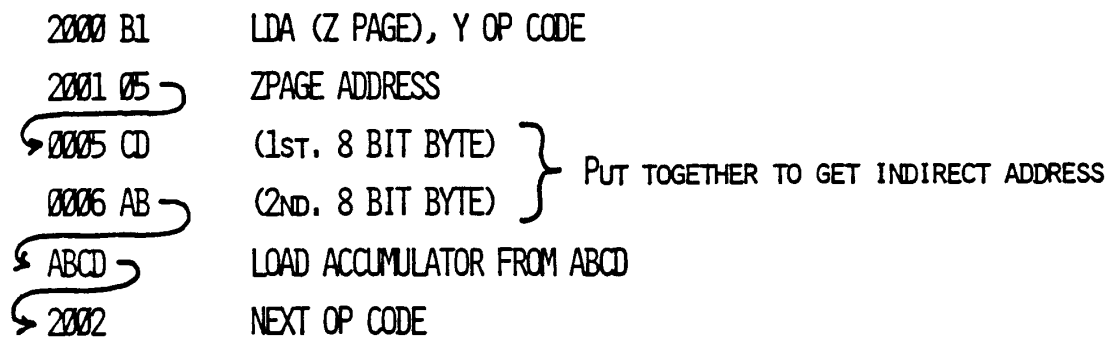
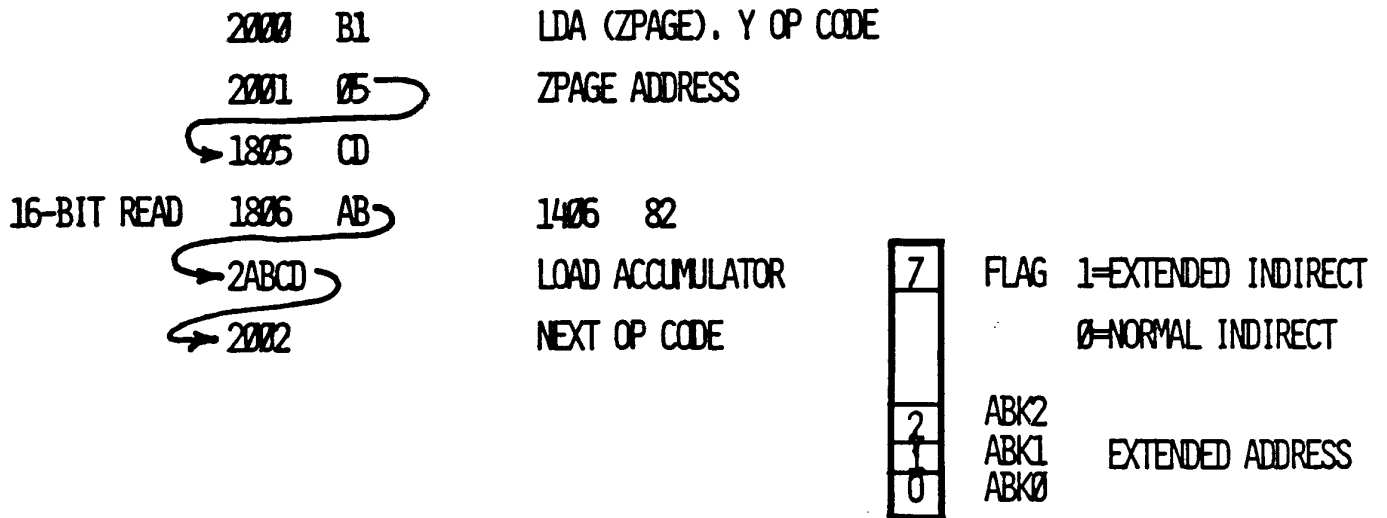


FIG 2.5

IF ZPAGE = 18-1F (IF ZERO PAGE FALLS BETWEEN THE RANGE OF 18-1F)





[Refer to Figure 2.4]

Normal or Direct Addressing looks at the address and current bank selection, and enables the appropriate array of 64K RAM. This Direct Addressing always uses RAM 0 and RAM 3. Bank switches determine which of the RAM pairs is used for the other 32K of RAM.

It should be noted that on any read cycle, one RAS line and two CAS lines are selected. In this way, two bytes are always presented to the video circuits and the data selector. During a write cycle only one of each is selected.

The dual byte read usually provides only the information for the new video modes, but there is a new special memory fetch cycle built into the hardware. It is a special extended Indirect Addressing scheme which places the entire memory in virtual access.

By using Indirect Addressing through the zero page containing the 16 bit address, an instruction can address any of the 64K bytes contained in the bank pair. Thus any of the 32K byte RAM banks can be paired with any of their neighbors to form a 64K byte virtual address space.

If, during a zero page reference the zero page register has a value between \$18 and \$1F (\$ means hexadecimal), a special Indirect Mode is called up. This mode looks at the sister fetched data byte on the RAM address bus and also looks at the high order bit. See Figures 2.5 and 2.6.

This special Indirect Mode is determined by the zero page register (X page = Z page EOR \$0C) If the bit is zero, the mode is not actualized and the reference continues in a normal manner in the presently selected bank arrangement. But if the High Order Bit (DA7) is high, the bank control mux latch switches to the state determined by the state of the DA0-DA2 lines. This allows the program to have access to another array of special zero pages.

When the system is in this special extended Indirect Mode, the I/O and LSI are totally disabled and the RAM is enabled to the data bus.

ALTERNATE STACK

Alternate Stack, the new feature of the Apple ///, is not shown in the block diagrams. One of the bits of the Environmental Register (from one of the VIAs) is the Alternate Stack Switch. If the Alternate Stack Switch is selected, the stack associated with that zero page is either the one after the zero page, if the zero page reference is even, or the one previous if the reference is odd (i.e., if zero page is 2B then the stack is located in 2C; if the zero page is 31 then the stack is in 30).

OTHER BANK SWITCHING

Earlier it was mentioned that the processor can access RAM associated with the addresses that are normally with I/O, ROM, and other circuits. Looking again at the Environmental Register, there are several switches that enable or disable I/O, ROM, and other circuit address decoding. If these switches are



selected to disable their associated function, the control ROM, which develops the enable for the RAM data selector, senses the fact that no hardware is being selected and allows RAM data to be read on the bus. No other special enables are needed since RAM is always read for every address presented. It should be noted that a special RAM write enable is used to prevent inadvertant writing into the RAM space associated with the hardware while the hardware is enabled.



MEMORY & MEMORY ADDRESSING APPENDIX

The attached figures and illustrations are provided for your reference. Little or no explanation has been provided.

This Appendix contains:

- o THE APPLE /// MEMORY MAP
- o MEMORY MAP SPACE ALLOCATIONS
- o ADDRESS LOGIC TRUTH TABLE
- o 128K 12V MEMORY BOARD: PHYSICAL MEMORY
- o THE 5V MEMORY BOARD: PHYSICAL MEMORY
- o ADDRESSING LOGIC EXPRESSIONS
- o MPU REGISTERS

APPLE /// MEMORY MAPSOS MEMORY ALLOCATION

Location	Assignment
0000-1FFF	SOS and Interpreter Workspace
2000-9FFF	Bank 0 Graphics Page 1 and 2
2000-9FFF	Bank 1 Program
2000-9FFF	Bank 2 Driver and Interpreter
A000-BFFF	Interpreter
C000-CFFF	I/O or SOS Kernal (Bank switchable to RAM)
D000-EFFF	SOS Kernal
F000-FFFF	Boot ROM OR SOS Kernal

ADDRESS ASSIGNMENT

ADDRESS (HEX)	ASSIGNMENT (FUNCTION)
0000-00FF	Zero Page
0100-01FF	Stack
0200-02FF	Input Buffer
0300-03FF	Open
0400-07FF	Lo-Res Display (Primary) and text
0800-0BFF	Lo-Res Display (Secondary) and text
0C00-0FFF	Open-Reserved for system space
1000-1FFF	Open
2000-3FFF	Hi-Res Pgl (Primary) switchable to RAM
4000-5FFF	Hi-Res Pgl (Secondary) switchable to RAM
6000-7FFF	Hi-Res Pg2 (Primary) switchable to RAM
8000-9FFF	Hi-Res Pg2 (Secondary) switchable to RAM
A000-BFFF	Open
C000-C07F	System I/O
C000	Keyboard "A" bus data
C001-C007	Same as C000 but not used
C008	Keyboard "B" bus data
C009-C00F	Same as C008 but not used
C010	Keyboard reset
C011-C02F	Not used in Apple III
C030	Toggle the speaker like in A-11
C031-C03F	Same as C030 but not used
C040-C040	Sound hardware beeper
C04E	Character Ram Disable
C04F	Character Ram Enable
C050	Clear Text Mode
C051	Set Text Mode
C052	Clear Mix Mode
C053	Set Mix Mode
C054	Clear PG2 Mode



C055	Set PG2 Mode
C056	Clear HIRES Mode
C057	Set HIRES Mode
C058	Clear EMSOT PDLO
C059	SET ENSOT PDLO
C05A	Clear PDL2 (A/D Addr 2)
C05B	Set PDL2
C05C	Clear PDLEN (A/D Ramp Start)
C05D	Set PDLEN
C05E	Clear AXCO (A/D Addr 1)
C05F	Set AXCO
C060,C068	Read SW0
C061,C069	Read SW1/MGNSW
C062,C06A	Read SW2
C063,C06B	Read SW3/SCO
C064,C06C	Read IRQ3
C065,C06D	Read IRQ4
C066,C06E	Read PDL0T (A/D Ramp Stop)
C067,C06F	Read MUXI (PRAS Control)
C070	Access Real Time Clock
C071-C07F	Same as C070 but not used
C080-C0FF	I/O Scot Device Enable
C08F	
C09X	NDevice Select 1
COAX	NDevice Select 2
COBX	NDevice Select 3
COCX	NDevice Select 4
COD0	Clear DS A0 A0,A1=0,0=no select
COD1	Set DS A0 1,0=Ena 1 Exit
COD2	Clear DS A1 0,1=Ena 2 Exit
COD3	Set DS A1 1,1=Ena 3 Exit
COD4	Clear Enable 1 Int
COD5	Set Enable 1 Int
COD6	Clear Side 2
COD7	Set Side 2
COD8	Clear SCR
COD9	Set SCR
CODA	Clear ENCWRT
CODB	Set ENCWRT
CODC	Clear ENSEL
CODD	Set ENSEL
CODE	Clear ENSIC
CODF	Set ENSIO
COE0	Clear DPH0 (also VAL)
COE1	Set DPH0
COE2	Clear DPH1 (also VBI)
COE3	Set DPH1
COE4	Clear DPH2 (also VCI)
COE5	Set DPH2
COE6	Clear DPH3
COE7	Set DPH3
COE8	Disable Motor Drive (strt 2 sec to) COE9 Enable Motor Drive
COEA	Enable Ext
COEB	Enable Int



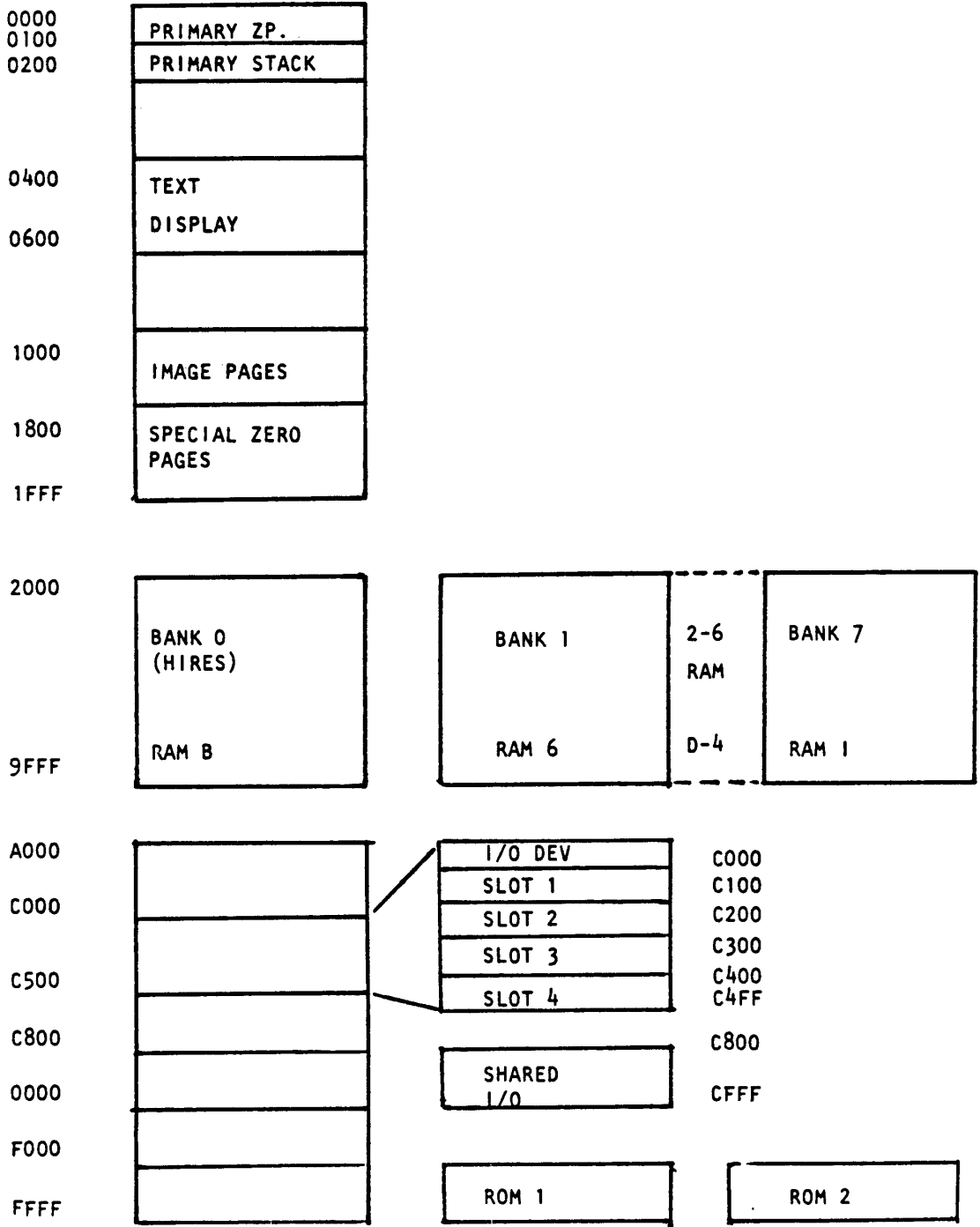
COEC	Clear Q6 Note:Q6,Q7 control read
COED	Set Q6 write, and sense write
COEE	Clear Q7 protect
COEF	Set Q7
COF0r	6551 Rec Data Reg
COF0w	6551 Xmit Data Reg
COF1r	6551 Status Reg
COF1w	6551 Program reset
COF2r/w6551	Command Reg
COF3r/w6551	Control Reg
C1XX	NIO Select 1
C100-C7FF	I/O Slot individual ROM Space
C100	Slot 1 Firmware
C1FF	
C2XX	NIO Select 2
C200-C2FF	Slot 2 Firmware
C3XX	NIO Select 3
C300-C3FF	Slot 3 Firmware
C3FF	
C4XX	NIO Select 4
C400-C4FF	Slot 4 Firmware
C500-C7FF	Run Space only
C800-CFFF	Expansion Rom Firmware
D000-DFFF	Open (system software)
E000	Bank switchable between Rom and Ram
FFCX	Always Ram
FFD0	Port B VIA-73 "Zero Page Reg"
FFD1	Port A VIA-73
FFD2	DDR B VIA-73
FFD3	DDR A VIA-73
FFD4	Timer 1 low Latch (w)/Counter (r)
FFD5	Timer 1 High Counter
FFD6	Timer 1 Low Latches
FFD7	Timer 1 High Latches
FFD8	Timer 2 Low Latch (w)/Counter (r)
FFD9	Timer 2 High Counter
FFDA	Shift Register (serial print)
FFDB	Aux Control Reg VIA-73
FFDC	Peripheral Control Register
FFDD	Interrupt Flag Register (73)
FFDE	Interrupt Enable Register (73)
FFDF	ORA/IRA With no handshake
FFE0	Port B (97) (sound and slot NMI)
FFE1	Port A (97) Banksw and IRQ's
FFE2	DDR B (97)
FFE3	DDR A (97)
FFE4	Timer 1 Low Latch/Counter
FFE5	Timer 1 High Counter
FFE6	Timer 1 Low Latches
FFE7	Timer 1 High Latches
FFE8	Timer 2 Low Latch/Counter
FFE9	Timer 2 High Counter
FFEA	Shift Register (97)
FFEB	Aux Control Register (97)



FFEC	Peripheral Control Register (97)
FFED	Interrupt Flag Register (97)
FFEE	Interrupt Enable Register (97)
FFEF	ORA/IRA with no handshake
FFE0	Ram/Rom Bank
FFF1	E/O Bank Switch
FFF2	2 MHz/MHZ Mode Switch
FFF3	Hires Bank Switch
FFF4	Screen Enable
FFF5	Display Modes
FFF6	Zero Page Register
FFF7	Interrupt Control
FFFF	



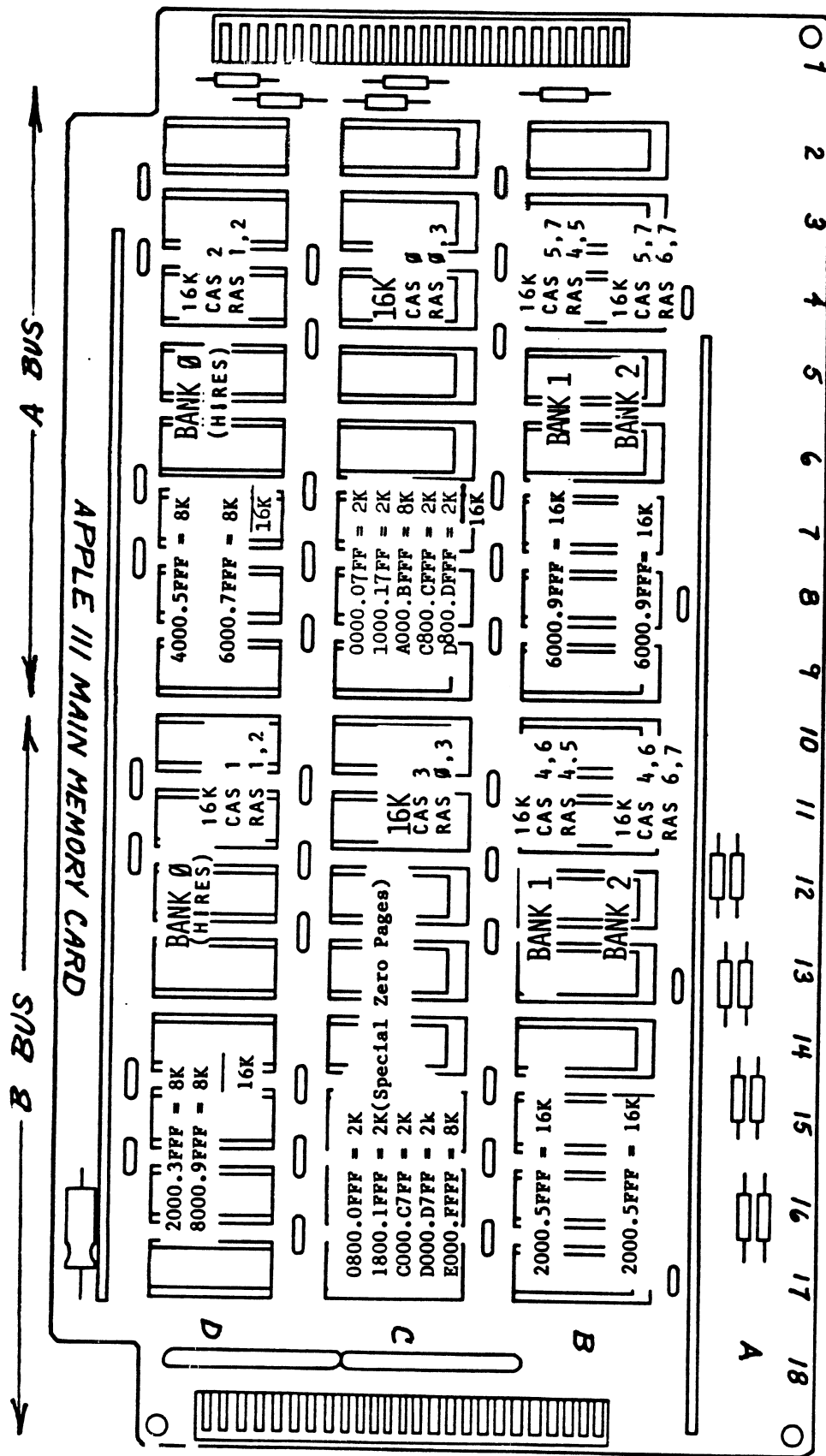
MEMORY MAP SPACE ALLOCATIONS



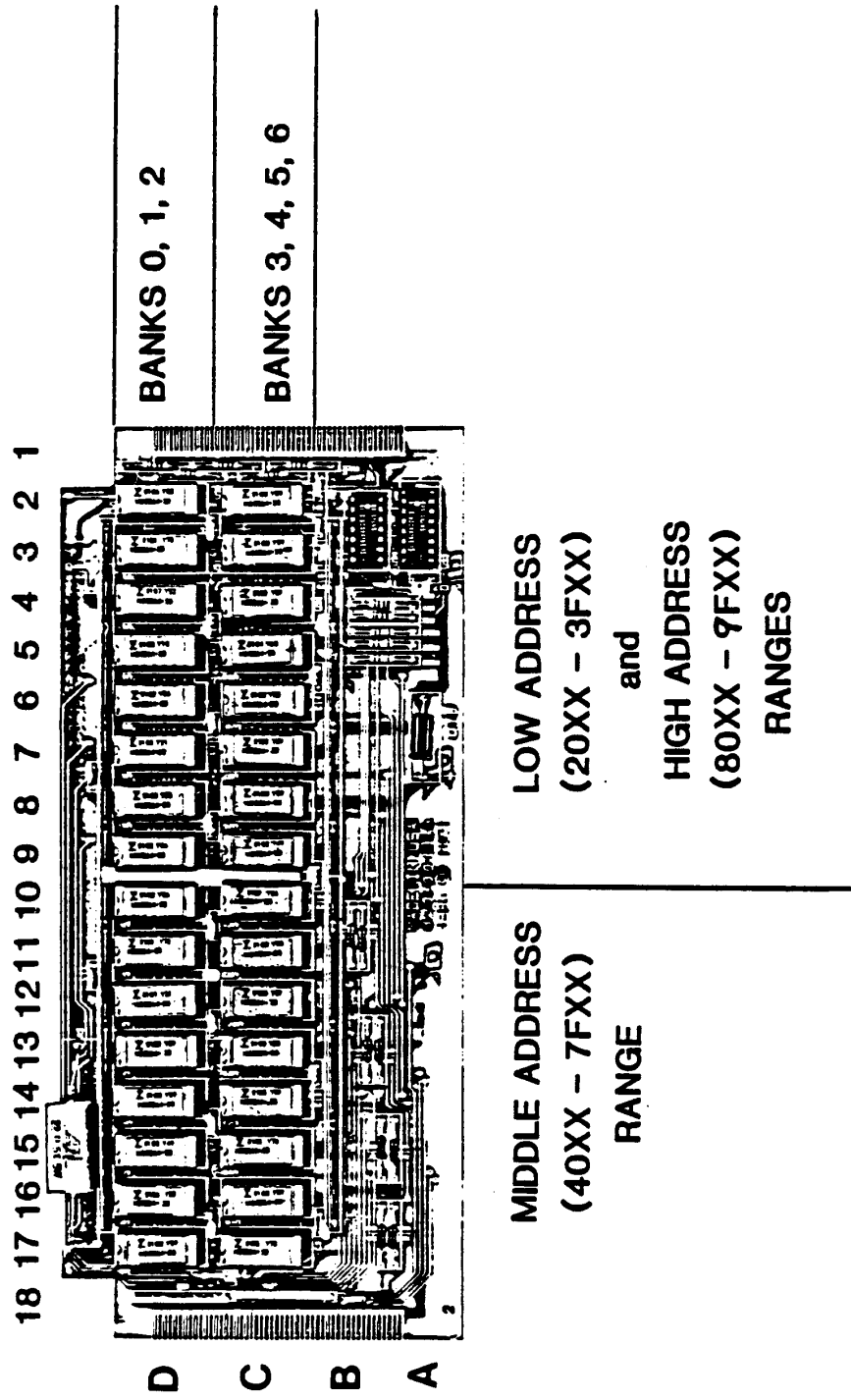
** FFCX, FFDX, FFEX ARE ENVIRONMENT ADDRESSES

\overline{AX}	\overline{AY}	AR ϕ	AR1	AR2	AR3	AR4	AR5	AR6	STATE FUNCTION
0	0	A ϕ	A1	A2	A3	A4	A5	A7	MPU GEN RAS ADD.
0	1	H ϕ	H1	H2	≤ 1	≤ 2	≤ 3	V ϕ	VIDEO GEN CAS APP
1	0	A6	A8	A9	A10 \oplus $\overline{RAS1}$	A11 \oplus $\overline{RAS2}$	A12	$\overline{RAS1}$ \oplus (A15 \oplus $\overline{RAS3}$)	MPU GEN CAS ADD
1	1	≤ 4	V1	V2-V5	MUX1	MUX2	MUX3	$\overline{PG2}$	VIDEO GEN CAS APP

ADDRESS LOGIC TRUTH TABLE



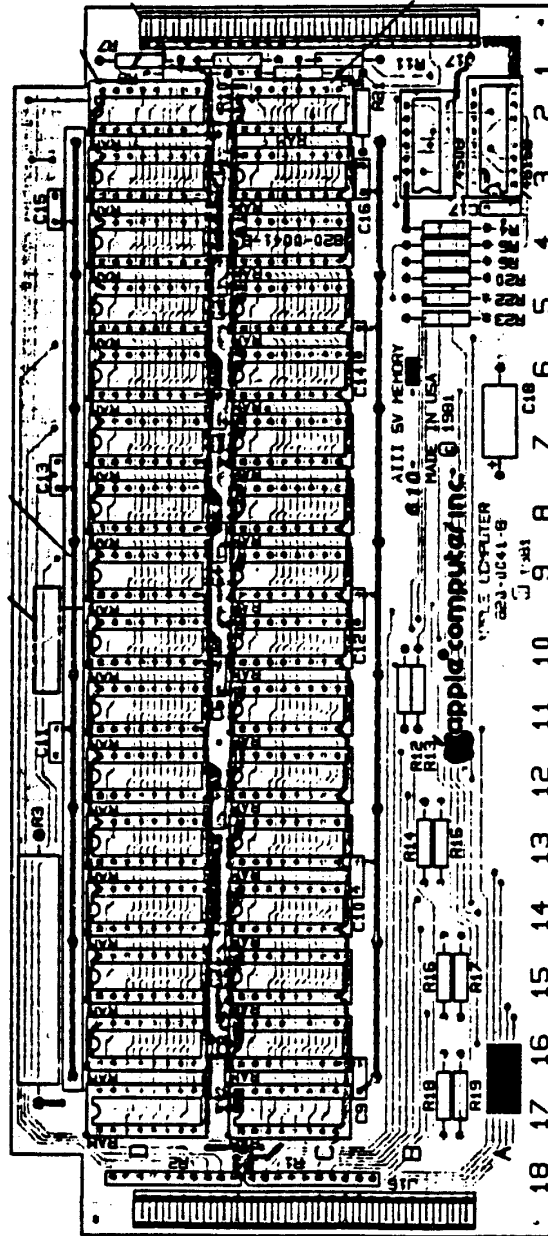
2.19



2.20



THE 5V MEMORY BOARD (256K)



4553: PROM 342-0061 64K RAM MT4264-20
 342-0063 D264A-15

HOW TO READ PROM (ROM) LOGIC EXPRESSIONS

- X' -- The single quote at the end of an expression means that the state of the signal is true when low, or it may represent the inversion of the state of the signal.

- * -- Defines a logic AND operation. The expressions on either side of the asterisk is ANDed.

- + -- Defines a logic OR operation. The expression on either side of the asterisk is OR'd.

- () -- Defines the boundaries of a logic expression. A new expression is defined by what ever is inside of the brackets.

RULES FOR INTERPRETATION

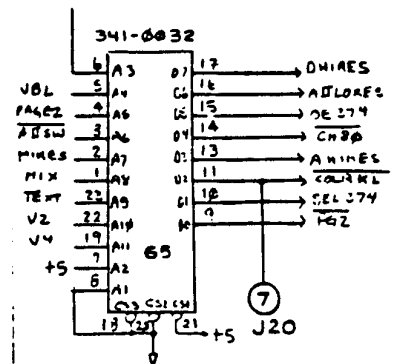
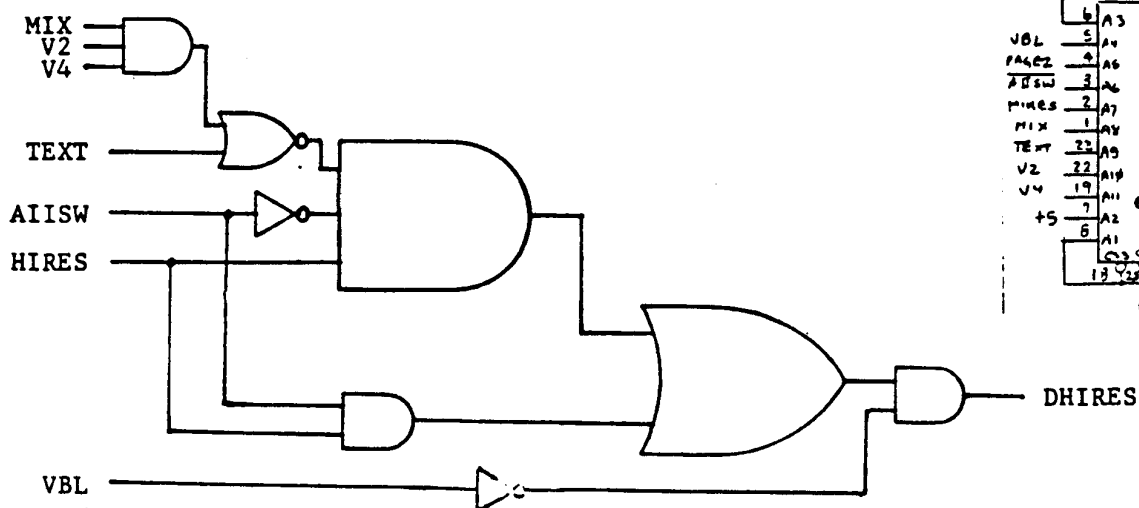
1. Always interpret (transform) the expression within the brackets first.
2. Interpret AND (*) logic operations before OR (+) operations.

EXAMPLE:

Given:	INPUTS: AIISW'		HIRES		TEXT
	MIX		V2		V4
	VBL				

OUTPUT: DHIRES = (AIISW*HIRES*(TEXT+MIX*V2*V4)'+AIISW'*HIRES)*VBL'

LOGIC REPRESENTATION:





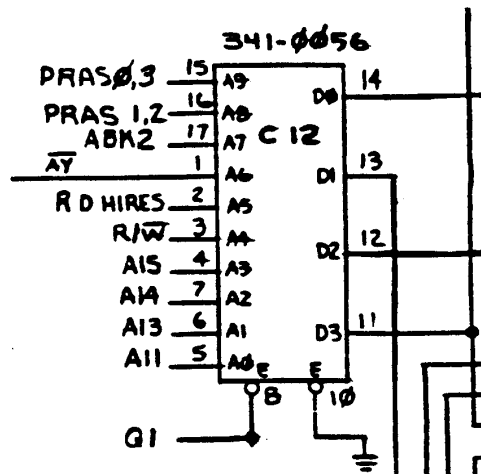
A=A11
 B=A13
 C=A14
 D=A15
 E=R/WN
 F=DHIRES
 G=AY'
 H=ABK2
 I=PRAS1,2
 J=PRAS0,3
 DO=PCASO'
 D1=PUSELB
 D2=PCAS3
 D3=PCAS3'

$$PCASO' = (PRASO, 3 * (DHIRES' * AY' + AY * (A15' * A14' * A13' * A11' * R/WN' + A15' * A14' * A13' * R/WN + A15 * A14 * A13 * A11)))'$$

$$PUSELB = PRASO, 3 * (A15' * A14' * A13' * A11 + A15 * A14 * A13 * A11' + A15 * A14 * A13) + PRASO, 3 * PRAS1, 2 * (A15' * A14' * A13 + A15 * A14 * A13') + PRASO, 3 * PRAS1, 2 * (A15' * A14' * A13 + A15 * A14 * A13') + PRASO, 3 * PRAS1, 2 * (A14' * A13' + A14 * A13) + PRASO, 3 * PRAS1, 2 * A14'$$

$$PCAS3 = PRASO, 3 * (DHIRES' * AY' + AY * (A15' * A14' * A13' * A11 + A15 * A14 * A13' * A11' + A15 * A14 * A13))$$

$$PCAS3' = (PRASO, 3 * (DHIRES' * AY' + AY * (A15' * A14' * A13' * A11 + A15 * A14 * A13' * A11' + A15 * A14 * A13)))'$$





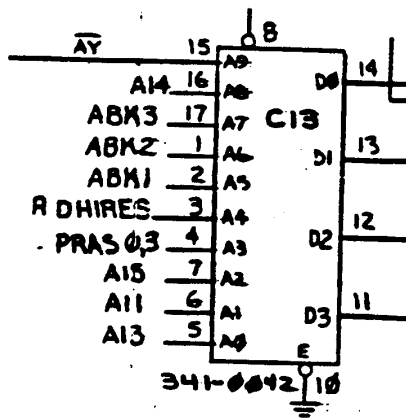
A=A13
 B=A11
 C=A15
 D=PRAS0,3
 E=DHIRES
 F=ABK1
 G=ABK2
 H=ABK3
 I=A14
 J=AY'
 D0=PCAS4,7'
 D1=PCAS5,6'
 D2=PCAS1'
 D3=PCAS2

$$PCAS4,7' = (AY * PRAS0,3 * (ABK1 * ABK2 * ABK3' + ABK3 * (ABK1' + ABK2')) * (A15' * A14) + AY * PRAS0,3 * ((ABK1' * ABK2 * ABK3' * A14 + ABK1 * ABK2 * ABK3' + ABK2' * ABK3 + ABK1' * ABK2 * ABK3 * A15') * (A14' * A13 + A14 * A13)))'$$

$$PCAS5,6' = (AY * PRAS0,3 * (ABK1 * ABK2 * ABK3' + ABK3 * (ABK1' + ABK2')) * (A15' * A14' * A13 + A15 * A14' * A14' * A13') + AY * PRAS0,3 * ((ABK1' * ABK2 * ABK3' * A15 + ABK1 * ABK2 * ABK3' + ABK2' * ABK3 + ABK1' * ABK2 * ABK3 * A15') * (A14' * A13' * A14 * A13)))'$$

$$PCAS1' = (DHIRES * AY' + AY * PRAS0,3 * (ABK3' * (ABK1' + ABK2') + ABK1 * ABK2 * ABK3) * (A15' * A14' * A13 + A15 * A14' * A13') + AY * PRAS0,3 * ((ABK2' * ABK3' + ABK1' * ABK2 * ABK3' * A15') * (A14' * A13' + A14 * A13)))'$$

$$PCAS2' = (DHIRES * AY' + AY * PRAS0,3 * (ABK3' * (ABK1' + ABK2') + ABK1 * ABK2 * ABK3) * A15' * A14 + AY * PRAS0,3 * ((ABK2' * ABK3' + ABK1' * ABK2 * ABK3' * A15) * (A14' * A13 + A14 * A13')))'$$

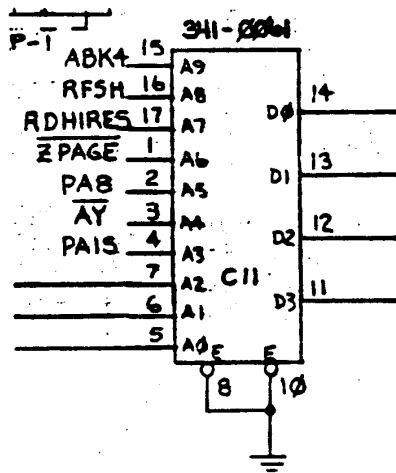




RAS 258

A=ABK1
 B=ABK2
 C=ABK3
 D=PA15
 E=AY'
 F=PA8
 G=ZPAGE'
 H=DHIRES
 I=RFSH
 J=ABK4
 D0=PRAS0,3
 D1=PRAS1,2
 D2=PRAS4,5
 D3=PRAS6,7

$$\begin{aligned}
 \text{PRAS0,3} = & \text{AY}' * (\text{DHIRES}' + \text{RFSH}) + ((\text{ABK4}' * (\text{ZPAGE}' * \text{PA8}')')' + \text{ABK4}' * (\text{ZPAGE}' * \text{PA8}')' \\
 & * \text{ABK1}' * \text{ABK2}' * \text{ABK3}') * \text{AY}' \text{ PRAS1,2} = \text{AY}' + \text{AY}' \text{ PRAS4,5} = \text{AY}' + \text{AY}' * (\text{ABK4}' * (\text{ZPAGE}' * \\
 & \text{PA8}')')' * (\text{ABK1}' * \text{ABK2}' * \text{ABK3}' + \text{ABK1}' * \text{ABK2}' * \text{ABK3}' * \text{PA15}' + \text{ABK1}' * \text{ABK2}' * \text{PA15}' \\
 & + \text{ABK1}' * \text{ABK2}' * \text{ABK3}') \text{ PRAS6,7} + \text{AY}' * \text{DHIRES}' + \text{AY}' * (\text{ABK4}' * (\text{ZPAGE}' * \text{PA8}')')' * \\
 & (\text{ABK1}' * (\text{ABK2}' + \text{ABK3}')) + \text{AY}' * \text{ABK4}' * (\text{ZPAGE}' * \text{PA8}')' * (\text{PA15}' * \text{ABK1}' * (\text{ABK2}' \\
 & + \text{ABK3}') + \text{PA15}' * \text{ABK1}' * (\text{ABK2}' + \text{ABK3}'))
 \end{aligned}$$



2.25



RAS.2.TEXT

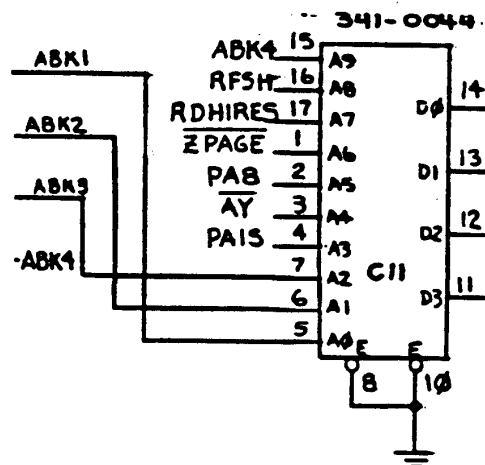
A=ABK1
 B=ABK2
 C=ABK3
 D=PA14 5
 E=AY'
 F=PA8
 G=ZPAGE
 H=DHIRES
 I=RFSH
 J=ABK4
 DO=PRAS0,3
 D1=PRAS1,2
 D2=PRAS4,5
 D3=PRAS6,7

PRAS0,3 =AY'* (DHIRES'+RFSH) + ((ABK4* (ZPAGE*PA8'))' +ABK1*ABK2*ABK3)
 *AY

PRAS1,2 =AY'* (DHIRES+RFSH) =AY* (ABK1'* ABK2'*ABK3'* (ABK4* (ZPAGE*PA8')
 '**PA15)'+ABK1*ABK2*ABK3) +AY*ABK3'* (ABK1'* ABK2* ABK4* (ZPAGE*
 PA8') '*PA15+ABK1*ABK2*(ABK4* (ZPAGE*PA8') *PA15)')

PRAS4,5 =RFSH*AY'+AY*ABK2'*ABK3'* (ABK1'*ABK4* (ZPAGE*PA8') 'PA15+ABK1*
 (ABK4* (ZPAGE*PA8') '*PA15)')

PRAS6,7 =RFSH*AY'+AY*ABK3'* (ABK1'*ABK2'*ABK4* (ZPAGE*PA8') '*PA15+ABK1'*
 ABK2* (ABK4* (ZPAGE*PA8') '*PA15)')



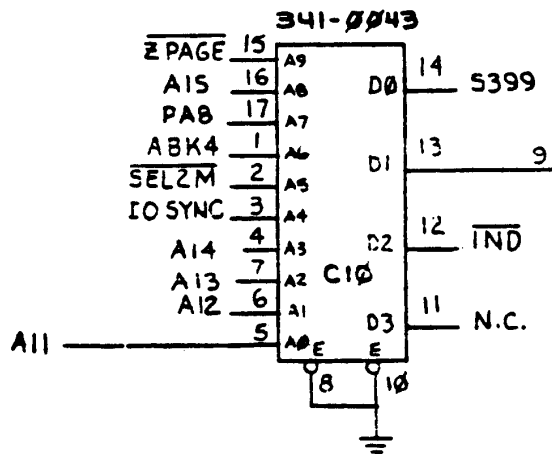


A=A1
 B=A12
 C=A13
 D=A14
 E=IOSYNC
 F=SEL2M'
 G=ABK4
 H=PA8
 I=15
 K=PAGE'
 D0=S399
 D1=PRDY'
 D2=IND'

$$S399 = ZPAGE * PA8' * A15' * A14' * A13' * A12 * A11$$

$$PRDY' = IOSYNC * SEL2M * ABK4$$

$$IND' = (ABK4 * (ZPAGE * PA8')')$$





U175 1. TEXT

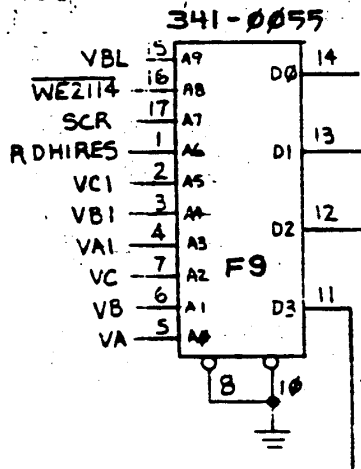
A=VA
 B=VB
 C=VC
 D=VA1
 E=VB1
 F=VC1
 G=DHIRES
 H=SCR
 I=WE2114'
 J=VBL
 DO=MUX1
 D1=MUX2
 D2=MUX3
 D3=ENHREG'

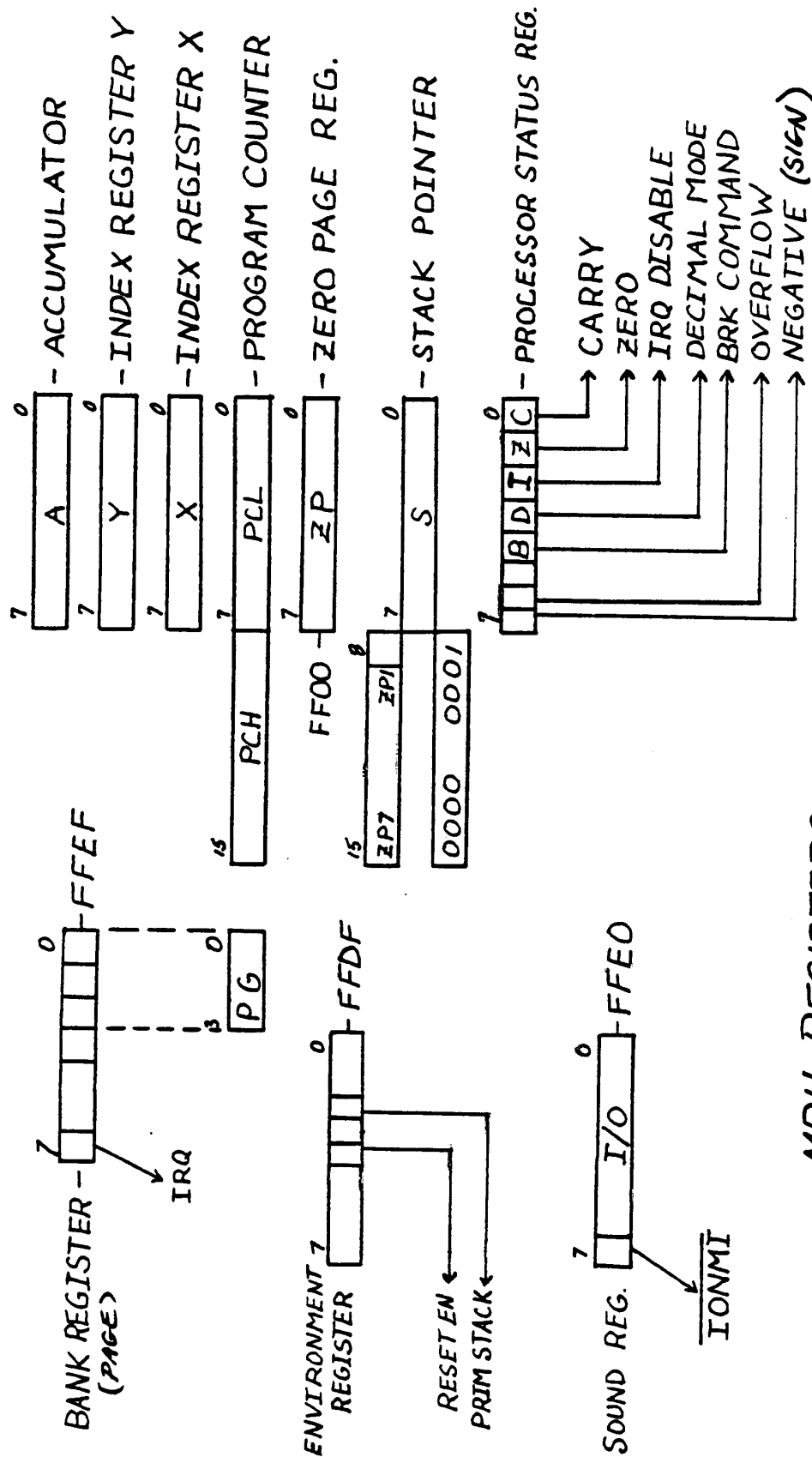
$$\text{MUX1} = (\text{DHIRES}' + \text{SCR} * (\text{VA} * \text{VA1}' + \text{VA}' * \text{VA1}) + \text{SCR}' * \text{VA}) * (\text{VBL}' + \text{WE2114}) + \text{VBL} * \text{WE2114}'$$

$$\text{MUX2} = \text{DHIRES} * (\text{SCR} * (\text{VA} * \text{VA1}' * (\text{VB} * \text{VB1}' + \text{VB}' * \text{VB1})' + (\text{VA} * \text{VA1})' * (\text{VB} * \text{VB1}' + \text{VB} * \text{VB1})) + \text{VB} * \text{SCR}')$$

$$\text{MUX3} = \text{DHIRES} * (\text{SCR} * ((\text{VA} * \text{VA1}' * (\text{VB} + \text{VB1}) + \text{VB} * \text{VB1}) * (\text{VC} * \text{VC1}' + \text{VC}' * \text{VC1}) + (\text{VA} * \text{VA1}' * (\text{VB} + \text{VB1}) + \text{VB} * \text{VB1}) * (\text{VC} * \text{VC1}) + \text{VC} * \text{SCR}')$$

$$\text{ENHREG}' = \text{DHIRES}' * \text{WE2114}'$$





MPU REGISTERS

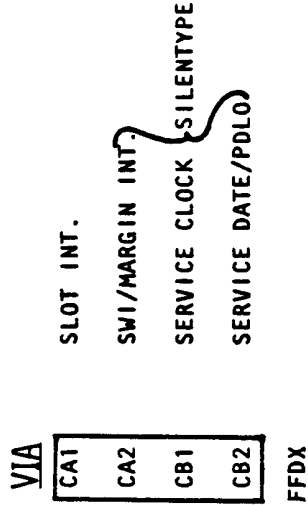
I/O DEVICE STROBES

0X	1X	2X	3X	4X	5X	6X	7X	8X	9X	AX	BX	CX	DX	EX	FX	COXX
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	------

- KEYBOARD READ
- RESET KEYBOARD
- N.C.
- Speake Toggle
- Beeper
- Display Register
- Switch Register
- Real Time Clock
- N.C.

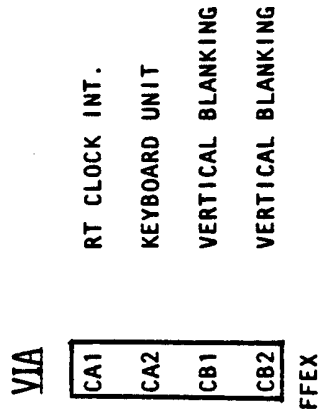
KEYBOARD DATA

0	1	2	3	4	5	6	7	C000
ASCII0	ASCII1	ASCII2	ASCII3	ASCII4	ASCII5	ASCII6	KBO FLAG	

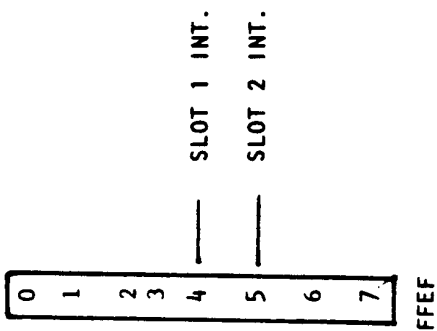


KEYBOARD STATUS

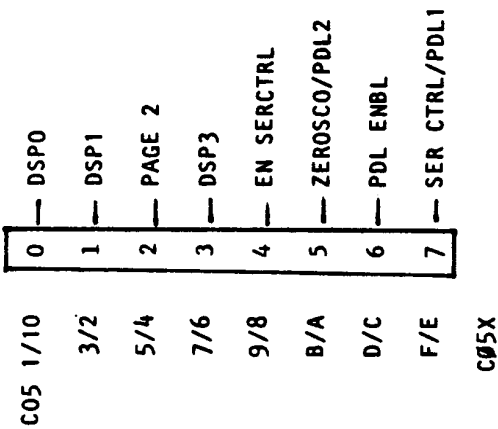
0	1	2	3	4	5	6	7	C008
ANY KEY DOWN	SHIFT	CONTROL	CAPSLCK	APPLE I	APPLE II	KEYBOARD PRESENT	ASCTI7	



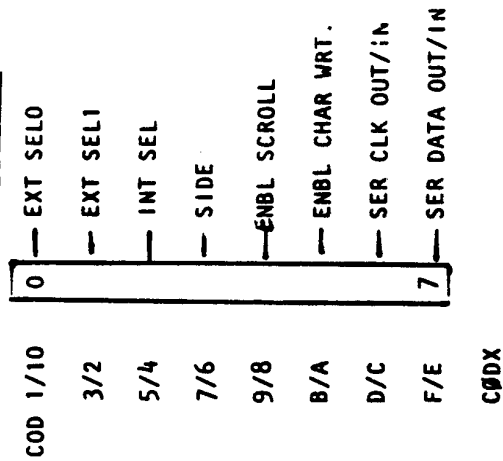
BANK PAGE REGISTER



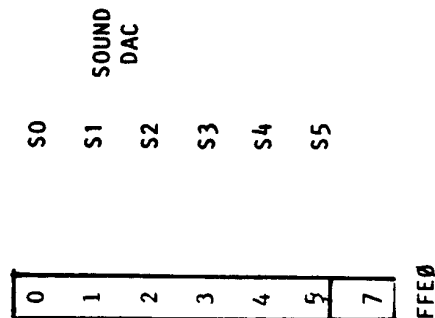
DISPLAY REGISTER



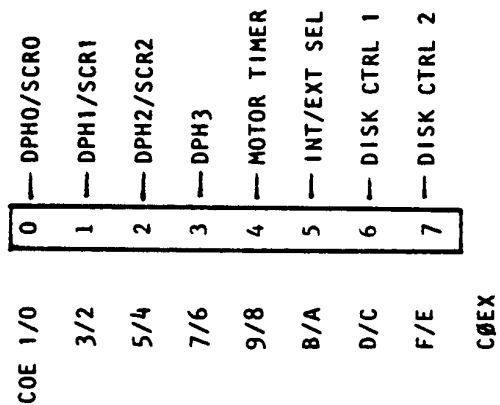
I/O CONTROL REGISTER



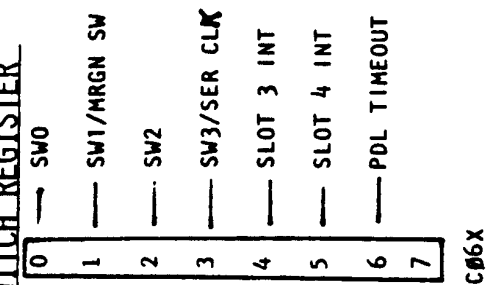
SOUND REGISTER



DISK REGISTER



SWITCH REGISTER

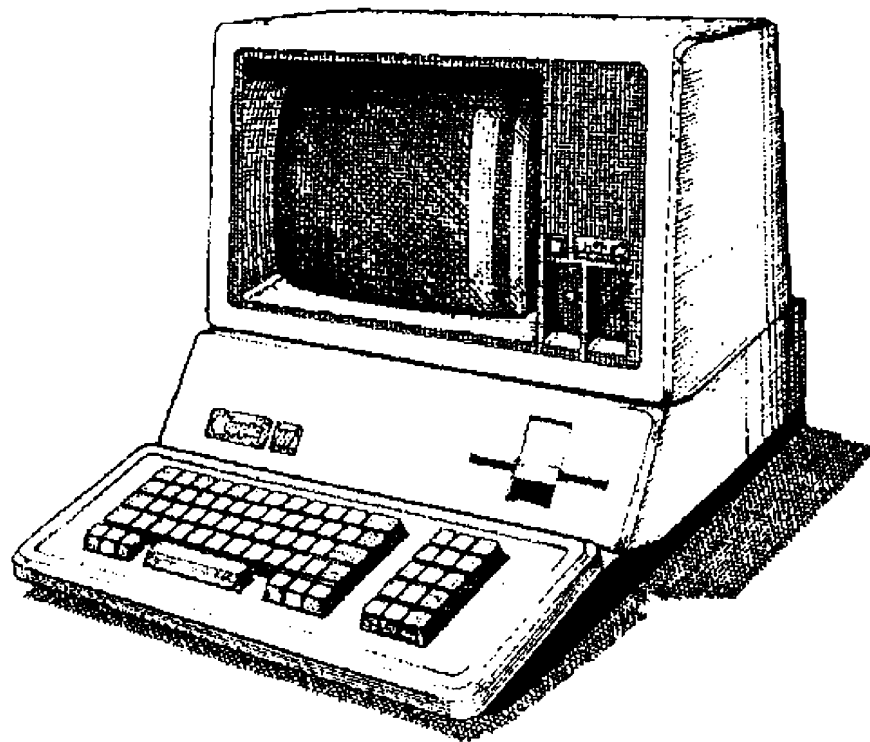


2.31



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 3 • The Versatile Interface Adapter (VIA)

Written by Apple Computer • 1982



THE VERSATILE INTERFACE ADAPTER

GENERAL

The Versatile Interface Adapter (VIA), as used in the Apple ///, is a very flexible I/O control device which minimizes the discrete control circuitry on the Main Logic Board. There are two VIAs used in the system.

Each VIA contains two 8-bit I/O ports, a serial port, and two 16-bit interval timers, as shown in the VIA Block Diagram. Each of the sections is very flexible and can be utilized in many operating modes. In the Apple ///, however, some of these modes cannot be used because of certain hardware design considerations.

Control of peripheral devices is handled primarily through two 8-bit bi-directional ports. Each line can be programmed as either an input or an output. Several peripheral I/O lines can be controlled directly from the interval timers for generating programmable frequency square waves, or for counting externally generated pulses. To facilitate control of the many powerful features of this chip, an interrupt flag register, an interrupt enable register, and a pair of function control registers are provided.

Before we get into a functional description of how the VIA is used in the Apple ///, let's first go through a pin description of the it.

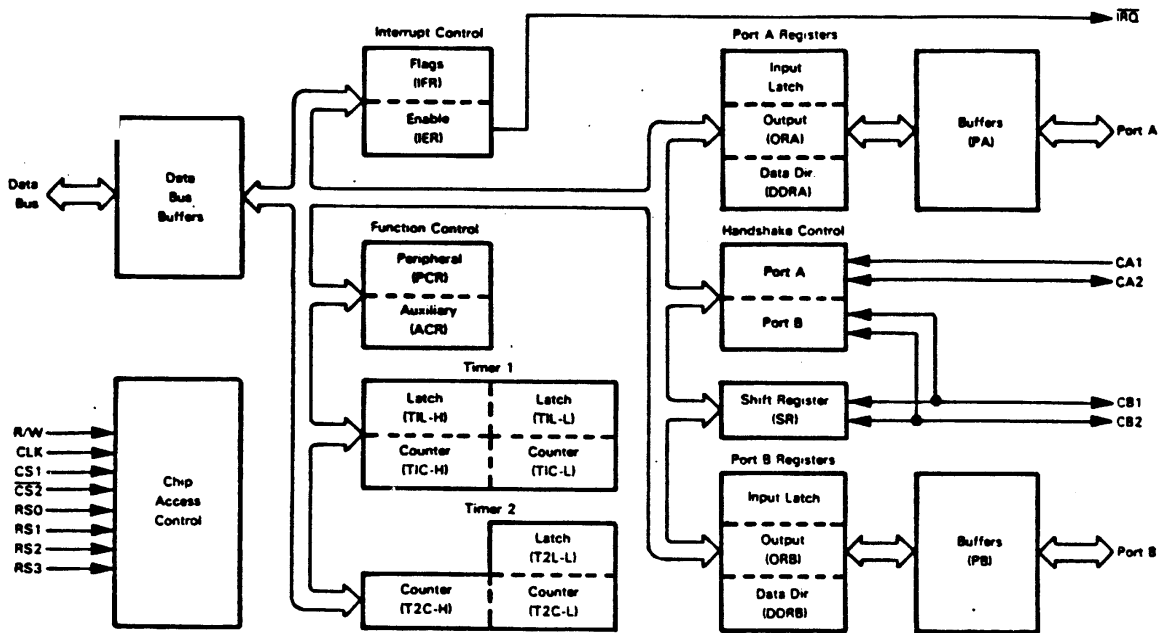
VIA [6522] PIN DESCRIPTIONS

RES* (Reset) (34)

The Reset input clears all internal registers to logic 0 (except T1 and T2 latches and counters, and the shift register). This places all peripheral interface lines in the input state, disables the timers, shift register, etc., and disables interrupting from the chip. On both VIAs this pin (34) is connected to the RESET* of the system. The system generates the reset at power on, or at depression of the Reset switch in conjunction with the Control key. (Note the latter may be disabled.)

O2 (Input Clock) (25)

The Input Clock is the system PRE1M (PRE-1 MHZ) clock which operates at 1MHZ and is used to trigger all data transfers between the system processor and the VIA. The PRE1M, developed in the system timing circuits, is used within the device to clock the various functions of the registers and timers.



Block Diagram of the 6522 Versatile Interface Adapter

Addressing 6522 VIA Internal Registers

Label	Select Lines				Addressed Location
	RS3	RS2	RS1	RS0	
DEV	0	0	0	0	Output register for I/O Port B
DEV+1	0	0	0	1	Output register for I/O Port A, with handshaking
DEV+2	0	0	1	0	I/O Port B Data Direction register
DEV+3	0	0	1	1	I/O Port A Data Direction register
DEV+4	0	1	0	0	Read Timer 1 Counter low-order byte Write to Timer 1 Latch low-order byte
DEV+5	0	1	0	1	Read Timer 1 Counter high-order byte Write to Timer 1 Latch high-order byte and initiate count
DEV+6	0	1	1	0	Access Timer 1 Latch low-order byte
DEV+7	0	1	1	1	Access Timer 1 Latch high-order byte
DEV+8	1	0	0	0	Read low-order byte of Timer 2 and reset Counter interrupt Write to low-order byte of Timer 2 but do not reset interrupt
DEV+9	1	0	0	1	Access high-order byte of Timer 2, reset Counter interrupt on write
DEV+A	1	0	1	0	Serial I/O Shift register
DEV+B	1	0	1	1	Auxiliary Control register
DEV+C	1	1	0	0	Peripheral Control register
DEV+D	1	1	0	1	Interrupt Flag register
DEV+E	1	1	1	0	Interrupt Enable register
DEV+F	1	1	1	1	Output register for I/O Port A, without handshaking



R/W* (Read/Write) (22)

The direction of the data transfers between the 6522 and the system processor is controlled by the R/W* line, which is driven by the Internal Read/Write, I R/W*, signal generated by the processor.

If pin 22 is low, data will be transferred out of the processor into the selected VIA register, as in a write operation.

If R/W is high, data will be transferred out of the selected 6522 register, as in a read operation.

DB0-DB7 (Data Bus) (33-26)

The eight bi-directional data bus lines (in the Internal Data Bus) are used to transfer data between the VIA and the system processor.

- o During read cycles, the contents of the selected register are placed on the data bus lines and transferred into the processor.
- o During the write operation, these lines are high-impedance inputs and data is transferred from the processor into the selected register.
- o When the 6522 is unselected, the data bus lines are at high-impedance. These lines are connected to the Internal Data Bus and are not accessible from the outside world.

CS1, CS2* (Chip Selects) (24,23)

These two chip select lines are connected to direct decodes of the processor's address bus.

- o CS1 on both VIAs is controlled by the signal CS6522, a signal which is qualified by other system consideration.
- o CS2 is driven by the signal FFDX for IC at location B5, and FFEX for IC9 at location B6. The selected VIA register will be accessed when CS1 is high and CS2 is low.

RS0-RS3 (Register Selects) (38-35)

The four Register Select inputs permit the system processor to select one of the 16 internal registers, each of which performs a specific function, of the VIA as shown in the accompanying table.

IRQ (Interrupt Request) (21)

The Interrupt Request output goes low whenever an internal (to the VIA) flag is set and the corresponding interrupt enable bit is at 1. This output is



open-drain" to allow the interrupt request signal to be "wired-OR"ED" with other equivalent signals in the system.

PA0-PA7 (Peripheral A Port) (2-10)

The PA port consists of 8 lines which can be individually programmed to act as inputs or outputs under control of a data direction register.

The output is controlled by an output register. Input data may be latched into an internal register under control of the CA1 line. All of these modes of operation are controlled by the system processor through the internal control registers.

On standard TTL load, these lines are present in the input mode. Conversely, on standard TTL load, they will drive in the output mode.

CA1, CA2 (Peripheral A Control Lines) (40,39)

The two PA control lines can act as interrupt inputs (as used in the Apple ///) or as handshake outputs. Each line controls an internal interrupt flag with a corresponding interrupt enable bit.

PB0-PB7 (Peripheral B Port) (11-17)

The PB port consists of 8 bi-directional lines which are controlled by an output register and a data direction register in much the same manner as the PA port.

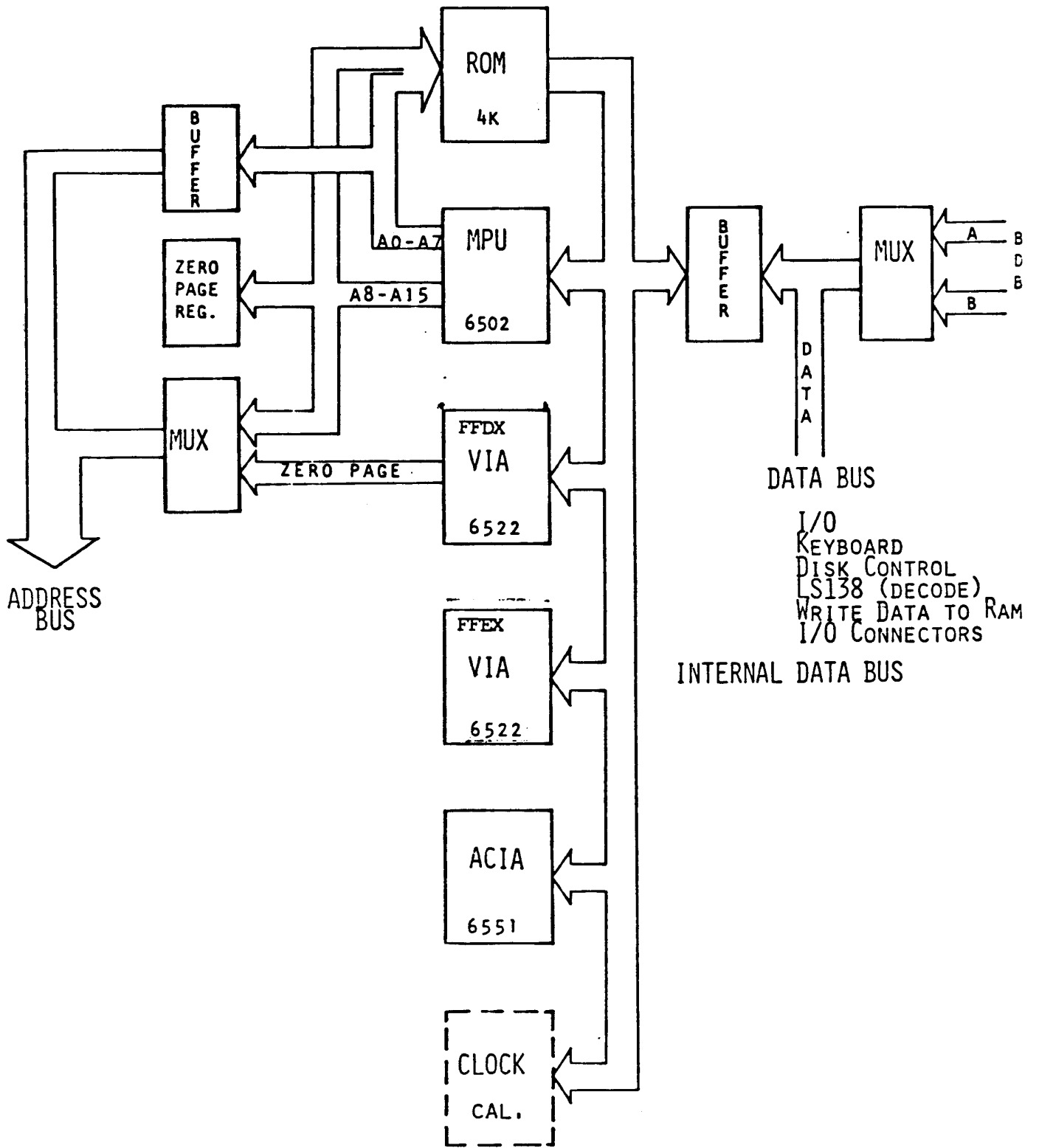
The polarity of the PB7 output signal can be controlled by one of the interval timers, while the second timer can be programmed to count pulses on the PB6 pin.

Peripheral B lines represent one TTL load in the input mode and drive one standard load in the output mode.

CB1, CB2 (Peripheral B Control Lines) (18-19)

The PB control lines act as interrupt inputs or as handshake outputs. As with the CA lines, each line controls an interrupt flag.

These lines also act as a serial port under control of the Shift Register. They have loading and driving characteristics identical to the CA control lines.





VIA FUNCTIONAL DESCRIPTION

Port A and Port B Operation

Each 8-bit peripheral port has a Data Direction Register for specifying whether the Peripheral pins are to act as inputs or outputs.

- o A "0" in a bit of the DDR causes the corresponding peripheral pin to act as an input.
- o A "1" causes the pin to act as an output.

Each peripheral pin is also controlled by a bit in the Output Register and a bit in the Input Register. When the pin is programmed as an output, the voltage on the pin is controlled by the corresponding bit of the output Register. A one in the ORX causes the output to go high; a zero causes the output to go low.

Data may be written into ORX bits corresponding to pins which are programmed as inputs. In this case, however, the output signal is unaffected.

The IRB Register operation is similar to that of the IRA. For pins programmed as outputs, however, there is a difference. When reading the IRB, it is the bit stored in the ORB that is sensed. That means the buffering and gating on the two ports differ in respect to pins programmed as outputs.

See the figures below detailing the data bytes programmed into the DDRs and the two control registers for each of the VIAs. Compare these with the environments of each device.

Timer Operation

Interval Timer T1 consists of two 8-bit latches and a 16 bit counter. The latches are used to store data which is to be loaded into the counter. After loading, the counter decrements at the rate of PRE1M.

Upon reaching zero, an interrupt flag will be set; if the interrupt is enabled, the IRQ* will go low. The timer will then disable any further interrupts, or will automatically transfer the contents of the latches into the counter and will continue to decrement.

In addition, the timer may be programmed to invert the output signal on a peripheral pin each time it "times out." Each of these modes is discussed below.

Timer 1 One-Shot Mode

The interval timer one-shot mode allows generation of a single interrupt for each timer load operation. As with any interval timer, the delay between the write T1C-H (FFD5, FFE5) operation and generation of the processor interrupt

Apple Computer Inc.

is a direct function of the data loaded into the timer.

In the one-shot mode, writing into the high order latch has no effect on the operation of Timer 1. However, it is necessary to assure that the low order latch contains the proper data before initiating the count-down with a "write T1C-H" operation.

When the processor writes into the high order counter

- o the T1 interrupt flag is cleared;
- o the contents of the low order latch are transferred into the low order counter;
- o the timer again begins to decrement at the PRE1M rate.

After the timer reaches its time out, it continues to decrement until it is reset with the proper write operation.

The processor may read the current count of the timer to determine how long it has been since the interrupt has been set. Reading the counter does not reset the interrupt flag or the timer.

Timer 1 Free-Run Mode

The most important advantage associated with the latches in T1 is the ability to produce a continuous series of evenly spaced interrupts. These interrupts are accomplished in the "free-running mode."

In the free-running mode, the interrupt flag is set each time the counter reaches zero. However, instead of continuing to decrement from zero, the timer automatically reloads the contents of the high and low latches, and continues to decrement from there. In this mode, the interrupt flag can be cleared by writing T1C-H, or reading T1C-L, or by writing directly to the interrupt flag.

It is not necessary to rewrite the timer to enable setting the interrupt flag on the next time-out. All of the interval timers are "retriggerable." Rewriting the counter will always re-initialize the time-out period.

Timer 2 Operation

Timer 2 operates as an interval timer in the one-shot mode only, or as a counter of negative pulses on the PB6 peripheral pin. A single bit in the ACR is provided for this mode selection.

Timer 2 is comprised of a write-only low order latch, a read-only low order counter, and a read/write high order counter.

Timer 2 One Shot-Mode



As an interval timer, T2 operates very much like T1 in the one-shot mode. Setting of the interrupt flag, however, will be disabled after the first time out, and it will not be set again until the write T2C-H operation.

The flag can be reset by reading T2C-L, or by writing T2C-H.

Timer 2 Pulse Counting Mode

In the pulse counting mode, T2 serves primarily to count a predetermined number of negative-going pulses on the P6 pin. This is accomplished by first loading a number into T2, which clears the interrupt flag and allows the counter to be decremented by the pulses on PB6. The interrupt flag will be set when T2 is decremented to zero. The counter will continue to decrement.

Note that it is necessary to rewrite the timer to re-enable the subsequent interrupt flags.

Shift Register Operation

The Shift Register performs serial data transfers into and out of the CB2 pin under control of an internal modulo-8 counter. Shift pulses can be applied to the CB1 pin from an external source or, with the proper mode selection, shift pulses generated internally will appear on the CB1 pin for controlling external devices.

The control bits which set the various register modes are located in the ACR. In total, there are eight modes for this register. The primary use of the shift register is control of the serial printer port.

In the FFEX VIA, the shift register can be activated to "count" 8 VBL (Vertical Blanking) pulses. The shift register sets its interrupt flag each time it completes 8 shifts.

Refer to the figure below for descriptions of the various modes of the shift register.

Interrupt Operation

Controlling interrupts within the VIA involves three principal operations. These are:

- o flagging the interrupts
- o enabling the interrupts
- o signalling the processor that an interrupt condition has occurred.

Interrupt flags are set by interrupting conditions which exist within the chip, or on inputs to the chip. These flags normally remain set until the



interrupt has been serviced by the processor. To determine the source of an interrupt, the processor must examine these flags in order from highest to lowest priority.

Procedure:

1. Read the flag register into the accumulator.
2. Shift it right or left.
3. Use conditional branch instructions to detect an active interrupt.

Associated with each flag bit is an enable bit. This can be set or cleared by the processor to enable or disable the flag respectively. If a flag bit is enabled and set, it will cause the IRQ* output to go low, thus sending a direct request to the processor. In addition, bit 7 of the flag register is set to allow quick determination of which chip contains an interrupt condition.

The IFR may be read directly by the processor. In addition, individual flag bits may be cleared by writing a "1" into the appropriate bit of the IFR. When the proper chip select and register controls are applied to the chip, the contents of the IFR are placed on the Data Bus. Bit 7 indicates the condition of the IRQ output, however it cannot be directly cleared; all other bits must be cleared in order for this bit to become inactive.

For each interrupt flag in the IFR there is a corresponding bit in the Interrupt Enable Register. This is accomplished by writing to the IER. There are two steps to consider: the enabling write cycle and the disabling write cycle. The cycle is determined by the logic state of bit 7:

- o If bit 7 and the corresponding 0-6 bits are at "1", the 0-6 bits enable their matching flag.
- o If bit 7 is a 0, then the 0-6 bits containing a "1" disable the matching flag.

Refer to the figures to see the data byte configuration and device.



6522 VIA Environment and Control

The description so far has been very general and has not completely detailed the functions of the two VIAs within the Apple ///. However, one must be familiar with the many registers and modes to understand how the VIA is utilized and how it can be configured for various operations.

The following figures and tables detail the environment and address locations of each of the registers and I/O pins for the two chips. (Many of the signals are common controls or busses; their descriptions relate to both devices.)

IDO-ID7

The data bus pins of the VIA are connected to the associated pins of the Internal Data bus. When the chip is properly addressed and selected, each of the various registers may be accessed for reading or writing data, control bytes, status or timer states. It is this bus through which all data passes to the system and/or processor.

Reset

The RES pin is connected to the RESET line. The system generates the reset at Power On or when the Reset switch is depressed with the Control key.

(Note: the Control key may be software disabled.)

I R/W*

The R/W* pin is connected directly to the processor's internal Read/Write line. This signal cannot be accessed from the outside world. It is obviously used to control the direction of the data to or from the device.

PRE1M

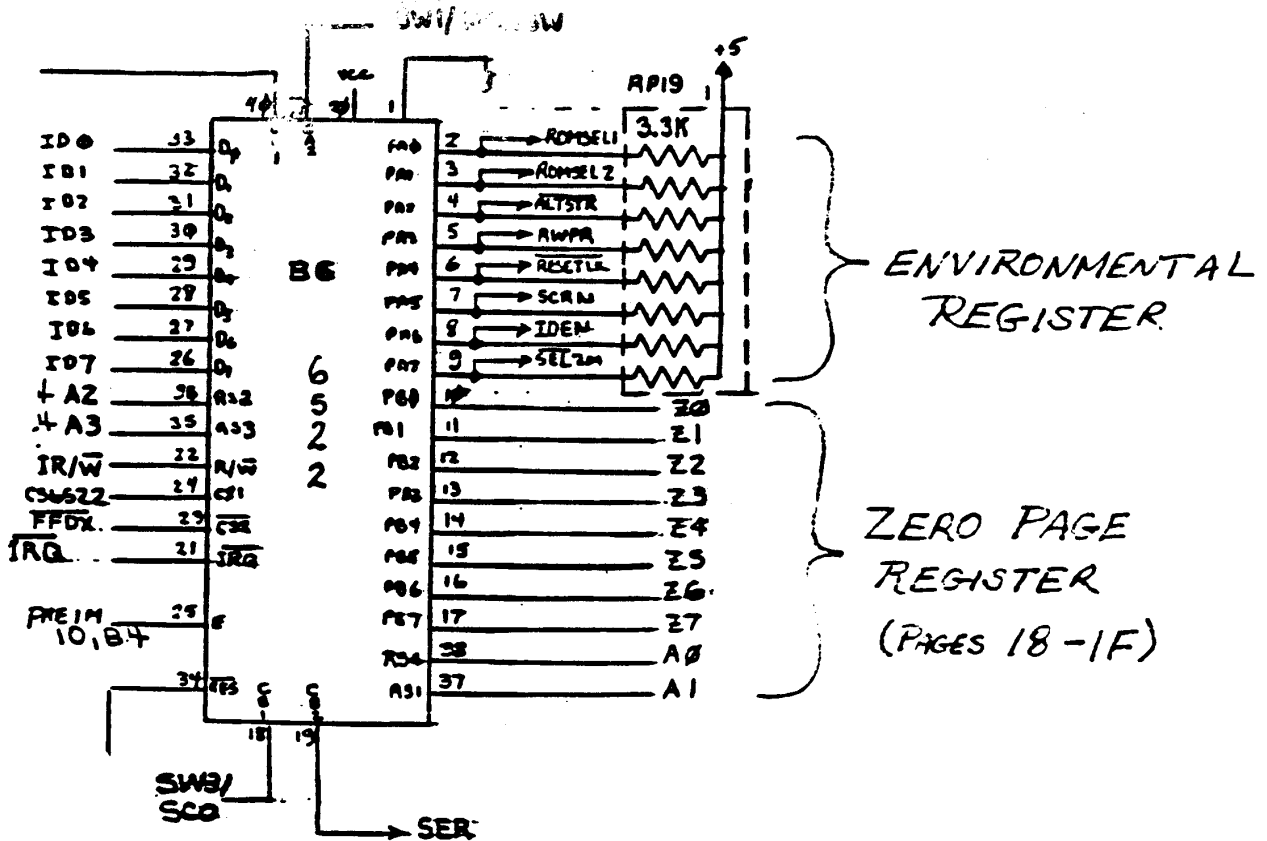
The O2 clock pin is driven by the PRE1M signal developed in the system timing circuits. It is used within the device to clock the various functions of the registers and timers.

CS6522

This signal is developed under Rom decode of various address states or ranges and other mode selections, and drives the CS1 line of both VIAs.

A0-A3

The processor's address lines directly drive the Register Select input lines.



ADDRESS	Register Number	RS Coding				Register Desig.	Description	
		RS3	RS2	RS1	RS0		Write	Read
FFD0	0	0	0	0	0	ORB/IRB	Output Register "B"	Input Register "B"
FFD1	1	0	0	0	1	ORA/IRA	Output Register "A"	Input Register "A"
FFD2	2	0	0	1	0	DDRB	Data Direction Register "B"	
FFD3	3	0	0	1	1	DDRA	Data Direction Register "A"	
FFD4	4	0	1	0	0	T1C-L	T1 Low-Order Latches	T1 Low-Order Counter
FFD5	5	0	1	0	1	T1C-H	T1 High-Order Counter	
FFD6	6	0	1	1	0	T1L-L	T1 Low-Order Latches	
FFD7	7	0	1	1	1	T1L-H	T1 High-Order Latches	
FFD8	8	1	0	0	0	T2C-L	T2 Low-Order Latches	T2 Low-Order Counter
FFD9	9	1	0	0	1	T2C-H	T2 High-Order Counter	
FFDA	A	1	0	1	0	SR	Shift Register	
FFDB	B	1	0	1	1	ACR	Auxiliary Control Register	
FFDC	C	1	1	0	0	PCR	Peripheral Control Register	
FFDD	D	1	1	0	1	IFR	Interrupt Flag Register	
FFDE	E	1	1	1	0	IER	Interrupt Enable Register	
FFDF	F	1	1	1	1	ORA/IRA	Same as Reg 1 Except No "Handshake"	

SY522 Internal Register Summary



These four addresses, along with the two chip select lines, activate the chip for access to and from the data bus. You can see that the decimal equivalent of the address (Register Select) lines equals the register number, as detailed in the port address tables.

To determine the complete I/O location:

- o Express the A0-A7 bits in Hex
- o Substitute the Hex expression for the "x" in the signal name applied to the CS2 input.

IRQ*

The Interrupt Request line is a wire OR'ED bus, shared with the other LSI devices within the system which directly interrupt the processor. The VIA can generate an interrupt due to internal status and can be configured to interrupt on various input conditions. Each VIA provides indirect interrupts to the processor for the other devices in the system, including the slot interrupts. By polling the VIAs, the processor can quickly determine which device or group of devices needs servicing.

VIA (FFDX)

This VIA has the following responsibilities:

- o Environmental Register
- o Zero Page Register
- o Global Interrupt Request
- o Serial Data Port Interrupt & Clocking (Silentype)

The signals of the IC at location B6 which are not common to both 6522s are either the port I/Os or the chip select line, CS2.

FFDX

This signal comes from the Device Select logic, and is true for the 16 address states, FFDO through FFDF.

The table below shows how the signal FFDX (CS2) and the Address Lines on the Register Select Inputs access each of the 16 registers within the VIA.



Port A Description

The ORA for this device is also called the "Environmental Register" by the system. It is programmed to all outputs, and contains various control mode states which may be software modified.

PA0 and PA1 are used as software switches to control which set of Roms are to be used by the system, and also to indicate there is an expanded set of Rom in the system.

PA2 provides the software switch that enables the Apple /// to switch between two memory stacks.

PA3 is a software switch which will enable or disable the system for writing into the Ram. It can be set for entire banks or it can be set at will to disallow writing into memory.

PA4 is a software switch which will cause the function of the Reset switch to be ignored, and will simultaneously disable the Non-Maskable Interrupts from the I/O slots.

PA5 contains the switch "Scrn" (Screen) which is used to modify the Blanking signal.

Global IRQ

The CA1 line is connected to "OR" function of the slot IRQs. If any slot is requesting an interrupt, this line will toggle. The VIA then generates an IRQ and sets the associated Interrupt flag.

Sw1/Mgnsw

This line connected to the CA2 line can be programmed to cause an interrupt on either edge of transition. It could be used to have a Function Only run while the switch is depressed, or vice versa.

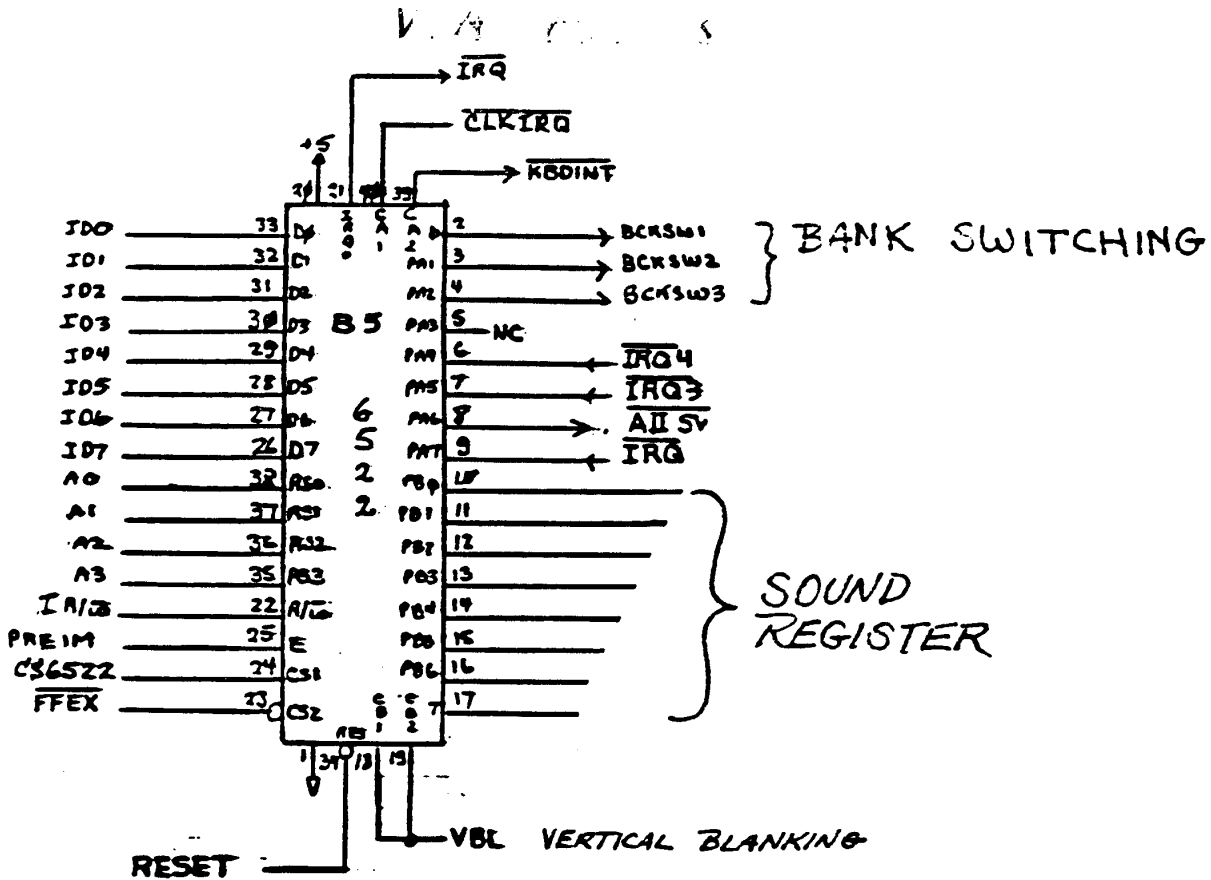
SCO/SER

These two signals form a very elementary serial data port. They are usually programmed as data, and strobe to a serial RO printer.

The port may also be configured as a serial input register or simply as interrupts for the two external lines.

SCO/SER are connected to CB1 and CB2 respectively.

Z0-Z7 (Zero Page Register)



ADDRESS	Register Number	RS Coding				Register Desig.	Description	
		RS3	RS2	RS1	RS0		Write	Read
FFE0	0	0	0	0	0	ORB/IRB	Output Register "B"	Input Register "B"
FFE1	1	0	0	0	1	ORA/IRA	Output Register "A"	Input Register "A"
FFE2	2	0	0	1	0	DDRB	Data Direction Register "B"	
FFE3	3	0	0	1	1	DDRA	Data Direction Register "A"	
FFE4	4	0	1	0	0	T1C-L	T1 Low-Order Latches	T1 Low-Order Counter
FFE5	5	0	1	0	1	T1C-H	T1 High-Order Counter	
FFE6	6	0	1	1	0	T1L-L	T1 Low-Order Latches	
FFE7	7	0	1	1	1	T1L-H	T1 High-Order Latches	
FFE8	8	1	0	0	0	T2C-L	T2 Low-Order Latches	T2 Low-Order Counter
FFE9	9	1	0	0	1	T2C-H	T2 High-Order Counter	
FFEA	10	1	0	1	0	SR	Shift Register	
FFEB	11	1	0	1	1	ACR	Auxiliary Control Register	
FFEC	12	1	1	0	0	PCR	Peripheral Control Register	
FFED	13	1	1	0	1	IFR	Interrupt Flag Register	
FFEE	14	1	1	1	0	IER	Interrupt Enable Register	
FFEF	15	1	1	1	1	ORA/IRA	Same as Reg 1 Except No "Handshake"	



The PB port is referred to as the Zero Page Register. In other words, its primary function is to store the current "Zero Page" of the memory. The Apple /// is capable of moving the Zero Page, a feature which greatly enhances the flexibility of the system.

The PB port has a secondary function; it also serves as the address register for the Real Time Clock. The RTC must have the addresses stable for an extended period of time, so instead of adding another latch in the hardware, the designers use this port for control.

Note that each time the port is used for the RTC function it must be restored to the current Zero Page setting.

VIA (FFEX)

This VIA aids in providing the following:

- o Bank Switching
- o Sound Register
- o Interrupt Recognition-clock, keyboard

The signals connected to the VIA located at B5 which are not common to both VIAs are:

- o the A and B port I/Os,
- o the port control lines, and
- o the CS2 line.

FFEX

This signal corresponds to FFDX in that it is true for a group of 16 addresses. See the corresponding table for the complete detail of the register access locations.

Port A Description

The PA0-PA2 lines are configured as outputs and contain the software switches for the Ram banks. Currently, the system segregates the 128K memory into three banks. The decimal decode of the binary bit weight reveals the bank.

The PA4-PA7 lines are configured as inputs. Slot 1 and 2 IRQs, the Solid Apple switch, and the IRQ line itself form the respective inputs.

If it's interrupted, the processor will poll the VIAs first to get a quick look at most of the system. It can then identify and service the requesting device faster, since it doesn't need to poll each individual device.



The processor can, at certain times, read this port for the status of the special Apple switch on the keyboard without disturbing the keyboard circuit.

The IRQ* signal is wrapped around to the PA7 line for special diagnostic purposes.

CLK IRQ*

The Real Time Clock's interrupt is connected to the CA1 line, which is programmed to be a negative edge active input. When the clock generates an interrupt, it will set the IRA flag in the IFR. The PA port is conditioned for non-latching, however, resulting in a basically independent interrupt for the clock.

Keyboard Interrupt

The keyboard's interrupt is connected to the CA2 input, which is programmed to be an independent negative edge interrupt. It will set Bit 0 in the IFR and cause the IRQ* line to go low.

Note: The keyboard can, for the most part, be disabled by disabling the interrupt flag for the CA2 line.

VBL (Vertical Blanking)

This input can perform two functions, depending on how the CB1, CB2, and Shift Register are programmed.

- o The system may want to be interrupted at each vertical blanking cycle. If so, you would program the CB2 line to be an independent interrupt OR let it strobe the IPB and set the corresponding bit flag.
- o The system may want to synchronize an operation to the display, but may not want to be interrupted at each VBL. If this is the case, the system can configure the Shift Register to count 8 occurrences of the VBL signal. An interrupt will then occur after each set of 8 Vertical Blanking cycles (about once every second), in sync with the display scan.

PB Port Description

The first 6 lines of the B port are configured to be outputs. They are inputs to the sound Generator.

- o The tone generated at the speaker can be varied by changing the bit values of these lines.
- o There are 127 possible tone combinations; the missing one turns the tone off completely.

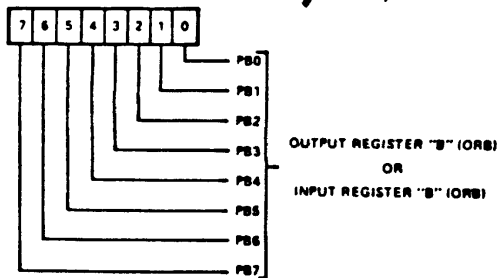


PB6 is connected to the I/O Count line. Depending on the device in the slots, the VIA may be programmed to count a certain number of pulses generated or to determine that only one pulse occurred. Either way, the VIA will generate an IRQ and set the appropriate bit flag.

The last bit is used to monitor the NMI (Non Maskable Interrupt) line generated by the devices in the I/O slots.

VIA APPENDICES

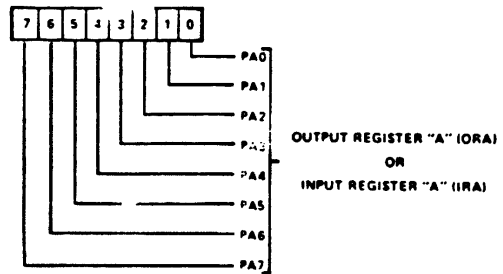
REG 0 – ORB/IRB **FFD0/E0**



Pin Data Direction Selection	WRITE	READ
DDRB = "1" (OUTPUT)	MPU writes Output Level (ORB)	MPU reads output register bit in ORB. Pin level has no effect.
DDRB = "0" (INPUT) (Input latching disabled)	MPU writes into ORB, but no effect on pin level, until DDRB changed.	MPU reads input level on PB pin.
DDRB = "0" (INPUT) (Input latching enabled)		MPU reads IRB bit, which is the level of the PB pin at the time of the last CB1 active transition.

Figure 9. Output Register B (ORB), Input Register B (IRB)

REG 1 – ORA/IRA **FFE1/D1**



Pin Data Direction Selection	WRITE	READ
DDRA = "1" (OUTPUT) (Input latching disabled)	MPU writes Output Level (ORA)	MPU reads level on PA pin.
DDRA = "1" (OUTPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition.
DDRA = "0" (INPUT) (Input latching disabled)	MPU writes into ORA, but no effect on pin level, until DDRA changed.	MPU reads level on PA pin.
DDRA = "0" (INPUT) (Input latching enabled)		MPU reads IRA bit which is the level of the PA pin at the time of the last CA1 active transition.

Figure 10. Output Register A (ORA), Input Register A (IRA)

REG 12 – PERIPHERAL CONTROL REGISTER **FFDC/EC**

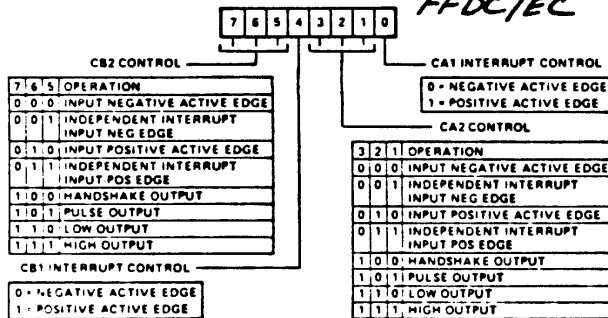
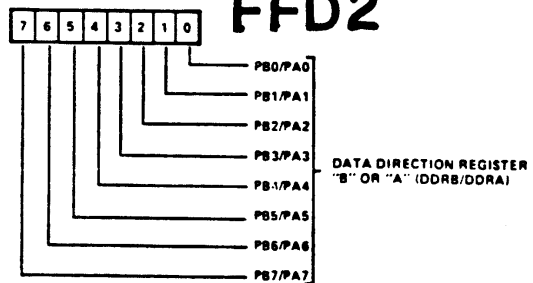


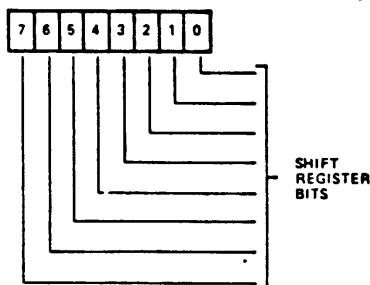
Figure 14. CA1, CA2, CB1, CB2 Control

REG 2 (DDRB) AND REG 3 (DDRA) **FFD2**



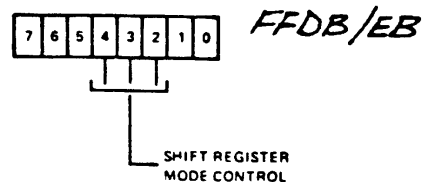
"0" ASSOCIATED PB/PA PIN IS AN INPUT (HIGH IMPEDANCE)
 "1" ASSOCIATED PB/PA PIN IS AN OUTPUT, WHOSE LEVEL IS DETERMINED BY ORB/OR A REGISTER BIT.

REG 10 – SHIFT REGISTER **FFEA/DA**



NOTES:
 1. WHEN SHIFTING OUT, BIT 7 IS THE FIRST BIT OUT AND SIMULTANEOUSLY IS ROTATED BACK INTO BIT 0.
 2. WHEN SHIFTING IN, BITS INITIALLY ENTER BIT 0 AND ARE SHIFTED TOWARDS BIT 7.

REG 11 – AUXILIARY CONTROL REGISTER **FFDB/EB**



4	3	2	OPERATION
0	0	0	DISABLED
0	0	1	SHIFT IN UNDER CONTROL OF T2
0	1	0	SHIFT IN UNDER CONTROL OF $\cdot\frac{1}{2}$
0	1	1	SHIFT IN UNDER CONTROL OF EXT CLK
1	0	0	SHIFT OUT FREE RUNNING AT T2 RATE
1	0	1	SHIFT OUT UNDER CONTROL OF T2
1	1	0	SHIFT OUT UNDER CONTROL OF $\cdot\frac{1}{2}$
1	1	1	SHIFT OUT UNDER CONTROL OF EXT CLK

Figure 22. SR and ACR Control Bits

Two bits are provided in the Auxiliary Control Register (bits 6 and 7) to allow selection of the T1 operating modes. The four possible modes are depicted in Figure 17.

ating modes. The four possible modes are depicted in Figure 17.

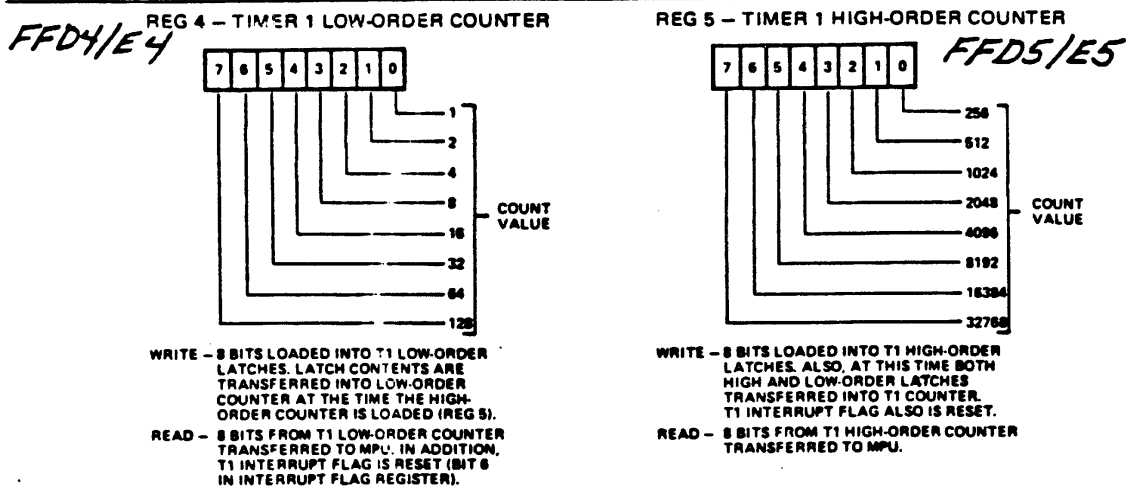


Figure 15. T1 Counter Registers

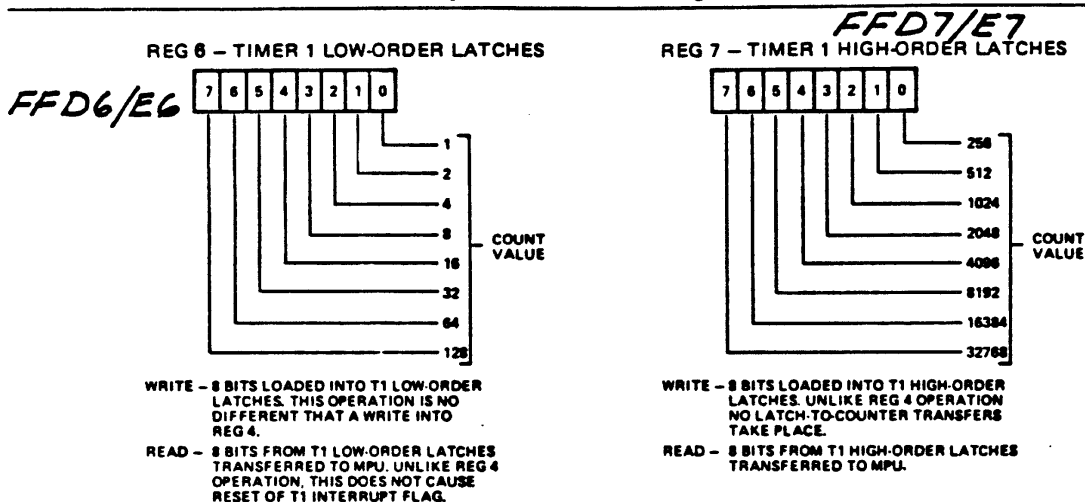


Figure 16. T1 Latch Registers

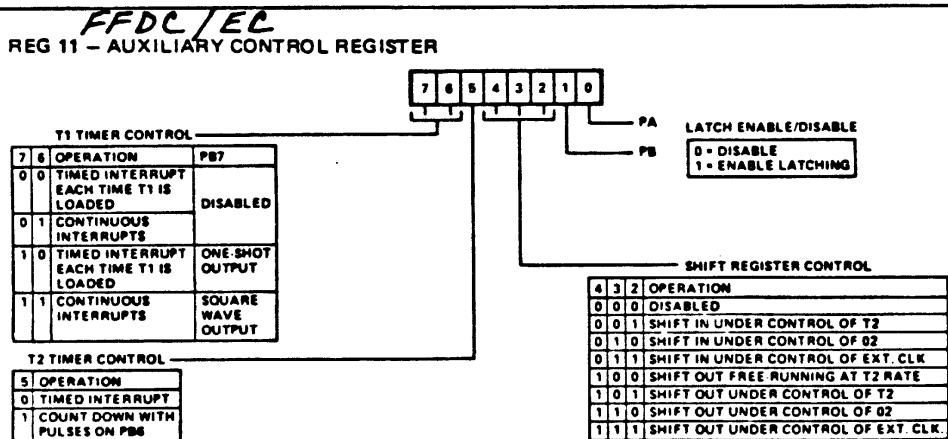
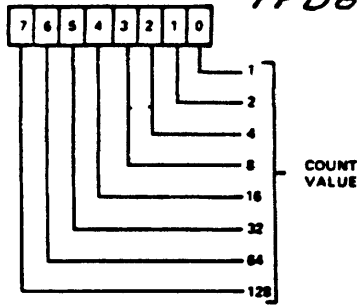


Figure 17. Auxiliary Control Register

Note: The processor does not write directly into the low order counter (T1C-L). Instead, this half of the counter is loaded automatically from the low order latch when the processor writes into the high order counter. In fact, it may not be necessary to write to the low order counter in some applications since the timing operation is triggered by writing to the high order counter.

REG 8 – TIMER 2 LOW-ORDER COUNTER

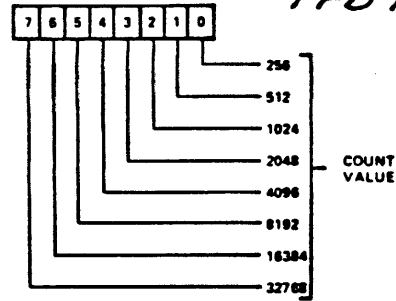
FFDB/EB



WRITE – 8 BITS LOADED INTO T2 LOW-ORDER LATCHES.
 READ – 8 BITS FROM T2 LOW-ORDER COUNTER TRANSFERRED TO MPU. T2 INTERRUPT FLAG IS RESET.

REG 9 – TIMER 2 HIGH-ORDER COUNTER

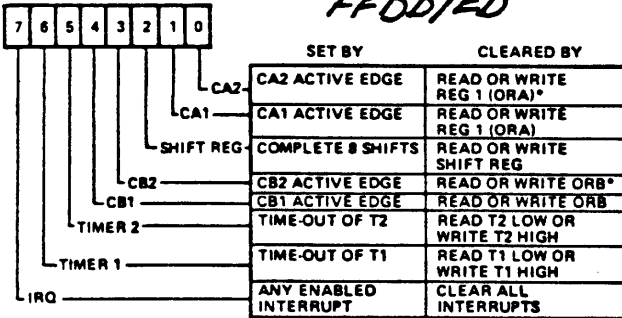
FFD9/E9



WRITE – 8 BITS LOADED INTO T2 HIGH-ORDER COUNTER ALSO, LOW-ORDER LATCHES TRANSFERRED TO LOW-ORDER COUNTER IN ADDITION, T2 INTERRUPT FLAG IS RESET.
 READ – 8 BITS FROM T2 HIGH-ORDER COUNTER TRANSFERRED TO MPU.

REG 13 – INTERRUPT FLAG REGISTER

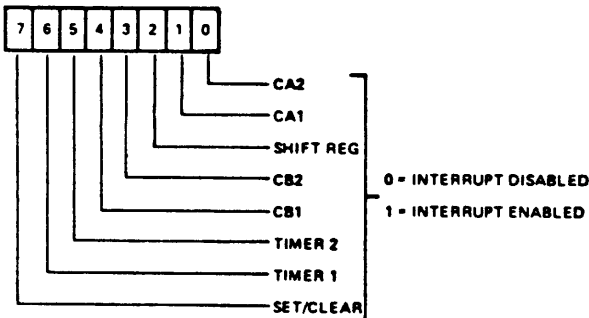
FFDD/ED



* IF THE CA2/CB2 CONTROL IN THE PCR IS SELECTED AS "INDEPENDENT" INTERRUPT INPUT, THEN READING OR WRITING THE OUTPUT REGISTER ORA/ORB WILL NOT CLEAR THE FLAG BIT. INSTEAD, THE BIT MUST BE CLEARED BY WRITING INTO THE IFR, AS DESCRIBED PREVIOUSLY.

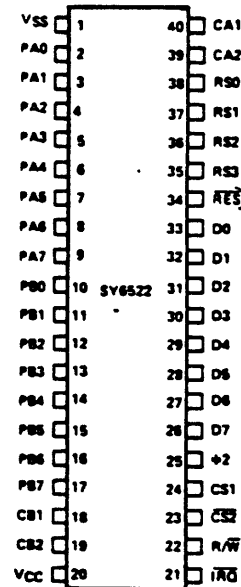
Figure 25. Interrupt Flag Register (IFR)

REG 14 – INTERRUPT ENABLE REGISTER



NOTES:
 1. IF BIT 7 IS A "0", THEN EACH "1" IN BITS 0 - 6 DISABLES THE CORRESPONDING INTERRUPT.
 2. IF BIT 7 IS A "1", THEN EACH "1" IN BITS 0 - 6 ENABLES THE CORRESPONDING INTERRUPT.
 3. IF A READ OF THIS REGISTER IS DONE, BIT 7 WILL BE "0" AND ALL OTHER BITS WILL REFLECT THEIR ENABLE/DISABLE STATE.

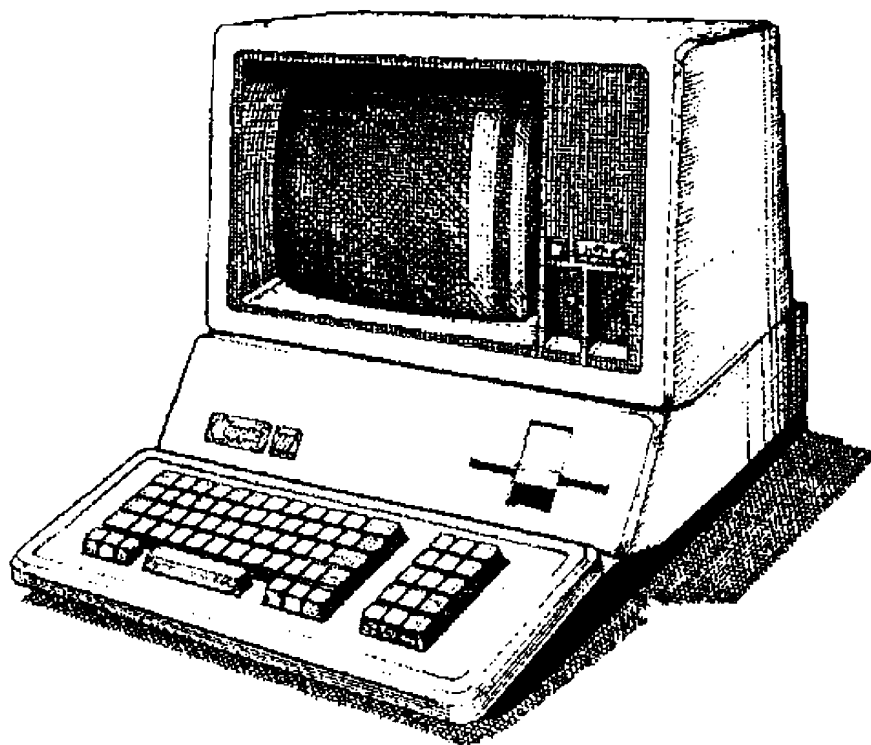
PIN CONFIGURATION





Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

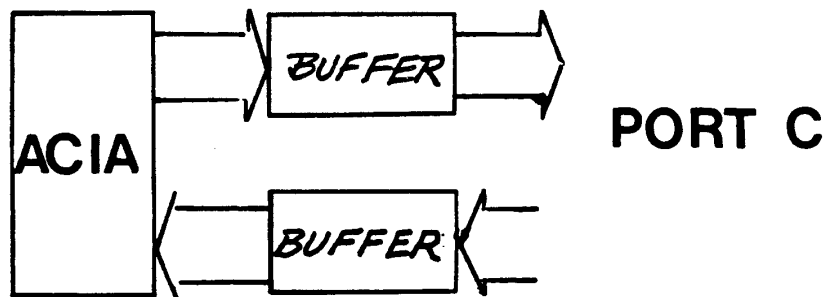
Chapter 4 • The ACIA

Written by Apple Computer • 1982



THE 6551 ASYNCHRONOUS COMMUNICATIONS INTERFACE ADAPTER

As you know, the Apple /// can be used to communicate to all devices that use the RS-232-C standard communications format. This means the Apple /// can communicate with letter-quality printers, modems, high speed data collection devices, and other computers. The Apple /// has a built in Asynchronous Communications Interface Adapter (ACIA). It is located at addresses COF0 through COF3. This device is solely for use as a serial port input/output controller. This RS-232-C protocol, specified by the Electrical Industries Association (EIA), is provided at port C connector through two buffer devices as shown below.



The 6551 contains seven registers and five control circuits dealing with the control, timing, and interrupt logic of transmitting and receiving data through the serial EIA port. A block diagram of the 6551 ACIA is shown in Figure 1.

Before using the 6551, the system (or programmer) must initialize it for the I/O mode. Once initialized it will be set to transmit and/or receive data within the parameters set in the control bytes.

The Apple /// I/O addresses for data and control are presented in the following discussions. The function of the data byte (depending on location) will also be shown. The characters in parentheses next to each register represent the function of the associated pin of the ACIA (i.e. R=Read, W=Write).

RECEIVE DATA REGISTER (RxD)

The Receive Data Register is accessed with I/O location COF0 (r). The contents of the Receive Data Register will be the data bits of the completely received serial input character. This register is used as temporary data storage for the 6551 receive circuit. The first data bit received will be the LSB of the data byte (Bit 0).

COF0 (R) RECEIVE DATA REGISTER

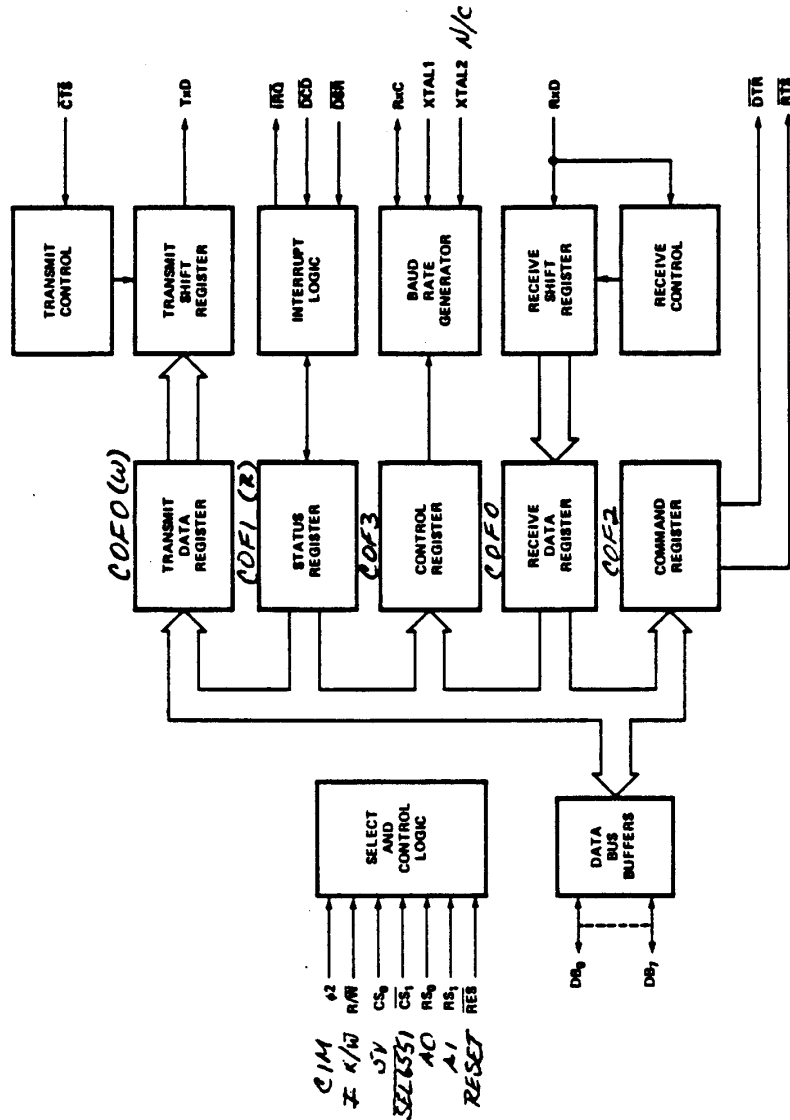


FIG 1

THE ACIA BLOCK DIAGRAM



TRANSMIT DATA REGISTER (TxD)

To load a byte to be transmitted out the serial EIA port, I/O location COFO (w) must be accessed. The LSB will be the first data bit transmitted.

COFO (w) TRANSMIT DATA REGISTER (w)

STATUS REGISTER

The 6551 continually monitors the condition of the registers and the quality of the incoming data. When I/O location COF1 (R) is accessed the contents of the Status Register is placed on the data bus. The meanings of the eight bits of the status byte are detailed following the illustration.

COF1 (R) STATUS REGISTER

IRQ--Bit 7 indicates that the 6551 generated an IRQ to the system for one of the following conditions:

1. Change of status of the DCD line.
- 2.. Change of status of the DSR line.
3. The Transmit Register has emptied.
4. The Receive Register has filled with a new incoming character.

(Note: the third and fourth conditions are program controlled as to whether or not they initiate an IRQ).

DSR---Bit 6 indicated the status of the DSR line from the interface.
0=DSR is low and ready. 1=DSR is high and not ready.

DCD---Bit 5 shows the condition of the Data Carrier Detect line from the interface. 0=DCD is low and the carrier is detected. 1=DCD is high and not ready.

TDRO--Bit 4 informs the host that the Transmit Data Register has transferred its contents to the outputs shift register and now ready to accept another character (byte). 0=not empty, 1=empty.

RDRF1-Bit 3 indicates that the Received Data Register has been filled with a character from the line. 0=not full, 1=full.

OVERR--Bit 2 shows that there has been an overrun error, that is, a new character has been transferred to the Received Data Register before the previous received character was taken by the program and the RDRF1 flag reset. This error will not generate an IRQ but should be checked by the program so that the lost data can be recovered. 0=no overrun, 1=overrun has occurred.



FRMERR--Bit 1 informs the program that the incoming data did not conform to the parameters set in the Control Register. That means usually that when the 6551 checked for the position and number of stop bits if found a discrepancy. This error most often happens when the remote device is transmitting at a different baud rate. 0=no framing error, 1=framing error detected. This error does not generate an IRQ.

PARERR--Bit 0 indicates that there has been a parity error in the incoming data. This error does not cause an IRQ to be generated. 0=no parity error, 1=parity error detected.

PROGRAMMED RESET

By accessing COF1 (w) the Status Register will be reset to all 0's. The data byte on the bus at that cycle does not have to be any particular structure.

COF1 (w) PROGRAMMED RESET (w)

COMMAND REGISTER

To access the Command Register to initialize or modify the Command byte a COF2 (w) must be executed with the data byte configured for the desired effect. To inspect the contents (or current command structure) a COF2 (R) will cause the 6551 to place the contents of the Command Register on the data bus. The meaning of each bit is explained below.

COF2 (r/w) COMMAND REGISTER (r/w)

Parity ck ctrls--Bits 7 through 5 command the 6551 in regard to parity checking. Bit 5 is parity enable. Bit 7 determines whether the parity bit position will have odd/even or fixed one/zero function. Bit 6, depending on the condition of Bit 7 selects either odd or even parity, or fixed mark (one) or fixed zero (space). Table -- below clearly shows the conditions of these bits.

ECHO--Bit 4 determines whether the 6551 will echo (transmit a duplicate image of what is received) the received data to the remote device. 0=no echo, 1=echo data.

TRANS CTRLS--Bits 3 and 2 control the 6551 in three (3) functions. They are Transmit Interrupt, the state of the RTS line, and the Transmit BRK (break, a continuous space on the line for approximately 200 milliseconds). Table -- shows which state controls which function.

INT--Bit 1 command the 6551 to either enable or disable interrupt on Received Data Register full (bit 3 of the status byte). 0=IRQ on RDRF1, 1=no interrupt.



DTR--Bit 0 enable the 6551 to transmit and receive or not. It also changes the state of the DTR output to match the condition of the 6551. 0=disable xmit/rcv (DTR high), 1=enable Xmit/Rcv (DTR low).

CONTROL REGISTER

The final I/O location on the 6551 is the Control Register. By writing to CPF3 (w) with a properly structured data byte four functions are controlled. They are Number of Stop bits appended to outgoing data byte (and checked for on the incoming data byte), the source of the receiver clock, and the baud rate selection. By reading location COF3 (x) the current control configuration can be seen on the data bus.

COF3 (R/W) CONTROL REGISTER (R/W)

STOPB--Bit 7 controls the number of stop bits that will be added to the transmitted data word. 0=1stop bit, and depending on the word length selection 1=2 stop bits or 1 bit if word length is 8 and parity is selected, or 1/2 bits if word length is 5 and no parity is selected.

WORD LEN--Bits 6 and 5 determine the number of data bits that will be transmitted or received in the data word. The values (6,5) are as follows: 0,1=7 bits; 1,0=6 bits; 1,1=5 bits.

'XCLK--Bit 4 indicates the source of the receiver clock. 0=external clock source, 1=baud rate generator (internal). When the internal selection is made the RxC pin becomes an output.

BAUD RATE--Bits 3 through 0 select the baud rate at which the 6551 will operate. Table -- details the various selections available based on a standard 1.8432 MHZ clock input. It should be noted that 0,0,0,0 selects the external clock as the baud rate source but the clock rate is actually divided by 16 so the external clock should be 16x the desired baud rate.



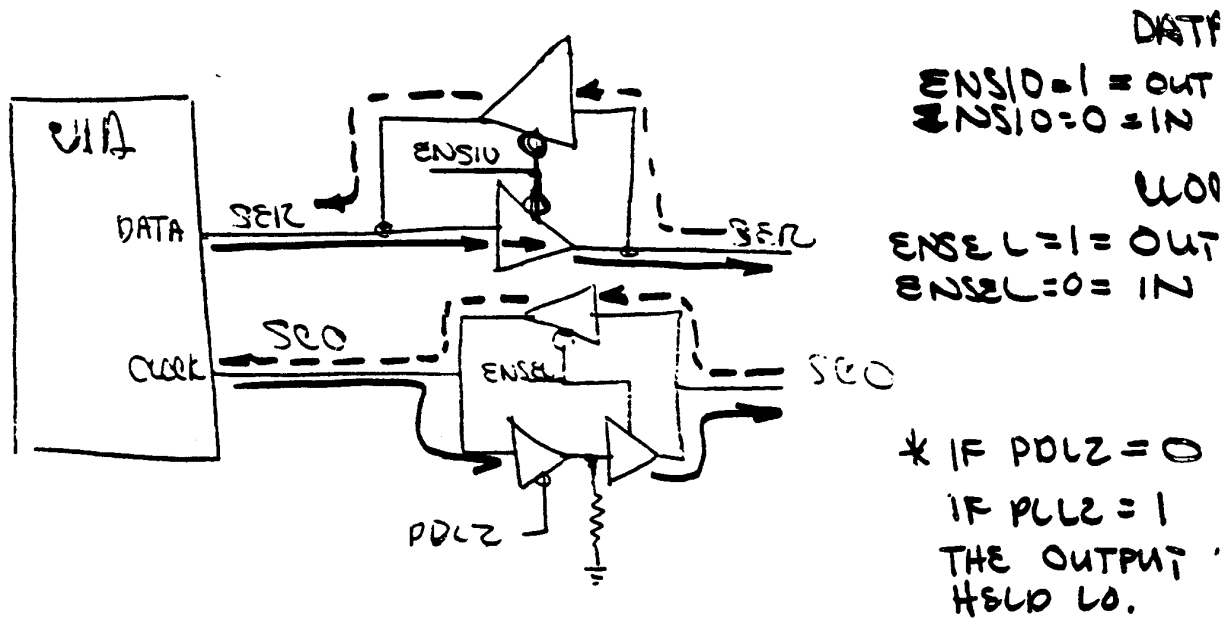
SIMPLE SERIAL PORT

Incorporated into the A /// logic design is a provision for a simple serial port. Its main purpose is to support a Silentype printer. However, it is not a dedicated port since it shares the same connector with one of the joysticks. The user has the option of having the Silentype or a second joystick. For most applications, one joystick is more than enough since the A /// is not really intended for a "games" player. The joystick would be used more for applications like cursor control.

The serial port is actually derived from a feature of the VIA. Two of the port control lines (PB1, PB2) can be configured to be a shift register, hence a simple serial port (see the section on VIA's and read the spec sheet in the appendices). In the system certain other software controlled switches must be set to enable the port. These switches are found in another addressable latch (U177). They control the direction of data and clock to and from the board. The VIA is programmed internally to determine its function.

The signal ENSIO when high enables data from the VIA to the port, when low it enables data from the port to the VIA. The signal ENSEL determines the direction of the clock, high enables clock from the VIA to the port (if and only if PDL2 is low), low enables the port to supply the clock to the VIA. ENSEL when high also enables the AXCO signal to the port, this can be used as a select or acknowledge to the remote device. The remote device may put status bits on the "switch" line which can be used to notify the processor of some requirement mutually agreed upon.

SIMPLE SERIAL PORT -



NOTE: THE VIA MAY SEND DATA USING IT'S OWN CLOCK OR IT MAY BE SHIFTS WITH A REMOTE CLOCK.
 ALSO IT MAY STROBE DATA IN WITH IT'S OWN CLOCK OR IT MAY BE STROBEN IN WITH A REMOTE CLOCK



PINOUT OF THE RS-232-C SERIAL INTERFACE (PORT C)

<u>PIN</u>	<u>NAME</u>	<u>DESCRIPTION</u>
1	SGND	Shield GrouND.
2	TXD	Transmitted Data; serial data output from the Apple.
3	RCD	ReCieved Data; serial input to the Apple.
4	RTS	Request To Send output; this indicates that the Apple is ready to transmit data. This line is active whenever the Serial Card emulation is used.
5	CTS	Clear To Send input; this acknowledges that the Apple may begin transmission. This line is ignored by the Serial Card emulation.
6	DSR	Data Set Ready input; this acknowledges that the remote device is operational. The Serial Card emulation checks this line and will not send characters if this line is held inactive. This can be used to prevent the Apple from overflowing a printer input buffer.
7	GND	Signal GrouND.
8	DCD	Data Carrier Detect; this acknowledges that the remote device is ready to transmit data. The Serial Card emulation checks this line and will not send characters if this line is held inactive. This line can be used to prevent the Apple from overflowing a printer input buffer.
9-19		No connect.
20	DTR	Data Terminal Ready output; this indicates that the Apple is on and operational. This line will be active anytime the Serial Card emulation is used.



THE ACIA

The Apple /// has a built-in 6551 ACIA (Asynchronous Communication Interface Adapter). It is located at addresses \$COF0 thru \$COF3 (decimal -16144 to -16141). The ACIA has five registers: transmit data, receive data, status, command, and control.

The transmit register (\$COF0) is used to send data out the Apple /// to an external device, such as a modem or a printer. A byte is transmitted by setting the control and command registers appropriately and then polling the status register. When bit 4 of this register is one the ACIA is ready to shift the next byte out. Often bits 5 and 6 are tested for zero to assure the Data Carrier Detect and Data Set Ready are valid as some printers use these lines as handshake signals.

Care must be taken when writing to the transmit data register as it is at the same address as the receive data register. The 6502 will do false reads when certain address modes are used, thus discarding whatever was in the receive data register.

The receive data register (\$COF0) contains the last byte received from an external source, such as a modem. Bit 3 of the status register is one whenever this register is full.

The status register (\$COF1) indicates the states of Data Set Ready, Data Carrier Detect, whether the transmit and receive registers are full, and whether a framing, overrun or parity error has occurred on input.

The command register (\$COF2) sets the parity, echo mode, transmit and receive enables, and BRK transmission.

The control register (\$COF3) sets the number of stop bits, data word length, receiver clock source, and baud rate.

PHYSICAL PINOUT OF THE RS-232-C SERIAL INTERFACE

13	12	11	10	9	8	7	6	5	4	3	2	1
25	24	23	22	21	20	19	18	17	16	15	14	



RS232 CONNECTOR USAGE (PORT C)

The Apple /// is classified as Data Terminal Equipment (DTE) under the EIA RS-232-C interface specification. It can be directly connected to a piece of Data Communications Equipment (DCE), such as a modem. To connect the Apple to another piece of Data Terminal Equipment (such as a printer), you must use a modem eliminator.

All output levels are minimum +6 volts when logic 0 and maximum -6 volts when logic 1, measured into a 3K ohm load.

All inputs have a turn-on positive going threshold of +1.25 volts and a turn-off negative going threshold of +.8 volts, typical. All inputs sink a 10 mA current, maximum.

CONTROL REGISTER

The Control Register is used to select the desired mode for the SY655. The word length, number of stop bits, and clock controls are all determined by the Control Register, which is depicted in Figure 6.

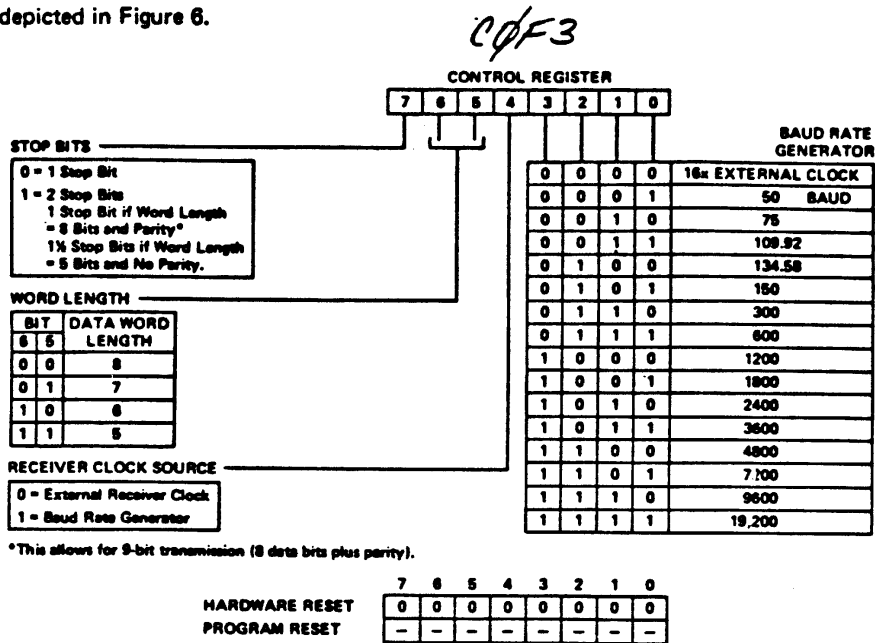


Figure 6. Control Register Format

COMMAND REGISTER

The Command Register is used to control Specific Transmit/Receive functions and is shown in Figure 7.

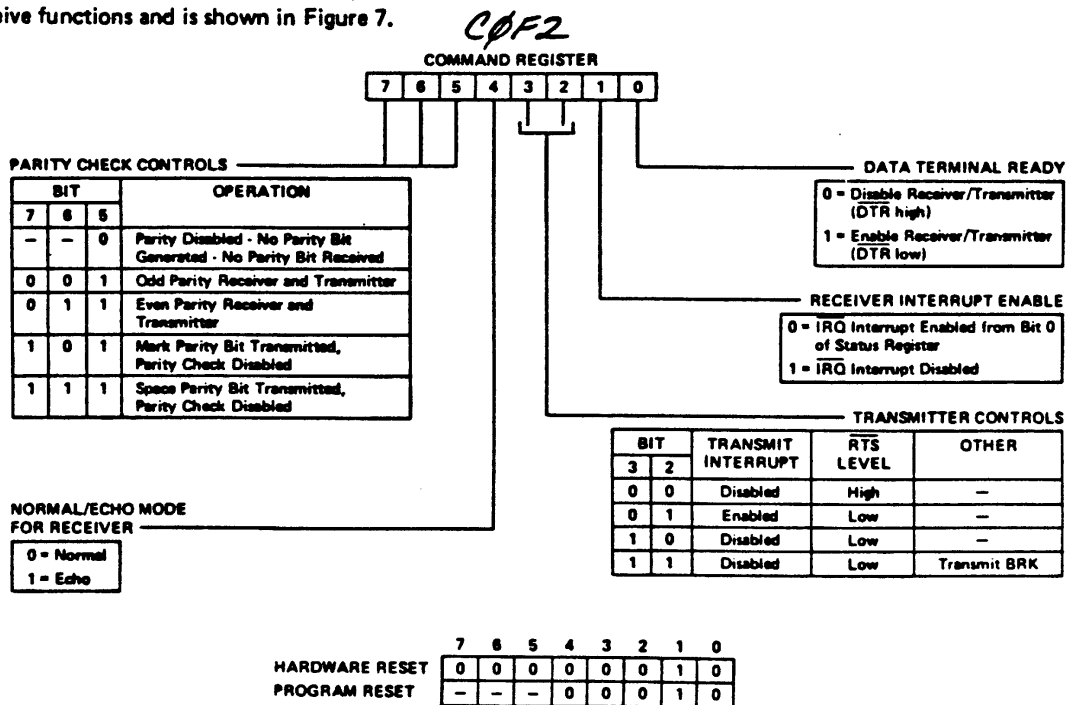
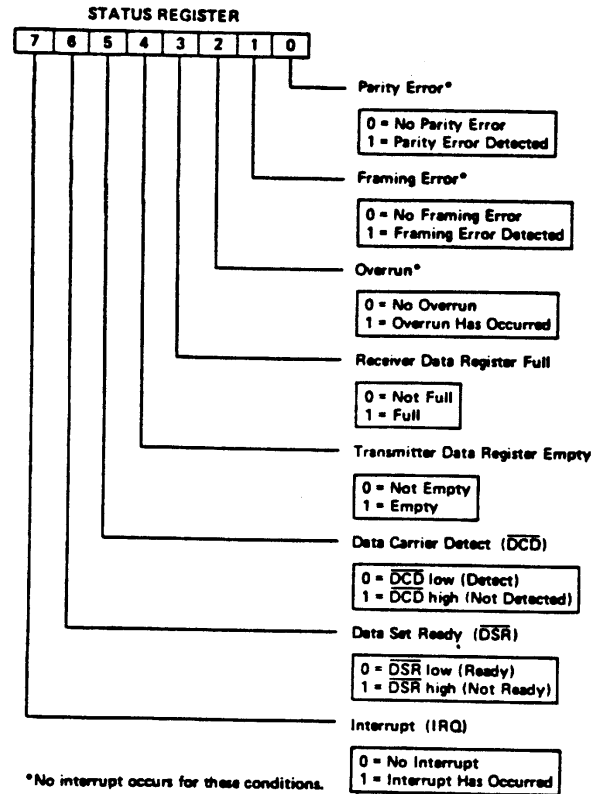


Figure 7. Command Register Format

STATUS REGISTER *COFI*

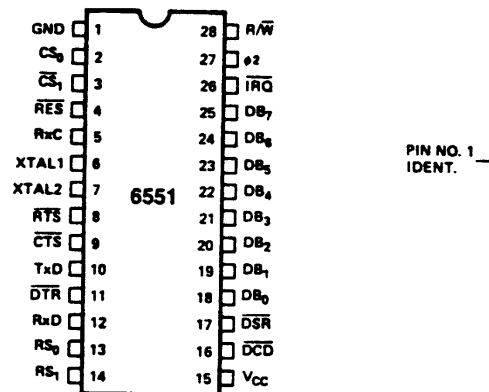
The Status Register is used to indicate to the processor the status of various SY6551 functions and is outlined in Figure 8.



	7	6	5	4	3	2	1	0
HARDWARE RESET	0	-	-	1	0	0	0	0
PROGRAM RESET	-	-	-	-	-	0	-	-

Figure 8. Status Register Format

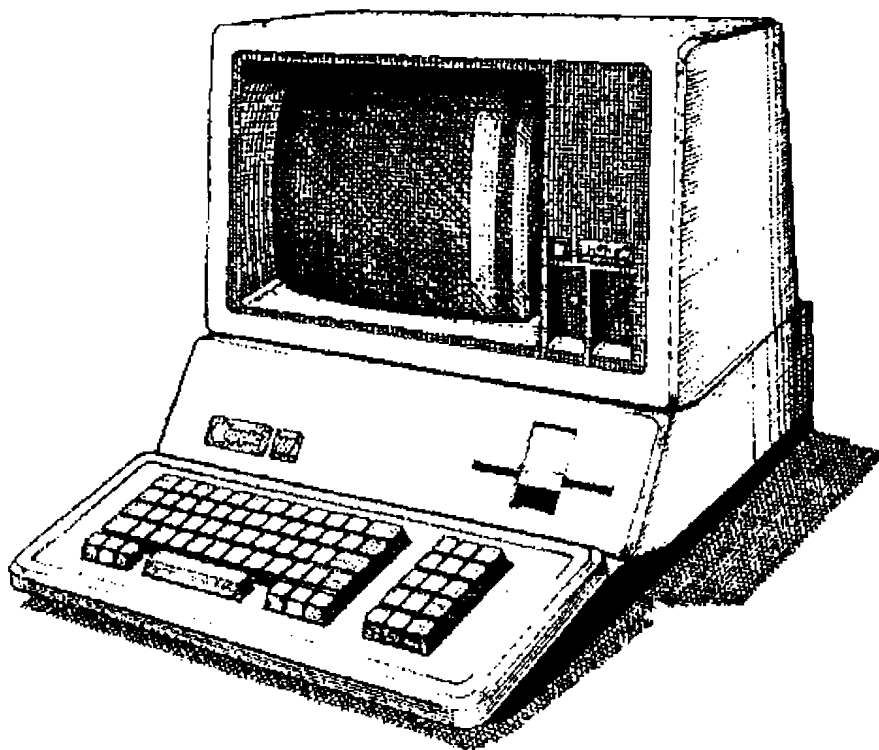
PIN CONFIGURATION





Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 5 • System Clocks & Timing

Written by Apple Computer • 1982



SYSTEM CLOCKS & TIMING

MAIN CLOCK (C14M)

The Apple /// has as its master clock a 14 megahertz crystal controlled oscillator. The active components of the clock circuitry are Q10, Q11, and Y1. The exact frequency of the oscillator is 14.318 MHz. The slight increase over 14 MHz is compensated for in other logic. Device B13 provides buffering and power amplification to drive all the other loads on the C14M and C14M* lines.

FREQUENCY DIVIDER

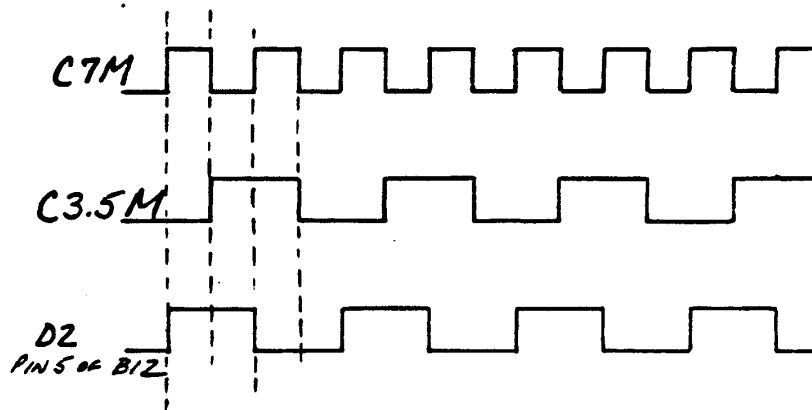
The next circuit in the system clock section is the frequency divider formed by device B12 and B13. This develops both the C7M* and the C3.5M*.

The C7M signal is developed by clocking the Q* output into the data input of a D-type latch. This results in a divide-by-two function of the clock frequency.

The C3.5M clock is developed in the same manner. However, the data input to the latch is an Exclusive-Or function of C7M and C3.5M at B13. This accomplishes two functions:

- o it divides the C14M clock by 4, and
- o it gives a definite phase relationship of C7M to C3.5M clocks.

Looking at the timing diagram below we see that the D2 input of B12 is high if either the C7M or the C3.5 clock is high but not when both are high. This function is effectively at 3.5 MHz which toggles on the positive edge of the 7MHz clock. The true C3.5M signal toggles on the negative edge of the C7M clock.

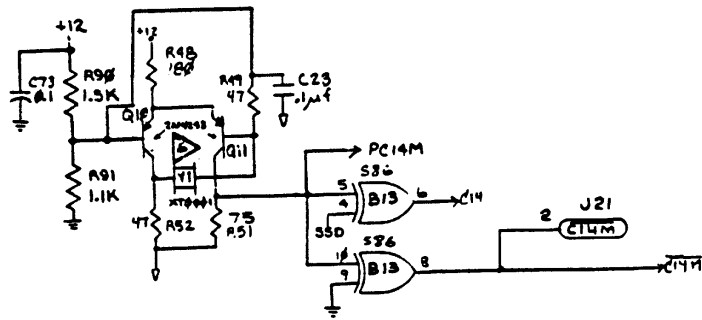


"Q" TIMING

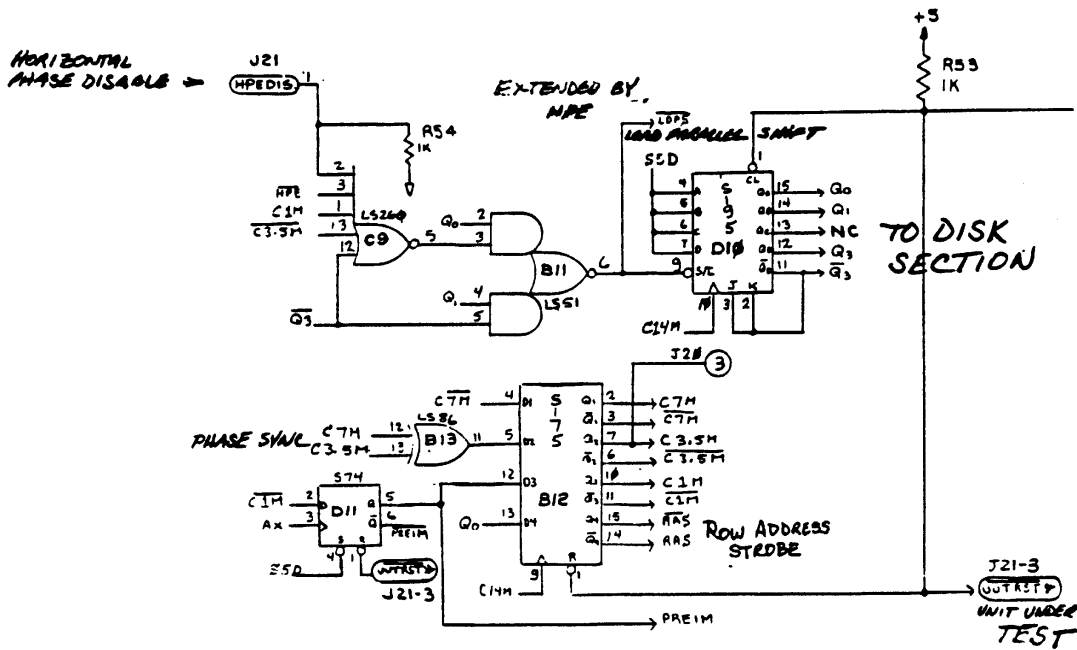
The Q clocks are a series of 2 MHz clocks which are out of phase with one another by one clock time (refer to the Apple /// Timing diagrams). The rest of the system timing depends on the states of the "Q" outputs. They provide the basis

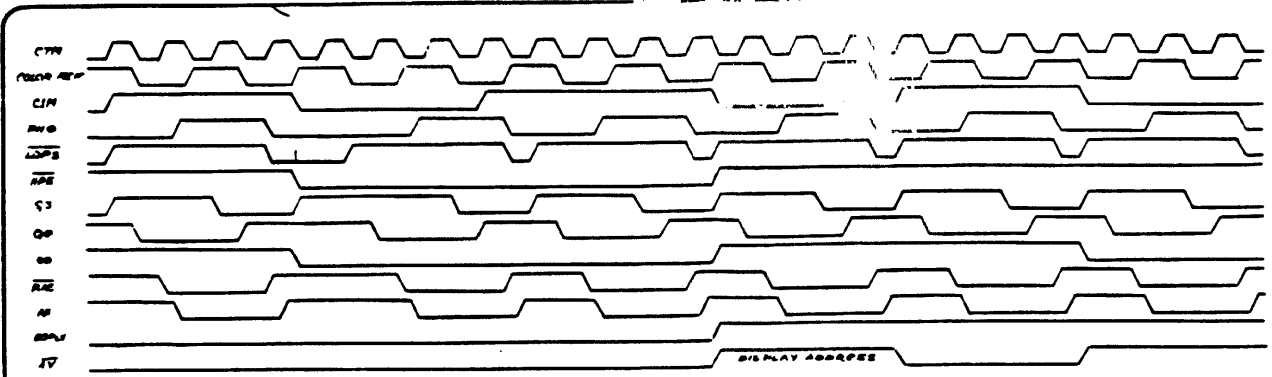


MAIN CLOCK - 14MHz

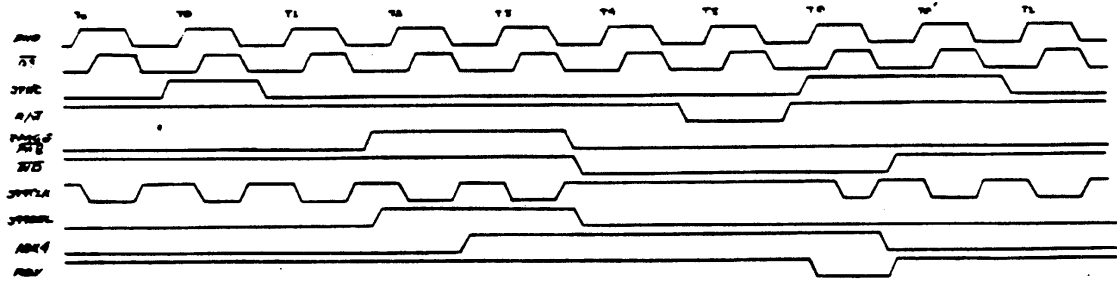


FREQUENCY DIVIDER.

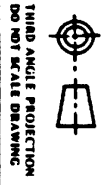




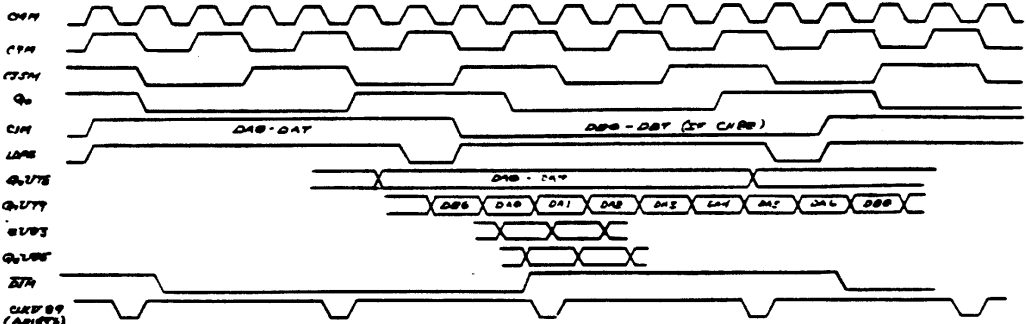
EXTRA HPE CYCLE: DISPLAY TIMING



INDIRECT ADDRESS TIMING

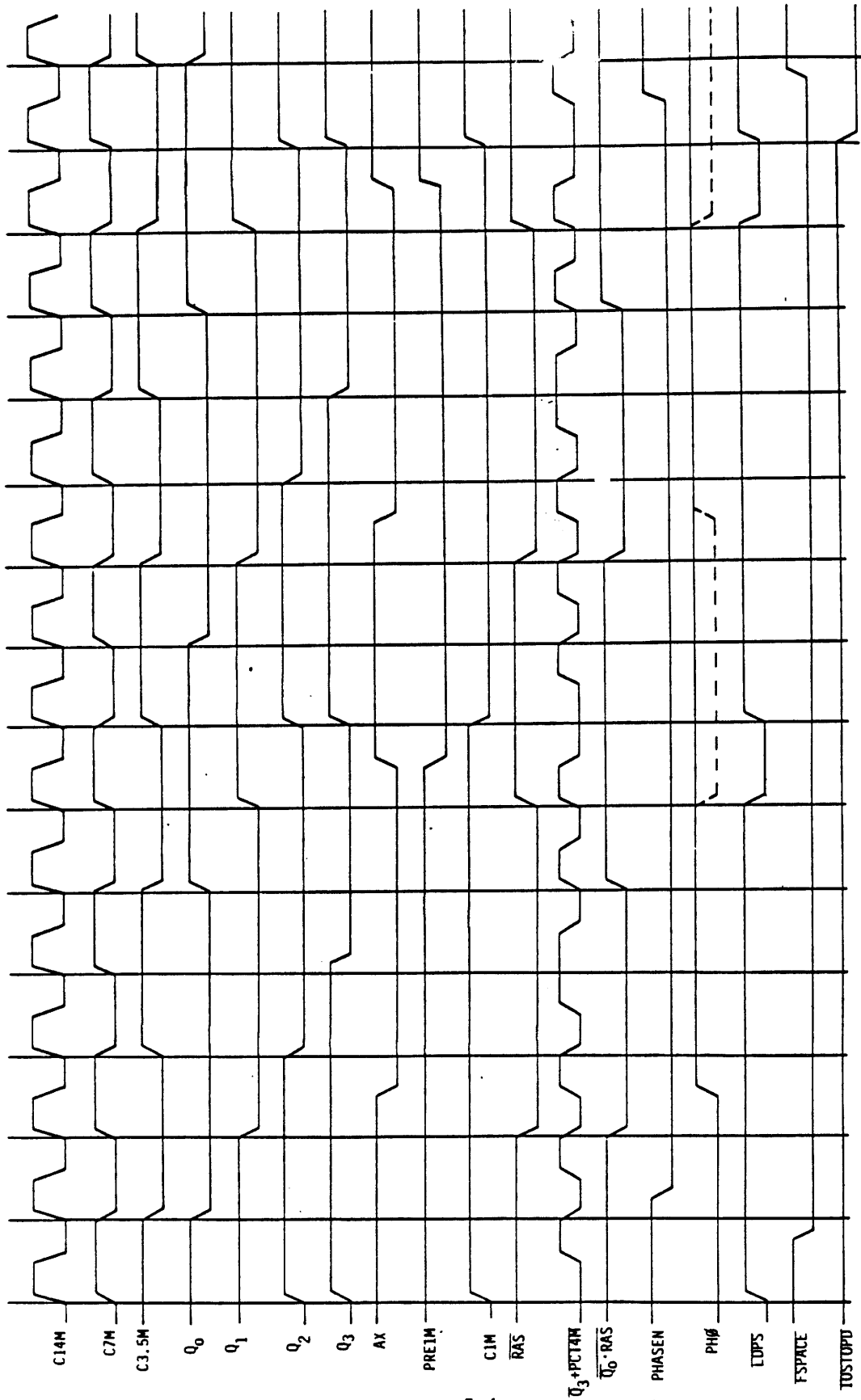


2	NEXT ASST	MATERIAL	FINISH	SCALE	SIZE	DRAWING NUMBER
TOLERANCES UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES DECIMALS ANGLES FRACTIONS IN PARENTHESES ARE IN HIGHLIGHTS		CHECKED BY DATE APPROVED BY DATE RELEASED BY DATE	DRAWN BY DATE PART NUMBER DATE	TITLE DESCRIPTION		
1	SCALE	SIZE	DRAWING NUMBER	SHEET		



VIDEO SIGNALS

A /// SYSTEM TIMING



5.4

APPLE III TIMING



for processor and RAM address timing.

The Q clocks are initialized with each Load Parallel to Serial pulse (LDPS), which changes the mode of the LS195 (D10) from a shift register to a parallel loaded register when low. At each load, all of the bits are set high. When LDPS* returns high the next clock will shift the zero of Q3* across the register. This means that Q0 will stay high for one clock cycle. Q1 will stay high for 2 clocks, etc. When the Q3* goes high at the fourth clock edge after LDPS the JK input will now see a "1". Subsequent clocks will start shifting that one across the register. When Q1 goes high again, LDPS* will be enabled low and another load will be accomplished at the next clock edge. The waveforms are asymmetrical. Each of Q0-Q2 are up for three clocks and down for four. Q3 is up for four and down for three.

This type of cycling will continue for 128 cycles. Then the Horizontal Phase Disable (HPE*) "freeze" will occur.

HPE* FREEZE

The HPE* signal will cause the Q states to extend their next cycle by two clock times. The purpose of the shift is to shift the phase of the color reference signals to the data in the video generator. A detailed discussion of this phenomenon is described in the video generator section. How this shift occurs is discussed below.

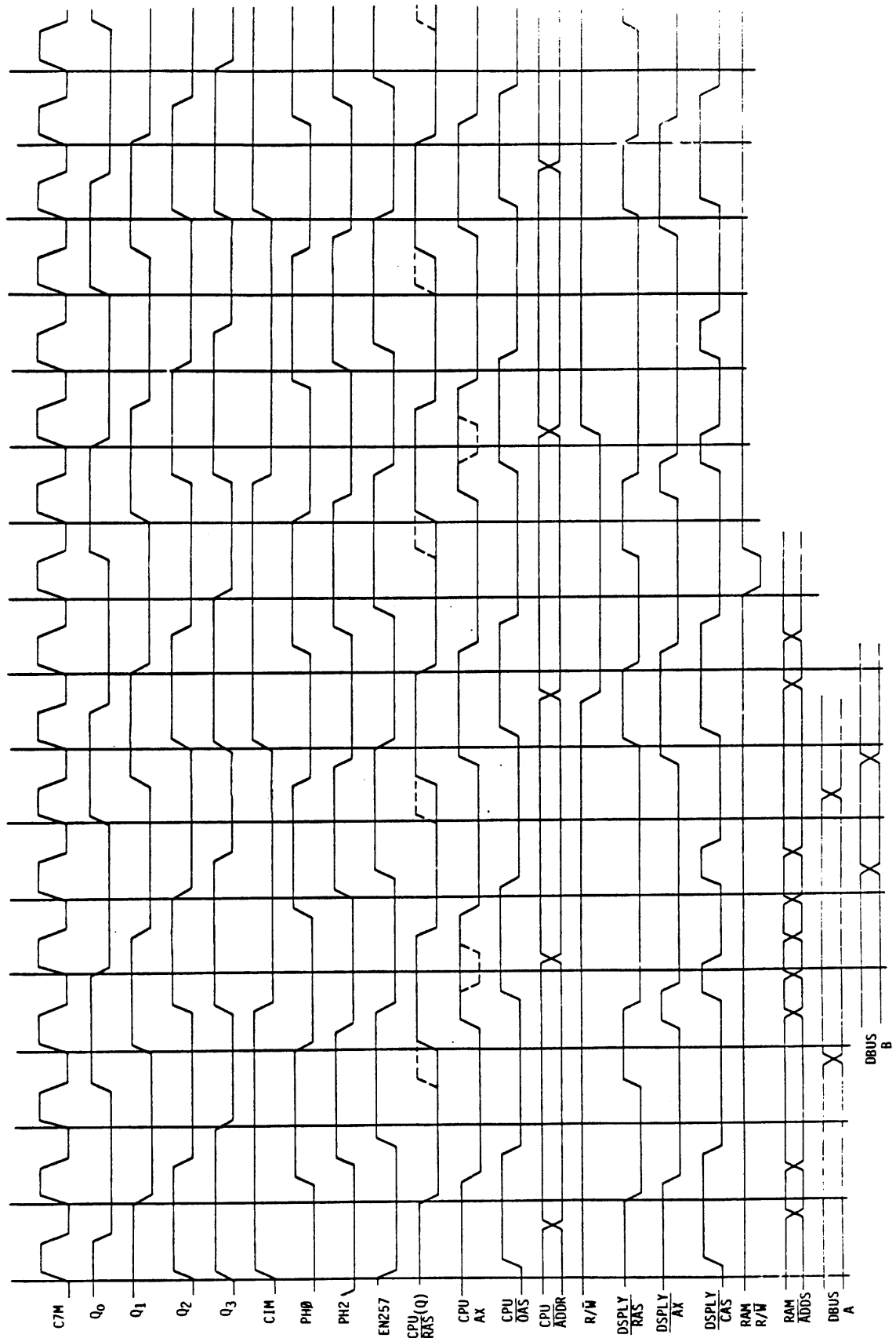
Looking at the gate array of C9 and B11 we see that since HPE* is normally high, the output of C9 is normally low. This de-gates the And input shared with Q0, and allows LDPS to function as usual. But when HPE* goes low, which will always coincide with C14M going low, the And gate shared with Q0 will become enabled and cause the extension of the LDPS* for two extra clock cycles. This state will exist until C3.5M returns high and relieves LDPS*. The next clock will change the state of Q0 and that will not be able to "disrupt" the clocks until the next HPE pulse.

AX, PRE1M, & C1M

The AX, RAM address (the signal used to select which addressing source [row or column] is presented to the RAMs: see RAM Address Logic) is another 2 Mhz signal which lags Q1 by one half clock cycles. It is developed at A11 pin 9.

PRE1M can toggle at each positive edge of AX, if the data input to the latch is at the opposite state. Looking at D11 (C1M*) we will be able to see just that. If PRE1M has just toggled low, one half clock cycle later C1M* will toggle high which forms the data input to the PRE1M Flip-Flop. But remember, the clock to the flip-flop is AX, a 2 MHz signal. D11 pin 5 acts much like B12 pin 2 in that the data is always opposite to the "Q" output of the latch at the time of the clock edge, therefore we have a "clock divided by 2" function, of a 1 MHz output.

RAS (Row Address Strobe)



5.6



The RAS signal is Q0 delayed by one clock and inverted. This is accomplished at D10 pin 15 & 14. (Note: the inversion is done by calling the "Q" output the active low signal RAS*, clever huh?) RAS is used in the RAM address logic to develop the "row select" signals for the RAMs.

VIDEO HORIZONTAL & VERTICAL STATE COUNTERS

This circuit is made up of four 4-bit binary counters, (F10, F11, G11, G12), which develop the essential signals for partitioning the screen and addressing the RAM for all the video data.

Basically, there are two sections of the circuit:

- 1 - horizontal position counter
- 2- vertical position counter

These two counters form the X and Y coordinates of each addressable byte on the screen. Each byte contains 7 bits or dots in 40 character modes.

From the various discussions about the Apple ///, we have learned that in the 40 character mode there are 280 dots across the horizontal line that can be defined, and 192 of these horizontal line (280 X 192). In the Apple /// modes there are 560 dots in the horizontal line, however, there are still only 192 horizontal lines (560 X 192).

The Video Counter works identically in either of these modes. It provides the resolution of 40 by 192 matrix. Each one of the 40 horizontal positions defines either 7 or 14 dots (40 or 80 character modes, respectively). These dots are actually bits of data bytes in memory that are parallel loaded into a shift register and shifted out serially to the video monitor. In the 40 character modes the system loads the shift register at a 1 MHz rate and shifts at a 7 MHz rate. In the 80 character modes it loads at a 2 MHz rate and shifts at a 14 MHz rate. It is interesting to note that in either mode the state counter increments at a 1 Mhz rate.

The system provides two complete accesses for the video output per increment of the state counter, but in the 40 character modes one of these are masked out.

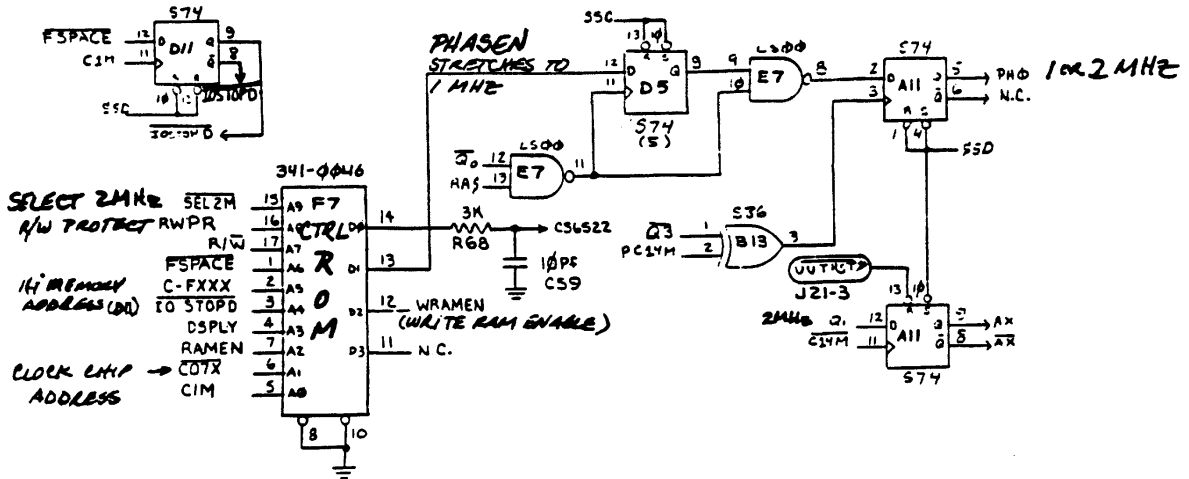
HORIZONTAL SECTION

The Horizontal section of the state counter uses 7 of the counter stages and develops the H0 through H5 and the HPE* signals. The remaining 9 stages of the counter develop the Vertical states VA, VB, VC, and V0 through V5.

The Horizontal section provides the capability of a 128 state counter, however, it only provides 65 states. This is due to the action of the most significant stage, HPE*. The counter actually counts from 64 to 128 then resets to count 64. Simply, HPE* starts high and when the counter increments HPE* low after 64 counts it is then reset to state 64 after the next clock input, this yields

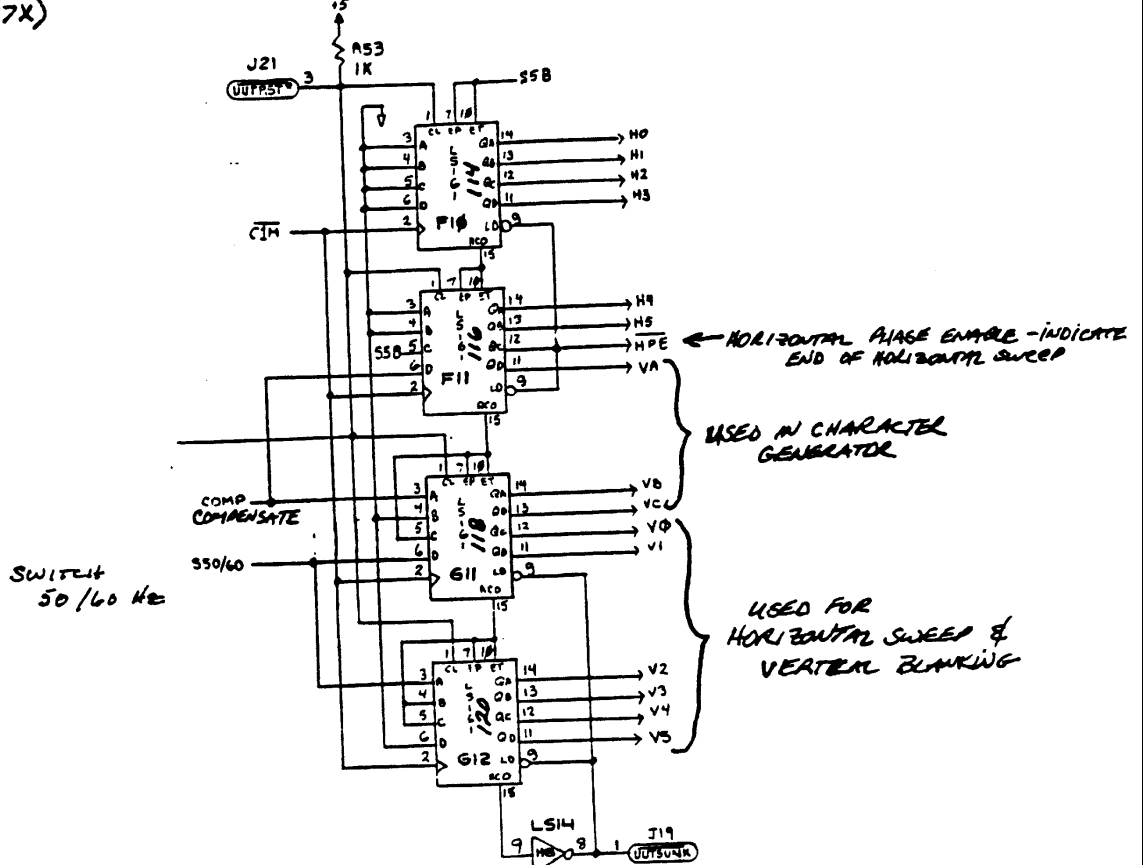
1 TO 2 MHz GEARSHIFT

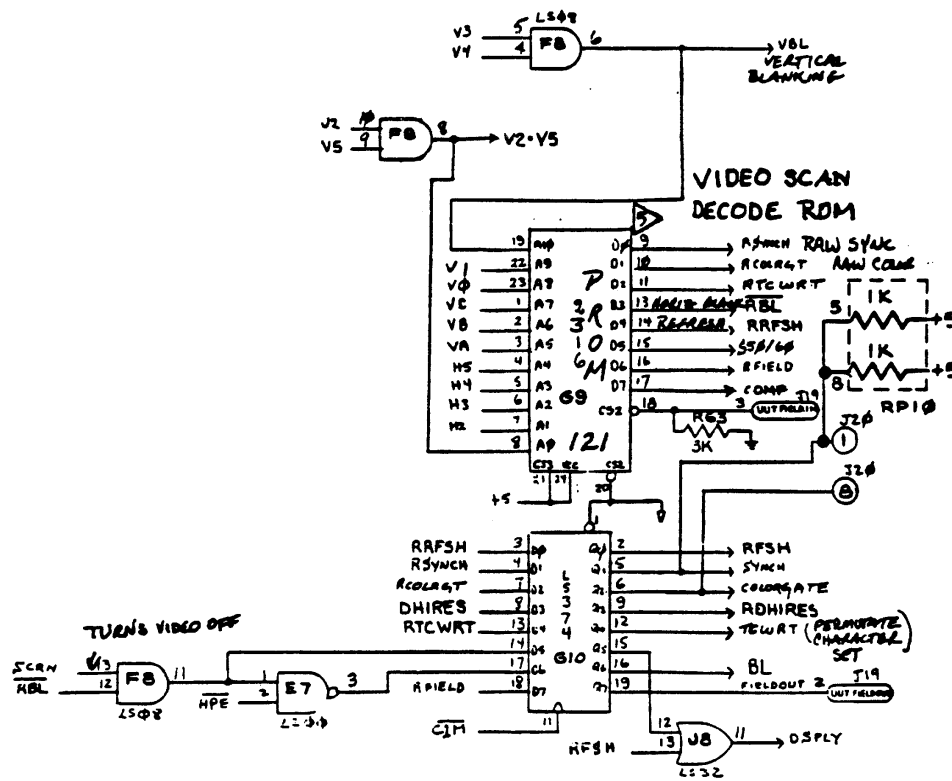
J4
PIN 8



FSPACE - RESULTS WHEN YOU ADDRESS THE VIA'S, THE ACIA, OR THE INTERNAL CLOCK (C07X)

VIDEO STATE COUNTER







65 states all together.

Looking at F11 we see that the HPE* output is connected to the "load" input of F10 and F11. At the next clock input these two devices will be loaded with the state determined by what is on the data inputs. All inputs except pin 5 of F11 are tied low (disregard the input to pin 6 of F11 at this time). This binary state equals 64. So rather than starting back at "zero" the counter jumps to 64.

For real time considerations of the monitor, it takes 25 states or 25 microseconds for the sweep to return from the right hand side to the left side. So the system ignores the first 25 states and blanks the video output and the returning trace is not shown. The boolean expression for the horizontal blanking would be expressed:

$$(H4^* \text{ and } H5^*) \text{ or } (H4 \text{ and } H3^*)$$

This logical function is done within the G9 Control ROM. Refer to page 10 of 10 of the schematic diagram.

In summary, the horizontal counter provides the address necessary for the display. It divides the horizontal line into forty (40) sections, and yields the timing for horizontal blanking. The HPE* signal is used to momentarily "freeze" some of the system timing.

VERTICAL SECTION

The Vertical State Counter provides the Y-axis of the display matrix. The nine stages, if left alone to count, would provide 512 states. As in the horizontal counter, it is preset to count higher than zero each time it reaches the "terminal count". Also, some of the states are used to blank the video while the trace returns from the bottom of the screen to the top (VBL).

The vertical counter effectively counts the number of HPE*'s that have occurred, or simply, the number of horizontal lines that have been generated in this scan.

At this time the counter is reset to count 250. Look at the timing diagram of the vertical counter. One can see that all vertical signals would normally go low, but instead the counter is loaded with the data inputs. VA will not be affected by the terminal count/load and will continue as discussed before. VB will be loaded to the present state of "comp" (or VA) which is high. VC and V5 will be loaded to a low and the rest of the bit states will be loaded to "1". This will decode to decimal 250.

Six counts later VA through V4 will toggle low and V5 will toggle high. This is the point where the logic assumes to be at scan line "zero". It will now take 256 counts to reach the terminal count sequence and start again. Using some math we see that the counter defines 262 states ($256+6 = 262$).

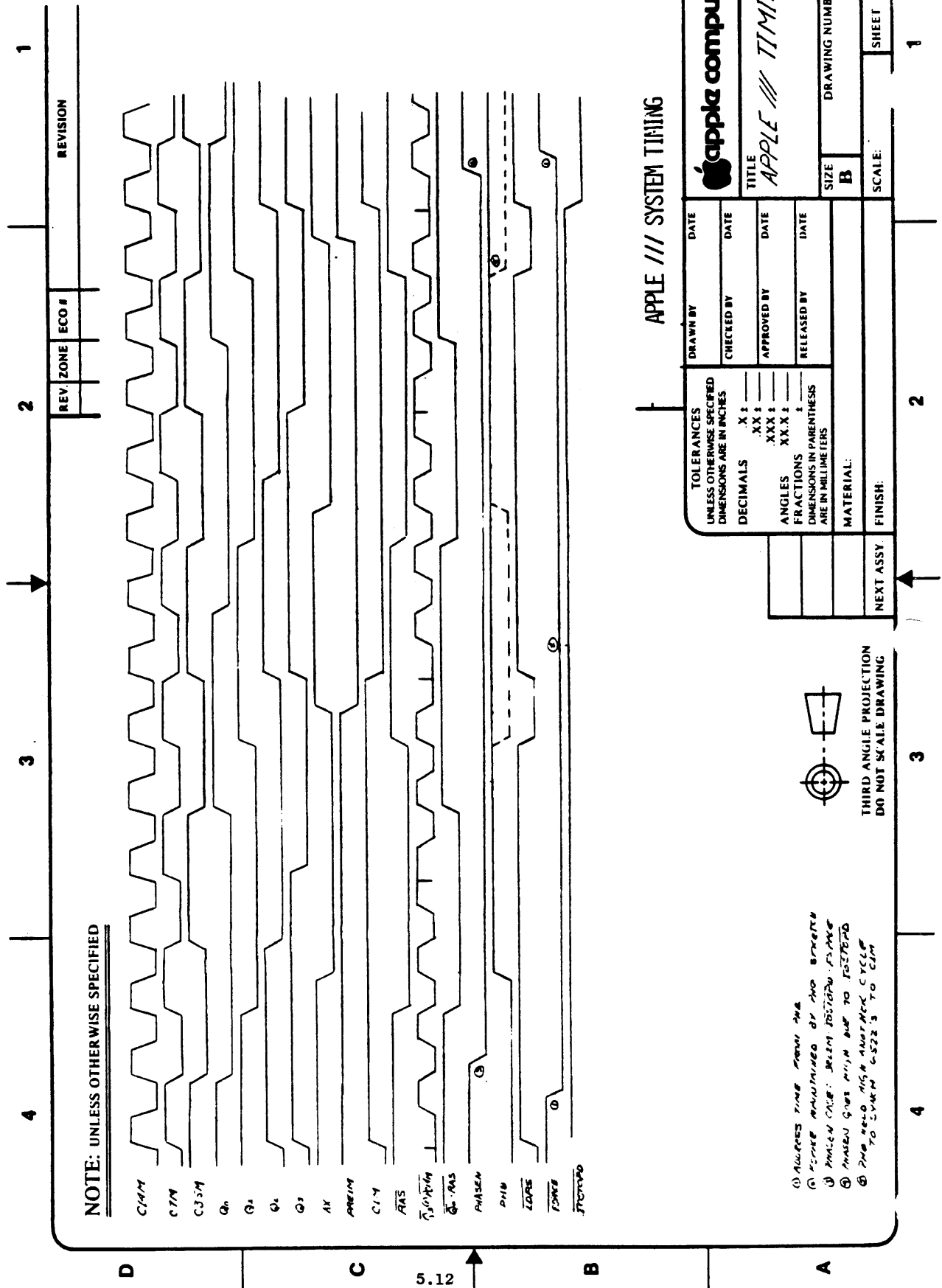
Vertical blanking takes 70 of the 262 states developed by the counter. The boolean expression for the vertical blanking signal would be:

$$(V3 \text{ and } V4)$$



This signal is developed at F8 pin 6 and is used in the system to indicate that a complete scan of the current display page has occurred. It is also an input to the Control Rom, G9, and therefore is a modifier to its outputs.

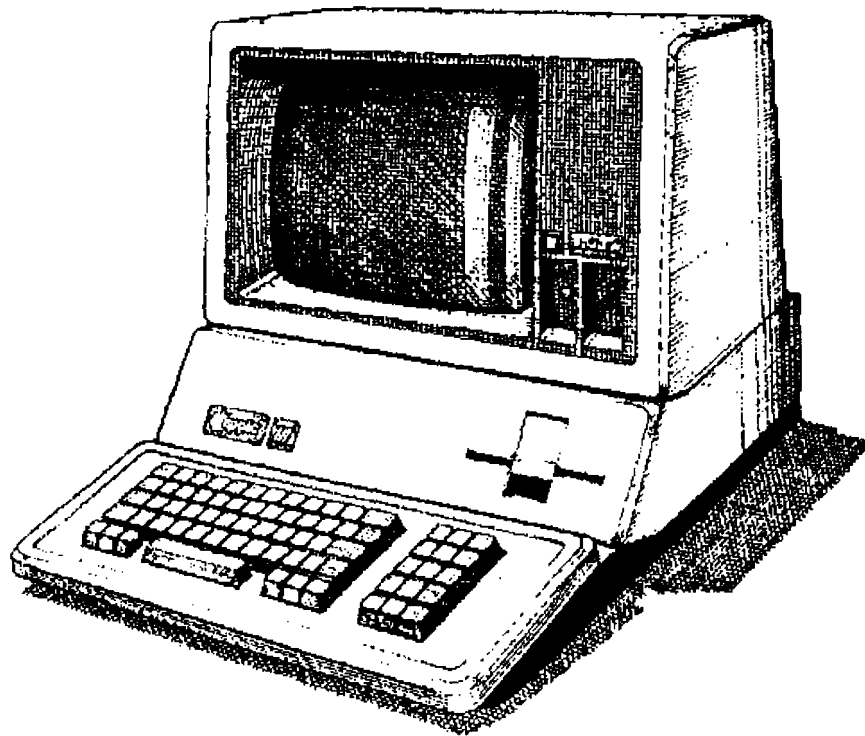
We were looking for the magic number of 192. Well, if you've been keeping track, it's simply the difference of 272-70.





Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 6 • Video Display Logic

Written by Apple Computer • 1982



DISPLAY MODES

- o 40 CHARACTER APPLE II : 40x24 CHARACTER B/W TEXT (2K BYTES RAM)
- o 40 CHARACTER APPLE ///: 40x24 CHARACTER COLOR TEXT - 16 BACKGROUND, 16 TEXT COLORS
- o 80 CHARACTER BLACK & WHITE APPLE ///: 80x24 CHARACTER B/W TEXT
- o BLACK & WHITE HIRES: 280x192 B/W HIRES (8K RAM)
- o MEDIUM RESOLUTION 16 COLOR GRAPHICS APPLE ///: 280x192 16 COLOR HIRES WITH 40x192 BACKGROUND/ FOREGROUND RESOLUTION
- o SUPER HIRES APPLE ///: 560x192 B/W HIRES
- o APPLE /// HIRES: 140x192 16-COLOR HIRES
- o RAM CHARACTER GENERATOR (128 CHARACTER)

APPLE /// VIDEOINTRODUCTION

The Apple /// has 11 defined video modes of operation. There are 5 Apple][modes and 6 new Apple /// modes. There are now 3 text modes and 8 graphics modes. Though the Apple /// can emulate all of the Apple][video modes, there are many differences in the video hardware between the Apple][and Apple ///, including:

- o 80 column text with full upper and lower case character capability
- o New color text mode
- o Super high resolution black and white graphics
- o 2 new color hires modes

AND

- o A modifiable character set

The modifiable character set is a major new feature of the Apple ///. You can now change the character set by changing the pattern in the character generator. This is possible because of a ram, instead of a fixed rom configuration.

There are also improved video outputs. An NTSC (National Television Standards Committee) composite Black and White and color composite, plus the primary video signals, are available at the back panel for mixing into the input of a high quality RGB monitor.

The Apple][emulation mode has the very same video modes as the Apple][. The Apple ///, while in its native mode, can have the following modes.

40 Character Apple][

This mode is equivalent to the Apple][text mode. The only difference is it has upper and lower characters.

- o The screen is divided into 40 horizontal columns and 24 vertical lines.
- o The characters are usually white dots on a black background.
- o This mode has inverse video and flashing characteristics.
- o This mode has no color.



- o This mode has two screen pages mapped into memory:
 - Page 1 is located at 0400-07FF
 - Page 2 is located at 0800-0BFF.

40 Character Apple ///

This second 40 character text mode is the most interesting and, in a way, the most powerful. This is the only color text mode. It has the same screen resolution as the Apple][, and the same video attributes. BUT it also has the ability to select both the color of the foreground (dots) and the color of the background. Sixteen (16) colors are available as in the Apple][Lores Graphics.

- o The color resolution can be selected for each character and can change for each character.
- o It is interesting to note that by down loading a character set, a new low resolution graphics mode can be manufactured from a text mode.

The page mode is different for this mode since both pages are used at once. Why? Because the first page contains the character data and the second page contains the color information. The page 2 mode reverses the mapping, that is, the characters in page 2 are stored where the color was stored in page 1, and vice versa.

In the color byte, bits 4-7 set the foreground color and bits 0-3 set the background color. The mapping between color and character is 1:1. That is, a character located in 0409, for example, has its foreground color determined by the byte in location 0809.

In the page 1 mode the mapping is as follows:

0400-07FF contain the characters
0800-0BFF contain the color information.

In the page 2 mode:

0800-0BFF contain the characters.
0400-07FF contains the color.

80 Character Black & White Apple ///

This new text mode is the same as the 40 column mode with the obvious exception that it has 80 columns instead of 40. This 80 column display has full upper and lower case, and inverse video.

Unlike the 40 character mode, it does not have 2 distinct pages. It uses both



pages to hold the characters.

The memory mapping for Page 1 utilizes:

0400-07FF for the primary fetch

0800-0BFF for the secondary.

In this mode, location 0400 contains the first character and 0800 contains the second. The third and the fourth characters come from locations 0401 and 0801 respectively.

In the Page 2 mode the primary fetch is from 0800-0BFF and the secondary from 0400-07FF. Therefore, the first and third characters come from 0800 and 0801 and the second and fourth come from 0400 and 0401.

Black & White Hires

This is a new graphics mode that has a 280 by 192 resolution in Black and White only.

It has two distinct pages:

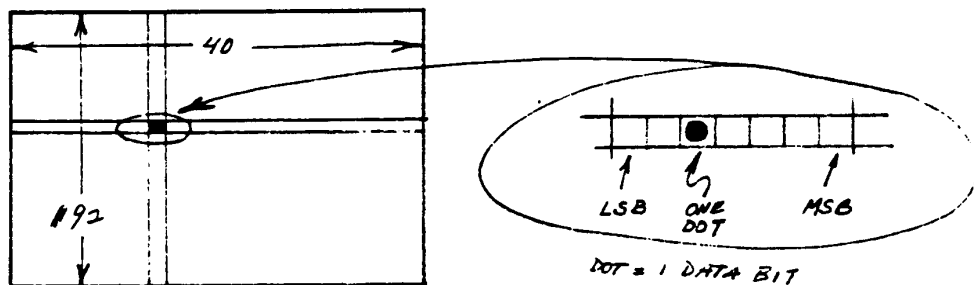
Page 1 is located at 2000-3FFF

Page 2 is located at 4000-5FFF.

Medium Resolution 16 Color Graphics Apple ///

This is a new graphics mode for the Apple ///. It has the same dot resolution as the Apple][Hires (280 by 192), but it has an expanded color capability of 16 background colors. The B/W Output will yield 16 levels of grey scale.

The screen is divided into a 40 wide by 192 high matrix. That is, the color selection for foreground and 16 background can change for each 7 dot [ooooooo] pixel segment. You can think of each segment as a one-bit-high slice across a character space, as illustrated below.



6.4



The memory mapping is as follows:

Page 1: 2000-3FFF each byte represents 7 pixels in the segment
4000-5FFF each byte represents the foreground and background colors
for the corresponding 2000-3FFF byte.

Page 2: 2000-3FFF each byte represents the colors
4000-5FFF each byte represents 7 pixels.

Super Hires Apple ///

This is the Apple /// Hires equivalent of 80 character mode. It is a Black and White mode which has the dot resolution of 560 Horizontal by 192 vertical spaces.

There are two distinct screen pages, each with a primary and secondary page. Because it is like the 80 character modes, this mode draws its information from alternating ram. Each memory byte contributes 7 pixels. In Page 1 mode, the primary contains the odd dot groups and the secondary contains the even dot groups. The primary (first 7 pixels) is located at 2000-3FFF, and the secondary (second 7 pixels) is found at 4000-5FFF. In Page 2 the primary is at 6000-7FFF, and the secondary is 8000-9FFF.

In each byte the Most Significant Bit (MSB) is ignored and the data is displayed with the Least Significant Bit (LSB) first from left to right.

Apple /// Hires

This is the third new graphics mode. It has 140 by 192 pixel resolution, and 1 of 16 color selection per pixel. In this mode the pixel is formed by a group of four dots of the same color.

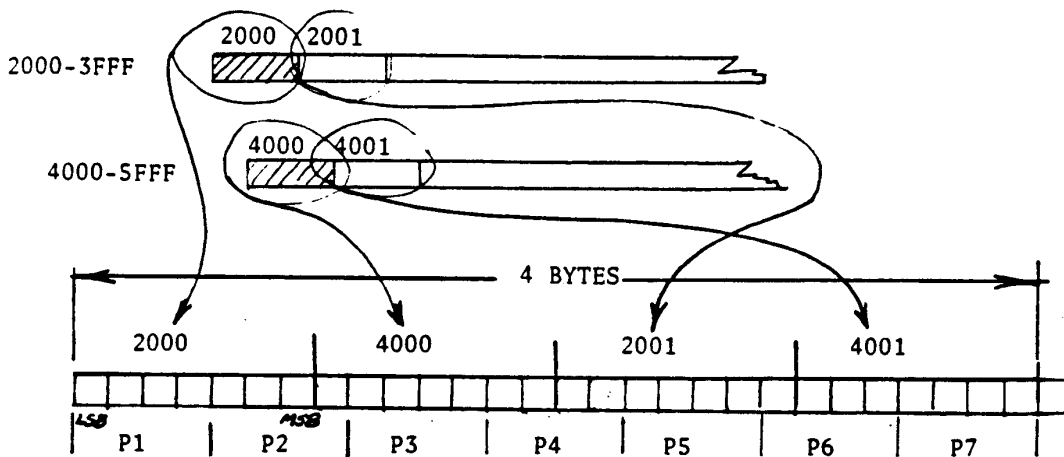
There are two distinct screen pages in this mode but the mapping of the individual pages is, at first encounter, a bit difficult to master. Good luck!

- o The display dot represents a sequence of 4 data bits in the RAM display area.
- o Two rams are used starting at 2000 and 4000 respectively and alternate bytes are fetched from each ram area.



- o In any video mode only 7 of the 8 bits of each byte are displayed.

With this information in mind...and remembering that each pixel in this mode is made from 4 bits...you can see that you need 4 bytes of information to get 7 pixels. The way in which these bytes map into picture elements is shown below.





4000-5FFF each byte represents 7 pixels.

Super Hires Apple ///

This is the Apple /// Hires equivalent of 80 character mode. It is a Black and White mode which has the dot resolution of 560 Horizontal by 192 vertical spaces.

There are two distinct screen pages, each with a primary and secondary page. Because it is like the 80 character modes, this mode draws its information from alternating ram. Each memory byte contributes 7 pixels. In Page 1 mode, the primary contains the odd dot groups and the secondary contains the even dot groups. The primary (first 7 pixels) is located at 2000-3FFF, and the secondary (second 7 pixels) is found at 4000-5FFF. In Page 2 the primary is at 6000-7FFF, and the secondary is 8000-9FFF.

In each byte the Most Significant Bit (MSB) is ignored and the data is displayed with the Least Significant Bit (LSB) first from left to right.

Apple /// Hires

This is the third new graphics mode. It has 140 by 192 pixel resolution, and 1 of 16 color selection per pixel. In this mode the pixel is formed by a group of four dots of the same color.

There are two distinct screen pages in this mode but the mapping of the individual pages is, at first encounter, a bit difficult to master. Good luck!

- o The display dot represents a sequence of 4 data bits in the ram display area.
- o Two rams are used starting at 2000 and 4000 respectively and alternate bytes are fetched from each ram area.
- o In any video mode only 7 of the 8 bits of each byte are displayed.

With this information in mind...and remembering that each pixel in this mode is made from 4 bits...you can see that you need 4 bytes of information to get 7 pixels. The way in which these bytes map into picture elements is shown below.



It is apparent, from the diagram, that picture elements overlap the byte boundaries for 7 picture elements and 4 bytes. The basic pattern then repeats.

The four bytes are shifted out in a fashion similar to the other Apple /// modes:

- o The first byte comes from the primary and the second byte comes from the secondary.
- o The first byte contains the first pixel and the second byte comes from the secondary.
- o The first byte contains the first pixel and 3 bits of the second pixel.
- o The second byte contains the fourth bit of the second pixel, the third pixel, and the first two bits of the fourth pixel.
- o The third byte contains the last two bits of the fourth pixel, the fifth pixel, and the first bit of the sixth pixel.
- o The fourth byte contains the last three bits of the sixth pixel and the entire seventh pixel.

We hope the preceding diagram will help you picture what has already been described.

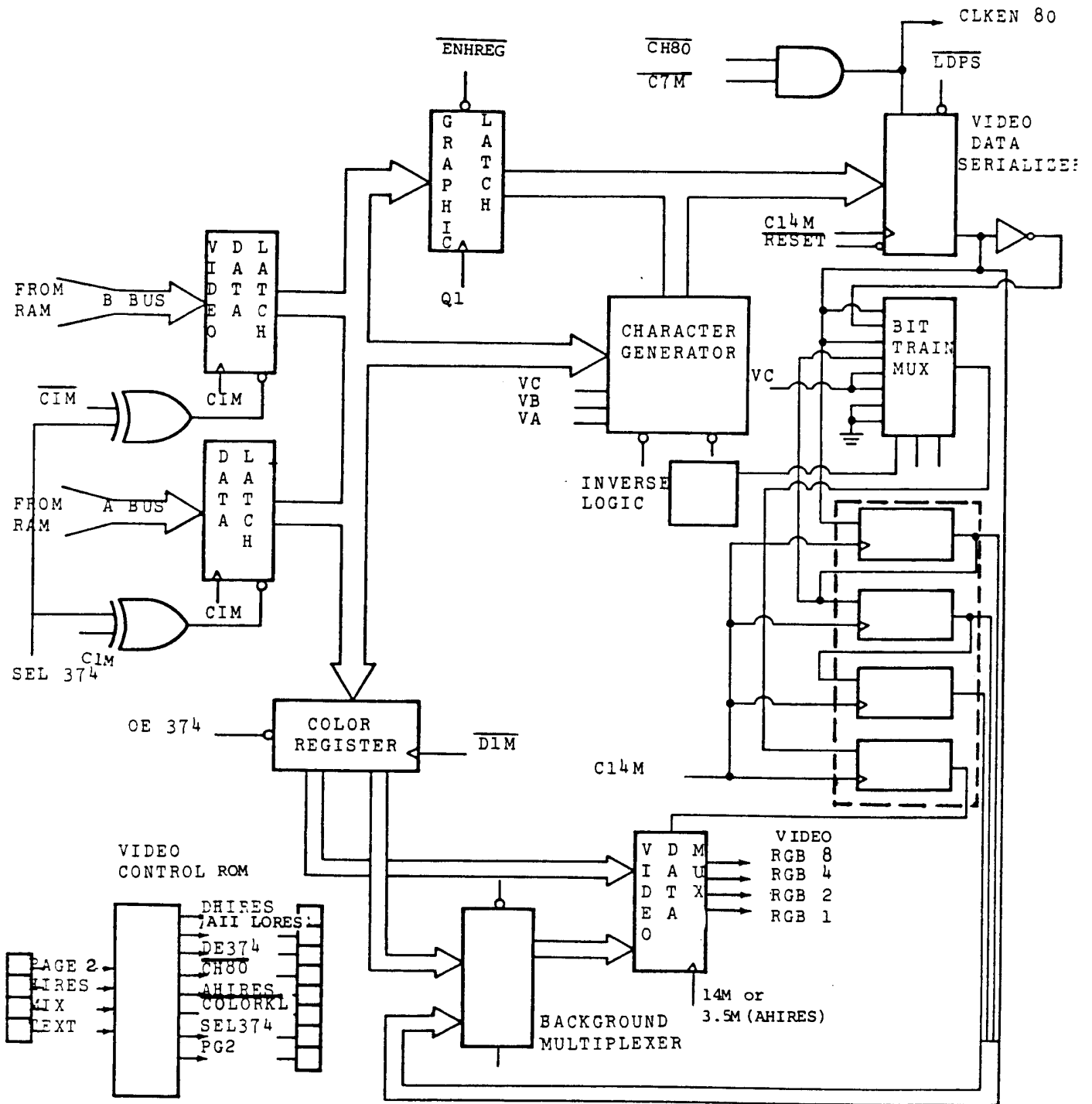
For this mode, Page 1 is mapped with the primary fetch in 2000-3FFF, and the secondary in 4000-5FFF. In Page 2 the primary is in 6000-7FFF, and the secondary is in 8000-7FFF.



APPENDIX (VIDEO)

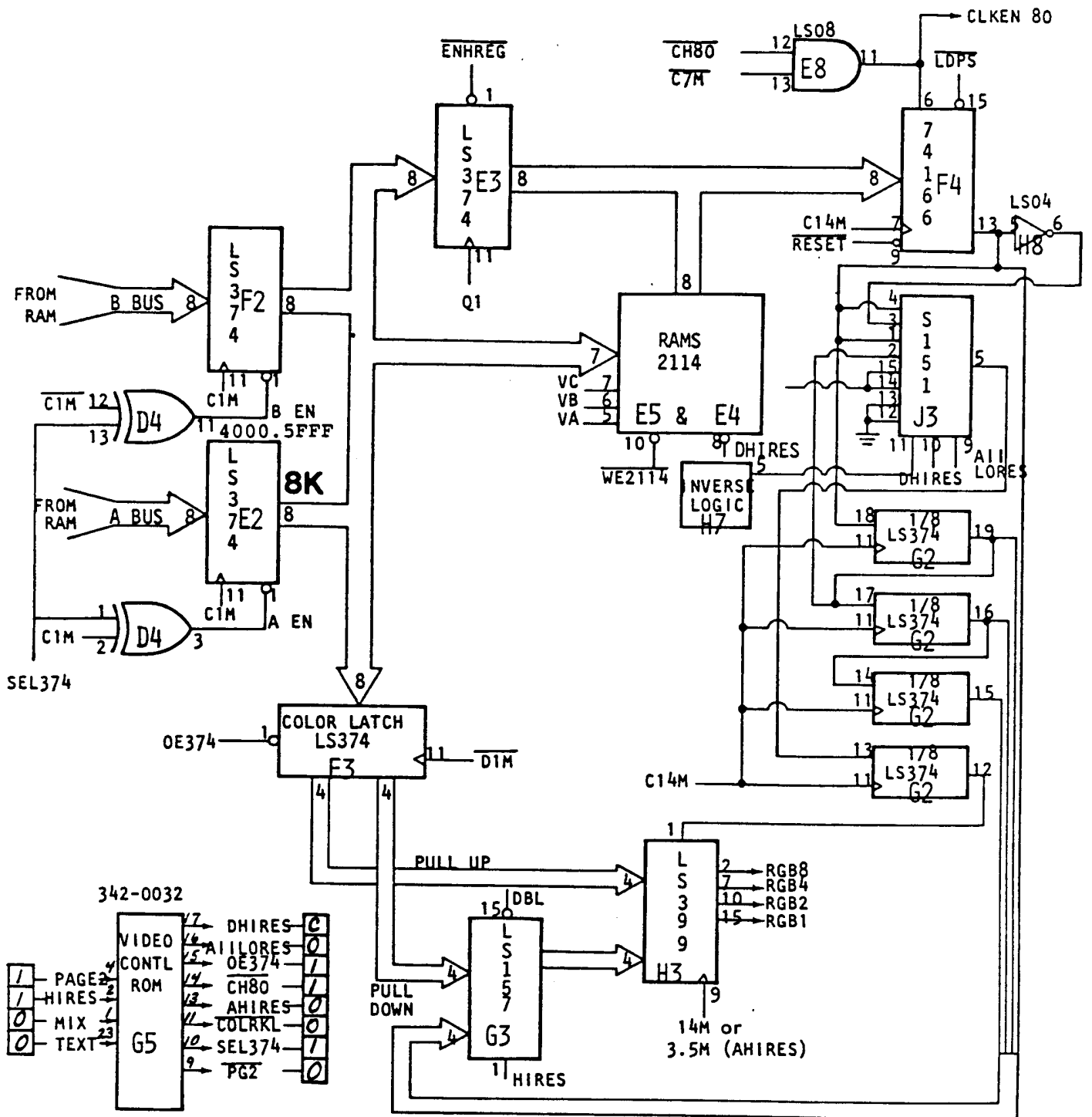
- o Apple /// Video Logic Block Diagram
- o Video Logic Diagrams for:
 1. Hires Mode Page 1, B/W 280 X 192
 2. Hires Mode Page 2, B/W 280 X 192
 3. Color Hires Mode Page 1, 280 X 192
 4. Color Hires Mode Page 2, 280 X 192
 5. Super Hires Mode Page 1, 560 X 192
 6. Super Hires Mode Page 2, 560 X 192
 7. Ahires Test Page 1, 140 X 192
 8. Ahires Test Page 2, 140 X 192
 9. Color Bar & Grey Scale Test
 10. Apple II Text Mode Page 1, B & W, 40 Column
 11. Apple II Text Mode Page 2, B & W, 40 Column
 12. Sara 40 Column Text Mode Test, 16 Colors
 13. Sara 80 Column Text Mode Test, B & W
- o Apple /// Video Modes Truth Tables
- o Video Prom Listing
- o Video Prom Equivalent Logic

APPLE III VIDEO LOGIC BLOCK DIAGRAM



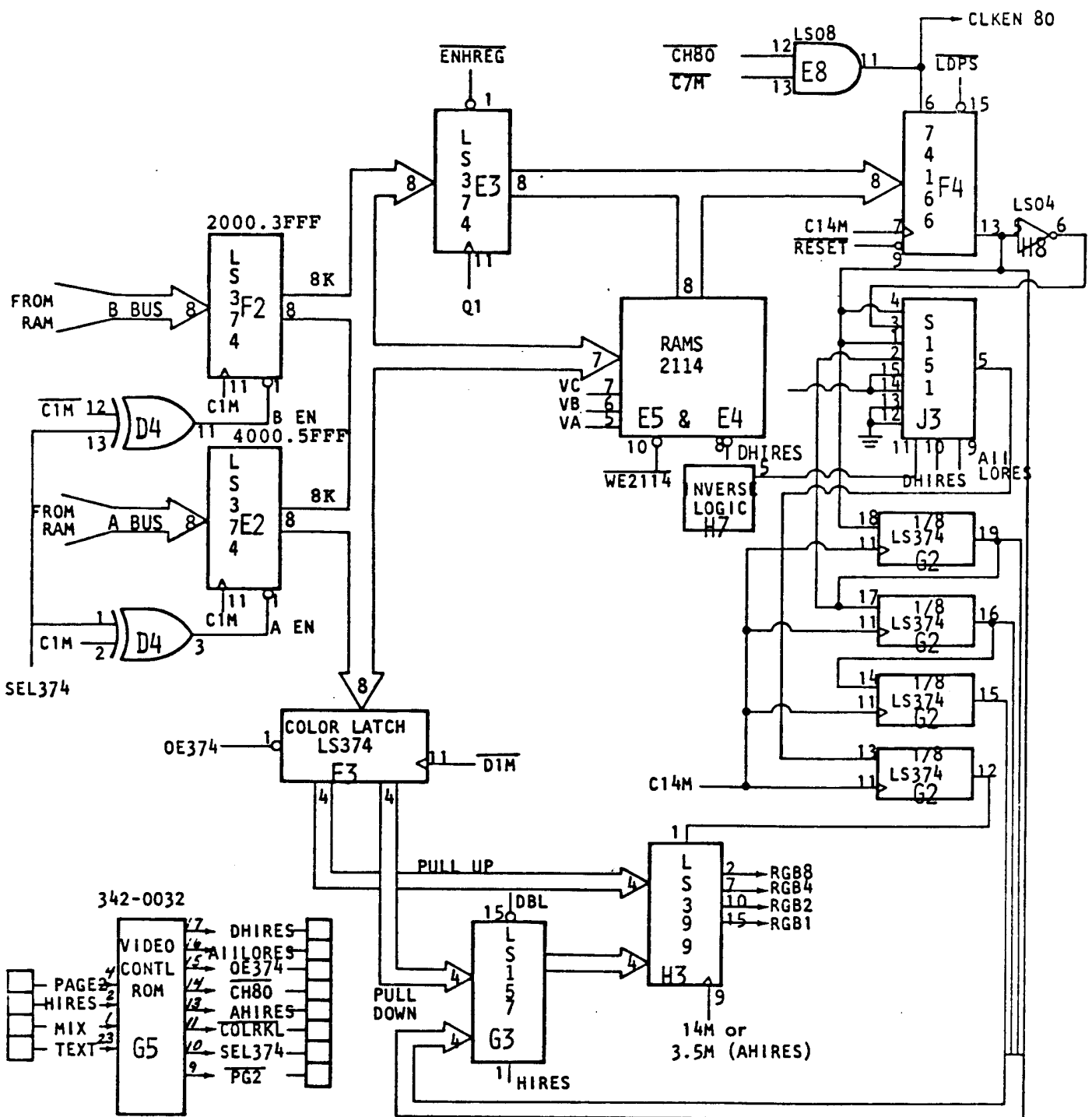
6.10

A/// VIDEO LOGIC DIAGRAM



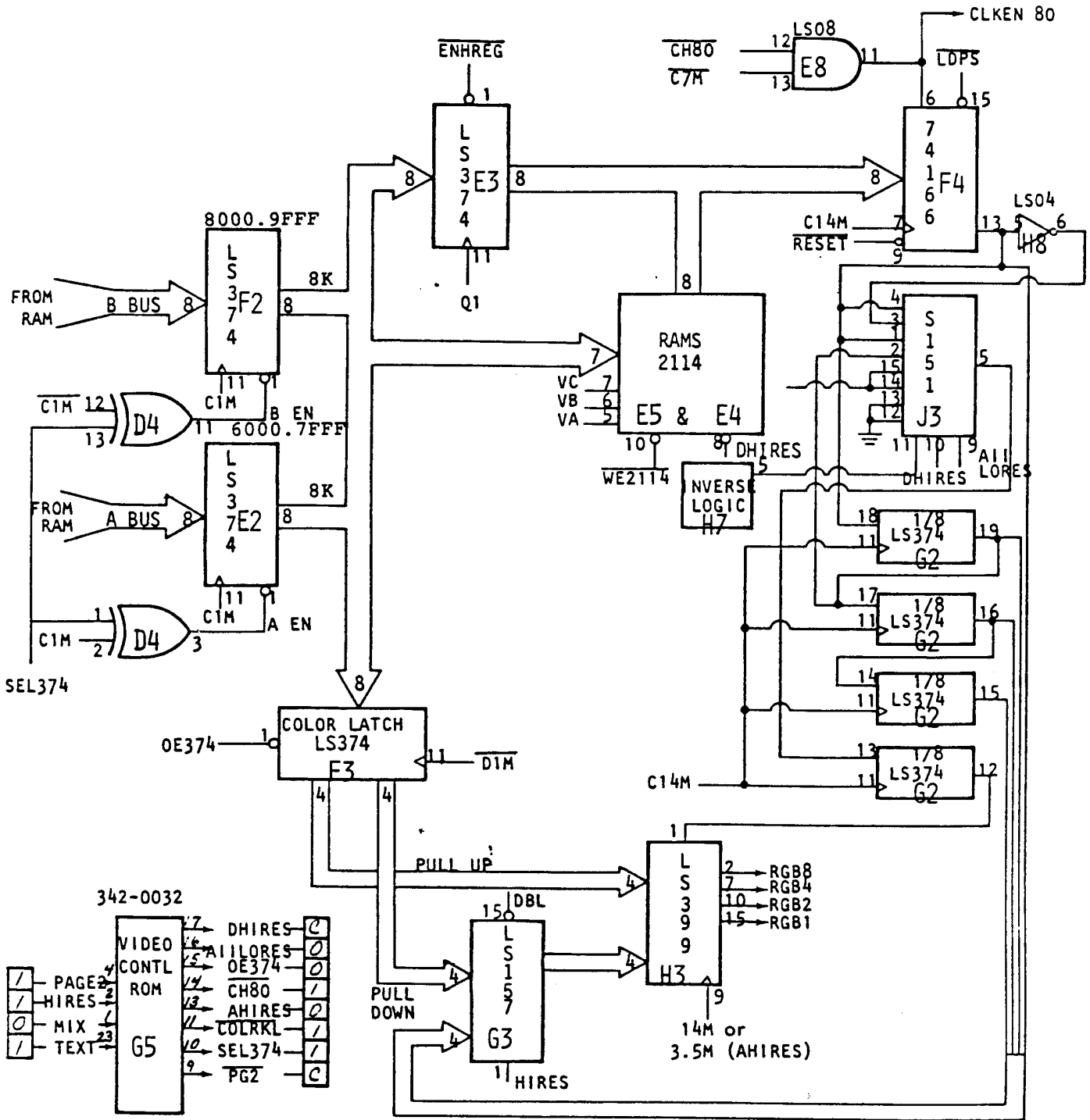
2. HIRES MODE PAGE 2 - B & W - 280 X 192
4000.5FFF (8K)

A/// VIDEO LOGIC DIAGRAM



3. 280 X 192 COLOR HIRES MODE PAGE 1 (FGD/BKGD HIRES)
2000.5FFF (16K)

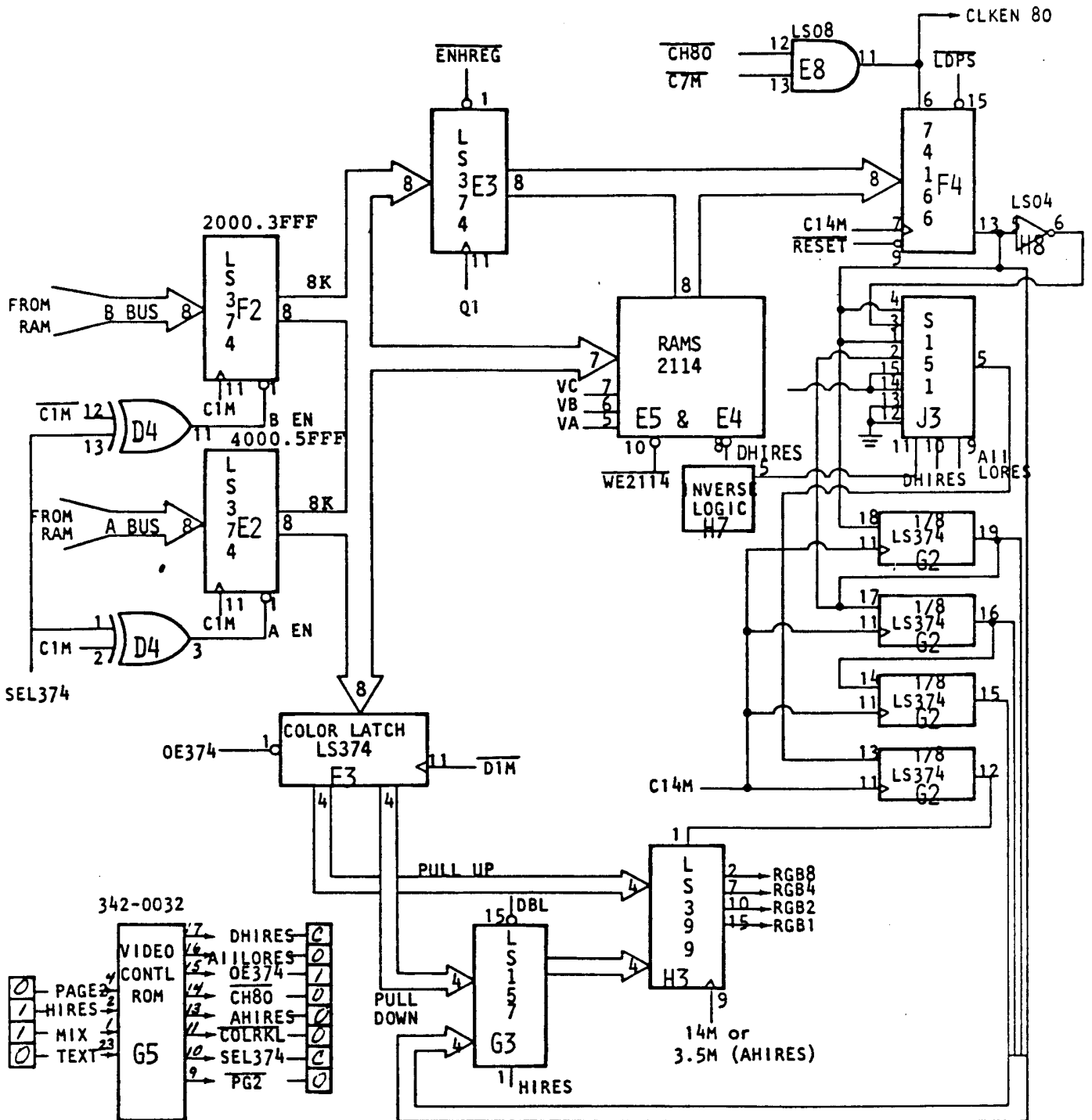
A/// VIDEO LOGIC DIAGRAM



4. 280 X 192 COLOR HIRES MODE PAGE 2 (FGB/BKGD HIRES)
6000.9FFF (16K)

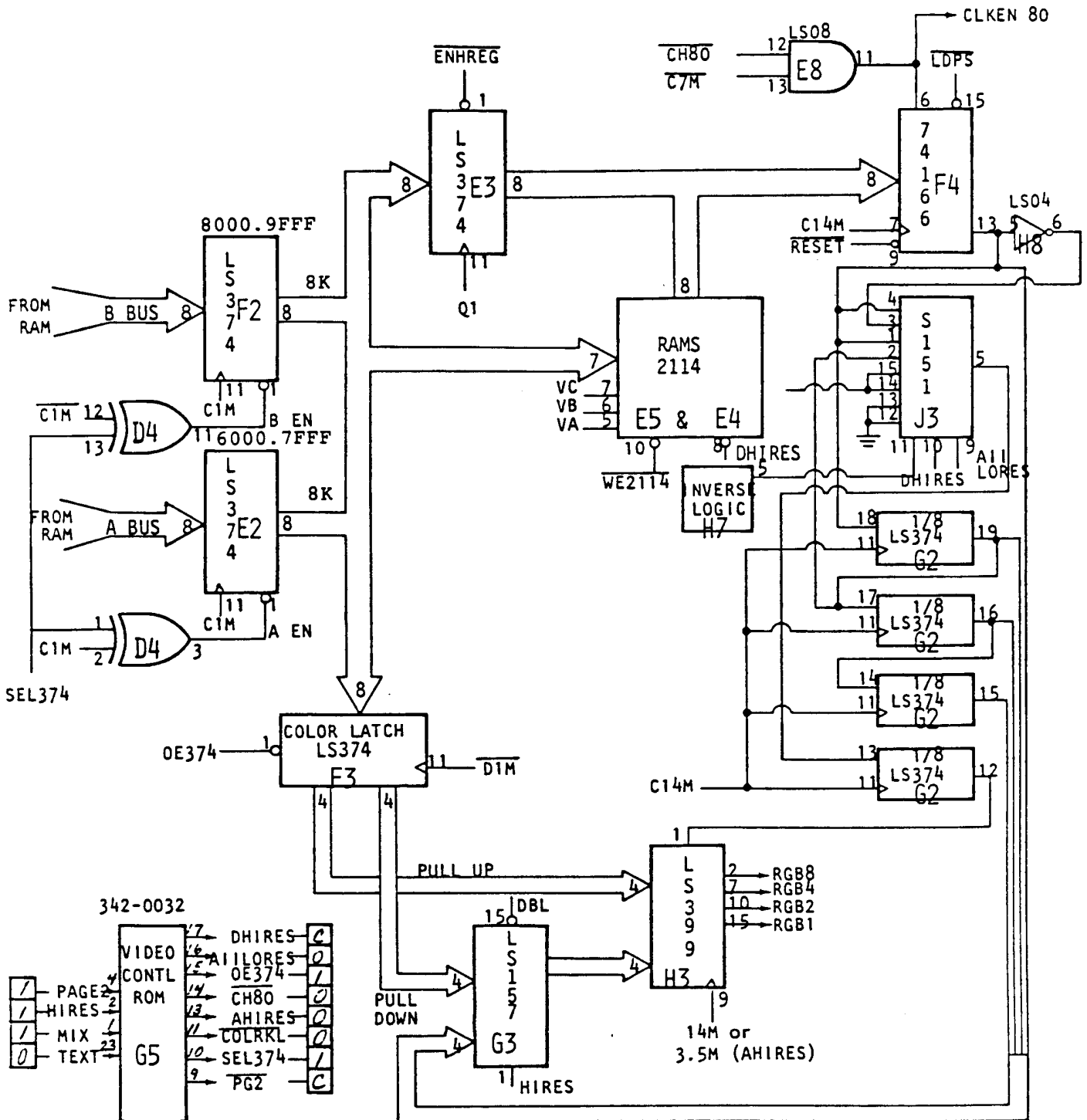
6.14

A/// VIDEO LOGIC DIAGRAM



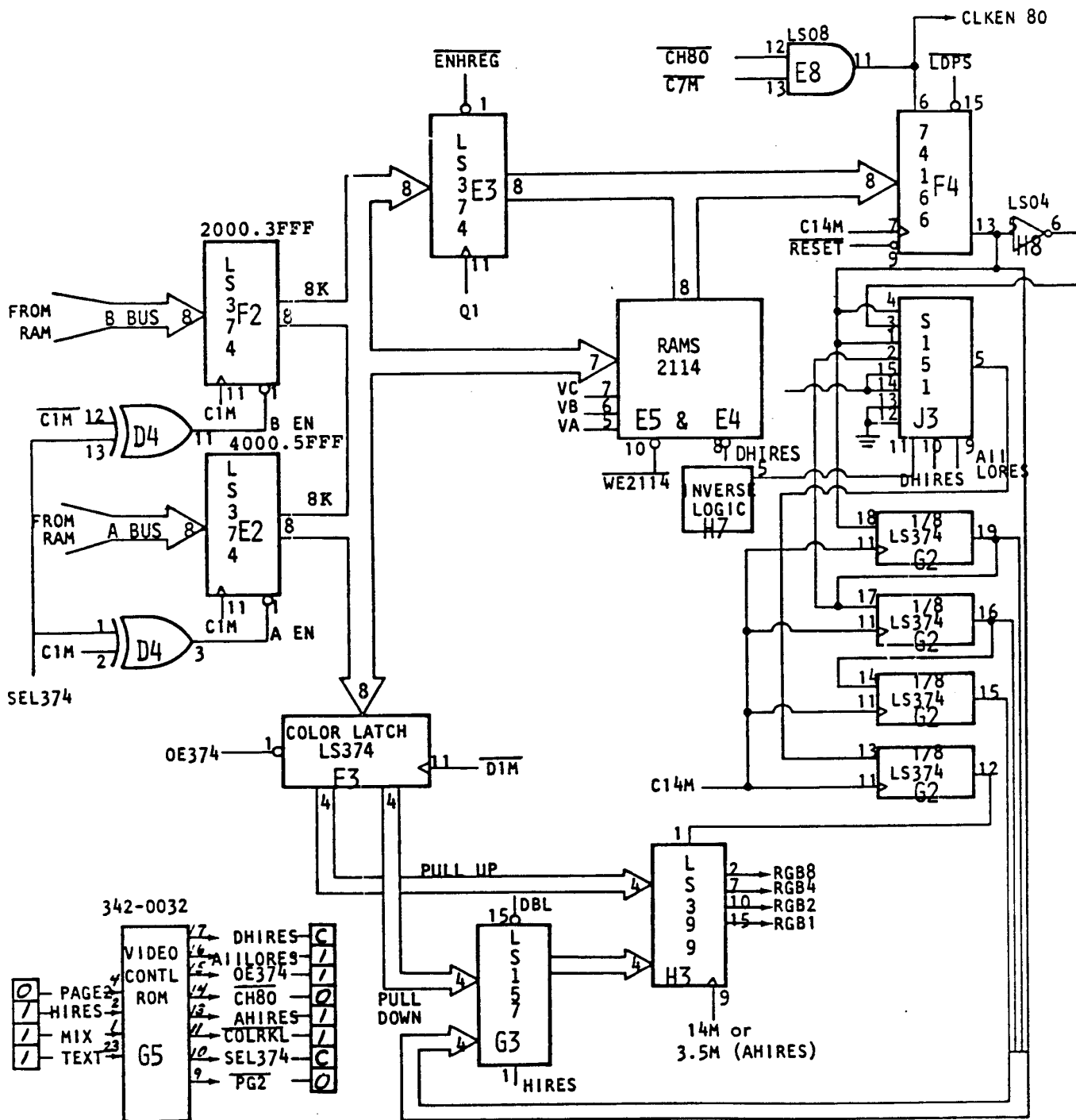
5. SUPER HIRES MODE PAGE 1 - B & W - 560x192
2000.5FFF (16K)

A/// VIDEO LOGIC DIAGRAM



6. SUPER HIRES MODE PAGE 2 - B & W - 560 X 192
6000.9FFF (16K)

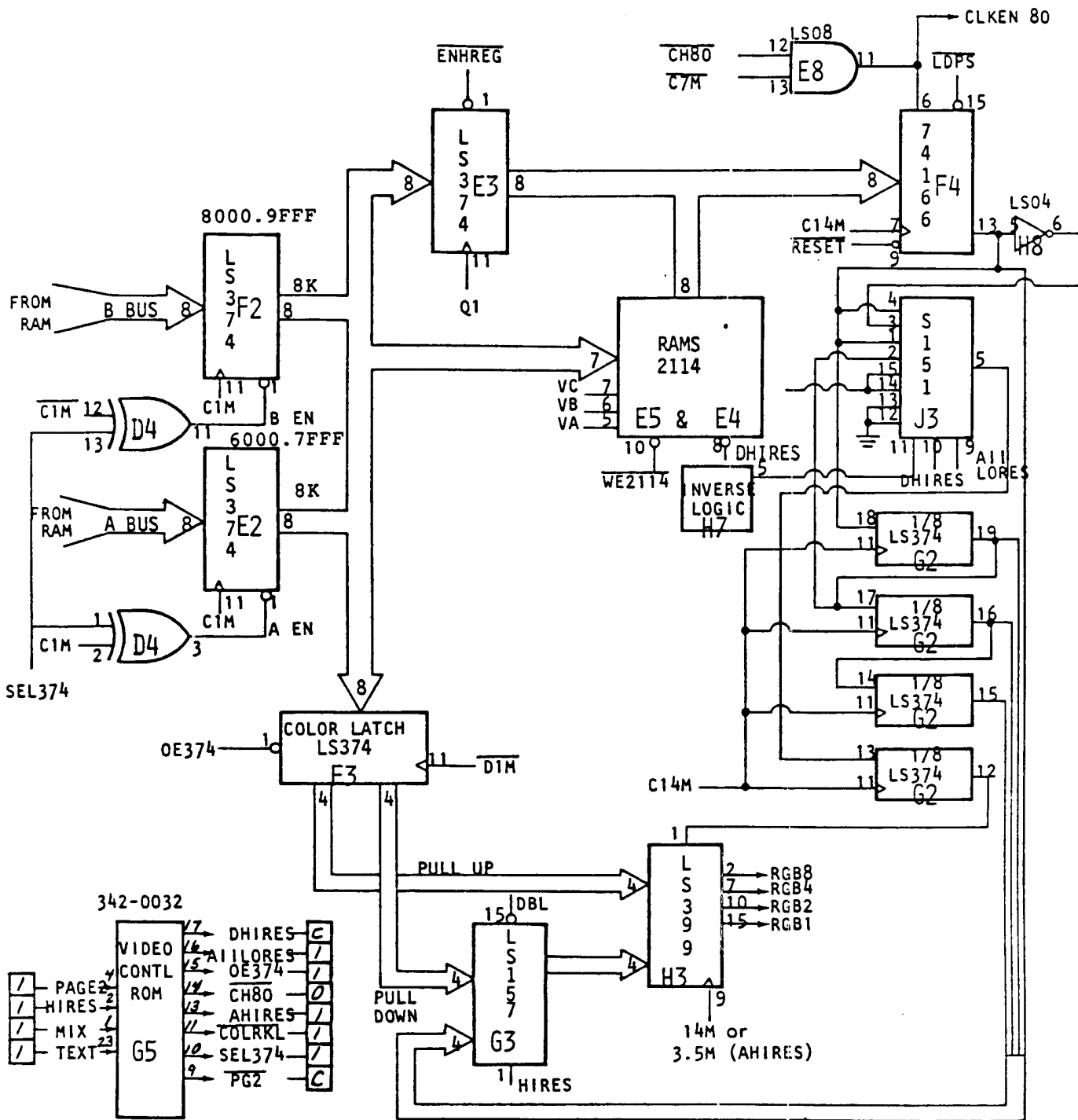
A/// VIDEO LOGIC DIAGRAM



7. AHIREs TEST PAGE 1 - 140x192
2000.5FFF (16K)

6.17

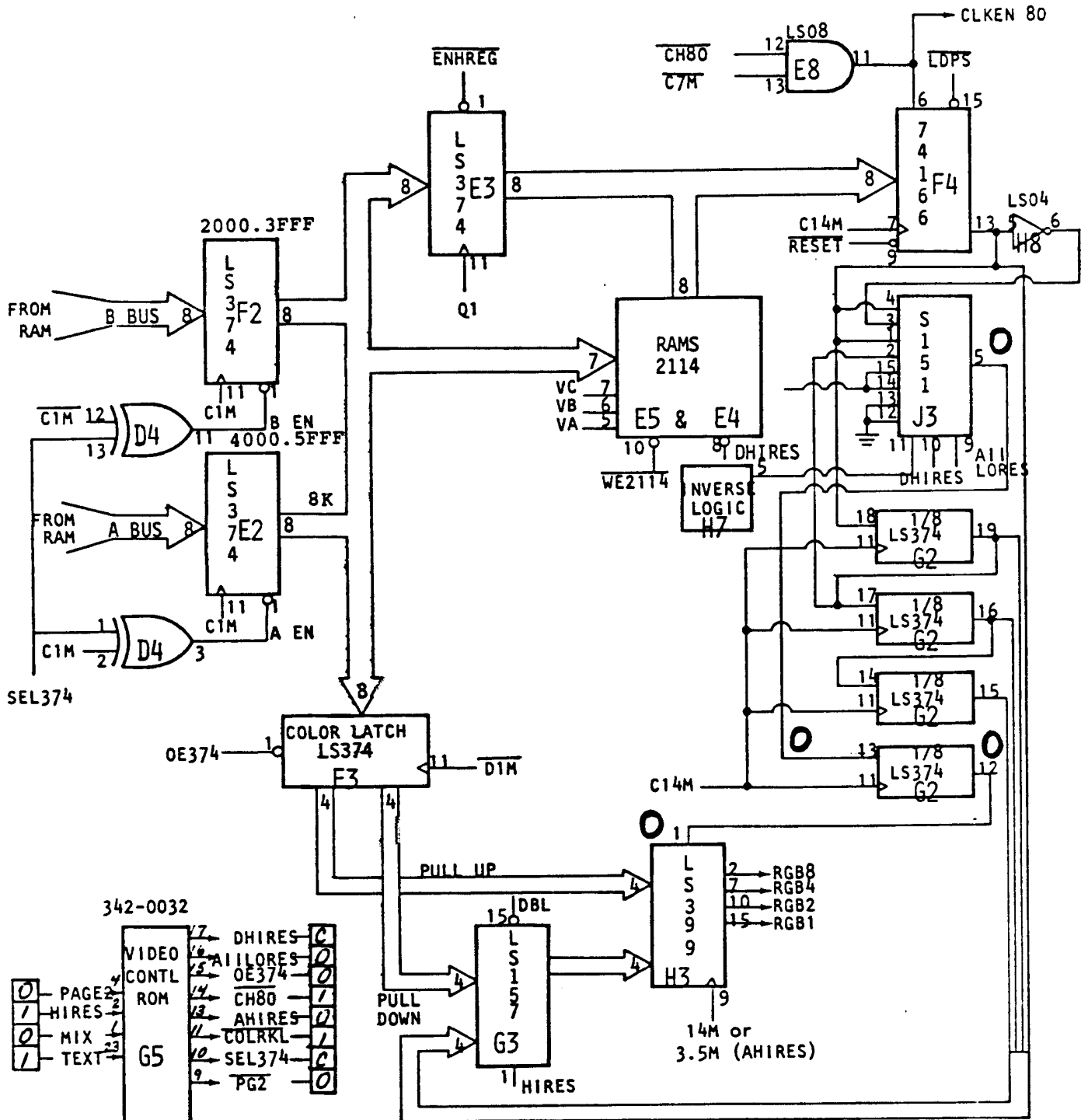
A/// VIDEO LOGIC DIAGRAM



8. AHIRES TEST PAGE 2 - 140X192
6000.9FFF (16K)

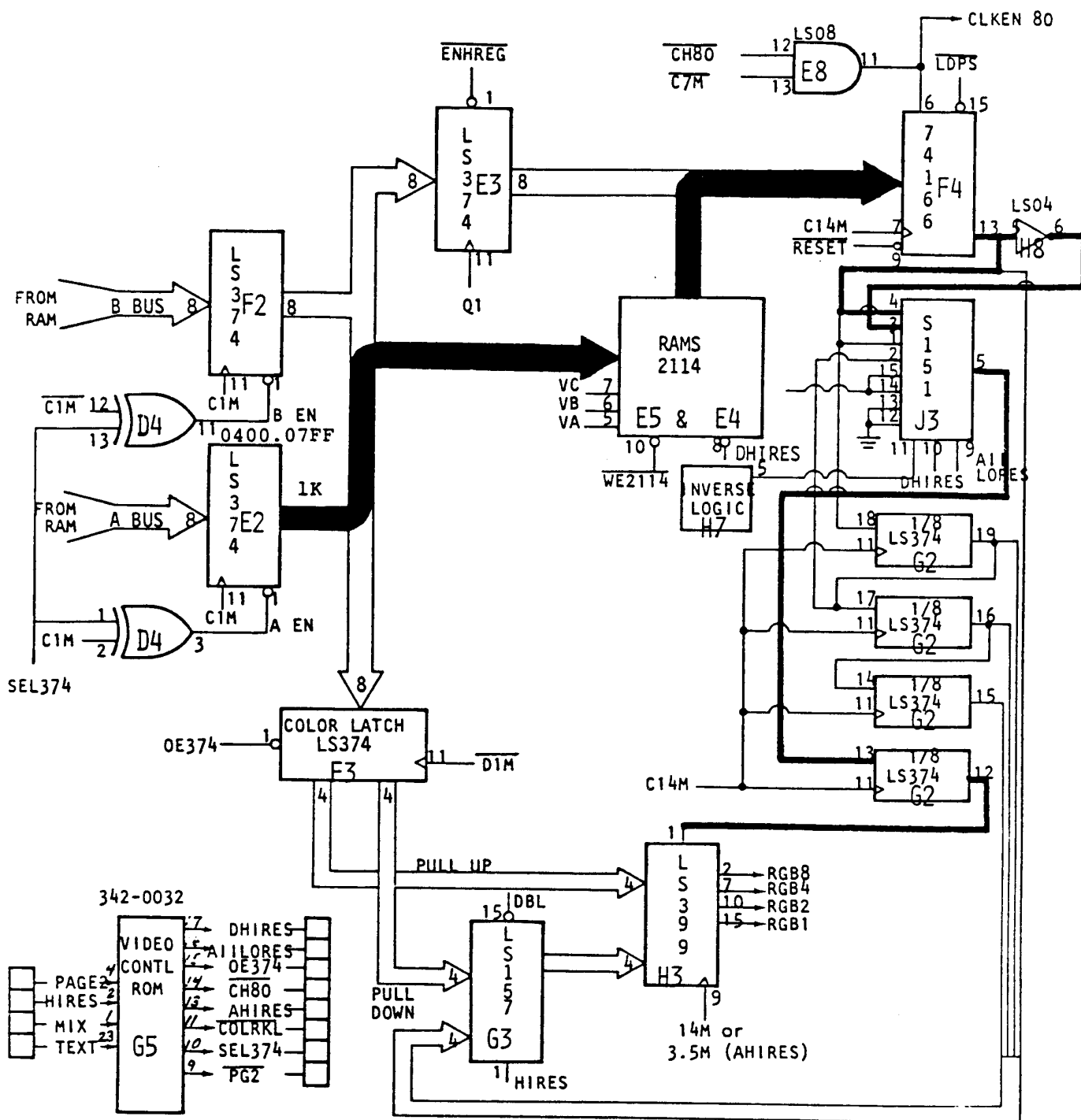
6.18

A/// VIDEO LOGIC DIAGRAM



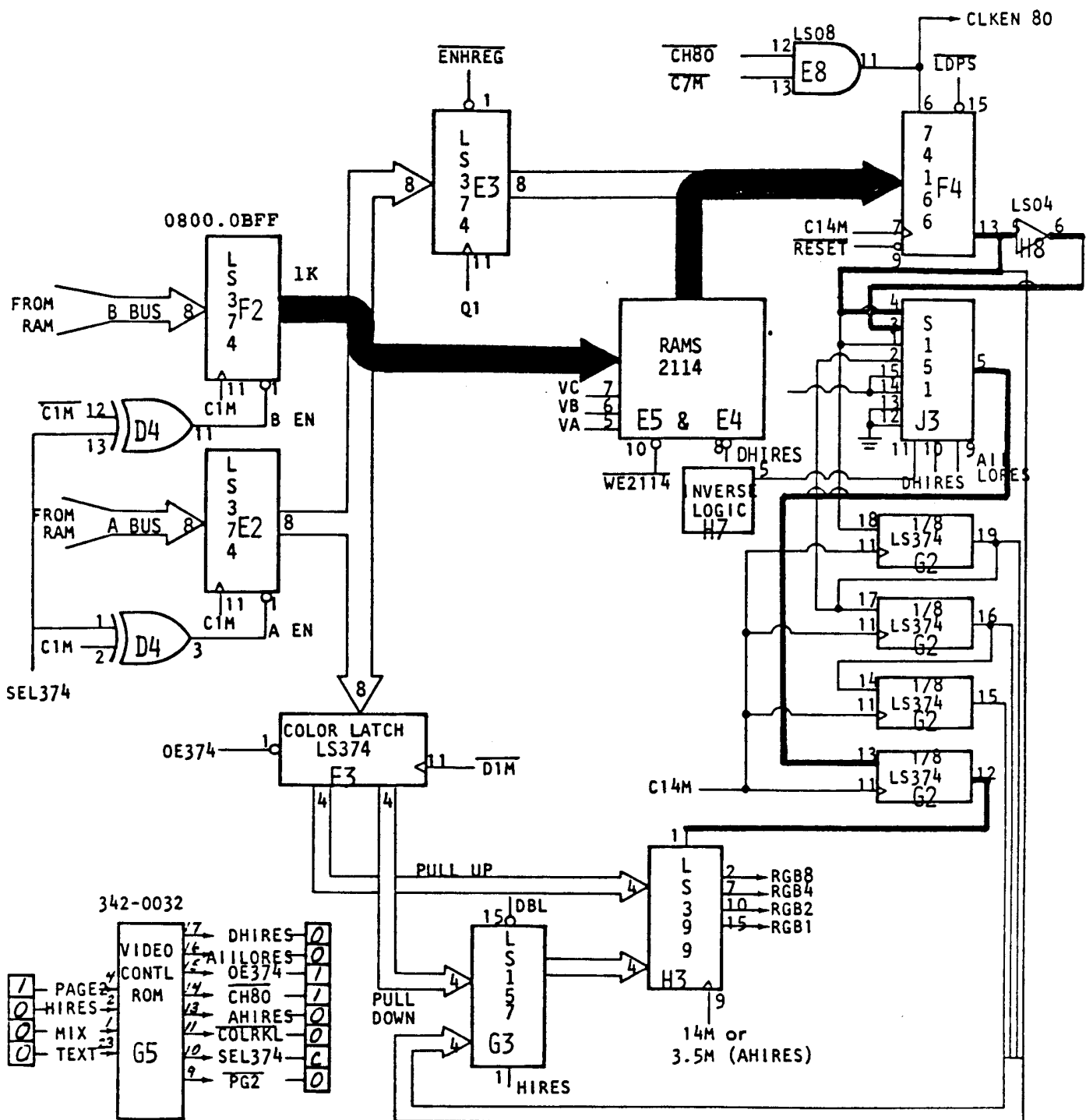
9. COLOR BAR & GRAY SCALE TEST
2000.5FFF (16K)

A/// VIDEO LOGIC DIAGRAM



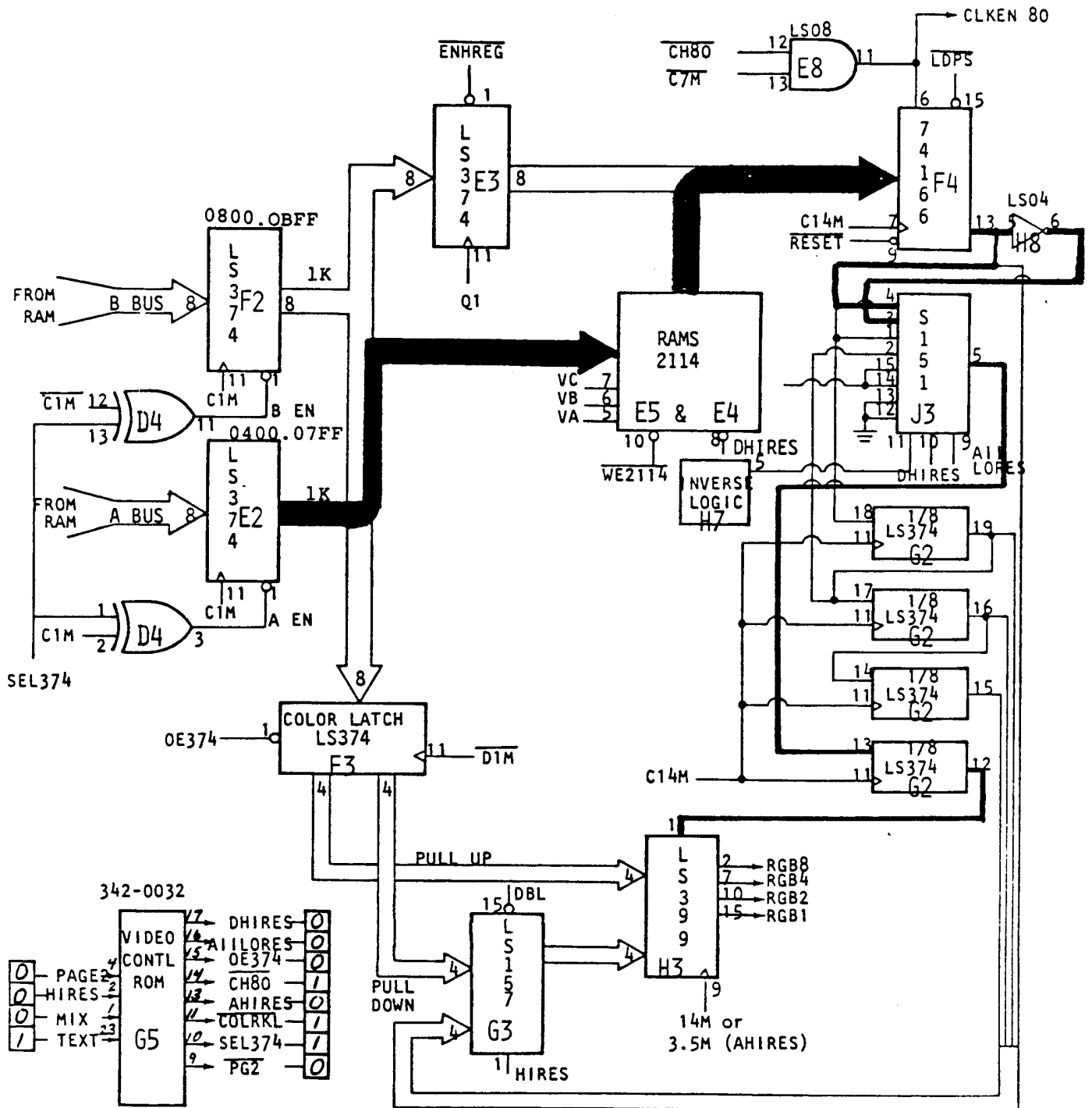
10. APPLE II TEXT MODE PAGE 1 - B & W - 40 COLUMN
0400.07FF (1K)

A/// VIDEO LOGIC DIAGRAM



11. APPLE II TEST MODE PAGE 2 - B & W - 40 COLUMN
0800-0BFF (1K)

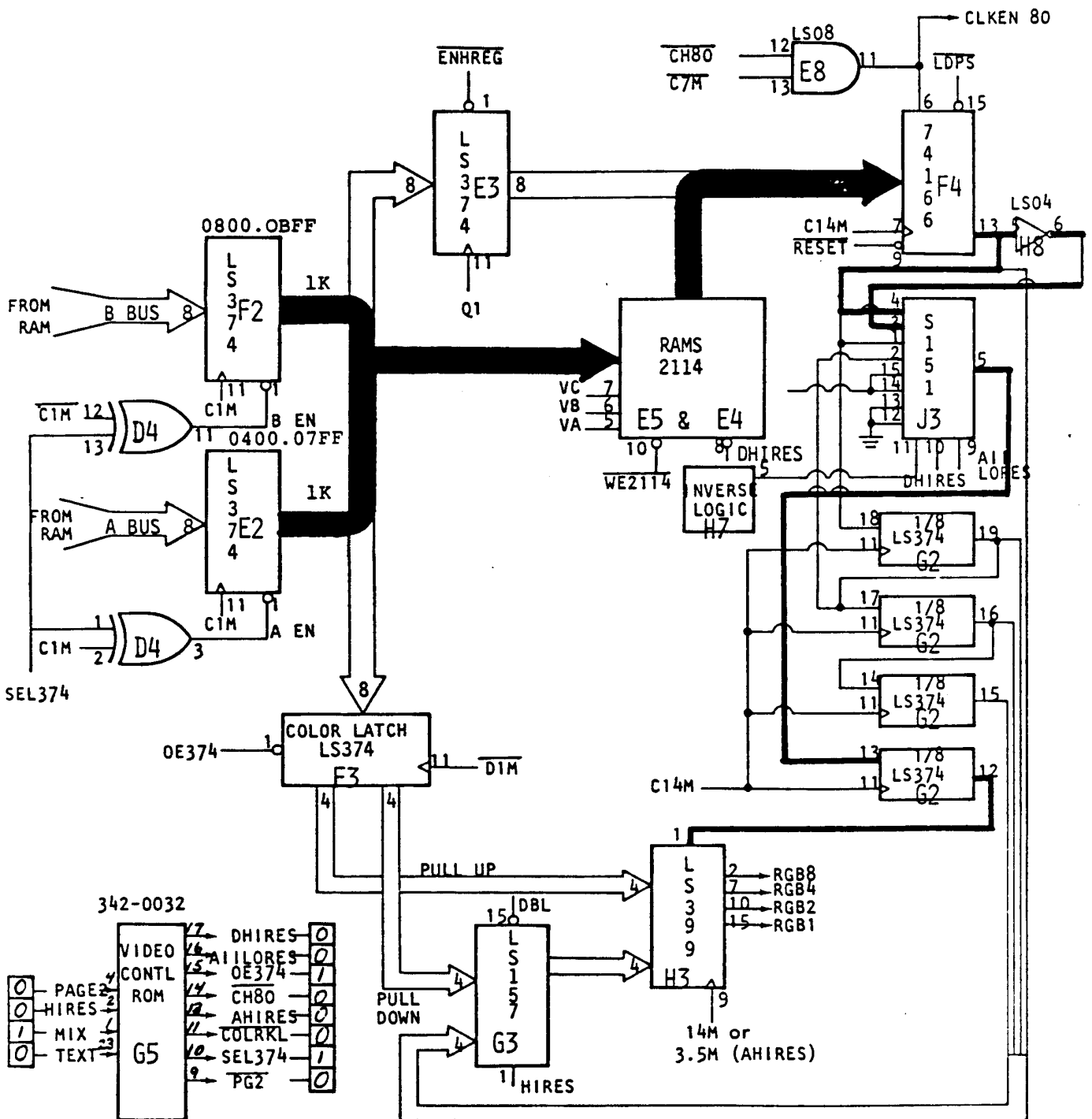
A/// VIDEO LOGIC DIAGRAM



12. SARA 40 COLUMN TEXT MODE TEST- 16 COLORS
0400.0BFF (2K)

6.22

A/// VIDEO LOGIC DIAGRAM



13. SARA 80 COLUMN TEXT MODE TEST - B & W
0400.OBFF (2K)

VIDEO MODES

APPLE III

OUTPUTS
 INPUTS
 ROM G5 (0032)

H6 (9334) pin 7	H6 (9334) pin 5	H6 (9334) pin 4	H6 (9334) pin 6
RDM G5 pin 2	ROM G5 pin 1	ROM G5 pin 23	ROM G5 pin 4
\emptyset 1 \emptyset 1 \emptyset 1 \emptyset 1	(055/057) 052/053	050/051	054/055
HIRES	MIX	GR/TEXT	GPI+2

OUTPUTS FROM ROM G5 (0032)

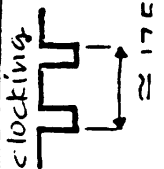
PG 2	SEL 374	COLOR KL	A HIRES	CH 80	OFF 374	A II LORES	V HIRES
G5 pin 9	G5 pin 10	G5 pin 11	G5 pin 13	G5 pin 14	G5 pin 15	G5 pin 16	G5 pin 17
\emptyset	1	\emptyset	\emptyset	1	\emptyset	\emptyset	\emptyset
\emptyset	1	1	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
\emptyset	1	\emptyset	\emptyset	\emptyset	1	\emptyset	\emptyset
\emptyset	1	\emptyset	\emptyset	\emptyset	1	\emptyset	\emptyset
\emptyset	clocking	\emptyset	\emptyset	1	\emptyset	\emptyset	clocking
\emptyset	clocking	1	\emptyset	1	\emptyset	\emptyset	clocking
\emptyset	clocking	\emptyset	\emptyset	\emptyset	1	\emptyset	clocking
\emptyset	clocking	1	1	1	1	1	clocking

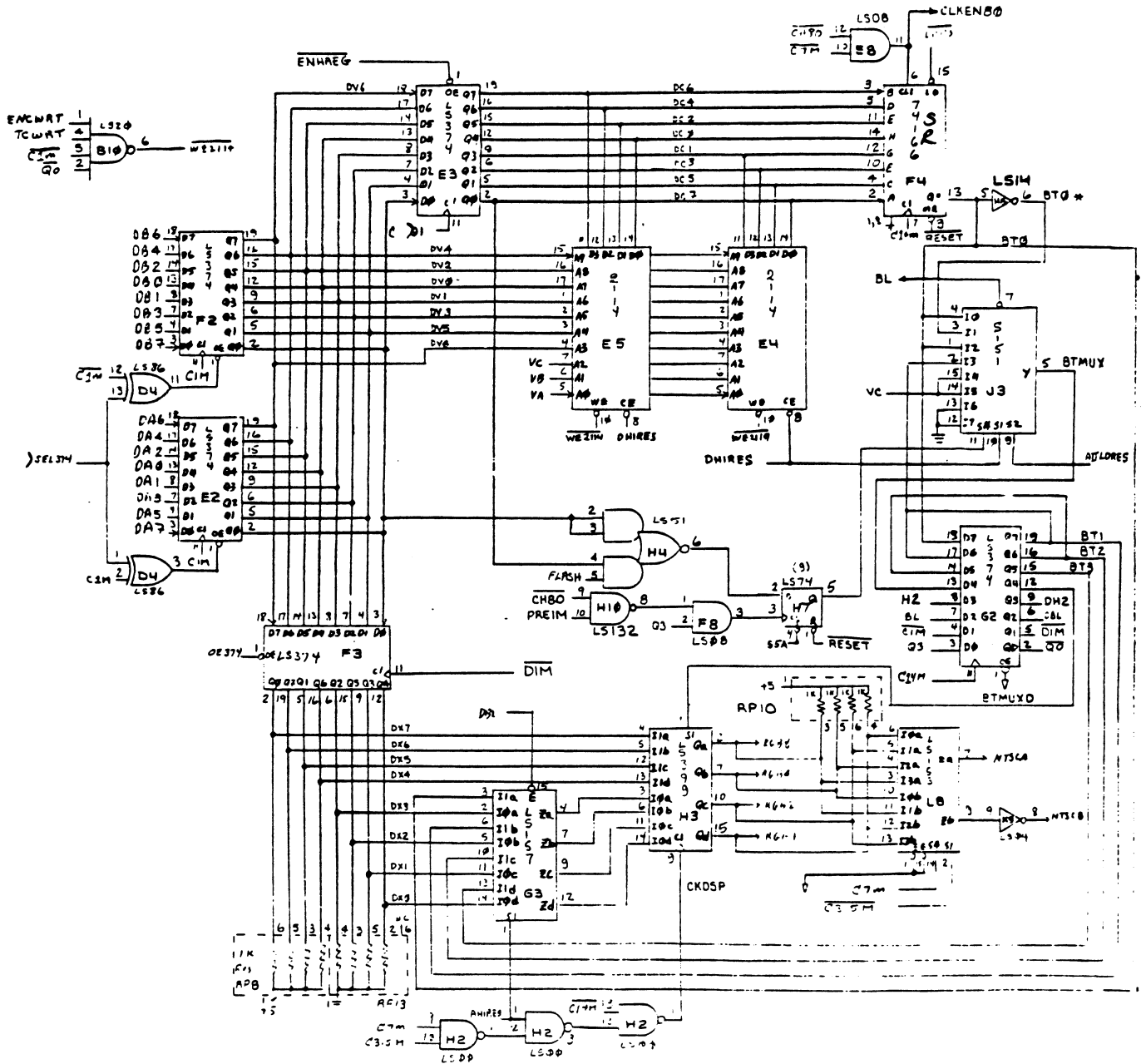
PAGE 1

40 CHAR A II (B/W)	\emptyset	\emptyset	\emptyset	\emptyset
40 CHAR SARA (color)	\emptyset	\emptyset	\emptyset	\emptyset
80 CHAR (B/W)	\emptyset	1	\emptyset	\emptyset
80 CHAR (B/W)	\emptyset	1	\emptyset	\emptyset
A II HIRES (280x192, B/W)	1	\emptyset	\emptyset	\emptyset
FGD/BKGD HIRES (280x192, 16 colors)	1	\emptyset	1	\emptyset
SUPER HIRES (560x192, B/W)	1	1	\emptyset	\emptyset
140 x 192 A HIRES (140x192, color)	1	1	1	\emptyset

PAGE 2

40 CHAR A II (B/W)	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
40 CHAR SARA (color)	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
80 CHAR (B/W)	\emptyset	1	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
80 CHAR (B/W)	\emptyset	1	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
A II HIRES (280x192, B/W)	1	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
FGD/BKGD HIRES (280x192, 16 colors)	1	\emptyset	1	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
SUPER HIRES (560x192, B/W)	1	1	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
140 x 192 A HIRES (140x192, color)	1	1	1	1	1	1	1	1	1





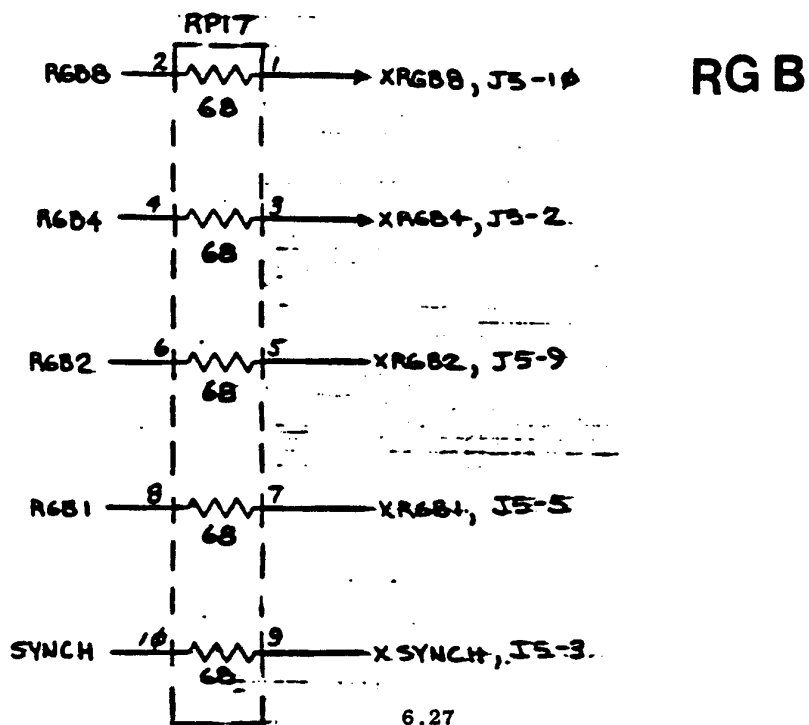
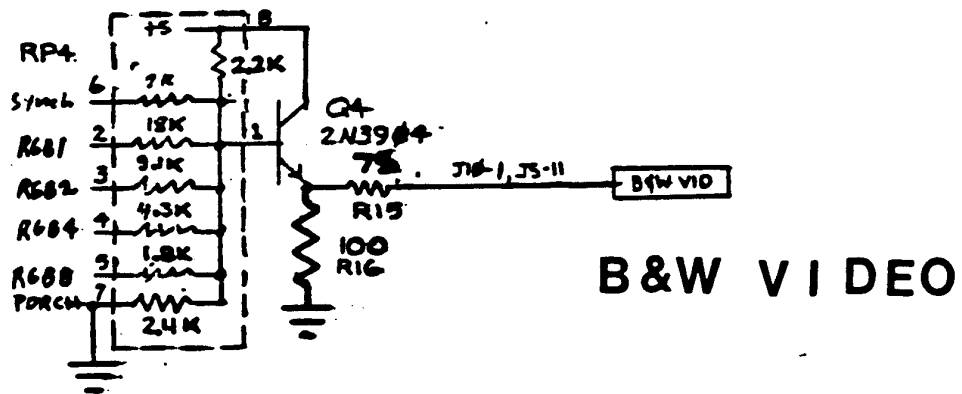
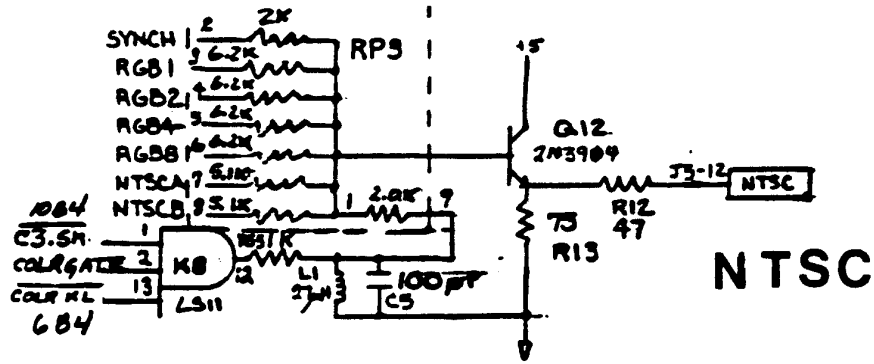
6.25



VIDEO OUTPUTS

- o NTSC COLOR COMPOSITE VIDEO
- o NTSC B/W COMPOSITE VIDEO
- o SYNC
- o FOUR PRIMARY INDEPENDENT VIDEO LINES
- o MIX TO FORM RGB APPLE COLORS
THREE LINES CAN DRIVE TTL RGB MONITOR

VIDEO OUTPUTS



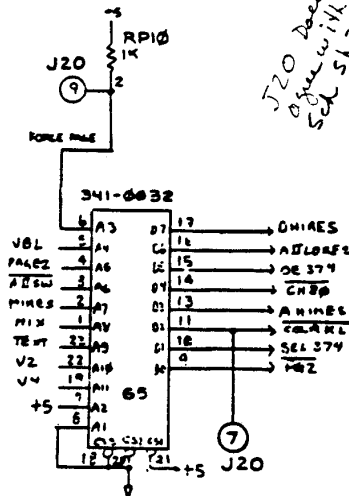


6.28



ROM 341-0032

- A=A
- B=B
- C=S4
- D=VBL
- E=PAGE2
- F=AIISW'
- G=HIRES
- H=MIX
- I=TEXT
- J=V2
- K=V4
- DO=PG2
- D1=SEL374
- D2=COLRKL'
- D3=AHIRES
- D4=CH80'
- D5=OE374
- D6=AILORES
- D7=DHIRES



```

PG2=AIISW'*(HIRES*MIX'*TEXT ')*HIRES*PAGE2*S4*VBL'

SEL374=(VBL'*(AIISW*(PAGE2'*(TEXT +MIX*V2*V4)' +PAGE2*(TEXT +MIX*V2*V4))+
AIISW'*(HIRES*(PAGE2*S4)' +HIRES'*(PAGE2*S4))))'

COLRKL'=(AIISW*TEXT +AIISW'*(HIRES'*(MIX+TEXT ')+HIRES*TEXT '))'

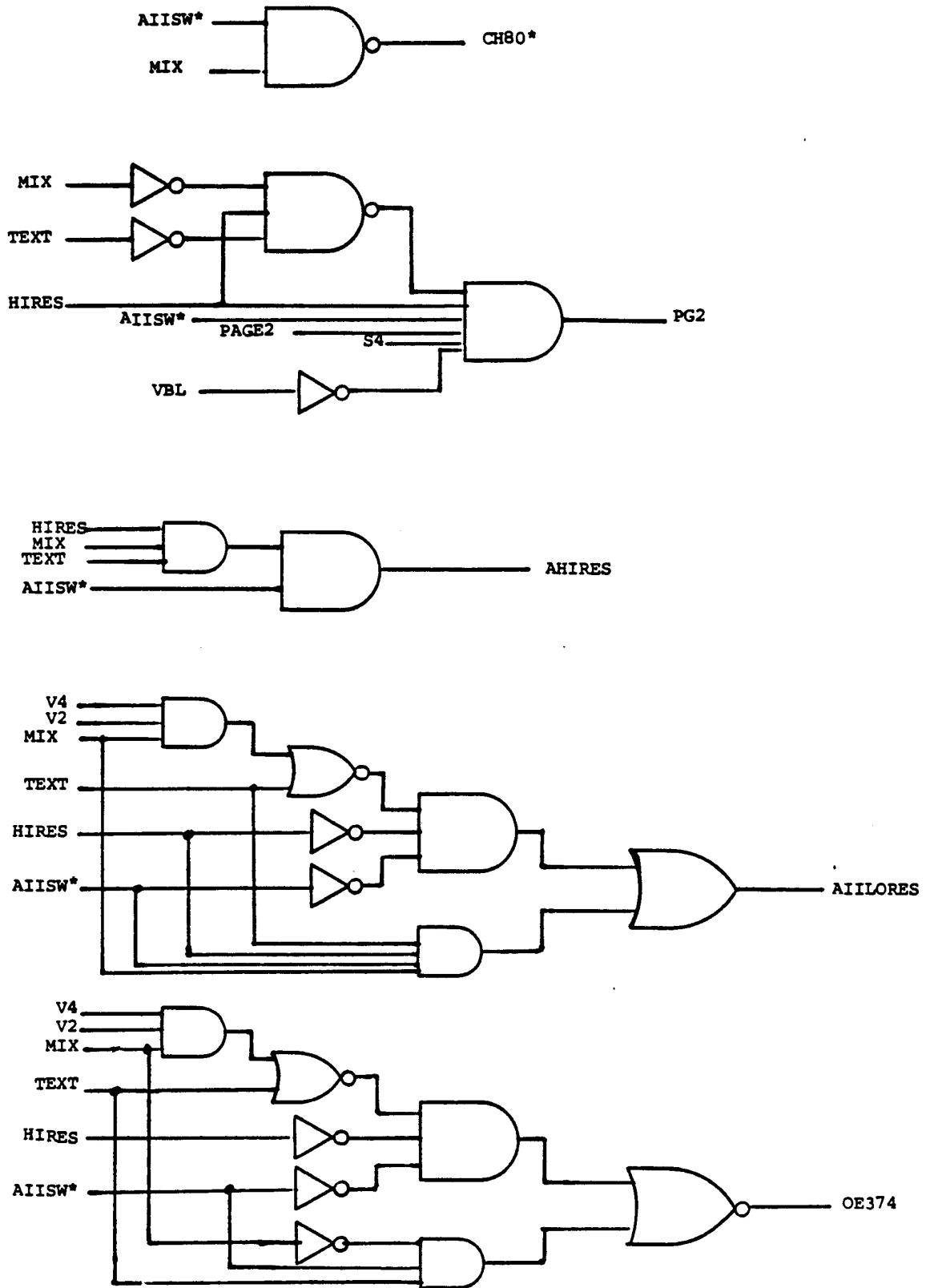
AHIRES=AIISW'*(HIRES*MIX*TEXT )

CH80'=(AIISW'*MIX)'

OE374=(AIISW*HIRES'*(TEXT +MIX*V2*V4)' +AIISW'*MIX'*TEXT )'

AILORES=AIISW*HIRES'*(TEXT +MIX*V2*V4)' +AIISW'*HIRES*MIX*TEXT

DHIRES=(AIISW*HIRES*(TEXT +MIX*V2*V4)' +AIISW'*HIRES)*VBL'
    
```

6.30



THE COLOR VIDEO CONNECTOR

<u>Pin</u>	<u>Name</u>	<u>Description</u>
1	SG	Shield Ground.
2	XRGB4	One of four GRB outputs. This (and pins 5, 9, and 10) is a TTL output with instantaneous color information. A linear weighted sum of these four signals will form a true 16-color RGB video signal.
3	SYNCH	Composite synchronization signal with negative-going tips.
4	PDI	Not used.
5	XRGB1	See pin 2.
6	GND	Power and signal ground.
7	-5V	-5 volt power supply. A device may draw up to 200 ma through this pin.
8	+12V	+12 volt power supply. A device may draw up to 500 ma through this pin.
9	XRGB2	See pin 2.
10	XRGB8	See pin 2.
11	BWVID	Black and white composite video. This is an NTSC composite video signal with negative-going synch tips, 1 volt peak-to-peak into a 75 ohm load. Color information is encoded as a linear grey scale.
12	NTSC	Color composite video. This is an NTSC-compatible video signal with negative-going synch tips, 1 volt peak-to-peak into a 75 ohm load.
13	GND	Power and signal ground.
14	-12V	-12 volt power supply. A device may draw up to 200 ma through this pin.
15	+5V	+5 volt supply. A device may draw up to 1 amp through this pin.

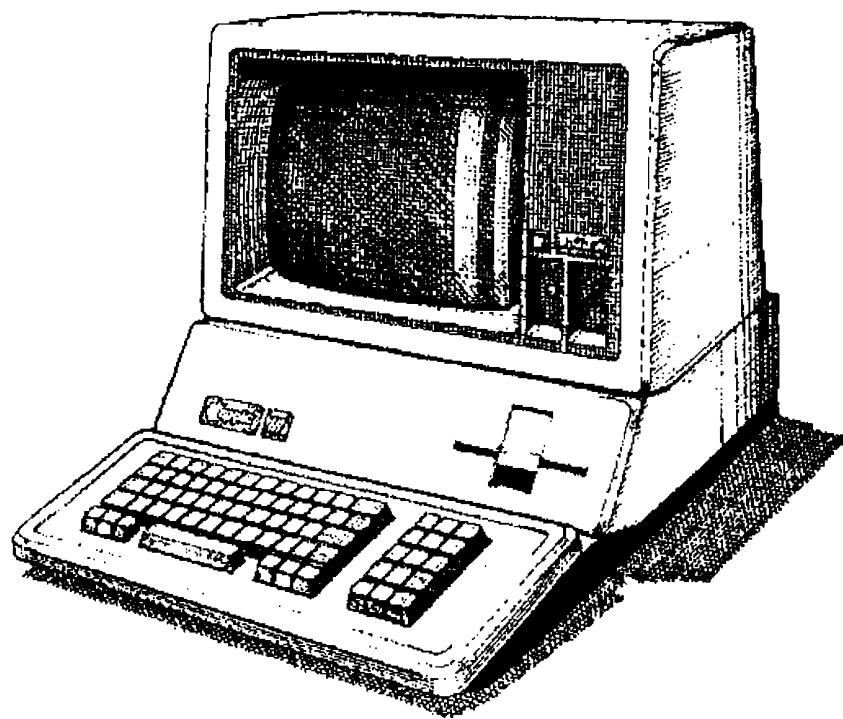
This connector supplies 7 different video signals and 4 power supply voltages. Through this connector you can hook up the Apple to any NTSC color or black and white video monitor. With an additional circuit you can hook up the Apple to a studio-quality RGB color monitor.

All power supply current ratings assume that no peripheral cards are installed in the system. If there are cards in the system, be sure to account for the current drawn by those cards.



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 7 • Input / Output

Written by Apple Computer • 1982

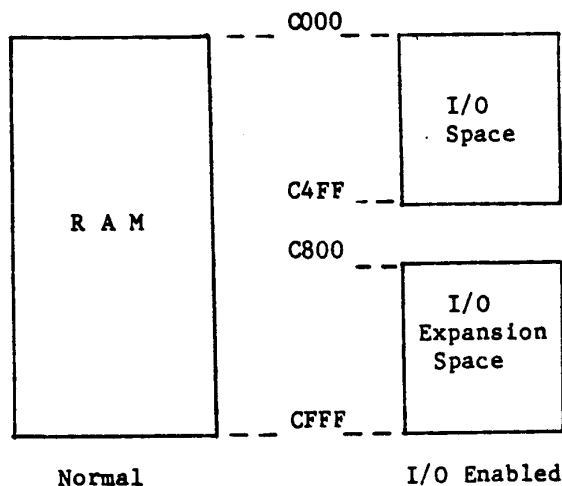


INPUT/OUTPUT (I/O)

DESCRIPTION

In Apple computers I/O devices are treated very much like memory locations. Most of the system's I/O functions are mapped into the address range C000 to CFFF. The I/O space is enabled by a bit in the Environmental Register. Addresses COXX control the on-board I/O. Addresses C1XX, C2XX, C3XX, and C4XX are reserved for the exclusive use of the four I/O slots. The I/O expansion space from C800 to CFFF is switched between the I/O slots. Addresses C500 to C7FF are always Ram, regardless of the setting of the I/O enable bit.

I/O ADDRESS SPACE



INTERFACE CONTROL SIGNALS

For every I/O slot, the Apple /// provides 16 locations that set the Device Select* signal and 256 locations that set the I/O Select* signal.

The Device Select* signal is a signal specific to each slot. It is active for for a 16-address block. This signal is ususally used as an enable signal.

The I/O Select* signal can be used to control a page of memory (256 addresses) which could be placed in ROM, in the interface circuitry, for executing "driver routines". The I/O Select* signal could also be used in circumstances where a small amount of read/write memory for temporary storage is needed. Each I/O slot has its own I/O Select* signal, and each signal is active when a specific page of memory is addressed.

The I/O Strobe is common to all I/O slots. This signal will be low (true) when an address location within the range of C800 to CFFF (2K of memory).



INTERRUPTS

Interface cards that are capable of generating interrupts MUST latch the interrupt output until it is reset by the software. In addition, they MUST include the ability to mask and unmask their interrupt through software, and MUST default to the masked state when the system is reset.

C09X	Slot 1 Device Select
C1XX	Slot 1 I/O Select
COAX	Slot 2 Device Select
C2XX	Slot 2 I/O Select
COBX	Slot 3 Device Select
C3XX	Slot 3 I/O Select
COCX	Slot 4 Device Select
C4XX	Slot 4 I/O Select

The method of accomplishing this transmission between the interface and the computer is called handshaking. In the Apple ///, the handshaking is normally accomplished through the exchange of Device Select*, I/O Select*, IRQ*, and R/W*. The R/W* control signal is used to synchronize the flow of data to and from I/O devices. When the Read/Write* signal is a logic one, the processor is reading information from the data bus. Conversely, when R/W* is low, we are performing a write to the data bus.

As you can see, the handshaking between the Apple /// and the interface is dependent upon the software. Let us again emphasize the role of addressing plays in the I/O process.

ADDRESSING THE I/O

There are certain addresses that you can write to or read from to control the operation of the interface card. Where "n" is the number of the slot where the interface is installed, these hardware addresses are in the range:

$$C080 + n0 \text{ to } C087 + n0$$

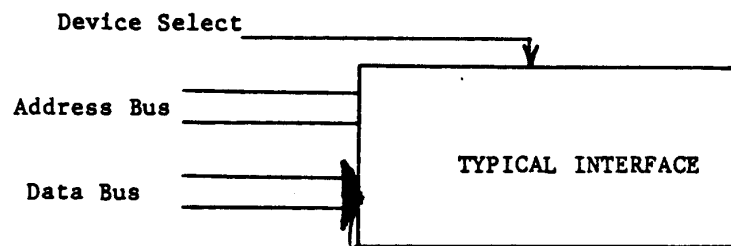
For example, if you install an interface in slot 2, you should write to the addresses from COA0 through COA7.

The operations that the interface card performs are initiated by the read or write operations presented on these hardware addresses.



THE INPUT OPERATION

In the input operation, whenever the correct address is presented to the interface card, the data is present on the data lines D7-D0. For example, if the location C083 + n0 corresponds in the software to an input, the data presented on D7-D0 is accepted by the computer. If the interface card contains a control ROM, the code in ROM is being addressed whenever the I/O Select line is low; that is when the address on the address lines is between Cn00 and CnFF). Recall that "n" is the slot number.



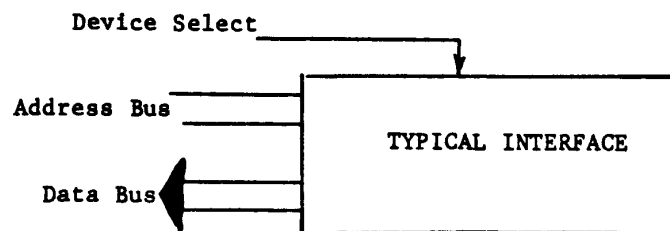
THE OUTPUT OPERATION

In the output operation,

IF the address C081 + n0 represents an output operation in the software

AND the Read/Write* signal is low,

THEN the data is presented on the data bus and latched into the interface whenever the address C081 + n0 is presented.

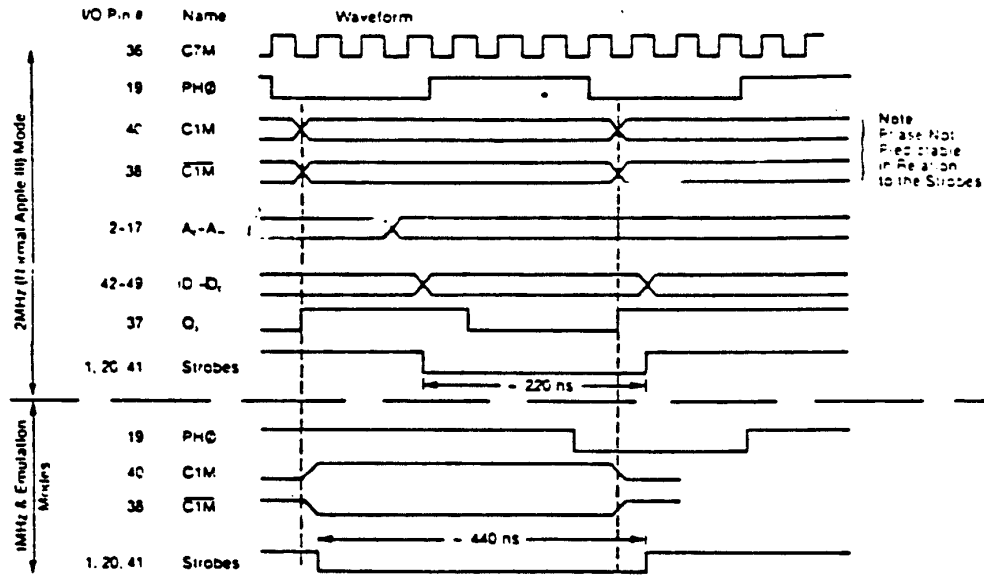


SYSTEM TIMING

A system timing diagram is provided. This diagram shows the timing of some signals at the I/O slots in the 1 MHz and 2 MHz frequency modes.



I/O TIMING DIAGRAMS





THE A/// JOYSTICK

The A/// has two ports designated for joystick (x and y axis paddles) each with two switches. However, unlike the Apple II the ports do not have "annunciator" outputs. One of the switches is a momentary contact and the other is a toggle.

The Analog X and Y inputs are read through a ramp type Analog to Digital converter (A/D). These values derived must be interpreted by the program. The switches are read to the data bus directly through a multiplexor.

The 9708 has multiplexed inputs. To select which input channel is to be read the proper address must be set in an addressable latch and must be held during the PDLEN (ramp start) low cycle.

The I/O address for the setting and clearing of the A/D addresses and the ramp start is as shown in the following table:

I/O Address	A/D Function	Signal Name
C058	A0 Clear	PDLO
C059	A0 Set	
C05A	A2 Clear	PDL2
C05B	A2 Set	
C05C	Ramp Start Clear	PDLEN
C05D	" " Set	
C05E	A1 Clear	AXCO
C05F	A1 Set	

To read the various signals associated with the joysticks the following addresses should be read:

I/O Address	Function
C060,8	Switch 0
C061,9	Switch 1/Margin Switch
C062,A	Switch 2
C063,B	Switch 3/Serial Clock
C066,E	A/D ramp stop (PDL0T)

Note: The joystick port at J3 (Port A) can be configured to be a serial port to support a device like-the Silentye. Care must be taken to insure that the port has been configured for the proper device or signal contention will occur and give erroneous results.

The sequence of operation for the A/D would be as follows:

- 1) set the desired channel address nto A/D 0 through A/D 2.



- 2) start the A/D by cycling the PDLEN signal low for 40 micro seconds then back high
- 3) set up one of the timers to count.
- 4) test for ramp stop
- 5) read the counter
- 6) compute the value of the channel input.

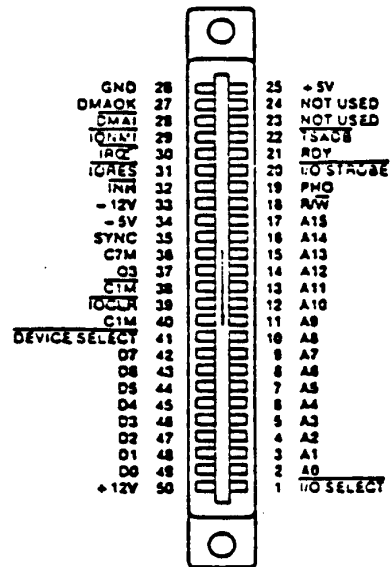


Peripheral Connector Pinout

GND	26	25	+5V
DMAOK	27	24	NOT USED
<u>DMAI</u>	28	23	NOT USED
<u>IONMI</u>	29	22	<u>TSADE</u> (Open collector)
<u>IRQ</u>	30	21	<u>RDY</u> (Open collector)
<u>IORES</u>	31	20	<u>I/O STROBE</u>
<u>INH</u>	32	19	PHO
-12V	33	18	R/W
-5V	34	17	A15
SYNC	35	16	A14
C7M	36	15	A13
Q3	37	14	A12
<u>C1M</u>	38	13	A11
<u>IOCLR</u>	39	12	A10
C1M	40	11	A9
<u>DEV SEL</u>	41	10	A8
D7	42	9	A7
D6	43	8	A6
D5	44	7	A5
D4	45	6	A4
D3	46	5	A3
D2	47	4	A2
D1	48	3	A1
D0	49	2	A0
+12V	50	1	<u>I/O SELECT</u>

A/// Peripheral Connector Slot

TOP VIEW
BACK OF P.C. BOARD



FRONT OF P.C. BOARD



Table 33: Peripheral Connector Signal Description

Pin:	Name:	I/O	Description:
1	<u>I/O SELECT</u>	0	This line, normally high, will become low when the microprocessor references page \$Cn, where n is the individual slot number. This signal become active during PHO (nominally 500ns) and will drive 12 LSTTL loads.
2-17	A0-A15	I, 0	The buffered address bus. The address on these lines becomes valid within 300ns after the beginning of $\overline{C1M}$ and remains valid through PHO. These lines will each drive 8 LSTTL loads.
18	R/ \overline{W}	I, 0	Buffered Read/ \overline{Write} signal. This becomes valid at the same time the address bus does, and goes high during a read cycle and low during a write. This line can drive up to 10 LSTTL loads.
19	PHO	0	A 1 MHz signal which is identical to C1M. This line will drive 5 LSTTL inputs.
20	<u>I/O STROBE</u>	0	This line will go low during C1M when the address bus contains an address between \$C000 and \$CFFF. This line will drive 12 LSTTL loads.
21	RDY	I	The 6502's RDY input. This line should change only during C1M, and when low will halt the microprocessor on the next read cycle. This line has a 1K ohm pullup to +5V. This line should be driven from an open collector output.
22	<u>TSADB</u>	I	A low on this line from the peripheral will cause the address bus to tri-state for Direct Memory Access (DMA) applications. This has a 1 K ohm resistor pullup to +5V. This should be driven from an open collector output.
23		NA	Not used in an Apple /// (NO DAISY CHAINING OF PERIPHERALS!)
24		NA	Not used in an Apple ///.
25	+5V	0	Positive 5-volt supply, 2.0 amps total for all peripheral boards together (but note a limit of 1.5 Watts per board).
26	GND	NA	System circuit ground. 0 volt line from power supply. Do not use for shield ground.
27	DMAOK	0	Acknowledge signal to the peripheral following

its request for the special Direct Memory Access (DMA) mode. Informs the peripheral that the DMA can now proceed.

- | | | | |
|----|----------------------|----------|--|
| 28 | <u>DMAI</u> | <i>I</i> | Direct Memory Access (DMA) interrupt. Requests the A Apple /// DMA mode. Has a 1 K ohm pullup to +5. This should be driven from an open collector output. |
| 29 | <u>IONMI</u> | <i>I</i> | Input/Output Non-Maskable Interrupt. This is equivalent to the IORES (pin 31) line as it will execute the same code in the Autostart ROM. This line should be driven by an open collector output. <i>THE NMI DOES NOT GO DIRECTLY TO THE PROCESSOR, SO IT CAN BE MASKED BY THE SYSTEM RESET FUNCTION.</i> |
| 30 | <u>IRQ</u> | <i>I</i> | This line is ignored in Apple][emulation mode. It should be driven by a TTL output. |
| 31 | <u>IORES</u> | <i>O</i> | Input/Output Reset signal used to reset the peripheral devices. Pulled low by a power on or RESET key. This line will drive 12 LSTTL loads. |
| 32 | <u>INH</u> | <i>I</i> | Inhibit line. When a device pulls this line low, all system memory is disabled. This line has a 1 K ohm pullup resistor to +5V and should be driven from an open collector output. |
| 33 | -12V | <i>O</i> | Negative 12 volt supply, 200mA total for <u>all</u> peripheral boards together. |
| 34 | -5V | <i>O</i> | Negative 5 volt supply, 200mA total for <u>all</u> peripheral boards together. |
| 35 | SYNC | <i>O</i> | The 6502 opcode synchronization signal. Can be used for external bus control signals. Will drive 10 LSTTL loads. |
| 36 | C7M | <i>O</i> | Seven MHz high frequency clock. Will drive 10 LSTTL loads. |
| 37 | Q3 | <i>O</i> | A 2MHz (nonsymmetrical) general purpose timing signal. Will drive 10 LSTTL inputs. |
| 38 | <u>CIM</u> | <i>O</i> | Complement of CIM clock. This will drive 12 LSTTL loads. |
| 39 | <u>IOCLR</u> | <i>O</i> | Provides the \$C800 space disable function directly without address decoding (\$CFFF is used for Apple][peripherals. It is addressed from \$C02x. This line will drive 12 LSTTL loads. |
| 40 | <u>CIM</u> | <i>O</i> | Phase CIM clock. This is the same as the microprocessor's 1 MHz clock. This will drive 12 LSTTL loads. |
| 41 | <u>DEVICE SELECT</u> | | This line becomes active (low) on each peripheral |



connector when the address bus is holding address between \$COn0 and \$COnF where n is the slot number plus \$8. This line will drive 12 LSTTL loads.

42-49	D7-D0	<i>I,0</i>	<p>The 8-bit system data bus. During a write cycle, data is set up by the 6502 less than 300ns after the beginning of $\overline{C1M}$. During a read cycle the 6502 expects data to be ready no less than 100ns before the end of $\overline{C1M}$. These lines will drive 8 LSTTL inputs.</p>
50	+12V	<i>0</i>	<p>Positive 12 volt supply[*], 300mA total for <u>all</u> peripheral boards together.</p>

** NOTE: TOTAL POWER DRAWN BY ANY ONE PERIPHERAL BOARD IS NOT TO EXCEED 1.5 WATTS*



The Real Time Clock/Calendar

A real time clock has been incorporated into the A /// using 58167 CMOS Clock/Calendar chip. This chip has the resolution to count to thousandths of a second. The clock circuitry can be set to cause an interrupt at intervals from a tenth of a second to interrupts every month.

Since the clock is a CMOS circuit it consumes about 10ua in the standby (power off in the Apple ///) mode. This is about the same as a normal LCD watch. Three "AA" alkaline batteries are mounted in a battery pack that clips to the casing near the internal speaker. Wires attached to the battery holder connect to a 2 pin molex connector at location G13.

This clock chip is not a member of the 6500 family and is not directly compatible. Special considerations have been incorporated into the logic design to allow the Apple III to access and control of the clock chip.

The timing requirements for the clock chip require that the address lines be latched for much longer than the processor can accomplish in normal operations, so the clock is addressed with the "Zero Page Register" (the B port of VIA B6). The operating system will temporarily store the current zero page address at another location then write the desired clock address into the zpage register. The processor clock, PH0, is extended to μ sec by the action of the prom 180 and associated circuit. The clock chip also requires a separate read and write strobe so appropriate logic was designed to split the R/W signal into a read and write strobe.

When the processor has completed its access to the clock it will return the proper zero page address to the VIA and PH0 will return to its normal operation.

Please refer to the specification sheet in the Appendices for complete details of the clock.

The transistor array performs two functions. One it supplies Vcc from the power supply when the Apple /// is "on", and develops a power down strobe to the clock chip to set its standby mode just before the supply fully decays.

The clock may be programmed that while it is in the standby mode to provide a PDINT to an external device which may restore power to the Apple to service a particular device. This feature would be very useful in communications networks that poll at specific times in off-hours. However, at the time of this printing no such remote device has been specified.

The clock runs on a 32KHZ crystal may be adjusted to an operating tolerance of 5 minutes a month. There are two methods used, one is a verifications of operation using software, this however has the accuracy of approximately 5 minutes a month, which for many applications and users is close enough, but for those users who demand a closer setting a method of setting the clock using an acoustic probe and frequently meter is available. The only problem with this is the cost of the calibration equipment (nearly \$1000 per station). Level II centers will most likely be equipped with these devices.

It should be noted that there is a slight shift if frequently between power on



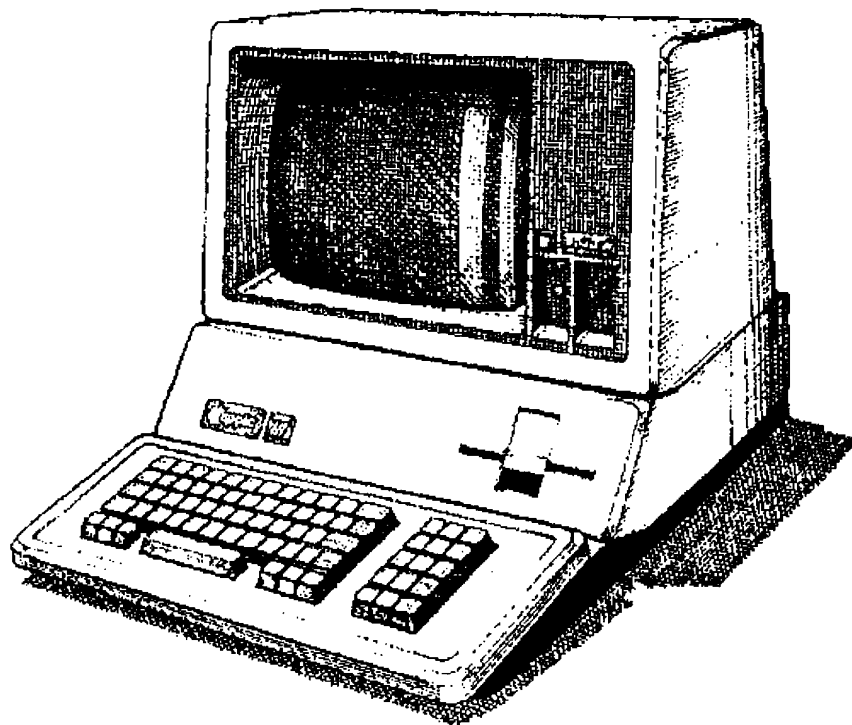
and standby modes. Depending on the actual usage of power on and off the clock may vary perceptibly over the course of a month. So it is best to describe the entire function as a clock, not a chronograph.

.



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 8 • The Keyboard

Written by Apple Computer • 1982



THE KEYBOARD

The Apple /// has a built-in 74 key typewriter-like keyboard which includes full alpha/numerics, four cursor control keys, two special function keys, and a numeric keypad. It has full upper and lower case ASCII code generation capability as well as full incorporation of Apple][functions.

The drawing on the previous page shows the standard keyboard legend and details the keystation number. Note that in addition to the 74 keys there is a recessed Reset key. Every key on the keyboard can be observed individually by the software. The Control and Shift keys modify the key codes when presented to the system.

The keyboard is electrically connected to the main circuit board by a 26 conductor ribbon cable. The cable plugs into a socket on the keyboard and the main circuit board. The signal assignment is shown on the Pin Signal Assignment table.

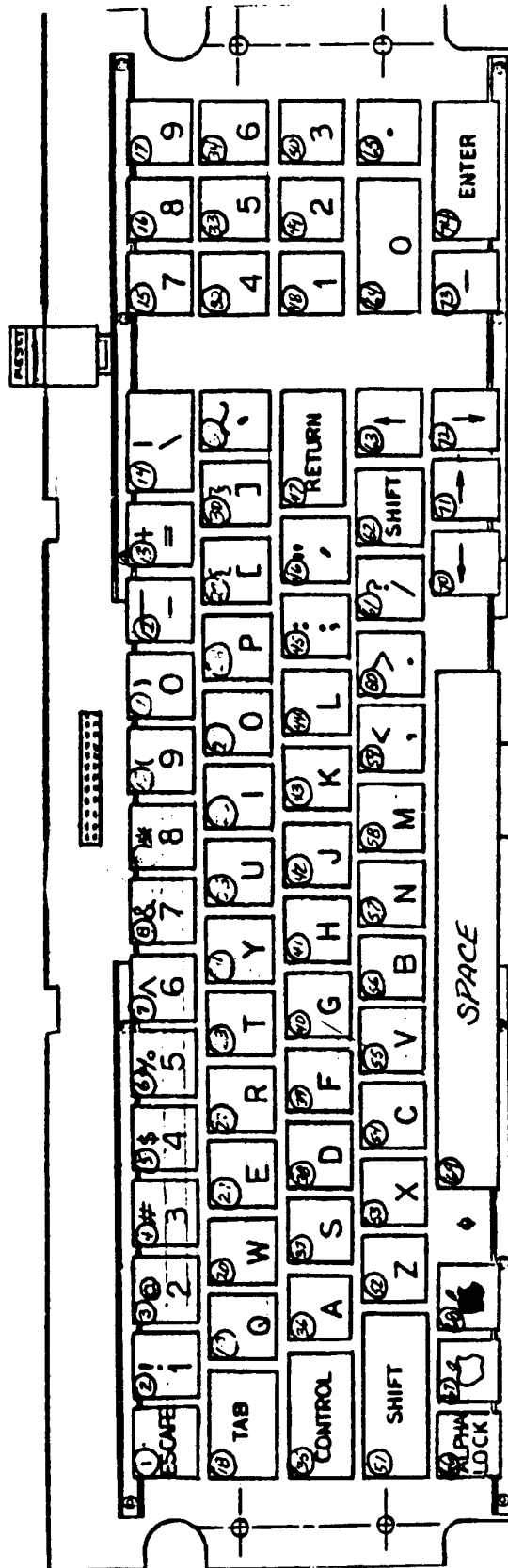
Repeat Functions

Any key held down for more than 1/2 second is automatically activated to repeat at a 10 CPS rate. A high speed (30 CPS) repeat function is activated by holding down the closed Apple key (Key #68) after depressing and holding the key to be repeated. An idiosyncrasy of the Apple /// is that if the closed Apple key is depressed before another key, it is displayed as only one character. If it is depressed after another key, the high speed repeat is activated.

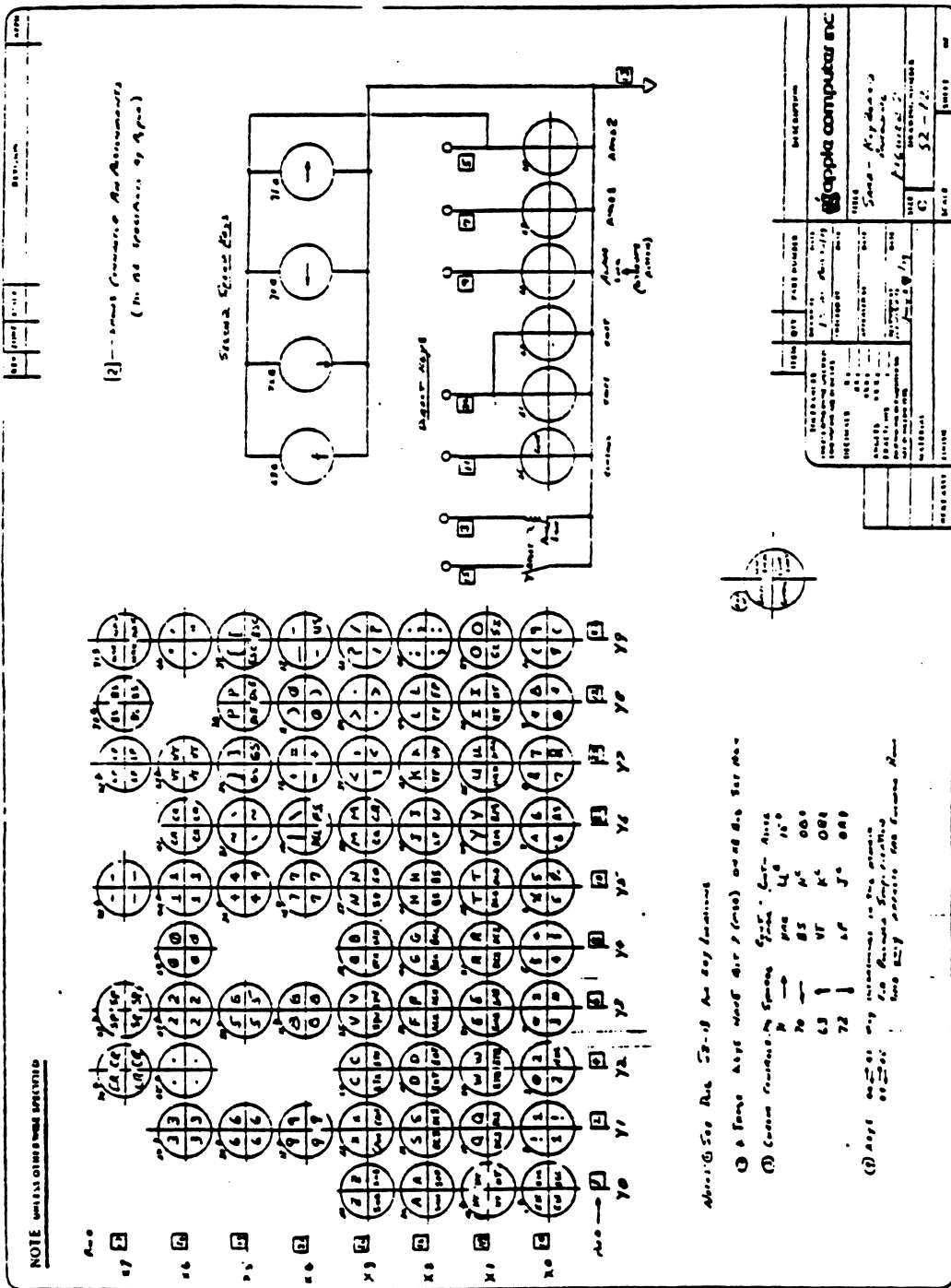
The four cursor control keys (63, 70, 71, 72) are two-contact keys. This means that as the key is partially depressed, it makes its first contact generating a signal code. When it is fully depressed, it will make a second contact, automatically activating a high speed repeat of that key.

READING THE KEYBOARD

The keyboard can be thought of as two hardware ports (busses) that can provide two distinct types of data. The first type is ASCII, which is addressed by Memory Address C000; we will call this the KA port. The KA port always contains the lower 7 bits of the ASCII code and, like the Apple][, uses the MSB as a "keyboard data ready" flag. The second type of data is addressed by Memory Address C008; we call this the KB port. The KB port looks at the "direct connect" keys and at the eighth bit of the key code. A summary of the bit meanings for these two types of data is shown in the table at the top of the following page.



8.2





KEYBOARD ENCODER MATRIX

	KEYBOARD		VERTICAL Y				KEY PAD		
Y0	A	Z	ESC	TAB					NONE
Y1	1	Q	S	X					9 6 3
Y2	2	W	D	C					. ENTER
Y3	3	E	F	V	SPACE				8 5 2
Y4	4	R	H	B					∅
Y5	5	T	G	N					7 4 1 -
Y6	6		Y	~	J	M	RETURN		NONE
Y7	7	+ =	U	}]	K	< ,	↑ ↓		NONE
Y8	8	∅	I	P	; :	—	> .		NONE
Y9		9	§ [O	" ' L		? /		NONE

	KEYBOARD		HORIZONTAL X								KEY PAD		
X0	ESC		1	2	3	4	5	6	7	8	9		NONE
X1	TAB		Q	W	E	R	T	Y	U	I	O		NONE
X2	A S		D	F	G	H	J	K	L	:	;		NONE
X3	Z X		C	V	B	N	M	< ,	> .	? /			NONE
X4	∅	=	+ =										7 8 9
X5	P	{ }	~										4 5 6
X6	" ' RETURN		↑										1 2 3 . ∅
X7	SPACE		← → ↓										- ENTER



MEMORY ADDRESS REFERENCE

KA PORT (C000)		KA PORT (C008)	
Bit 0	ASCII Bit 0	* Bit 0	"1"="any key down"
Bit 1	ASCII Bit 1	* Bit 1	"0"="shift depressed"
Bit 2	ASCII Bit 2	* Bit 2	"0"="control depressed"
Bit 3	ASCII Bit 3	* Bit 3	"0"="alpha lock set"
Bit 4	ASCII Bit 4	* Bit 4	"0"="Apple 1 switch depressed"
Bit 5	ASCII Bit 5	* Bit 5	"0"="Apple 2 switch depressed"
Bit 6	ASCII Bit 6	* Bit 6	"1"="start up uncommitted mode"
Bit 7	"1"="data ready flag"	* Bit 7	ASCII Bit 7

The KA data is used exactly like that in the Apple][keyboard. The KB data is provided for function expansion. The KB ports 1 to 5 are direct mechanical connections to defined function switches. Bit 0 is an output from the encoder circuitry and bit 7 is the eighth bit of the key code. Bit 6 is a special bit, a flag used during turn-on to show that the operational mode (Apple /// or Apple][) has not yet been determined..

It should be noted that the Reset key cannot act on its own but has to be depressed with another key. This is a safety feature to prevent blowing away a good night's programming effort. Now isn't that nifty!? A CONTROL-RESET will give a true system reset. However, it cannot be used for recovery from Apple][mode. The CONTROL-RESET will also give the system an NMI (Non-Maskable Interrupt). This provides Apple /// with two levels of "reset."



KEYBOARD CODES

A complete list of the codes generated by the encoder circuitry is presented in the following table:

Table: Apple /// KEYBOARD CODES (HEX)

Key #	Key Name	US	SH	CT	SU-CT	Key #	Key Name	US	SH	CT	SU-CT
1*	ESCAPE	9B	9B	9B	9B	39	F	46	46	04	04
2	1	31	21	31	31	40	G	47	47	07	07
3	2	32	40	32	00	41	H	48	48	08	08
4	3	33	23	33	23	42	J	4A	4A	0A	0A
5	4	34	24	34	24	43	K	4B	4B	0B	0B
6	5	35	25	35	25	44	L	4C	4C	0C	0C
7	6	36	5E	36	53	45	;	3B	3A	3B	3A
8	7	37	26	37	26	46	'	27	22	27	22
9	8	38	2A	38	2A	47	RETURN	0D	0D	0D	0D
10	9	39	28	39	28	48*	1	B1	B1	B1	B1
11	0	30	29	30	29	49*	2	B2	B2	B2	B2
12	-	2D	5F	2D	1F	50*	3	B3	B3	B3	B3
13	=	3D	2B	3D	2B	51	SHIFT	-----KB-1-----			
14	BACKLASH	5C	7C	7F	1C	52	Z	5A	5A	1A	1A
15*	7	B7	B7	B7	B7	53	X	58	58	18	18
16*	8	B8	B8	B8	B8	54	C	43	43	03	03
17*	9	B9	B9	B9	B9	55	V	56	56	16	16
18*	TAB	89	89	89	89	56	B	42	42	02	02
19	Q	51	51	11	11	57	N	4E	4E	0E	0E
20	W	57	57	17	17	58	M	4D	4D	0D	0D
21	E	45	45	05	05	59	,	2C	3C	2C	3C
22	R	52	52	12	12	60	.	2E	3E	2E	3E
23	T	54	54	14	14	61	/	2F	3F	2F	3F
24	Y	59	59	19	19	62	SHIFT	-----KB-1-----			
25	U	55	55	15	15	63*	UP-CURSOR	8B	8B	8B	8B
26	I	49	49	09	09	64*	O	BO	BO	BO	BO
27	O	4F	4F	0F	0F	65*	.	AE	AE	AE	AE
28	P	50	50	10	10	66	ALPHA-LK	-----KB-3-----			
29	RT-BRACK	5B	7B	1B	1B	67	APPLE 1	-----KB-4-----			
30	LT-BRACK	5D	7D	1D	1D	68	APPLE 2	-----KB-5-----			
31	LT-BRACK	60	7E	60	7E	69*	SPACE	A0	A0	A0	A0
32*	4	B4	B4	B4	B4	70*	LT-CURSOR	8B	8B	8B	8B
33*	5	B5	B5	B5	B5	71*	RT-CURSOR	95	95	95	95
34*	6	B6	B6	B6	B6	72*	DN-CURSOR	8A	8A	8A	8A
35	CONTROL	-----KB-2-----				73*	-	AD	AD	AD	AD
36	A	41	41	01	01	74*	ENTER	8D	8D	8D	8D
37	S	53	53	13	13						
38	D	44	44	04	04						

* Bit 7 (MSB) on these keys appears on bit 7 of KB port, on the KA port.

Note: the keys on the numeric keypad have only one code. Shift and Control



have no effect on these keys.

THE APPLE II EMULATION MODE

In this mode the Apple /// special functions are locked out, making the keyboard look exactly like the Apple][. Thus, the Apple][software does not look at the KB port and must get all the Apple][codes for the KA port. This is the reason for coding and bit arrangements. However, the Apple /// functions are not really locked out and could be read by an enterprising programmer, if desired.

Some of the keyboard codings which should be noted because of the Apple II emulation mode are:

1. "NUL" is a control-@ (Control-Shift-2). With the Apple][, the "NUL" is a Control-Shift-p.
2. "RS", record separator, is a control-Shift-6, which corresponds to control-Shift-n in the Apple][.
3. The Shift-m, for a left square bracket in the japple][, is not available in the emulation mode since the character is represented on the keyboard. The "GS", group separator, is a Control-left bracket rather than the Control-Shift-m.
4. "BS", backspace, has been retained for the left arrow and "NAK", negative acknowledgment, for the right arrow for both the Apple][and Apple /// modes.
5. "VT", vertical tab, and "LF", line feed, were chosen for the up and down cursor keys. In the Apple][mode these will not give a cursor movement (unless the operating system is changed) but will give the Control-k and Control-j codes. This could cause some slight confusion for those Apple][programs that use those codes (...now he tells me!).
6. The autorepeat and high speed repeat functions will work for the Apple][just like they do in the Apple /// mode. Nice!

ELECTRONIC CIRCUIT DESCRIPTION

Please refer to sheet 9 of 10 of the Schematic (Drawing Number 050-0039) for the following Keyboard Logic circuit description.

The Apple /// keyboard is simply an 8 by 10 X,Y matrix which is scanned by the encoder circuit [keyboard encoder rom H14] on the main logic board. All keys are scanned with the exception of five keys [shift, control, capslock, Apple1, Apple2] that are direct connected. The second contacts of the cursor keys (high speed repeat function are OR'd wired into the Apple2 switch line on KB-5.



On the main board the encoder scans the keyboard matrix and provides the correct code outputs plus a strobe and an "any key down" signal. A diagram for the key matrix shows the key locations, and their ASCII character representations are shown on the following page. The special function keys can be detected separately from the standard control keys by observing that the MSB of the KB port is set high.

The A3 signal to the Tri-state data selectors (LS257's) selects whether the output of the LS257 will be a KA or KB port. If A3 is high, selected by memory address C008, the KB port is selected. The KBD line enable the reading of data off the keyboard.

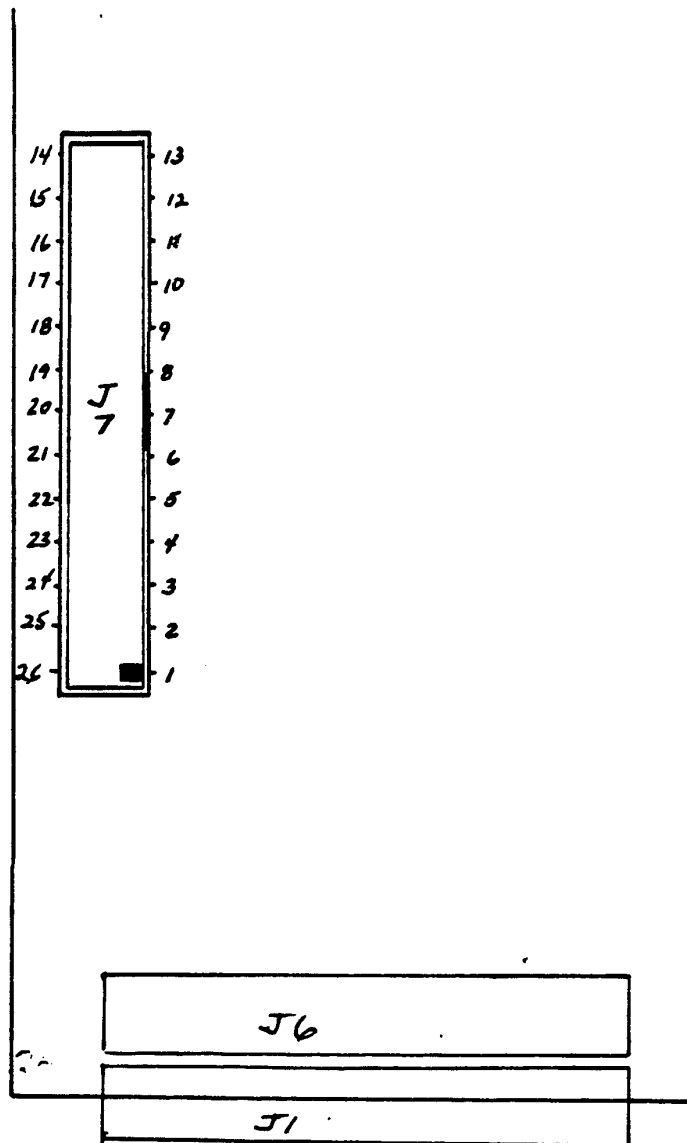
The Repeat Function: The normal repeat function (10 cps) that occurs when a key is held down is the result of clocking and resetting the flip-flop H11 (feeding into H12). This is accomplished by the AK (any key) and DTRDY (data ready) setting, H11 then having CLRSTB (clear strobe) resetting the flip-flop. The Apple2 key, when depressed after a character key, engages the high-speed repeat function. The combination of the Apple2 key signal clocking the edge triggered flip-flop (H11), and pulse change to the inputs of the 556 (L10) dual timers speeds up the timing.

The Reset Function: The power on reset is provided by the one shot (A5). Depressing the reset key results in a soft reset. This causes the KRESET line to go low and enable the LS139 (J11). If the Control key and the Reset key are both depressed, a hard reset results. This hard reset can be foiled through sophisticated programming. The RESETLK (reset lock signal) provided from the Environmental Register [6522 - B6], can disable the Reset and NMI. (Try it!)

Keyboard Light: The keyboard light indicates the VCC is provided to the keyboard. If no light is observed, check Q9 [MPV 51].

PIN SIGNAL ASSIGNMENT

Pin #	Description
1	Y0
2	Y1
3	Power Light
4	Y2
5	Apple 2 (high speed repeat)
6	Y3
7	Apple 1
8	Y4
9	Alpha Lock (alternate action)
10	Y5
11	Control
12	Y8
13	Signal Ground
14	X0
15	Reset
16	X2
17	X7
18	X2
19	X5
20	X3
21	X4
22	Y9
23	Y6
24	Shift (both keys)
25	Y7
26	X6



8.9

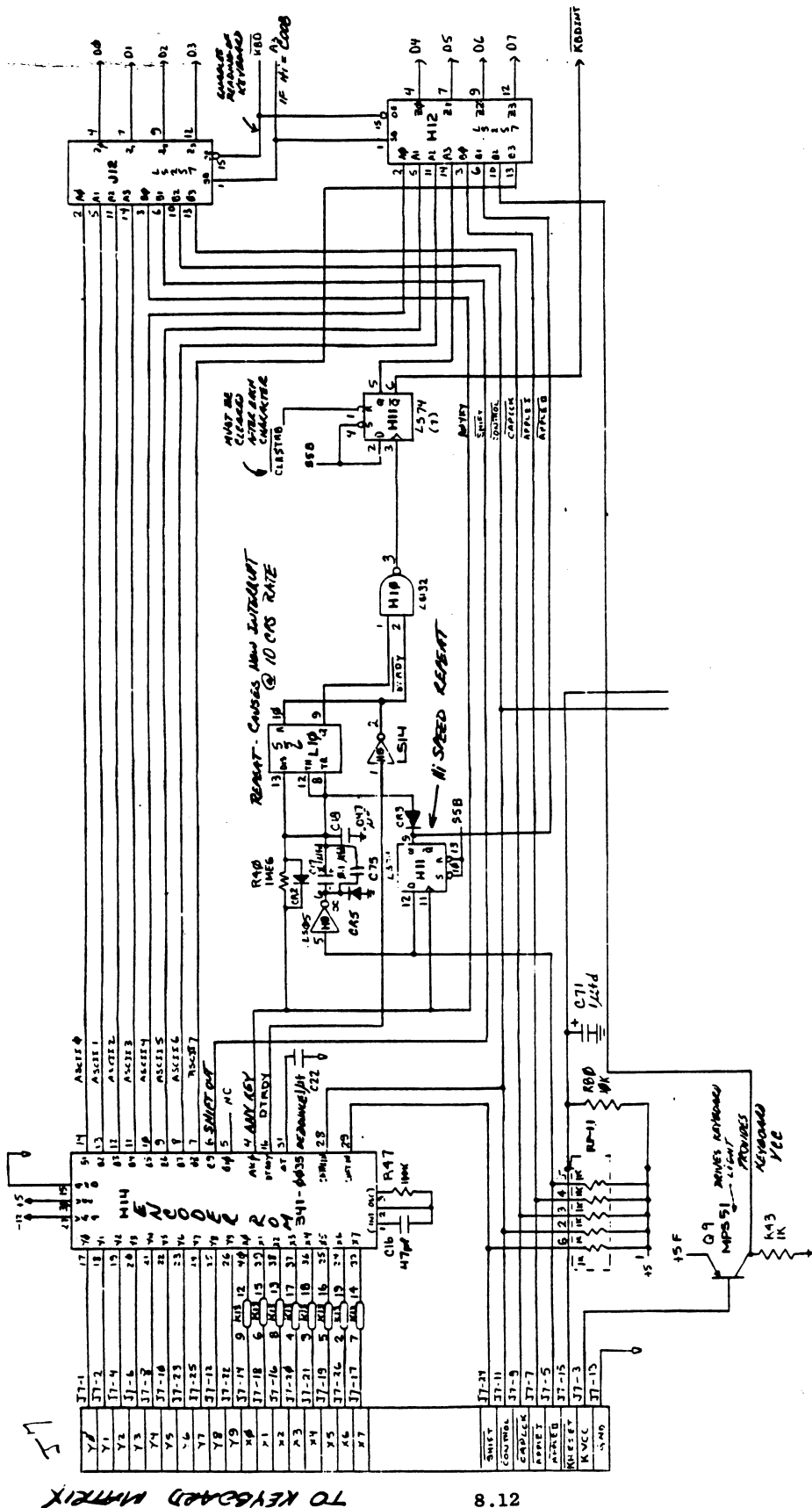
\$ = REPRESENTS HEXADECIMAL

Table 2: Keys and their Associated ASCII Codes (Bit 7 always set)

Key	Alone	CONTROL	SHIFT	Both
<space>	\$A0	\$A0	\$A0	\$A0
ESCAPE	\$9B	\$9B	\$9B	\$9B
1!	\$B1	\$B1	\$A1	\$A1
2@	\$B2	\$B2	\$C0	\$80
3#	\$B3	\$B3	\$A3	\$A3
4\$	\$B4	\$B4	\$A4	\$A4
5%	\$B5	\$B5	\$A5	\$A5
6^	\$B6	\$B6	\$DE	\$9E
7&	\$B7	\$B7	\$A6	\$A6
8*	\$B8	\$B8	\$AA	\$AA
9(\$B9	\$B9	\$A8	\$A8
0)	\$B0	\$B0	\$A9	\$A9
-	\$AD	\$AD	\$DF	\$9F
=+	\$BD	\$BD	\$AB	\$AB
\	\$DC	\$9C	\$FC	\$FF
TAB	\$89	\$89	\$89	\$89
{	\$DB	\$9B	\$FB	\$9B
}	\$DD	\$9D	\$FD	\$9D
"	\$A7	\$A7	\$A2	\$A2
RETURN	\$8D	\$8D	\$8D	\$8D
,<	\$AC	\$AC	\$BC	\$BC
.>	\$AE	\$AE	\$BE	\$BE
/?	\$AF	\$AF	\$BF	\$BF
<left arrow>	\$88	\$88	\$88	\$88
<right arrow>	\$95	\$95	\$95	\$95
<up arrow>	\$8B	\$8B	\$8B	\$8B
<down arrow>	\$8A	\$8A	\$8A	\$8A
.	\$AE	\$AE	\$AE	\$AE
-	\$AD	\$AD	\$AD	\$AD
ENTER	\$8D	\$8D	\$8D	\$8D
A	\$C1	\$81	\$C1	\$81
B	\$C2	\$82	\$C2	\$82
C	\$C3	\$83	\$C3	\$83
D	\$C4	\$84	\$C4	\$84
E	\$C5	\$85	\$C5	\$85
F	\$C6	\$86	\$C6	\$86
G	\$C7	\$87	\$C7	\$87
H	\$C8	\$88	\$C8	\$88
I	\$C9	\$89	\$C9	\$89
J	\$CA	\$8A	\$CA	\$8A
K	\$CB	\$8B	\$CB	\$8B
L	\$CC	\$8C	\$CC	\$8C
M	\$CD	\$8D	\$CD	\$8D
N	\$CE	\$8E	\$CE	\$8E
O	\$CF	\$8F	\$CF	\$8F
P	\$D0	\$90	\$D0	\$90
Q	\$D1	\$91	\$D1	\$91
R	\$D2	\$92	\$D2	\$92
S	\$D3	\$93	\$D3	\$93
T	\$D4	\$94	\$D4	\$94

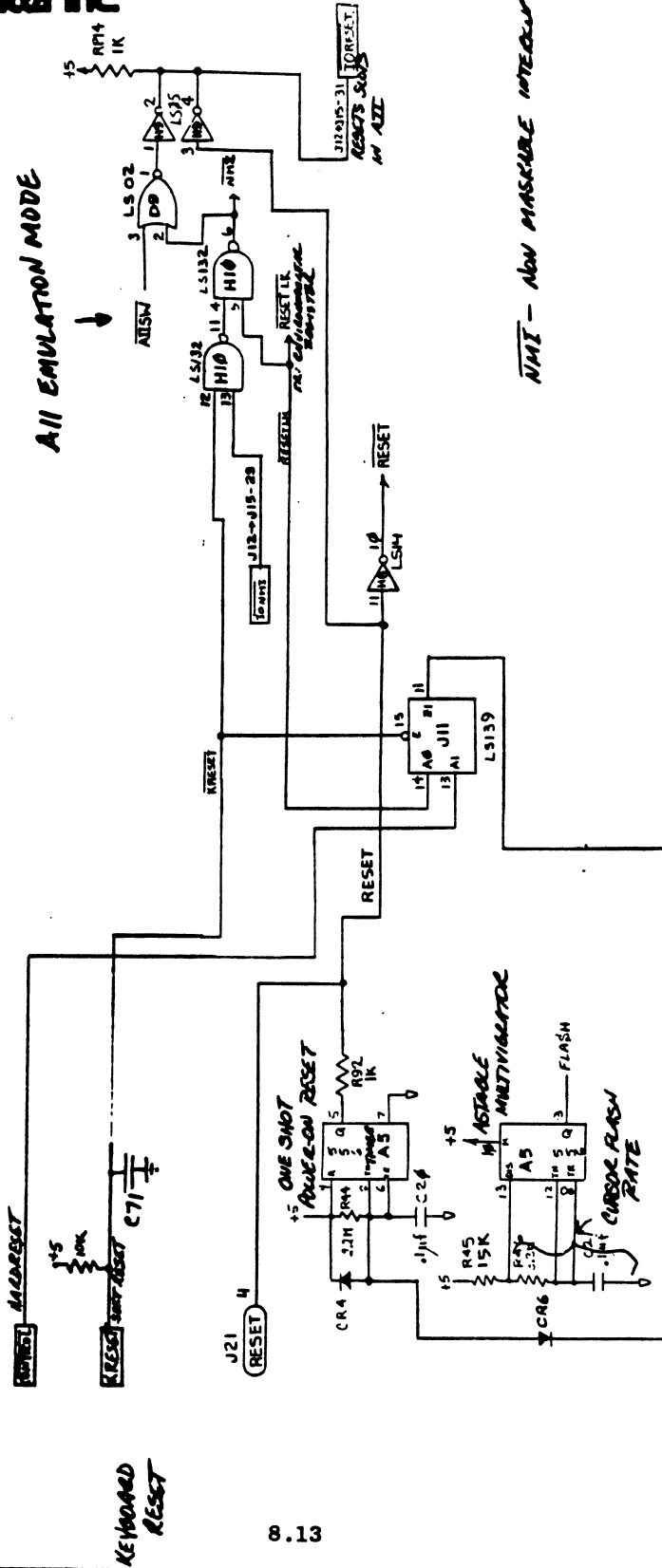
8.10

U	\$D5	\$95	\$D5	\$95
V	\$D6	\$96	\$D6	\$96
W	\$D7	\$97	\$D7	\$97
X	\$D8	\$98	\$D8	\$98
Y	\$D9	\$99	\$D9	\$99
Z	\$DA	\$9A	\$DA	\$9A



apple computer inc.

All EMULATION MODE

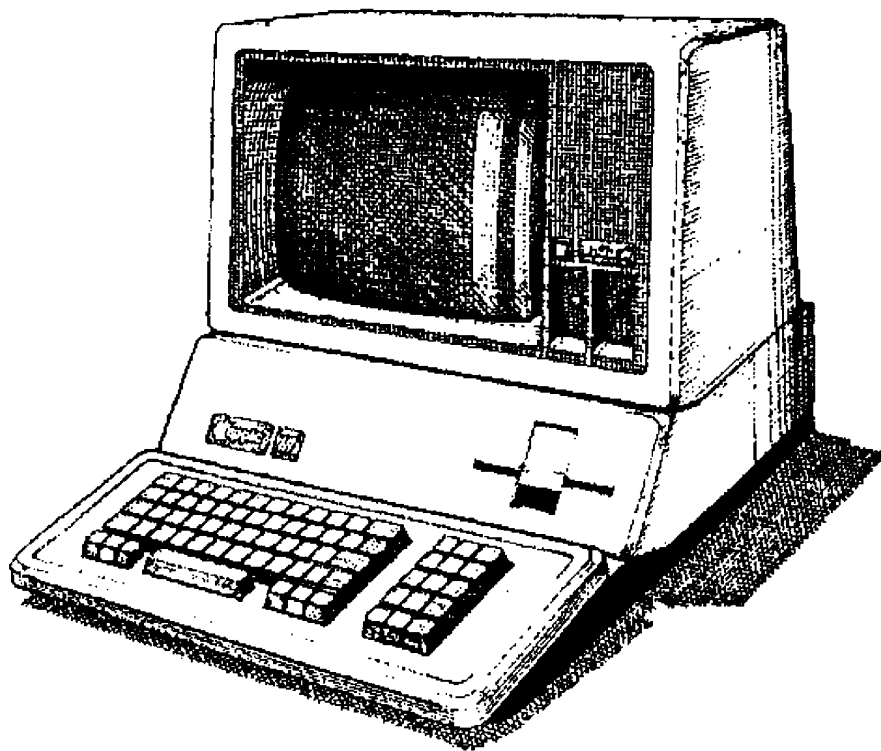


8.13



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 9 • Power Supply

Written by Apple Computer • 1982



THE APPLE /// POWER SUPPLY

The Apple /// power supply converts power from the AC line to DC. This is a constant voltage power supply. This means:

1. The output voltage is maintained constant regardless of changes in the load, line, or temperature.
2. The Apple /// power supply is a free running flyback type, off line switching power supply.
 - It can accept either 115VAC or 230VAC (jumper selectable) and delivers 4 regulated DC outputs at a total of 55 watts.
 - It supplies +5, -5, +11.8, and -12VDC.
 - It is called a flyback type power supply because energy is transferred from the primary of the transformer to the secondary when the switching transistor switches off (during flyback).

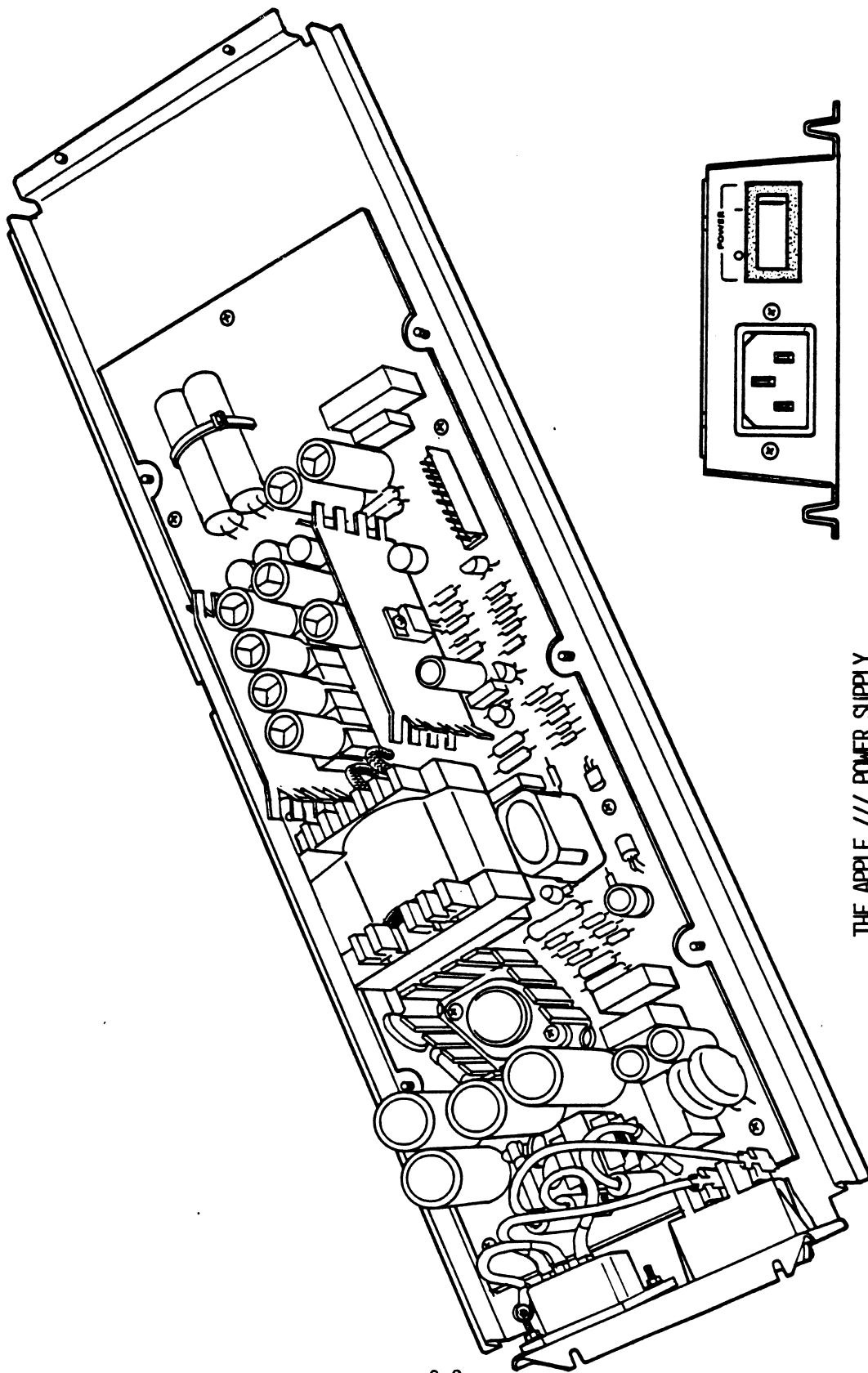
The following paragraphs will describe the switching power supply in more detail.

THE BASIC SWITCHING POWER SUPPLY

The regulating element of the switching power supply consists of a transistor that acts as a rapidly opened and closed switch. The AC input is rectified to unregulated DC, then "chopped" by the switching element components at a fast rate, approximately 25kHz. The resultant is transformer-coupled to an output network which provides the final rectification and filtering. Regulation is accomplished through control circuits that vary the on-off periods (duty cycles) of the switching components.

Advantages

1. Greater Efficiency
 - Lower power is dissipated because of the on/off role of the regulator. The switching transistors dissipate very little power when either saturated (on) or cutoff (off). With less wasted power, the switching power supply runs at cooler temperatures and costs less to operate.
2. Size and Weight
 - Because components such as capacitors, transformers, and inductors operate at high switching rates they can be smaller and weigh less than those that operate at power line frequencies.
3. Operating Conditions
 - The switching power supply can operate under low AC conditions and can sustain (holdup) its output if input power is momen-



THE APPLE /// POWER SUPPLY
PARTS LAYOUT

9.2



tarily lost. This is because the AC input is rectified and the filter capacitors charge to peak voltages on the AC line.

Disadvantages

1. Transient Recovery Time

- The dynamic loading regulation is slower than that of the series regulated supply. The recovery is limited mostly by the inductance of the output filter network.

2. EMI (Electro-Magnetic Interference)

- This is a natural byproduct of this type of power supply. This EMI can be conducted to the load (resulting in higher output ripple and noise), and it can be conducted back into the AC line. (Now you know where that stuff on TV came from).
- Apple designed this power supply with filter networks and shielding to greatly reduce EMI.

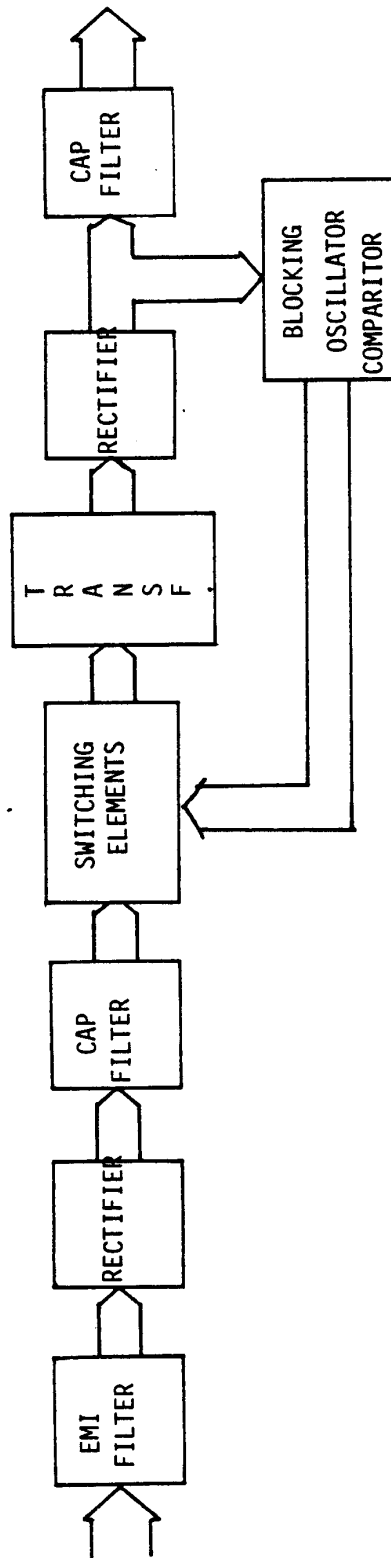
HOW IT WORKS!

Regulation is accomplished by a switching transistor Q2 operating under control of a feedback network. The feedback network, consisting of a voltage comparator and blocking oscillator, controls the duty cycle of the oscillator.

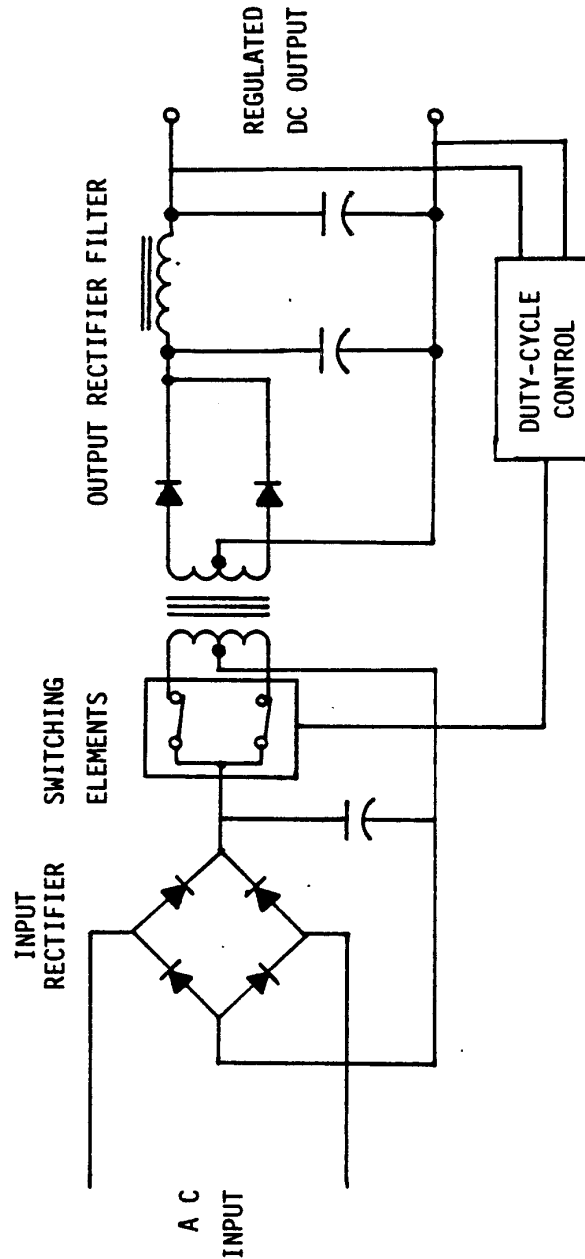
The energy is transferred from the primary to the secondary of the transformer and delivered to the output rectifier/filter. Here the waveform is rectified and averaged to provide a DC output level that is proportional to the duty cycle of the waveform.

Referring to the block diagram, Figure x.x below, note that:

- o The AC is passed through an EMI filter and then rectified to provide approximately 300 VDC across the capacitive input filters (C6, C7, C8, C9 of the schematic diagram). This voltage is applied to the primary of the transformer (T2) by the switching elements (turning on power transistor Q2). A linear current ramp is developed by the primary inductance of the transformer.
- o When the switching elements are turned off, the energy stored in the transformer is transferred to a second set of rectifiers through a capacitive filter network to provide filtering of the output.
- o The +5 volt output of the final rectifier network is compared to a reference voltage, and the error is fed back to a blocking oscillator.
- o The blocking oscillator basically changes the frequency depending on the output voltage. This in turn changes the repetition rate of the switching elements, which changes the energy transfer through the transformer and voltage output. This is how regulation is accom-



9.4



9.5



plished.

- o If the output voltage should change in such a way that the blocking oscillator goes into saturation, the output is essentially cut off.

DETAILED HARDWARE DESCRIPTION [Refer to Schematic]

A THERMISTOR, R1, is used to limit AC input surge current by its negative temperature coefficient of resistance. When cold, during turn on, R1 has a high resistance; after it heats up, R1 has a low resistance.

VDR1 is a varistor and is used as a transient suppressor. It keeps voltage spikes that result from power supply switching from affecting the performance of the power supply. It basically provides AC line surge current protection at turn on.

THE AC LINE SELECTABLE JUMPER, when connected to 220V position, causes the power supply to act as a conventional full wave rectifier. For 120V AC inputs the input circuitry becomes a voltage doubler.

THE EMI FILTER made up of T1, L1, L2, and C1, helps prevent high frequency RFI spikes from being conducted to the load or back into the AC line.

DB1 is a diode rectifier bridge.

THE SWITCHING ELEMENT consists of the circuitry associated with Q2 and Q1. You may recall that the linear current ramp, developed in the primary of the transformer when Q2 is turned on, is transferred to the secondary when Q2 is turned off.

The turn on of Q2 is accomplished by R2 for starting, and thereafter by the feedback winding in T2 driven by R4 and C10. This winding initiates turn on during the ringdown following the flyback.

If a sufficient voltage is developed across R9, Q1 will be forward bias. This would occur if by chance one of the output voltages were shorted. In that case, the oscillator would stop and shut off all the outputs, pause for 1/2 second, and attempt to restart.

THE OUTPUT RECTIFIER DIODES, D7 through D12, provide rectification, but also protect internal components against reverse currents that could be injected into supply by an active load.

IC1 helps accomplish regulation by comparing the output voltage against its own internal reference and delivering a voltage level to the base of Q3.

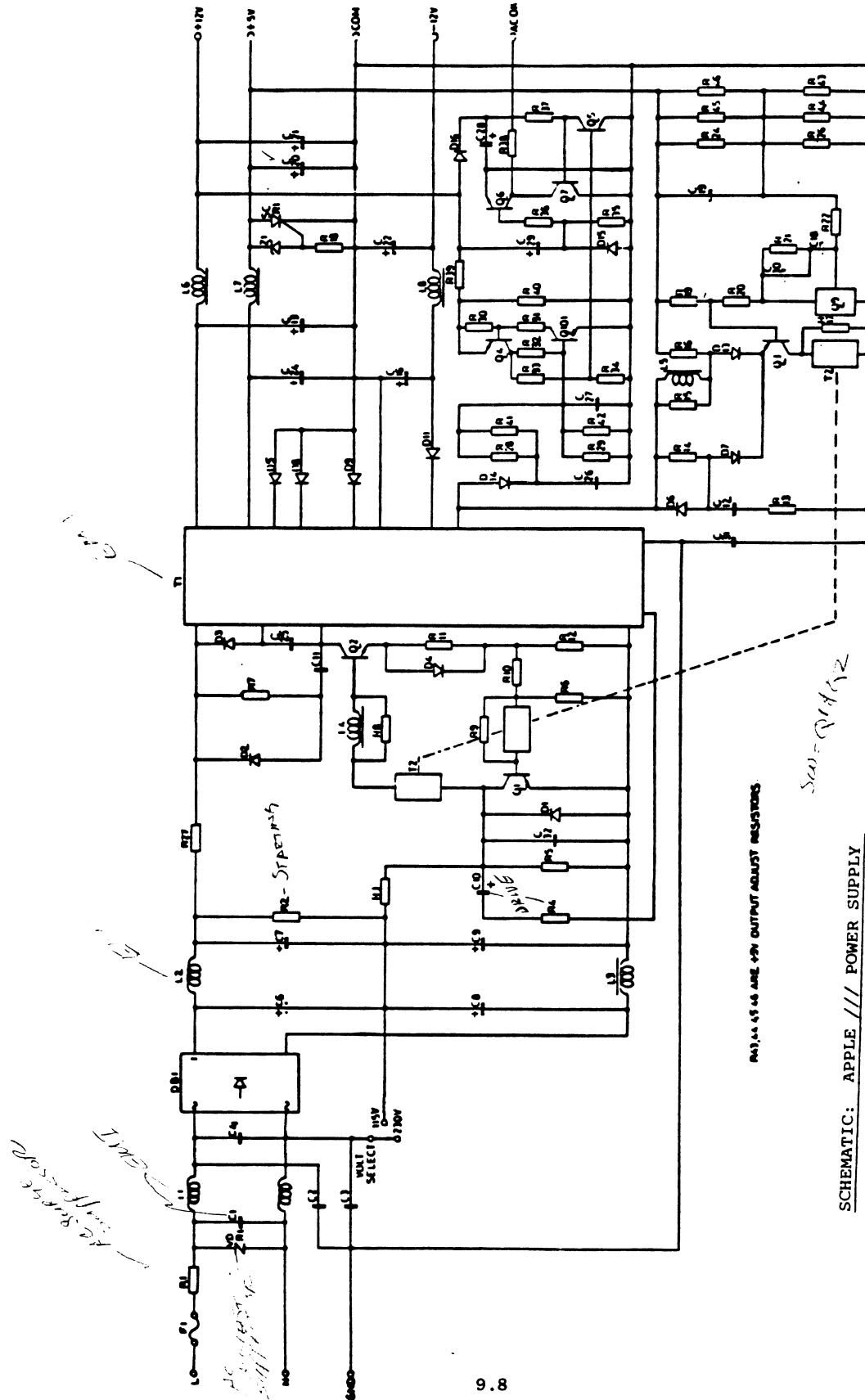
The emitter of Q3 is driven by a positive going ramp created by the inductive resistance associated with R14 while Q2 is on. When this voltage is sufficient to forward bias the emitter-base junction of Q3, conduction of Q2 is terminated.

You can now see that the operating frequency varies with the line and load.

OVERVOLTAGE PROTECTION is accomplished by sensing the +12V level via the re-

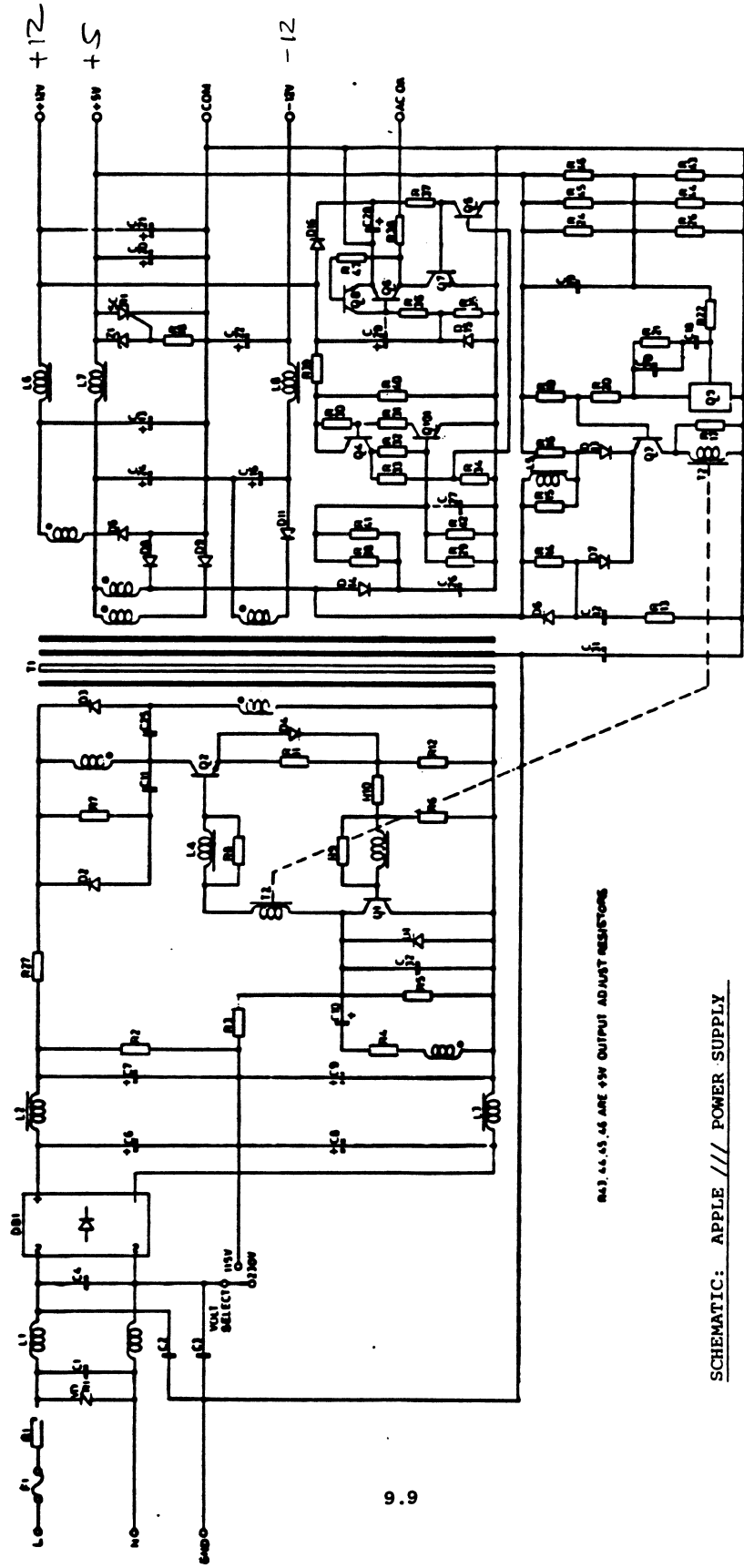


istor voltage divider of R17 and R18, referenced to the zener voltage on Z1. When the +12V output rises above tolerance, Q4 is turned on, which in turn triggers SCR1. SCR1 then clamps the +12V to ground, causing the power supply to fold back.



R41, R42, R43 ARE 4.7V OUTPUT ADJUST RESISTORS

SCHEMATIC: APPLE /// POWER SUPPLY



R4.3, 4.4, 4.5, 4.6 ARE 1%W OUTPUT ADJUST RESISTORS

SCHEMATIC: APPLE /// POWER SUPPLY

APPLE III POWER SUPPLY

REF	DESCRIPTION	QTY	PART NUMBER
	AC Input Socket		149-00200020
	Connector Housing 1 CCT	1	138-00000170
	Crimp Terminal	1	403-02200510
	Double Side Tape Width=3x4mm	20	027-01400010
	Faston Tab	1	403-02200700
	Heatsink	1	398-00200060
	Insulator 298. 45X88.9MM	1	183-00101410
	Nut M3 P=0.5 MS/NP	2	394-00400011
	On/Off Switch	1	278-01100010
	Pan	1	403-01101810
	PHL Pan M.C Screw M#x12 P=0.5 BS/NP	2	391-20204141
	PHL Pan M.C Screw M3X8 P=0.5 BS/NP	3	391-20204061
	PHL Pan M/C Screw M3X8 P=0.5 BS/NP	1	391-20204021
	PVC Coating CU Wire 100MM UL1015	1	356-12200571
	PVC Coating CU Wire 80MM UL1015	1	357-11800545
	PVC Coating CU Wire 95MM UL1015	1	356-12200566
	Rectifier RG3B	3	226-10700011
	Resistor, 68K \pm 5% 1/4W, Carbon Film	2	240-68306022
	Resistor, 82K \pm 5% 1/4W, Carbon Film	2	240-82306022
	SCR C122u	1	227-13000010
	Solder Bar	1	366-00130010
	Solder Bar 60/40	0	366-00130010
	Spring Washer M3 BS/NP	10	392-00800031
	Standoff M3	8	393-00200100
BRI	Bridge Rectifier KBP10		226-30500010
C01	Cap, 0.22uf, 250VAC, Metallized Paper	1	068-22400010
C02	Cap 0.1uf, 250VAC, Metallized Paper	1	068-10400010
C03	Cap, 4700pf, 400 VAC, Ceramic	2	055-47220001
C04	Cap, 4700pf, 400 VAC, Ceramic	2	055-47220001
C05	Cap, 0.1uf, 400 V, Polyester	1	058-10400100
C06	Cap, 100uf, 250V, Electrolytic	4	057-101201170
C07	Cap, 100uf, 250 V, Electrolytic	4	057-101201170
C08	Cap, 100uf, 250v, Electrolytic		057-10120170
C09	Cap, 100uf, 250V, Electrolytic	4	057-101201170
C10	Cap, 100uf, 250V, Electrolytic	1	057-22120080
C11	Cap, 0.001uf, 3KV, Ceramic		055-10261328
C12	Cap, 22uf, 100V, Polyester	2	058-22400120
C13	Cap, 1000uf, 10V, Electrolytic	6	057-10220020
C14	Cap, 1000uf, 10V, Electrolytic	6	057-10220020
C15	Cap, 1000uf, 10V, Electrolytic	6	057-10220020
C16	Cap, 1000uf, 10V, Electrolytic	6	057-10220020
C17	Cap, 330uf, 16V, Electrolytic		057-33120080
C18	Cap, 220uf, 10V, Electrolytic	1	057-22120060
C19	Cap, 0.22uf, 100V, Polyester	1	058-22300080
C20	Cap, 1000uf, 10V, Electrolytic		057-10220020
C21	Cap, 0.22uf, 100V, Polyester		058-22400120
C22	Cap, 1000uf, 10V, Electrolytic		057-10220020
C23	Cap, 330uf, 16V, Electrolytic	3	057-33120080
C24	Cap, 680uf, 16V, Electrolytic	1	057-68120010
C25	Cap, 330uf, 16V, Electrolytic	3	057-33120080
C26	Cap, 0.1/1KV, Ceramic	1	055-10360925
D01	Diode, Rectifier, RGP10A	1	226-10400050
D02	Diode, Rectifier, RGP10M	2	226-10400100
D03	Diode, Rectifier, RGP10M	2	226-10400100

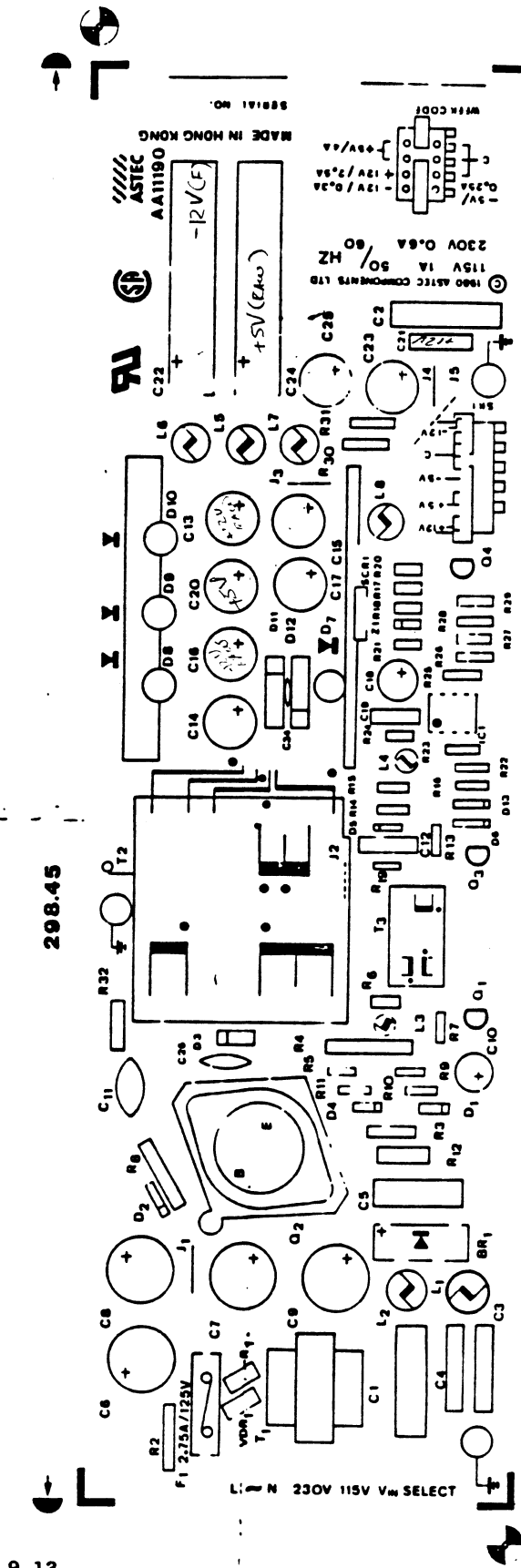
APPLE III POWER SUPPLY

REF	DESCRIPTION	QTY	PART NUMBER
D04	Diode, Rectifier, 1N4001GP	1	226-10400080
D05	Diode, Silicon, 1N5282	3	212-10700200
D06	Diode, Silicon, 1N5282	3	212-10700200
D07	Diode, Rectifier/Scr Assembly		853-00700010
D08	Diode, Rectifier assembly		853-00200140
D09	Diode, Rectifier Assembly		853-00200140
D10	Rectifier Assembly		853-00200140
D11	Schottky Diode S3SC3M		212-31100030
D12	Rectifier RG3B	1	226-10700010
D13	Diode, Silicon, 1N5282	3	212-10700200
F1	Fuse 2.75A 125V	1	084-00200040
IC1	Integrated Circuit, Regulator, TL431CP	1	211-10800070
J1	Jumper Wire	4	358-80810011
J2	Jumper Wire	4	358-80810011
J3	Jumper Wire	4	358-80810011
J4	Jumper Wire	40	358-80800001
L1	Choke	2	852-20100350
L2	Choke	2	852-20100350
L3	Base Choke	1	328-00100030
L4	Choke 1.5mH	1	328-00100010
L5	Choke Coil Assembly	1	852-20100010
L6	Choke Coil	1	852-10100370
L7	Choke Coil	1	328-00100060
L8	Choke Coil	1	328-00100060
Q1	Transistor SD467	1	209-11700463
Q2	Transistor 2SC1358	1	209-10200010
Q3	Transistor SB561	2	210-11700353
Q4	Transistor SB561	2	210-11700353
R01	Thermistor, 4R @25 C +10% 6R @ 25 C +20%	1	258-40970015
R02	Resistor, 150K +5% 1/2W	2	240-15406033
R03	Resistor, 150K +5% 1/2W		240-15406033
R04	Resistor, +5% 47R 2W, Metal Oxide	1	248-47006063
R05	Resistor, +5% 1/4W 1.2K	1	240-12206022
R06	Resistor, 5.6R +5% 1/4W	1	240-56906022
R07	Resistor, +5% 56R 1/4W, Carbon film		240-56006022
R08	Resistor, +5% 120R 2W	1	248-12106063
R09	Resistor, +5% 1/4W 15R	2	240-15006022
R10	Resistor, +5% 1/4W 10R, Carbon Film	1	240-10006022
R12	Resistor, 0.47R, Metal Film	1	247-04786054
R13	Resistor, -5% 1/4W 39R, Carbon Film	1	240-39006022
R14	Resistor, +5% 270R 1/4W	2	240-27106033
R15	Resistor, +5% 270R 1/4W, Carbon Film		240-27106033
R16	Resistor, 8.2 +5% 1/4W, Carbon Film	1	240-82906022
R17	Resistor, +5% 680R 1/4W	1	240-68106022
R18	Resistor, +5% 1.8K, Carbon Film	1	240-18206022
	Resistor, +5% 2.2K, Carbon Film	1	240-22206022
	Resistor, +5% 2.7K 1/4W, Carbon Film	1	240-27206022
R19	Resistor, +5% 560R 1/4W, Carbon Film	1	240-56106022
R20	Resistor, 22R 1/4W +5%, Carbon film	1	240-22006022
R21	Resistor, 100R +5% 1/4W, Carbon Film	1	240-10106022
R22	Resistor, 56R +5% 1/4W, Carbon Film	3	240-56006022
R23	Resistor, 56R +5% 1/4W, Carbon film		240-56006022
R243	Resistor, 12K +5% 1/4W, Carbon Film	1	240-12306022
R25	Resistor, +5% 1/4W 470R, Carbon Film		240-47106022

APPLE III POWER SUPPLY

REF	DESCRIPTION	QTY	PART NUMBER
R26	Resistor, +-2% 2.7K 1/4W, Metal Film	2	247-2701502
R27	Resistor, +-2% 2.7K 1/4W, Metal Film		247-27015022
R28	Resistor, 100K +-5% 1/4W, Carbon Film	2	240-10406022
R29	Resistor, 100K +-2% 1/4W, Carbon Film		240-10406022
R30	Resistor, -5% 56R 1W, Metal Oxide Filmm	1	248-56006052
R31	Resistor, +-5% 220R 1W, Metal Oxide Film	1	248-22106052
R32	Resistor, 1R 1w, Metal film	1	247-10086054
T1	Common Mode Choke Assembly	1	852-20200010
T2	Power Transformer assembly	1	852-10200760
T3	Control Transformer Assembly	1	852-10200680
VDRL	Varistor 260VAC	1	256-26100014
Z1	Zener Diode 9.6 to 10.V @ 1mA	1	222-98085002

THE APPLE /// POWER SUPPLY
COMPONENT LAYOUT



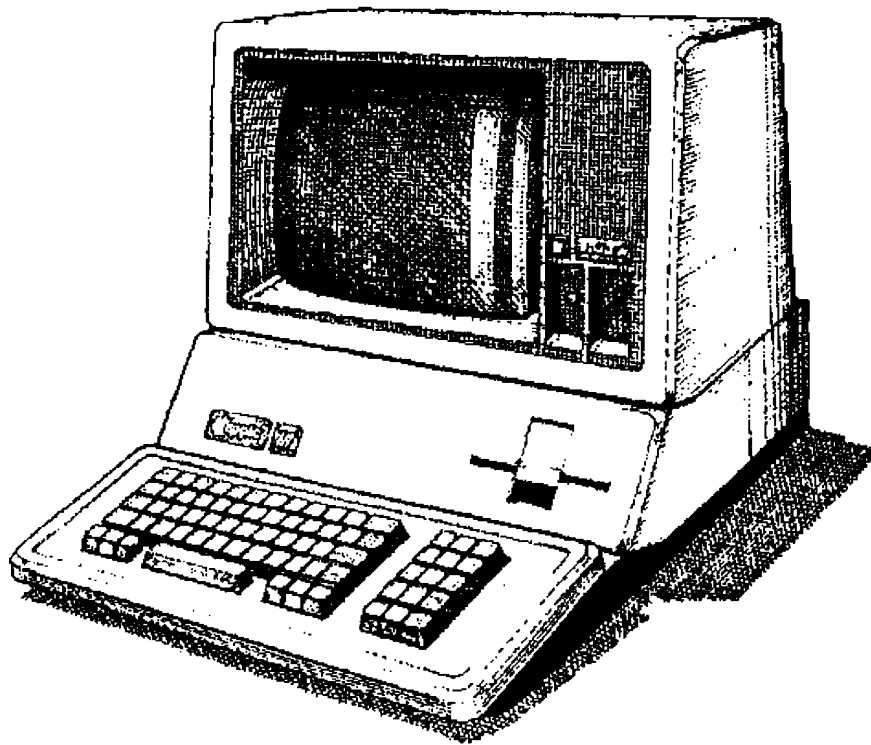
9.13

© JAN 1982
 09422015201
 Samsco



Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 10 • Apple][Emulation

Written by Apple Computer • 1982



APPLE II EMULATION RESTRICTIONS

- 0 NO LANGUAGE CARD
- 0 NO ROM CARD
- 0 PADDLES ARE DIFFERENT
- 0 ENTER WITH SOFTWARE BUT ONLY
RESET WILL EXIT

10.1

THE COLOR VIDEO CONNECTOR

<u>Pin</u>	<u>Name</u>	<u>Description</u>
1	SG	Shield Ground.
2	XRGB4	One of four GRB outputs. This (and pins 5, 9, and 10) is a TTL output with instantaneous color information. A linear weighted sum of these four signals will form a true 16-color RGB video signal
3	SYNCH	Composite synchronization signal with negative-going tips.
4	PDI	Not used.
5	XRGB1	See pin 2.
6	GND	Power and signal ground.
7	-5V	-5 volt power supply. A device may draw up to 200 ma through this pin.
8	+12V	+12 volt power supply. A device may draw up to 500 ma through this pin.
9	XRGB2	See pin 2.
10	XRGB8	See pin 2.
11	BWVID	Black and white composite video. This is an NTSC composite video signal with negative-going synch tips, 1 volt peak-to-peak into a 75 ohm load. Color information is encoded as a linear grey scale.
12	NTSC	Color composite video. This is an NTSC-compatible video signal with negative-going synch tips, 1 volt peak-to-peak into a 75 ohm load.
13	GND	Power and signal ground.
14	-12V	-12 volt power supply. A device may draw up to 200 ma through this pin.
15	+5V	+5 volt supply. A device may draw up to 1 amp through this pin.

This connector supplies 7 different video signals and 4 power supply voltages. Through this connector you can hook up the Apple to any NTSC color or black and white video monitor. With an additional circuit you can hook up the Apple to a studio-quality RGB color monitor.

All power supply current ratings assume that no peripheral cards are installed in the system. If there are cards in the system, be sure to account for the current drawn by those cards.



THE HIGH-RESOLUTION GRAPHICS (HI-RES) MODE

The Apple][emulation mode high resolution graphics are identical to the Apple][except some combinations of colors on the right edge of the screen will cause the left edge pixels to blink. This is normal though distracting.

THE SPEAKER

The speaker function is identical to the Apple][with the following additional features.

A reference to location 49216 (or the equivalent addresses -16336 or hexadecimal \$C040) will cause a 0.1 second 1 KHz tone to be produced which is similar to the sound the AUTOSTART monitor makes when the BELL character is sent to the screen. The advantage to this is 0.1 seconds of cpu time is returned to the user since only 1 microsecond is required to start the BELL sound.

The AUDIO connector at the back of the Apple /// provides the same signal as the speaker. When you insert a miniature phone-tip plug into this jack, the Apple's internal speaker is silenced; if there is an amplifier or other device properly connected to the plug, then that device will receive all audio signals generated by the Apple. The signal is a 0.5 volt peak-to-peak audio signal on its tip and signal ground on its ring.

THE CASSETTE INTERFACE

The cassette interface is completely eliminated on the Apple ///. References to the cassette output port at 49184 (or the equivalent -16352 or hexadecimal \$C020) will cause pin 39 of the I/O slots to go low for a microsecond. This is for use by Apple /// native mode peripherals to deselect to \$C800 ROM address space.

Reading the cassette input port at 49248 or the equivalents -16288 or hexadecimal \$C060 will read joystick switch 0 into bit 7.

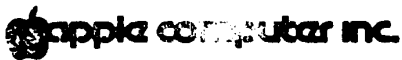


Table 10: Input / Output Special Locations

Function	Address:		Read/Write	
	Decimal	Hex		
Speaker	49200	-16336	§C030	R/W
Beep	49216	-16320	§C040	R/W
Deselect §C800 for Apple /// peripherals (pin 39 in slots)				
Joystick switch 0	49184	-16352	§C020	R/W
Joystick switch 1	49248	-16288	§C060	R(bit 7)
Joystick switch 2	49249	-16287	§C061	R(bit 7)
Joystick switch 3	49250	-16286	§C062	R(bit 7)
	49251	-16285	§C063	R(bit 7)
<hr/>				
A/D Select 0	49240	-16296	§C058	R/W
A/D select 0	49241	-16295	§C059	R/W
<hr/>				
A/D Select 1	49246	-16290	§C05E	R/W
A/D Select 1	49247	-16289	§C05F	R/W
<hr/>				
A/D Select 2	49242	-16294	§C05A	R/W
A/D Select 2	49243	-16293	§C05B	R/W
A/D Ramp charge	49244	-16292	§C05C	R/W
A/D Start timeout	49245	-16291	§C05D	R/W
<hr/>				
A/D Timeout Clock millisecond counter (§N0)	49254	-16282	§C066	R(bit 7)
	49264	-16272	§C070	R(bits 7-4)

Table 9: A/D Selection

A/D 2	A/D 1	A/D 0	Input
0	0	0	Ground
0	0	1	Joystick, Port B, X axis
0	1	0	Joystick, Port B, Y axis
0	1	1	Joystick, Port A, X axis
1	0	0	Joystick, Port A, Y axis
1	0	1	Clock Battery
1	1	0	No connection
1	1	1	Reference Voltage



ANALOG INPUTS

The system has two joystick ports with provisions for two A/D inputs each. Joystick Port A reads A/D inputs 0 and 2 while Port B reads inputs 1 and 3 as defined in BASIC and the monitor subroutine PREAD.

To read the A/D inputs, the software must select the desired input and charge the ramp capacitor for at least 500 microseconds. Then the ramp is started and the time measured until the A/D timeout goes low. The discharge time is proportional to the input voltage.

STROBE OUTPUT

The strobe output (\$C040) has been replaced by a 0.1 second 1 KHz tone from the speaker.

AUTOSTART ROM / MONITOR ROM

The Apple][emulation only comes with a modified version of the Autostart ROM. This is in write protected RAM which is loaded when the Apple][emulation disk is booted.



THE SYSTEM MONITOR

SAVING A RANGE OF MEMORY ON THE TAPE

Since there is no cassette port on the Apple /// the W (for WRITE) command has no effect. The code in the Emulation mode Autostart Monitor contains an RTS instruction followed by NOP instructions, followed by BRK instructions. This fills the space occupied by the WRITE subroutine (locations \$FECD-\$FEF4).

READING A RANGE FROM TAPE

Again, since there is no cassette port the R (READ) command has no effect. The READ subroutine contains an RTS followed by NOP instructions, followed by BRK instructions (locations \$FEFD-\$FF2C).

SOME USEFUL MONITOR SUBROUTINES

\$FB1E PREAD READ A JOYSTICK AXIS

PREAD will return a number which represents the position of a joystick axis. You should pass the number of the joystick axis (0 to 3) in the X register. If this number is greater than 3, port A, Y axis is read. PREAD returns a number from \$00 to \$FF in the Y register. The accumulator is scrambled.

Joystick	Reference #
Port A, X axis	0
Port B, X axis	1
Port A, Y axis	2
Port B, Y axis	3

Page Three Monitor Locations

Address:		Use:
Decimal	Hex	
1008	\$3F0	Holds the address of the subroutine which handles machine language "BRK" requests (normally \$FA59).
1009	\$3F1	
1010	\$3F2	Soft Entry Vector. These two locations contain the address of the reentry point for whatever language is in use. Normally contains \$E003.
1011	\$3F3	
1012	\$3F4	Power-up byte. Normally contains \$45.
1013	\$3F5	Holds a "JuMP" instruction to the subroutine which handles Applesoft]["G" commands. Normally \$4C \$58 \$FF.
1014	\$3F6	
1015	\$3F7	
1016	\$3F8	Holds a "JuMP" instruction to the subroutine which handles "USER" (CONTROL Y) commands.
1017	\$3F9	
1018	\$3FA	



Built-In I/O Locations

	\$0	\$1	\$2	\$3	\$4	\$5	\$6	\$7
\$C000	Keyboard Port A Input							
\$C008	Keyboard Port B Input							
\$C010	Clear Keyboard Strobe							
\$C020	Deselect all expansion I/O space (pin 39) for Apple /// cards							
\$C030	Speaker Toggle (lus) pulse							
\$C040	Speaker Beep (1 KHz for 0.1 second)							
\$C050	gr	tx	nomix	mix	pri	sec	lores	hires
\$C058	A/D 0	A/D 0	A/D 2	A/D 2	A/D CHG	A/D ST	A/D 1	A/D 1
\$C060	SW0	SW 1	SW 2	SW 3	IRQ 2	IRQ 1	A/D TM	MUX1
\$C070	Clock millisecond output \(\$N0)							
\$C090-\$C09F	Slot 1 Device Select (pin 41) goes low during CIM							
\$C0A0-\$C0AF	Slot 2 Device Select (pin 41) goes low during CIM							
\$C0B0-\$C0BF	Slot 3 Device Select (pin 41) goes low during CIM							
\$C0C0-\$C0CF	Slot 4 Device Select (pin 41) goes low during CIM							
\$COE0	<u>Disk Stepper Motor Phase A</u>							
\$COE1	Disk Stepper Motor Phase A							
\$COE2	<u>Disk Stepper Motor Phase B</u>							
\$COE3	Disk Stepper Motor Phase B							
\$COE4	<u>Disk Stepper Motor Phase C</u>							
\$COE5	Disk Stepper Motor Phase C							
\$COE6	<u>Disk Stepper Motor Phase D</u>							
\$COE7	Disk Stepper Motor Phase D							



\$COE8 Disk motor off
\$COE9 Disk motor on
\$COEA Select Drive 1 (Built-in)
\$COEB Select Drive 2 (First external)
\$COEC Q6L
\$COED Q6H
\$COEE Q7L
\$COEF Q7H
\$COF0 ACIA Receive/Transmit Data register
\$COF1 ACIA Status register
\$COF2 ACIA Command register
\$COF3 ACIA Control register
\$C100-\$C1FF Slot 1 I/O Select (Pin 1) goes low during C1M low
\$C200-\$C2FF Slot 2 I/O Select (Pin 1) goes low during C1M low
\$C300-\$C3FF Slot 3 I/O Select (Pin 1) goes low during C1M low
\$C400-\$C4FF Slot 4 I/O Select (Pin 1) goes low during C1M low



PERIPHERAL BOARD I/O

The Apple /// implements only slots 1 through 4. Slot 6 is always a disk interface card and slots 5 and 7 emulate either a SERIAL or COMMUNICATIONS card. Slot 0 scratchpad RAM exists but no provision is made to put a LANGUAGE card or FIRMWARE card into the system. Thus the RAM is limited to 48K with a 12K ROM chosen at Boot time.

PERIPHERAL CARD I/O SPACE

Slot 6 device I/O space \$COE0-\$COEF contains the hardware for the disk interface. Slot 7 device I/O space \$COF0-\$COF3 contains the addresses for the onboard ACIA.

PERIPHERAL CARD ROM SPACE

Slot 5 and slot 7 contain code which is functionally equivalent to the COMMUNICATIONS or SERIAL card for the Apple][. They differ in that they use the built-in ACIA. For a more complete explanation see "SERIAL AND COMMUNICATIONS CARD EMULATION".

Slot 6 contains a copy of the Apple][16 sector Boot PROM.



ROM MEMORY

The Applesoft, Integer Basic, and Autostart Monitor "ROMS" are actually write protected RAMs in the Apple ///. When the Emulation mode disk is booted it loads RAM memory with an image of each set of ROMs. Whichever language is selected when the Apple][disk is booted is loaded into the address space (\$D000-\$FFFF) and write protected.

RAM MEMORY

In Emulation mode there is always 48K of RAM. It is addressed \$0000 to \$BFFF. There is no provision for a slot 0 Language or Firmware card.

"USER 1" JUMPER

There is no "User 1" jumper in the Apple ///.

THE GAME I/O CONNECTOR

There is no 16 pin Game I/O connector in the Apple ///. However there are two 9 pin "D" - joystick connectors.

THE JOYSTICK PORTS

The Apple /// has two joystick ports (A and B). The A port will NOT operate a silentype printer in Emulation mode. The physical pinout is:

5	4	3	2	1
9	8	7	6	

PORT A PINOUT

Pin	Name	Description
1	SGND	Shield ground.
2	+5V	+5 volt power supply.
3	GND	Power and Signal Ground.
4	X0	Horizontal analog input, PDL (0) in BASIC.
5	SW1	Joystick switch 1, orange button.
6	+12V	+12 volt power supply.
7	GND	Power and signal Ground.
8	Y0	Vertical analog input, PDL (2) in BASIC.
9	SW3	Joystick switch 3.

PORT B PINOUT

Pin	Name	Description
1	SGND	Shield Ground.
2	+5V	+5 volt power supply.
3	GND	Power and Signal ground.
4	X1	Horizontal analog input, PDL (1) in BASIC.
5	SW2	Joystick switch 2, orange button.
6	+12V	+12 volt power supply.
7	GND	Power and signal ground.
8	Y1	Vertical analog input, PDL (3) in BASIC.
9	SW0	Joystick switch zero.



THE KEYBOARD

The keyboard is different in design but the locations of the Keyboard Data Input and the Clear Keyboard Strobe are the same. For more information see "THE KEYBOARD" in chapter 1.

CASSETTE INTERFACE JACKS

There are no cassette interface jacks in the Apple ///.

POWER CONNECTOR

The power connector is different but is not user accessible.

SPEAKER

The speaker is identical to the Apple][.

PERIPHERAL CONNECTORS

The Apple][emulation redefines a few of the pins on the connector and adds several new ones.

The most significant difference is that interrupts will not be sent to the 6502 from the slots. In fact the IRQ (pin 30) is an input to the cpu so the card can't even determine if an interrupt is occurring. Thus Emulation mode runs without interrupts, period.

The RES (pin 31) is an output to the card and goes low when the RESET key is pressed on the keyboard. However the microprocessor is actually performing an NMI not a RESET.

Peripheral Connector Pinout

GND	26	25	+5V
DMAOK	27	24	NOT USED
$\overline{\text{DMAI}}$	28	23	NOT USED
$\overline{\text{IONMI}}$	29	22	$\overline{\text{TSADE}}$ (Open collector)
$\overline{\text{IRQ}}$	30	21	RDY (Open collector)
$\overline{\text{IORES}}$	31	20	$\overline{\text{I/O STROBE}}$
$\overline{\text{INH}}$	32	19	PHO
-12V	33	18	R/ $\overline{\text{W}}$
-5V	34	17	A15
SYNC	35	16	A14
C7M	36	15	A13
Q3	37	14	A12
$\overline{\text{C1M}}$	38	13	A11
$\overline{\text{IOCLR}}$	39	12	A10
C1M	40	11	A9
$\overline{\text{DEV SEL}}$	41	10	A8
D7	42	9	A7
D6	43	8	A6
D5	44	7	A5
D4	45	6	A4
D3	46	5	A3
D2	47	4	A2
D1	48	3	A1
D0	49	2	A0
+12V	50	1	$\overline{\text{I/O SELECT}}$

Peripheral Connector Signal Description

Pin:	Name:	Description:
1	<u>I/O SELECT</u>	This line, normally high, will become low when the microprocessor references page \$Cn, where n is the individual slot number. This signal become active during PHO (nominally 500ns) and will drive 12 LSTTL loads.
2-17	A0-A15	The buffered address bus. The address on these lines becomes valid within 300ns after the beginning of <u>CIM</u> and remains valid through PHO. These lines will each drive 8 LSTTL loads.
18	<u>R/W</u>	Buffered Read/Write signal. This becomes valid at the same time the address bus does, and goes high during a read cycle and low during a write. This line can drive up to 10 LSTTL loads.
19	PHO	A 1 MHz signal which is identical to CIM. This line will drive 5 LSTTL inputs.
20	<u>I/O STROBE</u>	This line will go low during CIM when the address bus contains an address between \$C000 and \$CFFF. This line will drive 12 LSTTL loads.
21	RDY	The 6502's RDY input. This line should change only during CIM, and when low will halt the microprocessor on the next read cycle. This line has a 1K ohm pullup to +5V. This line should be driven from an open collector output.
22	<u>TSADB</u>	A low on this line from the peripheral will cause the address bus to tri-state for Direct Memory Access (DMA) applications. This has a 1 K ohm resistor pullup to +5V. This should be driven from an open collector output.
23		Not used in an Apple ///.
24		Not used in an Apple ///.
25	+5V	Positive 5-volt supply, 2.0 amps total for all peripheral boards together (but note a limit of 1.5 Watts per board).
26	GND	System circuit ground. 0 volt line from power supply. Do not use for shield ground.
27	DMAOK	Acknowledge signal to the peripheral following

10.15

		its request for the special Direct Memory Access (DMA) mode. Informs the peripheral that the DMA can now proceed.
28	$\overline{\text{DMAI}}$	Direct Memory Access (DMA) interrupt. Requests the A Apple /// DMA mode. Has a 1 K ohm pullup to +5. This should be driven from an open collector output.
29	$\overline{\text{IONMI}}$	Input/Output Non-Maskable Interrupt. This is equivalent to the IORES (pin 31) line as it will execute the same code in the Autostart ROM. This line should be driven by an open collector output.
30	$\overline{\text{IRQ}}$	This line is ignored in Apple][emulation mode. It should be driven by a TTL output.
31	$\overline{\text{IORES}}$	Input/Output Reset signal used to reset the peripheral devices. Pulled low by a power on or RESET key. This line will drive 12 LSTTL loads.
32	$\overline{\text{INH}}$	Inhibit line. When a device pulls this line low, all system memory is disabled. This line has a 1 K ohm pullup resistor to +5V and should be driven form an open collector output.
33	-12V	Negative 12 volt supply, 200mA total for all peripheral boards together.
34	-5V	Negative 5 volt supply, 200mA total for all periperal boards together.
35	SYNC	The 6502 opcode synchronization signal. Can be used for external bus control signals. Will drive 10 LSTTL loads.
36	C7M	Seven MHz high frequency clock. Will drive 10 LSTTL loads.
37	Q3	A 2MHz (nonsymmetrical) general purpose timing signal. Will drive 10 LSTTL inputs.
38	$\overline{\text{CIM}}$	Complement of CIM clock. This will drive 12 LSTTL loads.
39	$\overline{\text{IOCLR}}$	Provides the \$C800 space disable function directly without address decoding (\$CFFF is used for Apple][peripherals. It is addressed from \$C02x. This line will drive 12 LSTTL loads.
40	$\overline{\text{CIM}}$	Phase CIM clock. This is the same as the microprocessor's 1 MHz clock. This will drive 12 LSTTL loads.
41	$\overline{\text{DEVICE SELECT}}$	This line becomes acive (low) on each peripheral

connector when the address bus is holding address between \$C0n0 and \$C0nF where n is the slot number plus \$8. This line will drive 12 LSTTL loads.

42-49 D7-D0

The 8-bit system data bus. During a write cycle, data is set up by the 6502 less than 300ns after

the beginning of $\overline{C1M}$. During a read cycle the 6502 expects data to be ready no less than 100ns

before the end of $\overline{C1M}$. These lines will drive 8 LSTTL inputs.

50 +12V

Positive 12 volt supply, 300mA total for all peripheral boards together.



ROM LISTINGS

APPLE II EMULATION MODE AUTOSTART ROM LISTING

The following is a listing of addresses which changed content in the Autostart ROM to eliminate cassette I/O, read joysticks, and redirect the NMI vector to the RESET code.

```

; THE ACIA IS CAPABLE OF GENERATING INTERRUPTS IN EMULATION MODE.
; IF IT DOES THE INTERRUPT RECEIVER SETS THE PROCESSOR INTERRUPT
; INHIBIT BIT TO PREVENT THE SERVICING OF THIS INTERRUPT

FA49: 4C 10 FF      JMP IHBRQS      ;JMP TO CODE TO INHIBIT INTERRUPTS
FF10: 68           PLA           ;GET PROCESSOR STATUS BYTE
FF11: 09 04       ORA #$04       ;SET INTERRUPT INHIBIT BIT
FF13: 48           PHA           ;PUT STATUS BYTE BACK ON STACK
FF14: A5 45       LDA $45       ;RESTORE ACCUMULATOR
FF16: 40           RTI           ;RETURN WITH INTERRUPTS INHIBITED
    
```

```

; THE RESET KEY IN EMULATION MODE GENERATES AN NMI (NONMASKABLE
; INTERRUPT). THEREFORE THE NMI VECTOR IS SET TO POINT AT THE
; RESET CODE WHICH ALSO MAKES SURE THE DISK MOTOR STOPS
    
```

```

FFFA: 62 FA      DFB      RESET ;POINT NMI VECTOR TO RESET CODE
FA62: D8      RESET  CLD           ;BINARY ARITHMATIC PLEASE
FA63: AD EE CO  LDA      $COEE ;SET DISK READ
FA66: AD EC CO  LDA      $COEC
FA69: AD E8 CO  LDA      $COE8 ;TURN OFF DISK
FA6C: 20 84 FE  JSR      SETNORM
FA6F: 20 2F FB  JSR      INIT
FA72: 20 93 FE  JSR      SETVID
FA75: 20 89 FE  JSR      SETKBD
FA78: EA      NOP
FA79: EA      NOP
FA7A: EA      NOP
    
```

```

; THE CASSETTE READ ROUTINE SIMPLY RETURNS TO USER CALLS
    
```



```
FEFD: 60      READ   RTS      ;NO CASSETTE PORT - RETURN TO USER
FEFE-FF0A: EA          NOP      ;FILL CODE WITH NOPS
FF0B: 22
FF0C-FF0F: 00          BRK      ;STOP USER FROM JUMPING INTO MIDDLE OF CODE
FF17-FF2C: 00          BRK
```

```
;      THE CASSETTE WRITE ROUTINE SIMPLY RETURNS TO USER CALLS
```

```
FECD: 60      RTS      ;NO CASSETTE PORT - RETURN TO USER
FECE-FEF2: EA          NOP      ;FILL CODE WITH NOPS
FEF3-FEF5: 00          BRK      ;STOP USER JUMPING INTO CODE
```

```
;      READ JOYSTICK AXIS. THIS IS THE SAME ENTRY ADDRESS OF PREAD
;      WHICH READS THE GAME PADDLES IN THE APPLE ][
;
;      X REGISTER CONTAINS JOYSTICK AXIS AND Y RETURNS $00-$FF OF JOYSTICK
```

	X REGISTER	JOYSTICK AXIS
;	0	PORT A, X AXIS
;	1	PORT B, X AXIS
;	2	PORT A, Y AXIS
;	3	PORT B, Y AXIS

```
FB1E: 8A      PREAD   TXA          ;SAVE X REGISTER
FB1F: 48      PHA
FB20: 49 01    EOR # $01      ;REMAP JOYSTICK ADDRESS
FB22: AA      TAX
FB23: AD 59 C0 LDA $C059      ;SET ANALOG MUX TO PORT B, X AXIS
FB26: AD 5E C0 LDA $C05E
FB29: AD 5A C0 LDA $C05A
FB2C: 4C C9 FC JMP JOY2
```

```
FCC9: E8      JOY2    INX
FCCA: CA      DEX          ;SET FLAGS
FCCB: FO 12    BEQ JOY3      ;PORT B, X AXIS?
FCCD: AD 5F C0 LDA $C05F      ;NO
FCD0: CA      DEX
FCD1: FO 0C    BEQ JOY3      ;PORT A, X AXIS?
FCD3: AD 58 C0 LDA $C058      ;NO
FCD6: CA      DEX
FCD7: FO 06    BEQ JOY3      ;PORT B, Y AXIS?
FCD9: AD 5E C0 LDA $C05E      ;NO
```

10.19



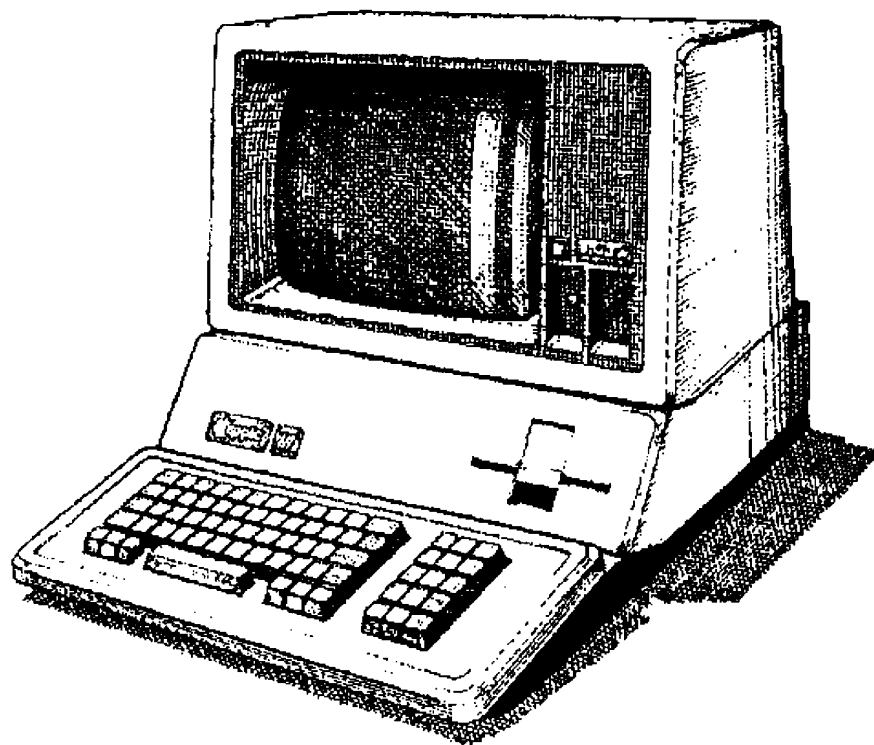
```

FCDC: AD 5B C0          LDA $C05B          ;MUST BE PORT A, Y AXIS
FCDF: AD 5C C0 JOY3    LDA $C05C          ;CHARGE CAPACITOR
FCE2: A9 0F            LDA #$0F           ;WAIT 800US
FCE4: 20 A8 FC        JSR WAIT
FCE7: A0 80            LDY #$80
FCE9: AD 5D C0        LDA $C05D          ;START TIMEOUT
FCEC: A2 48            LDX #$48           ;WAIT 370US
FCEE: CA              JOY4    DEX
FCEF: 10 FD          JOY5    BPL JOY4
FCF1: E8              JOY5    INX
FCF2: B9 E6 BF        LDA $BFE6,Y        ;FALSE READ
FCF5: 2A              ROL
FCF6: AD 66 C0        LDA $C066          ;BIT 7 IS VOLTAGE CROSSOVER
FCF9: 30 F6           BMI JOY5           ;HAS VOLTAGE CROSSED OVER?
FCFB: 8A              TXA               ;YES
FCFC: 10 04           BPL JOY6           ;WAS COUNT POSITIVE?
FCFE: A9 FF           LDA #$FF           ;NO
FD00: D0 01           BNE JOY7           ;USE $FF
FD02: 2A              JOY6    ROL         ;DOUBLE COUNT
FD03: A8              TAY               ;RETURN COUNT IN Y
FD04: 68              PLA               ;RESTORE X
FD05: AA              TAX
FD06: 60              RTS
FD07-FD0B: 00        BRK               ;FILL SPACE
    
```




Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 11 • Schematic Diagrams

Written by Apple Computer • 1982

NOTE: UNLESS OTHERWISE SPECIFIED

- = 750
- = ALL UNITS UNDER TEST MOLEX PINS TO BE LOCATED TOGETHER.
- = ALL TIMING MOLEX PINS TO BE LOCATED TOGETHER.
- THIS SCHEMATIC REPRESENTS ASSY 610-C105 AT REV LEVEL 1 F.

FOR STANDARD SYSTEMS G9 IS 341-0030, FOR EURO SYSTEMS G9 IS 341-0060.
 FOR STANDARD SYSTEMS Y1 IS H.316630 MHZ, FOR EURO SYSTEMS Y1 IS H.250450 MHZ.

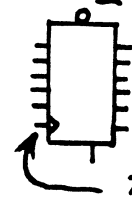
REV	Q001	REVISED	BY	DATE	DESCRIPTION	APPROVED
1	Q001	ENG. RELEASE				
2	Q002	PRE-PRODUCTION RELEASE				
3	Q003	PRE-PRODUCTION CHANGES				
4	Q004	ECO RELEASE				
5	Q005	DELETED CAP ARRAY (CR1), A10 N1B, M13, K13, N13, N10 WAS PC NETWORK, C1 WAS 47P, G1 WAS G0, J.C. IS NOW 105T050, J20, 7 WAS COLORHILL, J20, 9 WAS FORCAGE, DELETED MICROPROCESSOR (B3) -				
6	Q006	H8 WAS L504J R15 WAS 47 CHM, R16 WAS 75C-M, ADDED 1K04M RESISTOR AT K10.				

ENGINEERING RELEASE
 This revision supersedes all previous versions.
 Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
 The information contained herein is the
 proprietary property of Apple Computer, Inc.
 The possessor agrees to the following:
 (i) To maintain this document in confidence.
 (ii) Not to reproduce or copy it.
 (iii) Not to reveal or publish it in whole or part.

APPLE COMPUTER
CONFIDENTIAL

ITEM	QTY	PART NUMBER	DESCRIPTION
1	1		

DRAWING NUMBER: 050-0039-G SHEET 1 OF 10



SIGNAL SOURCES ARE DESIGNATED BY ALFA AND
PAGE LOCATION

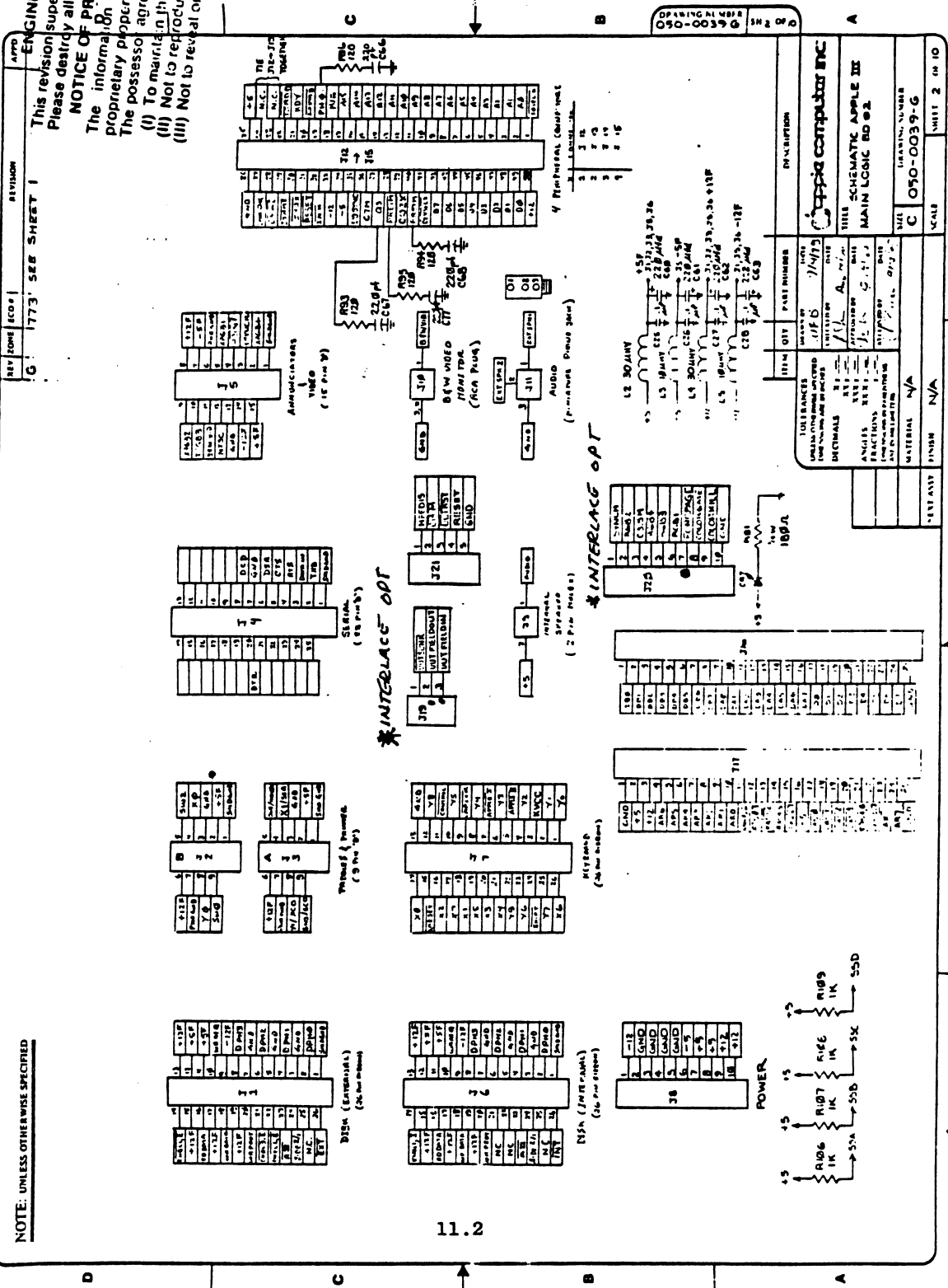
5 B4

ALPHA NOMENCLATURE
LOCATION

Page
5 of 10

**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROTECTION
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(I) To maintain this document in confidence.
(II) Not to reproduce or copy it.
(III) Not to reveal or publish it in whole or in part.



**APPLE COMPUTER
CONFIDENTIAL**

SEE SH11

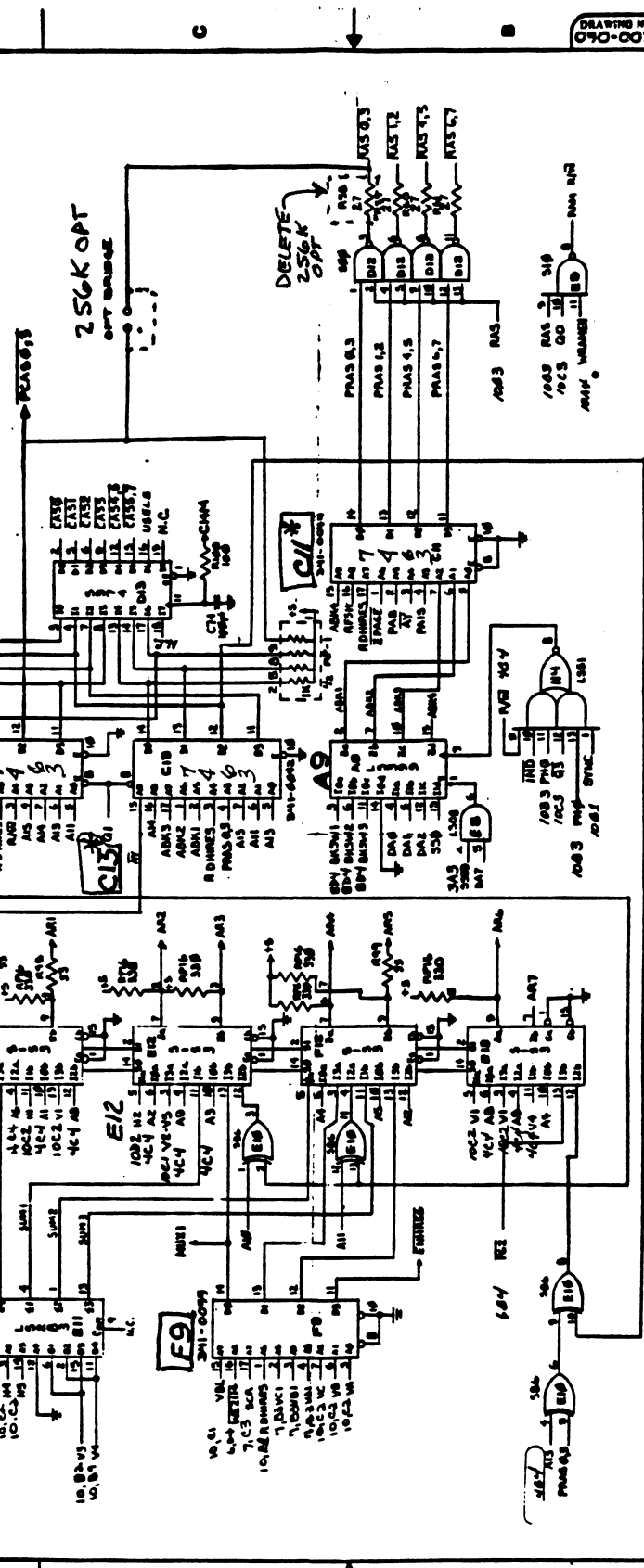
NOTE: UNLESS OTHERWISE SPECIFIED

- F9 - 342-0055 - PROM 1024X4
- C10 - 342-0043 - PROM 1024X4
- C12 - 342-0056 - PROM 65536 ADDRESSING

- C11 - 341-0094 - 12V MEM BRD 8MS65 PROM
- C13 - 341-0061 - 5V MEM BRD 8MS65 PROM
- C13 - 341-0042 - PROM 65536 128-12V MEM
- C13 - 342-0063 - PROM 65536 5V MEM

PRODUCTION RELEASE
This revision supersedes all previous versions.
Please destroy all old copies.

NOTICE OF PROPRIETARY RIGHTS
The information contained herein is the property of Apple Computer, Inc. The possessor agrees to the following:
(i) To maintain this document in confidence.
(ii) Not to reproduce or copy it.
(iii) Not to reveal or publish it in whole or part.



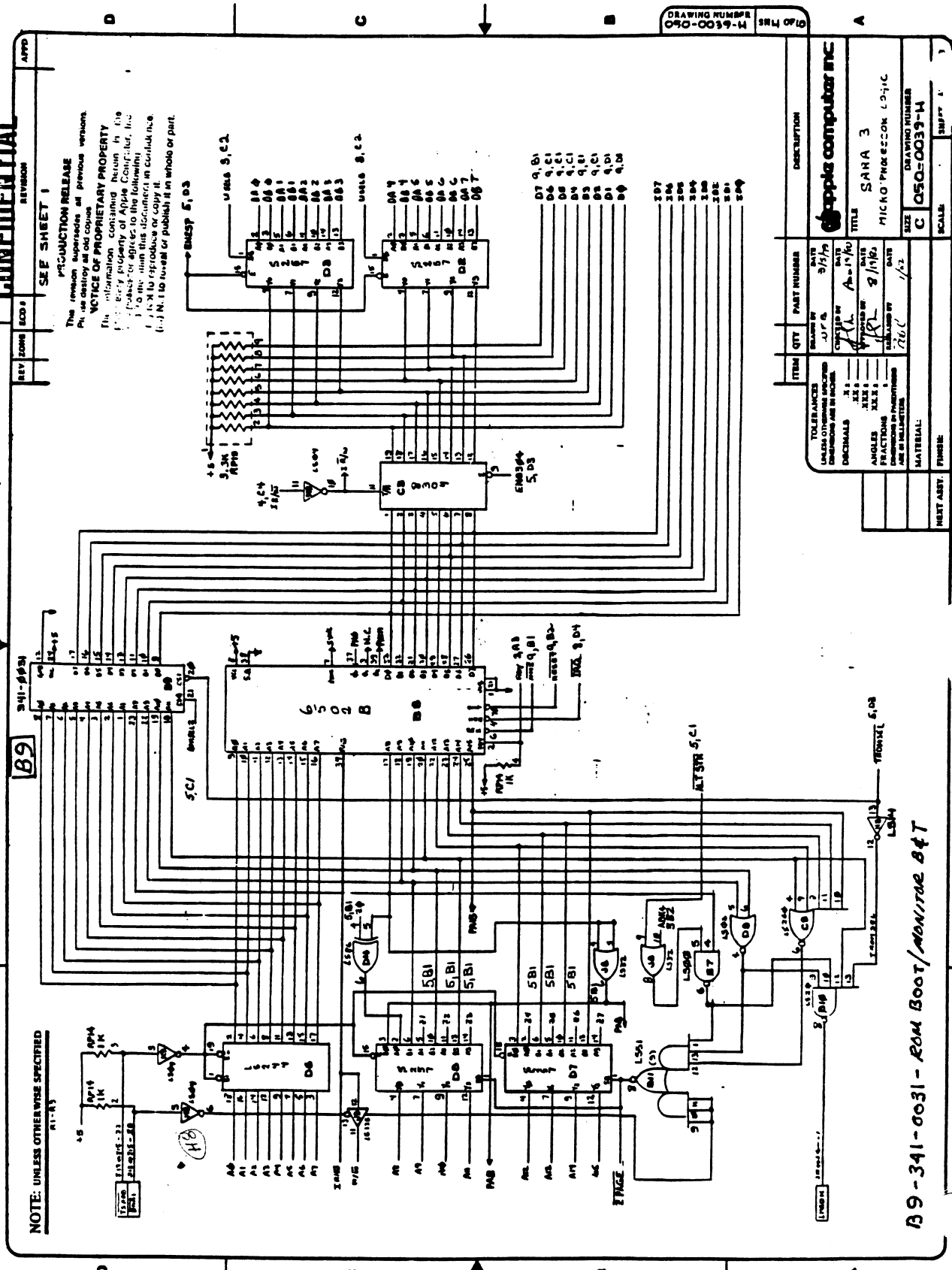
- * MEM BRD
C11 341-0043 341-0061
C13 341-0042 341-0063

ITEM	QTY	PART NUMBER	DATE	DESCRIPTION
TOLERANCES UNLESS OTHERWISE SPECIFIED:				
RESISTORS	1% 5% 10% 20%			
CAPACITORS	5% 10% 20%			
DECIMALS	X1 X2 X3 X4 X5 X6 X7 X8 X9 X0			
ANGLES	FRACTIONS			
MATERIALS: FINISH:				
DRAWING NUMBER: SARA 3 ADDRESS LOGIC				
SCALE: 1				
SHEET 3 OF 10				

**APPLE COMPUTER
CONFIDENTIAL**

REVISION: SEE SHEET 1

PRODUCTION RELEASE
This release supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the
property of Apple Computer, Inc.
It is to be used only for the purposes
indicated in the document in which it
appears. It is not to be reproduced,
copied, or otherwise disseminated
without the express written consent of
Apple Computer, Inc.



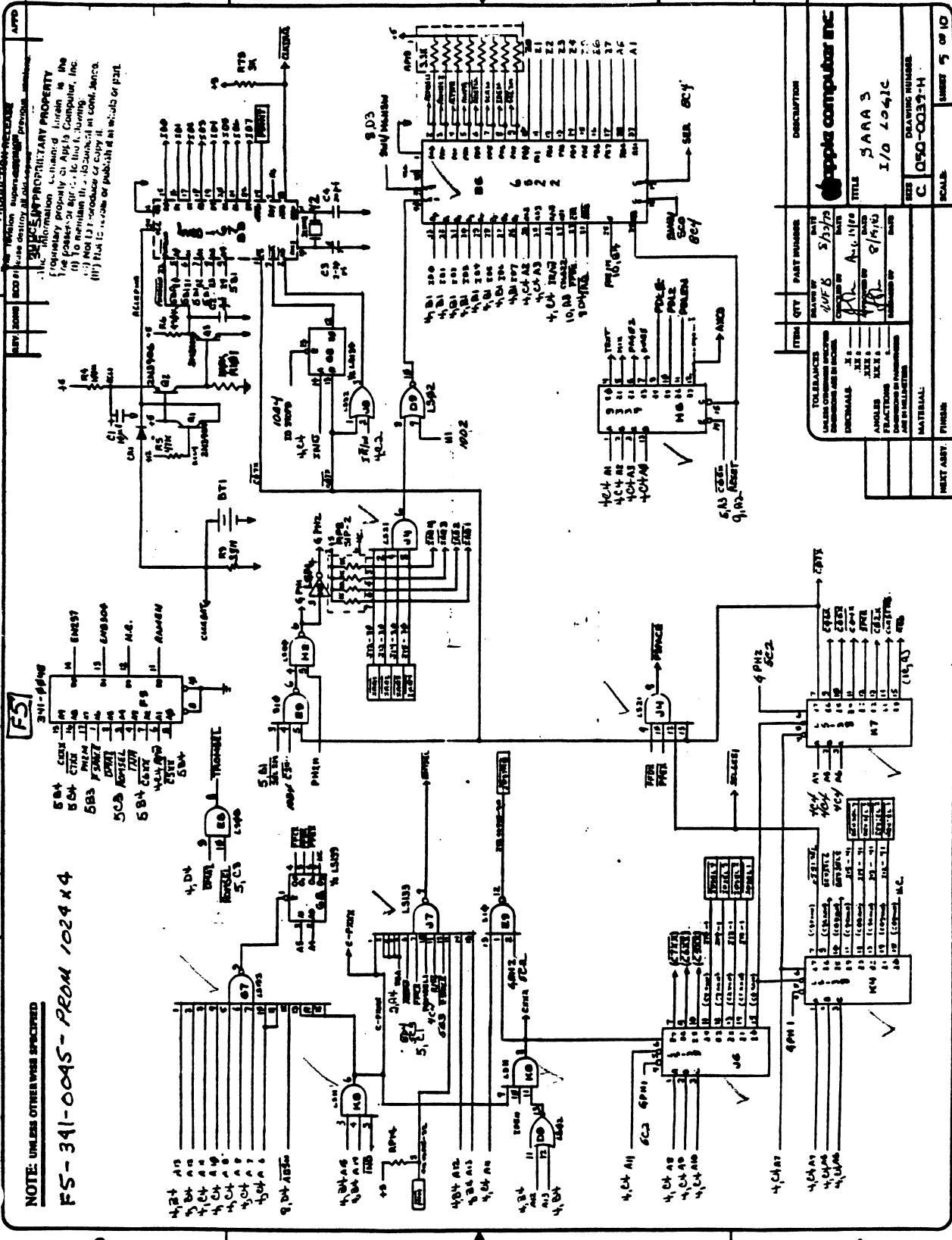
NOTE: UNLESS OTHERWISE SPECIFIED
R1-R3

DRAWING NUMBER: M-6800-060
REV: 01 OF 01

ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	68034 5.03	MICROPROCESSOR
2	1	5B1 5.03	COMPARATOR
3	1	5B1 5.03	COMPARATOR
4	1	5B1 5.03	COMPARATOR
5	1	5B1 5.03	COMPARATOR
6	1	5B1 5.03	COMPARATOR
7	1	5B1 5.03	COMPARATOR
8	1	5B1 5.03	COMPARATOR
9	1	5B1 5.03	COMPARATOR
10	1	5B1 5.03	COMPARATOR
11	1	5B1 5.03	COMPARATOR
12	1	5B1 5.03	COMPARATOR
13	1	5B1 5.03	COMPARATOR
14	1	5B1 5.03	COMPARATOR
15	1	5B1 5.03	COMPARATOR
16	1	5B1 5.03	COMPARATOR
17	1	5B1 5.03	COMPARATOR
18	1	5B1 5.03	COMPARATOR
19	1	5B1 5.03	COMPARATOR
20	1	5B1 5.03	COMPARATOR
21	1	5B1 5.03	COMPARATOR
22	1	5B1 5.03	COMPARATOR
23	1	5B1 5.03	COMPARATOR
24	1	5B1 5.03	COMPARATOR
25	1	5B1 5.03	COMPARATOR
26	1	5B1 5.03	COMPARATOR
27	1	5B1 5.03	COMPARATOR
28	1	5B1 5.03	COMPARATOR
29	1	5B1 5.03	COMPARATOR
30	1	5B1 5.03	COMPARATOR
31	1	5B1 5.03	COMPARATOR
32	1	5B1 5.03	COMPARATOR
33	1	5B1 5.03	COMPARATOR
34	1	5B1 5.03	COMPARATOR
35	1	5B1 5.03	COMPARATOR
36	1	5B1 5.03	COMPARATOR
37	1	5B1 5.03	COMPARATOR
38	1	5B1 5.03	COMPARATOR
39	1	5B1 5.03	COMPARATOR
40	1	5B1 5.03	COMPARATOR
41	1	5B1 5.03	COMPARATOR
42	1	5B1 5.03	COMPARATOR
43	1	5B1 5.03	COMPARATOR
44	1	5B1 5.03	COMPARATOR
45	1	5B1 5.03	COMPARATOR
46	1	5B1 5.03	COMPARATOR
47	1	5B1 5.03	COMPARATOR
48	1	5B1 5.03	COMPARATOR
49	1	5B1 5.03	COMPARATOR
50	1	5B1 5.03	COMPARATOR
51	1	5B1 5.03	COMPARATOR
52	1	5B1 5.03	COMPARATOR
53	1	5B1 5.03	COMPARATOR
54	1	5B1 5.03	COMPARATOR
55	1	5B1 5.03	COMPARATOR
56	1	5B1 5.03	COMPARATOR
57	1	5B1 5.03	COMPARATOR
58	1	5B1 5.03	COMPARATOR
59	1	5B1 5.03	COMPARATOR
60	1	5B1 5.03	COMPARATOR
61	1	5B1 5.03	COMPARATOR
62	1	5B1 5.03	COMPARATOR
63	1	5B1 5.03	COMPARATOR
64	1	5B1 5.03	COMPARATOR
65	1	5B1 5.03	COMPARATOR
66	1	5B1 5.03	COMPARATOR
67	1	5B1 5.03	COMPARATOR
68	1	5B1 5.03	COMPARATOR
69	1	5B1 5.03	COMPARATOR
70	1	5B1 5.03	COMPARATOR
71	1	5B1 5.03	COMPARATOR
72	1	5B1 5.03	COMPARATOR
73	1	5B1 5.03	COMPARATOR
74	1	5B1 5.03	COMPARATOR
75	1	5B1 5.03	COMPARATOR
76	1	5B1 5.03	COMPARATOR
77	1	5B1 5.03	COMPARATOR
78	1	5B1 5.03	COMPARATOR
79	1	5B1 5.03	COMPARATOR
80	1	5B1 5.03	COMPARATOR
81	1	5B1 5.03	COMPARATOR
82	1	5B1 5.03	COMPARATOR
83	1	5B1 5.03	COMPARATOR
84	1	5B1 5.03	COMPARATOR
85	1	5B1 5.03	COMPARATOR
86	1	5B1 5.03	COMPARATOR
87	1	5B1 5.03	COMPARATOR
88	1	5B1 5.03	COMPARATOR
89	1	5B1 5.03	COMPARATOR
90	1	5B1 5.03	COMPARATOR
91	1	5B1 5.03	COMPARATOR
92	1	5B1 5.03	COMPARATOR
93	1	5B1 5.03	COMPARATOR
94	1	5B1 5.03	COMPARATOR
95	1	5B1 5.03	COMPARATOR
96	1	5B1 5.03	COMPARATOR
97	1	5B1 5.03	COMPARATOR
98	1	5B1 5.03	COMPARATOR
99	1	5B1 5.03	COMPARATOR
100	1	5B1 5.03	COMPARATOR

B9-341-0031 - ROM BOOT/MONITOR B&T

APPLE COMP CONFIDENTIAL

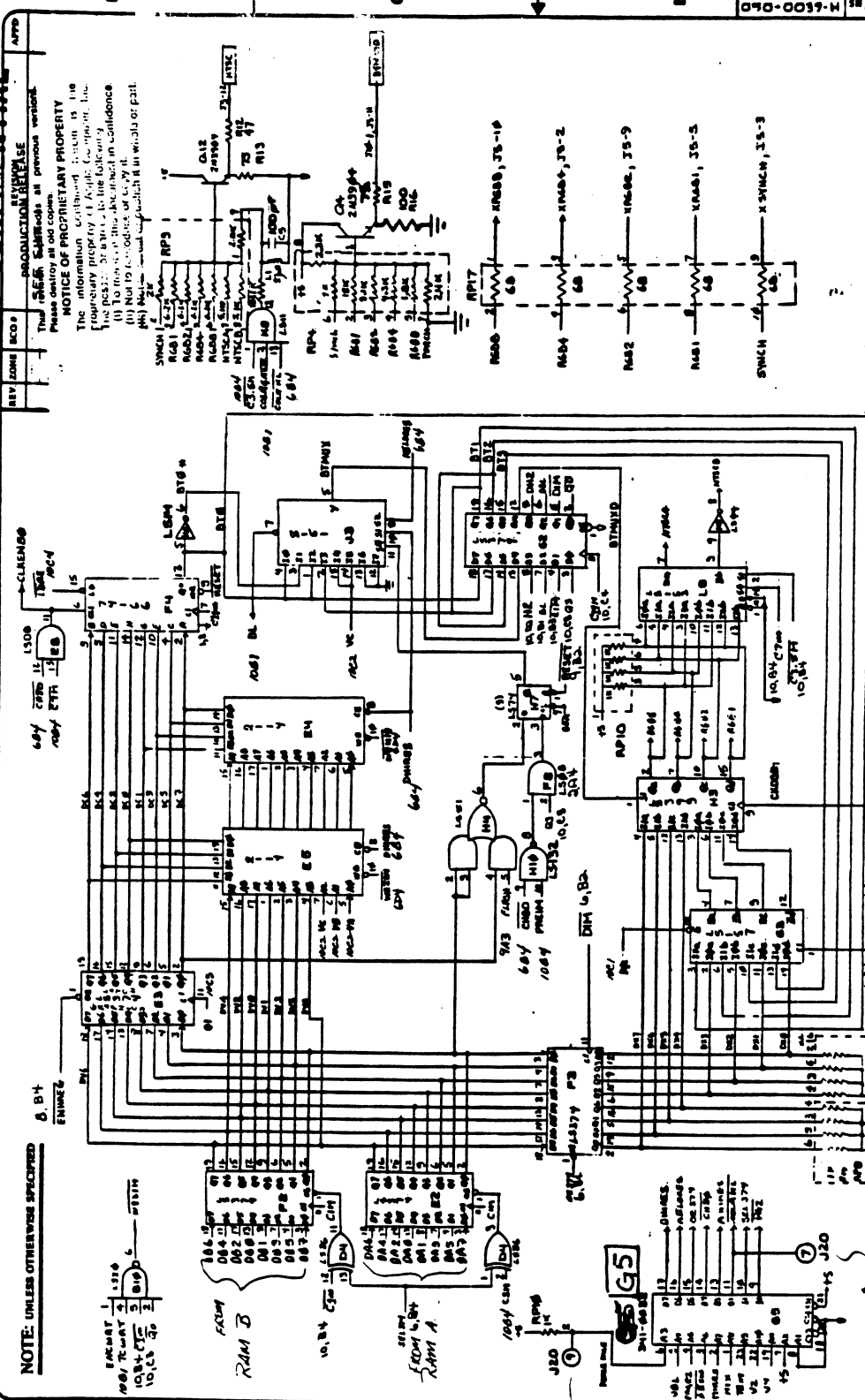


NOTE: UNLESS OTHERWISE SPECIFIED

FS-341-0045 - PROM 1024 X 4

DRAWING NUMBER A 050-0039-H		DRAWING TITLE SARA 3 I/O LOGIC	
SCALE C	SHEET 5	OF 10	
DRAWING NUMBER C 050-0039-H		DRAWING TITLE SARA 3 I/O LOGIC	
SCALE C		SHEET 5	
DRAWING NUMBER C 050-0039-H		DRAWING TITLE SARA 3 I/O LOGIC	
SCALE C		SHEET 5	

**APPLE COMPUTER
CONFIDENTIAL**



NOTE: UNLESS OTHERWISE SPECIFIED
8. BIT
ENVELOPE

PRODUCTION RELEASE
This release is on condition that you agree to the following:
1. You will not disclose the information contained herein to any third party without the prior written consent of Apple Computer, Inc.
2. You will not use the information contained herein for any purpose other than that for which it was provided.
3. You will not reproduce or copy any part of this document without the prior written consent of Apple Computer, Inc.

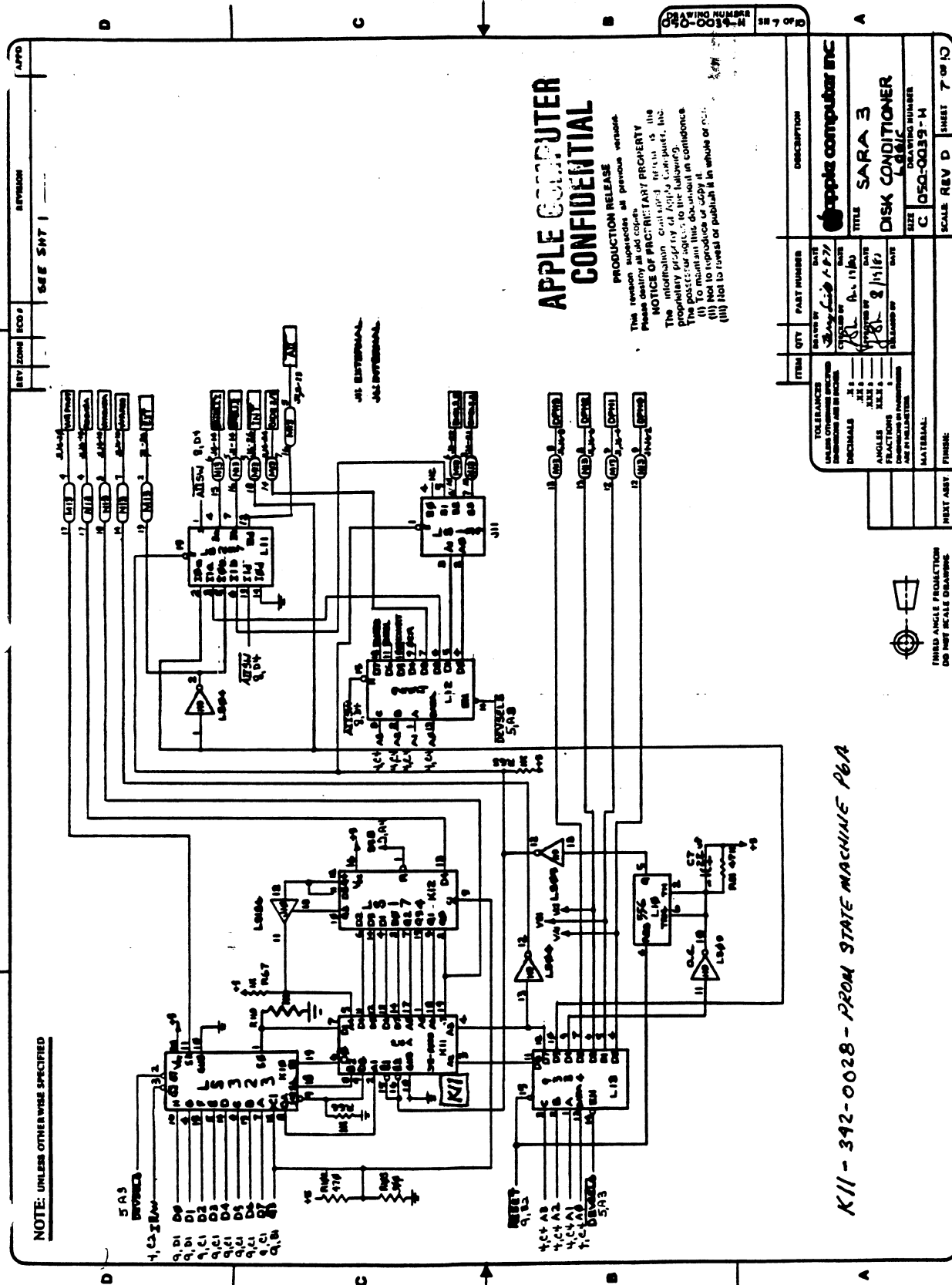
ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	AP17	1N4001, J5-19
2	1	AP17	1N4001, J5-2
3	1	AP17	1N4001, J5-9
4	1	AP17	1N4001, J5-5
5	1	AP17	1N4001, J5-3

REVISION	DATE	DESCRIPTION
1	3/10/79	INITIAL DESIGN
2	4/10/79	REVISED
3	5/10/79	REVISED
4	6/10/79	REVISED
5	7/10/79	REVISED
6	8/10/79	REVISED

G5-341-0032 - VIDEO CONTROL ROM

SEP 8 1979
E5530
SPP

11.6



NOTE: UNLESS OTHERWISE SPECIFIED

**APPLE COMPUTER
CONFIDENTIAL**

PRODUCTION RELEASE
This version supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the
proprietary property of Apple Computer, Inc.
The possessor of this document in confidence
(i) To manufacture or copy it
(ii) Not to reproduce or copy it
(iii) Not to reveal or publish it in whole or in part.

ITEM	QTY	PART NUMBER	DESCRIPTION
TOLERANCES UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES DECIMALS FRACTIONS ANGLES DIMENSIONS IN PARENTHESES ARE BY MANUFACTURER MATERIAL FINISH			
DATE	BY	DATE	DATE
1-8-78	W. J. P.	1-8-78	1-8-78
DESIGNED BY	APPROVED BY	DATE	DATE
W. J. P.	W. J. P.	1-8-78	1-8-78
DRAWN BY	DATE	SCALE	REV D
W. J. P.	1-8-78	1:1	1
DRAWING NUMBER		SHEET 7 OF 10	
K11-392-0028-H			

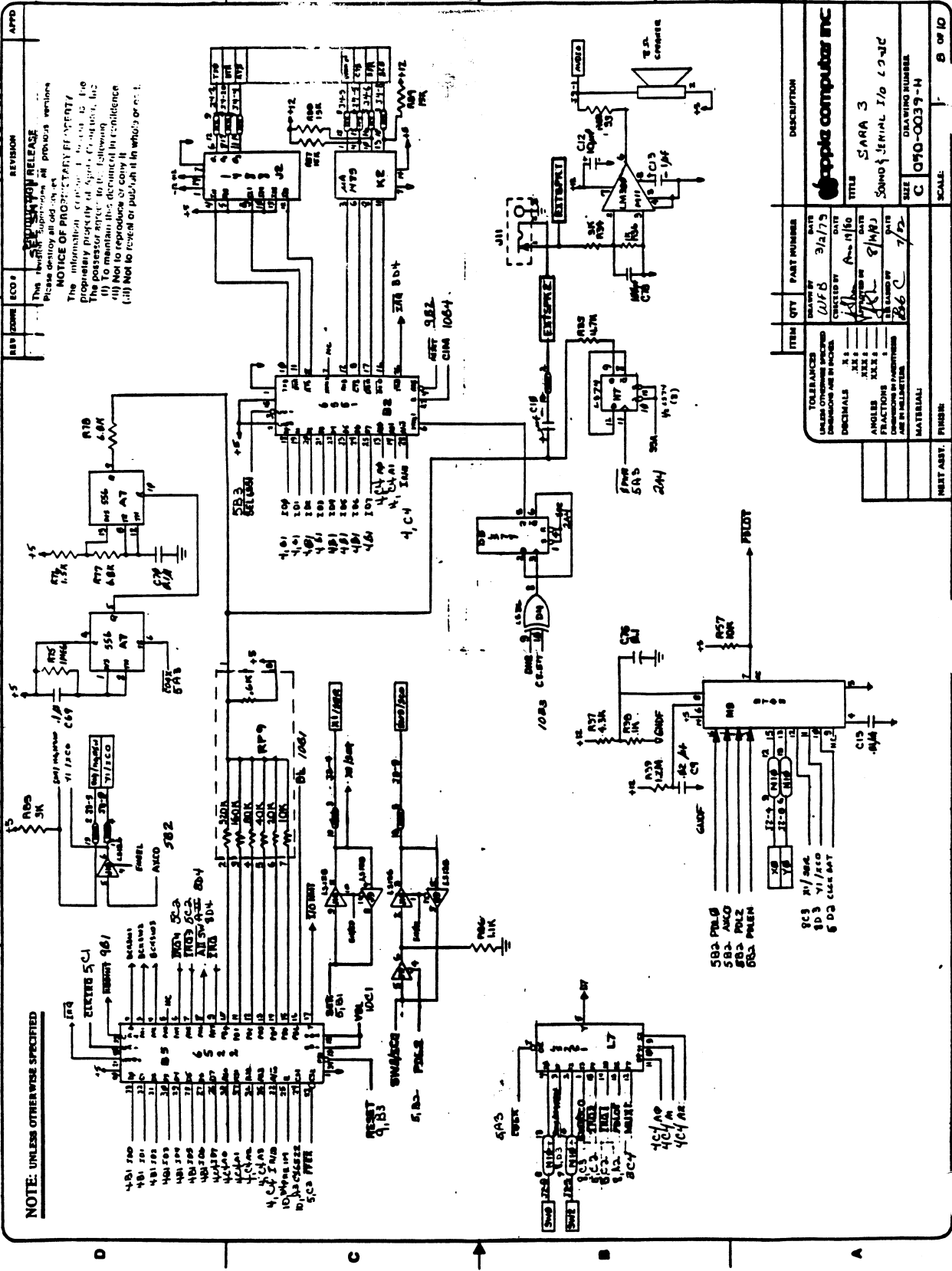
K11-392-0028 - PROM STATE MACHINE P6A



APPLE CONFIDENTIAL

NOTICE OF PROPRIETARY RIGHTS
 The information contained herein is the property of Apple Computer, Inc. It is loaned to you in confidence. It is not to be reproduced or copied in whole or in part.
 (i) Not to be used for advertising or promotional purposes.
 (ii) Not to be used for resale.

NOTE: UNLESS OTHERWISE SPECIFIED



ECO 1 1
 ECO 2 2
 ECO 3 3
 ECO 4 4

1 2 3 4

ITEM		QTY	PART NUMBER	DESCRIPTION
503	PBL9	1		
504	AKG	1		
505	PBL2	1		
506	PBLN	1		
507	MK	1		
508	9C3	1	31/73	
509	5B3	1	Apple-010	
510	5B2	1	Apple-010	
511	5B1	1	Apple-010	
512	5B4	1	Apple-010	
513	5B5	1	Apple-010	
514	5B6	1	Apple-010	
515	5B7	1	Apple-010	
516	5B8	1	Apple-010	
517	5B9	1	Apple-010	
518	5B10	1	Apple-010	
519	5B11	1	Apple-010	
520	5B12	1	Apple-010	
521	5B13	1	Apple-010	
522	5B14	1	Apple-010	
523	5B15	1	Apple-010	
524	5B16	1	Apple-010	
525	5B17	1	Apple-010	
526	5B18	1	Apple-010	
527	5B19	1	Apple-010	
528	5B20	1	Apple-010	
529	5B21	1	Apple-010	
530	5B22	1	Apple-010	

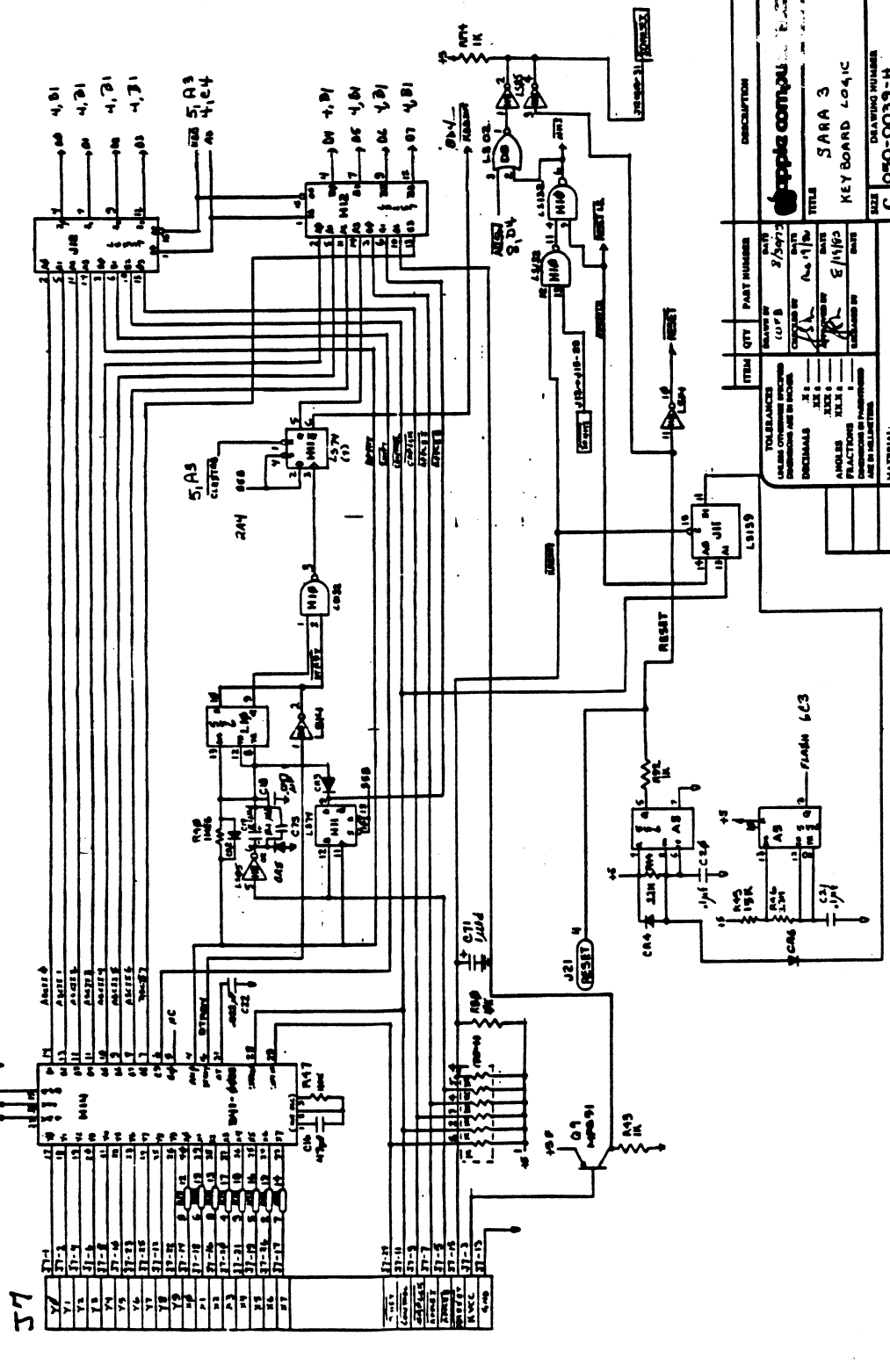
apple computer inc
 TITLE: SARA 3
 SERIAL: 1/0 C-1C
 SIZE: C
 DRAWING NUMBER: 070-0039-N
 SCALE: 8 OF 10

**APPLE COMPUTER
CONFIDENTIAL**

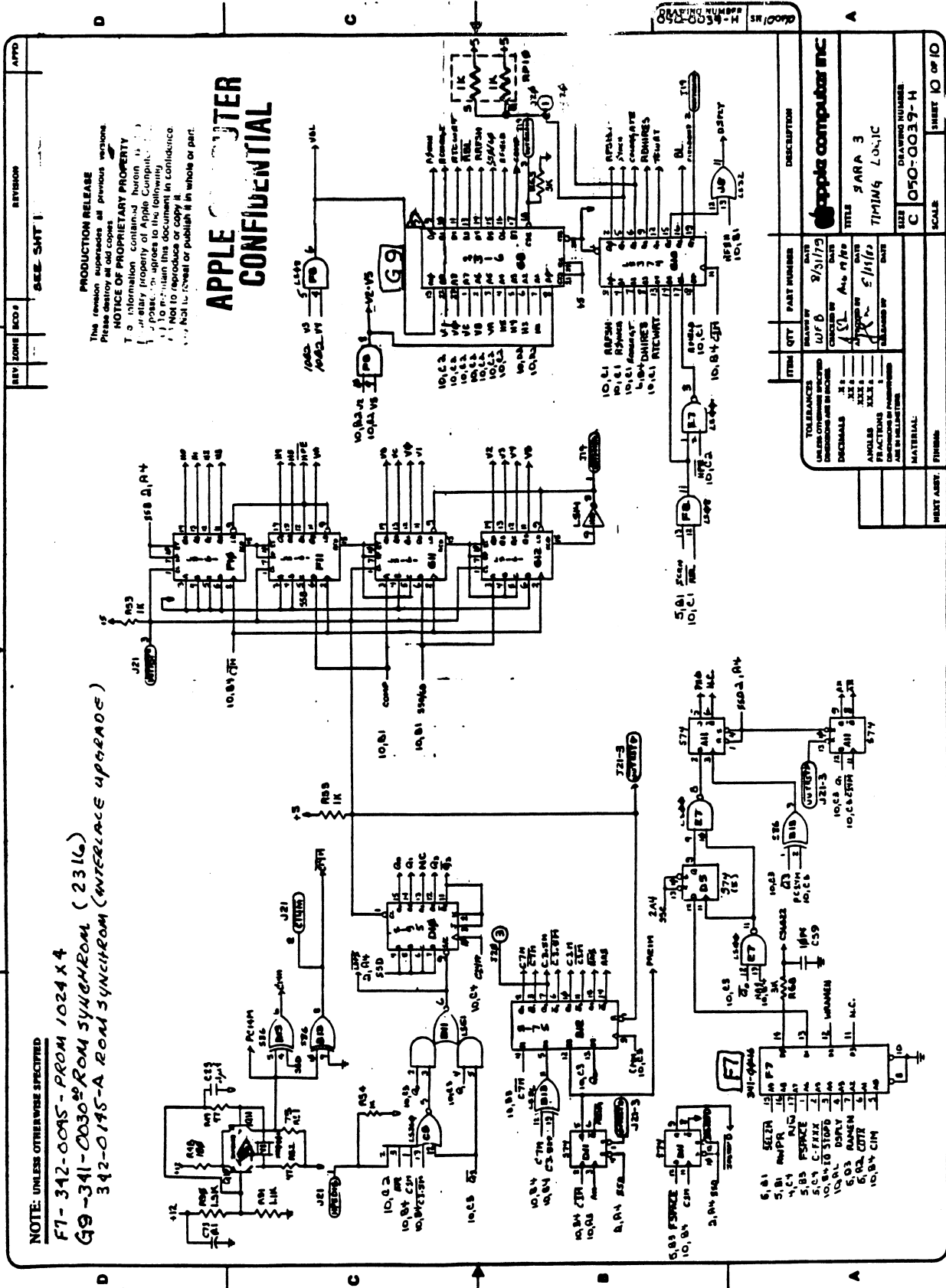
REVISION
PRODUCTION RELEASE
 This technology is Apple's proprietary information. It is not to be disclosed, copied, or reproduced in any form without the express written permission of Apple Computer, Inc. The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
 (i) To maintain this document in confidence.
 (ii) Not to reproduce or disclose its contents.
 (iii) Not to retransmit or publish it in whole or in part.

H14 - 342-0035 - ROM KEYBOARD ENCODER A3

NOTE: UNLESS OTHERWISE SPECIFIED



ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	7414	HEX INVERTER
2	1	7401	NAND GATE
3	1	7404	INVERTER
4	1	7400	NAND GATE
5	1	7402	NAND GATE
6	1	7403	NAND GATE
7	1	7404	INVERTER
8	1	7405	NAND GATE
9	1	7406	NAND GATE
10	1	7407	NAND GATE
11	1	7408	NAND GATE
12	1	7409	NAND GATE
13	1	7410	NAND GATE
14	1	7411	NAND GATE
15	1	7412	NAND GATE
16	1	7413	NAND GATE
17	1	7414	NAND GATE
18	1	7415	NAND GATE
19	1	7416	NAND GATE
20	1	7417	NAND GATE
21	1	7418	NAND GATE
22	1	7419	NAND GATE
23	1	7420	NAND GATE
24	1	7421	NAND GATE
25	1	7422	NAND GATE
26	1	7423	NAND GATE
27	1	7424	NAND GATE
28	1	7425	NAND GATE
29	1	7426	NAND GATE
30	1	7427	NAND GATE
31	1	7428	NAND GATE
32	1	7429	NAND GATE
33	1	7430	NAND GATE
34	1	7431	NAND GATE
35	1	7432	NAND GATE
36	1	7433	NAND GATE
37	1	7434	NAND GATE
38	1	7435	NAND GATE
39	1	7436	NAND GATE
40	1	7437	NAND GATE
41	1	7438	NAND GATE
42	1	7439	NAND GATE
43	1	7440	NAND GATE
44	1	7441	NAND GATE
45	1	7442	NAND GATE
46	1	7443	NAND GATE
47	1	7444	NAND GATE
48	1	7445	NAND GATE
49	1	7446	NAND GATE
50	1	7447	NAND GATE
51	1	7448	NAND GATE
52	1	7449	NAND GATE
53	1	7450	NAND GATE
54	1	7451	NAND GATE
55	1	7452	NAND GATE
56	1	7453	NAND GATE
57	1	7454	NAND GATE
58	1	7455	NAND GATE
59	1	7456	NAND GATE
60	1	7457	NAND GATE
61	1	7458	NAND GATE
62	1	7459	NAND GATE
63	1	7460	NAND GATE
64	1	7461	NAND GATE
65	1	7462	NAND GATE
66	1	7463	NAND GATE
67	1	7464	NAND GATE
68	1	7465	NAND GATE
69	1	7466	NAND GATE
70	1	7467	NAND GATE
71	1	7468	NAND GATE
72	1	7469	NAND GATE
73	1	7470	NAND GATE
74	1	7471	NAND GATE
75	1	7472	NAND GATE
76	1	7473	NAND GATE
77	1	7474	NAND GATE
78	1	7475	NAND GATE
79	1	7476	NAND GATE
80	1	7477	NAND GATE
81	1	7478	NAND GATE
82	1	7479	NAND GATE
83	1	7480	NAND GATE
84	1	7481	NAND GATE
85	1	7482	NAND GATE
86	1	7483	NAND GATE
87	1	7484	NAND GATE
88	1	7485	NAND GATE
89	1	7486	NAND GATE
90	1	7487	NAND GATE
91	1	7488	NAND GATE
92	1	7489	NAND GATE
93	1	7490	NAND GATE
94	1	7491	NAND GATE
95	1	7492	NAND GATE
96	1	7493	NAND GATE
97	1	7494	NAND GATE
98	1	7495	NAND GATE
99	1	7496	NAND GATE
100	1	7497	NAND GATE
101	1	7498	NAND GATE
102	1	7499	NAND GATE
103	1	7500	NAND GATE



NOTE: UNLESS OTHERWISE SPECIFIED

F7- 342-00A5 - PROM 1024 X 4

G9- 341-0030 ROM SYNCHROM (23LG)

342-01A5-A ROM SYNCHROM (INTERLACE UPGRADE)

PRODUCTION RELEASE
 This revision supersedes all previous revisions.
 Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
 The information contained herein is the property of Apple Computer, Inc. and is to be used only for the purposes specified herein. It is not to be reproduced or copied in confidence, in whole or in part.

APPLE COMPUTER
CONFIDENTIAL

ITEM	QTY	PART NUMBER	DESCRIPTION
10.01	1	80254	ROM
10.02	1	80254	ROM
10.03	1	80254	ROM
10.04	1	80254	ROM
10.05	1	80254	ROM
10.06	1	80254	ROM
10.07	1	80254	ROM
10.08	1	80254	ROM
10.09	1	80254	ROM
10.10	1	80254	ROM
10.11	1	80254	ROM
10.12	1	80254	ROM
10.13	1	80254	ROM
10.14	1	80254	ROM
10.15	1	80254	ROM
10.16	1	80254	ROM
10.17	1	80254	ROM
10.18	1	80254	ROM
10.19	1	80254	ROM
10.20	1	80254	ROM
10.21	1	80254	ROM
10.22	1	80254	ROM
10.23	1	80254	ROM
10.24	1	80254	ROM
10.25	1	80254	ROM
10.26	1	80254	ROM
10.27	1	80254	ROM
10.28	1	80254	ROM
10.29	1	80254	ROM
10.30	1	80254	ROM
10.31	1	80254	ROM
10.32	1	80254	ROM
10.33	1	80254	ROM
10.34	1	80254	ROM
10.35	1	80254	ROM
10.36	1	80254	ROM
10.37	1	80254	ROM
10.38	1	80254	ROM
10.39	1	80254	ROM
10.40	1	80254	ROM
10.41	1	80254	ROM
10.42	1	80254	ROM
10.43	1	80254	ROM
10.44	1	80254	ROM
10.45	1	80254	ROM
10.46	1	80254	ROM
10.47	1	80254	ROM
10.48	1	80254	ROM
10.49	1	80254	ROM
10.50	1	80254	ROM
10.51	1	80254	ROM
10.52	1	80254	ROM
10.53	1	80254	ROM
10.54	1	80254	ROM
10.55	1	80254	ROM
10.56	1	80254	ROM
10.57	1	80254	ROM
10.58	1	80254	ROM
10.59	1	80254	ROM
10.60	1	80254	ROM
10.61	1	80254	ROM
10.62	1	80254	ROM
10.63	1	80254	ROM
10.64	1	80254	ROM
10.65	1	80254	ROM
10.66	1	80254	ROM
10.67	1	80254	ROM
10.68	1	80254	ROM
10.69	1	80254	ROM
10.70	1	80254	ROM
10.71	1	80254	ROM
10.72	1	80254	ROM
10.73	1	80254	ROM
10.74	1	80254	ROM
10.75	1	80254	ROM
10.76	1	80254	ROM
10.77	1	80254	ROM
10.78	1	80254	ROM
10.79	1	80254	ROM
10.80	1	80254	ROM
10.81	1	80254	ROM
10.82	1	80254	ROM
10.83	1	80254	ROM
10.84	1	80254	ROM
10.85	1	80254	ROM
10.86	1	80254	ROM
10.87	1	80254	ROM
10.88	1	80254	ROM
10.89	1	80254	ROM
10.90	1	80254	ROM
10.91	1	80254	ROM
10.92	1	80254	ROM
10.93	1	80254	ROM
10.94	1	80254	ROM
10.95	1	80254	ROM
10.96	1	80254	ROM
10.97	1	80254	ROM
10.98	1	80254	ROM
10.99	1	80254	ROM
10.100	1	80254	ROM

Apple Computer Inc.
 TITLE: SARA 3
 TIMING LOGIC

DATE: 9/3/75
 DRAWN BY: J. L. ...
 CHECKED BY: ...
 DATE: 5/11/80

UNDER OVERHAUL SPECIFIED
 DIMENSIONS ARE IN INCHES
 DECIMALS
 ANGLES
 FRACTIONS
 ARE IN MILLIMETERS

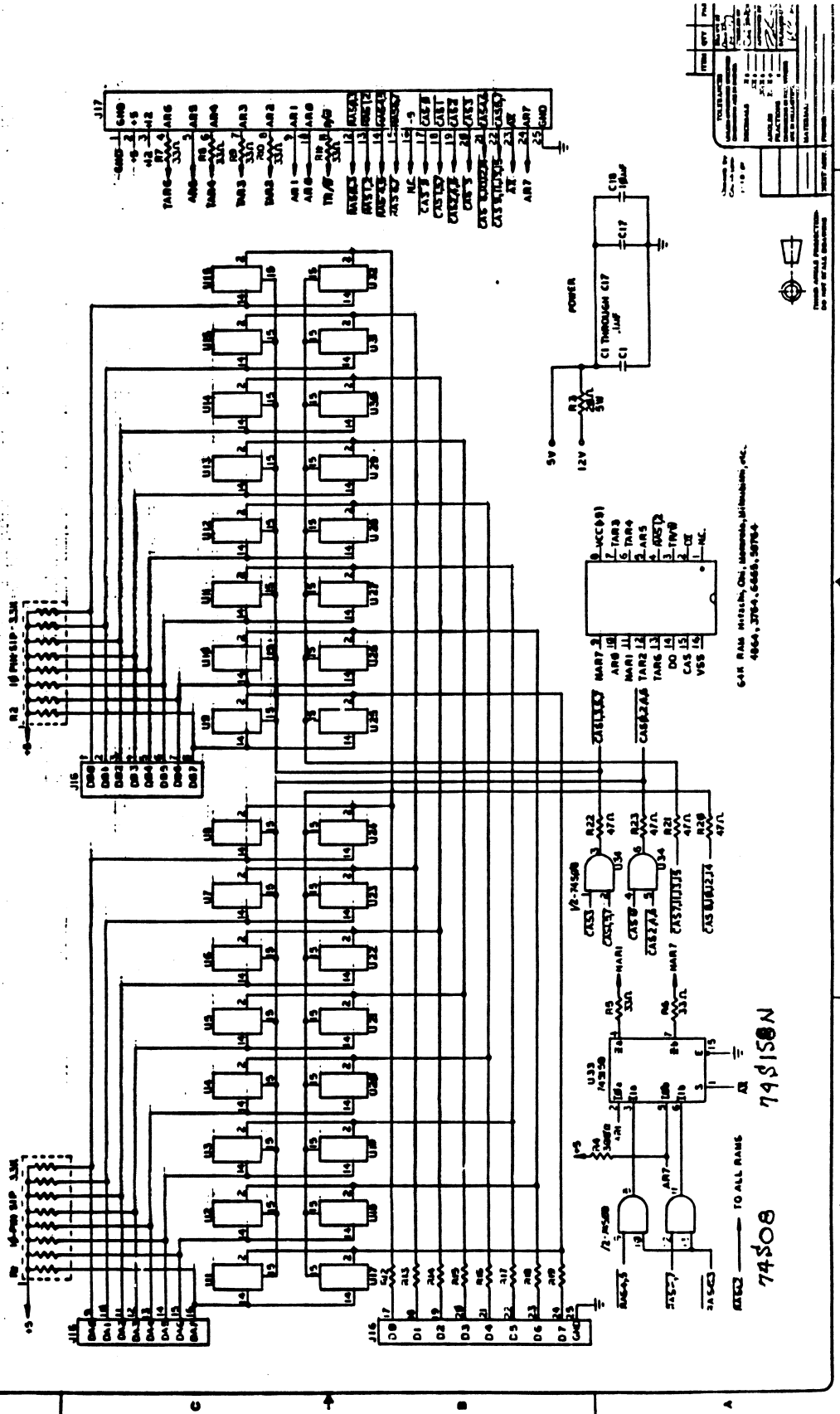
MATERIAL
 FINISH

SCALE
 SHEET 10 OF 10

APPLE /// 5V MEMORY BOARD (PIN 610-8101)

REV	DATE
A	7/74
B	5/78

NOTE: 1. THIS BOARD OPERATES AT 5V.
2. THROUGH BOARD ARE: 100 Ω.

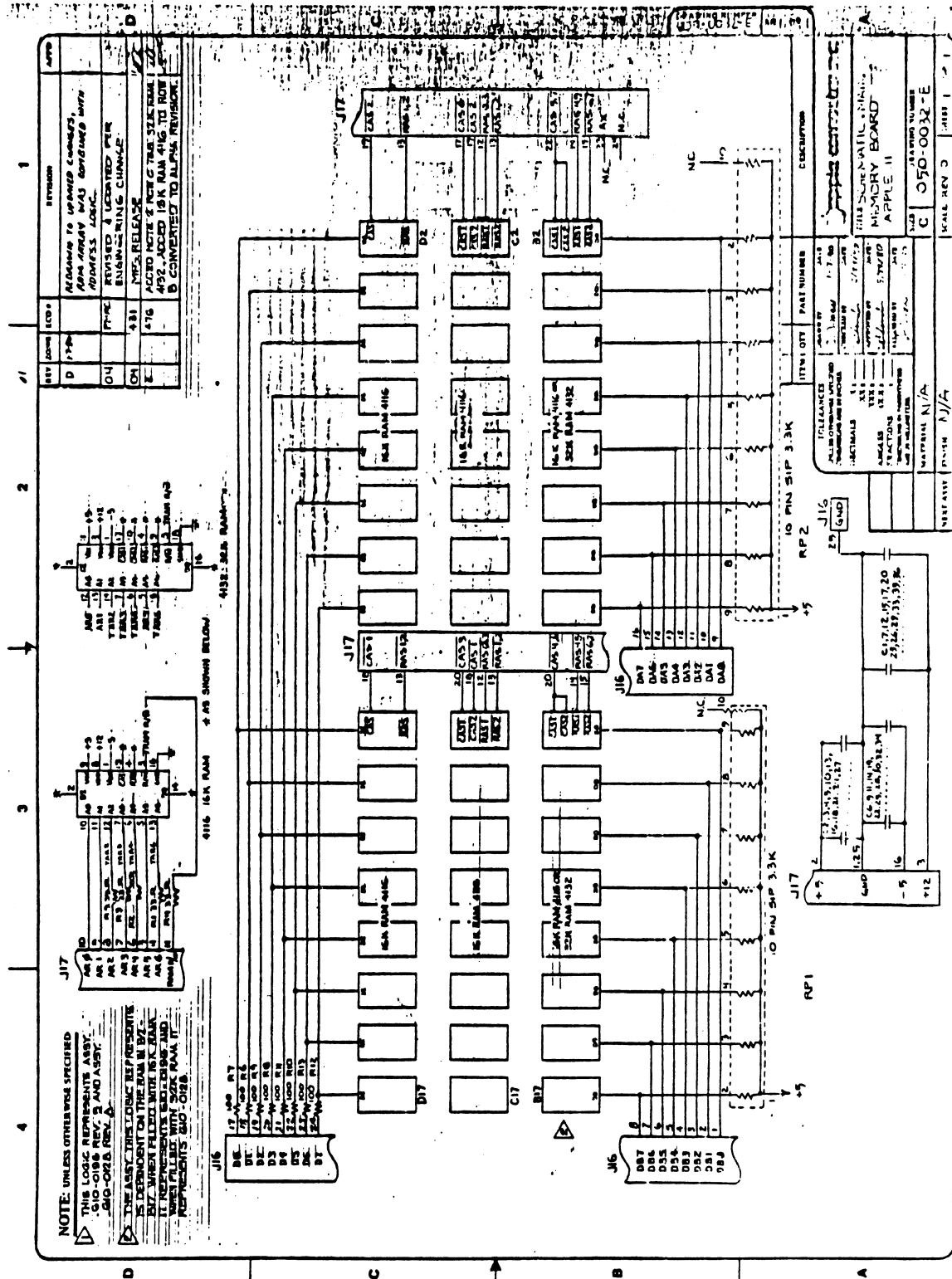


REV	DATE
A	7/74
B	5/78

6411 RAM MEMORY, CHIP, MANUFACTURED BY INTEL CORPORATION, 4004, 7064, 6468A, 50764

The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
 (I) To maintain this document in confidence.
 (II) Not to reproduce or copy it.
 (III) Not to reveal or publish it in whole or part.

APPLE COMPUTER CONFIDENTIAL



11.12

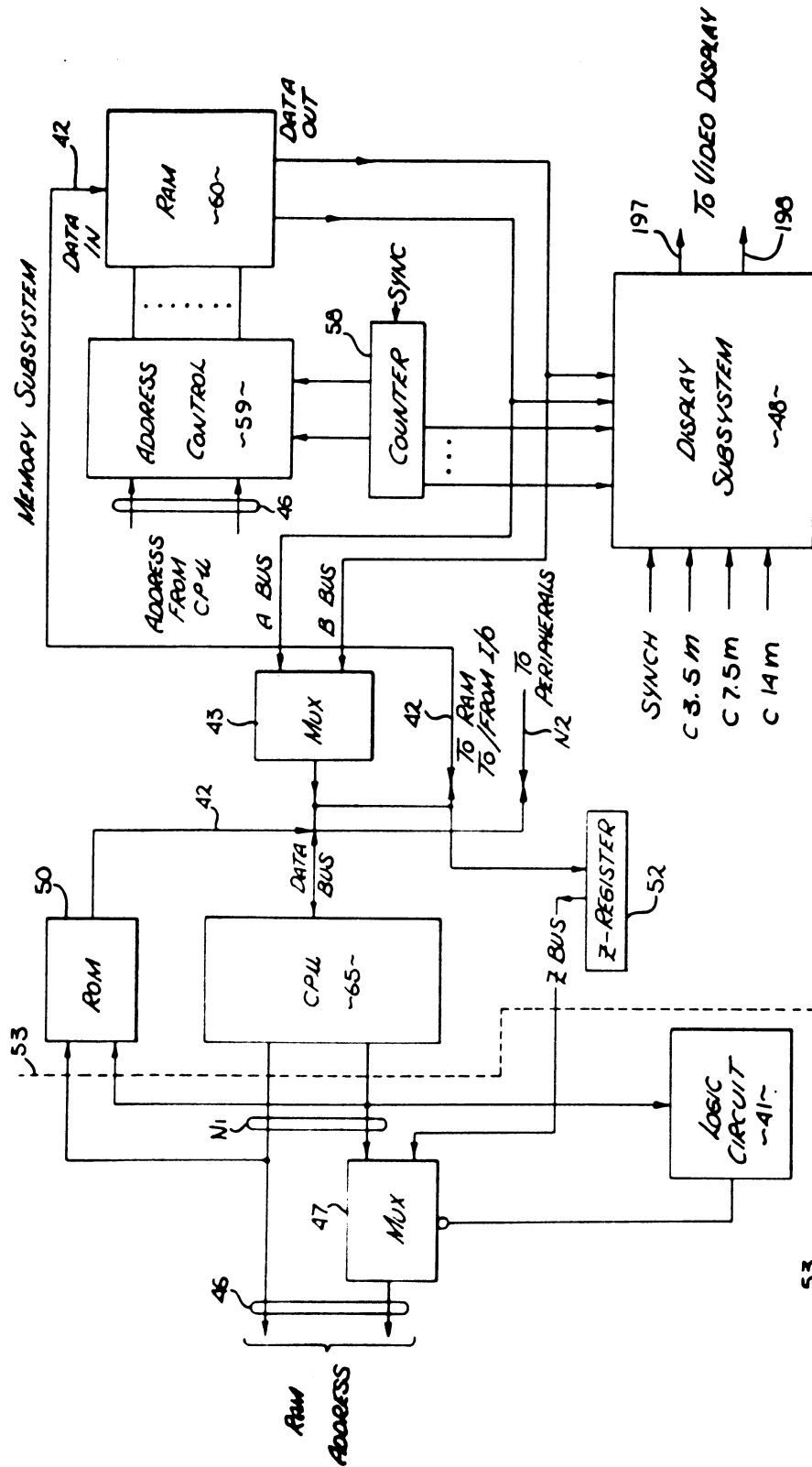
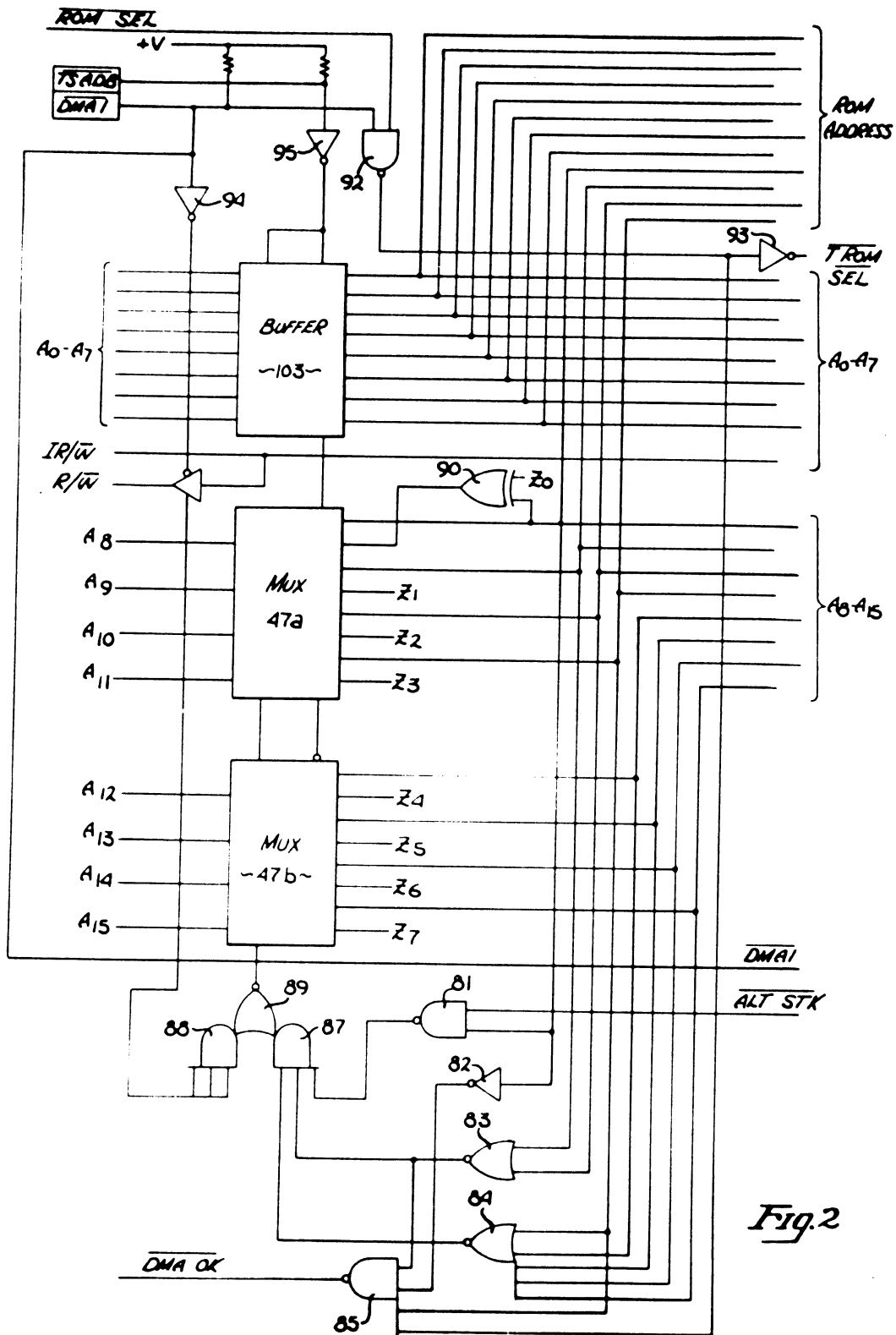


Fig. 2

MAJOR COMPONENTS and SUBSYSTEMS

Source /// PATENT - MAY 1983



ADDRESS and DATA BUS ARCHITECTURE

Source III PATENT - MAY 1983

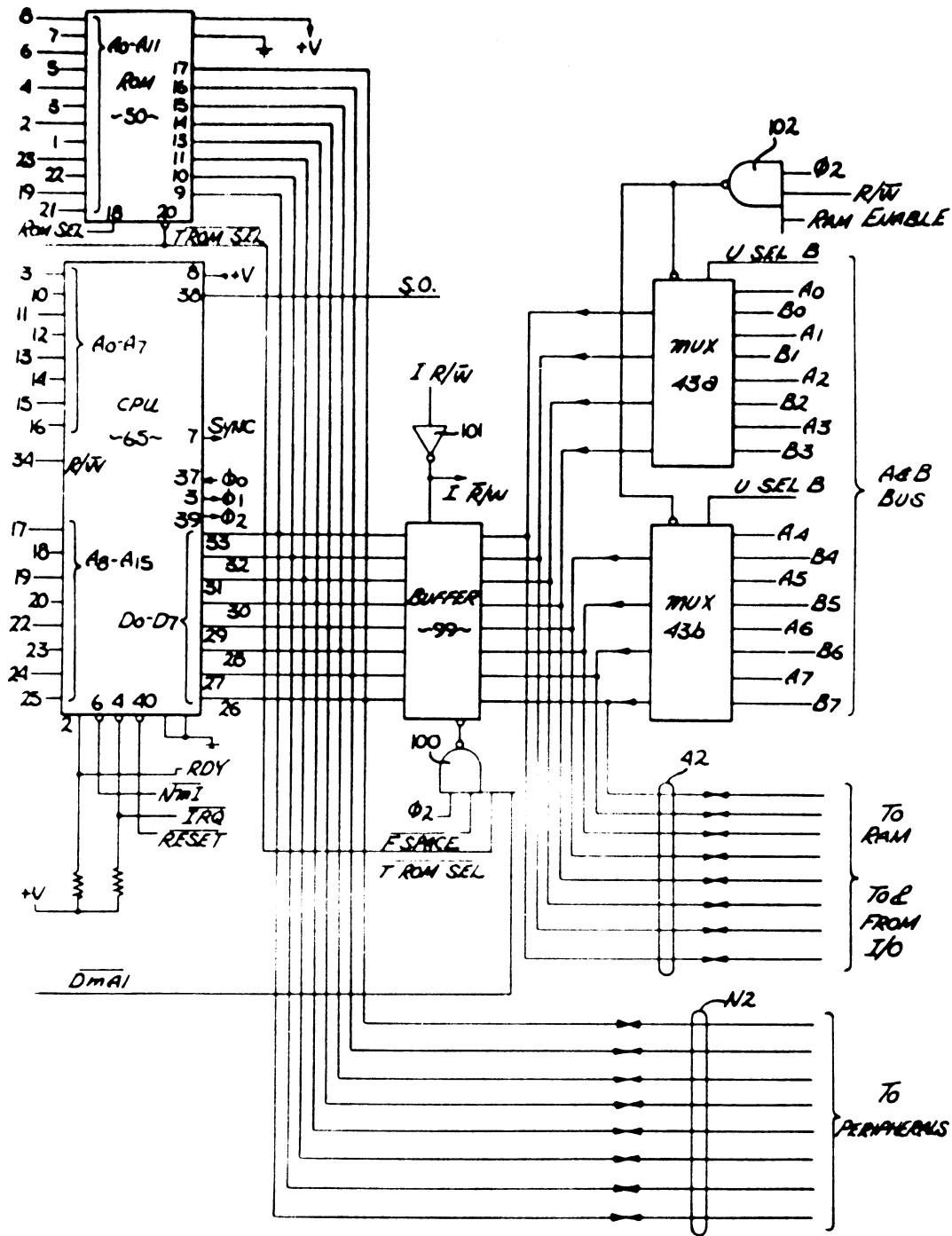


Fig. 3

CPU and ADDRESS/DATA BUS INTERFACE

Source /// PATENT - MAY 1983

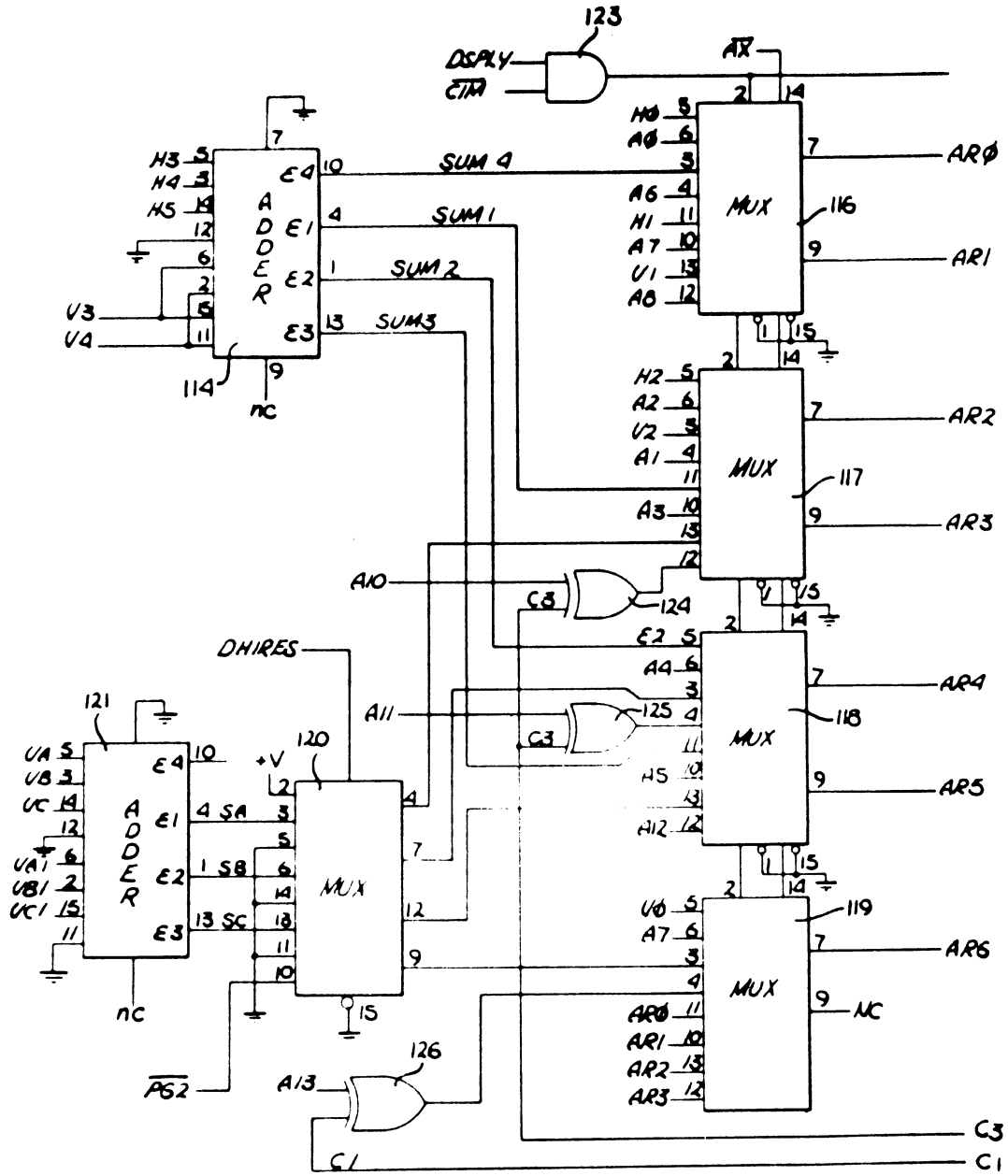
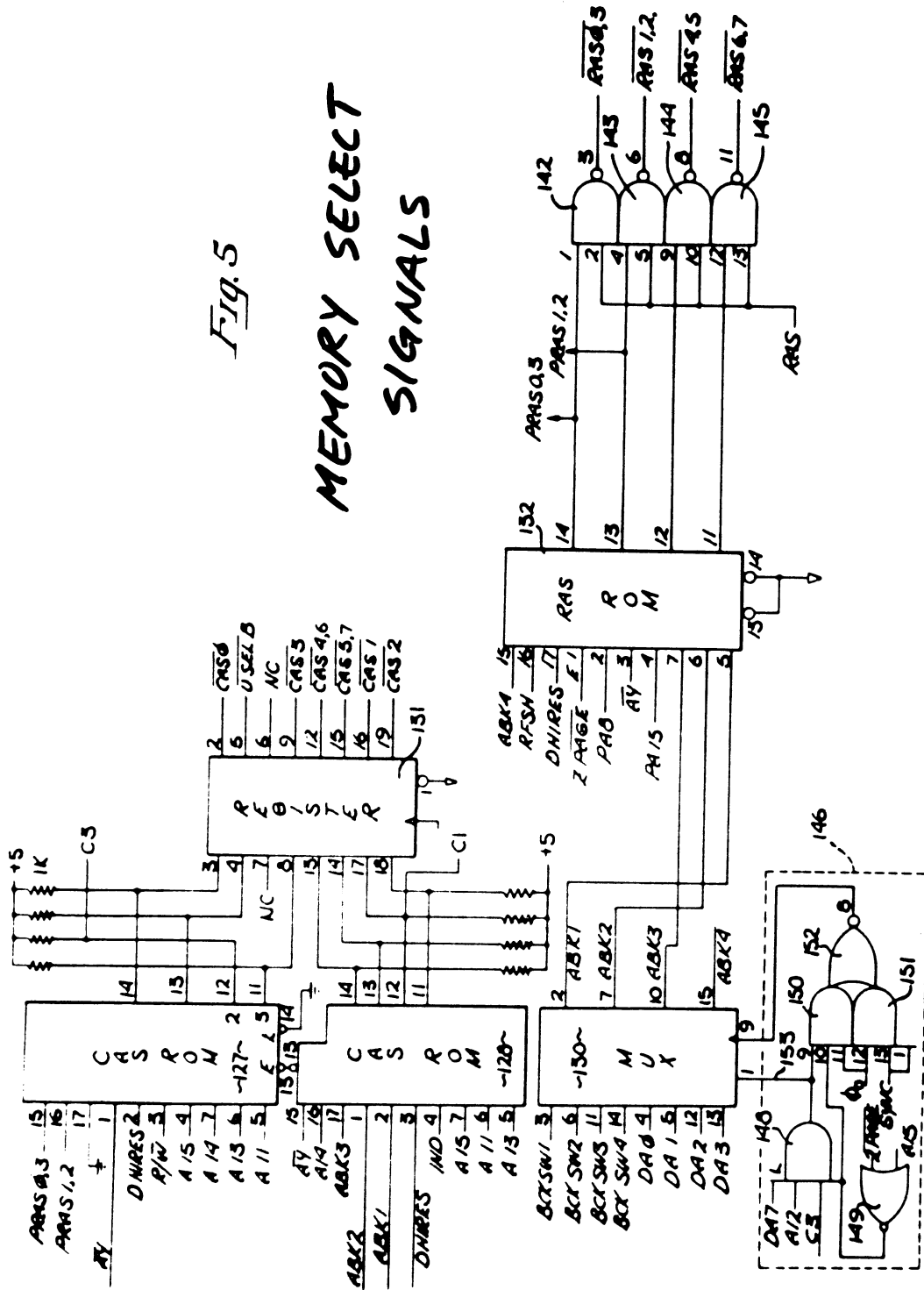


Fig. 4

**ADDRESS BUS and DISPLAY COUNTER
SIGNAL SELECTION CIRCUITRY**

Source III PATENT—MAY 1983

FIG. 5
MEMORY SELECT SIGNALS



Source /// PATENT - MAY 1983

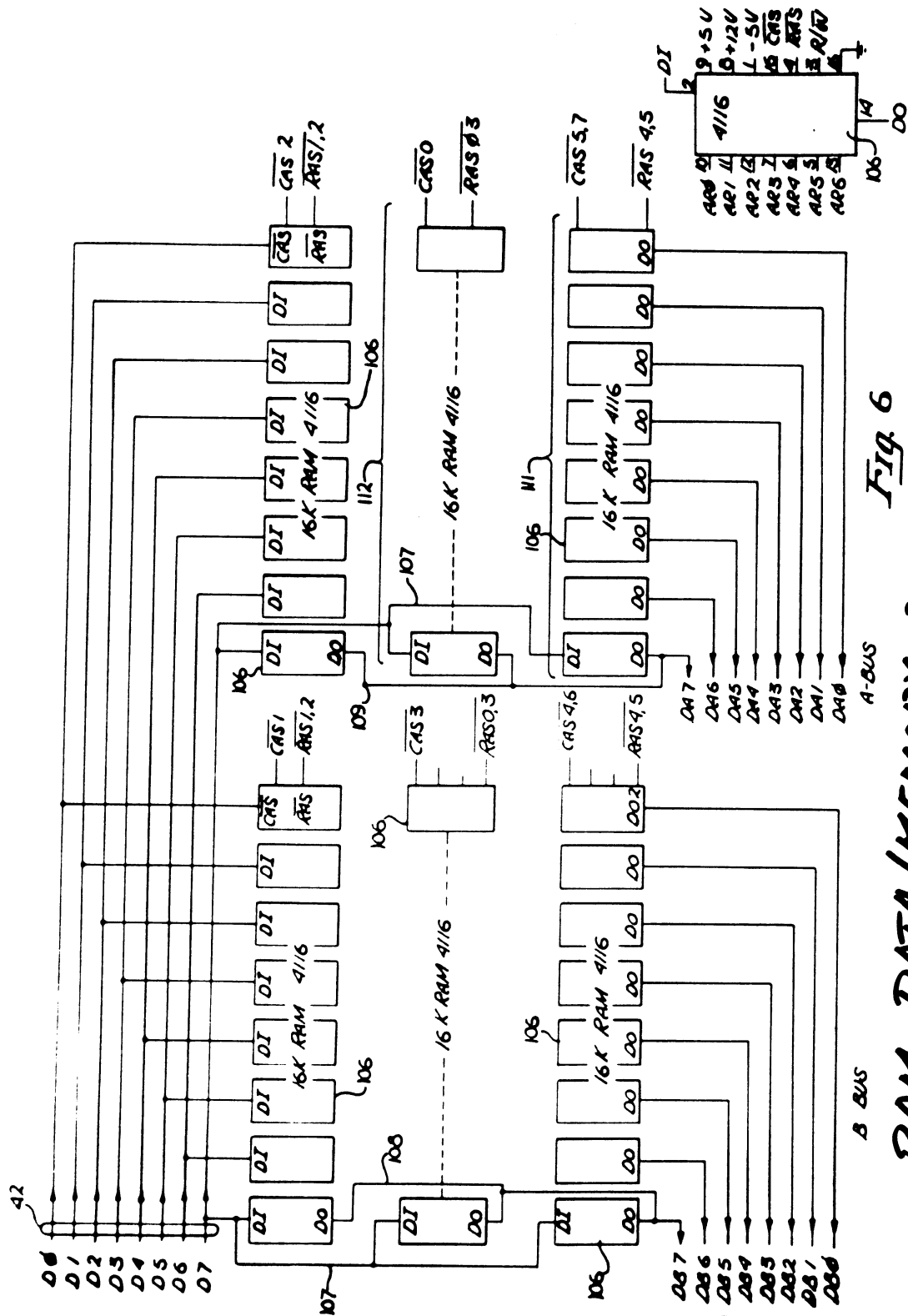


Fig. 6
RAM DATA/MEMORY BUSES

Source III PATENT - MAY 1983

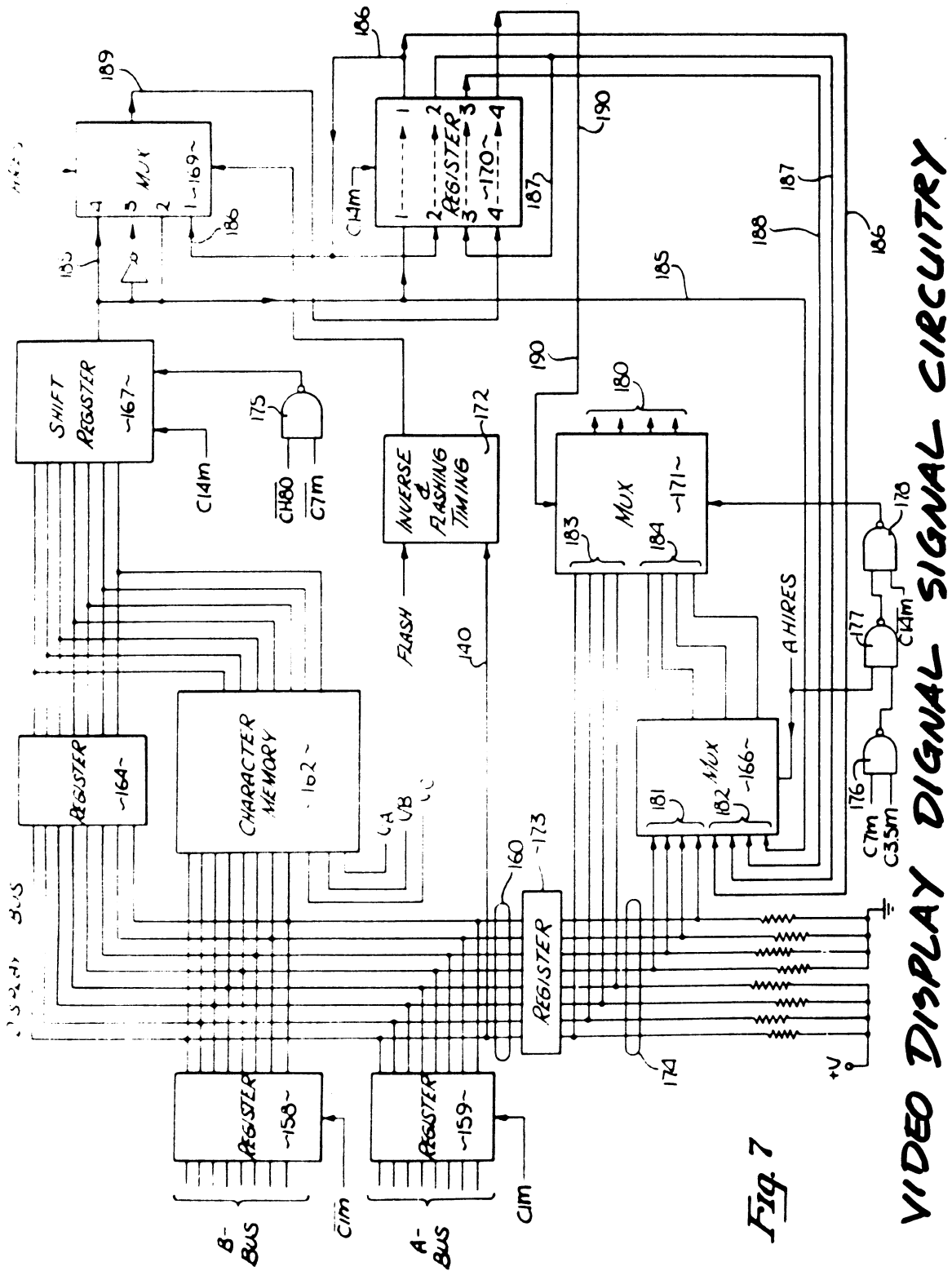


Fig. 7

VIDEO DISPLAY DIGITAL SIGNAL CIRCUITRY

Source III PATENT - MAY 1983

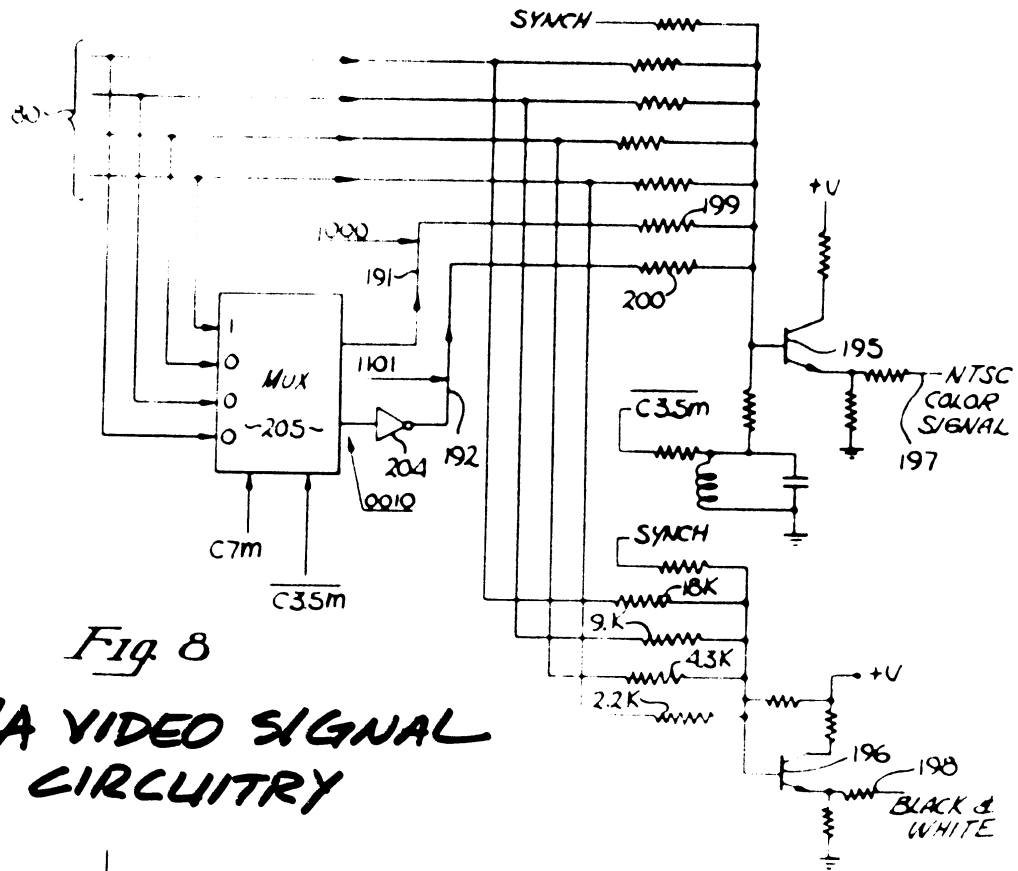


Fig 8
**D/A VIDEO SIGNAL
 CIRCUITRY**

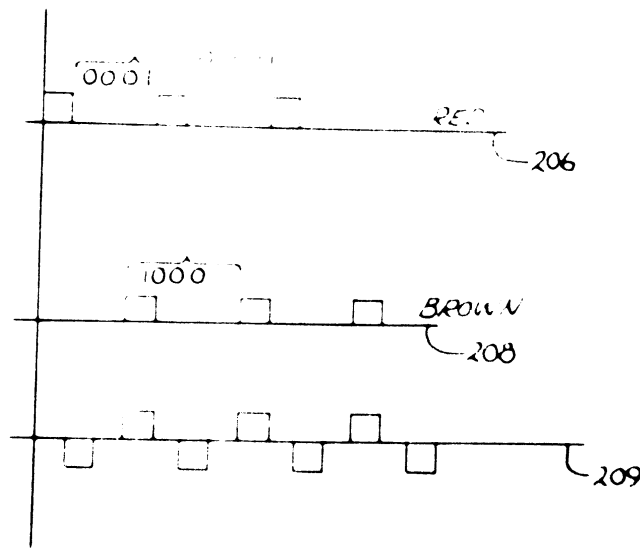


Fig. 9

VIDEO SIGNALS

Source /// PATENT - MAY 1983

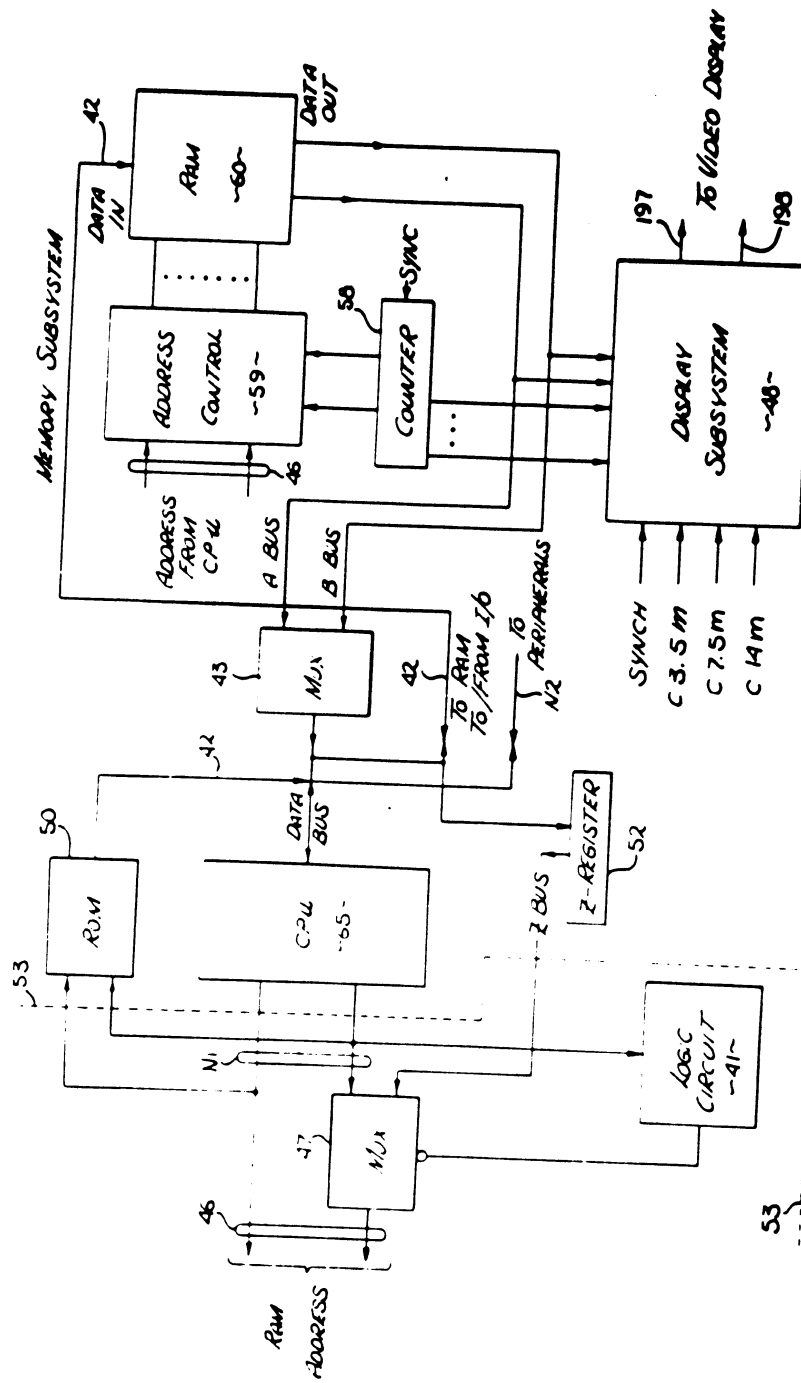


Fig. 1

Source III PATENT - AUG 1985

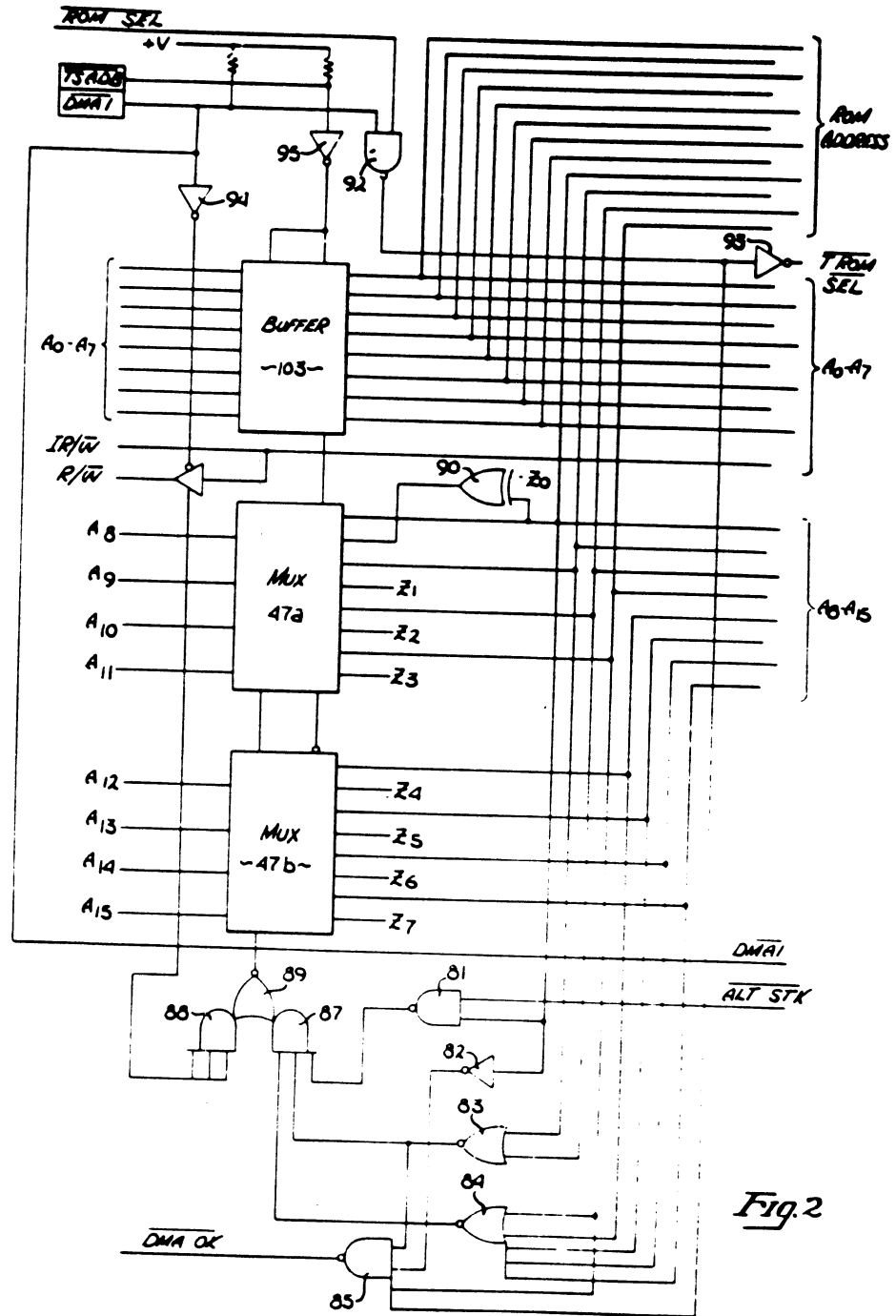


Fig. 2

Source III PATENT—AUG 1985

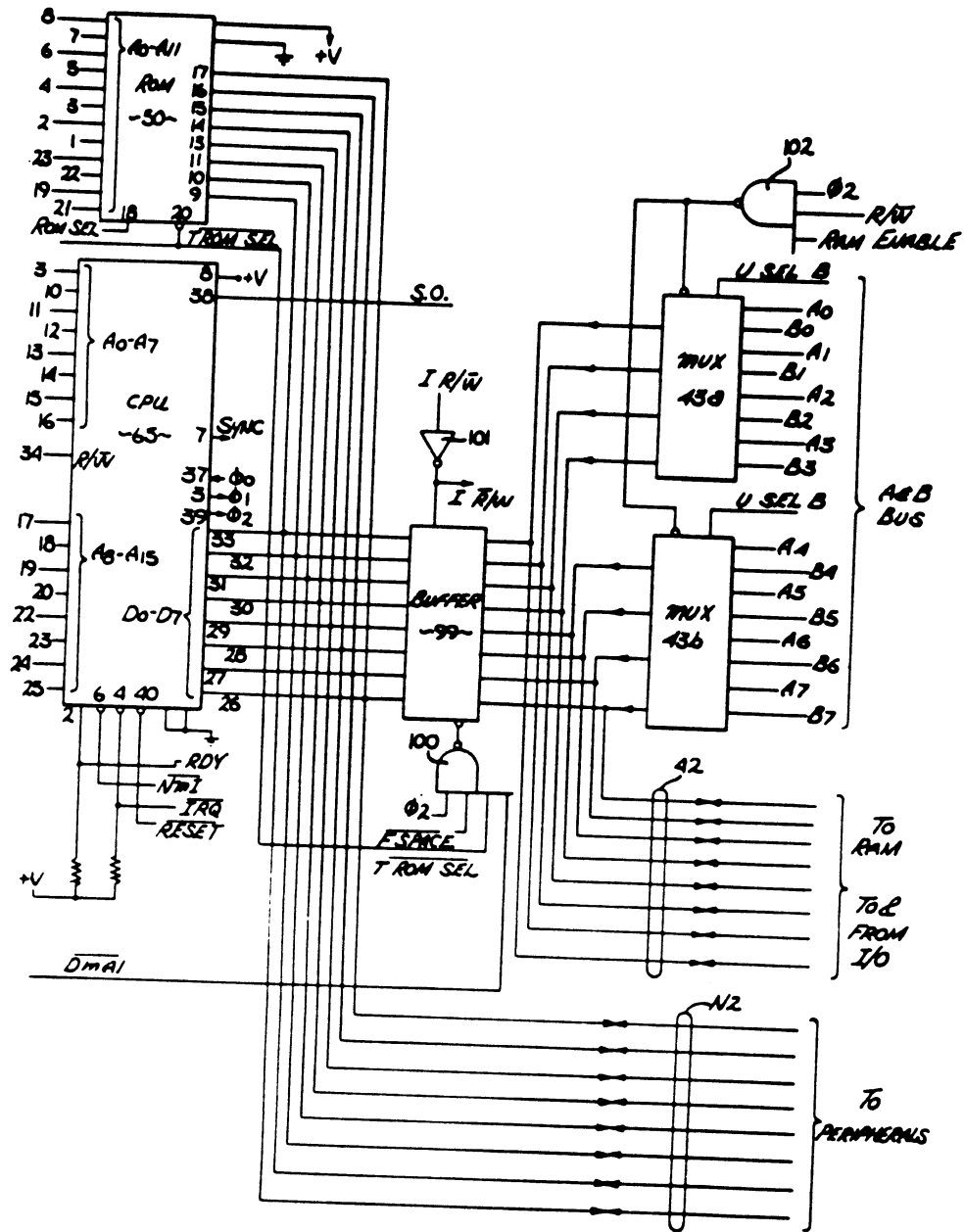


Fig. 3

Source III PATENT—AUG 1985

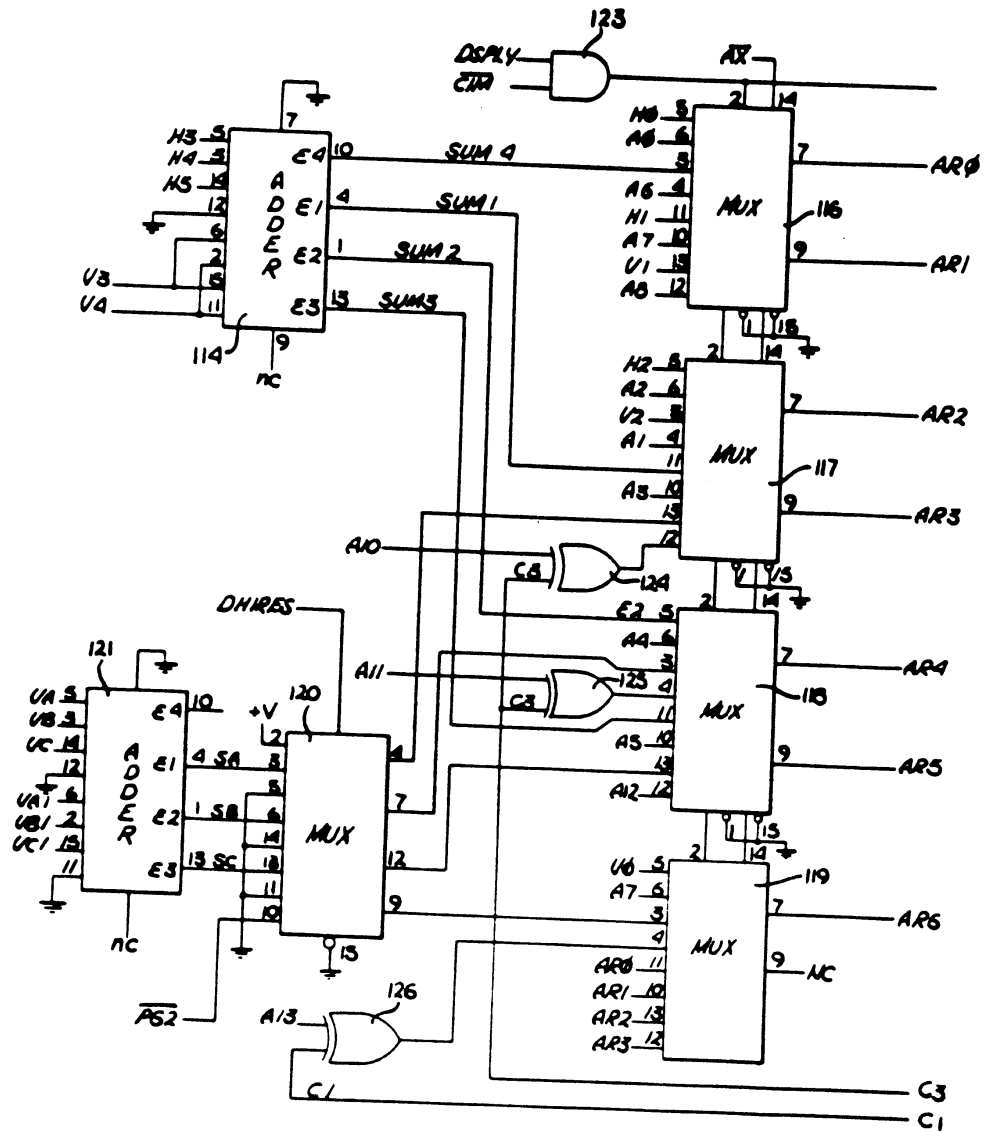
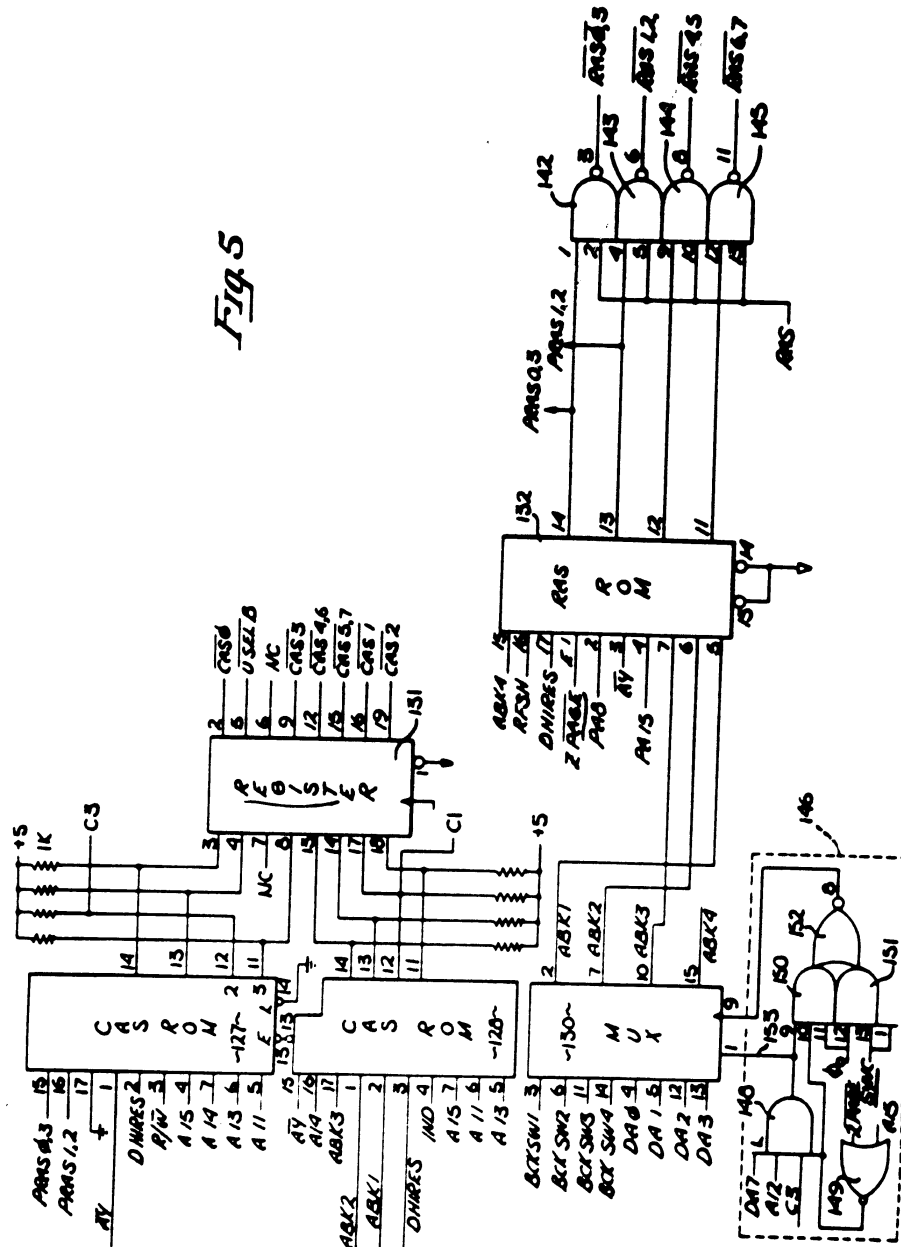


Fig. 4

Source /// PATENT - AUG 1985



Source III PATENT - AUG 1985

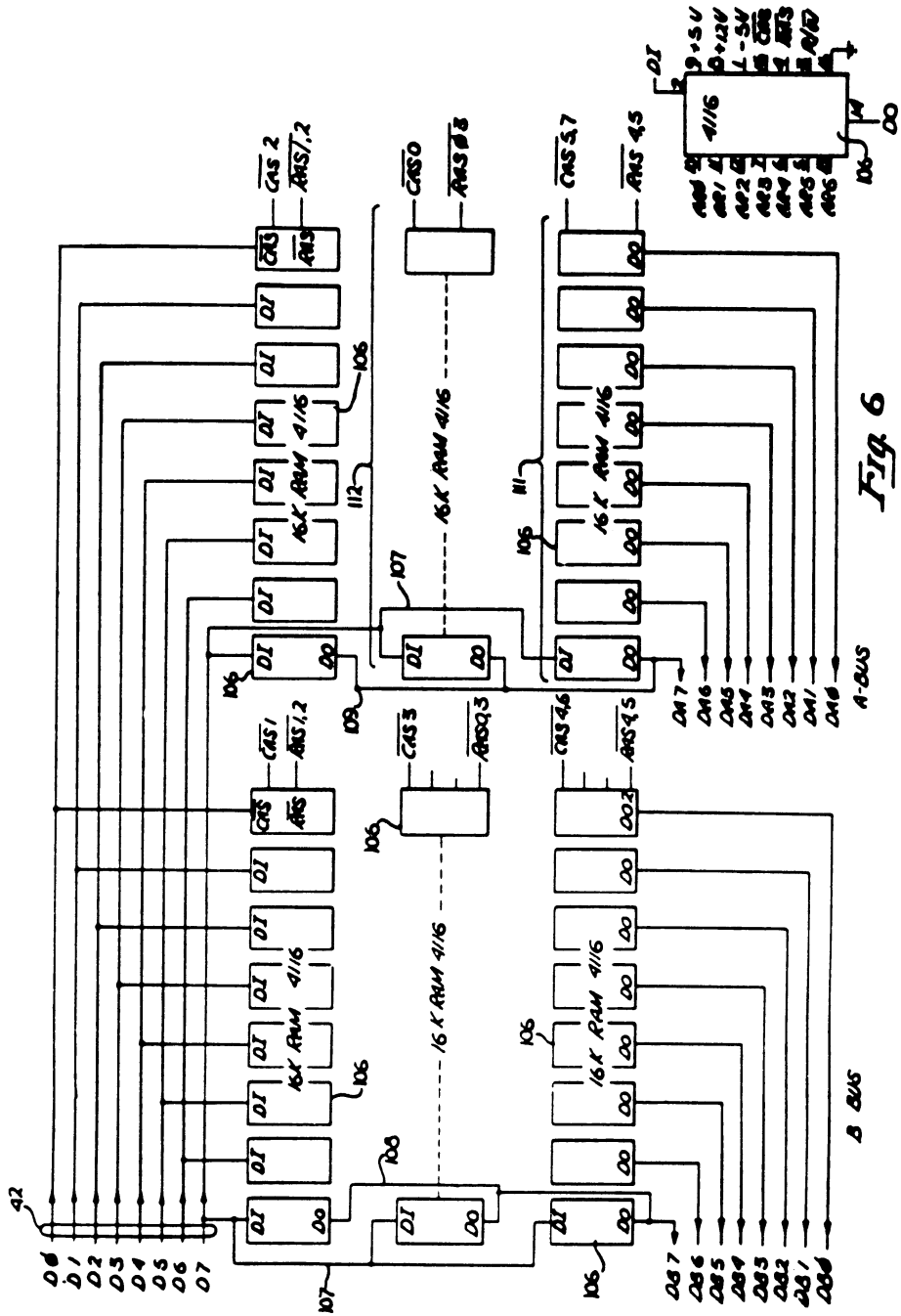


Fig 6

Source III PATENT — AUG 1985

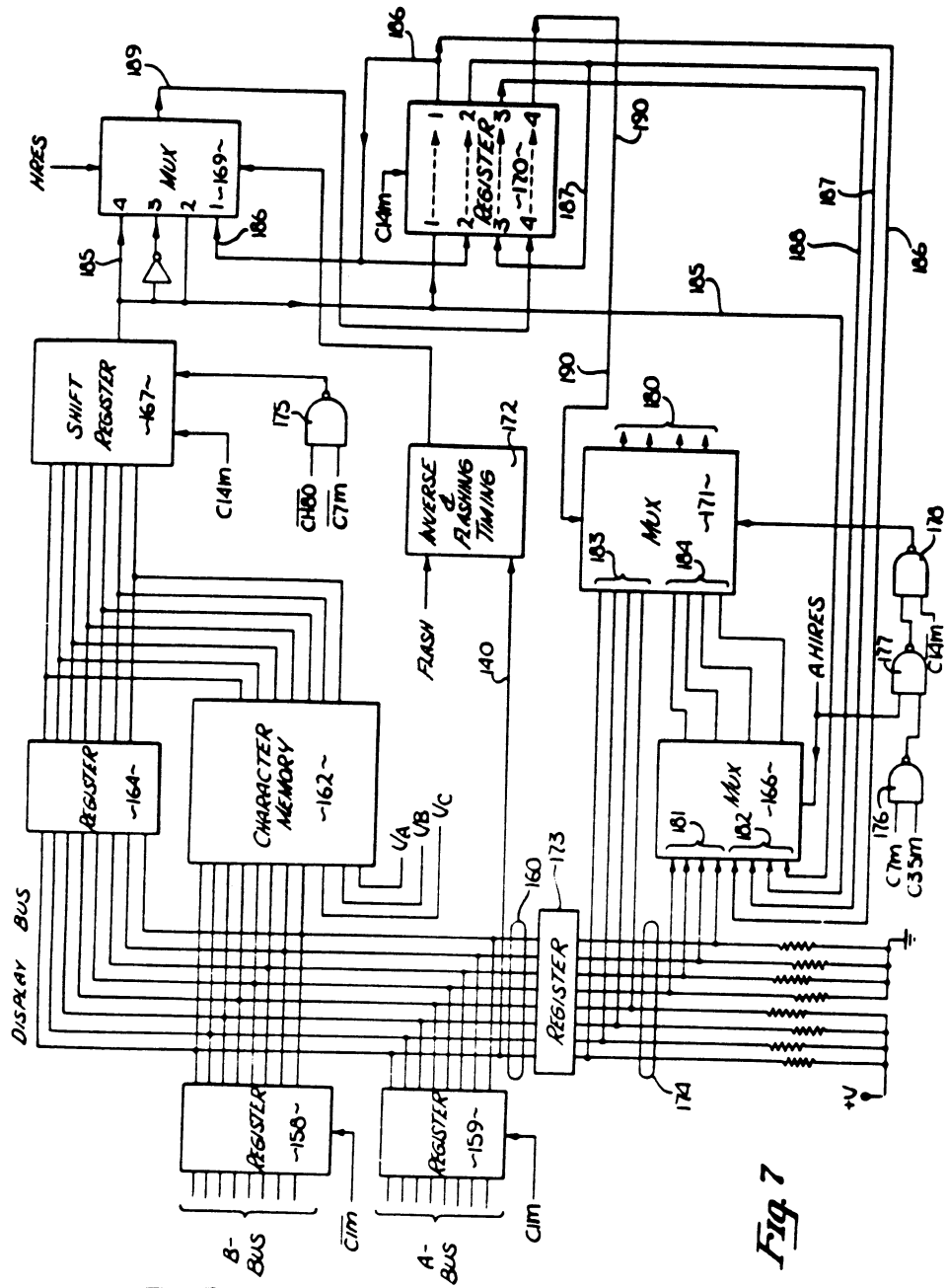


Fig. 7

Source /// PATENT - AUG 1985

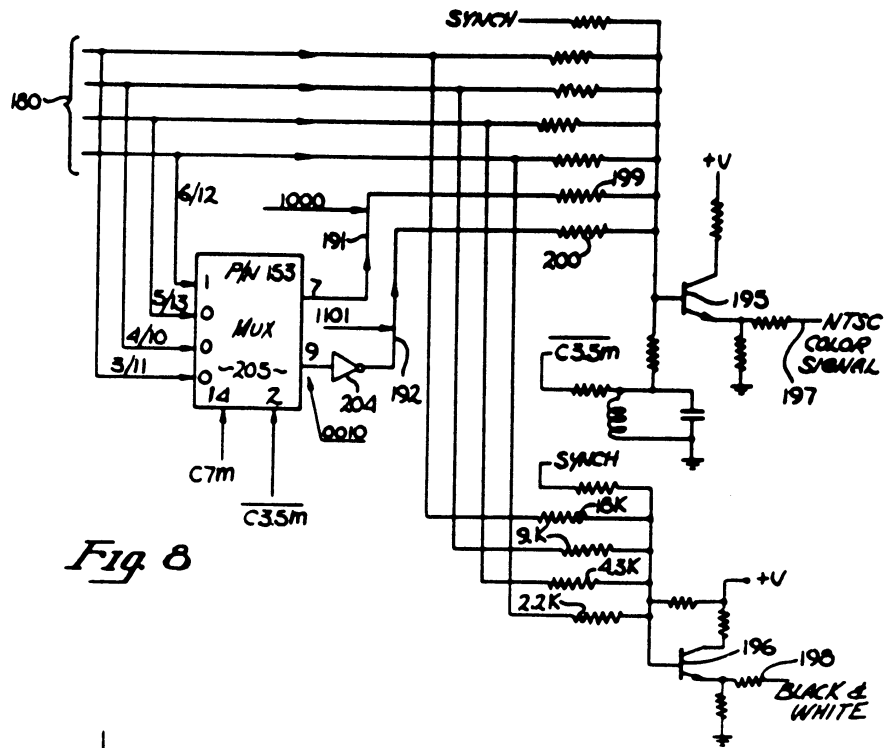


Fig 8

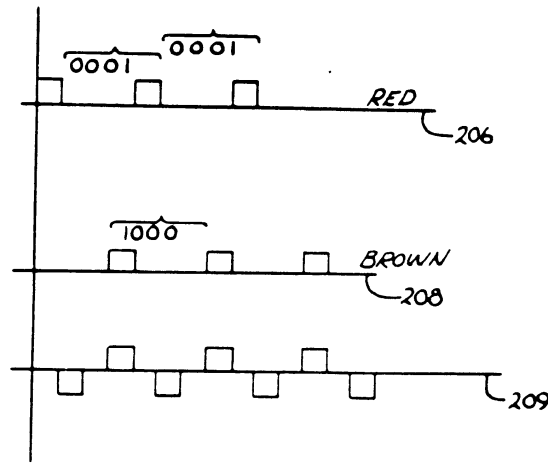
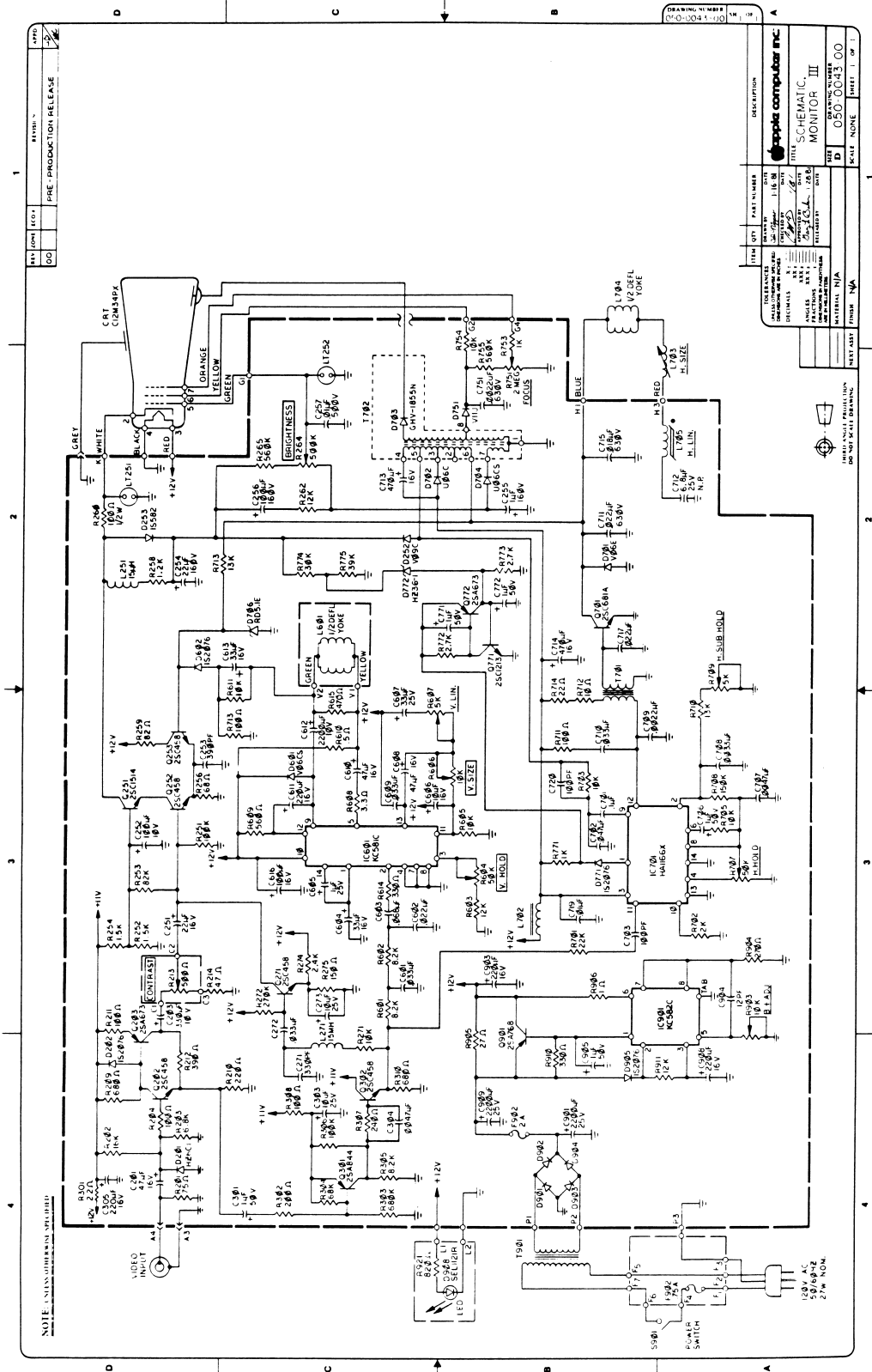


Fig 9

Source /// PATENT - AUG 1985



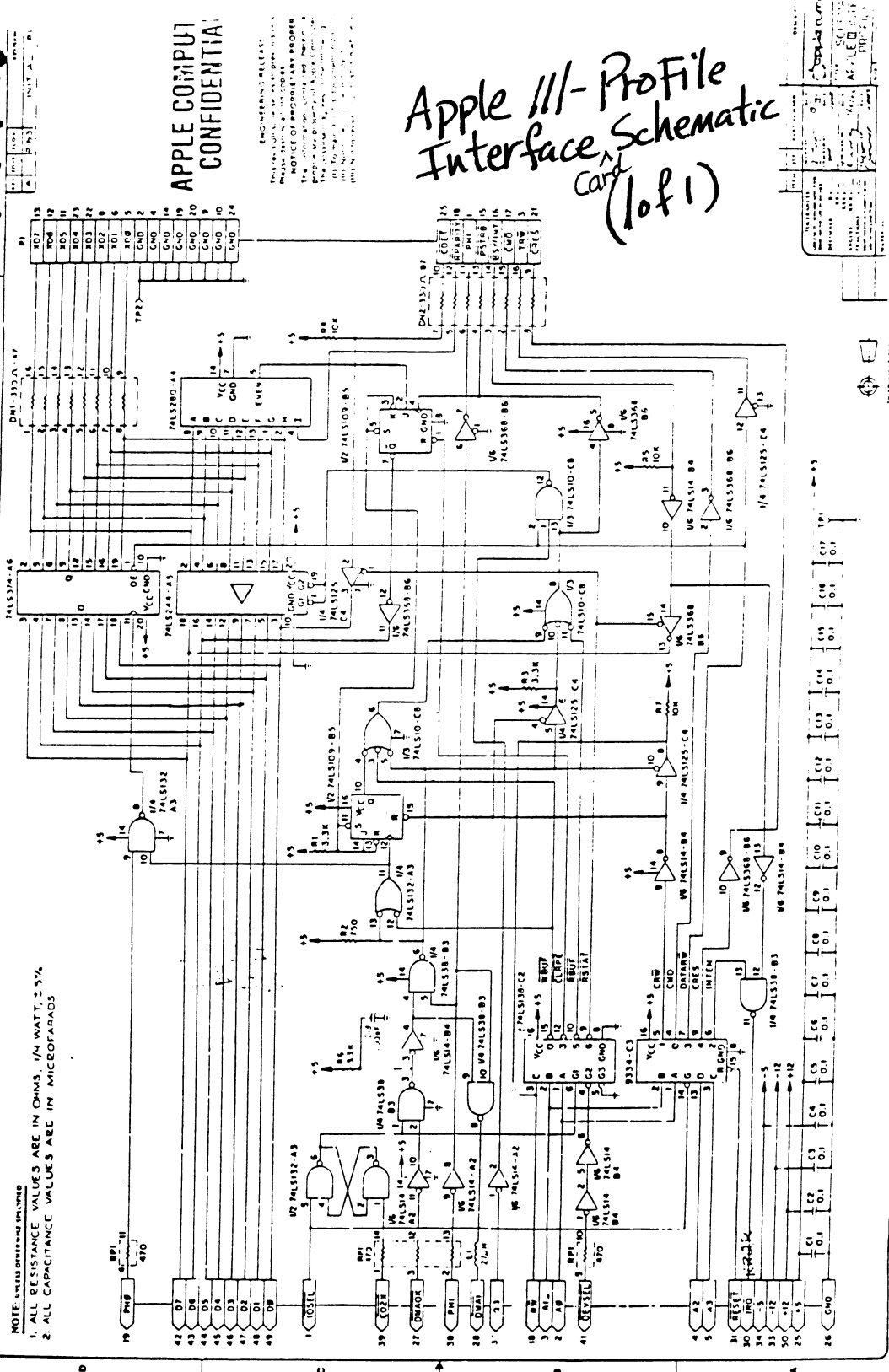
Source *MONITOR III USER'S MANUAL*

JEFF DENNIS

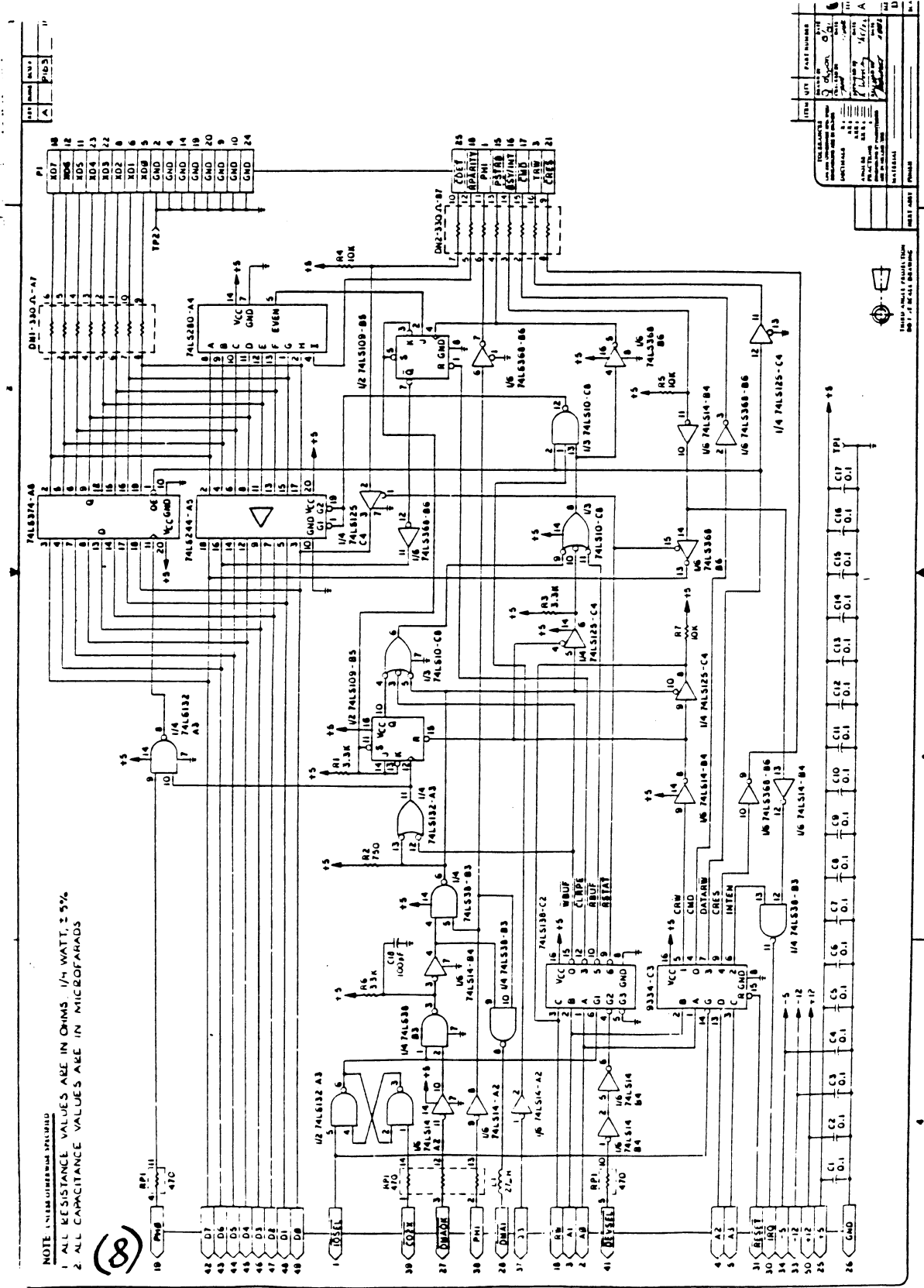
APPLE COMPUT
CONFIDENTIAL

ENGINEERING RELEASE
PROPERTY OF APPLE COMPUTER, INC.
NOTICE OF PROPRIETARY RIGHTS
The information contained herein is the property of Apple Computer, Inc. and is intended for use only by those individuals authorized by Apple Computer, Inc. to receive this information. It is not to be distributed, copied, or otherwise used in any manner without the express written permission of Apple Computer, Inc.

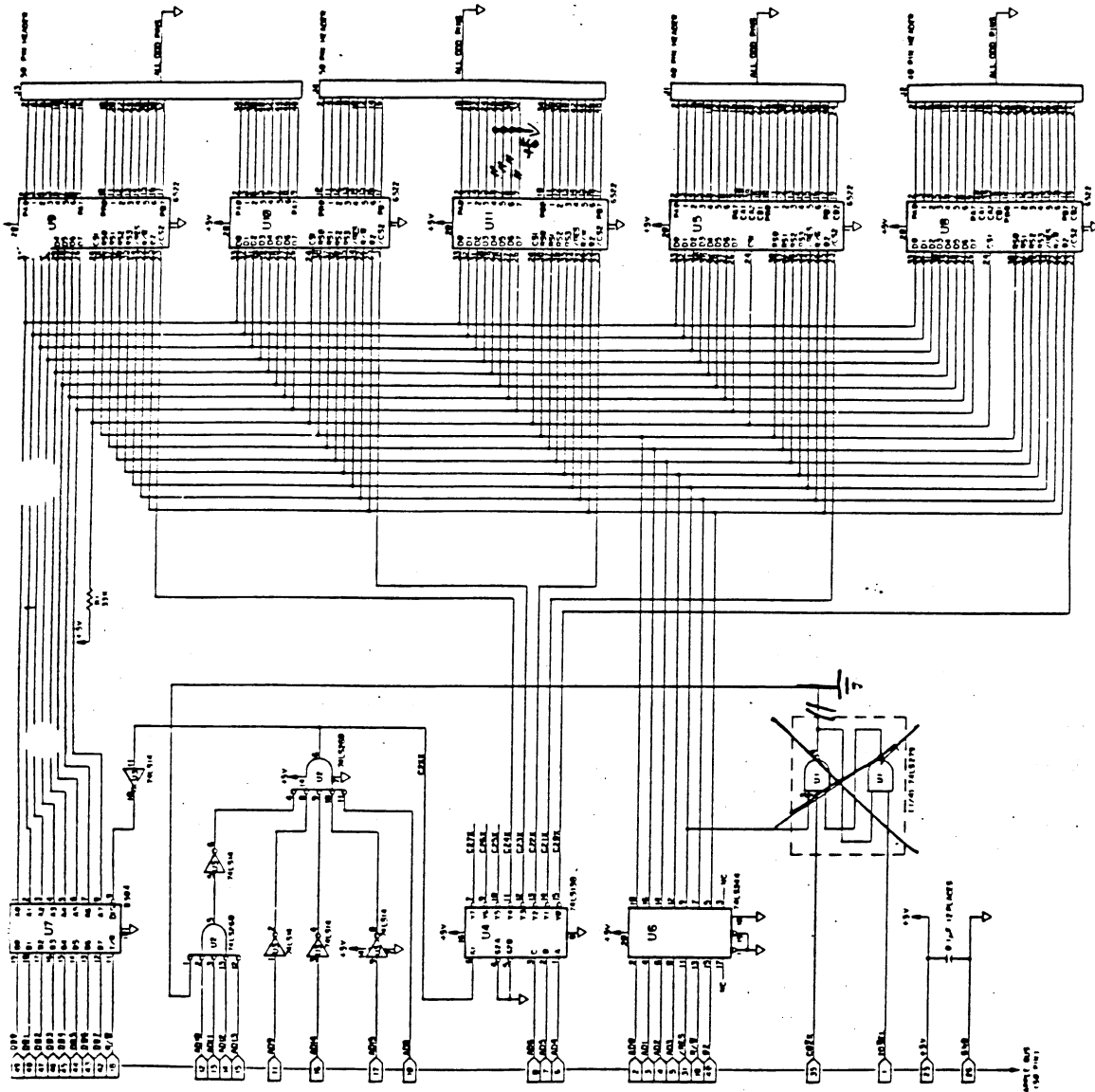
Apple ///-Profile
Interface Schematic
Card (of 1)



Source PROFILE SERVICE

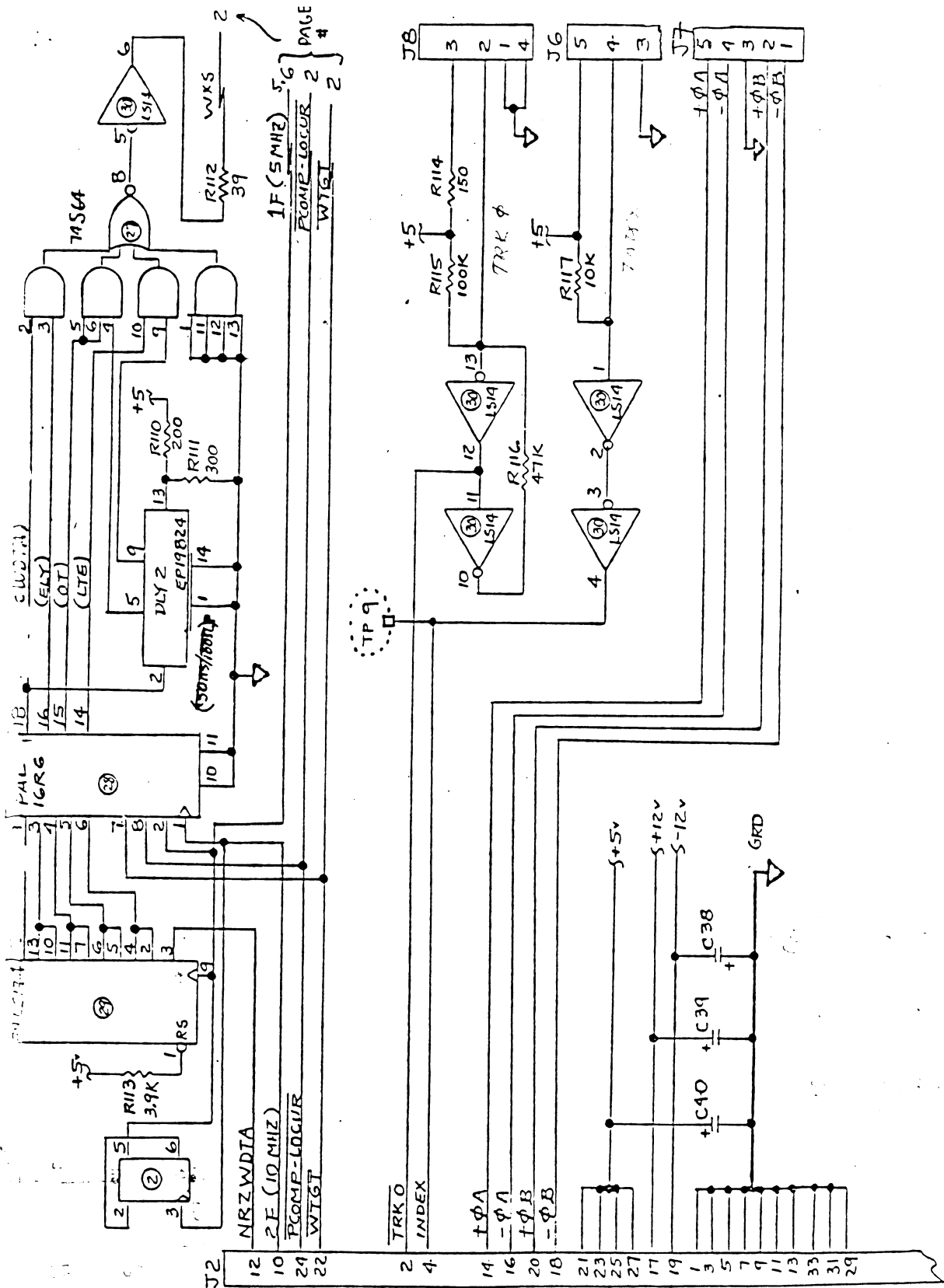


Source PROFILE SERVICE

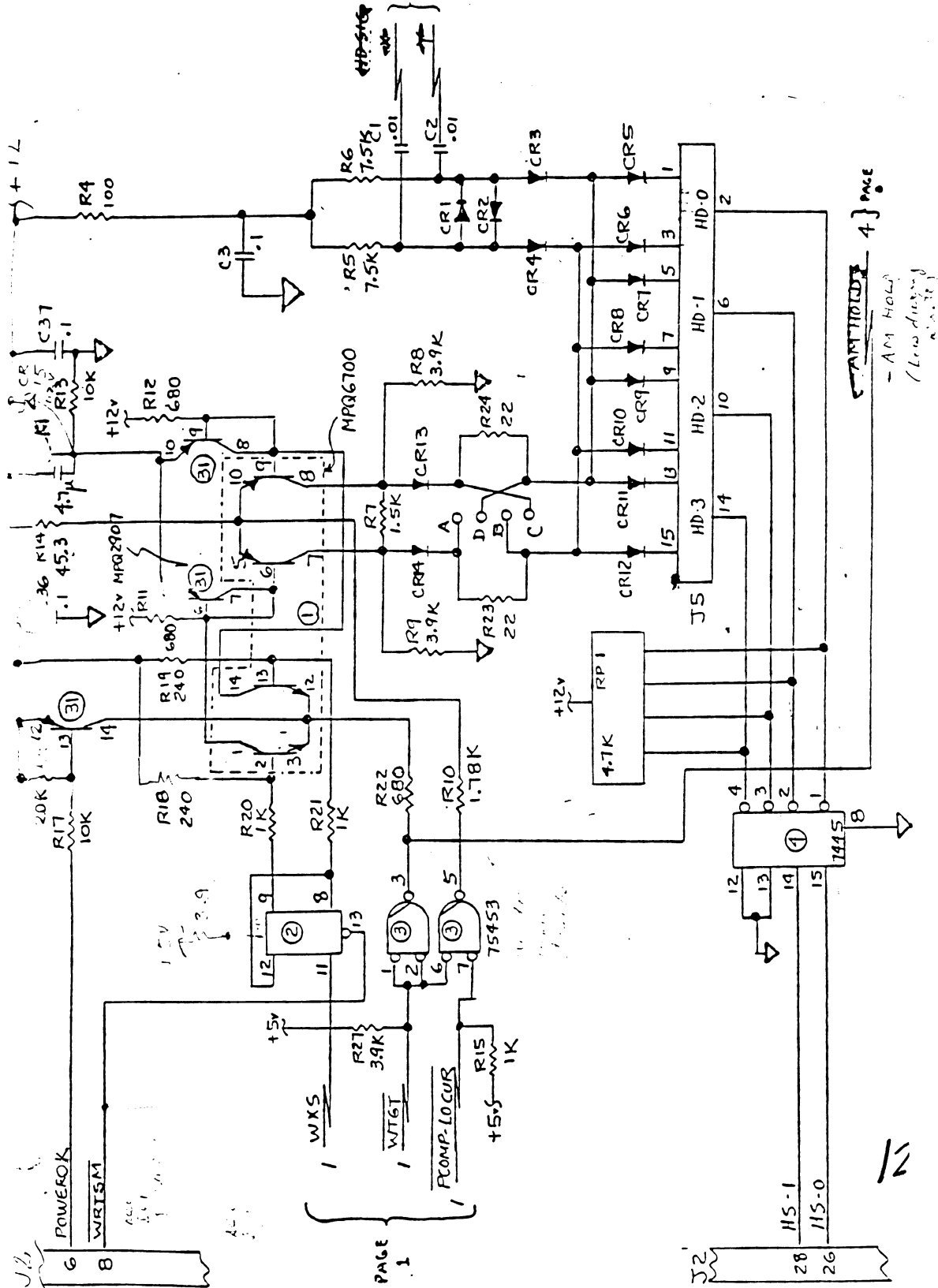


Apple Computer, Inc.
 100 Apple Way
 Cupertino, CA 95014
 (415) 947-8800
 FAX (415) 947-8801
 Telex 0841000 Apple
 Cable 0841000 Apple
 Apple Computer, Inc.
 100 Apple Way
 Cupertino, CA 95014
 (415) 947-8800
 FAX (415) 947-8801
 Telex 0841000 Apple
 Cable 0841000 Apple

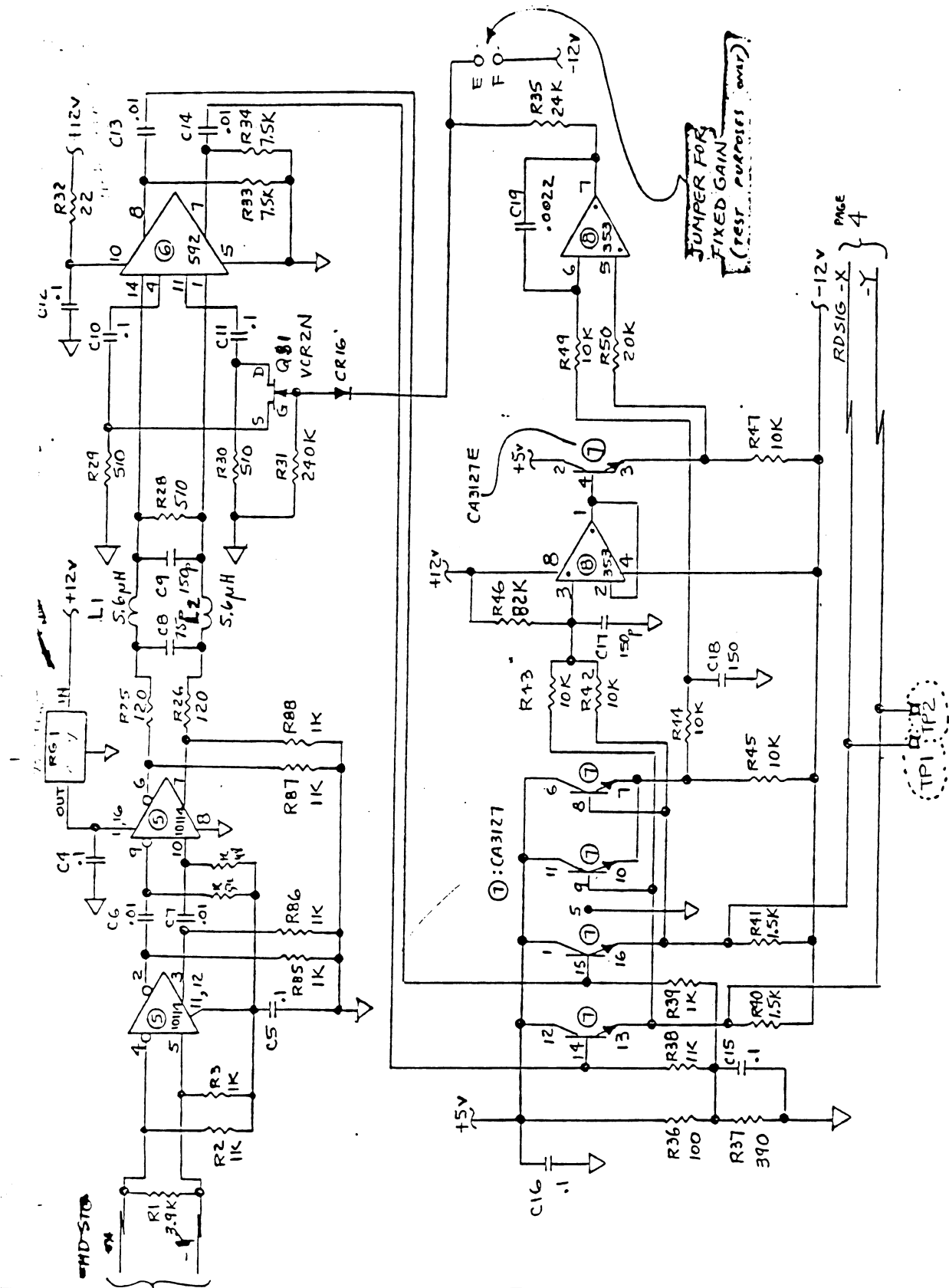
Source PROFILE SERVICE



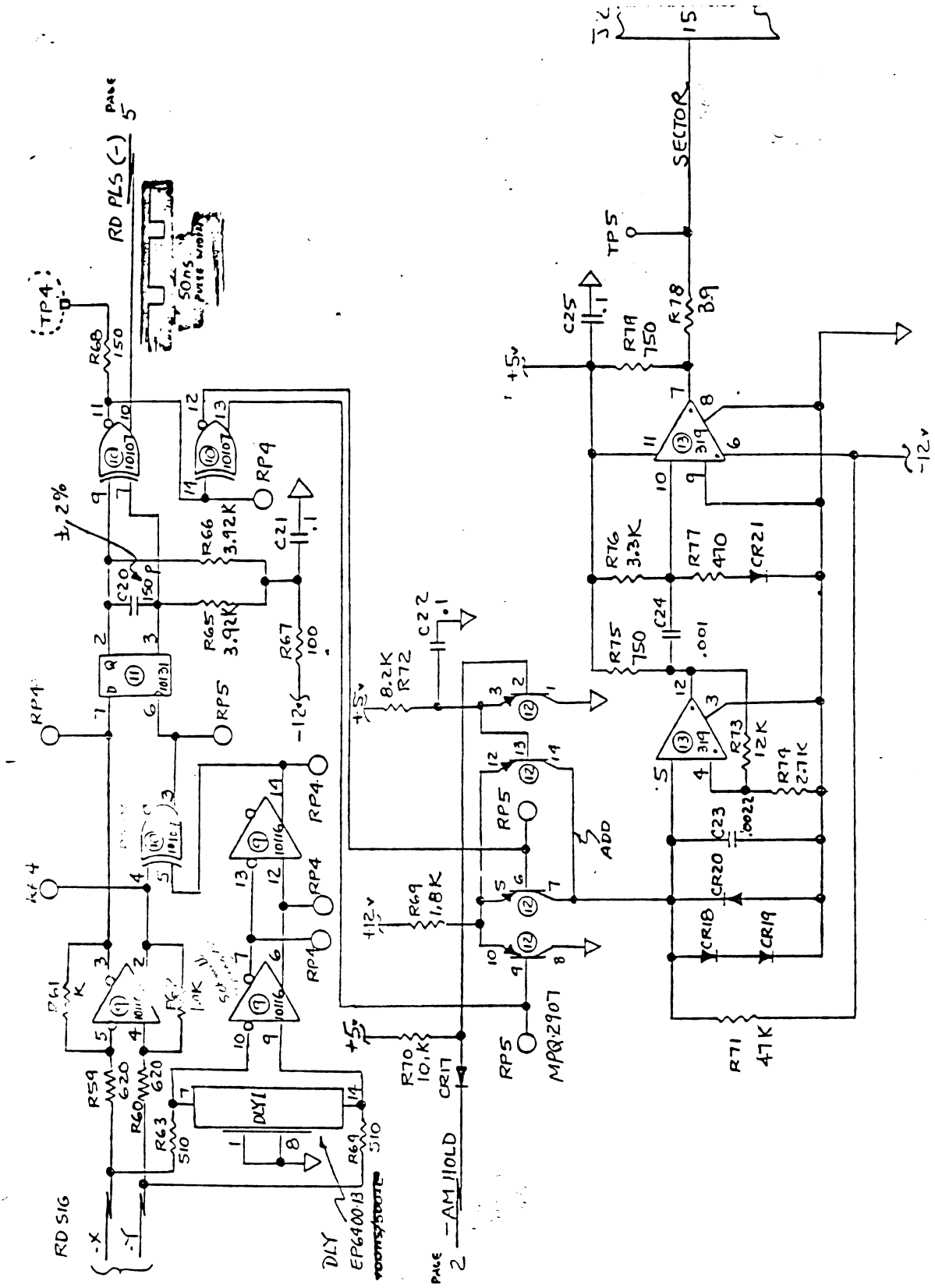
Source PROFILE SERVICE



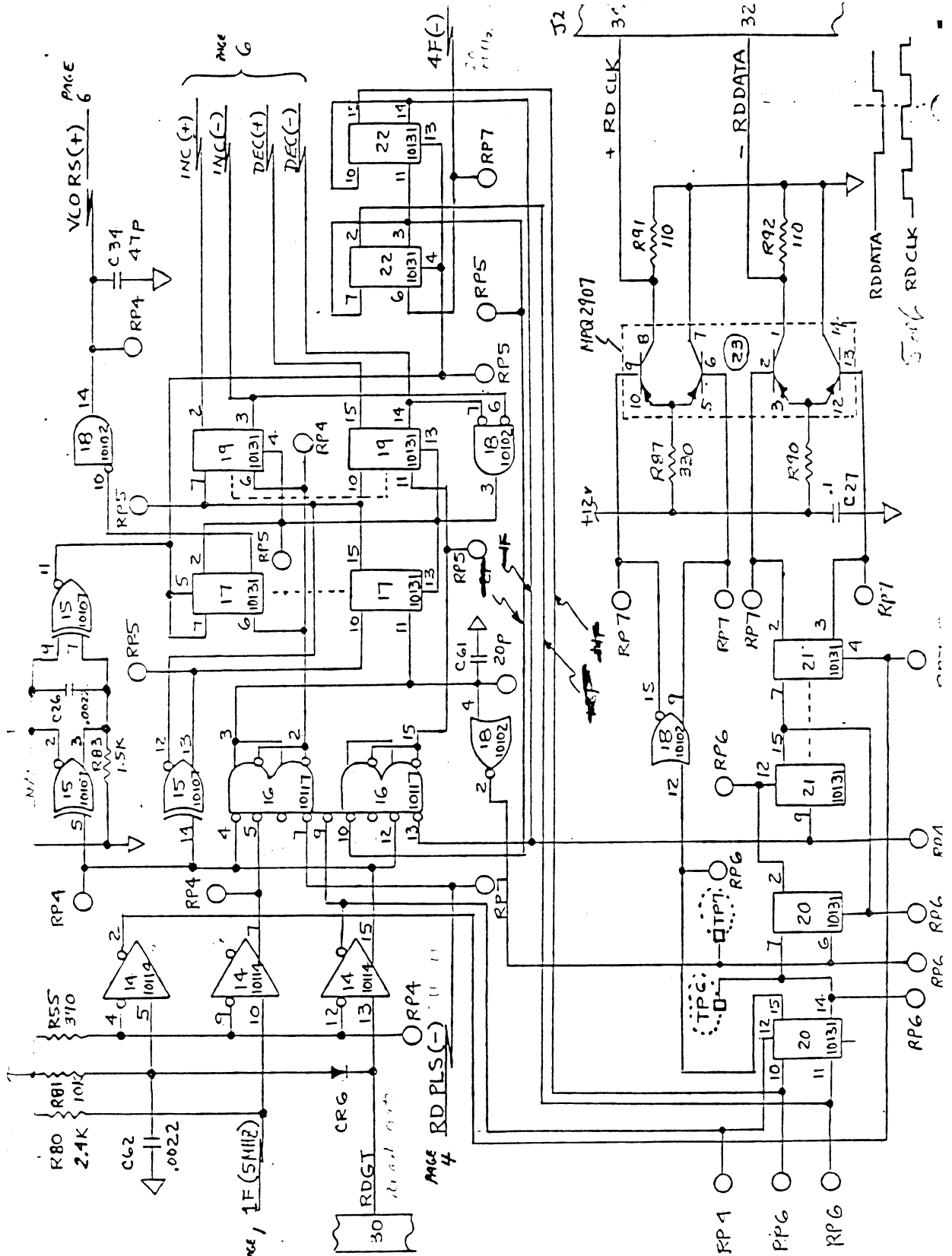
Source PROFILE SERVICE



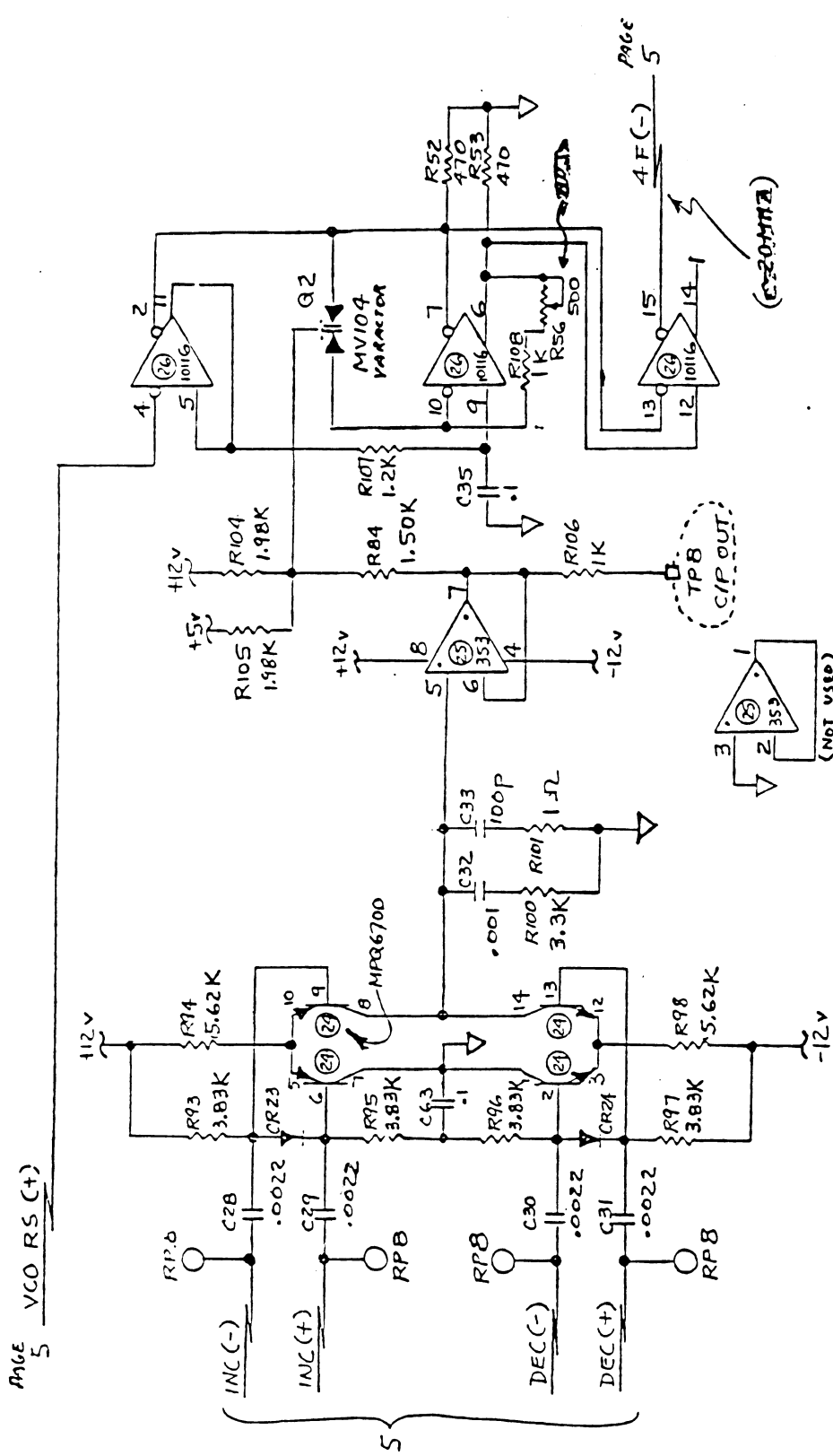
Source PROFILE SERVICE



Source PROFILE SERVICE



Source PROFILE SERVICE



REV DATE : 5/5/81
 0P7

Source PROFILE SERVICE



Apple /// Computer Schematics

Source
Apple Service
Level II Technical Reference
204-1022
Volume II

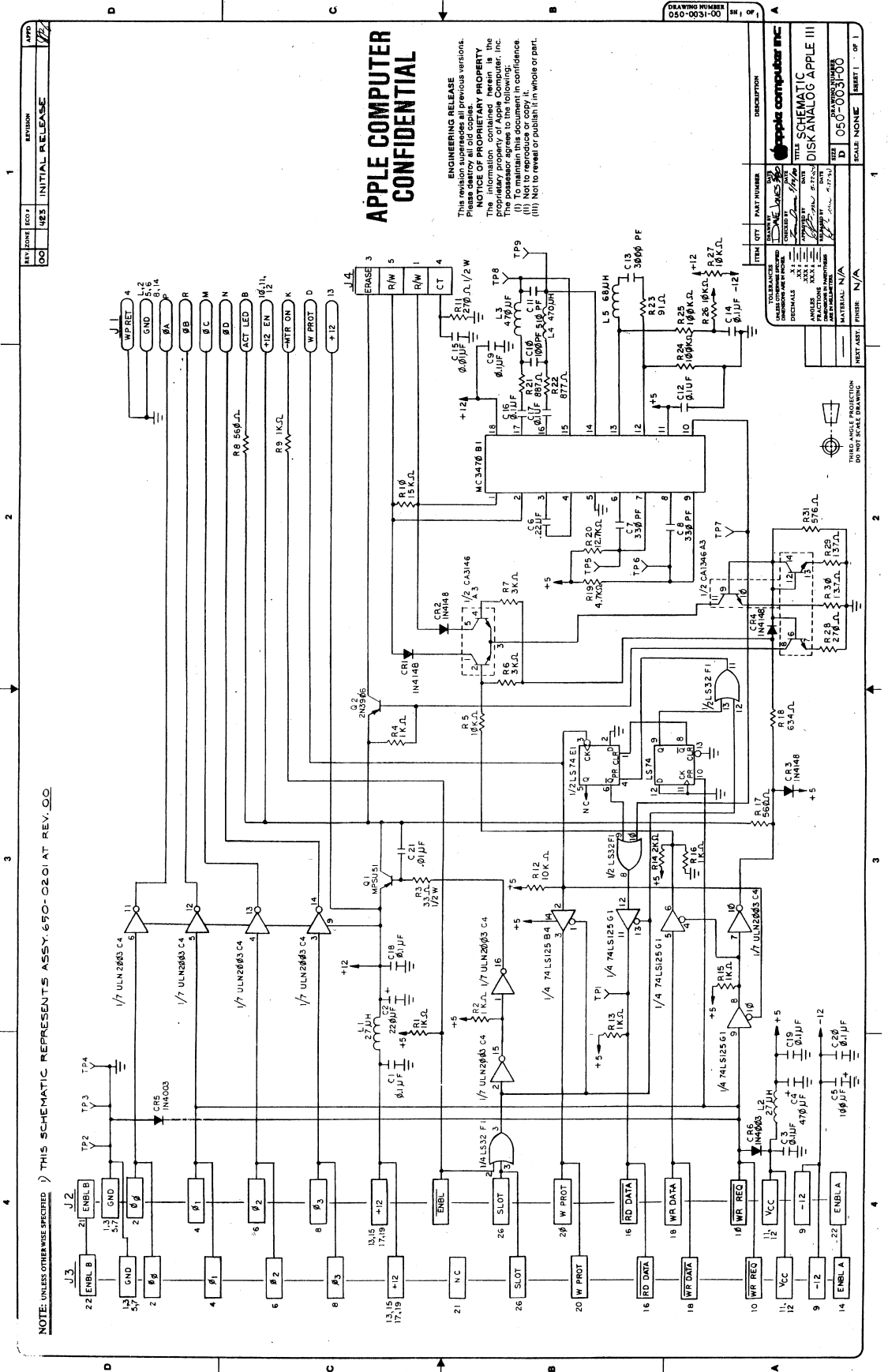
SCHEMATIC LIST

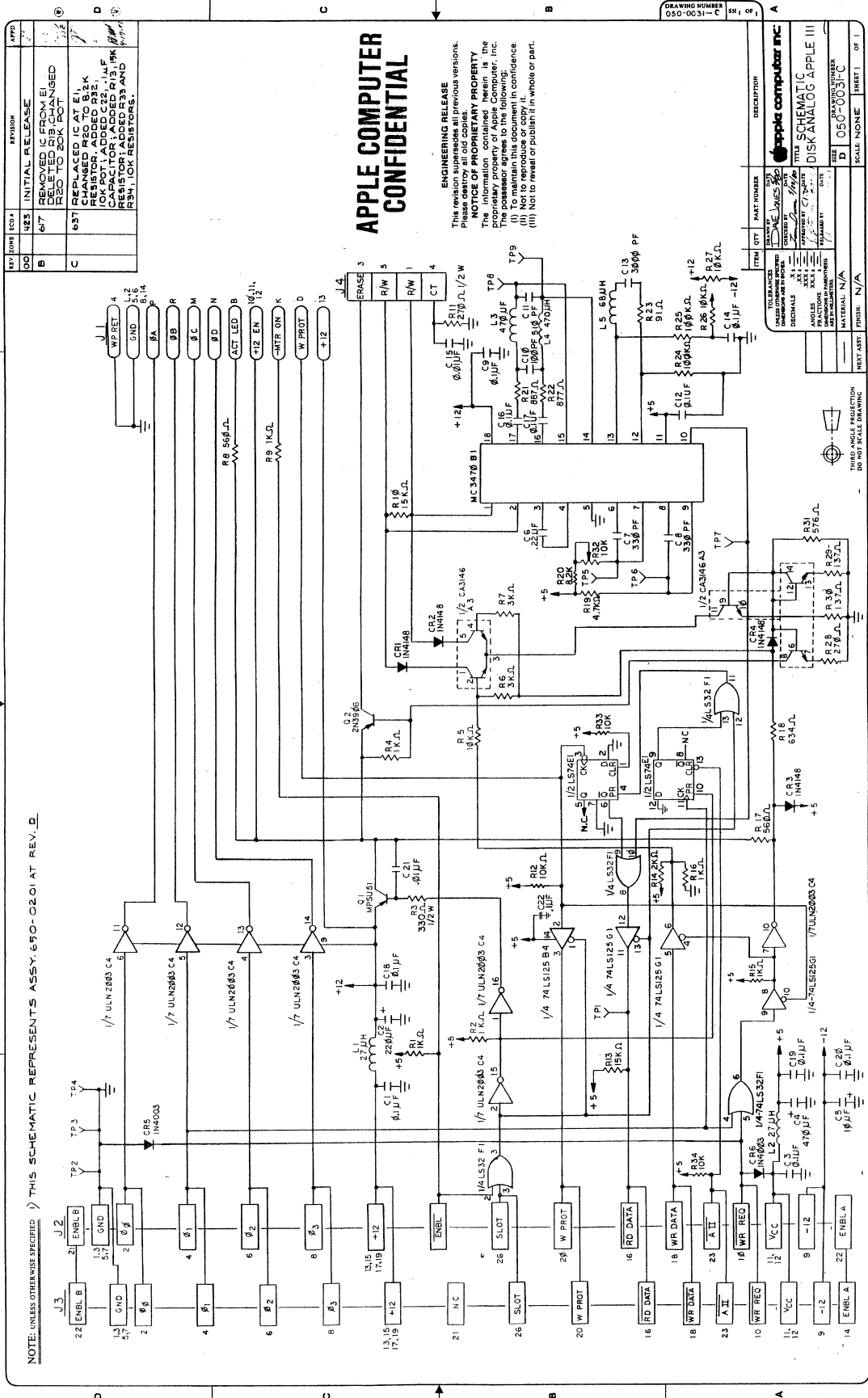
Disk Analog	050-0031-00	05/1980
Disk Analog	050-0031-C	04/1981
Main Memory Board	050-0032-E	05/1980
Main Logic # 2	050-0039-H	06/1982
Parallel Printer Card	050-0042-A	04/1982
5V Memory Board	050-0044-B	05/1981
ProFile Analog Board	050-5005-B	01/1982
ProFile Controller Board	050-5006-A	01/1982
ProFile Apple /// Interface Card	050-5007-A	01/1982

Compiled by

David T Craig

April 2003





REV	DATE	BY	DESCRIPTION
B	6/7		REMOVED IC FROM E1 DELETED R18 CHANGED R20 TO 20K POT
C	6/31		REPLACED IC AT E1, 10K POT, ADDED C22, 10K RESISTOR, ADDED R32, RESISTOR FOR ADDED R15, 10K RESISTOR FOR ADDED R13, 10K RESISTORS

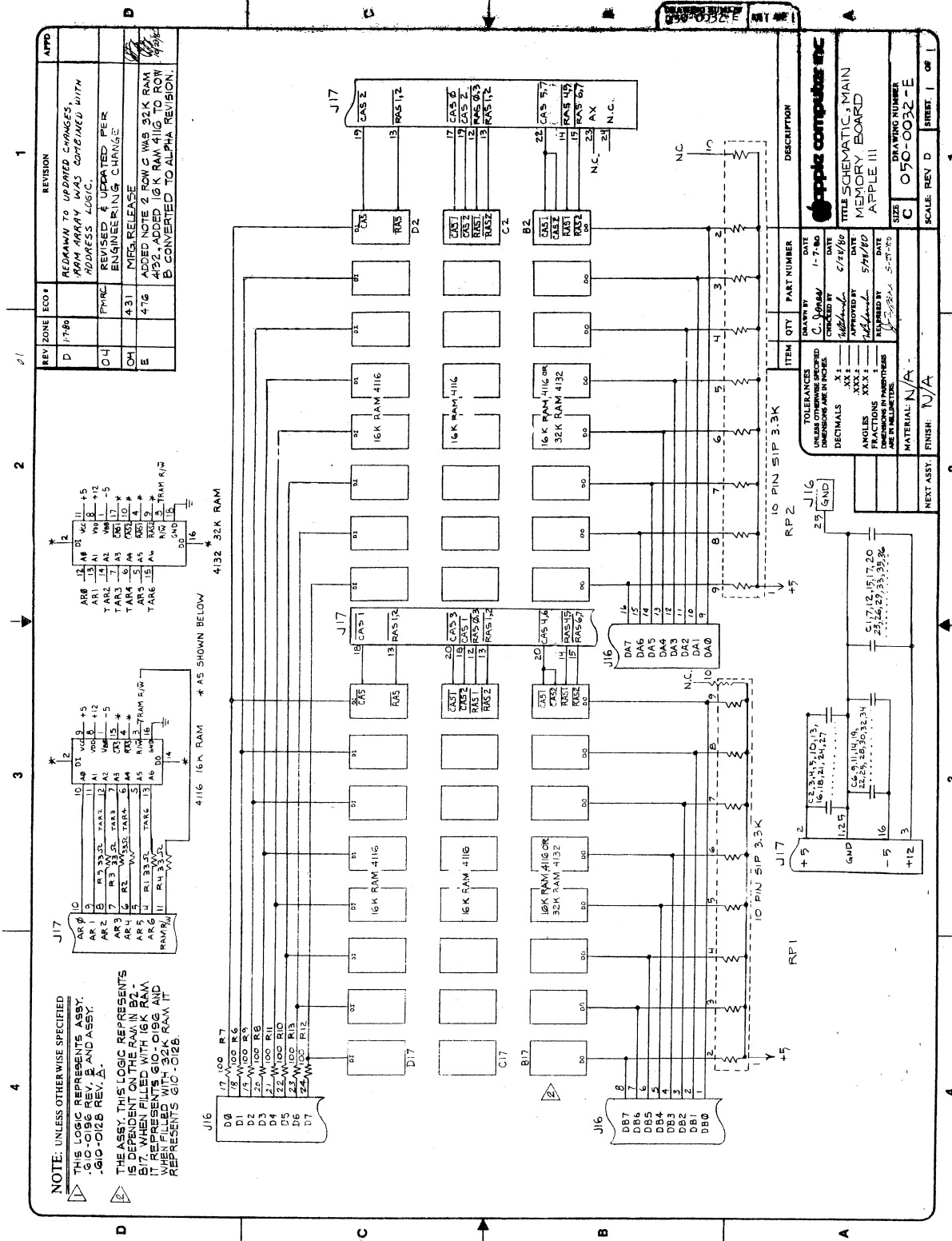
APPLE CONFIDENTIAL

ENGINEERING RELEASE
 This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY:
 This document contains proprietary information of Apple Computer, Inc. The possessor agrees to the following:
 (i) To maintain this document in confidence
 (ii) Not to reveal or publish it in whole or part.

DRAWING NUMBER 050-0031-C SH 1 OF 1

ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	NC3476 B1	DISK ANALOG APPLE III
2	1	74LS2683 C4	DISK ANALOG APPLE III
3	1	74LS125 G1	DISK ANALOG APPLE III
4	1	74LS125 G2	DISK ANALOG APPLE III
5	1	74LS125 G1	DISK ANALOG APPLE III
6	1	74LS125 G1	DISK ANALOG APPLE III
7	1	74LS125 G1	DISK ANALOG APPLE III
8	1	74LS125 G1	DISK ANALOG APPLE III
9	1	74LS125 G1	DISK ANALOG APPLE III
10	1	74LS125 G1	DISK ANALOG APPLE III
11	1	74LS125 G1	DISK ANALOG APPLE III
12	1	74LS125 G1	DISK ANALOG APPLE III
13	1	74LS125 G1	DISK ANALOG APPLE III
14	1	74LS125 G1	DISK ANALOG APPLE III
15	1	74LS125 G1	DISK ANALOG APPLE III
16	1	74LS125 G1	DISK ANALOG APPLE III
17	1	74LS125 G1	DISK ANALOG APPLE III
18	1	74LS125 G1	DISK ANALOG APPLE III
19	1	74LS125 G1	DISK ANALOG APPLE III
20	1	74LS125 G1	DISK ANALOG APPLE III
21	1	74LS125 G1	DISK ANALOG APPLE III
22	1	74LS125 G1	DISK ANALOG APPLE III
23	1	74LS125 G1	DISK ANALOG APPLE III
24	1	74LS125 G1	DISK ANALOG APPLE III
25	1	74LS125 G1	DISK ANALOG APPLE III
26	1	74LS125 G1	DISK ANALOG APPLE III
27	1	74LS125 G1	DISK ANALOG APPLE III
28	1	74LS125 G1	DISK ANALOG APPLE III
29	1	74LS125 G1	DISK ANALOG APPLE III
30	1	74LS125 G1	DISK ANALOG APPLE III
31	1	74LS125 G1	DISK ANALOG APPLE III
32	1	74LS125 G1	DISK ANALOG APPLE III
33	1	74LS125 G1	DISK ANALOG APPLE III

NOTE: UNLESS OTHERWISE SPECIFIED, THIS SCHEMATIC REPRESENTS ASSY. 690-02.01 AT REV. D.



NOTE: UNLESS OTHERWISE SPECIFIED THIS LOGIC REPRESENTS ASSY. .G10-0128 REV. B AND ASSY. .G10-0128 REV. A.

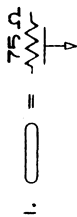
THE ASSY. THIS LOGIC REPRESENTS IS MOUNTED ON THE MAIN B2 - B7 WHICH FILLS IN B2 - B7. IT REPRESENTS G10-0128 AND WHEN FILLED WITH 32K RAM IT REPRESENTS G10-0128.

ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.

NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(I) To maintain this document in confidence.
(II) Not to reproduce or copy it.
(III) Not to reveal or publish it in whole or part.

**APPLE COMPUTER
CONFIDENTIAL**

NOTE: UNLESS OTHERWISE SPECIFIED



2. $\text{C} \text{---} \text{C}$ = ALL UNITS UNDER TEST MOLEX PINS TO BE LOCATED TOGETHER.

3. $\text{C} \text{---} \text{C}$ = ALL TIMING MOLEX PINS TO BE LOCATED TOGETHER.

4. THIS SCHEMATIC REPRESENTS ASSY 610-0105 AT REV LEVEL: F.

FOR STANDARD SYSTEMS G9 IS 341-0030, FOR EURO SYSTEMS G9 IS 341-0060.

FOR STANDARD SYSTEMS Y1 IS 14.318630 MHZ, FOR EURO SYSTEMS Y1 IS 14.250450 MHZ.

REV	ZONE	ECO #	REVISION	APPD
O2			ENG. RELEASE	
O3			PRE-PRODUCTION RELEASE	
E			PRE-PRODUCTION CHANGES	
E		504	ECO RELEASE	
E		544	DELETED CAP ARRAY (CBI), AND N15, N13, K13, N3, N10 WAS PC NETWORK, C5 WAS NOW IOSTOPD, J20, 7 WAS COLORKILL, J20, 9 WAS FORPAGE, DELETED MICROPROCESSOR (B3)	
G		778	H8 WAS L504; R15 WAS L7 OHM; R16 WAS 75 OHM. ADDED 1K OHM RESISTOR AT K10.	
H		923	CHANGE CAP C22 FROM 1 uF TO .022 uF, CHANGE RP-4.	

APPLE COMPUTER CONFIDENTIAL

ENGINEERING RELEASE
 This revision supersedes all previous versions.
 Please refer to the previous versions for details.
NOTICE OF PROPRIETARY PROPERTY
 The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
 (i) To maintain this document in confidence.
 (ii) Not to reproduce or copy it.
 (iii) Not to reveal or publish it in whole or part.

DRAWING NUMBER
050-0039-H SH1 OF 10

ITEM	QTY	PART NUMBER	DESCRIPTION																																
<table border="0"> <tr> <td>TOLERANCES UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES</td> <td>DRAWN BY</td> <td>DATE</td> <td></td> </tr> <tr> <td>DECIMALS .X ±</td> <td>WFB</td> <td>9-4-79</td> <td></td> </tr> <tr> <td>ANGLES .XX ±</td> <td>CHECKED BY</td> <td>DATE</td> <td></td> </tr> <tr> <td>FRACTIONS XXX ±</td> <td>JMcDonald</td> <td>8-19-80</td> <td></td> </tr> <tr> <td>DIMENSIONS IN PARENTHESES ARE IN MILLIMETERS</td> <td>APPROVED BY</td> <td>DATE</td> <td></td> </tr> <tr> <td></td> <td>JMcDonald</td> <td>8-19-80</td> <td></td> </tr> <tr> <td>MATERIAL: N/A</td> <td>RELEASED BY</td> <td>DATE</td> <td></td> </tr> <tr> <td></td> <td>JMcDonald</td> <td>8-28-80</td> <td></td> </tr> </table>				TOLERANCES UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES	DRAWN BY	DATE		DECIMALS .X ±	WFB	9-4-79		ANGLES .XX ±	CHECKED BY	DATE		FRACTIONS XXX ±	JMcDonald	8-19-80		DIMENSIONS IN PARENTHESES ARE IN MILLIMETERS	APPROVED BY	DATE			JMcDonald	8-19-80		MATERIAL: N/A	RELEASED BY	DATE			JMcDonald	8-28-80	
TOLERANCES UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES	DRAWN BY	DATE																																	
DECIMALS .X ±	WFB	9-4-79																																	
ANGLES .XX ±	CHECKED BY	DATE																																	
FRACTIONS XXX ±	JMcDonald	8-19-80																																	
DIMENSIONS IN PARENTHESES ARE IN MILLIMETERS	APPROVED BY	DATE																																	
	JMcDonald	8-19-80																																	
MATERIAL: N/A	RELEASED BY	DATE																																	
	JMcDonald	8-28-80																																	
TITLE SCHEMATIC APPLE /// MAIN LOGIC #2			DRAWING NUMBER 050-0039-H																																
NEXT ASSY: FINISH: N/A			SCALE: NONE. SHEET 1 OF 10																																



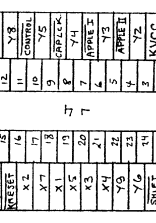
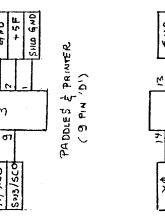
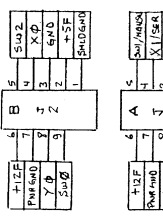
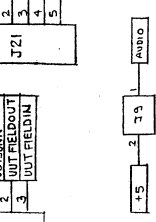
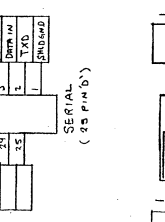
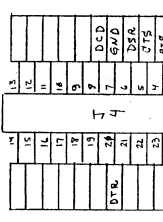
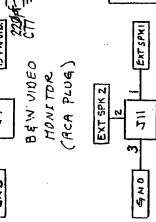
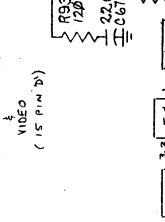
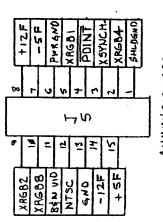
REV: ZONE ECO # 1 SHEET REVISION APPD
DRAWING NUMBER 050-0039-H SH 2 OF 10

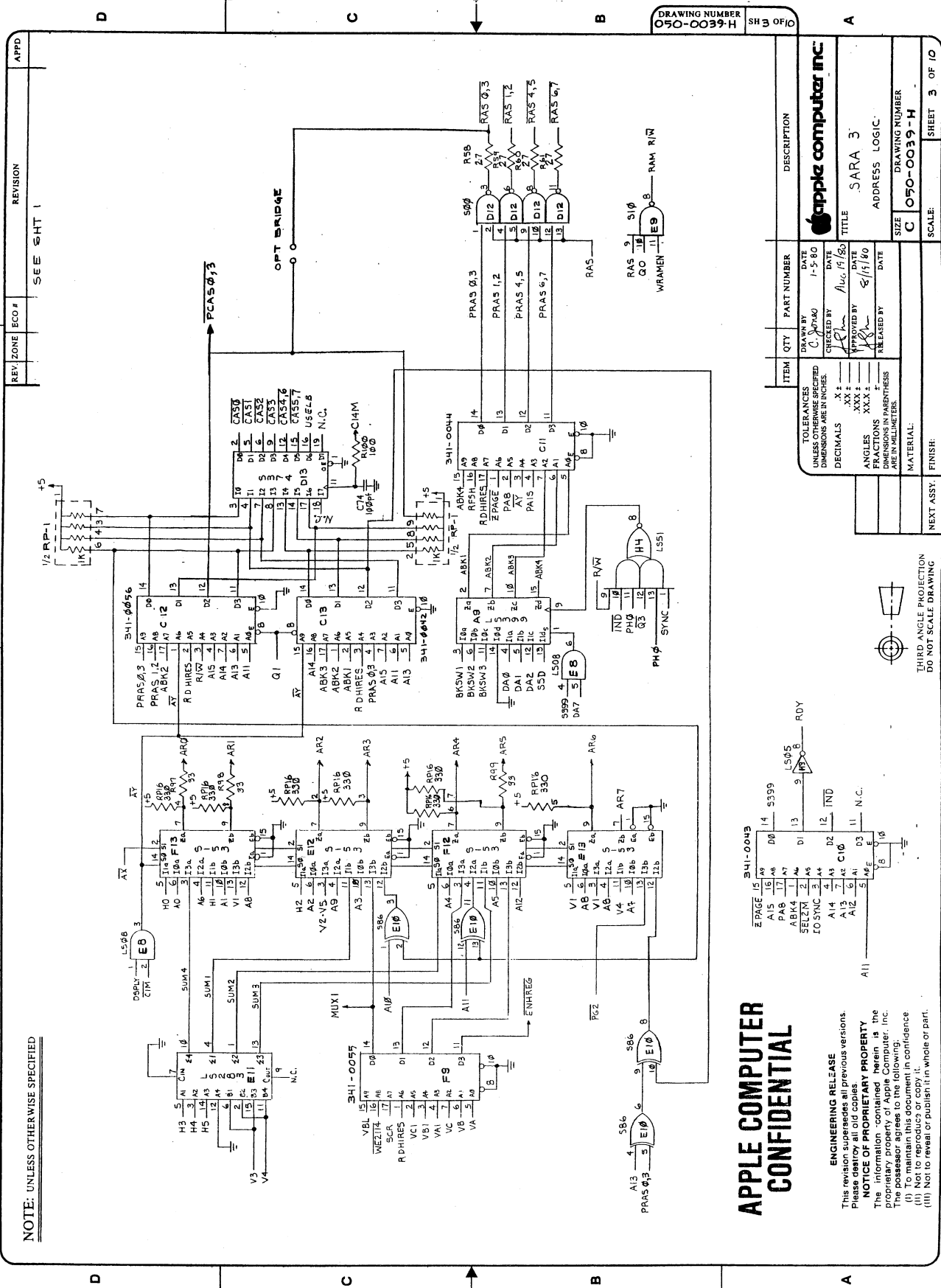
SEE SHEET
APPLE CONFIDENTIAL

ENGINEERING RELEASE
This revision supersedes all previous versions.
Please refer to all previous versions.
No part of this publication may be reproduced or transmitted in any form or by any means electronic, mechanical, photocopying, recording, or by any information storage or retrieval system, without the prior written permission of Apple Computer, Inc.

Peripheral Connectors

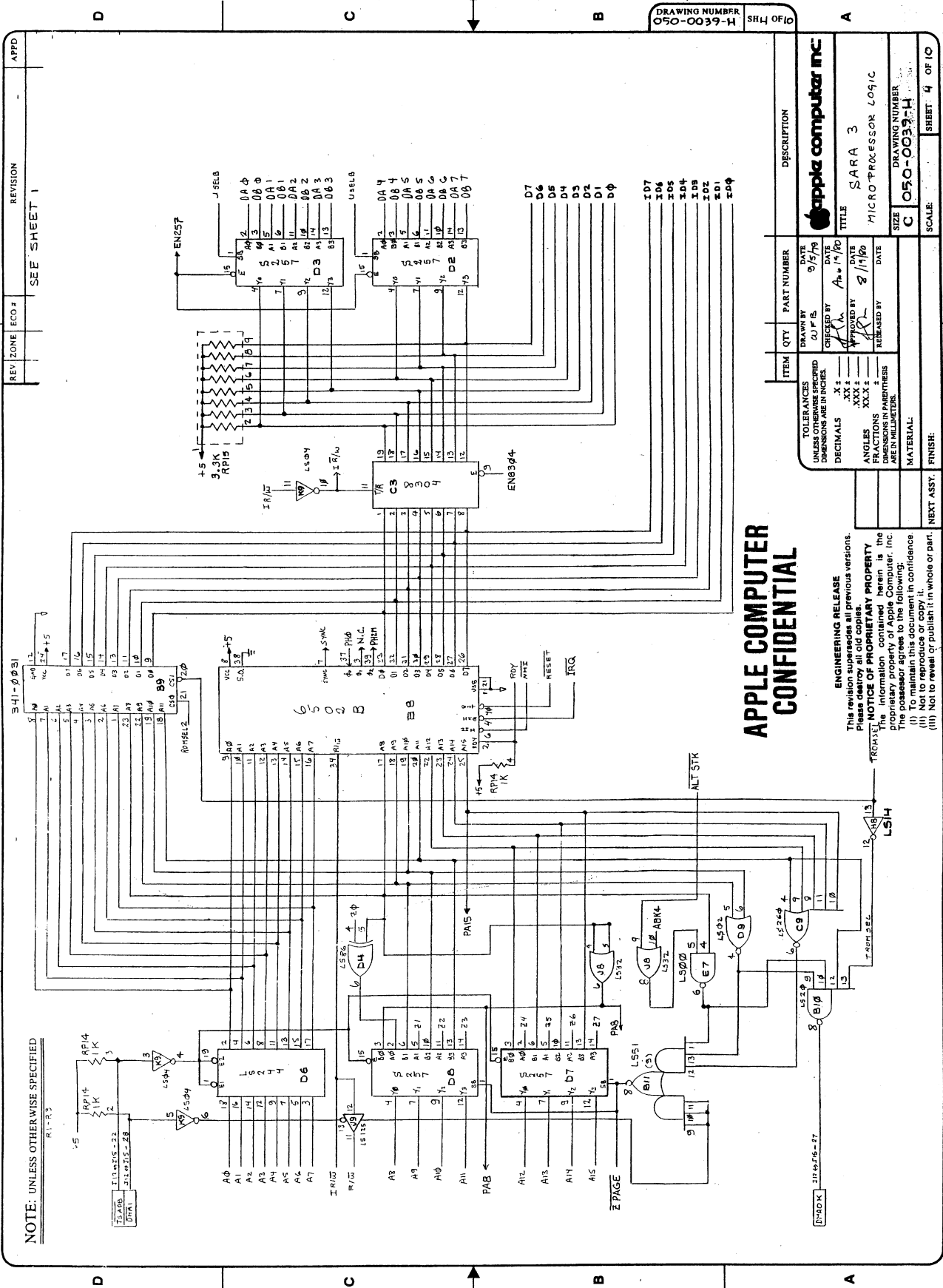
CON	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
POWER	DATA 1	DATA 2	DATA 3	DATA 4	DATA 5	DATA 6	DATA 7	DATA 8	DATA 9	DATA 10	DATA 11	DATA 12	DATA 13	DATA 14	DATA 15	DATA 16	DATA 17	DATA 18	DATA 19	DATA 20	DATA 21	DATA 22	DATA 23	DATA 24	DATA 25	DATA 26	DATA 27	DATA 28	DATA 29	DATA 30	DATA 31	DATA 32	DATA 33	DATA 34	DATA 35	DATA 36	DATA 37	DATA 38	DATA 39	DATA 40	DATA 41	DATA 42	DATA 43	DATA 44	DATA 45	DATA 46	DATA 47	DATA 48	DATA 49	DATA 50





**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions.
PLEASE DO NOT REPRODUCE OR DISSEMINATE THIS INFORMATION WITHOUT THE WRITTEN PERMISSION OF APPLE COMPUTER, INC.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(i) To maintain this document in confidence
(ii) Not to reproduce or copy it.
(iii) Not to reveal or publish in whole or in part.



NOTE: UNLESS OTHERWISE SPECIFIED
R1-R3
R4-R10
R11-R15
R16-R20
R21-R25
R26-R30
R31-R35
R36-R40
R41-R45
R46-R50
R51-R55
R56-R60
R61-R65
R66-R70
R71-R75
R76-R80
R81-R85
R86-R90
R91-R95
R96-R100

REV ZONE ECO #
SEE SHEET 1
REVISION
APPD

ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	EN8394	MICRO-PROCESSOR LOGIC
2	1	LS51	AND GATE
3	1	LS52	AND GATE
4	1	LS53	AND GATE
5	1	LS54	AND GATE
6	1	LS55	AND GATE
7	1	LS56	AND GATE
8	1	LS57	AND GATE
9	1	LS58	AND GATE
10	1	LS59	AND GATE
11	1	LS60	AND GATE
12	1	LS61	AND GATE
13	1	LS62	AND GATE
14	1	LS63	AND GATE
15	1	LS64	AND GATE
16	1	LS65	AND GATE
17	1	LS66	AND GATE
18	1	LS67	AND GATE
19	1	LS68	AND GATE
20	1	LS69	AND GATE
21	1	LS70	AND GATE
22	1	LS71	AND GATE
23	1	LS72	AND GATE
24	1	LS73	AND GATE
25	1	LS74	AND GATE
26	1	LS75	AND GATE
27	1	LS76	AND GATE
28	1	LS77	AND GATE
29	1	LS78	AND GATE
30	1	LS79	AND GATE
31	1	LS80	AND GATE
32	1	LS81	AND GATE
33	1	LS82	AND GATE
34	1	LS83	AND GATE
35	1	LS84	AND GATE
36	1	LS85	AND GATE
37	1	LS86	AND GATE
38	1	LS87	AND GATE
39	1	LS88	AND GATE
40	1	LS89	AND GATE
41	1	LS90	AND GATE
42	1	LS91	AND GATE
43	1	LS92	AND GATE
44	1	LS93	AND GATE
45	1	LS94	AND GATE
46	1	LS95	AND GATE
47	1	LS96	AND GATE
48	1	LS97	AND GATE
49	1	LS98	AND GATE
50	1	LS99	AND GATE
51	1	LS100	AND GATE

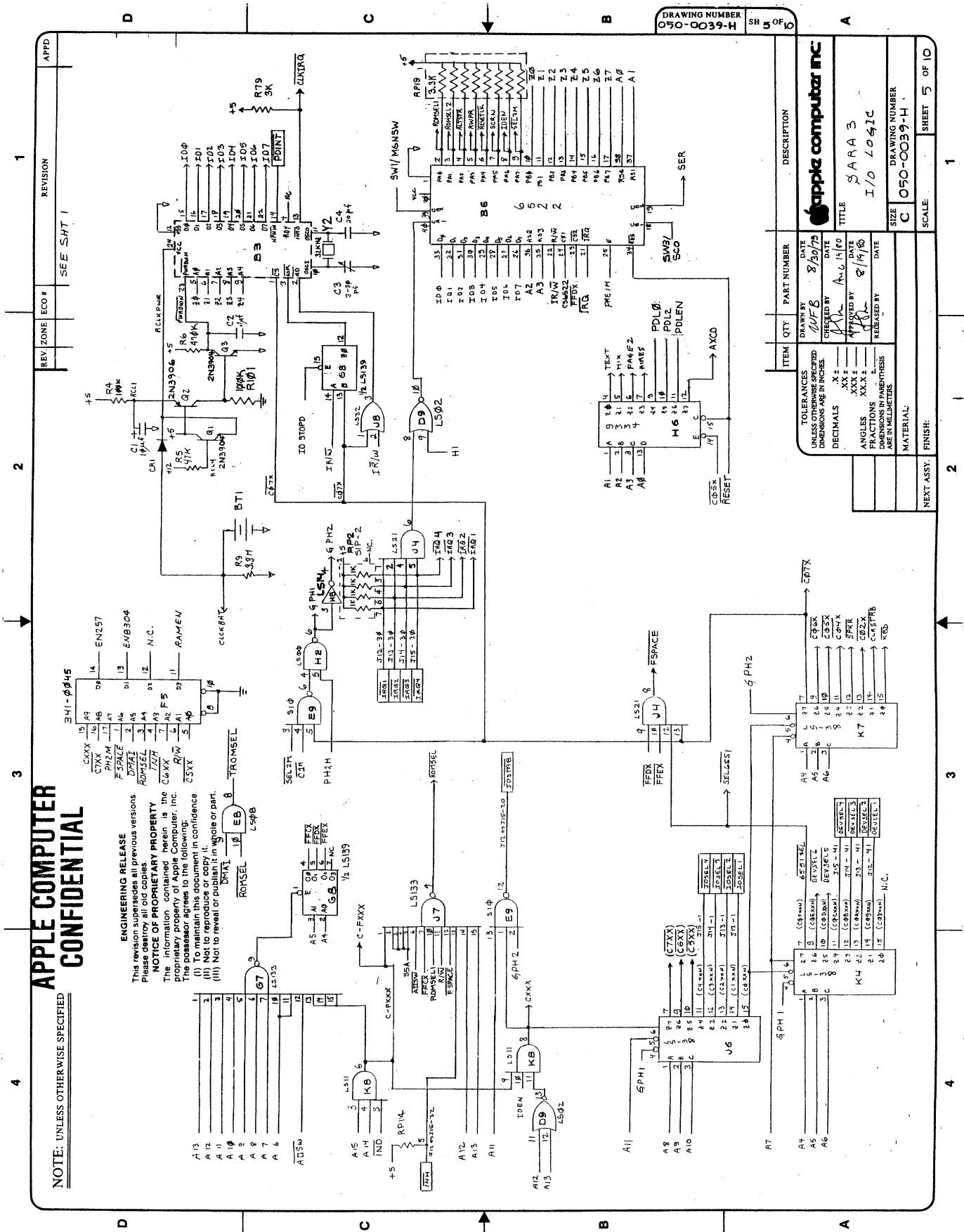
DRAWING NUMBER
050-0039-11
SHEET 4 OF 10

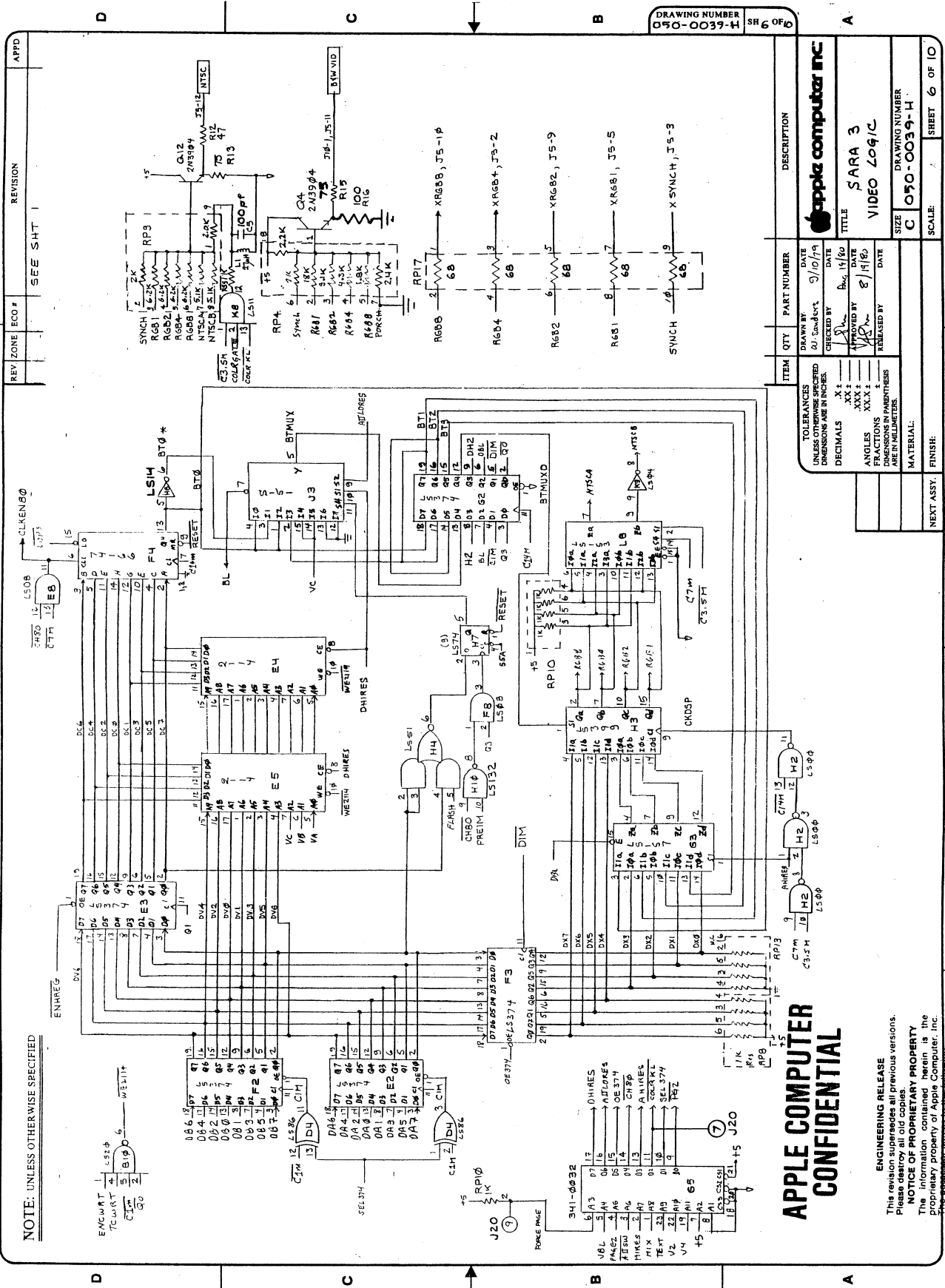
apple computer inc.
TITLE
SARA 3
MICRO-PROCESSOR LOGIC
DATE
8/1/80
DRAWN BY
SARA 3
CHECKED BY
SARA 3
DATE
8/1/80
RELEASER BY
SARA 3

TOLERANCES
UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES
DECIMALS .XX
FRACTIONS 1/16
ANGLES XXX.X
HOLE POSITION UNLESS OTHERWISE SPECIFIED
ARE IN MILLIMETERS
MATERIAL:
FINISH:

ENGINEERING RELEASE
This revision supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the
proprietary property of Apple Computer, Inc.
The possessor agrees to the following:
(i) To maintain this document in confidence
(ii) Not to reproduce, copy, or disseminate
(iii) Not to reveal or publish it in whole or part.

SCALE: SHEET 4 OF 10





REV ZONE	ECO #	SEE SHT 1
REVISION		
1		

DRAWING NUMBER
050-0039-H SH 6 OF 10

DRAWN BY		DATE	
CHECKED BY		DATE	
APPROVED BY		DATE	
RELEASED BY		DATE	

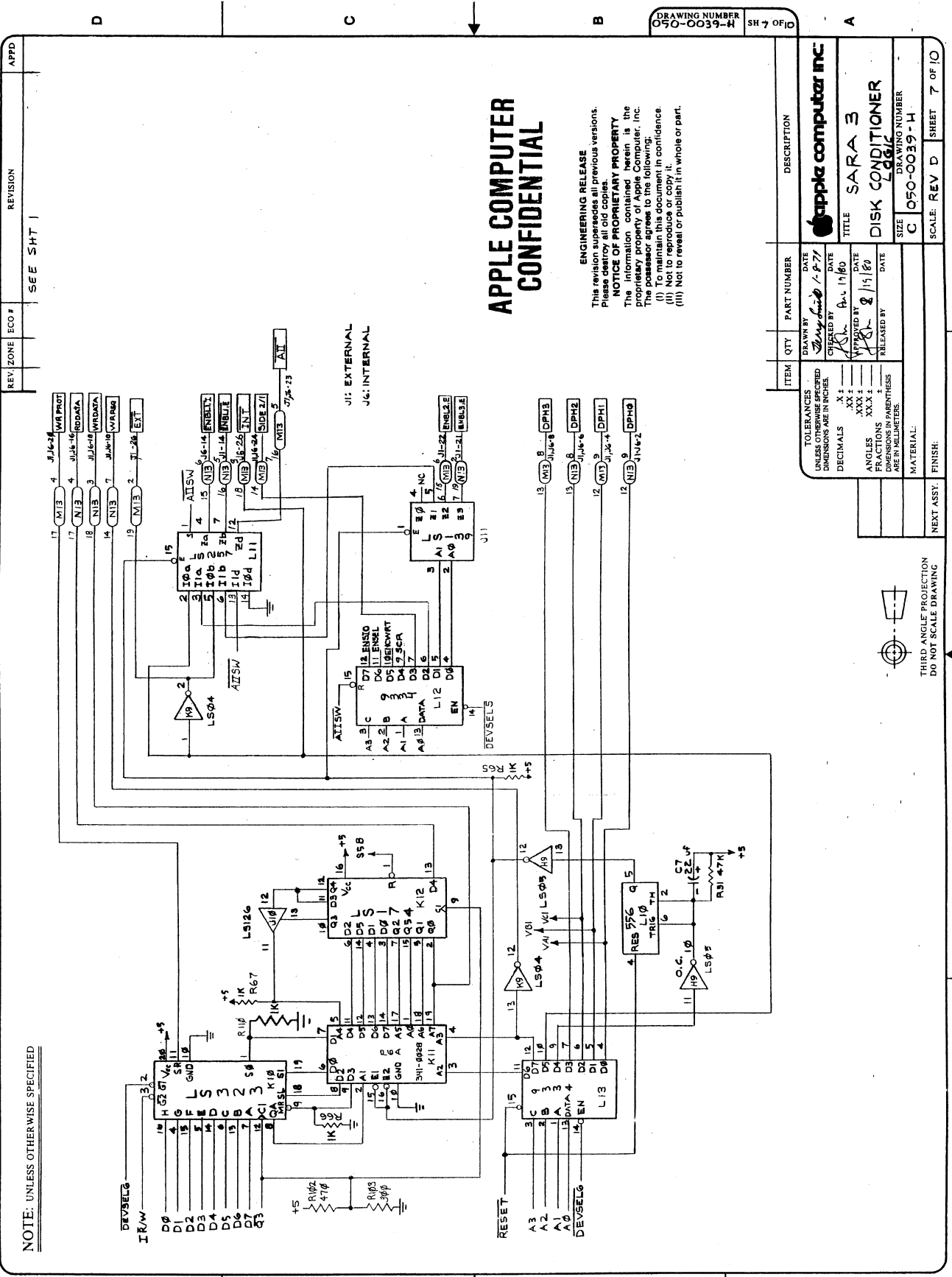
TOLERANCES UNLESS OTHERWISE SPECIFIED	
DECIMALS	XXX ±
FRACTIONS	XXX ±
DIMENSIONS IN PARENTHESES ARE IN MILLIMETERS	
MATERIAL:	
FINISH:	
NEXT ASSY:	

apple computer inc
SARA 3
VIDEO LOGIC
DRAWING NUMBER
050-0039-H
SCALE: 1
SHEET 6 OF 10

NOTE: UNLESS OTHERWISE SPECIFIED

**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the
proprietary property of Apple Computer, Inc.
No part of this document may be reproduced
without the prior written permission of
Apple Computer, Inc.

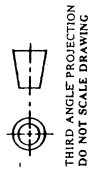


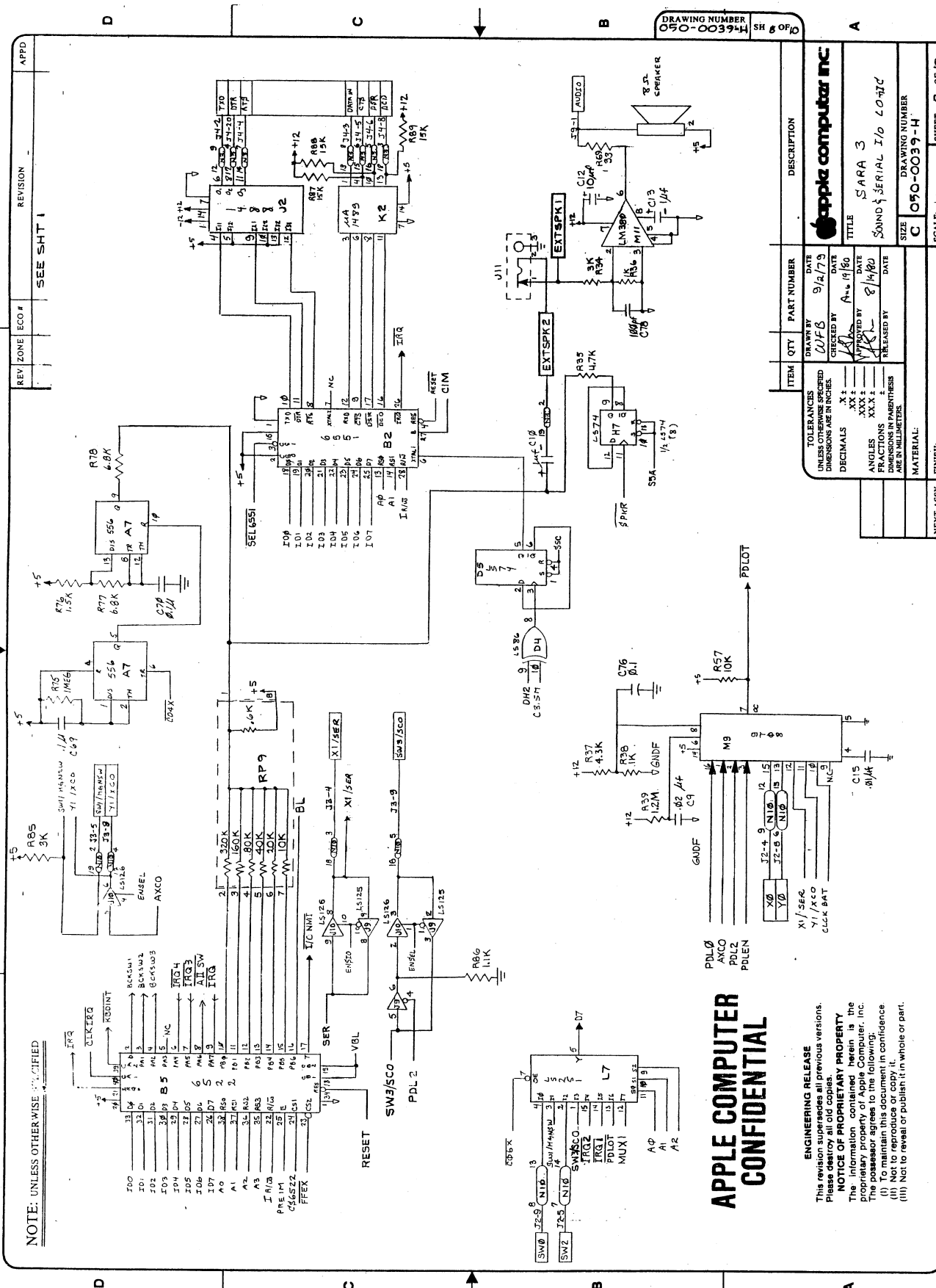
NOTE: UNLESS OTHERWISE SPECIFIED

APPLE COMPUTER CONFIDENTIAL

ENGINEERING RELEASE
This release supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(i) Not to reproduce or copy it.
(ii) Not to reveal or publish it in whole or part.

DRAWING NUMBER 050-0039-H		SH 7 OF 10	
ITEM	QTY	PART NUMBER	DESCRIPTION
TOLERANCES UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES DECIMALS .XX ± ANGLES .XXX ± DIMENSIONS IN PARENTHESES ARE IN MILLIMETERS. MATERIAL: FINISH: NEXT ASSY:			
DRAWN BY A. J. P. 1/1/80		DATE 1/1/80	
CHECKED BY A. J. P. 1/1/80		DATE 1/1/80	
APPROVED BY A. J. P. 1/1/80		DATE 1/1/80	
RELEASED BY A. J. P. 1/1/80		DATE 1/1/80	
TITLE SARA 3 DISK CONDITIONER SIZE DRAWING NUMBER C 050-0039-H SCALE: REV D SHEET 7 OF 10			





NOTE: UNLESS OTHERWISE SPECIFIED

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

REV. ZONE ECO # SEE SHT 1

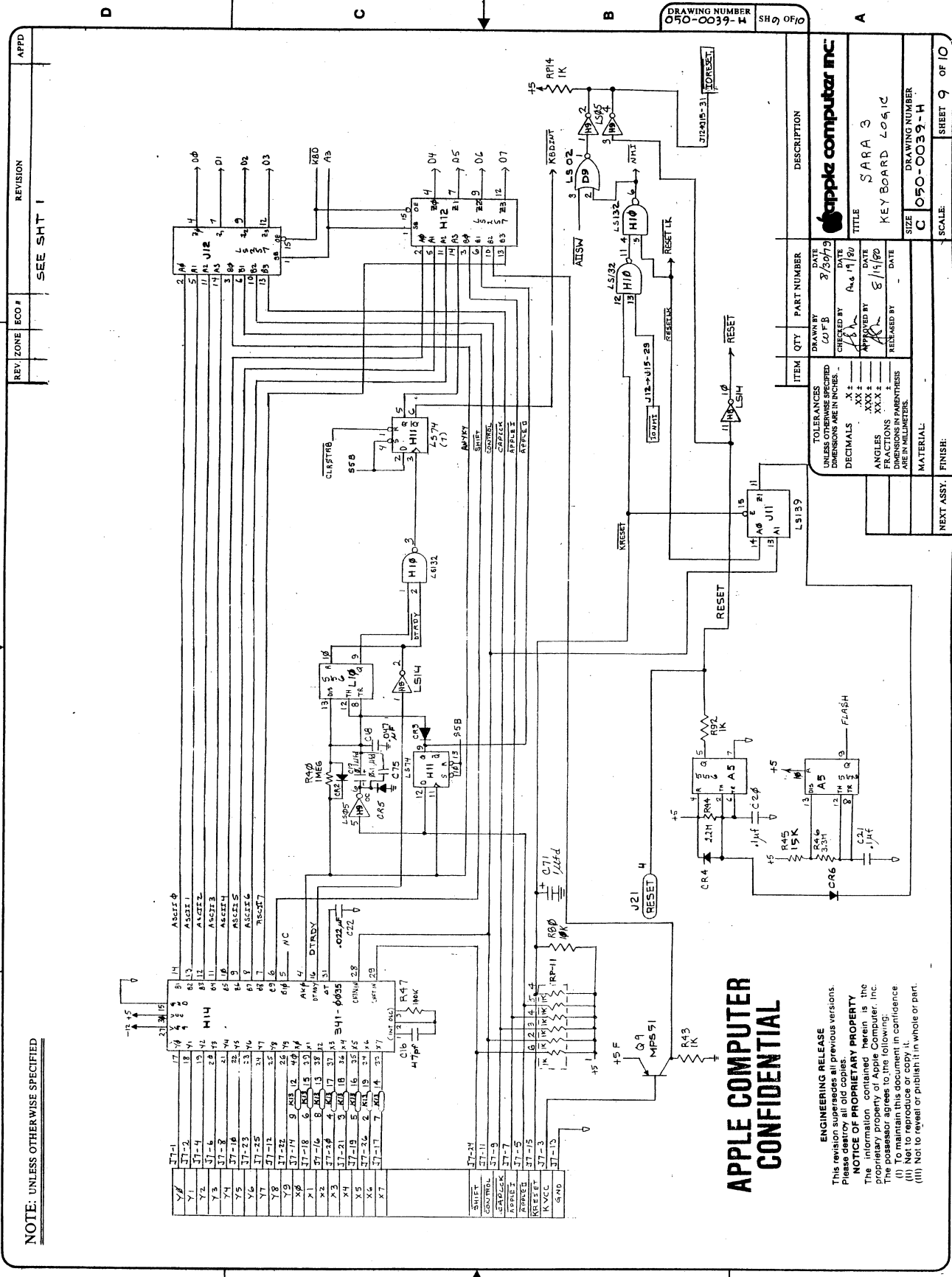
DRAWING NUMBER 1050-0039-H SH # OF 1

ITEM	QTY	PART NUMBER	DESCRIPTION
apple computer inc			
TITLE SARA 3			
SOUND SERIAL I/O CONT			
SIZE	DRAWING NUMBER		SHEET B OF 10
C	1050-0039-H		1

TOLERANCES UNLESS OTHERWISE SPECIFIED	DATE	DATE
DIMENSIONS ARE IN INCHES	9/2/79	9/2/79
DECIMALS .XX	APPROVED BY	APPROVED BY
ANGLES .XXX	8/14/80	8/14/80
FRACTIONS ARE IN MILLIMETERS	RELEASED BY	RELEASED BY
MATERIAL:		
FINISH:		
NEXT ASSY:		

APPLE COMPUTER CONFIDENTIAL

ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(I) To maintain this document in confidence.
(II) Not to reproduce or copy it in whole or part.
(III) Not to reveal or publish it in whole or part.



NOTE: UNLESS OTHERWISE SPECIFIED

REV. ZONE	ECO #	1
REVISION		SEE SHT 1

DRAWING NUMBER
050-0039-H SH 0 OF 10

apple computer inc	
TITLE	SARA 3
DESCRIPTION	KEY BOARD LOGIC
DATE	8/15/80
APPROVED BY	[Signature]
REVISION	
SIZE	C 050-0039-H
DRAWING NUMBER	
SHEET	9 OF 10

TOLERANCES UNLESS OTHERWISE SPECIFIED:	ITEM	QTY	PART NUMBER	DESCRIPTION
DECIMALS .XX				
ANGLES XXX.X				
FRACTIONS XX/XX				
FINISH: ARE IN MILLIMETERS				
MATERIAL:				
NEXT ASSY:				
FINISH:				

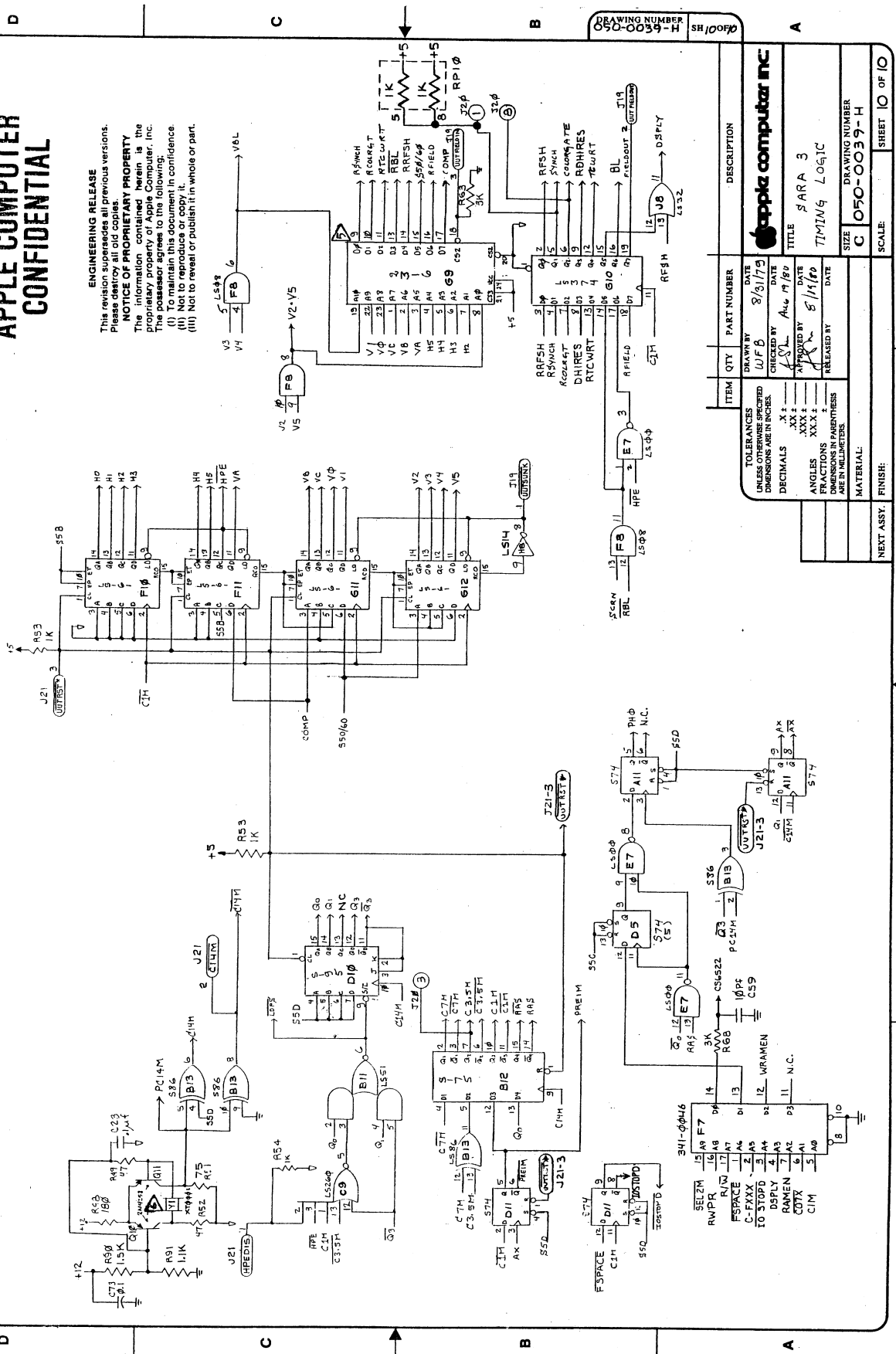
**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(i) Not to maintain this document in confidence
(ii) Not to repeat or publish it in whole or part.

REV. ZONE	ECO #	REVISION
		SEE SHT 1

APPLE COMPUTER CONFIDENTIAL

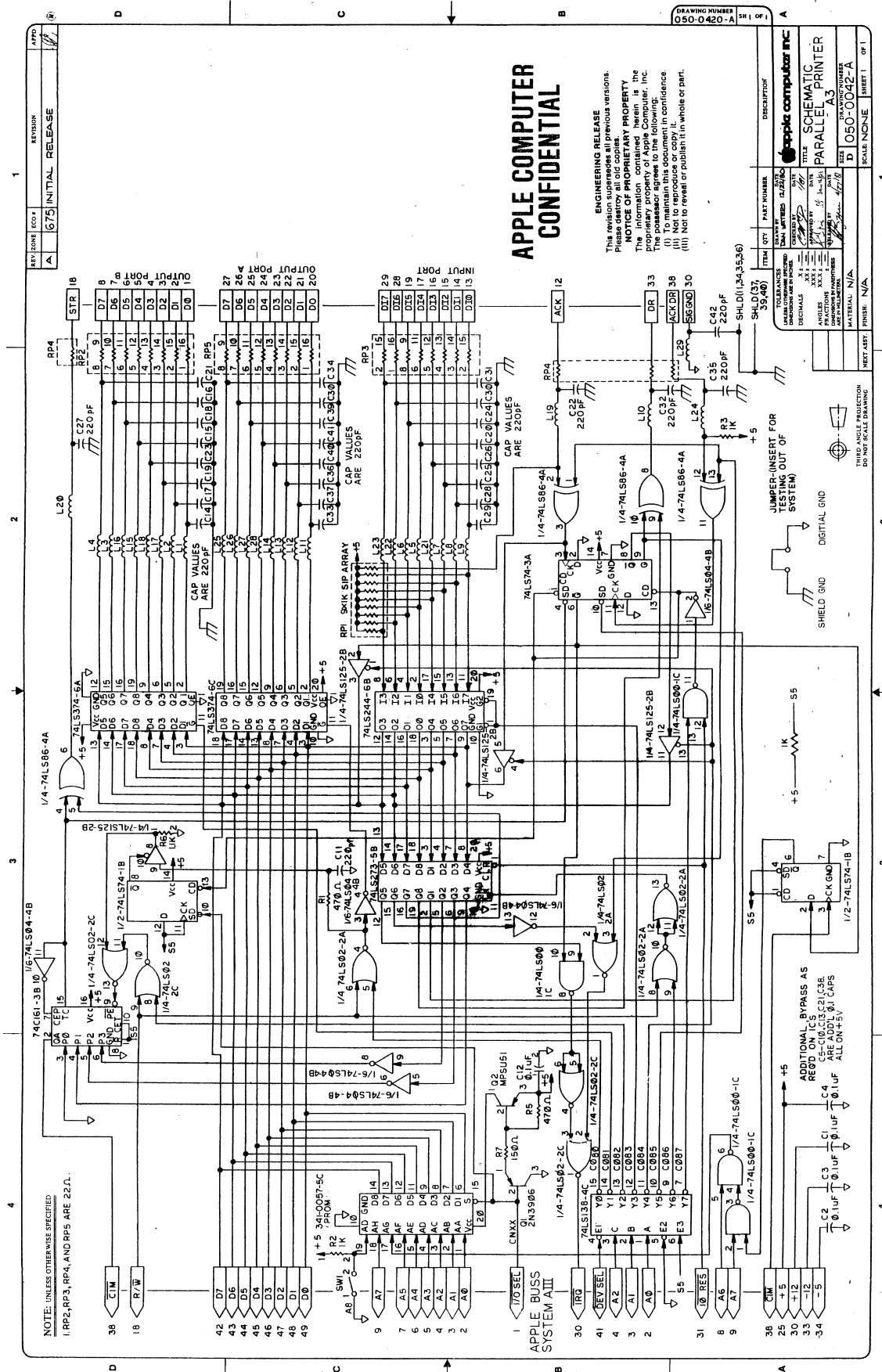
ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY:
This document contains information that is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(I) To maintain this document in confidence.
(II) Not to reproduce or copy it.
(III) Not to reveal or publish it in whole or part.



NOTE: UNLESS OTHERWISE SPECIFIED

DRAWING NUMBER	SH/00FP0
050-0039-H	

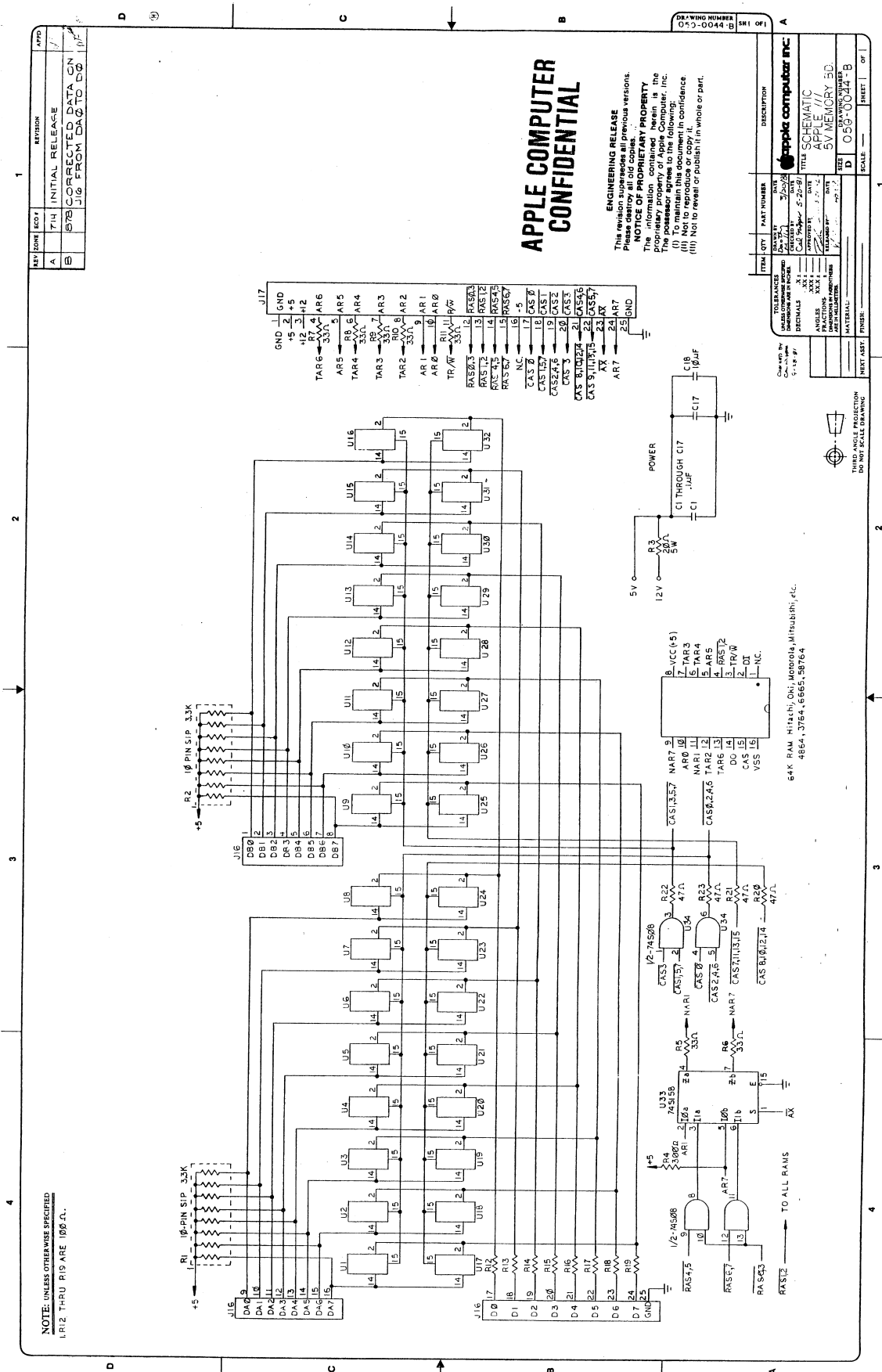
DRAWING NUMBER		SH/00FP0	
050-0039-H			
DRAWN BY		DATE	
WFB		8/31/79	
CHECKED BY		DATE	
AUG		8/17/80	
APPROVED BY		DATE	
SARA 3		8/11/80	
RELEASED BY		DATE	
TITLE		DRAWING NUMBER	
SARA 3		050-0039-H	
SIZE		SCALE	
C			
NEXT ASSY.		FINISH:	
SHEET 10 OF 10			



**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This release supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the property of Apple Computer, Inc. The recipient agrees to the following:
(i) To maintain this document in confidence.
(ii) Not to reproduce or disseminate this document.
(iii) Not to reveal or publish it in whole or part.

ITEM	QTY	PART NUMBER	DESCRIPTION
1	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
2	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
3	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
4	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
5	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
6	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
7	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
8	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
9	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
10	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
11	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
12	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
13	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
14	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
15	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
16	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
17	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
18	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
19	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
20	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
21	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
22	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
23	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
24	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
25	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
26	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
27	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
28	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
29	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
30	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
31	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
32	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
33	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
34	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
35	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
36	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
37	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
38	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
39	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
40	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
41	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
42	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
43	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
44	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
45	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
46	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
47	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
48	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
49	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
50	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
51	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
52	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
53	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
54	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
55	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
56	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
57	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
58	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
59	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
60	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
61	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
62	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
63	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
64	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
65	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
66	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
67	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
68	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
69	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
70	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
71	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
72	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
73	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
74	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
75	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
76	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
77	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
78	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
79	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
80	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
81	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
82	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
83	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
84	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
85	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
86	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
87	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
88	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
89	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
90	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
91	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
92	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
93	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
94	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
95	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
96	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
97	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
98	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
99	1	050-0420-A	SCHEMATIC PARALLEL PRINTER
100	1	050-0420-A	SCHEMATIC PARALLEL PRINTER



NOTE: UNLESS OTHERWISE SPECIFIED
1R12 THRU R19 ARE 100 Ω.

**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the property of Apple Computer, Inc. The possessor agrees to the following:
(i) To maintain this document in confidence.
(ii) Not to reveal or publish it in whole or part.

REV. NO.	ECOY	REVISION
A	714	INITIAL RELEASE
B	578	CORRECTED DATA ON J16 FROM DATA ON J17

DATE	1982-04-14
BY	059-0044-B
SCALE	1
SHEET	1
OF	1

ITEM	QTY	PART NUMBER	DESCRIPTION
U1-U16	16	64K16	64K RAM
U17	1	74181	RAM CONTROLLER
U18	1	74138	RAM DECODER
U19	1	74125	RAM BUFFER
U20-U25	6	74105	RAM DRIVER
U26-U32	7	74105	RAM SENSE AMPLIFIER
U33-U38	6	74105	RAM REFRESH
U39-U44	6	74105	RAM PARITY
U45	1	74105	RAM PARITY GENERATOR
U46	1	74105	RAM PARITY CHECKER
U47	1	74105	RAM PARITY GENERATOR
U48	1	74105	RAM PARITY CHECKER
U49	1	74105	RAM PARITY GENERATOR
U50	1	74105	RAM PARITY CHECKER
U51	1	74105	RAM PARITY GENERATOR
U52	1	74105	RAM PARITY CHECKER
U53	1	74105	RAM PARITY GENERATOR
U54	1	74105	RAM PARITY CHECKER
U55	1	74105	RAM PARITY GENERATOR
U56	1	74105	RAM PARITY CHECKER
U57	1	74105	RAM PARITY GENERATOR
U58	1	74105	RAM PARITY CHECKER
U59	1	74105	RAM PARITY GENERATOR
U60	1	74105	RAM PARITY CHECKER

DATE	1982-04-14
BY	059-0044-B
SCALE	1
SHEET	1
OF	1

DATE	1982-04-14
BY	059-0044-B
SCALE	1
SHEET	1
OF	1

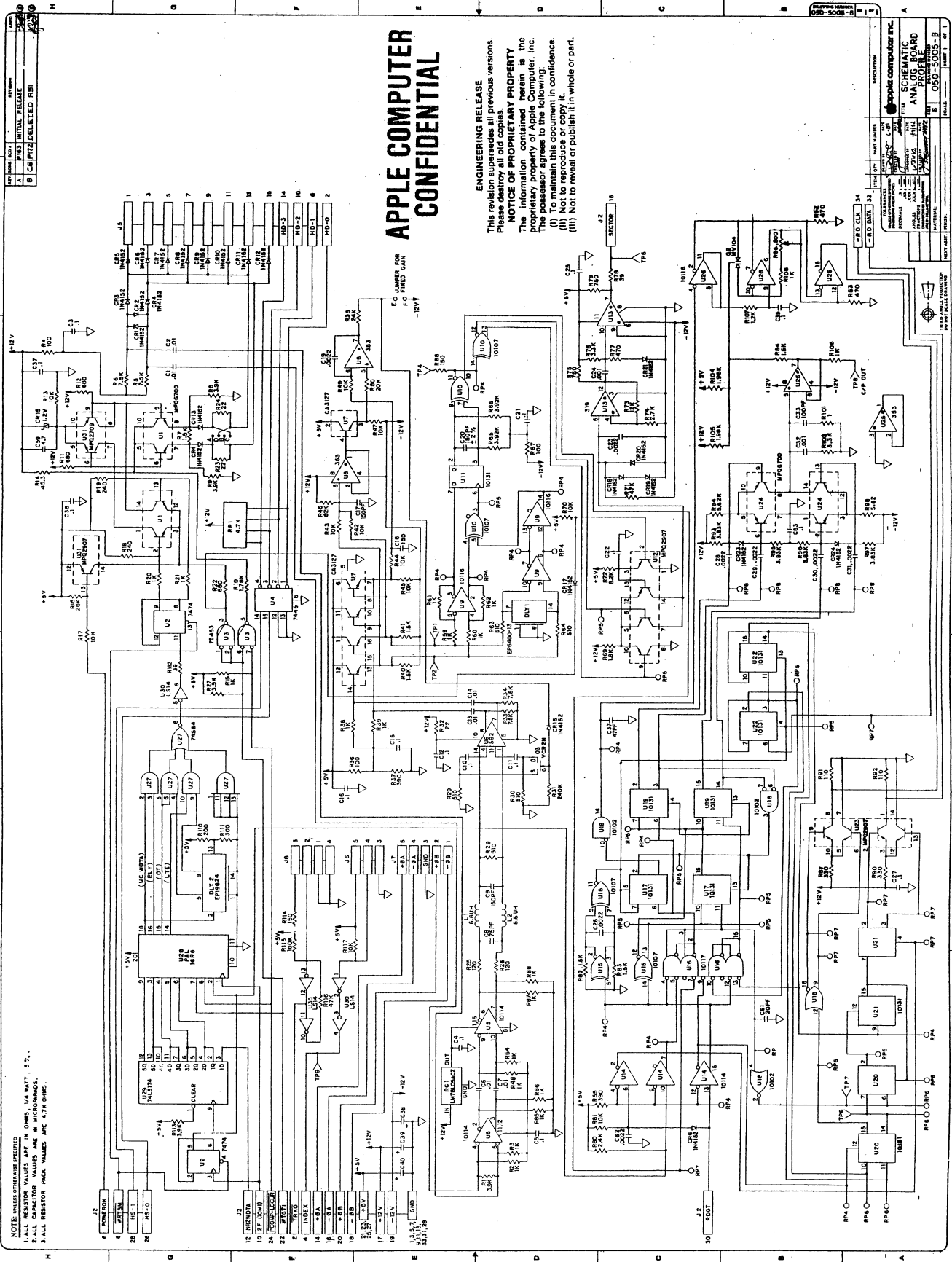
DATE	1982-04-14
BY	059-0044-B
SCALE	1
SHEET	1
OF	1

THIRD ANGLE PROJECTION
DO NOT SCALE DRAWING

**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions.
Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the
proprietary property of Apple Computer, Inc.
The possessor agrees to the following:
(i) To maintain this document in confidence.
(ii) Not to reproduce or copy it.
(iii) Not to reveal or publish in whole or part.

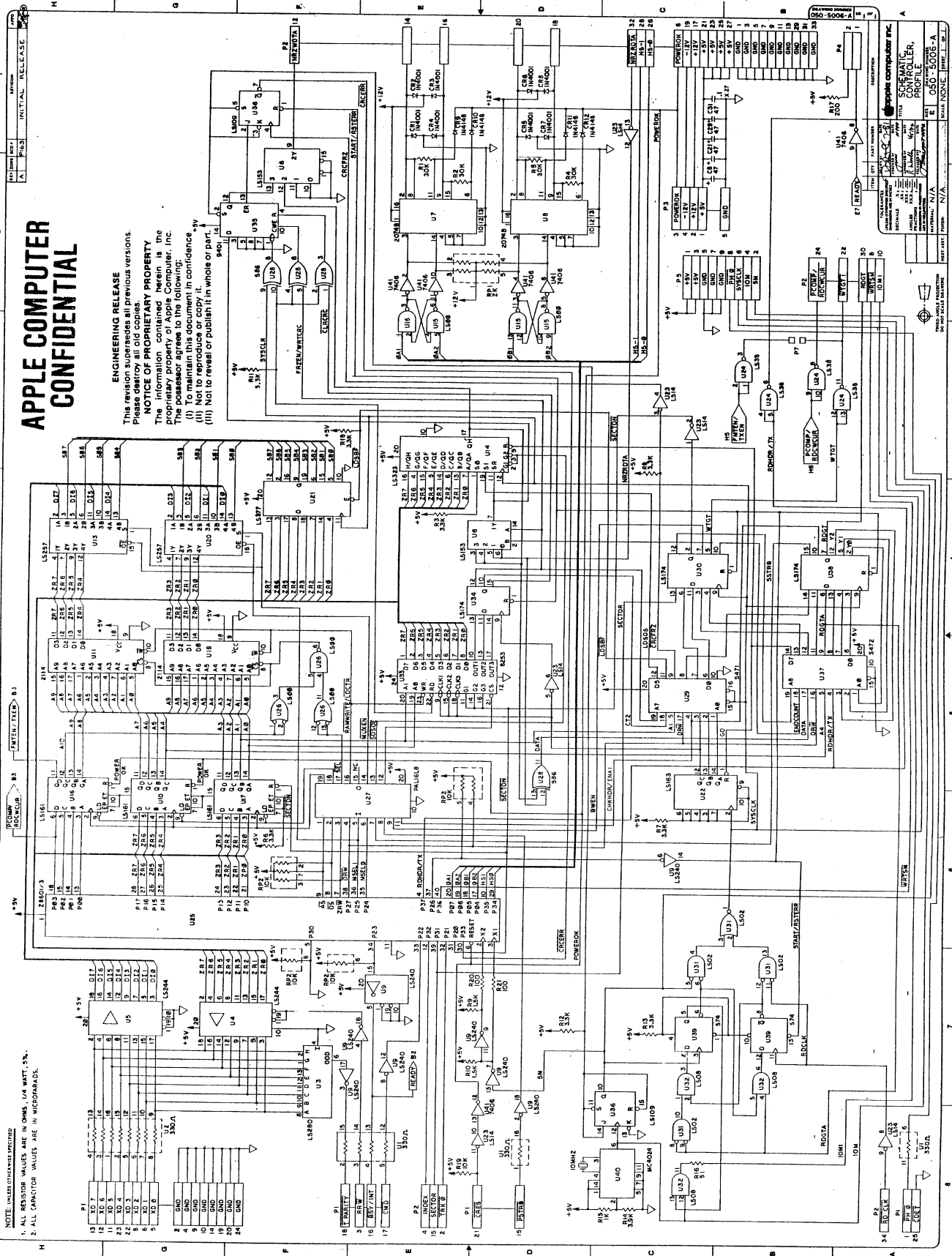
Apple Computer Inc.
SHEMATIC
ANALOG BOARD
PROJECT
050-5005-B
REV. 1 OF 1



NOTE: UNLESS OTHERWISE SPECIFIED
1. ALL RESISTOR VALUES ARE IN OHMS, 1/4 WATT, 5%.
2. ALL CAPACITOR VALUES ARE IN MICROFARADS.
3. ALL RESISTOR PACK VALUES ARE 4% TOL.

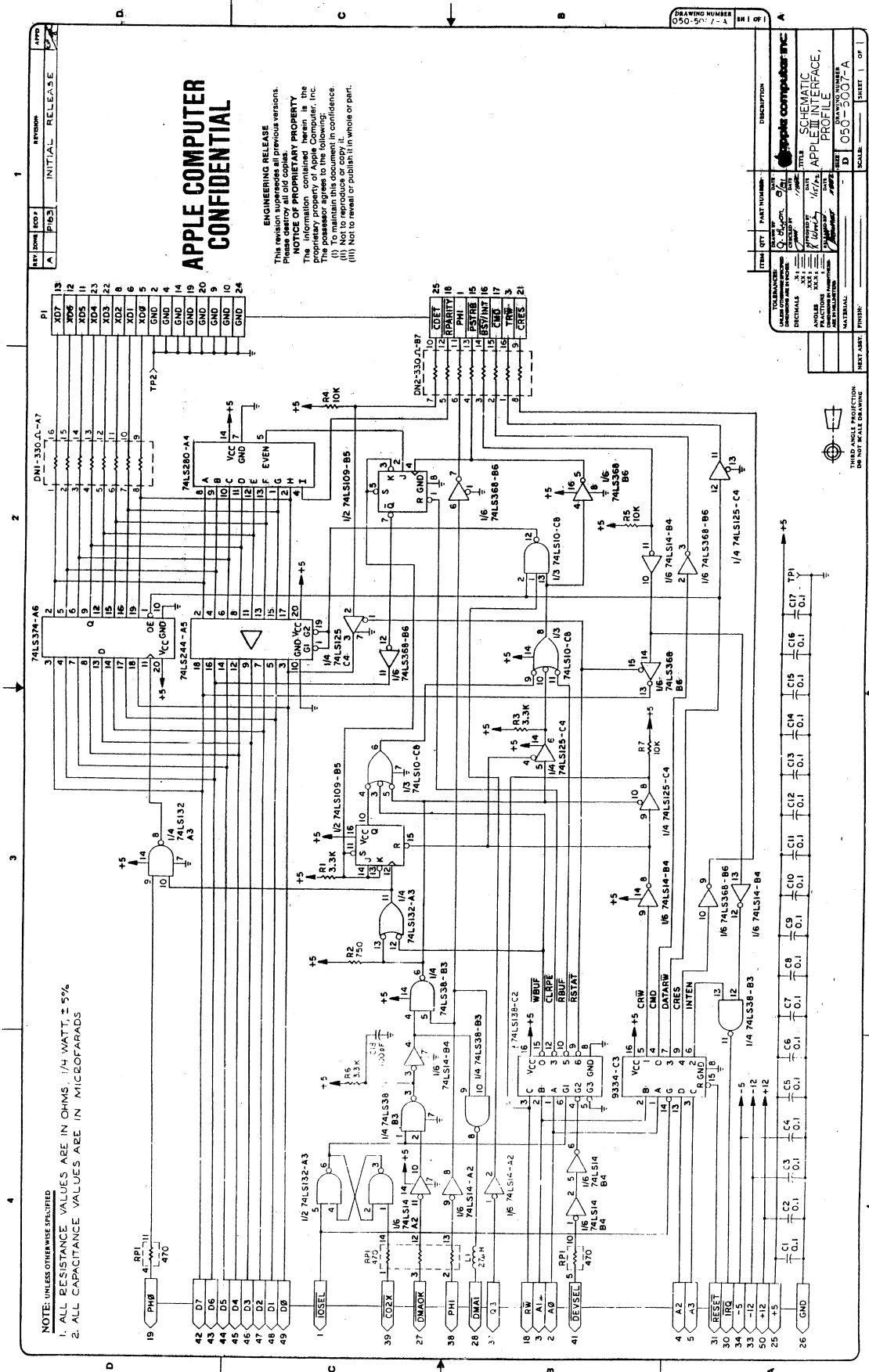
**APPLE COMPUTER
CONFIDENTIAL**

ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
This document contains confidential information of Apple Computer, Inc. The possessor agrees to the following:
(i) To maintain this document in confidence.
(ii) Not to reproduce or copy it.
(iii) Not to reveal or publish it in whole or part.



NOTE: UNLESS OTHERWISE SPECIFIED
1. ALL RESISTOR VALUES ARE IN OHMS, 1/4 WATT, 5%.
2. ALL CAPACITOR VALUES ARE IN MICROFARADS.

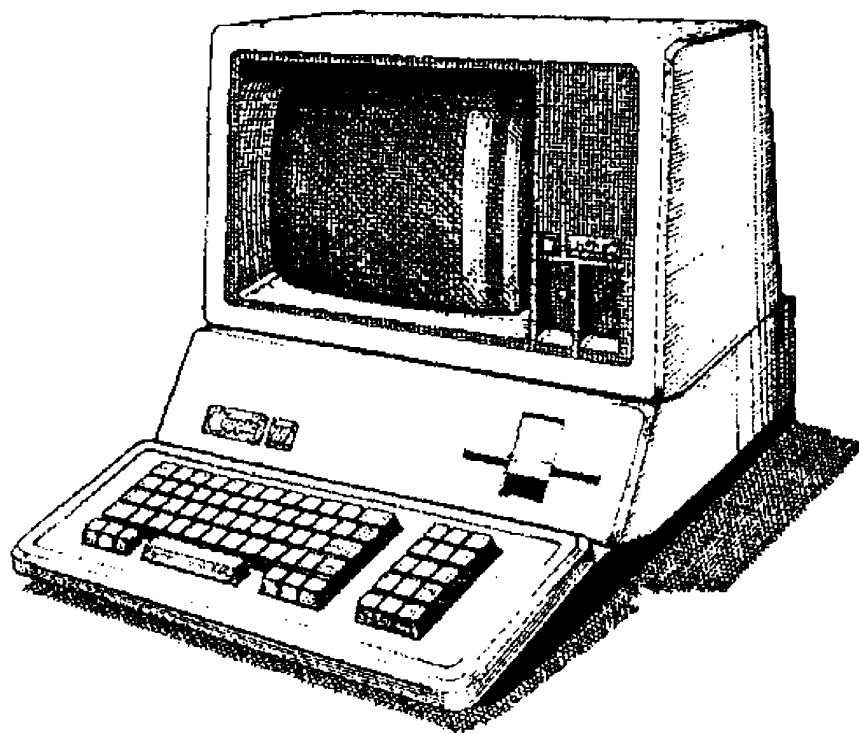
Apple Computer Inc.
SHEWAN TOMES & CO. LTD.
SCHEMATIC
DRAWING
NO. 050-5005-A
REV. 1
DATE: 11/18/79
DRAWN: J. J. WILSON
CHECKED: J. J. WILSON
APPROVED: J. J. WILSON
PARTS LIST FROM: N/A
PART NO. 050-5005-A
REV. 1
DATE: 11/18/79
DRAWN: J. J. WILSON
CHECKED: J. J. WILSON
APPROVED: J. J. WILSON





Apple /// Computer Information

Apple /// Service Reference Manual



Section I of II • Theory of Operation

Chapter 12 • Floppy Disk Subsystem

Written by Apple Computer • 1982



THE DISK /// SUBSYSTEM

I THEORY OF OPERATION

The Disk /// subsystem, is a self contained Apple /// peripheral which allows user programs and data to be stored and retrieved on 5 1/4" floppy diskettes. The Apple /// supplies DC power, control signals, and a parallel data path to the Disk /// via the A///s main logic board Disk Conditioner (Controller) Circuit. The Disk Conditioner sends DC power, control signals, and serial data to the Analog Card via a 26-conductor ribbon cable.

The Analog Card contains disk read-write electronics, drivers for positioning Stepper Motor, and a transistor power switch. Analog Card also contains circuitry which causes its output signals to the disk conditioner circuit to be active only when the card is enabled. This allows up to 4 Analog Cards to share the same data path for a 4 drive system (one internal, three external drives).

Within the drive itself, movement of Stepper Motor rotates Actuator Cam. Head and Carriage Assembly's Cam Follower, rides in Actuator Cam's spiral groove. Two Guide Rods allow motion of Head and Carriage Assembly either towards, or away from Drive Door. This positions Read/Write Head to appropriate track, so that serial data may then be transferred to and from disk using high-level commands.

A - DISK CONDITIONER CIRCUIT

There are seven sections in the Disk Conditioner circuit:
(Refer to Schematic of Disk Conditioner Logic)

1. Power-On Reset

This circuit consists of one half of 556 Timer and one open-collector inverter. When power is applied, the outputs of the 9334/LS259 Address Latch (DPh0-3) are brought low which places the Disk Conditioning logic in the read mode, the Q output of the 556 timer (pin 5) goes high for about 65 ms, bringing inverter's output to ground. This resets the State Machine (Prom P6A), and the Drive Enable Multiplexers. The drive enable multiplexer's Z output is prevented from enabling the internal and external drives.

2. The BOOT ROM

The Boot ROM, though not a part of the disk conditioning circuit, contains the routine which down loads the operating system (SOS) from disk and then jumps into it. The LS323, an 8 Bit Parallel/Serial Register, is enabled (DEVSEL6*) whenever the address COExxn is presented. the data, from the boot rom, is converted to serial at the output of pin 1 in the 74LS323 in a write operation.

3. Addressable Latch (9334 or 74LS259).

**APPLE
CORP.**

This revision
Please describe
NOTICE
The information
is proprietary
to Apple Computer, Inc.
(I) To maintain
(II) Not to
(III) Not to

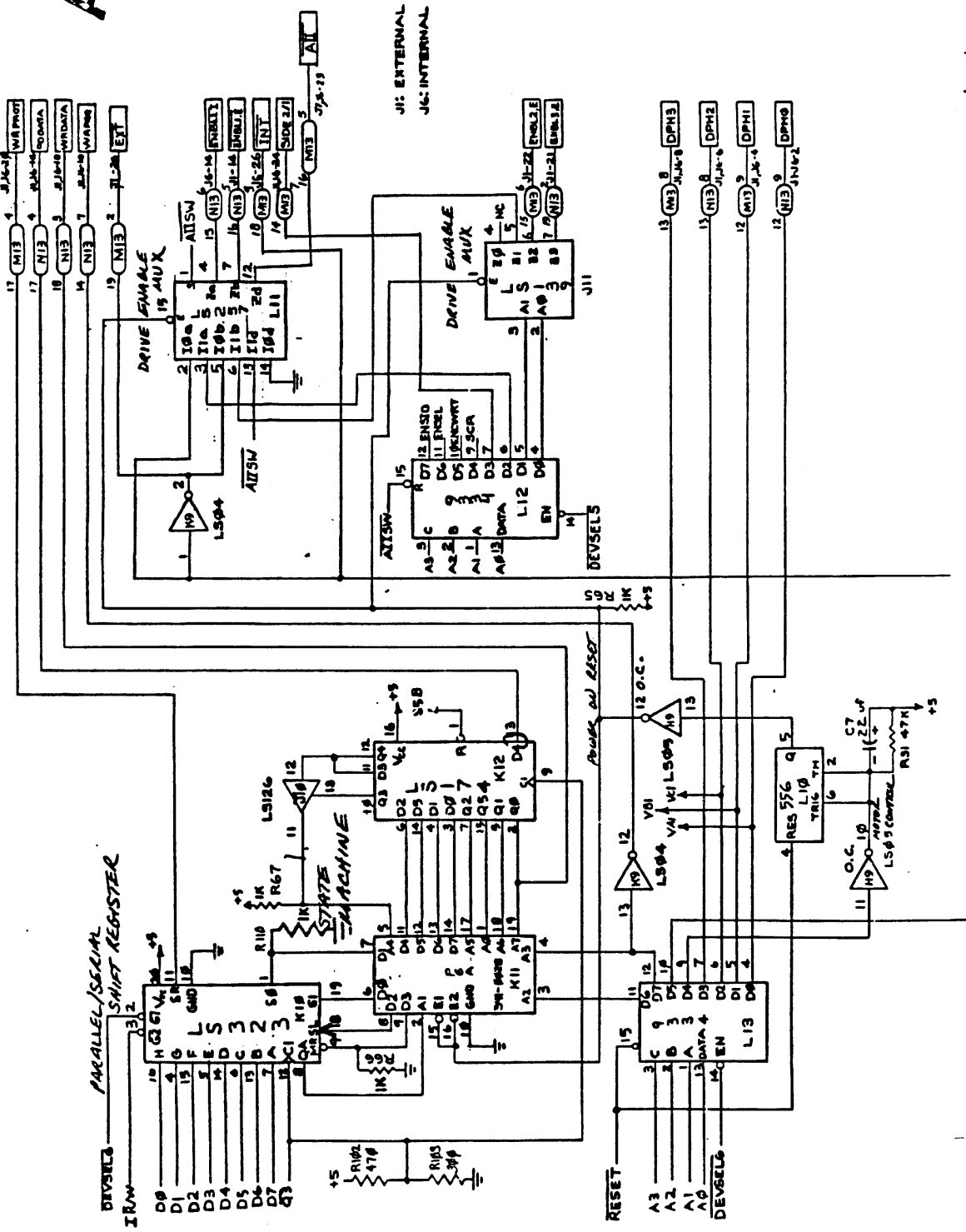
REV	ZONE	ECO #	
G		778	SEE SHT 1

2

3

4

NOTE: UNLESS OTHERWISE SPECIFIED



J1: EXTERNAL
J2: INTERNAL

ITEM	QTY.	PART NUMBER
------	------	-------------



This chip provides an eight software-controlled output. When DEVSEL6* (pin 9 of LS138 I/O Address Decoder) signal goes low (one of 16 addresses starting at [COExxt], value of Address Line A0 becomes the new value of D output pointed to by Address Lines A1-A3. Latch output D6 and D7 set operating mode of controller (read, write, etc). Q5 selects the internal drive. D4 controls MOTOR ON signal and D0 to D3, set position of the Stepper Motor. Drive enable signals ENBL1I and ENBL1E are true when MOTOR ON signal is true and appropriate drive is selected.

4. State Machine (PROM P6A and 74LS174)

These two parts contain nucleus of Disk Conditioning Logic. A State Machine is a device capable of storing a value in a register. That value, in conjunction with external input, determines what the next register value should be and what output should be generated. Value in register is, machine's Current State and next value is machine's Next State. State machine updates at each clocking, going from state to state and producing output base on state its in and value of its input just before clocking.

Current State, is value at Q2, Q5, Q1, and Q0 from 74LS174. Next state is value of D4-D7 from P6A PROM. Input to State Machine includes D6 and D7 from 9334, QA* output from 74LS174 Shift Register and A2-11. Output from State Machine is PROM signals D0-D3, which effect Shift Register. Q0 from 74LS174, which is high-order bit of Current State, forms WR DATA output. Since State Machine is clocked by two Megahertz Q3* signal, the state lasts for 500 ns.

Flux transitions on diskette are detected by Analog Card and are sent to Disk Conditioning Circuitry as RD DATA. These one microsecond positive going pulses appear at D4 input to 74LS174. NAND Gate and Inverter, which connect to 74LS174's Q3 and Q4 output, cause A4 input to P6A PROM to go low for one state following the falling edge of RD DATA. This information is ignored when writing out to disk. When reading from disk, however, this input forms the basis for determining whether a logic one or zero has been read. A logic one is indicated by a four microsecond period. A logic zero, by an eight microsecond period between flux transitions.

5. Shift Register

74LS323 Universal 8 Bit Parallel/Serial Shift Register, transfers data to and from the Apple ///. When writing to disk, Shift Register under control of State Machine, parallel loads a data byte from the A/// and shifts it left. This causes QA* output to effect State Machine's A1 input. State Machine goes through a different state sequence for a logic one and a logic zero, which causes WR DATA to change value every four and eight microseconds, respectively.

When reading from disk, State Machine shifts appropriate logic levels into Shift Register's SL input. Resulting byte can be read by the A///.

Status of Write Protect switch in Disk Drive, comes into SR input of



Shift Register from Analog Card. Under control of State Machine, Shift Register is placed into a shift right mode, allowing status of switch to be read by software.

6. MOTOR ON Circuit.

Q output from one half of 556 timer, forms MOTOR ON signal. MOTOR ON becomes true when B2-6 is brought to ground under software control. An enable is sent to one of the drives, and Drive Motor in selected drive turns on. When software causes B2-6 to become high-impedance in order to turn off drive, C7 and R31 give drive a 2/3 second grace period before MOTOR ON times out. This prevents drive from being turned on and off when repeated accesses are made.

7. AII EMULATION MODE

With the AII SW* true and DEVSEL5* (addressed by CODxxn) selected, the internal and external drives are fooled into operating as an Apple II Disk Drive.

B - ANALOG CARD (Refer to Analog Card Schematic Diagram)

1. Enable Circuit

A Disk Drive connected to the Apple /// is always in one of three operating modes:

a - Read Mode

Flux transitions on diskette are detected by MC3470P Floppy Disk Read Amplifier on Analog Card and are sent to the Disk Conditioner as one microsecond positive-going pulses across RD DATA* line.

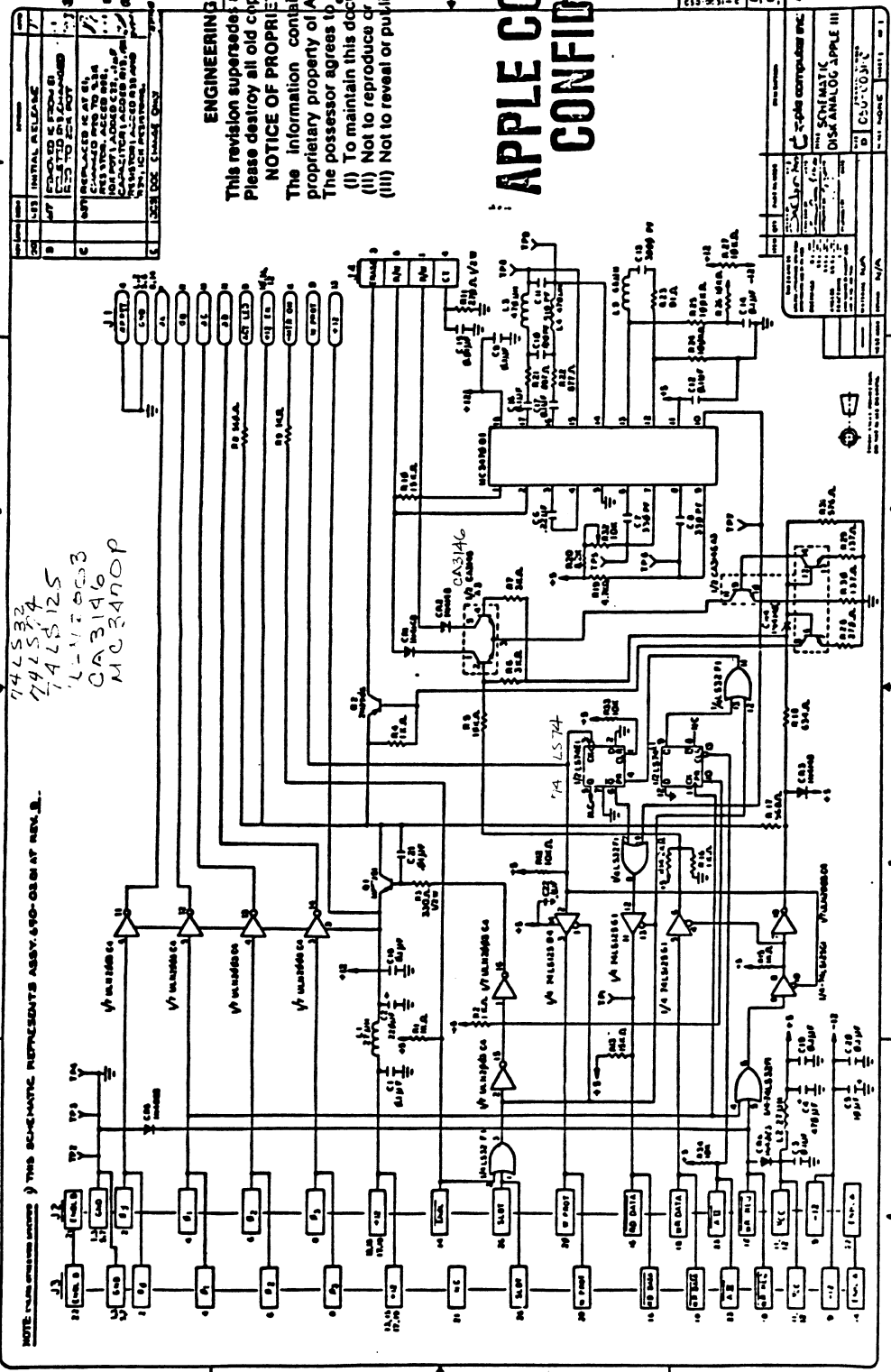
b. Write Mode

Here WR DATA input to Analog Card determines polarity of write current passing through head.

c. Deselected Mode

In this state, drive is not currently performing any data transfers with the Apple ///.

A drive becomes enabled, when ENBL1I input to the Analog Card on the selected drive goes low. This causes Q1 to turn on because current flows through R3 into pin 16 of ULN2003's Darlington output. +12 volts is then supplied to Stepper Motor. Q1 also provides a power source for Write and Erase Current circuits. In addition, ENBL1I signal also enable RD DATA* and W PROT tristate buffers and supplies a MTR ON control signal to Motor Control Board through resistor R9.



NOTE: THIS SCHEMATIC REPRESENTS ASSY. 950-0281 AT REV. B.

74LS139
74LS14
74LS125
74LS04
74LS08
74LS10

ENGINEERING RELEASE
This revision supersedes all previous versions. Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
(I) To maintain this document in confidence
(II) Not to reproduce or copy it.
(III) Not to reveal or publish it in whole or part.

APPLE CONFIDENTIAL

Apple Computer, Inc.	Model 1
Apple Computer, Inc.	Model 2
Apple Computer, Inc.	Model 3
Apple Computer, Inc.	Model 4
Apple Computer, Inc.	Model 5
Apple Computer, Inc.	Model 6
Apple Computer, Inc.	Model 7
Apple Computer, Inc.	Model 8
Apple Computer, Inc.	Model 9
Apple Computer, Inc.	Model 10
Apple Computer, Inc.	Model 11
Apple Computer, Inc.	Model 12
Apple Computer, Inc.	Model 13
Apple Computer, Inc.	Model 14
Apple Computer, Inc.	Model 15
Apple Computer, Inc.	Model 16
Apple Computer, Inc.	Model 17
Apple Computer, Inc.	Model 18
Apple Computer, Inc.	Model 19
Apple Computer, Inc.	Model 20
Apple Computer, Inc.	Model 21
Apple Computer, Inc.	Model 22
Apple Computer, Inc.	Model 23
Apple Computer, Inc.	Model 24
Apple Computer, Inc.	Model 25
Apple Computer, Inc.	Model 26
Apple Computer, Inc.	Model 27
Apple Computer, Inc.	Model 28
Apple Computer, Inc.	Model 29
Apple Computer, Inc.	Model 30
Apple Computer, Inc.	Model 31
Apple Computer, Inc.	Model 32
Apple Computer, Inc.	Model 33
Apple Computer, Inc.	Model 34
Apple Computer, Inc.	Model 35
Apple Computer, Inc.	Model 36
Apple Computer, Inc.	Model 37
Apple Computer, Inc.	Model 38
Apple Computer, Inc.	Model 39
Apple Computer, Inc.	Model 40
Apple Computer, Inc.	Model 41
Apple Computer, Inc.	Model 42
Apple Computer, Inc.	Model 43
Apple Computer, Inc.	Model 44
Apple Computer, Inc.	Model 45
Apple Computer, Inc.	Model 46
Apple Computer, Inc.	Model 47
Apple Computer, Inc.	Model 48
Apple Computer, Inc.	Model 49
Apple Computer, Inc.	Model 50
Apple Computer, Inc.	Model 51
Apple Computer, Inc.	Model 52
Apple Computer, Inc.	Model 53
Apple Computer, Inc.	Model 54
Apple Computer, Inc.	Model 55
Apple Computer, Inc.	Model 56
Apple Computer, Inc.	Model 57
Apple Computer, Inc.	Model 58
Apple Computer, Inc.	Model 59
Apple Computer, Inc.	Model 60
Apple Computer, Inc.	Model 61
Apple Computer, Inc.	Model 62
Apple Computer, Inc.	Model 63
Apple Computer, Inc.	Model 64
Apple Computer, Inc.	Model 65
Apple Computer, Inc.	Model 66
Apple Computer, Inc.	Model 67
Apple Computer, Inc.	Model 68
Apple Computer, Inc.	Model 69
Apple Computer, Inc.	Model 70
Apple Computer, Inc.	Model 71
Apple Computer, Inc.	Model 72
Apple Computer, Inc.	Model 73
Apple Computer, Inc.	Model 74
Apple Computer, Inc.	Model 75
Apple Computer, Inc.	Model 76
Apple Computer, Inc.	Model 77
Apple Computer, Inc.	Model 78
Apple Computer, Inc.	Model 79
Apple Computer, Inc.	Model 80
Apple Computer, Inc.	Model 81
Apple Computer, Inc.	Model 82
Apple Computer, Inc.	Model 83
Apple Computer, Inc.	Model 84
Apple Computer, Inc.	Model 85
Apple Computer, Inc.	Model 86
Apple Computer, Inc.	Model 87
Apple Computer, Inc.	Model 88
Apple Computer, Inc.	Model 89
Apple Computer, Inc.	Model 90
Apple Computer, Inc.	Model 91
Apple Computer, Inc.	Model 92
Apple Computer, Inc.	Model 93
Apple Computer, Inc.	Model 94
Apple Computer, Inc.	Model 95
Apple Computer, Inc.	Model 96
Apple Computer, Inc.	Model 97
Apple Computer, Inc.	Model 98
Apple Computer, Inc.	Model 99
Apple Computer, Inc.	Model 100



d. Read Electronics

MC3470P Floppy Disk Read Amplifier and associated discrete components provide a one-chip interface between magnetic head of Disk Drive and RD DATA* input to Disk Conditioning circuit of the main logic board. MC3470P contains both analog and digital circuits which cause a TTL compatible pulse to be generated for each positive and negative peak of input signal.

Input voltage from head appears across pins 1 and 2 of MC3470P. This signal passes through an amplifier and is differentially applied to a noise filter made up of R21 and R22, C10 and C11 and L3 and L4. Filter's output feeds back into MC3470P, where a differentiator circuit provides an output proportional to rate of change, with time, of input signal. In addition, a 90 degree phase lead is introduced which causes a zero crossing at differentiator's output to correspond with a peak at its input. L5, C13 and R23 determine characteristics of differentiator. R27 allows for correction of current imbalances within differentiator so that a sinusoidal input waveform produces evenly spaced RD DATA* pulses.

Zero crossings at output of differentiator cause output of a comparator within MC3470P, in conjunction with digital circuits, create a Time Domain Filter which checks against false zero crossing readings due to distorted input waveforms or noise. When a zero crossing is detected, mono #1, formed by R20, R32, and C7, is triggered. At end of its two microsecond period, output of comparator is again checked. If it has not changed (valid zero - crossing), mono #2 gets a trigger pulse that uses R19 and C8 to generate one microsecond RD DATA* pulse at pin 10 of MC3470P.

3. Write Electronics

During Read Mode (WR REQ* high), ULN2003 Darlington output at C4-10, is close to ground potential. This prevents erase current switch Q2 from turning on and disables write current return path. During Write Mode, the anode of CR3 is pulled up to +5.7 V. Q2 receives base current through A3-8, and provides current to erase coil of head. R11 serves as return for erase current, which is roughly 44 milliamps. Erase coil in head straddles both sides of read-write head, preventing write current from spreading into adjacent tracks on diskette.

When writing out to diskette, flux transitions are placed on surface of diskette by changing polarity of current flow in head's read-write coil. Write current enters read-write coil through its center tap, which is connected to return side of erase coil. Two of CA3146's transistors connected to R29 and R30 form a current mirror which drives pin three of CA3146.

This establishes write current return path. When writing out to disk, WR DATA causes a differential voltage to be applied to pins 2 and 4 of CA3146, which causes a differential current flow in write coil. Each polarity reversal places a flux reversal on



diskette. (Current in R/W coil -6.8 ma P-P)

4. Write Protect Circuit

Analog Card only allows erase and write current to be generated when diskette has its write protect notch uncovered. If notch is absent (write protected), the Write Protect switch is held open even though diskette is fully inserted into drive. This causes pullup resistor R12 to disable G1-8, which causes WR REQ* signal to be pulled up to a false level by R15. This prevents write current mirror from supplying write current to head. Write Protect status is sent to Disk Conditioning circuitry on Main Logic board as W PROT.

If Write Protect notch has been uncovered, it causes Write Protect switch within drive to close. Phase 1 signal from the Disk Conditioning circuitry provides return path for current passing through switch. If Phase 1 signal is high, it indicates that Stepper Motor is in one of its two transient states between tracks. Write Protect circuits behave as they do when the diskette is write protected. This provides partial coverage against writing when Stepper Motor is off track.

5. Stepper Motor Drivers

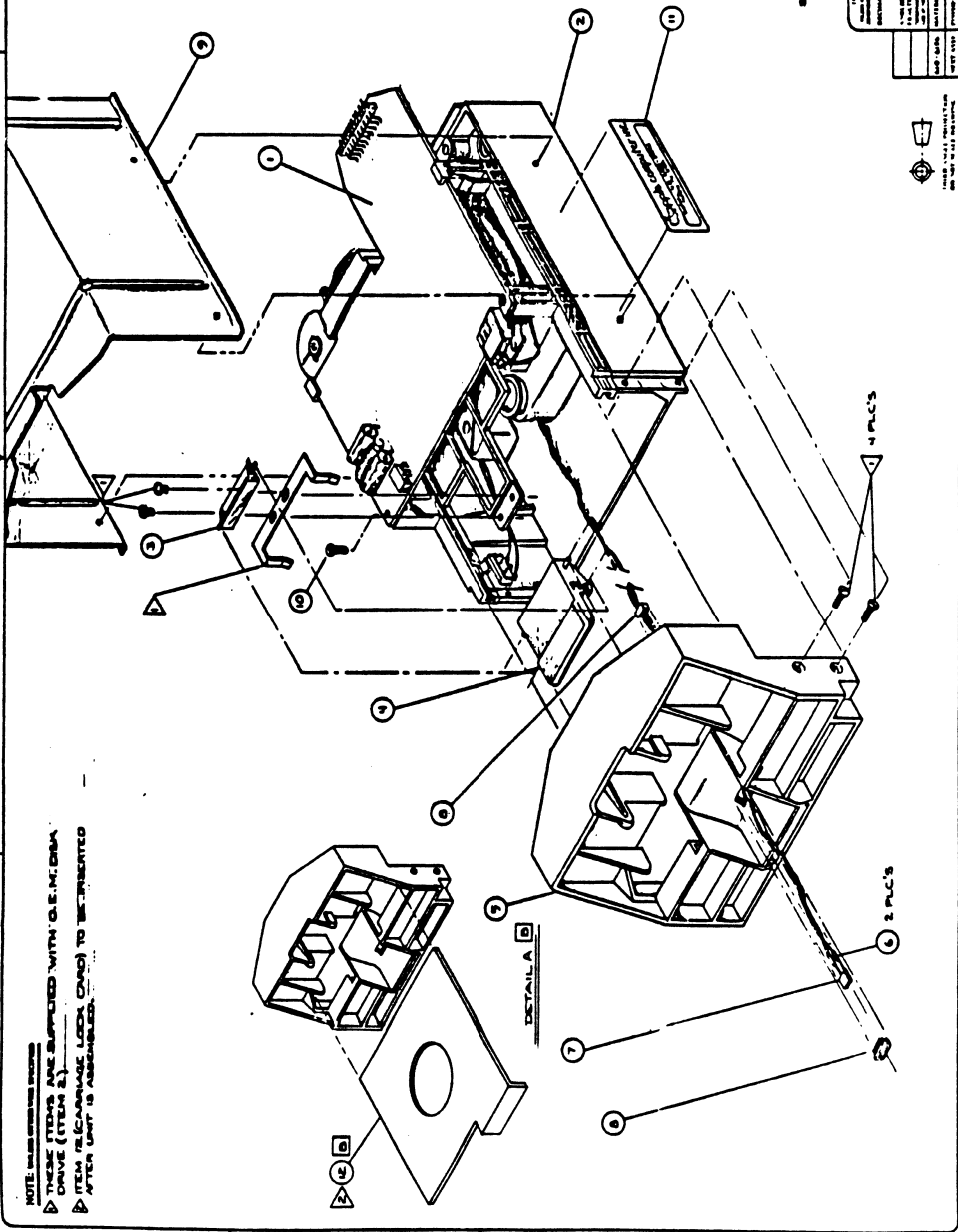
A ULN2003 Darlington Buffer-Inverter, provides a current return path for each of four Stepper Motor windings, Phase A through Phase D. Q1 provides windings with source current when drive is enabled. Since input to each ULN2003 stage is provided by the Disk Conditioning circuit's 9334 Addressable Latch output, Stepper Motor is then under software control.

Stepping in from Track 0 towards Track 34 (towards hub), requires Stepper Windings to be energized in Phase A, B, C, D order. Each phase rotates Cam Follower enough to provide one-half track movement of Head and Carriage assembly. Phase A and C are energized when head is on track, and Phase B and D are between track positions. Once head is positioned to desired track, power is removed from Stepper Motor to reduce power consumption.

Stepping out requires Stepper Motor windings to be energized in Phase D, C, B, A order. When booting, windings are pulsed enough times to guarantee that head is positioned over Track 0.

REV	DATE	DESCRIPTION
1	04/11/82	INITIAL RELEASE
2	05/11/82	REVISED FOR U.S. NOTE 8.1
3	06/11/82	REVISED FOR U.S. NOTE 8.1

ENGINEERING RELEASE
 This revision supersedes all previous versions.
 Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
 The information contained herein is the proprietary property of Apple Computer, Inc. The possessor agrees to the following:
 (i) To maintain this document in confidence.
 (ii) Not to reproduce or copy it.
 (iii) Not to reveal or publish it in whole or part.



SEE SEPARATE BILL OF MATERIALS 650-3201-B

ITEM	QTY	DESCRIPTION
1	1	DISK II ANALOG BOARD
2	1	DISK II ANALOG BOARD
3	1	DISK II ANALOG BOARD
4	1	DISK II ANALOG BOARD
5	1	DISK II ANALOG BOARD
6	1	DISK II ANALOG BOARD
7	1	DISK II ANALOG BOARD
8	1	DISK II ANALOG BOARD
9	1	DISK II ANALOG BOARD
10	1	DISK II ANALOG BOARD
11	1	DISK II ANALOG BOARD

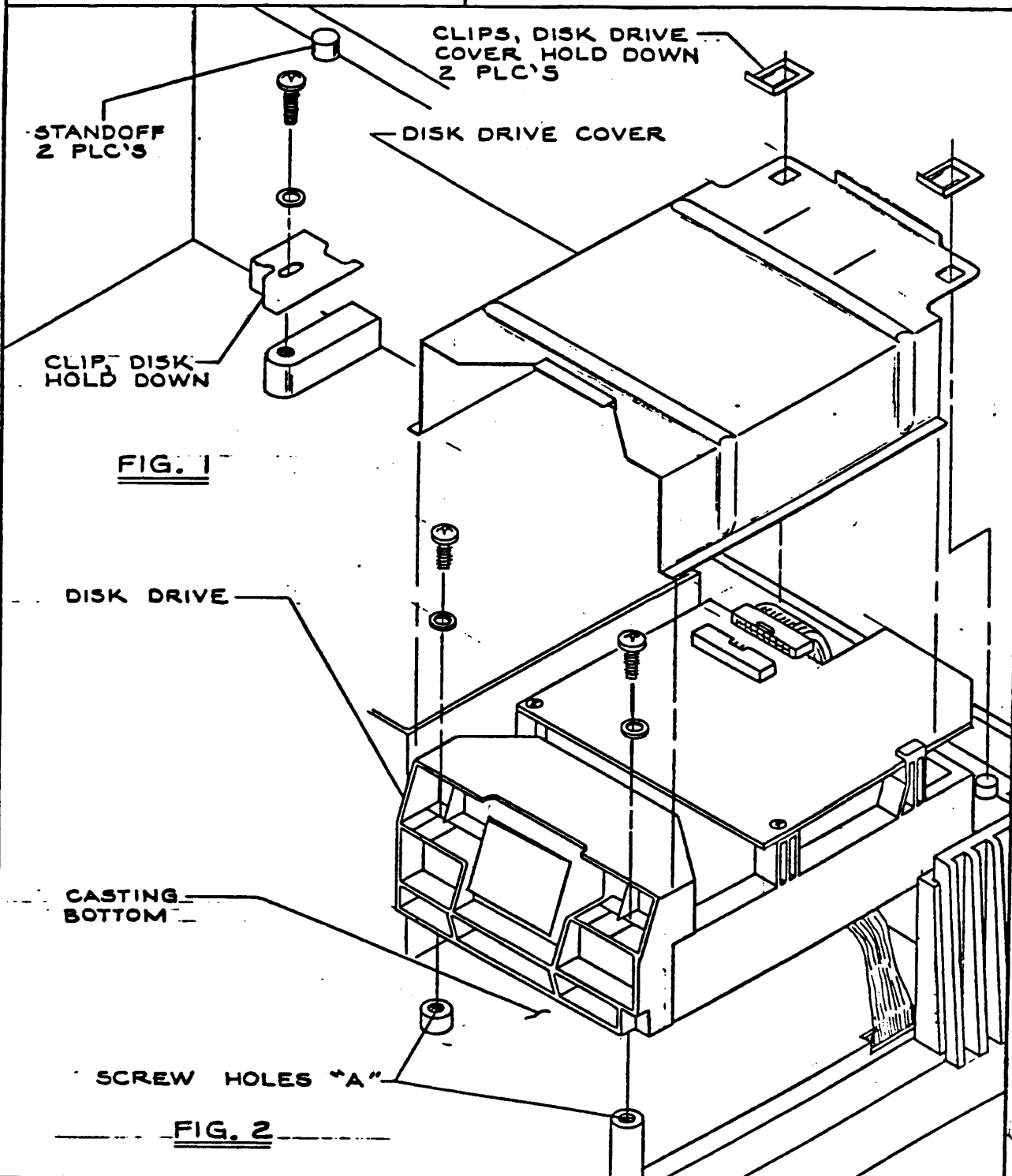
Apple Computer Inc.
 1000 AVENUE AVENUE
 CUPERTINO, CALIF. 95014
 (415) 931-2000
 650-3201-B
 REV 1.00 1

12.8

apple computer inc.

DOCUMENT NO. 064-0156

PAGE 5 OF 14



DRAWN BY
LeTetter

OPERATION NO.
010

TITLE
INSTALL DISK DRIVE

12.9

APPLE III

DISK ENABLE(S)

1) TO ENABLE THE INTERNAL DRIVE 1, THE FOLLOWING CODE SHOULD BE TYPED IN: $C\phi EA C\phi PIN \#10 (L13)$

2) TO ENABLE THE EXT. DRIVES (1,2,OR3), THE FOLLOWING CODE MUST BE TYPED IN:

$C\phi EB C\phi PIN \#10 (L13)$

3) THE FOLLOWING TRUTH TABLE WILL EXPLAIN HOW DRIVES (INT AND EXT) ϕ , 1, 2 AND 3 ARE ENABLED

* $\phi \equiv$ INT. DRIVE

DRIVE*	CODES TO TYPE IN	ENABLE	A1	A ϕ	Z ϕ	Z1	Z2	Z3	FROM J11 (LS)
		1	Don't CARE	Don't CARE	1	1	1	1	
1 I	ϕ C $\phi D4, C\phi D2, C\phi D\phi$	ϕ	ϕ	ϕ	ϕ	1	1	1	*
1 E	1 C $\phi D2, C\phi D1$	ϕ	ϕ	1	1	ϕ	1	1	*
2 E	2 C $\phi D3, C\phi D\phi$	ϕ	1	ϕ	1	1	ϕ	1	PIN *6 J11 (LS139)
3 E	3 C $\phi D3, C\phi D1$	ϕ	1	1	1	1	1	ϕ	PIN *7 J11 (LS139)

* \equiv COINCIDENTAL LOGIC LEVELS.

4) $C\phi E9$ WILL ENABLE J11 (LS139), HENCE THE APPROPRIATE DRIVE WILL BE ENABLED.

EXAMPLE:

LETS ENABLE DRIVE #2

A) TYPE IN THE CODE : $C\phi EB$

B) TYPE IN THE CODE : $C\phi D3$ AND $C\phi D\phi$

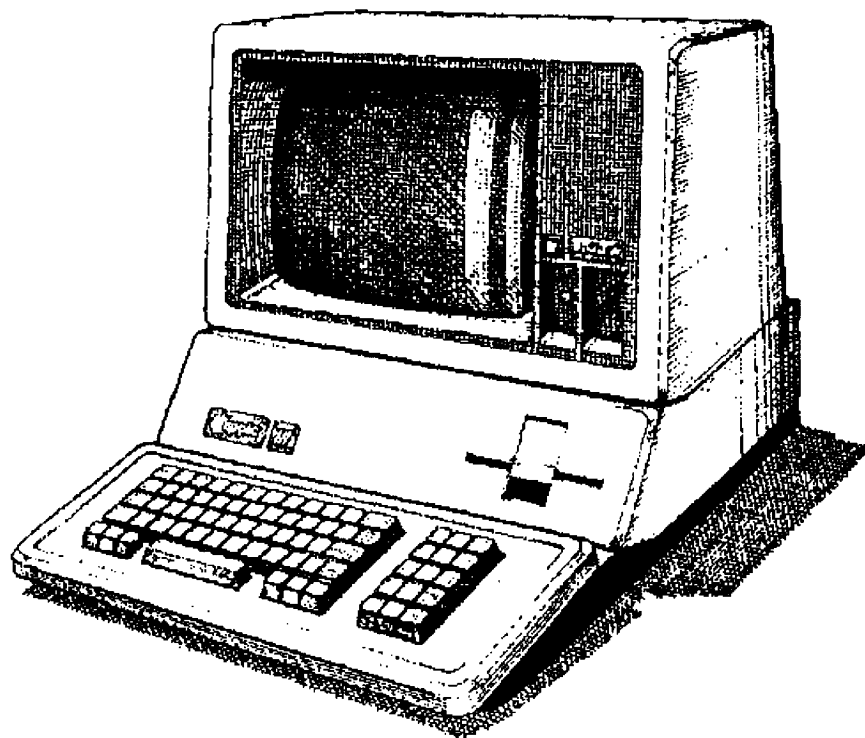
C) TYPE IN THE CODE : $C\phi E9$ NOW DRIVE #2 IS ON

D) TYPE IN THE CODE : $C\phi E8$ NOW DRIVE #2 IS OFF. ²⁸¹ S. WORTH



Apple /// Computer Information

Apple /// Service Reference Manual



Section II of II • Servicing Information

Chapter 13 • Testing and Troubleshooting

Written by Apple Computer • 1982



Title: Apple /// Final Test

Purpose: This test is for the assembled Apple /// and for testing Apple /// modules.

A. Equipment:

1. B/W Monitor with cable
2. Color Monitor, Sup'r'mod II, and adaptor (make your own adaptor)
3. (3 cables) External Disk. Paddle Port, External Speaker Cables
4. External Speaker Test Box
5. RS-232 Test Adaptor
6. Printer test card
7. (4 each) Interrupt test cards
8. Apple /// Test Diskette (889-0009 rev R)

B. Equipment Setup for Part 1 of test: (Note: Unless noted otherwise, ALWAYS make sure that the power is OFF, before connecting or disconnecting ANYTHING from the Apple ///.)

1. Connect all of the power cords to a suitable AC outlet.
Note: Make sure that all power switches are in the OFF position before plugging in any equipment.
2. Connect the B/W monitor cable to the RCA video output jack.
3. Connect the Color Monitor/Sup'r'mod II to the DB-15 jack.
4. Connect the joystick as follows:
 - a. Connect the paddle port cable connector to the external disk drive socket.
 - b. Connect the 2 DB-9 plug to the 2 DB-9 sockets. (The one with the shortest cable connects to the socket nearest the external drive connector, Port A. The other connects to Port B.)
5. Plug one Interrupt test card into each of the four slots on the Apple /// logic board.
6. Plug the RS-232 Test Adaptor into the DB-25 connector on the logic board. (P/N 890-0130)
7. Insert the Apple /// Test Diskette into the drive and close the drive door.

C. Test Procedure: (for part 1 of the test.)
Follow the test procedure described in this section. The test should run as described. If there is a failure, some of the tests will automatically proceed to the next test, while others will require the operator to press certain keys, to tell the system what has failed. Proceed through all of the tests. If the system will not proceed through all of the tests, indicate on the RRT which test failed. Reject and repair any unit which does not perform as described in this procedure.



*TEST DISKETTES HAVE BEEN KNOWN TO CRASH - KEEP AN EXTRA COPY ON HAND

1. Power On. Turn the power supply switch on. The unit will perform a self test first. If the self test passes, the disk will boot. If there is a failure in the self test, or the disk drive does not boot, record the failure on the RRT and send the logic board for rework. (ALWAYS TURN THE POWER OFF BEFORE DISCONNECTING ANYTHING FROM THE LOGIC BOARD.) When the disk drive boots, you will hear an audio signal of three beeps, followed by two beeps. You will then see a menu. Press the 1 key on the keyboard to run Automatic Test 1.

2. Interrupt test. The interrupt test will run automatically. If you see any of the following error messages, attempt to repair the A3 system and re-run the test. Write the failure down on the RRT.

ERROR MESSAGE. SUGGESTED ACTION TO TAKE
unable to set or clear D.xxx 6522 at H-10 (U 73) *
unable to set or clear E.xxx 6522 at G-10 (U 97) *
(anything) from SLOT Xcheck slot for bad connector

* locations will be as follows for the "NEW" logic board:

D.xxx6522 at B-6
E.xxx6522 at B-4

3. Video Test. The video tests will be loaded and run next. At the beginning of each test the screen will briefly display the name of the test being performed next and which keys to press, depending on the results. For reference, the following table lists the keys used for all of the video tests:

Space bar Test passes
Return key Test fails
Escape key Abort video test (QUIT)
Left arrow key Retry the test

Except for the text mode test, each of the tests will display the same pattern. A picture of Winston Churchill will appear in the upper right corner. The lower half of the screen will display the following message:

If you can read this, and
the test patterns above
are clear, press space bar.
Otherwise, press return.

- a. HIRES MODE PAGE 1 - B & W pattern
- b. HIRES MODE PAGE 2 - B & W pattern (same as above)
- c. 280 x 192 COLOR HIRES MODE PAGE 1 - Will appear as a negative image. The color monitor will show red and black.



- d. 280 x 192 COLOR HIRES MODE PAGE 2 - Will appear as a green and white (or possibly green and yellow) pattern.
- e. SUPER HIRES MODE PAGE 1 - B & W pattern (same as above)
- f. SUPER HIRES MODE PAGE 2 - B & W pattern (same as above)
- g. AHires TEST PAGE 1 - On this and the following test the screen will be divided into 4 horizontal sections, each one being a different color. The top half of Winston Churchill and the diagonal pattern should be VIOLET. The bottom half of Winston Churchill and the diagonal pattern should be BLUE. The first two lines of the message should be GREEN, and the last two lines of the message should be GOLD or ORANGE.
- h. AHires TEST PAGE 2 - This test should display the same four color bars as the above test.
- i. COLOR BAR 7 GRAY SCALE TEST - will show vertical bars of different colors on the color monitor and bars of varying brightness on the B & W Monitor. The border is blue and the colors are : (from left to right) white, aqua, yellow, green, pink, grey, orange, brown, light blue, medium blue, grey, dark green, light purple, dark blue, magenta, and black. These colors correspond to white darkening to black on the Black & White monitor. (IMPORTANT. Make sure that there are sixteen (16) different shades on the Black & White monitor.)
- j. Apple II TEXT MODE PAGE 1 - The screen will display the following:

```

THE QUICK BROWN FOX JUMPS OVER LAZY DOGS

abcdefghijklmnopqrstuvwxyz 0123456789
                          (inverse)
    
```

. . .

(flashing)

- k. APPLE II TEXT MODE PAGE 2 - The screen will show the following:

```

22222222222222222222
22222222222222222222
22222222222222222222
                222
                222
                222
22222222222222222222
22222222222222222222
22222222222222222222
222
222
222
22222222222222222222
    
```



22222222222222222222
22222222222222222222

- l. APPLE /// 40 COLUMN TEXT MODE TEST - The screen will be filled with blocks of colors with the name of each color in each block.
- m. APPLE /// 80 COLUMN TEXT MODE TEST - The screen will contain characters that are smaller than before. There will be 80 characters to a line. The characters may not appear clear on the color monitor, and this is OK. It is mainly important that they are clear on the B & W monitor. The first line of the display should read:

THIS LINE OF TEXT IS EXACTLY 80 CHARACTERS
LONG AND USES THE ENTIRE SCREEN WIDTH

4. Keyboard Test. This test will load and display a pattern on the screen.
 - a. Main Keyboard. Press the Left shift key and while holding it down press the 2 key. Press the Right shift key and while holding it down press the = key. Press the ctrl key and while holding it down press the A key. Press all of the remaining keys on the MAIN key board. Each time a key is pressed its character should disappear from the screen. Press the space bar last.
 - b. Numeric Keypad. A new pattern should appear on the screen which corresponds to the numeric keypad. This test should perform the same as the main keyboard test.
 - c. Special Function Keys.
 1. Press the Alpha-Lock key ONCE. It should lock into its new position.
 2. Press the space bar and hold it down.
 3. While still holding the space bar, press and hold both apple keys at the same time.
 4. Release all of the keys at the same time.
 5. Press the solid apple key and hold it down.
 6. While still holding down the solid apple key, press the space bar and hold it down.
 7. Release all of the keys at the same time.
 - d. Keyboard Interrupt test. Press any key on the keyboard except the alpha-lock, shift, control, or either of the apple keys to perform this test.
5. Clock/Calendar Test: This test is available for testing the clock/calendar when it becomes incorporated into the system.
6. Serial Port Test: This test will also load and run automatically. If it fails, replace the ACIA chip (6551) and run the test again. If it still fails, write ACIA on the RRT and repair the main



logic board.

7. Joystick Port test: This test will run automatically. If any failures should occur, write the failure message on the RRT.
8. Test Results: The screen should show the following results:

TEST RESULTS

A. INTERRUPT	(PASSED)
B. VIDEO	(PASSED)
C. KEYBOARD	(PASSED)
D. CALENDAR/CLOCK	(NOT TESTED)
E. ACIA PORT	(PASSED)
F. PADDLE PORTS	(PASSED)
G. RAM	(NOT TESTED)
H. PRINTER PORT	(NOT TESTED)
I. DISK	(NOT TESTED)
J. SOUND	(NOT TESTED)
K. ROM	(NOT TESTED)
1. AUTOMATIC TEST 1	
2. AUTOMATIC TEST 2	
ESC ABORT TESTING	

Tests A through F should always show passed, (except for test D) and tests G through K and test D should always show not tested. If any of tests A through F show failed, mark the RRT with the test that failed. If all of tests A through F show passed, proceed with part 2 of the Final test.

D. Equipment Setup for part 2 of test:

1. TURN THE POWER OFF!
2. Remove the following items from the logic board:
 - a. The joystick cables
 - b. The four interrupt test cards.
 - c. The RS-232 test adaptor.
 - d. The DB-15 video connector
3. Plug the Printer Test Card into slot 1, and connect the printer cable to the DB-9 connector nearest the disk drive sockets and the other cable to the external disk drive socket. Connect the external speaker plug to the 2-pin connector on printer test card.

E. Test Procedure: (for part 2 of the test.)

Please follow the test procedure described in this section. The test should run as described. Reject any assembly or unit which does not perform as described in this procedure. Complete all of the tests if possible. If the system will not perform any test, indicate on the RRT which test failed and diagnose, repair, and retest.

1. POWER ON. Turn the power on and the unit should perform



a self-test and boot just as it did earlier. If there is a failure in the self-test or the drive does not boot, record the failure on the RRT and repair. ALWAYS TURN THE POWER OFF BEFORE DISCONNECTING ANYTHING FROM THE LOGIC BOARD. After the disk boots, a menu will appear on the screen. Press the 2 key to run automatic test 2.

2. Ram Address Test. This test will load and run automatically. The test results will depend on the amount of ram in the memory board. If the memory board is a 256K board the results should say "RAM MAP GOOD FOR A 256K SYSTEM". If the correct message appears, press the space bar, otherwise press the return key. Faulty RAM chips are reported in a message that identifies the board location of the chip in error.

Note: If a fault is discovered while testing the RAM on the 12 volt board, disregard the chip referred to in the error message and run the Final Test Revision 14. Revision 14 will correctly identify the chip in error. Revision R reports bad chip locations as defined on the 5 volt board and these messages are innaccurate for the older board.

If the space bar is pressed the system will perform a test on all of the ram in the system and report any failures. For this reason it is very important for you to have made the correct decision for the ram map address test above. If the system is a 256K system and the ram map says good for a 128K system and you press the space bar, only half of the ram will be tested and you may incorrectly PASS a system which FAILED.

3. Printer Port Test. This test will run automatically.
4. Disk Controller Test. This test will run automatically.
5. Sound Test.
 - a. C030 SOUND TEST - The speaker will beep on and off.
 - b. C040 SOUND TEST - The speaker will beep on and off as before but at a different pitch. Press the space bar if you hear the sound, and return if you do not.
 - c. Connect the external speaker cable from the printer port test card to the external speaker jack on the back of the Apple /// and press the return key.
 - d. FFEO SOUND TEST - The sound from the speaker should start quietly and grow gradually louder. It should then repeat. Press the space bar if it performs as described here, and press the return key if it does anything else. The sound should be coming from the EXTERNAL speaker.
 - e. Disconnect the external speaker cable from the logic board and press return. (NOTE: These are the ONLY times that you can connect or disconnect anything from the Apple /// with the power still on, and the ONLY thing that can be connected or



- disconnected is the external speaker cable.)
- f. The same sound as the previous test should be heard. The sound should come from the INTERNAL speaker again.
6. Rom Test. This test will load and run automatically.
 7. Test Results: The screen should show the following results:

TEST RESULTS

A. INTERRUPT	(NOT TESTED)
B. VIDEO	(NOT TESTED)
C. KEYBOARD	(NOT TESTED)
D. CALENDAR/CLOCK	(NOT TESTED)
E. ACIA PORT	(NOT TESTED)
F. PADDLE PORTS	(NOT TESTED)
G. RAM	(PASSED)
H. PRINTER PORT	(PASSED)
I. DISK	(PASSED)
J. SOUND	(PASSED)
K. ROM	(PASSED)

1. AUTOMATIC TEST 1
2. AUTOMATIC TEST 2

ESC ABORT TESTING

Tests A through F should always show not tested, and tests G through K should always show passed. If any of the tests G through K show failed, record which test failed on the RRT and repair the module under test.

8. Additional keyboard tests:
 - a. Press any key and hold it down. The key should automatically repeat.
 - b. While still holding the same key down, press the Apple key nearest the space bar and the repeating key should repeat at a faster rate. (Approximately twice the speed)
 - c. Press the right arrow key and the cursor dot should move to the right. Press the key harder and it should move twice as fast.
 - d. Repeat the same test with the left arrow and down arrow keys. They should behave in the same manner described for the right arrow key except that they will of course move left and down, respectively.
 - e. Press and hold the ctrl key with some finger on your left hand and then press and hold the Apple Key next to the alpha lock with your left thumb. Use your other hand to press the Reset key. The system should respond with a right



pointing arrow and a blinking line cursor.

- f. If any of these keyboard tests do not perform exactly as described here, record the failure on the reject tag, and send the unit for rework.
- F. If the logic board passes all of the tests as described above, turn the power off and complete the RRT. If any of the tests failed, record the appropriate information on the RRT and repair. Retest after repair.
- G. If you have a system or module that passes all these tests but suspect it to have a failure run other software on the unit/modules. Examples: Business Basic, AII Emulation, A /// Dealer Diagnostic.

PRE-RELEASE VERSION

16 SECTOR DISK III FINAL TEST (1000T)

DESCRIPTION

1000T is a general purpose internal disk exercizer. It performs 1000 reads of randomly selected tracks on the disk. It is to be used as a diagnostic tool and not as a qualification/acceptance test. The rest of this document provides a short description of how to interpret the displayed results.

When you first boot this diskette you will observe:

"*** 16 SECTOR DISK III FINAL TEST ***"

This will indicate that the test booted up correctly with no problems. You will then observe the following:

TRACK ERRORS							

3	(0)	0	*	19	(0)	0	
4	(0)	0	*	20	(0)	0	
5	(0)	0	*	21	(0)	0	
6	(0)	0	*	22	(0)	0	
7	(0)	0	*	23	(0)	0	
8	(0)	0	*	24	(0)	0	
9	(0)	0	*	25	(0)	0	
10	(0)	0	*	26	(0)	0	
11	(0)	0	*	27	(0)	0	
12	(0)	0	*	28	(0)	0	
13	(0)	0	*	29	(0)	0	
14	(0)	0	*	30	(0)	0	
15	(0)	0	*	31	(0)	0	
16	(0)	0	*	32	(0)	0	
17	(0)	0	*	33	(0)	0	
18	(0)	0	*	34	(0)	0	

TOTAL:	SEEK:	DATA:
TIME:	ADDR:	AVER:

WHAT DOES IT ALL MEAN?

The first column, numbered 3 to 18, and the column with numbers going from 19 to 34 represent track numbers. The column in brackets represents the number

of seek occurrences that occur for each track. The column that has zeros^e is the number of errors that were encountered for each track. You will observe that each time a track is read, it is shown in inverse, the number of occurrences is incremented. If any seek, address, or data errors are found the number of errors are displayed.

A summary of the disk test is displayed at the very bottom of the monitor screen. The following are definitions as to what the messages mean.

- o TOTAL: -- the total number of errors for all tracks
- o TIME: -- the number of track seeks performed for all tracks
- o SEEK: -- the total number of track seek errors observed for all tracks
- o ADDR: -- the total number of address errors observed for all tracks
- o DATA: -- the total number of data errors observed for all tracks
- o AVGE: -- the number of track seeks divided by the total number of errors observed for all tracks

NOTE: PLEASE MAKE BACKUP DISKETTES OF THIS 1000T DISKETTE

WHAT CAN THIS TOOL TELL ME?

This tool is useful for getting a good idea as to the performance of the A/// internal disk drive. Based on the results, 1000T can give you an indication of electrical and mechanical problems. Examples: Errors within a small range of tracks could indicate cam or rail problems. Multiple data errors could indicate head wear. Address/seek errors could indicate a poor motor control board. These examples are not absolute nor do they exhaust all possibilities. This tool is also very useful for debugging intermittent disk problems.

WARNING: Do not rely solely on this diskette as a pass/fail indicator. If you find very many errors, run other Disk tests such as the DSPEED and Disk Alignment Aide.

Catastrophic errors are easily to find - your monitor screen will display "FAILS TEST". Other than catastrophic errors your monitor screen will display "PASSED TEST" at the end of 1000 passes(seeks).

HAPPY HUNTING!!!

PRE-RELEASE
VERSION

13.10



APPLE /// TROUBLESHOOTING

The following flowchart is a guide for troubleshooting the Apple ///. You will carry out various troubleshooting steps based on symptoms that may occur when first booting up or that may be found by running the Dealer Service Diagnostics. Start with the instruction in Box 1 of the flowchart. Then follow the operation of the Apple /// through the flowchart until you reach one of the lettered boxes (A through Q). Each lettered box has a list of numbers in it; each number corresponds with one of the troubleshooting steps listed on the following page. The order of the troubleshooting steps in each box is based on two rules:

- 1) Check out the more likely causes of the problem before the less likely causes.
- 2) Make the checks that can be done quickly and easily before those that take more time and energy.

Rule 1 is broken only when rule 2 applies.

Once you have produced the problem symptom on the Apple ///, the first thing you should do before trying any of the numbered steps below is:

- a) Power OFF.
- b) Check to make sure all connecting cables are properly hooked up.
- c) Check all boards to make sure all IC chips are properly seated.
- d) Power ON again to see if the problem still exists.

ALWAYS POWER OFF BEFORE PERFORMING ANY OF THE STEPS BELOW.
CARRY OUT THE DESIGNATED STEP.
THEN POWER ON AGAIN TO SEE IF THE PROBLEM HAS BEEN ELIMINATED.

Each swap step listed below involves exchanging a known good part from your spares kit with the questionable part from the Apple ///. When swapping, first just connect the cable(s) to the new module so you can see if the swap fixes things or not. Don't fully install the new module and screw everything down—if the new module doesn't solve the problem you'll just have to take it out again.



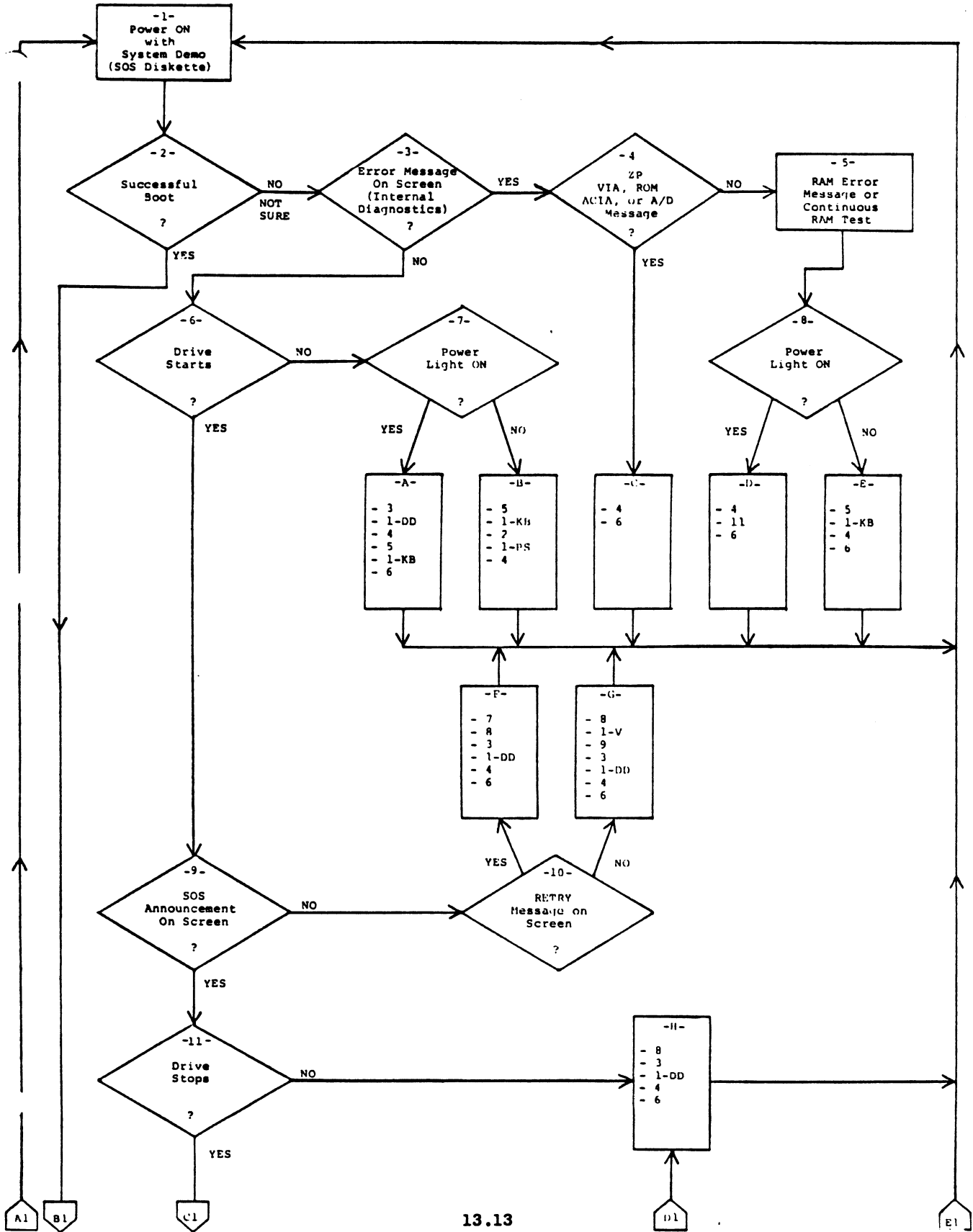
HERE ARE THE STEPS REFERRED TO IN THE BOXES ON THE DIAGNOSTIC FLOWCHART:

- 1) Swap the appropriate connecting cable.
 - V = Video cable (if available)
 - PS = Power Supply cable
 - DD = Disk Drive cable
 - KB = Keyboard cable

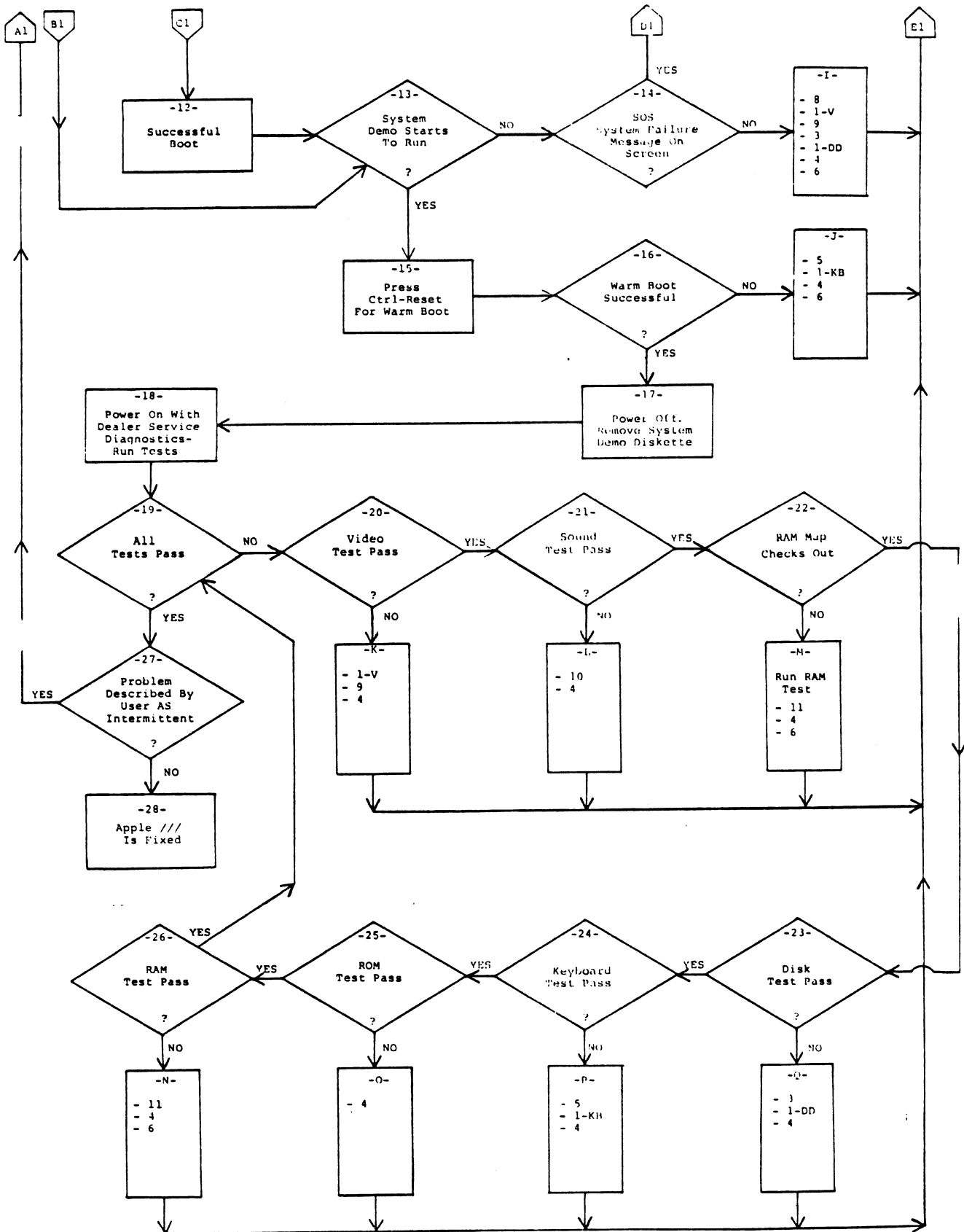
(The keyboard and disk drive cables are identical to each other. Your Spares Kit may only list the DD cable, but you can use it whenever you need to swap the KB cable.)
- 2) Swap the power supply.
 - a) Check the power supply fuse first; swap it if it's burned out.
- 3) Swap the drive.

If the drive proves to be the problem, take the problem drive and further isolate the defective module down to the analog card or mechanical assembly:

 - a) Swap the analog card.
 - b) Swap the mechanical assembly.
- 4) Swap the main logic board.
- 5) Swap the keyboard.
- 6) Swap the RAM memory board. (You may have to reload the new board with the RAM from the original board.)
- 7) Try booting again.
- 8) Try booting a different SOS boot diskette.
- 9) Swap the video monitor (if you have a spare available).
- 10) Swap the speaker (if you have a spare available).
- 11) Swap (or add) the designated RAM IC chips. (Consult the chip map in the Running Diagnostics Job Aid.)

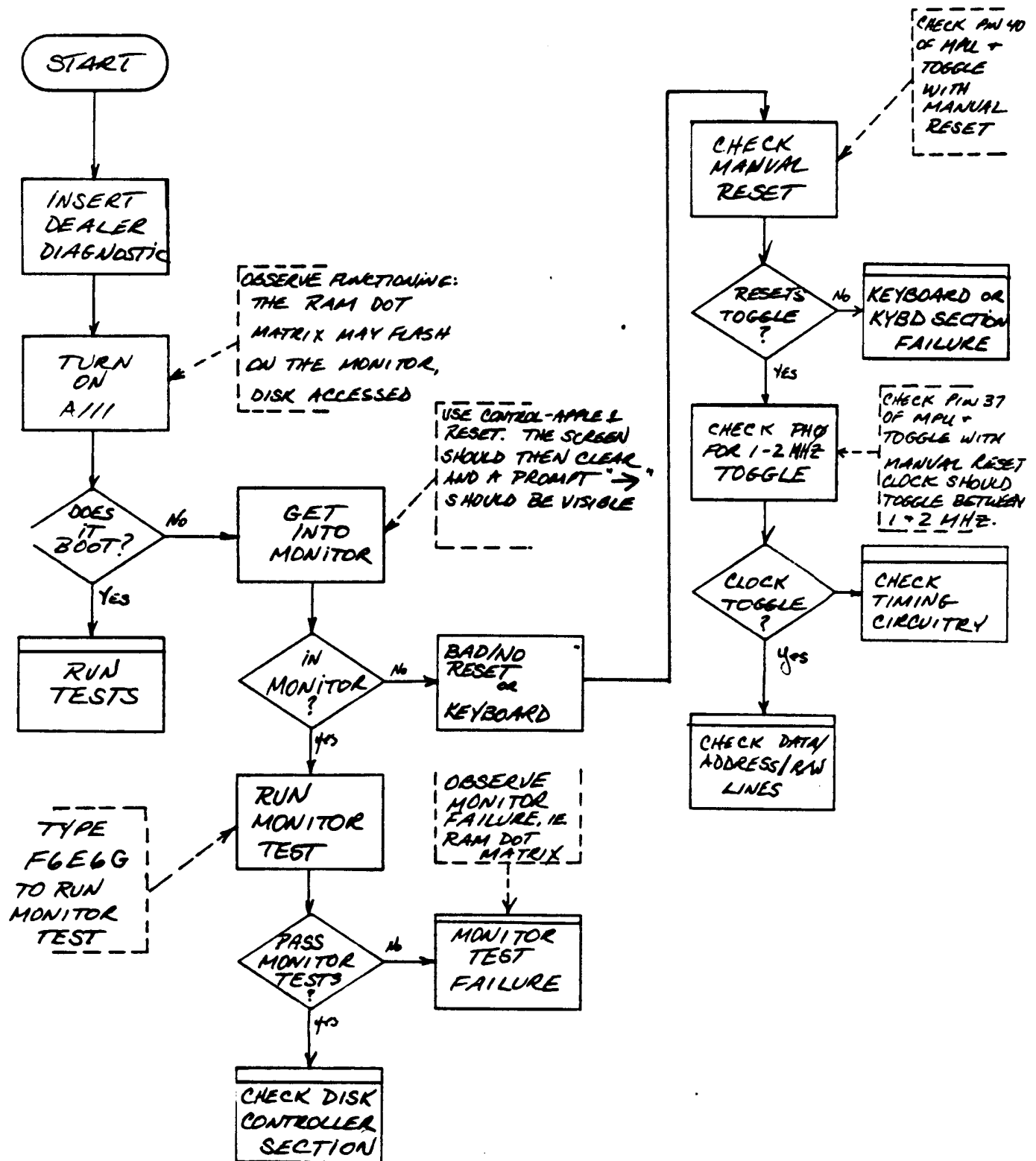


13.13



13.14

A/// NO BOOT FLOW DIAGRAM





5V MEMORY BOARD RAM TROUBLESHOOTING PROCEDURE

AUG 1982

13.16



5V VOLT MEMORY BOARD RAM TROUBLESHOOTING PROCEDURE

To start with, be sure that your problem is caused by the 5V Memory Card. Some problems on the motherboard of the Apple /// will cause the symptoms similar to those caused by a bad 5V Memory Card. To check, replace the Memory Card with a known good one and check to see if the symptoms have disappeared.

- 1) If the problem has been isolated to the Memory Card, reconnect the bad 5V Memory Card in the Apple /// system under test and try to boot the Apple /// Confidence Diskette Version 1.1.
- 2) If it boots, select the memory test.
- 3) Relate your system's symptoms to the symptoms on the Apple /// 5V Memory Card Troubleshooting Chart.
- 4) The corrective actions are listed in the order of most probable cause; therefore perform the corrective action in the order presented.
- 5) If all of the possibilities have been exhausted and the problem still exists, replace the bad 5V Memory Card with a good one and send the bad one back to a Level II repair center.

Note: This interim RAM troubleshooting procedure is to be used with the Confidence Diskette Version 1.1. The memory test, in the current Confidence Diskette, does not test each and every memory location in RAM. This procedure will be superseded by the next version of the A/// Dealer Diagnostic.



Apple /// 5V Memory Card Troubleshooting Chart

Symptom	Recommended Action
Black Screen on Monitor; drive does not boot.	Replace RAM chips B10 - B17 one at a time; Replace the non-RAM chips at locations D2 and E2.
Monitor Screen contains garbage; Drive may try to boot.	Replace RAM chips B2 - B9 one at a time; Replace the non-RAM chips at locations D2 and E2.
Confidence Program loads into memory, displays menu, but will not run.	Replace the non-RAM chips at locations D2 and E2; Replace RAM chips B2 - B17 one at a time.
Memory test runs; displays RAM error message at the bottom of the screen.	Decode the message using the procedure on Page 4. Replace the failed RAM.
Memory test runs; sections of the memory map are missing.	Determine which section on the Memory Card contains the failed RAM using the procedure on page 8. Replace the RAM chips in that section one at a time; Replace non-RAM chips at locations D2 and E2.

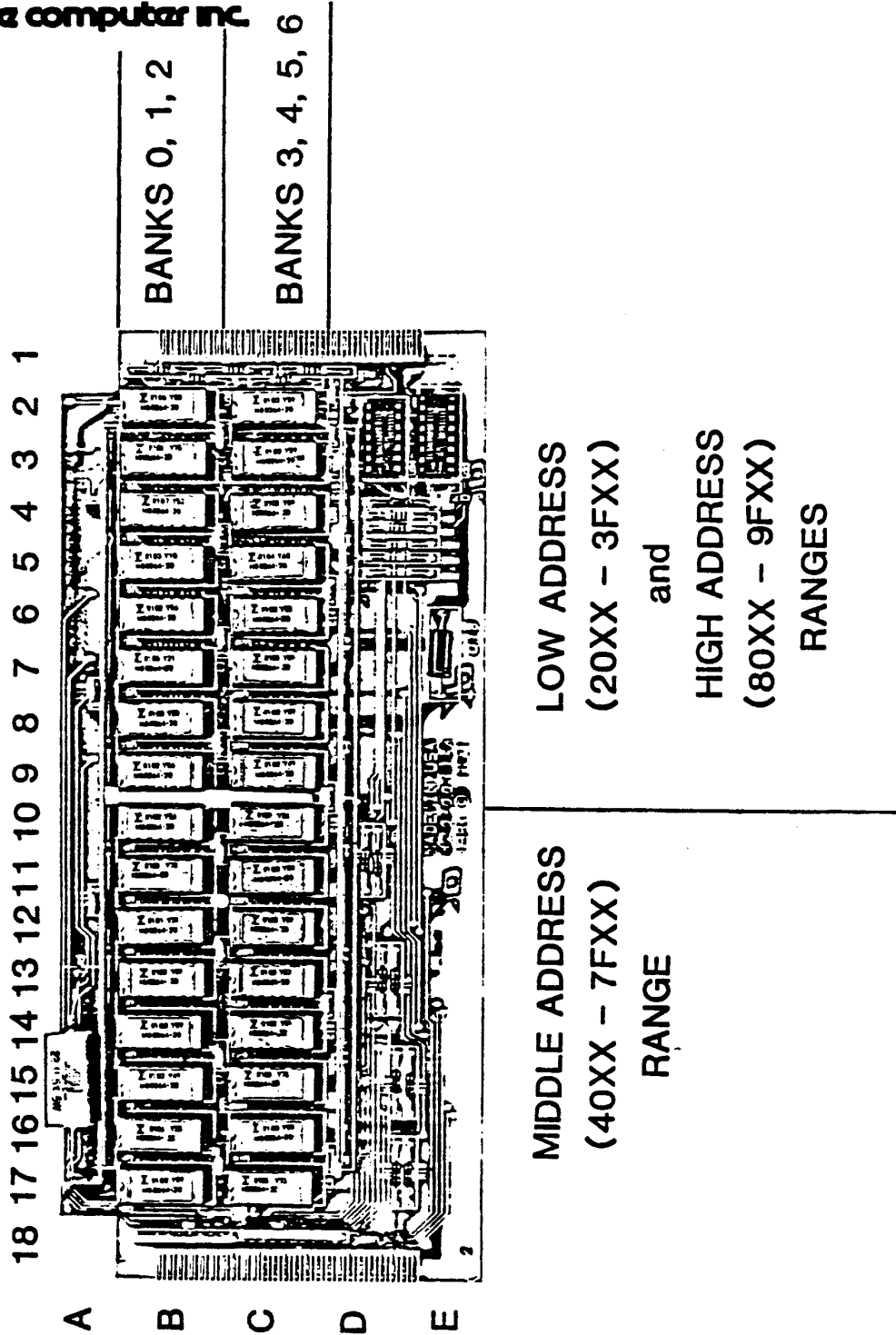


FIGURE 1

13.19



TRANSLATING ERROR MESSAGES INTO PHYSICAL RAM LOCATIONS

When you get an error message from the RAM test, you must translate it to determine which chip caused the failure. This procedure will show you how to do that. For example, suppose we get the error message:

BNK 83, ADR 20XX, EXP DF, GOT 5F

That means in Bank 83, Address Range 20XX, we expected DF but got 5F.

Now, how do you translate that into what to do?

- 1) To find out which row the failed RAM chip is in, disregard the 8, and look at the second number, in this case 3. If the second number is 0, 1, or 2, the bank is in row B. If the second number is 3, 4, 5, or 6, the bank is in row C. (See Figure 1). In our example, the bank was BNK 83, so we know it is in row C.
- 2) Now the meaning of the address. There are three address ranges, low, middle, and high. Low and high are in columns 2-9, and middle is in columns 10-17. Look at Figure 1 for the specific address ranges. In our example, the address was 20XX (which is in the low address range), so we see that the trouble is in columns 2-9.
- 3) The problem is now narrowed down to a block of eight chips, the ones located in row C, columns 2 - 9 (positions C2 - C9). To find which of the eight it is, we have to decode the EXP and GOT parts of the message.

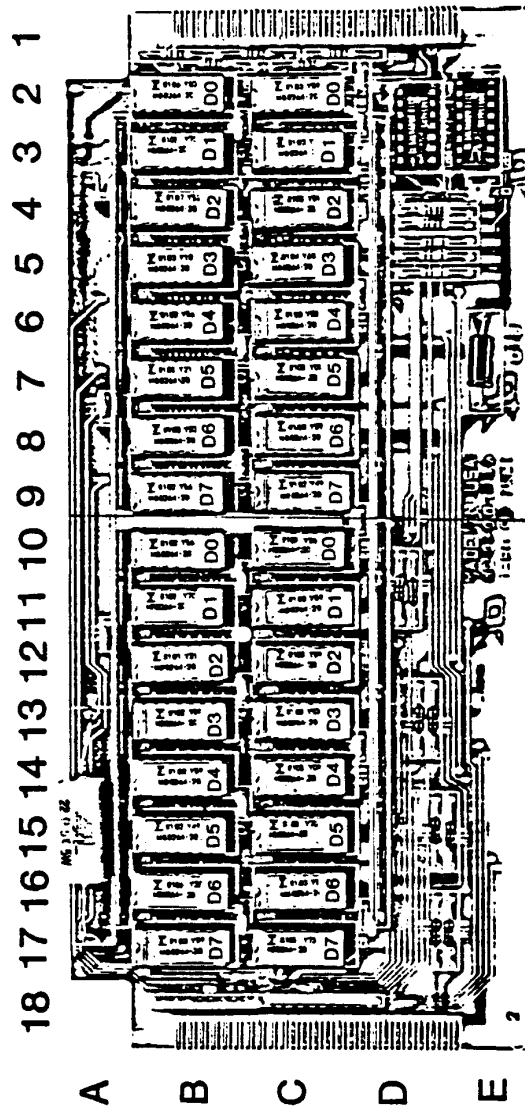


FIGURE 2

18.21



- 4) Translate the two hexadecimal digits from the EXP into binary.

HEX	BINARY	HEX	BINARY	HEX	BINARY	HEX	BINARY
0	= 0000	4	= 0100	8	= 1000	C	= 1100
1	= 0001	5	= 0101	9	= 1001	D	= 1101
2	= 0010	6	= 0110	A	= 1010	E	= 1110
3	= 0011	7	= 0111	B	= 1011	F	= 1111

EXAMPLE (DF): D=1101, F=1111, DF=11011111

- 5) Translate the two hexadecimal digits from the GOT onto binary.

EXAMPLE (5F): 5=0101, F=1111, 5F=01011111

- 6) Determine the binary digit (bit) that is different between the EXP and the GOT. The leftmost bit is D7 and the rightmost bit is D0. In our example the D7 bit is different. This indicates that the chip marked D7 in Figure 2 in the position C9 (remember, we already got it down to C2 - C9) is defective.

```

EXAMPLE:   D       76543210
           EXP    DF=11011111
           GOT    5F=01011111
                -----
                X----- (D7 is different)
    
```

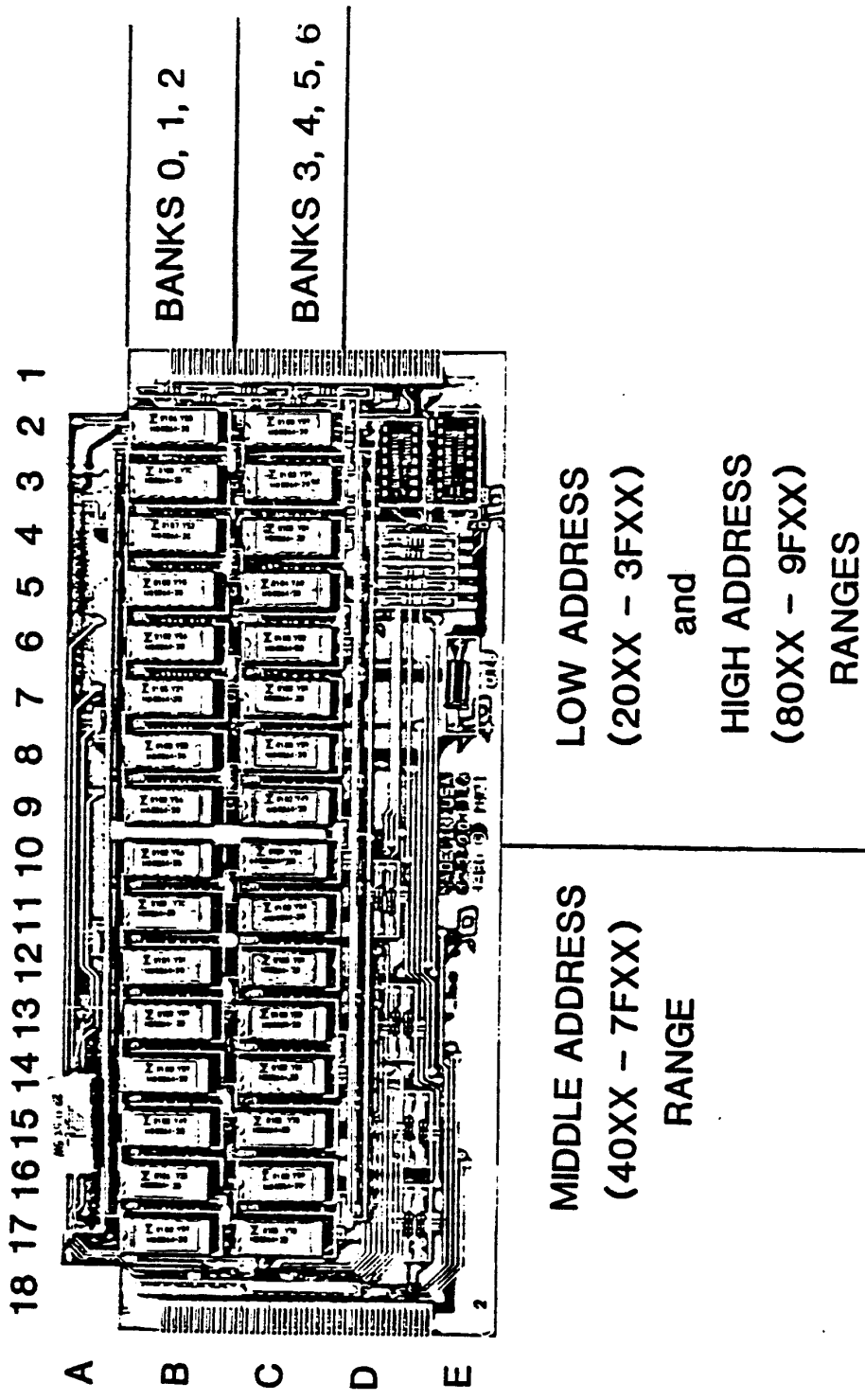


FIGURE 3

13.23



TRANSLATING MISSING SEGMENTS OF THE MEMORY MAP DISPLAY INTO PHYSICAL RAM LOCATIONS

Sometimes the Memory Test does not give an error message, but instead erases a portion of the memory map display, and continues to test the RAM. Figure 4 below is an example of a memory map display with a missing segment.

- 1) First, notice the bank(s) that are missing in Figure 4 (listed down the left side). Correlate the bank(s) to the physical row of memory. In our example, banks 3,4,5 and 6 are missing. This means that the failed RAM is somewhere in row C.
- 2) Determine the address range(s) that are missing (listed across the top.) Translate the address range(s) to the physical section. In our example, address ranges 50XX - 5FXX and 70XX - 7FXX are missing. This falls within the address range 40XX - 7FXX and tells us that the failed RAM is one of the eight which is physically located between C10 - C17 on the memory board.
- 3) Change the chips in that section, one at a time, retrying the RAM test each time. If the problem still remains, replace the old RAM back in the board and try another of the eight.
- 4) If you change all eight chips without fixing the problem, try the two non-RAM chips at location D2 and E2 on the 5V Memory Board.

		ADDRESS RANGES (20XX - 9FXX)								
K	Bank	2	3	4	5	6	7	8	9	9
		0	0	0	0	0	0	0	0	F
64	0	XX								
96	1	XX								
128	2	XX								
160	3	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX
192	4	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX
224	5	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX
256	6	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX	-----	XXXXXXXXXX
Extension (\$8F)		XX								

FIGURE 4

 **apple computer inc.**

13.25



SUMMARY

Translating Error Messages Into Physical RAM Locations

Error Message: BNK 84, ADR 37XX, EXP 40, GOT 48.

1. The physical row is determined by the bank number which is the bank is the last digit of the number in the BNK section. (Bank 4 in our example puts the problem in row C.)
2. Which half of the row is determined by the address range. Our example puts the problem in the right half of the Memory board (Columns 2-9).
3. The location of the failed RAM chip within the half row is determined by decoding the EXP and GOT messages.

HEX	BINARY	HEX	BINARY	HEX	BINARY	HEX	BINARY
0	= 0000	4	= 0100	8	= 1000	C	= 1100
1	= 0001	5	= 0101	9	= 1001	D	= 1101
2	= 0010	6	= 0110	A	= 1010	E	= 1110
3	= 0011	7	= 0111	B	= 1011	F	= 1111

EXP (40): 4=0100, 0=0000, 40=01000000

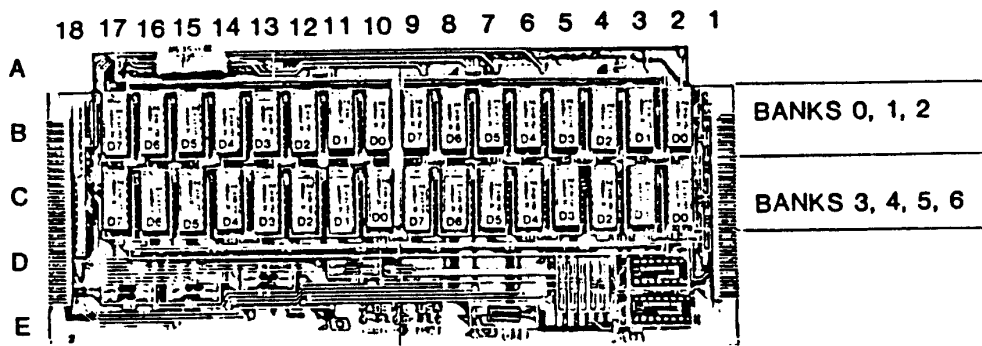
GOT (48): 4=0100, 8=1000, 48=01001000

```

76543210
40: 01000000
48: 01001000
-----

```

----X--- (D3 is different, so the RAM chip at C5 is bad in our example.)



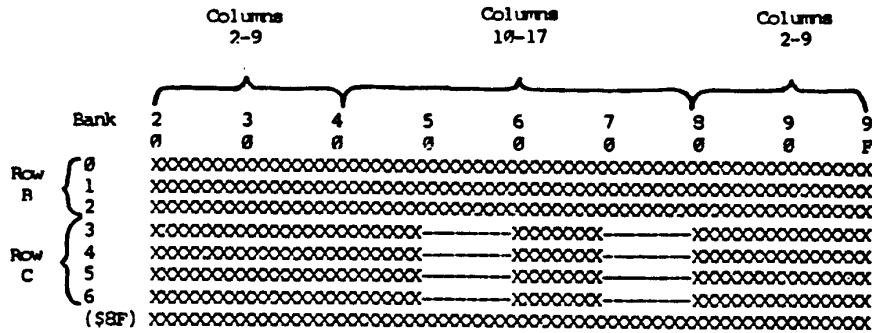
MIDDLE ADDRESS
(40XX - 7FXX)
RANGE

LOW ADDRESS
(20XX - 3FXX)
and
HIGH ADDRESS
(80XX - 9FXX)
RANGES



Translating Missing Sections on the Memory Map Display into Physical RAM Locations

1. Determine which row the failed RAM is located from the Memory Map display.
2. Determine the half row the failed RAM is located from the Memory Map display.
3. Replace the eight RAM chips in that half row, one at a time.
4. Replace the non-RAM chips D2 and E2.





USING THE APPLE /// FLOWCHART

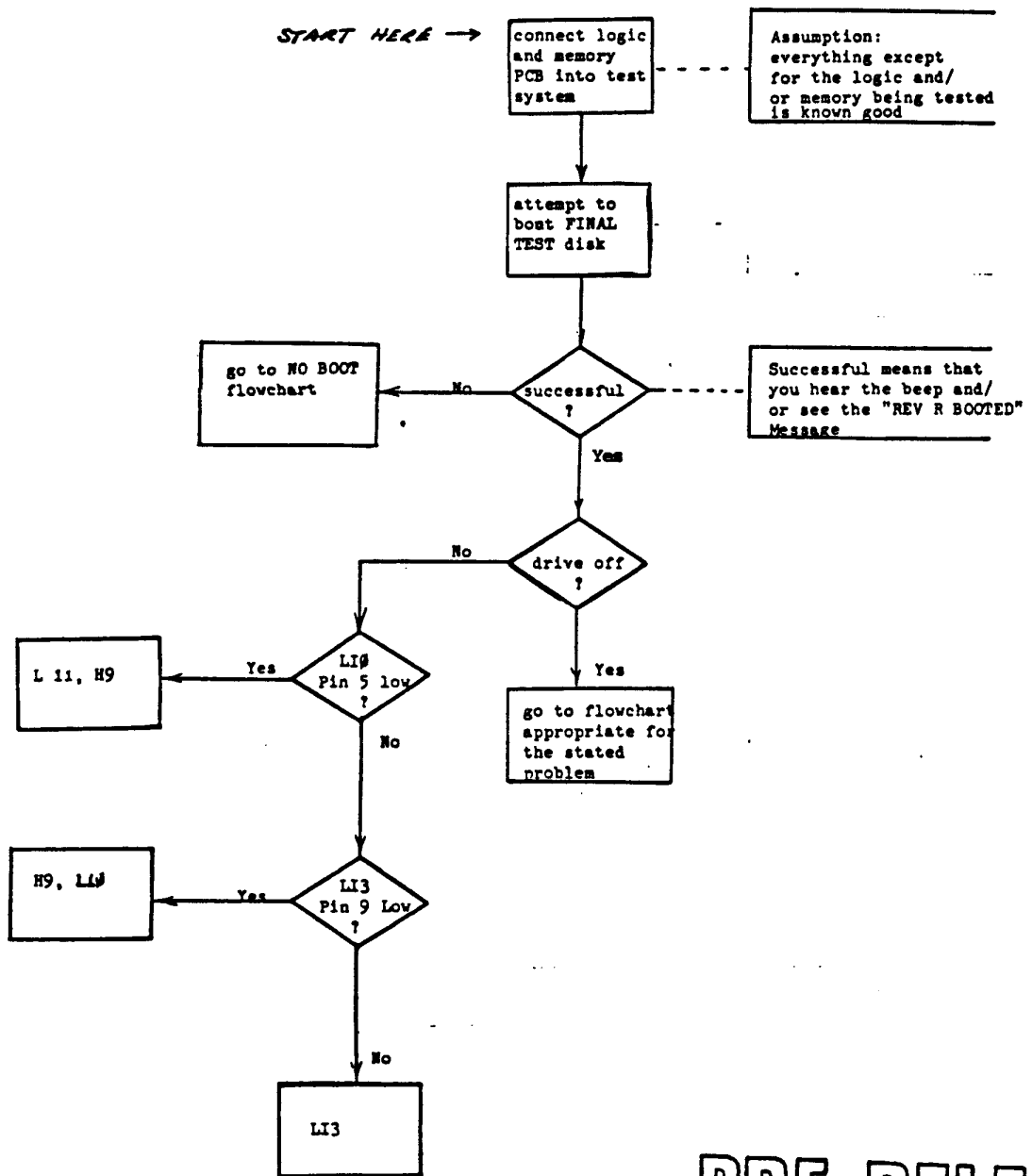
How to use this flow chart:

- 1) Start with the box at the top of the EXEC page.
- 2) Perform the action(s) indicated in each block. It's a good idea to take notes on what tests you've done and what the results were.
- 3) The diamond-shaped blocks are decision points. Many of them contain a test to be made, and a description of a possible result. After doing the test, take the YES exit path if the result you got from the test matches the one given; otherwise take the NO exit path.
Some decision blocks direct your path based on the results of a previous test (usually just before the decision point). Take the YES or NO exit path based on the results of the indicated test.
- 4) If the system successfully does everything that it should do in the EXEC flowchart, you will be directed to go to the flowchart section appropriate to the problem that you have (**** list here ****). If the system fails, a corrective action may be indicated, or you may be directed to the NO BOOT/NO RESET flowcharts for further tests.
- 5) Most terminal blocks (ones with no exit) contain a list of motherboard chip locations. Replace the chips at the indicated locations one at a time. After each substitution, test the system to see if the original problem has been fixed. If it is gone, great. If it is still there, try the next chip. If you run out of chips in the list, check the inputs and enables to the listed chips. If you find any that are faulty, trace the fault back towards the source.
Some terminal blocks will contain instructions for corrective action. Do what it says, then test the system.
If you haven't fixed the problem, you have reached a place where the flowchart won't help you (though you should suspect the area of the circuit that it has led you to). Good luck. Once you find the problem, see if you can fix the flowchart so that it will cover that problem. Notify Service Engineering in Cupertino of any errors you find in the flowchart, and of any additions or other suggestions you want to make.

**PRE-RELEASE
VERSION**

Flowcharting Worksheet

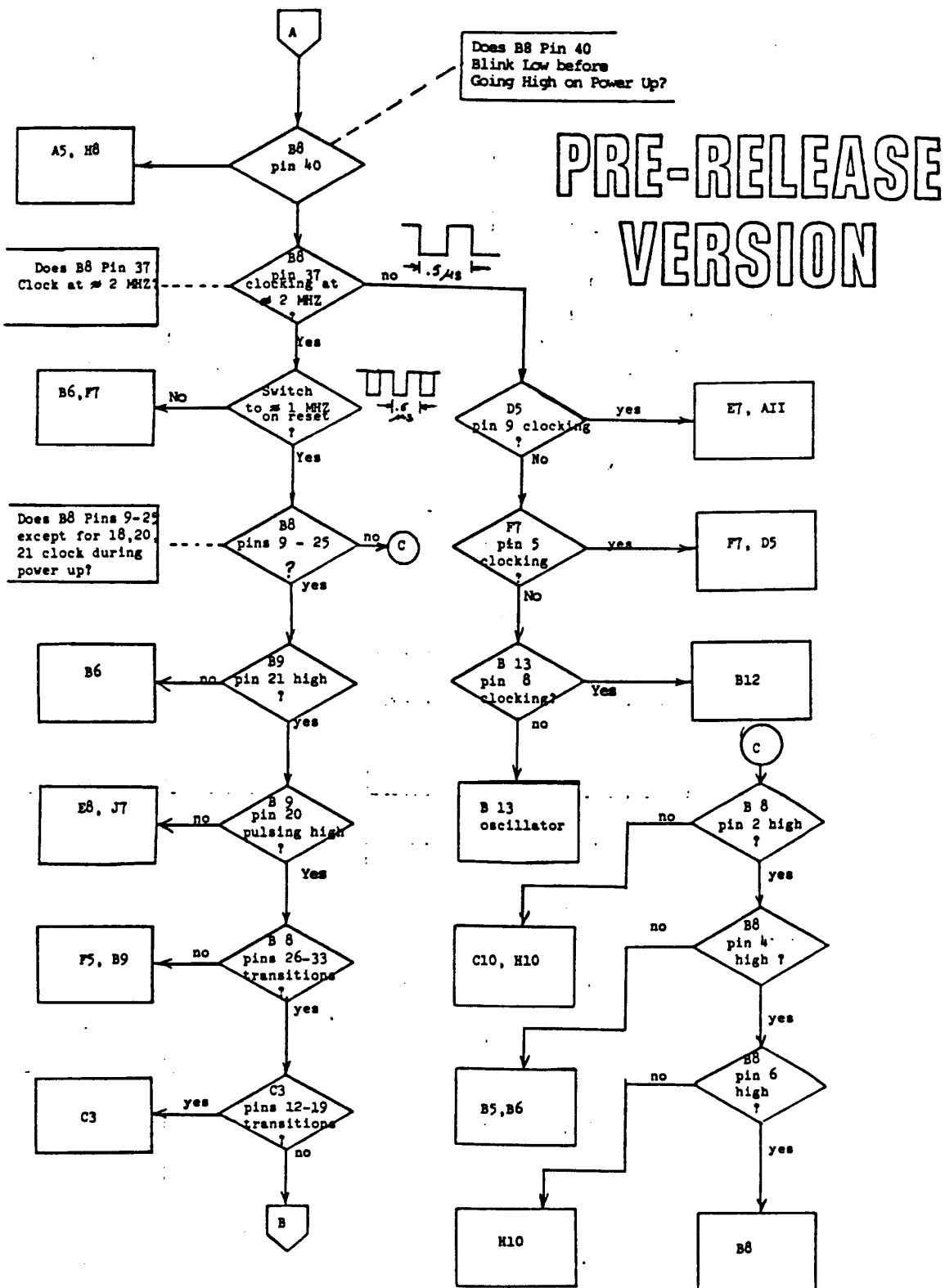
PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 1 OF 1
 CHART ID _____ CHART NAME A/// FLOWCHART PROGRAM NAME EXEC



PRE-RELEASE
VERSION

Flowcharting Worksheet

PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 2 OF 3
 CHART ID _____ CHART NAME NO BOOT (NO RESET) PROGRAM NAME _____

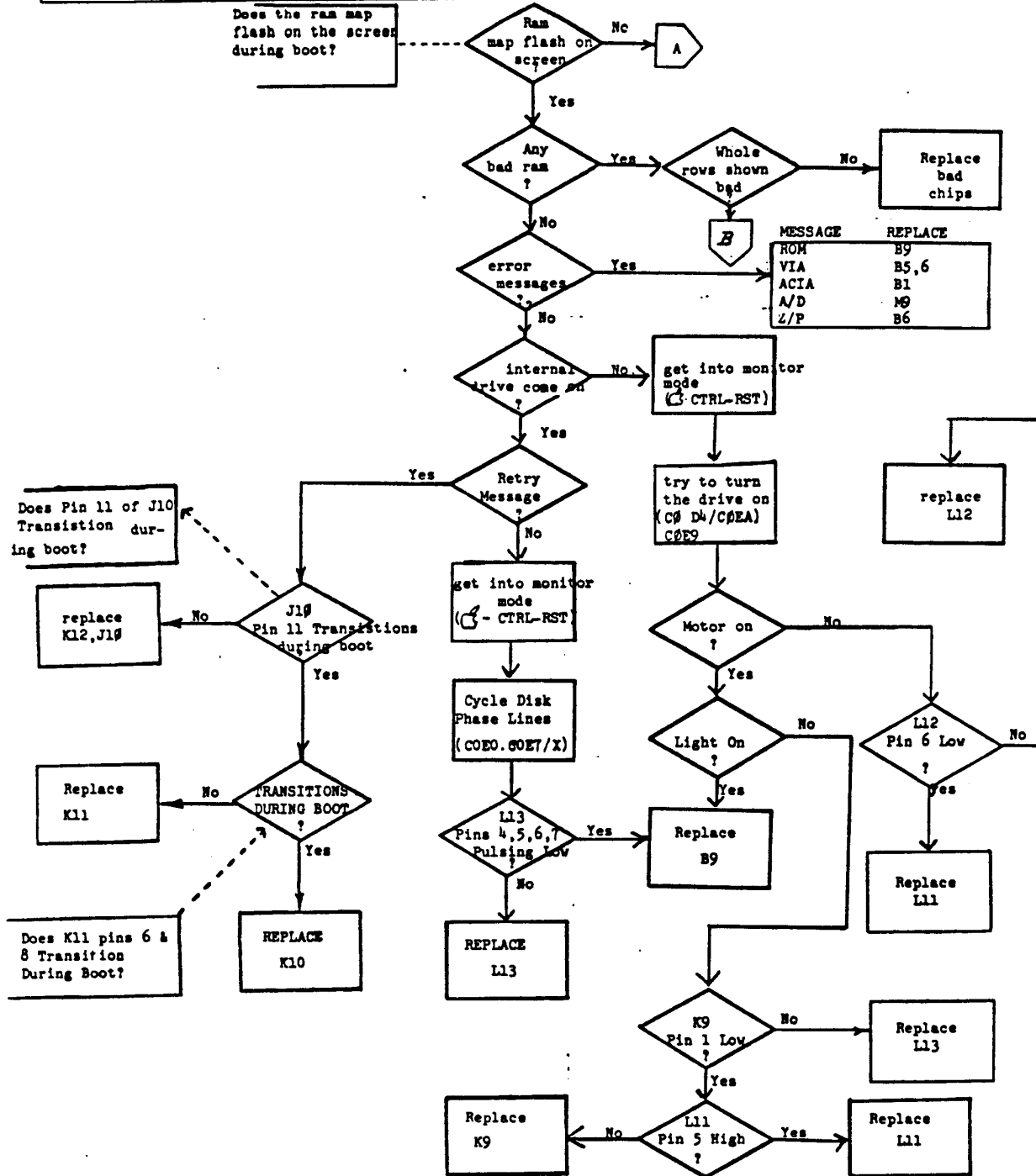


PRE-RELEASE
VERSION

13.30

Flowcharting Worksheet

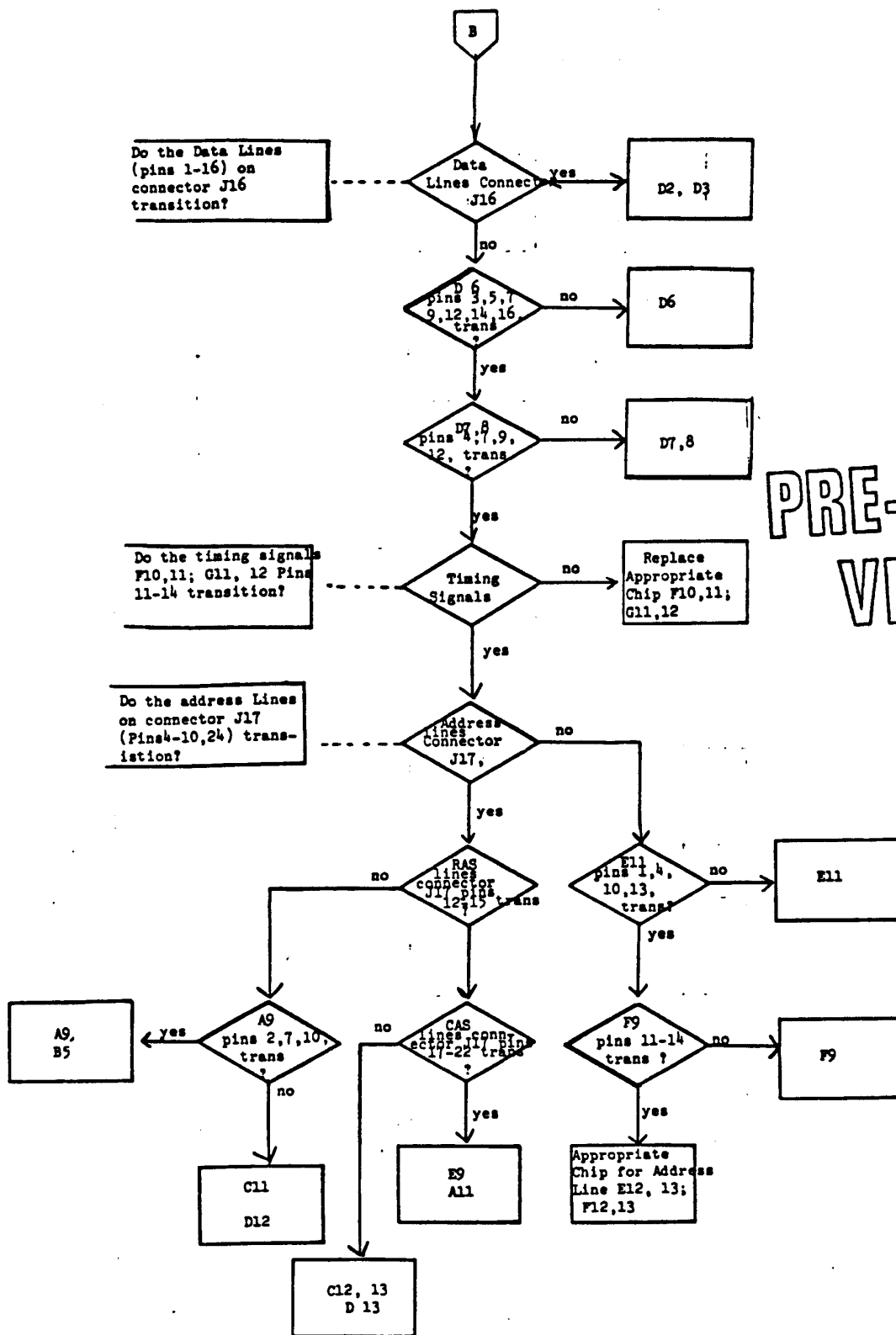
PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 1 OF 3
 CHART ID _____ CHART NAME NO BOOT PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

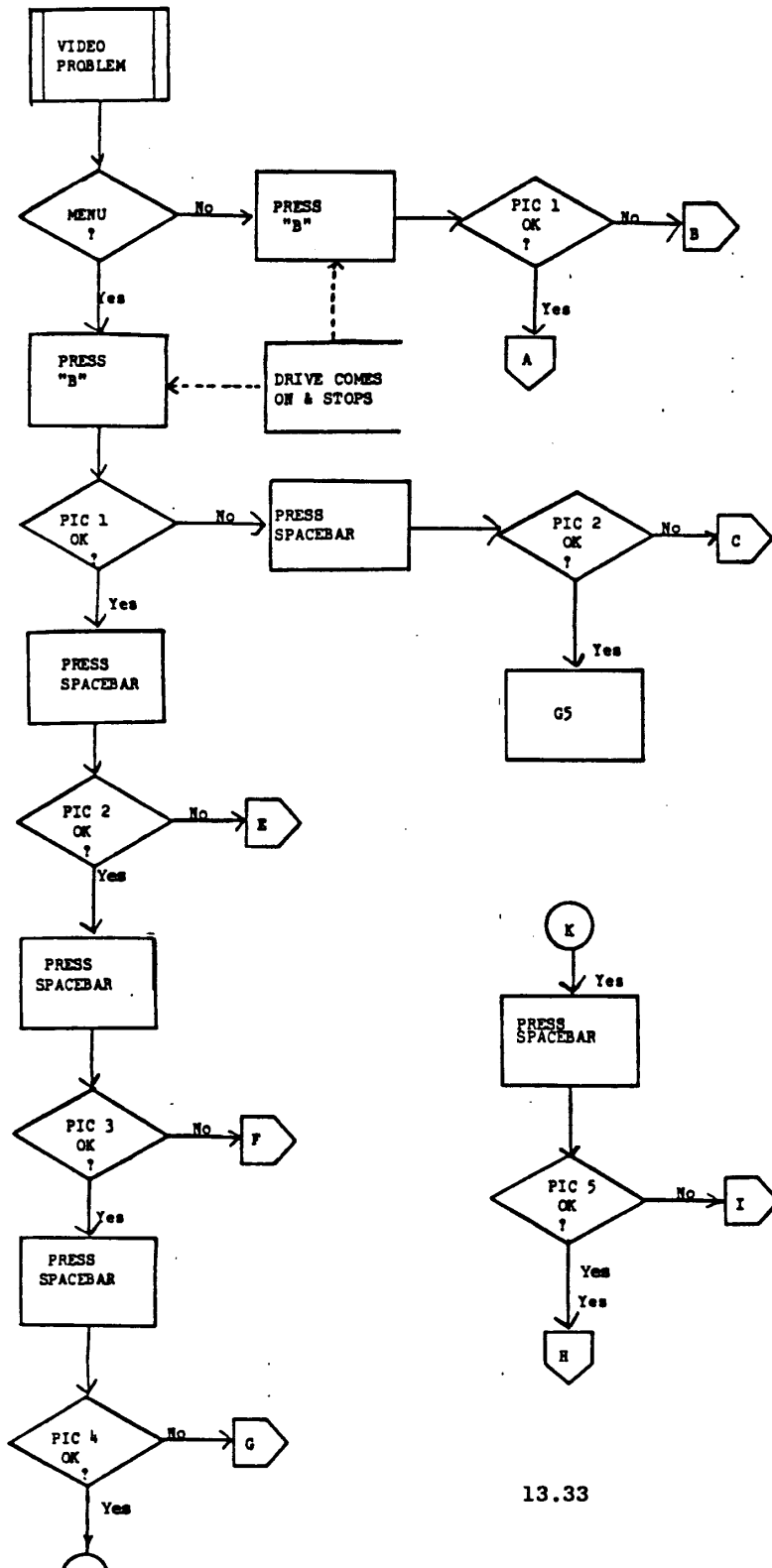
PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 3 OF 3
 CHART ID _____ CHART NAME NO BOOT (MEMORY) PROGRAM NAME _____



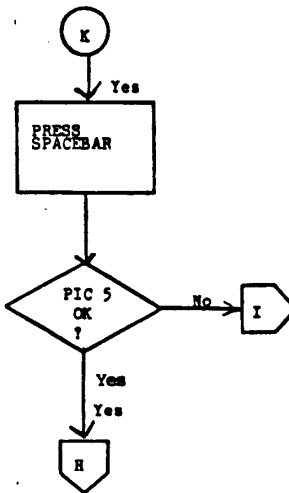
PRE-RELEASE
VERSION

Flowcharting Worksheet

PROGRAMMER _____ PROGRAM NO _____ DATE 3/10/82 PAGE 1 OF 6
 CHART ID _____ CHART NAME _____ VIDEO _____ PROGRAM NAME _____



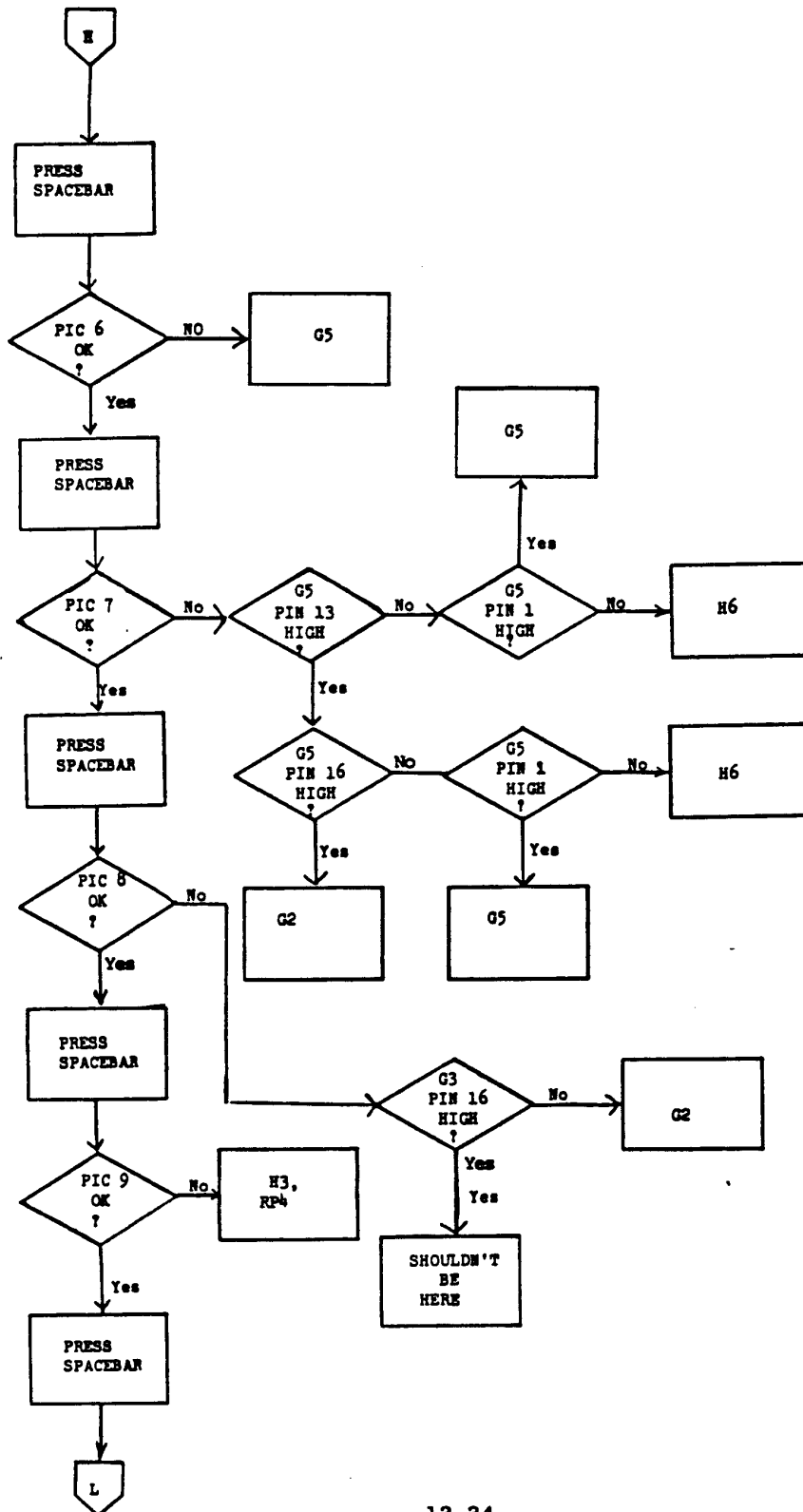
PRE-RELEASE
VERSION



13.33

Flowcharting Worksheet

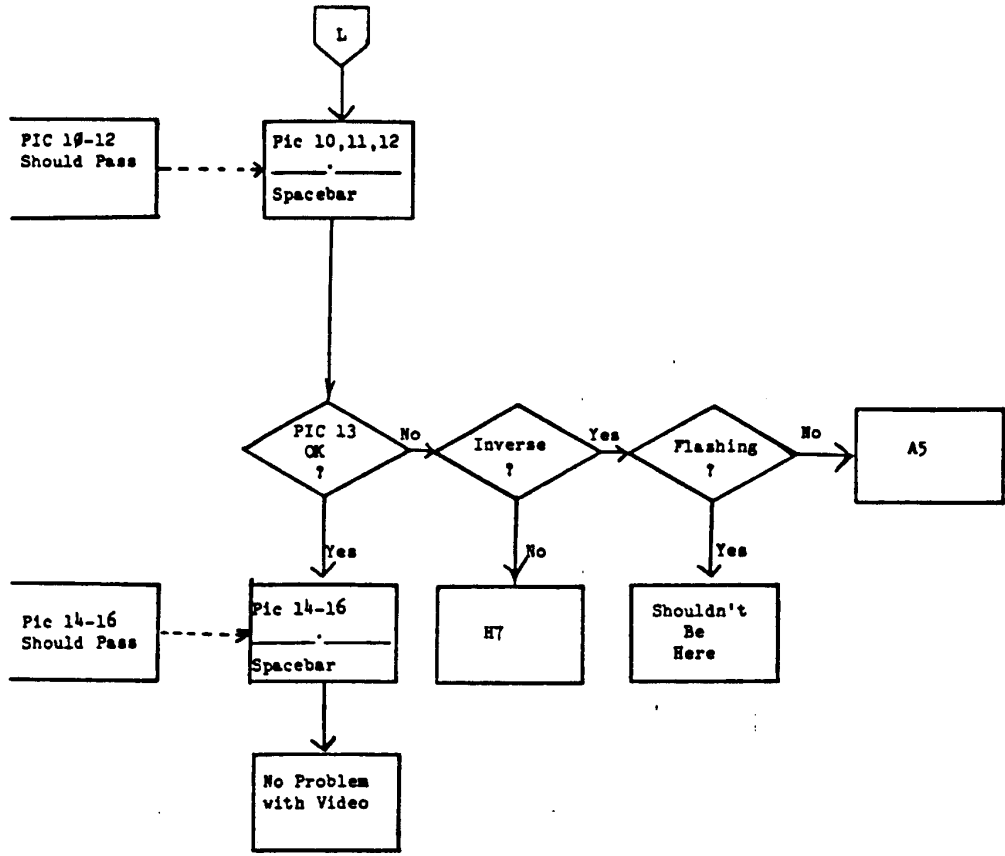
PROGRAMMER _____ PROGRAM NO _____ DATE 3/10 PAGE 2 OF 6
 CHART ID _____ CHART NAME VIDEO PROGRAM NAME _____



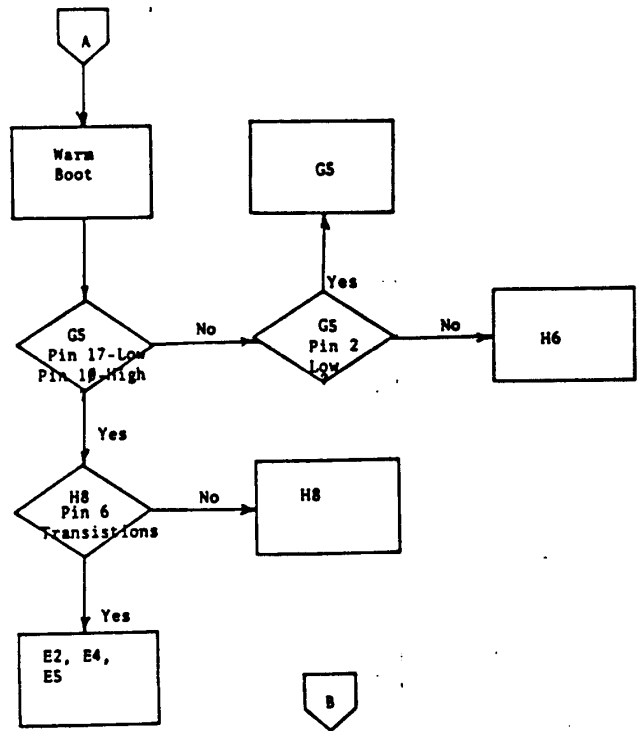
13.34

Flowcharting Worksheet

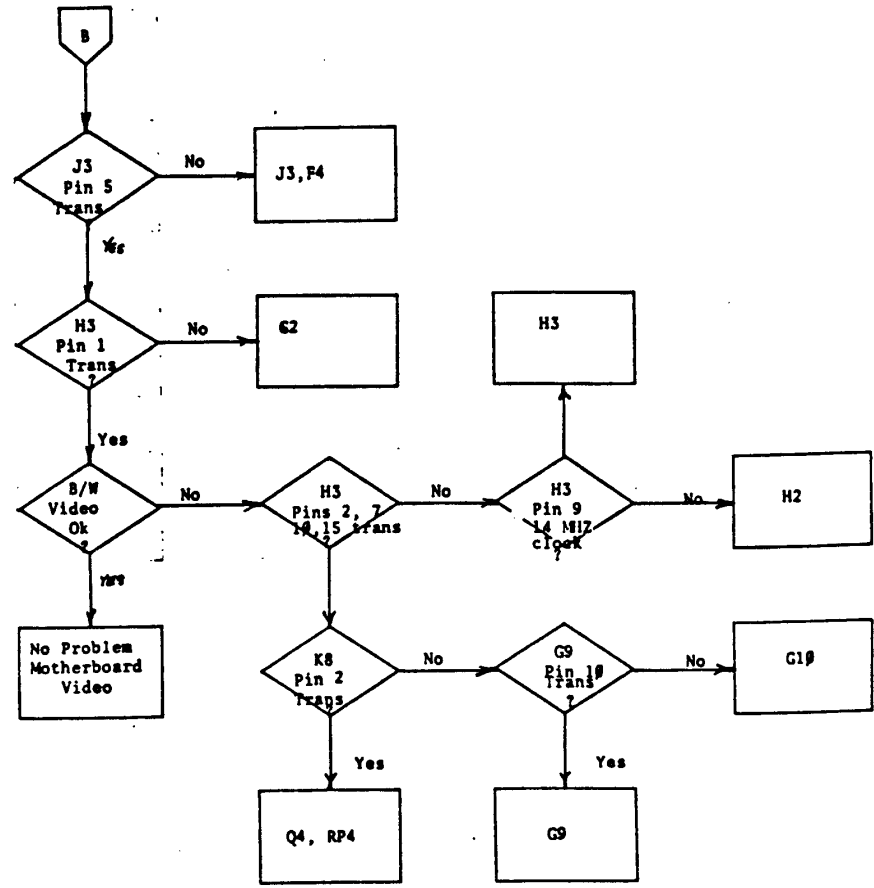
PROGRAMMER _____ PROGRAM NO _____ DATE 3/10 PAGE 3 OF 6
 CHART ID _____ CHART NAME VIDEO PROGRAM NAME _____



PROGRAMMER _____ PROGRAM NO. _____ DATE 3/10 PAGE 4 OF 6
 CHART ID _____ CHART NAME Video PROGRAM NAME _____



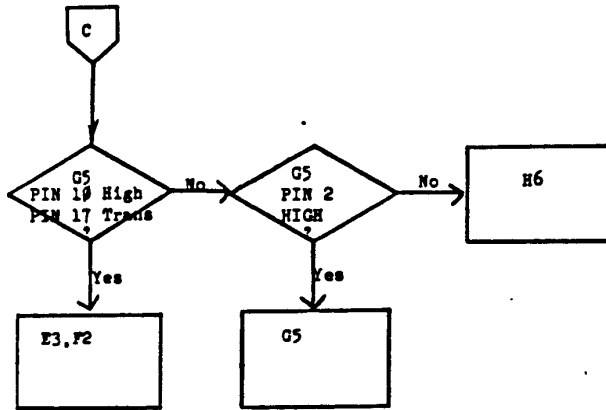
**PRE-RELEASE
VERSION**



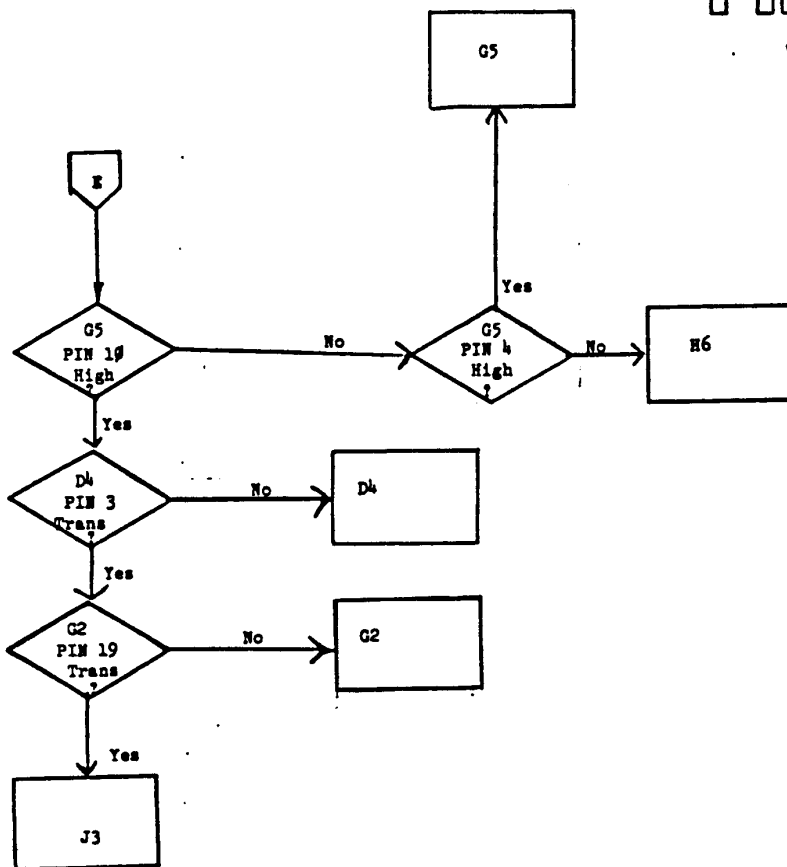
13.36

Flowcharting Worksheet

PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 5 OF 6
 CHART ID _____ CHART NAME VIDEO PROGRAM NAME _____

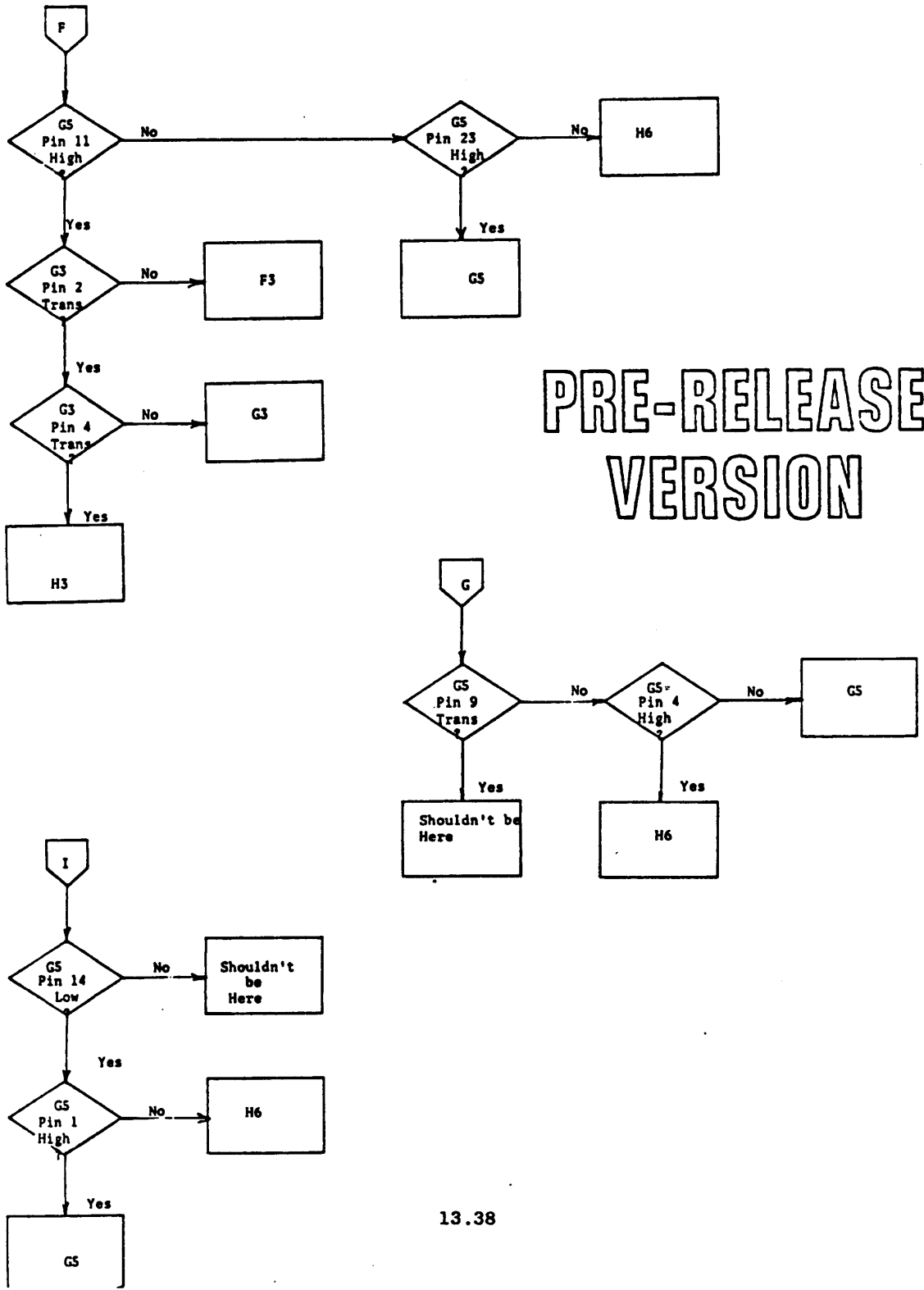


PRE-RELEASE
VERSION

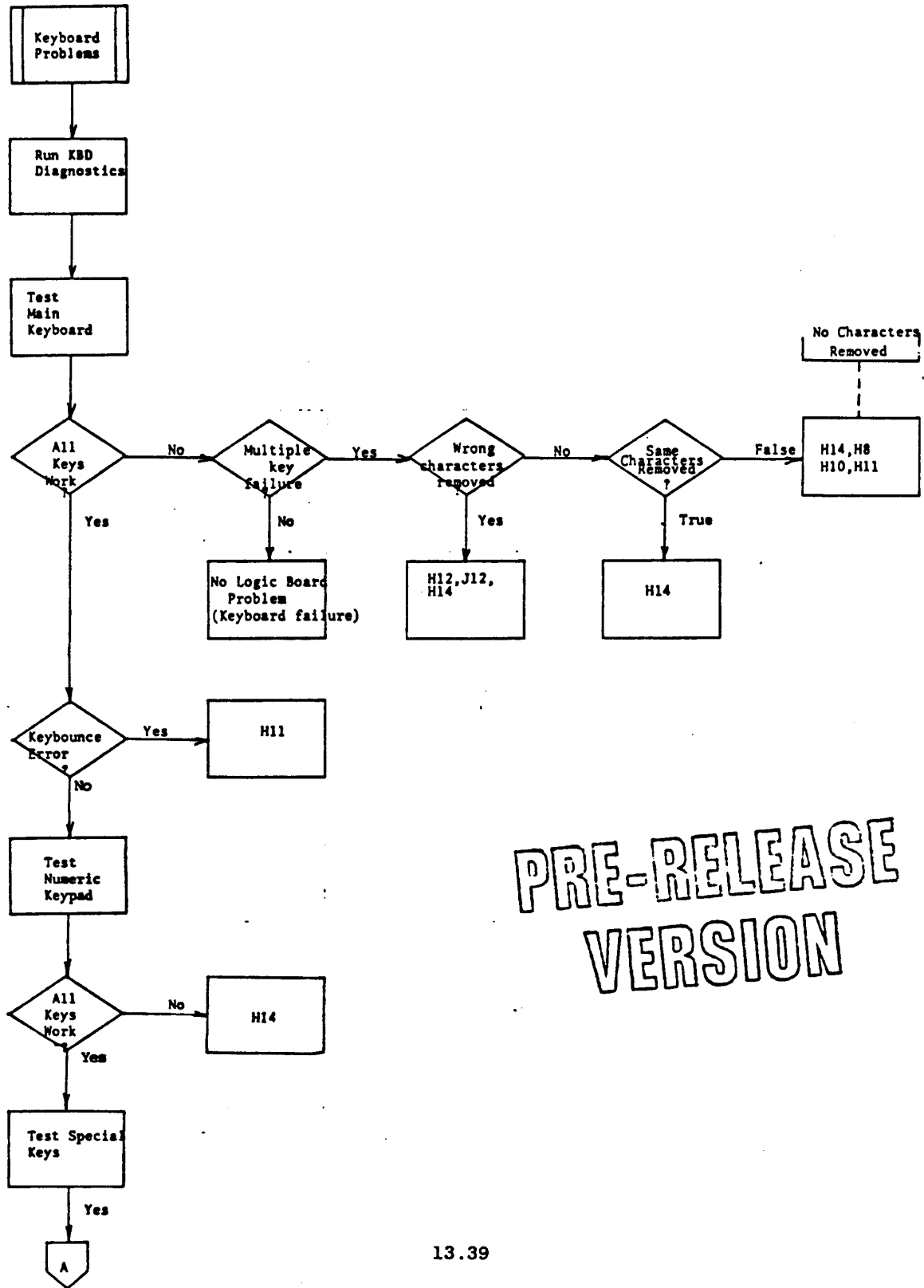


Continuing from page 13.37

PROGRAMMER _____ PROGRAM NO _____ DATE 3/10 _____ PAGE 5 OF 6
 CHART ID _____ CHART NAME Video _____ PROGRAM NAME _____

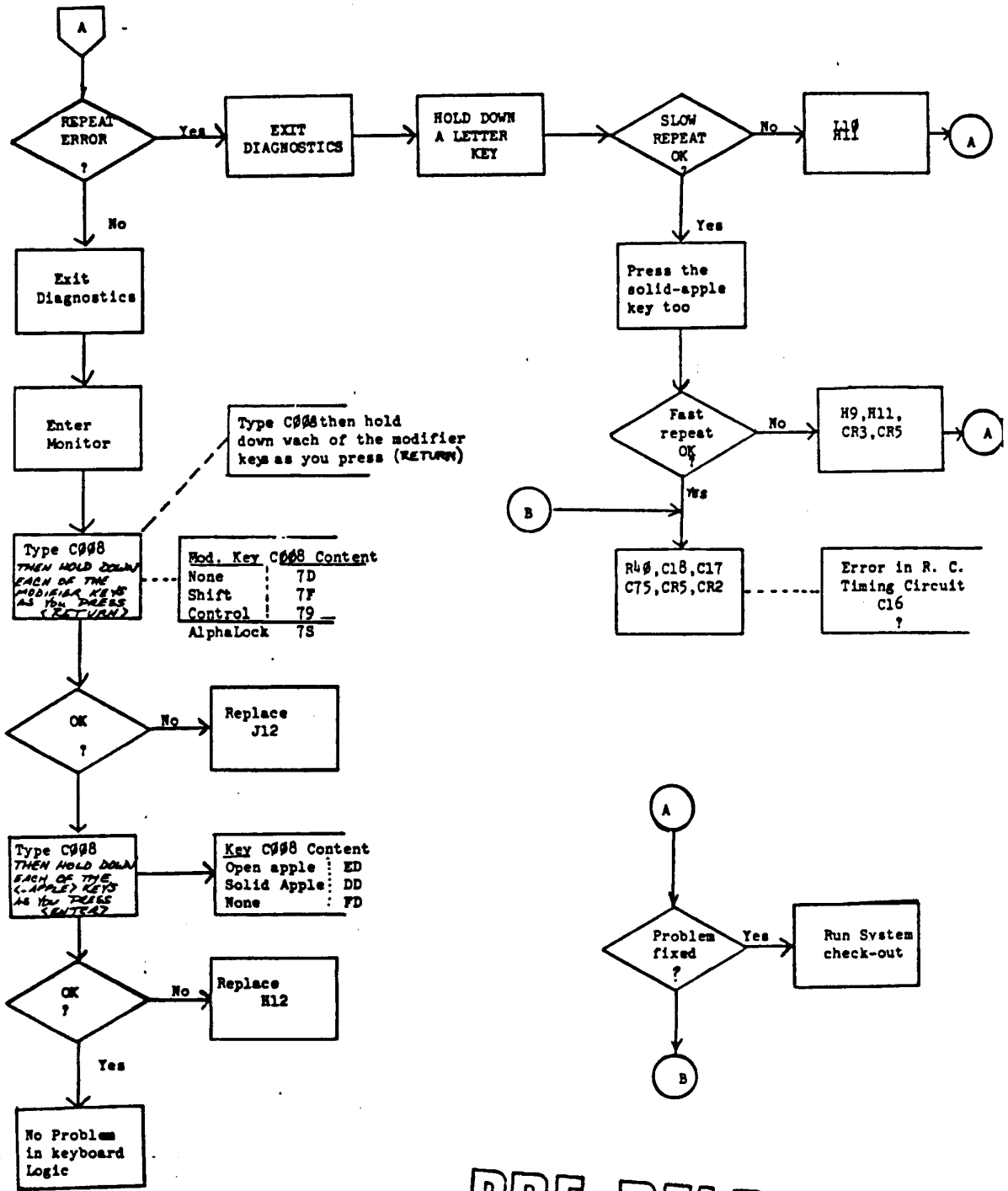


PROGRAMMER _____ PROGRAM NO. _____ DATE 3/11/82 PAGE 1 OF 2
 CHART ID _____ CHART NAME Keyboard PROGRAM NAME _____



PRE-RELEASE
VERSION

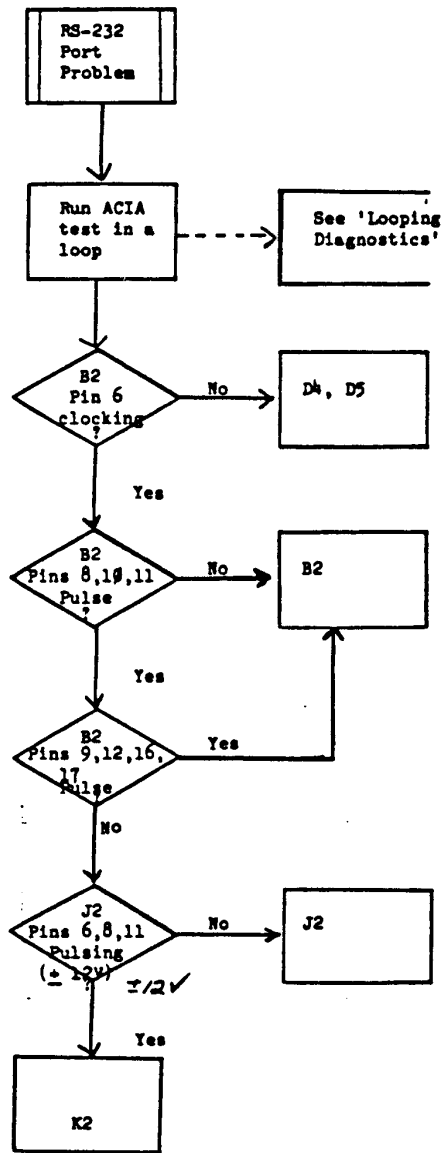
PROGRAMMER _____ PROGRAM NO _____ DATE 3/14/82 PAGE 2 OF 2
 CHART ID _____ CHART NAME _____ KEYBOARD PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

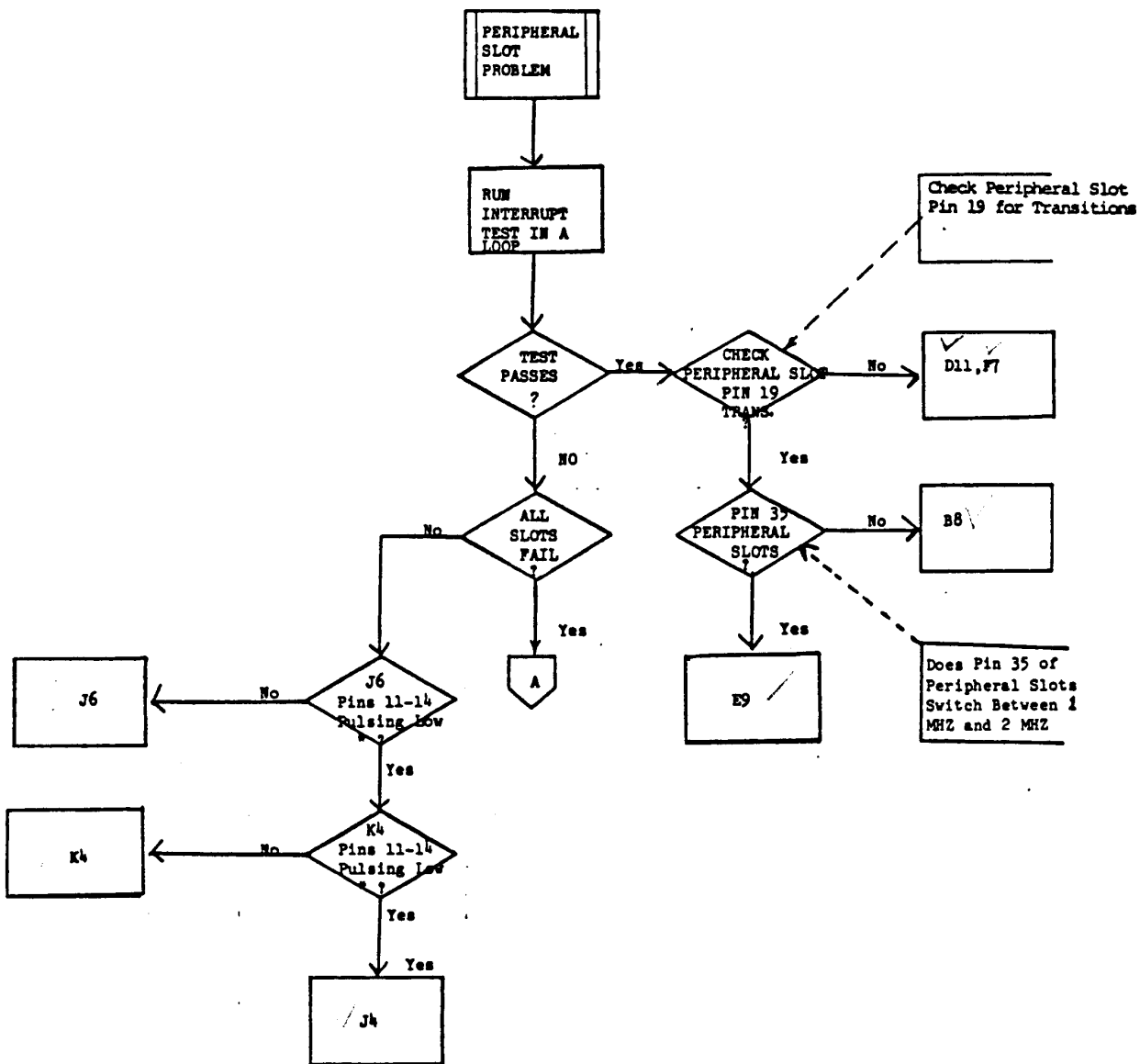
PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 1 OF 1
 CHART ID _____ CHART NAME RS-232 PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

PROGRAMMER _____ PROGRAM NO. _____ DATE _____ PAGE 1 OF 2
 CHART ID _____ CHART NAME PERIPHERAL SLOT PROBLEM PROGRAM NAME _____

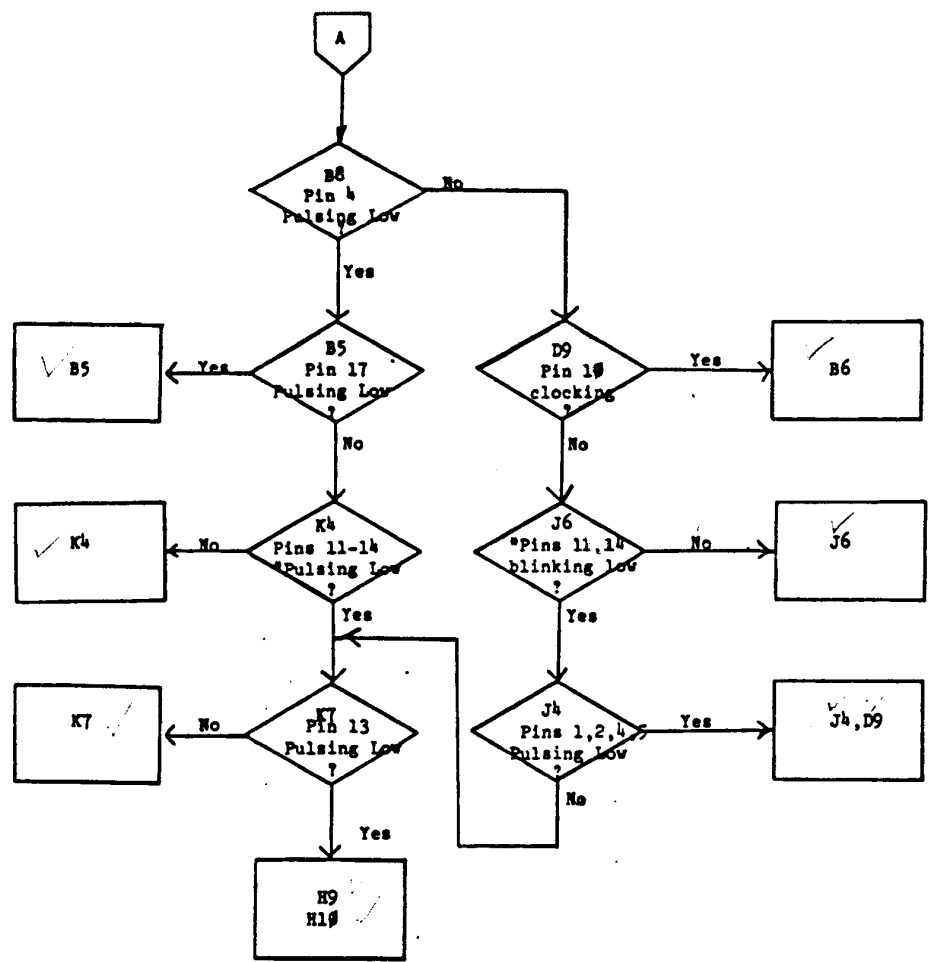


F5-4176 FROM 1024K4 342-0046

*K4 and J6 Pins 11-14 are very hard to see. If they trigger the scope, they're probably okay.

PRE-RELEASE
VERSION

PROGRAMMER _____ PROGRAM NO. _____ DATE _____ PAGE 2 OF 2
 CHART ID _____ CHART NAME PERIPHERAL SLOT PROBLEM PROGRAM NAME _____



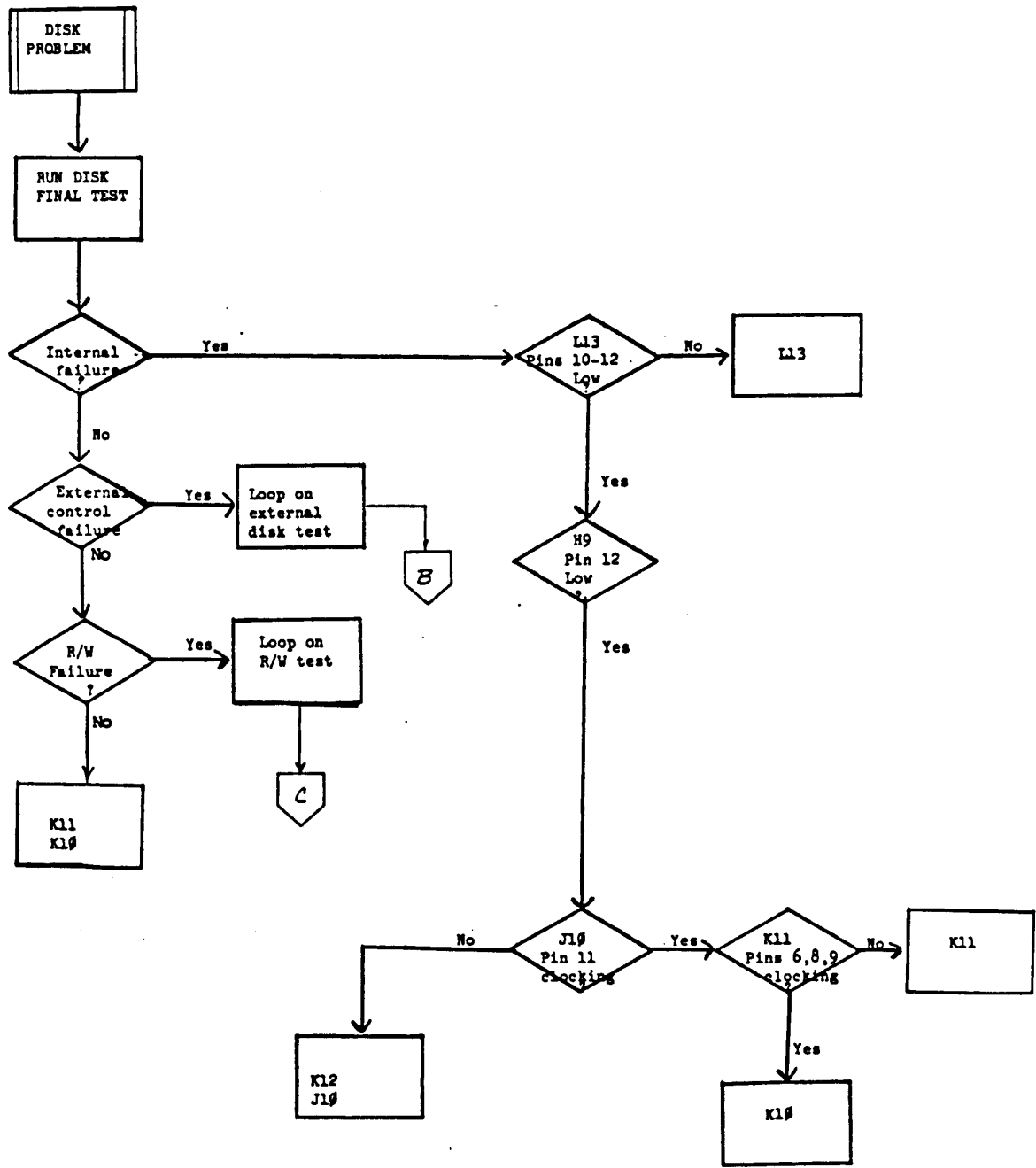
*K4 and J6 pins 11-14 are very hard to see. If they trigger the scope, they're probably okay.

PRE-RELEASE
VERSION

13.43

Flowcharting Worksheet

PROGRAMMER _____ PROGRAM NO _____ DATE 3/15 PAGE 1 OF 3
 CHART ID _____ CHART NAME _____ DISK _____ PROGRAM NAME _____

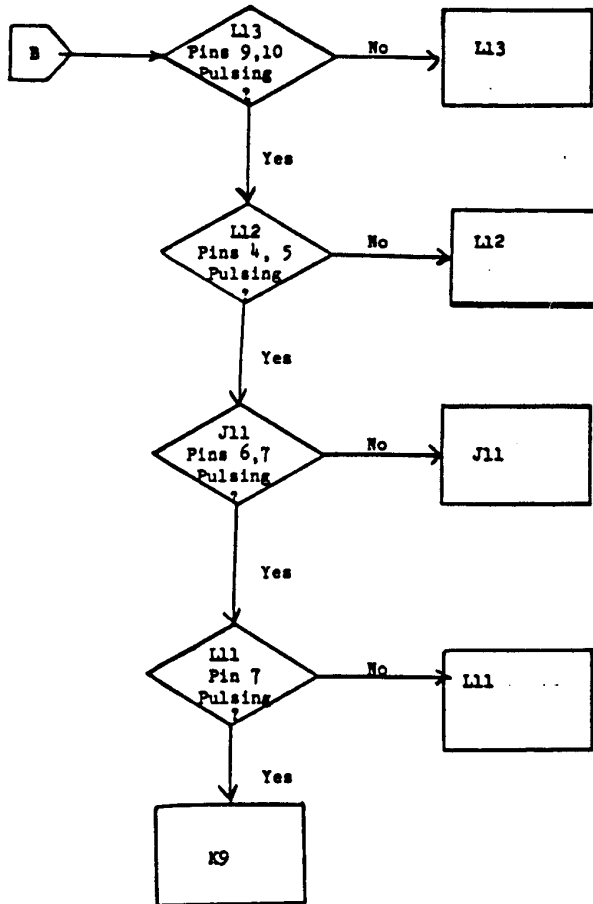


PRE-RELEASE
VERSION

13.44

Flowcharting Worksheet

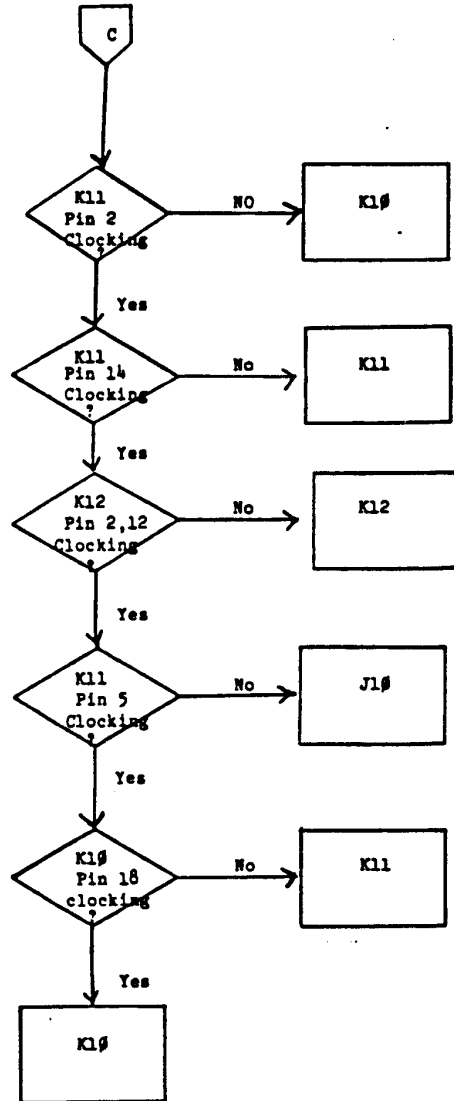
PROGRAMMER _____ PROGRAM NO _____ DATE 3/15 PAGE 2 OF 3
CHART ID _____ CHART NAME _____ DISK _____ PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

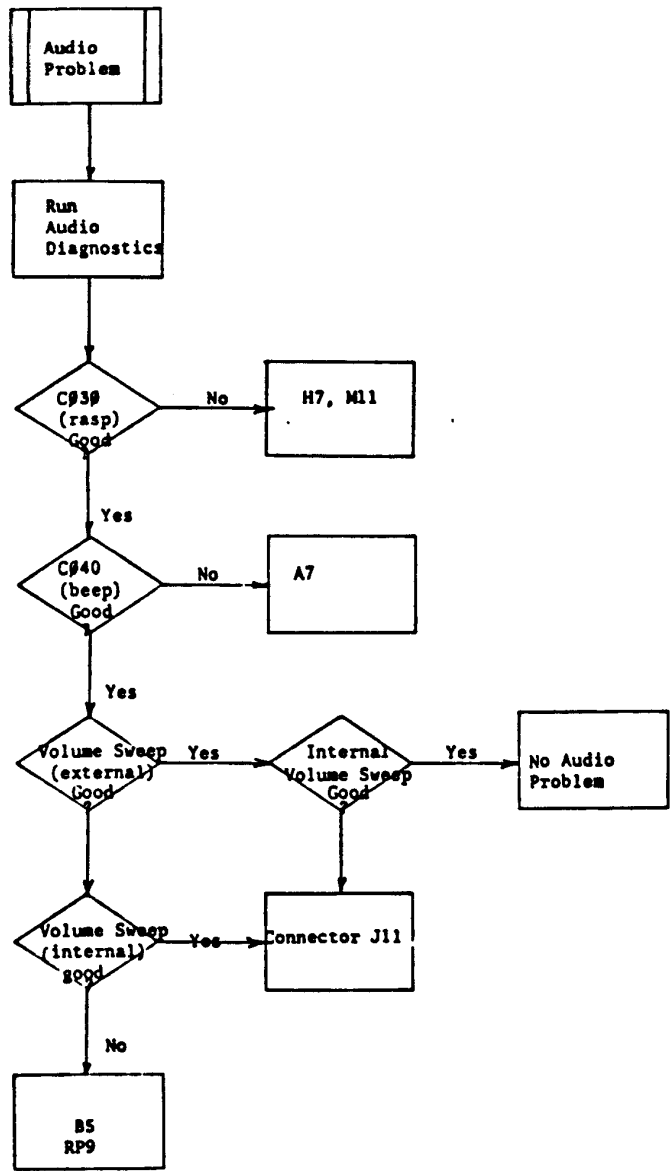
PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 3 OF 3
CHART ID _____ CHART NAME _____ DISK _____ PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

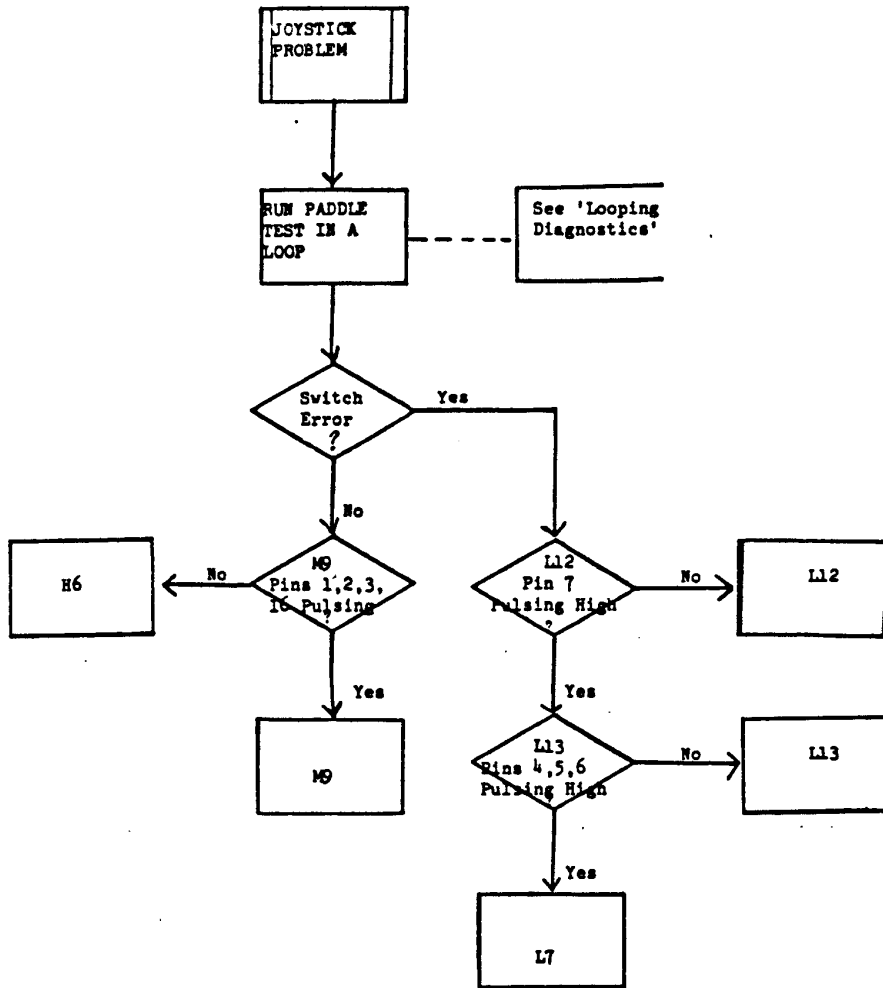
PROGRAMMER _____ PROGRAM NO. _____ DATE _____ PAGE 1 OF 1
CHAIR ID _____ CHART NAME _____ SOUND _____ PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

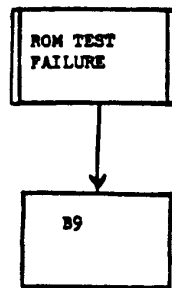
PROGRAMMER _____ PROGRAM NO _____ DATE _____ PAGE 1 OF 1
 CHART ID _____ CHART _____ JOYSTICK _____ PROGRAM NAME _____



PRE-RELEASE
VERSION

Flowcharting Worksheet

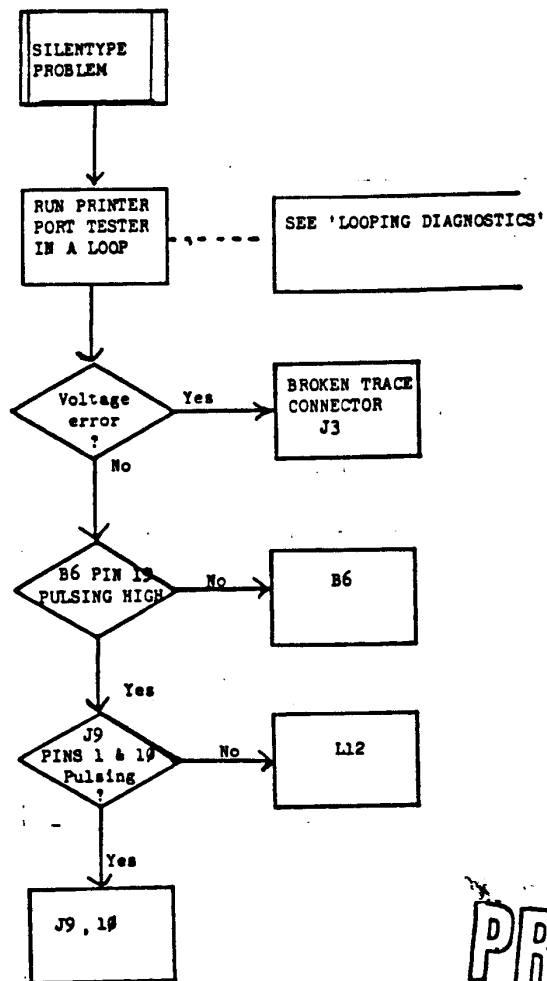
PROGRAMMER _____	PROGRAM NO _____	DATE _____	PAGE <u>1</u> OF <u>1</u>
CHART ID _____	CHART NAME _____	ROM _____	PROGRAM NAME _____



**PRE-RELEASE
VERSION**

Flowcharting Worksheet

PROGRAMMER _____ PROGRAM NO. _____ DATE _____ PAGE 1 OF 1
CHART ID _____ CHART NAME _____ SilenType _____ PROGRAM NAME _____



**PRE-RELEASE
VERSION**



FAULT ISOLATION - TIPS AND HINTS

The following pieces of information are by no means an absolute guarantee of success in isolating a failure mode. This is intended only as a GUIDE or AID to assist you in finding a logical and likely place to begin your troubleshooting of the particular failure mode.

F6E6 TEST (12V MAIN LOGICS)

RAM indicates a memory failure. Use the chart included below to determine which chip on the memory board is failing. Other possibilities are the memory board connectors. If it is an addressing problem, check J17, (the connector on the right) and if it is a data problem, check J16, (the connector on the left).

RAM FAILURE CHART 128K SYSTEM DISPLAY

DIAGNOSTIC RAM

.....

A 1 showing any place that a dot is shown here indicates a failure. The position of the one shows which chip has failed. The chart below shows the chip location on the memory board.

```

B9 B8 B7 B6 B5 B4 B3 B2
B17 B16 B15 B14 B13 B12 B11 B10
B9 B8 B7 B6 B5 B4 B3 B2
B17 B16 B15 C14 B13 B12 B11 B10
C17 C16 C15 C14 C13 C12 C11 C10
D9 D8 D7 D6 D5 D4 D3 D2
D17 D16 D15 D14 D13 D12 D11 D10
C9 C8 C7 C6 C5 C4 C3 C2
    
```

ROM indicates that the ROM failed. Check the ROM at B9 (341-0031-01). Usually the ROM itself is bad when this message appears.

VIA indicates that a register in one of the 6522's has failed. These two parts are the 40-pin IC's at location B6 and B5.

ACIA indicates that the 6551 located at B1 has failed. A/D is an indicator of a bad read of either the high or low reference voltage of the 9708 chip located at M9.

ZP indicates a zero page register failure. The zero page register is port B of



the 6522 located at B6. Other possible chips for this failure are the S257s at locations D7 and D8 or the Ls132 at D4. This is where the zero page portion of the address gets fed to the system memory. Also note the NOT ZPAGE signal which should originate at the LS51 at B11.

RETRY is a message the system gives when it is unable to "boot" the disk that is (or should be) in the internal disk drive.

OTHER FAILURE MODES:

These are usually identified when running the normal system test diskette.

NO RESET is probably the most difficult problem to fix and the easiest to identify. It's main symptom is that when power is turned on, absolutely NOTHING happens. The disk does not even ATTEMPT to boot and there are no beeps. Keep in mind that SIGNATURE ANALYSIS is a very good way to find a NO RESET problem. Below are some of the things that you can check fairly quickly:

Is the keyboard light lit and is the LED on the PC board lit. It not make sure all power is available. +5VDC, -5VDC, +12VDC, & -12VDC.

Make sure that +5VDC and ground is available at each row. (Especially rows B, D, F, & G).

Check the levels at the RESET, NMI, IRQ, and RDY pins of the CPU at B7.

Swap the ROM (B9) with a known good ROM. If there is no difference replace the original.

Swap the CPU (B7) with a known good CPU. If there is no difference replace the original.

NOTE: These last two items are the most common reasons for NO RESET.

Power OFF and check for shorts on the Address or Data Bus.

Make sure that all clocks are running. Phase 0, PRE1M, 14M, 3.5M, and 7M.

Other devices that are frequently causes of NO RESET are listed below:

<u>LOCATION</u>	<u>DEVICE</u>	<u>COMMENTS</u>
Row F & C	7643	high failure items
Row B	6522	B6 is more likely
C3	8304	high failure items
Row E & F	74S153	high failure items
A5	NE556	and supporting circuitry
D13	74S374	high failure items
D6	74LS244	high failure items
D3 & D7	74S257	Not very often
K9	74LS04	not very often
D9	74LS02	moderate failure item
D11	74S74	Make sure its not LS



G7 & J7	74LS133	high failure items
G8	74LS139	high failure items
H8	74LS04	high failure item
J8	74LS32	moderate failure item

For a RAM addressing problem check the memory board connector on the right side, (J17), and for a RAM data problem check the memory board connector on the left, (J16). NOTE: A RAM data problem will USUALLY run the F6E6 test.

Miscellaneous reset problems have been caused by the following:

- diode CR4reversed, open or missing
- RP19 (S19).....3.3K shorted
- Xtal.....not oscillating
- reset key.....bad
- power or ground.....missing due to open pin/trace

It may also help you to know that there is an 85% chance of the problem being found on sheets 2 and 3 of the Apple /// schematic, and a 13% chance of being found on sheets 9 and 4. Good luck on the other 2%.

NO BOOT is recognized by the fact that the system actually ATTEMPTING to boot even though it doesn't succeed. Video may or may not be present. You may see the RETRY message, and the disk may run itself off, or it may stay on. Some of the more common things to check are:

- The 9334's at L12 and L13.
- The 74LS04 at K9.
- The mostly likely choices are:

device	location
74LS323	K10
74S471	K11 (P6A prom)
74LS174	K12

Concentrate your troubleshooting efforts on sheet 6 of the Apple /// schematic, unless you find a problem with the NOT devel-6, or the NOT Q3 signals. Any other problem external to sheet 6 would most likely also show up in a RAM test, or video test.

VIDEO problems can be very simple at times and downright troublesome at other times. Some of the most common failures are:

By far, the most likely candidates, are the 74LS374's, located at E2, E3, F2, F3, G2, & G10. These IC's are the cause of >70% of the video problems.

It is also a good idea to check the two 2114's located at E4 and E5.

The next most frequent failure is the 74LS153 at L8. This is U90 as shown on sheet 5 of the schematic.

Video problems are also very likely to be misdiagnosed timing problems. For these check the 64S195 at D10 and the 74LS374 at D11.

If the video horizontal or vertical sync appears to be messed up, check the 74LS161's located at F10, F11, G11, and G12.



If none of the above items point you toward the real problem, begin your troubleshooting on sheet 5 of the schematic where 80% of the problems occur, and if necessary, go on to sheet 9, where most of the remaining video problems will be found.

Keep in mind that an off-frequently crystal can kill the color or produce bad color.

INTERRUPT problems almost always end up being one of the VIA's (or 6522's). However, there have been a few other reasons for interrupt failures as described below:

I/O NMI is a symptom for sheet 8 of the schematic. Check the 74LS132 at H10 and the 74LS139 at J11.

IRQ signal missing can be caused by the 74LS21 at J4.

Sometimes the problem is not actually an interrupt problem but the system's inability to communicate with the interrupt test cards. In the case of a missing IO SELECT, check the 74LS138 at J6, and in the case of a missing DEVICE SELECT, check the 74LS138 at K4.

Remember to check the connectors at the slots for continuity to the 74LS138's mentioned above.

Lastly, here's what happens during the INTERRUPT test:

- Step 1 Disable all interrupts, mask NMI, enable I.O space, reset ACIA, re set all four slots, and set up the 6522's as they are normally used.
- Step 2 Check both 6522 int enable register bits 0-6 to see if they can be set and cleared.
- Step 3 Verify that IRQ and IONMI are clear as they should be.
- Step 4 Clear both 6522 interrupt flag registers and verify that they are indeed clear.
- Step 5 Check the 6522 interrupt flags to see if they can be set and cleared when enabled and verify that they cannot be set when disabled.
- Step 6 Repeat once for each slot.
 - a. Set slot IRQ by using IO SELECT and verify by polling slot.
 - b. Clear slot IRQ using CO2X and verify by polling slot.
 - c. Verify that the 6522 at B6 caught the IRQ and that it was the correct one and that it can be cleared.
 - d. Set slot IONMI using DEVICE SELECT and verify by polling IONMI.
 - e. Clear slot IONMI using CO2X and verify by polling IONMI.
- Step 7 Check for shorts between slot IRQ's by setting IRQ on one slot at a time and checking each IRQ on the other slots.

AUDIO troubles are usually fairly simple to repair. The most common ailments are listed below:

The most common frequent failure is the LM380 which is located at M11, and rarely you may find a problem in its associated circuitry, notably R36, a 1K resistor and C12, a 10uF cap.

The second most frequent, usually noted on the C040 portion of the test, is the 556 located at A7.

Another common ailment is J11, the external speaker jack which is the mini-phone jack at N4. Note that a bad contact in this jack can also prevent you from hearing any internal sound.

Failure of the C030 test can be caused by the 74LS74 at H7.

And last, a failure of the FFE0 test can be caused by either the 6522 at



B5, or RP9 the SIP located at C5.

ACIA problems are also usually fairly simple and usually end up being caused by only a few items:

The most common failure is, of course, the 6551 itself. This is located at B2.

The next items to check, will be the 1488 at J2 and the 1489 at K2.

The third most common failure is J4, the 25 pin D connector at N2.

The less likely things to check, are, the SEL6551 from the 74LS138 at K4-7, and the RC network (or R-pak) at N3.

PADDLE PORT problems frequently coincide with printer port problems. This is because they both share the DB-9 connector N10, (J3). Some causes of paddle port problems are:

The 9708 located at M9. This is the most common cause of failures.

The second most frequent cause of failures is the RC network at N10.

Other things to check that can sometimes cause problems, are the 9334's at locations H6, L12, and L13. These IC's supply some of the enable necessary during the paddle port test, including the disk phases, and disk side 2/1 signals used to test the switches, as well as the ENSIO and ENSEL signals.

Also check the enables coming from the 74LS138's at K4 and K7.

Finally check the 74LS251 at L7 and its enables.

RAM FAILURES can be almost as much fun as no reset can be. Some of the things to look for are:

Of course, the most obvious thing to check is the F6E6 test results and see which row, group, chip(s) are causing problems. Also check the two memory board connectors J16 and J17 at this time.

You have about 98% chance of the problem being on sheet 2 of the schematic. With the highest probability, being the 74LS399 at A9 and the 74S3 74S374 at D13.

Next check the 7643's at locations C10, C11, C12, and C13, and then check the 74LS153's at E12, E13, F12, and F13.

It may be a little help to do the following IF and ONLY IF you can get into the monitor routines using a CONTROL-APPLE-RESET:

a. Type the following:

FFDO: 0/FFDF:IF/FFEF:0 (RETURN)

You have just set the bank register and zero page to 0 and disabled the screen and IO addresses. (Note that FFEF is the bank register and you may want to try settings other than 0 if this setup doesn't find your problem.

b. Now type:

0.FFFF/X (RETURN)

The address bus should now be one big 16-bit counter and therefore easier to trace with an oscilloscope.

PRINTER PORT failures will show up as paddle port failures most of the time. The few exceptions are, that some of the polarities of the enables are reversed, causing the signal direction to also be reversed. (For example, a paddle port INPUT becomes a printer port OUTPUT). Also, be sure to check the connections to slot 1 to insure that the test hardware will function properly. The best way to check this, is to check for continuity from slot 1 to slot 4, (except for pins 1, 30, and 41).



DISK failures will usually prevent the system from booting properly. There are, however, a few things which should be mentioned here.

J1 and J6, the EXTERNAL and INTERNAL drive connectors, respectively, are in parallel, with the exception of pins 14, 21, 22, and 26.

A disk phase of SIDE 2/1 failure will usually also show up as a switch failure in the paddle port test.

The RC filter networks at N13 and M13 are also a good place to check for problems. (See the section below on useful addresses in order to toggle a particular signal).

KEYBOARD failures are another of those problems that, while they occur frequently, are nevertheless, fairly simple to correct. Some of the more common failure modes are:

If the fast repeat or the super fast repeat either fail to work, work all the time, or repeat way too fast, check the 556 located at L10 and also the C16 capacitor.

Another thing to check if there is no fast repeat is diode CR5, located M7.

If the data from the keyboard is wrong, check the keyboard encoder chip at H14, (check its power connections too,) or one of the 74LS257's located at H12 or J12.

If one of the above hasn't led to the problem, check the 74LS05 at H9, or the 74LS132 at H10.

Lastly, if the power light of the keyboard fails to light, check transistor Q9.

ROM failures are almost ALWAYS, (99% of the time), the ROM chip itself located at B9. On rare occasions, it could be that the IC located at F9 is NOT a 341-0055, as it should be.

THERMAL or "HEAT SENSITIVE" failures: (These should be verified as heat sensitive as described in section 4.4 above).

After verifying that the unit is actually heat sensitive, you will have to locate the area of the board where the problem is. Do this by looping on the portion of the diagnostic that failed and moving the heat gun over the affected areas. If the unit does not fail within a few minutes, move the heat gun to a different area of the board and continue the process.

When the test begins to fail, you have found the correct AREA of the board.

You may now want to use, a monitor command of some type with the X (or repeat) command, for your troubleshooting.

The most frequently heat sensitive devices are the ROM located at B9, the 6502-B located at B7, and the 7643 PROMS (with the 342-00xx numbers), scattered throughout the board.

TROUBLESHOOTING INFORMATION

SYSTEM DEATH ERRORS and their meaning:

error code	meaning
\$01	bad break (BRK) from SOS
\$02	interrupt not found (but received at CPU)
\$03	bad zero page allocation
\$04	unable to lock NMI
\$05	event queue overflow
\$06	stack overflow
\$07	data manager detected invalid request code



```

$08          dmgr - too many device handlers (drivers)
$09          memory too small (<64K)
$0A          volume control block not usable
$0B          file control block crashed
$0C          allocation blocks invalid
$0D          directory is not correct
$0E          pathname buffer overflow (too long)
$0F          invalid buffer number
$10          invalid buffer size (=0 or >16K)
    
```

^ul

MONITOR AND ESCAPE COMMANDS

MONITOR COMMANDS:

COMMAND	REMARKS
addr1 addr2	memory dump, to dump all memory locations from address \$30FF you would type, 3000.30FF and press RETURN.
addr1:byte list	store byte(s) in memory, just like Apple },, to store data starting at address \$3000, you would type, 3000.00 A2 FF 45 9C etc. To store ascii data, you would type, 3000: 'DARRELL' for high bit off, or 3000: "PAYNE" for high bit on.
block <addr1.addr2	read block from disk into memory from address 1 to address 2. Block must be in the range \$0 to \$117 (0-280). For example, 0<2000.20FF would read block 0 of the disk and store the data in memory from \$2000 to \$20FF. NOTE: 1 block + 512 bytes, and the SOS directory is at block 2.
block <addr1.addr2W	same as above except writes data from address 1 to address 2 onto the disk starting at block.
addr3<addr1.addr2M	move data from memory beginning at location address 1 and ending at address 2 into memory beginning at address 3.
addr3<addr1.addr2V	Verify that the data in memory from address 1 to address 2 is the same as data in memory beginning at address 3.
byte<addr1.addr2S	search memory for data that matches byte starting at address 1 and continuing search through address 2.
addrG	call subroutine at address. This is the same as a JSR in assembly language, or a CALL in BASIC.
addrJ	jump to address and begin executing code. This is the same as a JMP in assembly language.
U	calls user routine. This actually does a JSR \$3F8.
X	repeat entire command line. For example, COEO/COE1/X will continue to toggle disk phase 0 on and off.

Note that most of the Monitor commands are the same as they are in the Apple },, except some have been added and others have been improved.

^UW, 1,2

ESCAPE COMMANDS: These are achieved by pressing the escape key followed by the desired command:

310 27N



COMMAND	REMARKS
Cntl-K or up arrow	moves cursor up
cntl-J or dn arrow	moves cursor down
cntl-H or left arrow	moves cursor left
cntl-U or rt arrow	moves cursor right
L	clear to end of Line
P	clear ro end of Page
S	clear screen
4	set 40-column display
8	sey 80-column display

using the monitor to cause a particular signal or group of signals to toggle back and forth between high and low levels, can be a valuable troubleshooting tool. Listed below are some commands you can use from the monitor to do this. They are listed as they are found on the Apple /// schematic, by signal name, (all signals on sheet 2 of the schematic are grouped together, page 3 signals likewise, etc.) The signal name is given first, and the next line gives the Monitor commands you must enter to toggle that signal. {C}, means hold down the control key while pressing the next key, and {R} means press the return key. (NOTE: After you have typed a sequence of monitor instructions and used the oscilloscope to look at the signals produced, if you need to look at some different signal, ALWAYS reset the system, and use the CONTROL, APPLE, RESET sequence to re-initialize the system since it could possibly be in an unknown state, caused by either the problem you are trying to isolate, or the last instruction that you typed in).

Page 1 signals are listed on the page where they originate.

Page 2 signals are normally all running. Address lines and bank switch signals are listed on the page where they originate.

Page 3 signals

Address lines:

FFD0: 0/FFDF :1F/FFEF" 0{R}

0.FFFF/X{R}

Data lines and R/W:

3000:A5 5A/3000.3001/X{R}

Page 4 signals:

GB outputs, (74LS139)

FFCO/FFDO/FFEO/FFFO/X{R}

J6 outputs, (74LS138)

C000/C100/C200/C300/C400/C500/C600/C700/X{R}

K4 OUTPUTS, (74LS138)

C080/C090/COA0/COB0/COC0/COD0/COE0/COFO/X{R}

K7 outputs, (74LS138)

C000/C010/C020/C030/C040/C050/C060/C070/X{R}

H6 outputs, (9334)

C050.C05F/X{R}

6522, Port A, (Environment register)

A000.A2 5A A0 A5 8E DF FF 8C DF FF 4C 04 A0{R}

A000G{R}

6522, PORT B, (Zero page register)

A000: A2 5A A0 A5 8E D0 FF 8C D0 FF 4C 04 A0{R}

A000G{R}

Page 5 signals should always be toggling if the screen is enabled, and something is being sent to the screen, (whether or not it actually gets there).



Page 6 signals

L13 outputs, (9334)

COEO.COEF/X{R}

L12 OUTPUTS, (9334)

CODO.CODF/X{R}

Page 7 signals

BCKSW1-BCKSW4

A000:A2 5A A0 A5 8E EE FF 8C EE FF 4C 04 A0{R}

A000G{R}

L7 outputs (bit 7 only), 74LS251

CO60. CO6F/X{R}

Sound signals will toggle continuously from sound test ACIA signals

COFO: 55/COFO/COFO:AA/COFO/X{R}

Page 8 signals

Keyboard A port read

CO00/CO10/X{R}

Keyboard B port read

CO08/CO10/X{R}

Page 9 signals should all be running. These are all of the main timing signals for the Apple ///.

Useful addresses to know:

If you wish to know the status of the CPU at the time of a system failure, (\$010-\$10), you can use CONTROL-APPLE-RESET, to enter the monitor and then examine memory beginning at \$19F0, for the information. NOTE, this applies ONLY when the failure occurred while running under control of SOS.

- \$19F0-19F1.....PROGRAM COUNTER
- \$19F2.....STACK POINTER
- \$19F3.....ENVIRONMENT REGISTER
- \$19F4.....ZERO PAGE REGISTER
- \$19F5.....BANK REGISTER
- \$19F6.....PROCESSOR STATE REGISTER
- \$19F7.....ACCUMULATOR (A REGISTER)
- \$19F8.....INDEX X (X REGISTER) *****
- \$19F9.....INDEX Y (Y REGISTER)

***** NOTE: If the failure was a \$02, (interrupt not found the index register X should contain one of the following codes:

- \$00..... IONMI was the interrupt
- \$01..... ACIA was the interrupt
- \$02..... CA2 from 6522.E was the interrupt
- \$03..... CA1 from 6522.E was the interrupt
- \$04..... shift register from 6522.E was the interrupt
- \$05..... CB2 from 6522 .E was the interrupt
- \$06..... CB1 from 6522 .E was the interrupt
- \$07..... Timer 2 from 6522> e was the interrupt
- \$08..... Timer 1 from 6522. E was the interrupt
- \$09..... CA2, 6522.D
- \$0A..... CA1, 6522.D (ANY SLOT but no slot found)
- \$0B..... Shift register, 6522.D
- \$0C..... CB2, 6522.D
- \$0D..... CB1, 6522.D
- \$0E..... Timer 2, 6522.D
- \$0F..... Timer 1, 6522.D



```

$10..... >>>INTERRUPT NOT FOUND <<<
$11..... SLOT 1 was the interrupt
$12..... Slot 2 was the interrupt
$13..... Slot 3 was the interrupt
$14..... Slot 4 was the interrupt
Keyboard:
C000 - "KA" Port
Bit   7     6     5     4     3     2     1     0
      drdy  d6    d5    d4    d3    d2    d1    d0
C008 - "KB" port
      7     6     5     4     3     2     1     0
      d7    kybd  A2    A1    alk   ctrl  sft   anyky
A2 is solid apple switch
A1 is open apple switch
alk ia alpha-lock
ctrl is control
sft is shift key (either one)
C010 - keyboard reset
Speaker
C030 - toggle speaker (same as Apple ){
C040 - Hardware bell (one beep)
FFEO - Bit 0 - 5 Apple /// sound (D-A)
Screen Control
C050 - C057 (see sheet 4 of schematic)
C050,C051 - TEXT mode
C052,C053 - MIX mode
C054,C055 - PAGE 2 mode
C056,C057 - HIRES mode
Joysticks, switches, and printer port
C058 - C05F (se sheet 4 of schematic)
C058,C059 - PDLO, Address 0 of A/D
(also disable/enable output handshake)
C05E, C05F - AXCO, Address 1 of A/D
(output handshake line false/true)
C05A,C05B - PDL2, Address 2 of A/D
C05C,C05D - A/D ramp start
(NOTE: To read a particular joystick pot set correct address
as follows:      A3   A2   A1   <---Address lines
joystick #1 (X0) 0    0    1
"              (Y0) 0    1    0
joystick #2 (X1) 0    1    1
"              (Y1) 1    0    0

Then use C05D to enable the RAMP of the A/D.
Bit 7 of C060 is switch 0
      C061 is switch 1
      C062 is switch 2
      C063 is switch 3
      C066 is the joystick timeout (selected above)
CODC, CODD is ENSEL (direction of CBI in 6522. D, SCO)
CODE, CODF is ENSIO (serial data R/W in 6522. D, SER)
C064, Bit 7 is IRQ3 and
C065, Bit 7 is IRQ4
C09X is Device select for Slot 1
    
```



COAX is Device select for Slot 2
 COBX is Device select for Slot 3
 COCX is Device select for Slot 4
 C1XX is IO select for Slot 1
 C2XX is IO select for Slot 2
 C3XX is IO select for Slot 3
 C4XX is IO select for Slot 4
 CODA disables Character Generator RAM write, and
 CODB enables Character Generator write.
 COD8 disables hire scroll, and
 COD9 enables hires scroll. If enables, then
 COE0, COE1 is set/clear address VAl
 COE2, COE3 is set/clear address VB1
 COE4, COE5 is set/clear address VC1
 VC1 VB1 VAl results in...

0	0	0	no scroll
0	0	1	1 horizontal line wrap
0	1	0	2 horizontal line wrap
0	1	1	3 horizontal line wrap
1	0	0	4 horizontal line wrap
1	0	1	5 horizontal line wrap
1	1	0	6 horizontal line wrap
1	1	1	7 horizontal line wrap

Disk drive Addresses
 COD0, COD1 clear/set external drive address A0
 COD2, COD3 clear/set external drive address A1
 COD4, COD5 enable, external drive power
 COD6, COD7 Side 1/Side 2 signal. If COD4 (enable), then
 A0 A1 Results in...

0	0	no external drive
0	1	external drive #1
1	0	external dirve #2
1	1	external drive #3

COE0, COE1 disk phase 0 set/clear
 COE2, COE3 disk phase 1 set/clear
 COE4, COE5 disk phase 3 set/clear
 COE6, COE7 disk phase 3 set/clear
 COE8, COE9 drive motor disable/enable (begins time out)
 COEA, COEB select internal/external drive
 RS-232 Port, ACIA, (6551)
 COF0 is received or transmitted data
 COF1 Writing any data causes a programmed reset, while Reading the following:

Bit 7	6	5	4	3	2	1	0
IRQ	NOT DSR	NOT DCD	TDRO	RDRf1	OVERR	FRMERR	PARERR

COF2 is the command register:
 Bit 7-5 are parity check controls
 Bit 4 is ECHO control
 Bit 3-2 are transmit controls
 Bit 1 is INT
 Bit 0 is DTR
 COF3 is the control register:
 Bit 7 is STOPB



Bit 6-5 are word length
 Bit 4 is NOT KCLOCK
 Bit 3-0 are the BAUD rate
 VIA 6522 system control registers
 FFD0 - Zero Page Register (Z register)
 FFDF - Environment register (E register)
 Bit 7 6 5 4 3 2 1 0
 1 MHz IOEN SCRN RSTEN WPROT PRSTK ROM 1 ROMEN
 FFD2 - data direction register B
 FFD3 - Data direction register A
 FFD4 - FFD7 Timer 1
 FFD8 - FFD9 Timer 2 (used by printer port)
 FFDA - Shift register (used by printer port)
 CA1 (in) Anyslot IRQ (will not clear)
 CA2 (in) printer port input handshake
 CB1 (out) Printer port clock
 CB2 (out) Printer port serial data
 CB2 (out) joystick address 0 set/clear
 FFEO - interrupt flags in / sound out
 Bit 7 - IONMI (in)
 Bit 6 - IOCT (in)
 Bit 5-0 interrupt flags in / bank register (in/out)
 Bit 7 - NOT GIRQ
 Bit 6 - Not A }{SW
 Bit 5 - Not IRQ2
 Bit 4 - Not IRQ1
 Bits 3-0 - Bank register (B register)
 FFE2 - Data direction register B
 FFE3 - Data direction register A
 FFE4 - FFE7 Timer 1
 FFE8 - FFE9 Timer 2 (input to IOCT flag)
 FFEA - Shift register (used for VBL)
 CA1 (in) clock/calendar IRQ
 CA2 (in) keyboard IRQ
 CB1, CB2 (out) vertical blanking
 List of interrupt flag location:
 C000 - Bit 7 = Keyboard
 C064 - Bit 7 = Slot 3
 C065 - Bit 7 = Slot 4
 C070 - = real time clock (function in Z reg)
 FFDD - Bit 0 = CA2, printer port input handshake
 Bit 1 = CA1, anyslot IRQ
 Bit 2 = shift register
 Bit 3 = CB2
 Bit 4 = CB1
 Bit 5 = Timer 2
 Bit 6 = Timer 1
 Bit 7 = IRQ, any of the above 7 IRQ's
 FFEO - Bit 6 = IOCT
 Bit 7 = global IRQ, any IRQ in the system
 Bit 5 = Slot 2
 Bit 4 = Slot 1
 FFED - Bit 0 = CA2, keyboard
 Bit 1 = CA1



Bit 2 = Shift register
Bit 3 = CB2, VBL x 8
Bit 4 = CB1, VBL x 1
Bit 5 = timer 2, IOCT (slot)
Bit 6 = Timer 1
Bit 7 = IRQ< any of the above 7 IRQ's

Verify the problem found. Whenever it is practical always try to replace the problem that you removed to see if the same symptom previously encountered returns. If you have indeed, found the correct problem, the same symptom will return. Otherwise, the problem probably still exists and further troubleshooting is called for.

Verify the fix. Once you are confident that you have repaired a problem, RETEST the ENTIRE system using ALL of the methods, described above in section 4.0. If no further problems are encountered log and label the board as repaired, making sure that ALL documentation, including SYMPTON, SOLUTION, and METHOD of fault isolation, (especially if it was a tricky one), are included. If you DO find another problem, then RESTART the diagnostic procedure and attempt to isolate and repair the NEW problem that you have encountered. Repeat the entire process as necessary, until ALL problems have been identified and corrected.



APPLE /// COMMON FAILURES (FOR SHOTGUNNING)

<u>SYMPTOM</u>	<u>PROBABLE CAUSE</u>
SELF TEST (F6E6G)	
Self-Test but characters wrong	74S257 (U66 [C12], U69 [C13]) 74LS374 (U4, [M13])
Self-Test with ACIA error message	6551 (U98 [B10])
No-Reset	B8, D13
A/// FINAL TEST: PART A	
No-Boot:	<i>CAS12S C13(12V) Prom 42 [M14], D3, G13, G11</i> <i>CASE256 C13 (5V) 68</i>
Video:	
Wrong colors	C5 100 pf
Random Patterns	B8, F10, G11
Sound:	
No sound or weak sound	LM380 (U103 [14])
Sound does't sweep	6522 (U97 [G10])
Serial Port:	
Fails	6551 (U98 [B10])
Paddle Port:	
Switch Fails	74LS251 (U101 [J11])
Paddle Fails	9708 (U105 [L5])
ROM:	
Fails:	341-0031 (U64 [D10])
A/// FINAL TEST: PART B	
Printer Port	6522 (U73 [H10]) 74LS125 (U160 [M4]), 74LS126 (U161 [I7])

This by no means exhaust the possible failures. Try to use your technical ability to find out what's wrong.

REPAIR MANUAL
 APPLE COMPUTER INC.
 VERMONT

*Bob Edgington
10-24-80*

RAM FAILURE MAP

8	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
6	B9	B8	B7	B6	B5	B4	B3	B2
5	B17	B16	B15	B14	B13	B12	B11	B10
4	C17	C16	C15	C14	C13	C12	C11	C10
3	D9	D8	D7	D6	D5	D4	D3	D2
2	D17	D16	D15	D14	D13	D12	D11	D10
1	C9	C8	C7	C6	C5	C4	C3	C2

96K

1 2 3 4 5 6 7 8

8	B9	B8	B7	B6	B5	B4	B3	B2
7	B17	B16	B15	B14	B13	B12	B11	B10
6	B9	B8	B7	B6	B5	B4	B3	B2
5	B17	B16	B15	B14	B13	B12	B11	B10
4	C17	C16	C15	C14	C13	C12	C11	C10
3	D9	D8	D7	D6	D5	D4	D3	D2
2	D17	D16	D15	D14	D13	D12	D11	D10
1	C9	C8	C7	C6	C5	C4	C3	C2

THE RAM FAILURE MAP INDICATES THE PHYSICAL LOCATION (COORDINATES OF THE RAM FAILURE. FOR EXAMPLE, IF WE OBSERVE INVERSED 1'S FOR THE LOWER ROW THIS WOULD CORRESPOND TO A RAM(S) OR RAM ADDRESS FAILURE AT LOCATIONS C9-C10 ON THE MEMORY BOARD.

128K

RAM FAILURES ARE INDICATED BY INVERSE 1'S
 RAS OR CAS FAILURES ARE DISPLAYED AS A FAILED ROW (INVERSE 1'S)
 INDIVIDUAL RAM FAILURE (INVERSE 1'S) CAN BE DUE TO RAM OR RAM ADDRESSING

6/30/81

All ROM/PROM 1.1.FD

LOC	ALL PIN	DESCRIPTION	SECTION	GENERIC PART	CHECK COLUMN
G9	341-0030 * 342-0145A	ROM, SYNCHROM ROM, SYNCHROM (INTERFACE HARDWARE)	TIMING	① ①-2	2E85
B9	341-0031	ROM, BOOT/MONITOR B&T		①-2	29EA
G5	341-0032	ROM, VIDEO CONTROL	VIDEO	①-2	5230
C13	341-0042	PROM, CAS128 (-12V)	ADDR LOGIC	②	
	* 342-0063	PROM, CAS256 (+5V)		②-7	
C11	341-0044	PROM, RAS65 (-12V)	ADDR LOGIC	②	
	* 342-0061	PROM, RAS65 (+5V)		②-7	
FS	341-0045	PROM, 1024 X4	I/O LOGIC	②	
K11	342-0028	PROM, STATE MACHINE P6A	DISK I/O		
C10	342-0043	PROM, 1024 X4	RAM ADDR	②	
F7	342-0046	PROM, 1024 X4	TIMING LOGIC	②-4	
F9	342-0055	PROM, 1024 X4	RAM ADDR	②	
C12	342-0056	PROM, CAS65 ADDRESSING	RAM ADDR	②-4	

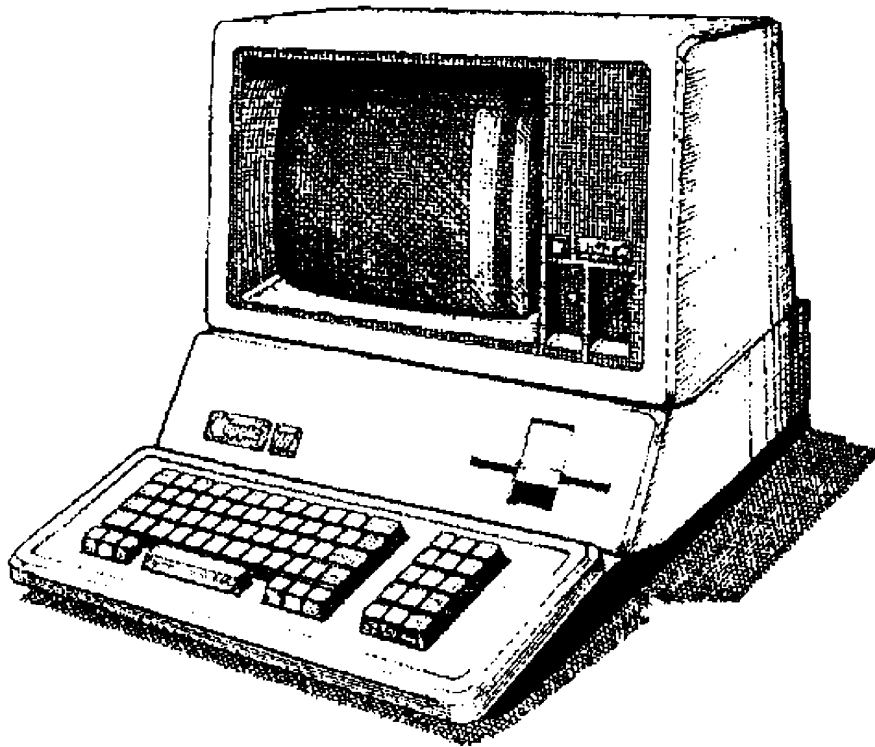
① -1D2716 (NFC)
-3AM12732DC CAMD

② 76A13
-1. TRP24841 (T.I.)
-2. HM 7643A (HARRIS)
-3. DM 748503 (CUMPT)
-4. 82S137AN
-5. AM127X23DC (CAMD)
-6. TBP24841 (T.I.)
-7. 82S137AN



Apple /// Computer Information

Apple /// Service Reference Manual



Section II of II • Servicing Information

Chapter 14 • Parts Layout and Parts List

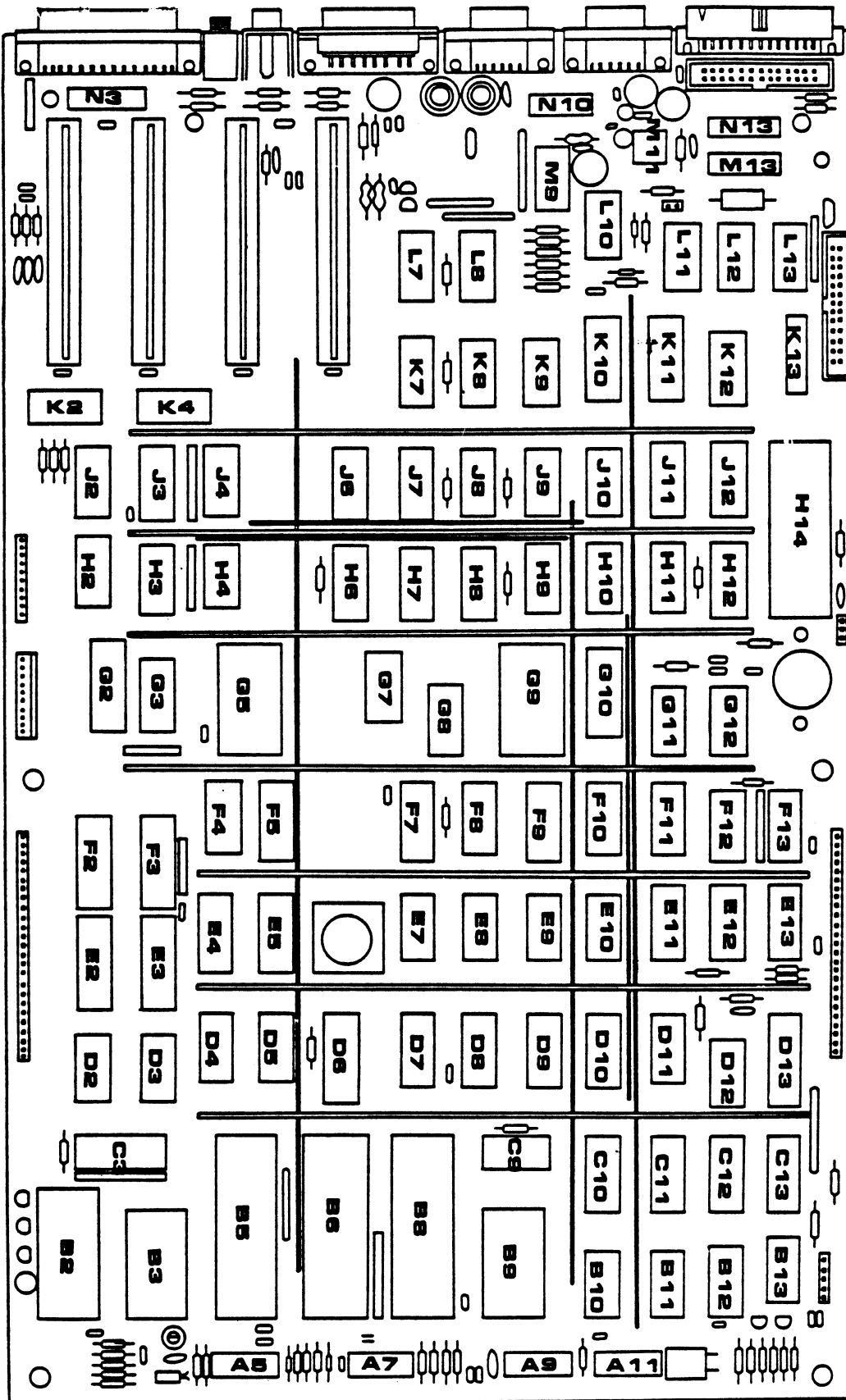
Written by Apple Computer • 1982

APPLE /// IC PARTS BY LOCATION

LOCATION	DESCRIPTION	APPLE PART NUMBER
A05	IC, 556 DUAL TIMER	330-0556
A07	IC, 556 DUAL TIMER	330-0556
A09	IC, 74LS399, QUAD 2 INPUT MUX	306-0399
A11	IC, 74S74, DUAL D-TYPE FLIP-FLOP	308-0074
B02	IC, ACIA 6551A	338-0002
B03	IC, MICROPROCESSOR CLOCK 58167	331-8167
B05	IC, 6522 VERSATILE INTERFACE ADAPTER	338-6522
B06	IC, 6522 VERSATILE INTERFACE ADAPTER	338-6522
B08	IC, MICRO 6502B 3MHZ TEST & BURN-IN	369-6502
B09	IC, ROM BOOT/MONITOR B & T	342-0031
B10	IC, 74LS20, DUAL 4-INPUT NAND	306-0020
B11	IC, 74LS51, AND OR INVERT	306-0051
B12	IC, 74S175, QUAD D-TYPE FLIP FLOP	308-0175
B13	IC, 74S86, QUAD 2 INPUT EXCLUSIVE-OR	308-0086
C03	IC, 8304B 8-BIT TRI-STATE	316-8304
C09	IC, 74LS260, DUAL 5 INPUT NOR	306-0260
C10	IC, PROM 1024 X 4	342-0043
C11	IC, PROM RAS 65, 12 VOLT MEMORY BD	341-0044
	IC, PROM RAS65, 5 VOLT MEMORY	342-0061
C12	IC, PROM CASB65 ADDRESSING	342-0056
C13	IC, PROM CAS128, 12 VOLT MEMORY	341-0042
	IC, PROM CASB256, 5 VOLT MEMORY	342-0063
D02	IC, 74S257, QUAD DATA MULTIPLEXER	308-0257
D03	IC, 74S257, QUAD DATA MULTIPLEXER	308-0257
D04	IC, 74LS86, QUAD 2 INPUT EXCLUSIVE OR	306-0086
D05	IC, 74S74, DUAL D-TYPE FLIP-FLOP	308-0074
D06	IC, 74LS244, OCTAL BUFFERS/DRIVERS	306-0244
D07	IC, 74S257, QUAD DATA MULTIPLEXER	308-0257
D08	IC, 74S257, QUAD DATA MULTIPLEXER	308-0257
D09	IC, 74LS02, QUAD 2 INPUT NOR	306-0002
D10	IC, 74S195, 4 BIT PARALLEL SHIFT REG	308-0195
D11	IC, 74S74, DUAL D-TYPE FLIP-FLOP	308-0074
D12	IC, 74LS00, QUAD 2 INPUT NAND	306-0000
D13	IC, 74LS374, OCTAL D-TYPE FLIP-FLOP	306-0374
E02	IC, 74LS374, OCTAL D-TYPE FLIP-FLOP	306-0374
E03	IC, 74LS374, OCTAL D-TYPE FLIP-FLOP	306-0374
E04	IC, 1024 X 4 STATIC RAM 2114	334-0005
E05	IC, 1024 X 4 STATIC RAM 2114	334-0005
E07	IC, 74LS00, QUAD 2 INPUT NAND	306-0000
E08	IC, 74LS08, QUAD 2 INPUT AND	306-0008
E09	IC, 74S10, TRIPLE 3 INPUT NAND	306-0010
E10	IC, 74S86, QUAD 2 INPUT EXCLUSIVE-OR	308-0086
E11	IC, 74LS283, 4 BIT BINARY ADDER	306-0283
E12	IC, 74S153, DUAL 4 TO 1 LINE MUX	308-0153
E13	IC, 74S153, DUAL 4 TO 1 DATA MUX	306-0153
F02	IC, 74S374, OCTAL D-TYPE FLIP-FLOP	308-0374
F03	IC, 74S374, OCTAL D-TYPE FLIP-FLOP	308-0374
F04	IC, 74166, 8 BIT SHIFT REGISTER	302-0166
F05	IC, IC PROM 1024 X 4	342-0045
F07	IC, PROM 1024 X 4	342-0046
F08	IC, 74LS08, QUAD 2 INPUT AND	306-0008
F09	IC, PROM, 1024 X 4	342-0055
F10	IC, 74LS161, SYNC BINARY 4 BIT COUNTER	306-0161

APPLE /// IC PARTS BY LOCATION

LOCATION	DESCRIPTION	APPLE PART NUMBER
F11	IC, 74LS161, SYNC BINARY 4 BIT COUNTER	306-0161
F12	IC, 74S153, DUAL 4 TO 1 LINE MUX	308-0153
F13	IC, 74S153, DUAL 4 TO 1 MULTIPLEXER	306-0153
G02	IC, 74S374, OCTAL D-TYPE FLIP-FLOP	308-0374
G03	IC, 74LS157, QUAD 2 TO 1 DATA MUX	306-0157
G05	IC, VIDEO CONTROL ROM	342-0032
G07	IC, 74LS133, 13 INPUT NAND	306-0133
G08	IC, 74LS139, 2 TO 4 LINE DECODERS	306-0139
G09	IC, ROM, SYNCHROM	342-0030
G10	IC, 74LS374, OCTAL D-TYPE FLIP-FLOP	306-0374
G11	IC, 74LS161, SYNC BINARY 4 BIT COUNTER	306-0161
G12	IC, 74LS161, SYNC BINARY 4 BIT COUNTER	306-0161
H02	IC, 74LS00, QUAD 2 INPUT NAND	306-0000
H03	IC, 74LS399, QUAD 2 INPUT MUX	306-0399
H04	IC, 74LS51, AND OR INVERT	306-0051
H06	IC, 9334, TESTED & BURN-IN	302-9334
H07	IC, 74LS74, DUAL D FLIP-FLOPS	306-0074
H08	IC, 74LS04, HEX INVERTERS	306-0004
H09	IC, 74LS05, OPEN COL HEX INV	306-0005
H10	IC, 74LS132, QUAD 2 INP NAND SCH TRIGG	306-0132
H11	IC, 74LS74, DUAL D-TYPE FLIP-FLOP	305-0074
H12	IC, 74LS257, QUAD DATA MUX	306-0257
H14	IC, ROM KEYBOARD ENCODER A3	342-0035
J02	IC, 1488 QUAD LINE DRIVER	360-1488
J03	<i>U136</i> IC, 74LS151, 1 OF 8 DATA MUX	308-0151
J04	IC, 74LS21, DUAL 4 INPUT AND	306-0021
J06	IC, 74LS138, 3-TO-8 LINE DECODERS	306-0138
J07	IC, 74LS133, 13 INPUT NAND	306-0133
J08	IC, 74LS32, QUAD 2 INPUT OR	306-0032
J09	IC, 74LS125, QUAD TRI-STATE BUFFERS	306-0125
J10	IC, 74LS126, QUAD TRI-STATE BUFFERS	306-0126
J11	IC, 74LS139, 2 TO 4 LINE DECODERS	306-0139
J12	IC, 74LS257, QUAD DATA MUX	306-0257
K02	IC, 1489 LINE RECEIVER	360-1489
K04	IC, 74LS138, 3-TO-8 LINE DECODER	306-0138
K07	IC, 74LS138, 3-TO-8 LINE DECODERS	306-0138
	IC, 74LS374, OCTAL D-TYPE FLIP-FLOP	306-0374
K08	IC, 74LS11, TRIPLE 3 INPUT AND	306-0011
K09	IC, 74LS04, HEX INVERTERS	306-0004
K10	IC, 74LS323, 8 BIT BIDIRECT SHIFT REG	306-0323
K11	IC, PROM, STATE MACHINE P6A	342-0028
K12	IC, 74LS174, HEX D-TYPE FLIP-FLOP	306-0174
K13	RESISTOR ARRAY, 47 OHMS	112-0102
L07	IC, 74LS251, IC, DATA MULTIPLEXER	306-0251
L08	IC, 74S153, DUAL 4 TO 1 LINE MUX	308-0153
L10	IC, 556 DUAL TIMER	330-0556
L12	IC, 9334,	302-9334
L13	IC, 9334	302-9334
M09	IC, 9708, 6 CHANNEL 8 BIT A/D	356-9708
M11	IC, LM380 AUDIO POWER AMPLIFIER	354-0380
M13	RESISTOR ARRAY, 47 OHMS	112-0102
N03	RESISTOR ARRAY, 47 OHMS	112-0102
N10	RESISTOR ARRAY, 47 OHM	112-0102
N13	RESISTOR ARRAY 47 OHM	112-0102



14.3

BIL VERSION 29-Mar-79 PAGE 1

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:26

EFFECTIVITY DATE: ALL

INDENTED BILLOF MATERIAL

PARENT PART: 610-8156

INACT FO, 256K APPLE III
 ERC: B SRCE CODE: A TYPE: *

UM: EA
 ABC: A
 ECN: A116 ASSEMBLY QTY: 1
 DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR CD	BT CY	SA YP	AP BL	UM	EXTENDED QTY PER	ECN CHG	START DATE	B/N	CLOSE DATE	S/N
2	610-8128D	H INACT. DMG ASSY, FINAL 128K A3	D	X	7	P	U EA	0	A116	18-Jan-82		08-Jul-82	
2	030-0102	E INACT. PACKING LST, A3 SYSTEM BOX	EX	X	7	P	C P EA	1	A116	18-Jan-82		08-Jul-82	
2	030-0103	C INACT. UNPACKING INST A3 SYS BOX	EX	X	7	P	C P EA	1	A116	18-Jan-82		08-Jul-82	
2	030-0104	C SYSTEM REGISTRATION CARD A2/A3	EX	P	7	P	C P EA	1	A001	18-Jan-82		08-Jul-82	
2	590-0003	C CABLE, A. C. POWER CORD	2M	P	1	P	C P EA	1	C-10	18-Jan-82		08-Jul-82	
2	590-0025	C CABLE, MONITOR	RC	P	1	P	C P EA	1	441	18-Jan-82		08-Jul-82	
2	610-0006	A INACT. ASSY, FOAM CAP SIDE A RIGHT	EX	X	7	P	C P EA	1	A116	18-Jan-82		08-Jul-82	
3	944-0052D	O ASSY DMG, FOAM INSIDE/OUTSIDE	EX	D	7	B	* U EA	0	13-Oct-80				
3	944-0052	A FOAM, ETHER DIE-CUT OUTSIDE	EX	X	7	P	D P EA	1	473	13-Oct-80			
3	944-0063	O FOAM, ETHER KNOCK-OUT INSIDE	EX	X	7	P	C P EA	1	473	13-Oct-80			
3	942-0091	A CORRUGATED PANEL, END CAP	EX	X	7	P	C P EA	1	473	13-Oct-80			
2	610-0007	A INACT. ASSY, FOAM CAP SIDE B LEFT	EX	X	7	P	C P EA	1	A116	18-Jan-82		08-Jul-82	
2	944-0062D	O ASSY DMG, FOAM INSIDE/OUTSIDE	EX	D	7	B	* U EA	0	13-Oct-80				
3	944-0062	A FOAM, ETHER DIE-CUT OUTSIDE	EX	X	7	P	D P EA	1	473	13-Oct-80			
3	944-0063	O FOAM, ETHER KNOCK-OUT INSIDE	EX	X	7	P	C P EA	1	473	13-Oct-80			
3	944-0063	A CORRUGATED PANEL, END CAP	EX	X	7	P	C P EA	1	473	13-Oct-80			
3	944-0062	A INACT. ASSY, FOAM CORRUGATED PAD	EX	X	7	P	C P EA	1	A116	18-Jan-82		08-Jul-82	
3	944-0063	H INACT. ASSY, 256K A3	MA	D	7	B	* U EA	0	A024	18-Jan-82		08-Jul-82	
3	944-0063	N DMG ASSY, CHASSIS 128K A3	D	X	7	P	C P EA	1	A024	18-Jan-82		08-Jul-82	
3	944-0063	A SUBASSY, TESTED, KEYBOARD A3	MS	A	1	A	P EA	1	948	21-Jan-82		08-Jul-82	
3	944-0063	A ASSY, UNTST, KEYBOARD A3, ALPS	A	X	7	P	C P EA	1	948	21-Jan-82		08-Jul-82	
3	944-0063	A DMG, UNTST, KEYBOARD A3, ALPS	A	X	7	P	C P EA	1	948	21-Jan-82		08-Jul-82	
3	944-0063	C SPECIFICATION, KEYBOARD A3	EX	D	7	B	* U EA	0	948	21-Jan-82		08-Jul-82	
3	944-0063	A SWITCH, KBD KCC ALPS	RP	P	1	P	C P EA	67	362	21-Jan-82		08-Jul-82	
3	944-0063	A SWITCH, ALB KCC ALT ACTION	P	*	*	* P EA	4	948	21-Jan-82		08-Jul-82		
3	944-0063	A SWITCH, ALPS KFF, DOUBLE ACTION	P	*	*	* P EA	4	948	21-Jan-82		08-Jul-82		
3	944-0063	A LAMP, BI-PIN 5V .115A	RP	P	1	P	C P EA	1	362	21-Jan-82		08-Jul-82	
3	944-0063	MA PIN, LAMP	RP	P	1	P	C P EA	2	362	21-Jan-82		08-Jul-82	
3	944-0063	A PLATE, KEYBOARD A3 ALPS	P	*	* P EA	1	948	21-Jan-82	948	21-Jan-82		08-Jul-82	
3	944-0063	A PCB, MAIN & PIGGYBACK, KBD, A3	P	*	* P EA	1	948	21-Jan-82	948	21-Jan-82		08-Jul-82	
3	944-0063	A CRANK, KEYBOARD A3 ALPS	P	*	* P EA	1	948	21-Jan-82	948	21-Jan-82		08-Jul-82	
3	944-0063	A CRANK, GUIDE, STRAIGHT, ALPS	P	*	* P EA	1	948	21-Jan-82	948	21-Jan-82		08-Jul-82	
3	944-0063	A ADAPTER, STRAIGHT KEYCAP	P	*	* P EA	1	948	21-Jan-82	948	21-Jan-82		08-Jul-82	
3	944-0063	A SWITCH, END ACTUATED PUSH BUTTON	P	*	* P EA	1	948	21-Jan-82	948	21-Jan-82		08-Jul-82	
3	944-0063	A CABLE ASSY, KEYBOARD A3	RP	P	1	P	C P EA	1	P190	21-Jan-82		08-Jul-82	
3	944-0063	B PIVOT SPACEBAR CRANK KCC	RM	P	1	P	C P EA	2	948	21-Jan-82		08-Jul-82	
3	944-0063	O CONN, 26 PIN HEADER	RP	P	1	P	C P EA	2	947	21-Jan-82		08-Jul-82	
3	944-0063	B SCREW, 6-32 3/8 CRPHD	EX	X	1	B	C U EA	5	CO86	21-Jan-82		08-Jul-82	
3	944-0063	O SPACER, RESET SWITCH	EX	X	1	B	C U EA	1	948	21-Jan-82		08-Jul-82	
3	944-0063	A SUBASSY, KEYCAPS, COMPLETE SET A3	MS	N	1	P	B P EA	1	441	05-Jun-80		08-Jul-82	
3	944-0063	O SPEC, APPLE III KEYCAPS	EX	D	7	B	* U EA	0	441	05-Jun-80		08-Jul-82	
3	944-0063	O KEY CAP, APPLE 3, / / 1	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, / / 2	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, # / 3	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, # / 4	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, # / 5	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, ^ / 6	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, & / 7	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		
3	944-0063	O KEY CAP, APPLE 3, * / 8	RM	P	1	* P EA	1	441	05-Jun-80		08-Jul-82		

MASTERS

BIL VERSION 29-Mar-79 PAGE 2

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:27

INDENTED BILLS OF MATERIAL

INACT.FG.256K APPLE III
 ERC: B SRCE CODE: A TYPE: *
 ECN: A116 ASSEMBLY QTY: 1
 DATE: 08-Jul-82

UM: EA
 ABC: A
 PROD CODE: U
 PLAN CODE: U

PARENT PART: 610-8156

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	S	T	S	A	P	UM	EXTENDED QTY PER	ECN	CHG	DATE	START DATE	S/N	DATE	CLOSE	S/N
6	815-0097	0 KEY CAP, APPLE 3, (/ 9	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0098	0 KEY CAP, APPLE 3,) / 0	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0099	0 KEY CAP, APPLE 3, + / -	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0100	0 KEY CAP, APPLE 3, - / +	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0101	0 KEY CAP, APPLE 3, : / \	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0102	0 KEY CAP, APPLE 3, 7	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0103	0 KEY CAP, APPLE 3, 8	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0104	0 KEY CAP, APPLE 3, 9	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0105	0 KEY CAP, APPLE 3, TAB	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0106	0 KEY CAP, APPLE 3, 0	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0107	0 KEY CAP, APPLE 3, M	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0108	0 KEY CAP, APPLE 3, E	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0109	0 KEY CAP, APPLE 3, R	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0110	0 KEY CAP, APPLE 3, T	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0111	0 KEY CAP, APPLE 3, Y	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0112	0 KEY CAP, APPLE 3, U	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0113	0 KEY CAP, APPLE 3, I	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0114	0 KEY CAP, APPLE 3, O	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0115	0 KEY CAP, APPLE 3, P	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0116	0 KEY CAP, APPLES, { / [RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0117	0 KEY CAP, APPLE 3, } /]	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0118	0 KEY CAP, APPLE 3, ~ / `	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0119	0 KEY CAP, APPLE 3, 4	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0120	0 KEY CAP, APPLE 3, 5 (W/SENSOR)	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0121	0 KEY CAP, APPLE 3, 6	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0122	0 KEY CAP, APPLE 3, CONTROL	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0123	0 KEY CAP, APPLE 3, A	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0124	0 KEY CAP, APPLE 3, B	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0125	0 KEY CAP, APPLE 3, D (W SENSOR)	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0126	0 KEY CAP, APPLE 3, F	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0127	0 KEY CAP, APPLE 3, 0	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0128	0 KEY CAP, APPLE 3, H	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0129	0 KEY CAP, APPLE 3, J	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0130	0 KEY CAP, APPLE 3, K (W SENSOR)	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0131	0 KEY CAP, APPLE 3, L	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0132	0 KEY CAP, APPLE 3, ; / ,	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0133	0 KEY CAP, APPLE 3, " / '	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0134	0 KEY CAP, APPLE 3, RETURN	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0135	0 KEY CAP, APPLE 3, 1	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0136	0 KEY CAP, APPLE 3, 2	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0137	0 KEY CAP, APPLE 3, 3	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0138	0 KEY CAP, APPLE 3, SHIFT	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0139	0 KEY CAP, APPLE 3, X	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0140	0 KEY CAP, APPLE 3, Z	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0141	0 KEY CAP, APPLE 3, C	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0142	0 KEY CAP, APPLE 3, V	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0143	0 KEY CAP, APPLE 3, B	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0144	0 KEY CAP, APPLE 3, N	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				
6	815-0145	0 KEY CAP, APPLE 3, M	RM	P	1				* P EA	1	441		05-Jun-80	05-Jun-80				

BIL VERSION 29-Mar-79 PAGE 3

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:29
EFFECTIVITY DATE: ALL

INDENTED BILB OF MATERIAL

PARENT PART: 610-8156

INACT.FO.256K APPLE III
ERC: B SRCE CODE: A TYPE: *
UM: EA
ABC: A
PROD CODE: U
PLAN CODE: U
ECN: A116 ASSEMBLY QTY: 1
DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	ST	BT	AP	BL	UM	EXTENDED QTY PER	ECN	CHQ	START DATE	B/N	CLOSE DATE	S/N
6	815-0147	0 KEY CAP, APPLE 3, > / /	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0148	0 KEY CAP, APPLE 3, ? / /	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0149	0 KEY CAP, APPLE 3, SHIFT	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0150	0 KEY CAP, APPLE 3, ARROW UP	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0151	0 KEY CAP, APPLE 3, 0 (ZERO)	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0152	0 KEY CAP, APPLE 3, . (DECIMAL)	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0153	0 KEY CAP, APPLE 3, ALPHA LOCK	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0154	0 KEY CAP, APPLE 3, APPLE 1	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0155	0 KEY CAP, APPLE 3, APPLE 2	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0156	0 KEY CAP, APPLE 3, SPACE BAR	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0157	0 KEY CAP, APPLE 3, ARROW LEFT	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0158	0 KEY CAP, APPLE 3, ARROW RIGHT	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0159	0 KEY CAP, APPLE 3, ARROW DOWN	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0160	0 KEY CAP, APPLE 3, - (MINUS)	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0161	0 KEY CAP, APPLE 3, ENTER	RM	P	1	*	P	EA	1	441		05-Jun-80			
6	815-0163	3 REBET KEY, APPLE III	RM	P	1	*	P	EA	1	427		05-Jun-80			
6	815-0179	1 SPACER, RESET SWITCH A3	RM	P	1	P	C	EA	1	427		05-Jun-80			
6	062-0032D	0 DMG, KEY CAP 1 SPACE 17 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0033D	0 DMG, KEY CAP 1 SPACE	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0034D	0 DMG, KEY CAP 1 SPACE 10 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0036D	0 DMG, KEY CAP 1 SPACE W/BUMP	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0037D	0 DMG, KEY CAP 1 SPACE W/BUMP 5 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0038D	0 DMG, KEY CAP 1-1/2 SPACE 17 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0039D	0 DMG, KEY CAP 1-1/4 SPACE 17 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0040D	0 DMG, KEY CAP 1-1/2 SPACE 5 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0041D	0 DMG, KEY CAP 1-3/4 SPACE 10 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0042D	0 DMG, KEY CAP 2 SPACE 10 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0043D	0 DMG, KEY CAP 2 SPACE 17 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0044D	0 DMG, KEY CAP 2-1/4 SPACE 17 DEG	EX	D	7	8	*U	EA	0	441		05-Jun-80			
6	062-0049D	0 DMG, KEY CAP 6-1/2 SPACE	EX	D	7	8	*U	EA	0	441		05-Jun-80			
4	063-0079	0 A3, KEYBOARD TEST AND INSPECTION	RM	X	1	P	C	EA	1	M281		21-Jan-82			
3	600-0009	B ASSY, SPEAKER	EX	P	7	P	B	EA	1	491		18-Jan-82			
4	731-0001	B WIRE, BLACK, 24 AWG, 7 X 32	EX	X	1	P	C	FT	1	491		15-Apr-82			
4	561-2400	A HOUSING, CONN SKT, CRIMP, 2 PIN	RP	P	1	*	P	EA	1	0.0057		15-Apr-82			
4	520-0100	B CONNECTOR, CONTACT CRIMP TYPE	RP	P	1	P	C	EA	2	491		15-Apr-82			
4	517-0002	B SBABY, DISK 3, A3 W/ANALOG BD, INT	WS	P	1	P	A	EA	1	816		18-Jan-82			
3	650-5201	0 ASSY, PCB, TST, DISK ANALOG BD A3	TS	A	1	P	A	EA	1	631		05-Jun-80			
4	650-4201	D ASSY, PCB, UNTEST, DISK ANALOG BD A3	UN	A	1	P	B	EA	1	805		18-Mar-81			
5	650-0201	C SCHEMATIC, DISK ANALOG BD APPLE III	EX	D	7	P	D	EA	0	805					
6	050-0031	C PCB, DISK ANALOG BD APPLE III	RP	P	1	P	B	EA	1	805		12-May-80			
6	820-0024	0 RES 1/2W 5% 270 OHM	RP	P	1	P	C	EA	1	426		12-May-80			
6	101-2271	0 RES 1/2W 5% 330 OHM	RP	P	1	P	C	EA	1	426		12-May-80			
6	101-2331	0 RES 1/2W 5% 330 OHM	RP	P	1	P	C	EA	1	426		12-May-80			
6	101-2561	A RES 1/2W 5% 560 OHM	RP	P	1	P	C	EA	1	426		12-May-80			
6	101-4102	A RES 1/4W 5% 1K OHM	RP	P	1	P	C	EA	6	426		12-May-80			
6	101-4103	A RES 1/4W 5% 10K OHM	RP	P	1	P	C	EA	5	426		22-Jul-81			
6	101-4104	A RES 1/4W 5% 100K OHM	RP	P	1	P	C	EA	2	426		12-May-80			
6	101-4153	A RES 1/4W 5% 15K OHM	RP	P	1	P	C	EA	2	426		22-Jul-81			
6	101-4202	A RES 1/4W 5% 2K OHM	RP	P	1	P	C	EA	1	426		12-May-80			

PRINTED 23-Jul-82 13:30

PERSONAL COMPUTER SYSTEMS

INDENTED BILLS OF MATERIAL

BILL VERSION 29-Mar-79 PAGE 4

EFFECTIVITY DATE: ALL PARENT PART: 610-8156 INACT.FO.256K APPLE III UM: EA PROD CODE: U ECN: A116 ASSEMBLY QTY: 1
 SRC CODE: A TYPE: * SRC CODE: A TYPE: * ABC: A PLAN CODE: U DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	S	T	B	A	P	UM	EXTENDED QTY PER	ECN CHG	START DATE	S/N	CLOSE DATE	S/N
6	101-4302	A RES 1/4W 5% 3K OHM	RP	P	1	P	C	P	EA	2	426	12-May-80			
6	101-4472	A RES 1/4W 5% 4.7K OHM	RP	P	1	P	C	P	EA	1	426	12-May-80			
6	101-4910	A RES 1/4W 5% 91 OHM	RP	P	1	P	C	P	EA	1	426	12-May-80			
6	101-4822	A RES 1/4W 5% 8.2K OHM	RP	P	1	P	C	P	EA	1	C127	23-Jul-81			
6	108-1372	O RES 1/4W 1% 137 OHM	RP	P	1	P	C	P	EA	2	442	12-May-80			
6	108-5762	O RES 1/4W 1% 576 OHM	RP	P	1	P	C	P	EA	1	442	12-May-80			
6	108-6342	O RES 1/4W 1% 634 OHM	RP	P	1	P	C	P	EA	1	442	12-May-80			
6	108-8872	O RES 1/4W 1% 887 OHM	RP	P	1	P	C	P	EA	2	442	12-May-80			
6	109-0003	O POT, TRIM 10K OHMS 20%	RP	P	1	P	C	P	EA	2	442	12-May-80			
6	120-0001	A CAP, 22uF 5% 100V	RP	P	1	P	C	P	EA	1	442	12-May-80			
6	125-3101	A CAP, 10uF 16V	RP	P	1	P	C	P	EA	1	433	12-May-80			
6	125-6401	A CAP, 220uF 16V	RP	P	1	P	C	P	EA	1	433	12-May-80			
6	125-6701	A CAP, 470uF 6.3V	RP	P	1	P	C	P	EA	1	433	12-May-80			
6	132-7601	O CAP, 3000pF 10% Z5F 50V	RP	P	1	P	C	P	EA	1	426	12-May-80			
6	132-8102	O CAP, .010uF 20% Z5U 50V	RP	P	1	P	C	P	EA	2	426	12-May-80			
6	135-9101	O CAP, 1uF +80-20% Z5U/Y5V 50V	RP	P	1	P	C	P	EA	11	433	22-Jul-81			
6	137-6101	O CAP, 100uF 5% 200PPH 500VDC	RP	P	1	P	C	P	EA	1	442	12-May-80			
6	137-6601	O CAP, 330uF 5% 200PPH 500VDC	RP	P	1	P	C	P	EA	2	442	12-May-80			
6	137-6701	O CAP, 510uF 5% 200PPH 500VDC	RP	P	1	P	C	P	EA	1	442	12-May-80			
6	151-5801	B CHOME, 68uH 10%	RP	P	1	P	B	P	EA	1	479	12-May-80			
6	151-6701	O CHOME, 470uH 10%	RP	P	1	P	B	P	EA	2	442	12-May-80			
6	305-0125	A IC, 74LS125	RI	P	1	P	B	P	EA	1	442	12-May-80			
6	327-2003	O IC, 2003A DARLINGTON TRANSISTOR ARRAY	RI	P	1	P	C	P	EA	1	449	12-May-80			
6	351-3146	O IC, 3146 HI VOLT. TRANSISTOR ARRAY	RP	P	1	P	C	P	EA	1	449	12-May-80			
6	359-3470	O IC, MC3470 FLOPPY DISK READ AMPLIF.	RI	P	1	P	B	P	EA	1	449	12-May-80			
6	371-4148	A DIODE, 1N4148	RP	P	1	P	C	P	EA	4	731	12-May-80			
6	372-3906	A TRANSISTOR, PNP SM, AMP, 2N3906	RP	P	1	P	C	P	EA	1	C026	12-May-80			
6	376-0001	A TRANSISTOR, MPB-U51	RP	P	1	P	C	P	EA	1	C111	12-May-80			
6	019-0080	O HEADER, 26 PIN RT. ANGLE W/D EARS	RP	P	1	P	C	P	EA	2	482	17-Nov-80			
6	511-1801	C SOCKET, IC 18 PIN	RP	P	1	P	C	P	EA	1	799	12-May-80			
6	511-1401	C SOCKET, IC 14 PIN	RP	P	1	P	C	P	EA	4	799	12-May-80			
6	511-1601	C SOCKET, IC 16 PIN	RP	P	1	P	C	P	EA	1	799	12-May-80			
6	515-0005	B PIN, SINGLE	RP	P	1	P	C	P	EA	9	C115	12-May-80			
6	515-0009	70 PIN CONN, 4 POS, APPLE SPEC.	RP	P	1	P	C	P	EA	1	NONE	12-May-80			
6	159-7103	B CHOME, COIL 28uH	RP	P	1	P	B	P	EA	2	C031	23-Feb-81			
6	371-4003	MA DIODE, 1N4003	RP	P	1	P	B	P	EA	2	C076	12-May-80			
6	305-0074	A IC, 74LS74	RI	P	1	P	B	P	EA	1	442	11-Mar-81			
6	101-4561	A RES 1/4W 5% 560 OHM	RP	P	1	P	C	P	EA	1	426	12-May-80			
6	515-0015	O HEADER, 26 PIN T&B #609-2627	RP	P	1	P	C	P	EA	0	SARA	12-May-80		17-Nov-80	
6	199-5602	A INDUCTOR FILTER 27 UH	RP	P	1	P	B	P	EA	2	DC44	12-May-80		23-Feb-81	
6	305-0032	A IC, 74LS32	RI	P	1	P	B	P	EA	1	442	11-Mar-81			
6	108-1274	O RES, 1/8W 1% 12.7K OHM	RP	P	1	P	C	P	EA	1	SARA	12-May-80		11-Mar-81	
6	109-0004	O POT, TRIM 20K OHMS 20%	RP	P	1	P	C	P	EA	1	442	11-Mar-81			
4	699-0032	O ASSY, DEM ALPS DISK DRIVE	VX	X	1	P	C	P	EA	1	441	09-Jun-80			
4	815-0162	E DOOR CLIP, DISK APPLEIII	RM	X	1	P	C	P	EA	1	657	05-Jun-80			
4	815-0164	ME DOOR, DISK APPLEIII	RM	X	1	P	C	P	EA	1	542	05-Jun-80			
4	815-0165	MH FRONT YDKE, DISK APPLE III	RM	X	1	P	C	P	EA	1	499	05-Jun-80			
4	517-0015	B CONNECTOR, BUTT CRIMP	RP	X	1	P	C	P	EA	2	766	05-Jun-80			
4	710-0004	MA LED, RED APPLE III DISK/CABLE	RP	X	1	P	C	P	EA	1	952	05-Jun-80			
4	815-0166	C SOCKET, FRONT MAT, 50P RECT, 1.5mm	RM	X	1	P	C	P	EA	1	145	05-Jun-80			

PRINTED 23-JUL-82 13:30
EFFECTIVITY DATE: ALL
PARENT PART: 610-8196

PERSONAL COMPUTER SYSTEMS
INDENTED BILLS OF MATERIAL

BIL VERSION 29-Mar-79 PAGE 5

INACT. FG, 256K APPLE III
ERC: B SRCE CODE: A TYPE: *
UM: EA
ABC: A

ECN: A116 ASSEMBLY QTY: 1
DATE: 08-JUL-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	BT	SA	P	L	UM	EXTENDED QTY PER	ECN	CHG	DATE	START DATE	S/N	CLOSE DATE	S/N
4	805-0035	Ø SHIELD, DRIVE APPLEIII	RM	P	1	P	C	P	EA	763		05-Jun-80				
4	410-1206	B SCREW, M2.5 X 0.45 X 6MM POZI DR	EX	X	1	P	C	P	EA	C086		28-Jan-81				
4	825-0084	O LABEL, SERIAL#, INT. DISK A3	EX	X	7	P	C	P	EA	441		09-Jun-80				
4	914-0028	A CARD, CARRIAGE LOCK	P						EA	816		15-Sep-81				
3	825-0094	C LABEL, APPLE LOGO NAMEPLATE	EX	X	7	P	C	P	EA	A035		18-Jan-82			09-May-82	
3	590-0017	C CABLE, POWER SUPPLY/MN BD APPLE III	RC	P	1	P	C	P	EA	A044		18-Jan-82				
4	517-0040	A CONN, RECEPT 10 PIN	RP	P	1	P	C	P	EA	559		19-Jan-81				
4	561-2209	O WIRE, #22 AWG STRANDED W/PVC INS.	EX	P	1				P	RL		19-Jan-81				
4	550-0107	B TUBING, HEAT SHRINK 3/8 ID WHITE	EX	P	7				P	FT		19-Jan-81				
4	561-2200	B WIRE, BLACK, 22 AWG, 7 X 30	EX	X	*				P	FT	0.8000	C082	17-May-82			
4	561-2211	B WIRE, BROWN, 22 AWG, 7 X 30	P	*					P	FT	0.0080	C057	17-May-82			
4	561-2222	B WIRE, RED, 22 AWG, 7 X 30	P	*					P	FT	0.0080	764	17-May-82			
4	561-2333	B WIRE, ORANGE, 22 AWG, 7 X 30	P	*					P	FT	0.0080	764	17-May-82			
4	561-2244	B WIRE, YELLOW, 22 AWG, 7 X 30	P	*					P	FT	0.0080	764	17-May-82			
4	561-2255	B WIRE, GREEN, 22 AWG, 7 X 30	P	*					P	FT	0.0080	764	17-May-82			
4	561-2266	B WIRE, BLUE, 22 AWG, 7 X 30	EX	P	1				P	FT	0.0080	764	17-May-82			
4	561-2277	B WIRE, VIOLET, 22 AWG, 7 X 30	P	*					P	FT	0.0080	C116	17-May-82			
4	561-2288	B WIRE, GRAY, 22 AWG, 7 X 30	P	*					P	FT	0.0080	764	17-May-82			
4	561-2399	B WIRE, WHITE, 22 AWG, 7 X 30	P	*					P	FT	0.0080	764	17-May-82			
4	830-0010	O STUD, 1/4 TURN FASTENER	P	*					P	FT	0.0080	764	17-May-82			
3	830-0011	O RETAINER, 1/4 TURN	EX	X	1	P	C	P	EA	449		18-Jan-82				
3	830-0012	O CLIP-ON RECEPTACLE, 1/4 TURN	EX	X	1	P	C	P	EA	449		18-Jan-82				
3	805-0057	C CLIP, DISK HOLD DOWN #2 A3	EX	X	1	P	C	P	EA	449		18-Jan-82				
3	830-0014	O FASTENER, RETAINING CLIP	EX	X	7	P	C	P	EA	449		18-Jan-82				
3	865-0005	MA FOOT, 52HT, 8Ø	EX	X	7	P	C	P	EA	545		18-Jan-82				
3	946-0000	A TAPE, FOAM, DOUBLE SIDED ADHESIVE	EX	X	1	P	B	P	RL	0.2300		18-Jan-82				
3	810-0364	A CHASSIS, PAINTED A3	RM	P	1	A	P	EA	EA	559		18-Jan-82				
3	800-0364	C CHASSIS, MACHINED A3	RM	P	1	A	P	EA	EA	983		28-Feb-81				
3	815-0364	B BASE, DIECAST CHASSIS A3	RM	P	1	A	P	EA	EA	A012		28-Feb-81				
3	908-0003	1 ADHESIVE, RTV #3145 GRAY	EX	X	1	P	B	P	EA	0.0100		18-Jan-82				
3	815-0087	MC KEYBOARD COVER, APPLEIII	RM	P	1	P	C	P	EA	A024		18-Jan-82				
3	825-0163	1 LABEL, POWER ON	EX	X	7	P	C	P	EA	427		18-Jan-82				
3	430-1005	C SCREW, TAPPING, 4 X 20 .500	EX	X	*				P	EA	C086	18-Jan-82				
3	825-0321	C LABEL, A3 SERIAL NO. & UL	EX	X	7	P	C	P	EA	A024		18-Jan-82				
3	825-0057	1 LABEL, REAR CONNECTOR, APPLEIII	EX	X	1	P	C	P	EA	427		18-Jan-82				
3	410-1206	B SCREW, M2.5 X 0.45 X 6MM POZI DR	EX	X	1	P	C	P	EA	C086		18-Jan-82				
3	810-1410	B SCREW, M3.5 X .6 X 10MM PHMS	EX	X	1	P	C	P	EA	C086		18-Jan-82				
3	860-0020	O WASHER, FLAT M3.5	EX	X	1	P	C	P	EA	C086		18-Jan-82				
3	825-0066	C LABEL, MODEL NAMEPLATE A3	EX	X	7	P	C	P	EA	441		18-Jan-82				
3	699-0031	E POWER SUPPLY, OEM ASTEC A3 (110V)	VX	A	1	P	A	P	EA	A035		18-Jan-82				
3	590-0032	A CABLE ASSY, 15" LG, 26 COND RIBBON	RC	P	1	P	B	P	EA	877		18-Jan-82				
4	517-0030	A CONN, 26P SOCKET INSUL. DISPLACEMNT	RC	P	1	P	B	P	EA	482		19-Jan-81				
4	420-1006	B SCREW, FLAT 26 CONDUCTOR	RC	P	1	P	B	P	EA	2.5000		19-Jan-81				
3	490-1006	B SCREW, TAP P.H.C.R M3.5X.6X10	RC	P	1	P	B	P	EA	19		19-Jan-81				
3	610-0011	A SUBASSY, TOP COVER, A3	RM	X	1	P	C	P	EA	C086		26-Apr-82				
3	819-0006	PM TOP COVER, APPLE III	RM	X	1	P	C	P	EA	559		18-Jan-82				
4	805-0031	J COVER, PERIPHERAL CARD, APPLEIII	RM	P	1	P	C	P	EA	C036		19-Jan-81				
3	610-4106	A ASSY, TST.A3 LOGIC FOR SV RAM	RM	P	1	P	C	P	EA	A031		19-Jan-81				
4	610-0106	E ASSY, UNIT#A3 LOGIC FOR SV RAM	A	*					P	EA	923	18-Jan-82				
4	610-0106	E ASSY, UNIT#A3 LOGIC FOR SV RAM	A	*					P	EA	A091	18-Jan-82				

BIL VERSION 29-Mar-79 PAGE 6

PERSONAL COMPUTER SYSTEMS

INDENTED BILLS OF MATERIAL

PRINTED 23-JUL-82 13:32

EFFECTIVITY DATE: ALL

PARENT PART: 610-8156

INACT.FO.256K APPLE III
 ERC: B SRCE CODE: A TYPE: *

UM: EA
 ABC: A

PROD CODE:
 PLAN CODE: U

ECN: A116 ASSEMBLY QTY: 1

DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR B T S A P CD C Y P B L UH	EXTENDED QTY PER	ECN CHG	START DATE	S/N	DATE	CLOSE S/N
5	050-0047	A SCHEMATIC, MN LOGIC BD A3/5V RAM	D * * U EA	0	923	18-Jan-82			
5	112-0102	B REB. ARRAY, 47 OHM	RP P 1 P C P EA	5	611	18-Jan-82			
5	306-0011	A IC, 74LS11, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0011	A IC, 74LS11	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0125	A IC, 74LS123, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0125	A IC, 74LS123	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0000	A IC, 74LS00, TESTED & BURNED-IN	A * * C P EA	2	807	18-Jan-82			
6	303-0000	A IC, 74LS00	RI P 1 P B P EA	2	442	15-Jul-81			
5	306-0002	A IC, 74LS02, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0002	A IC, 74LS02	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0004	A IC, 74LS04, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0004	A IC, 74LS04	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0005	A IC, 74LS05, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0005	A IC, 74LS05	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0008	A IC, 74LS08, TESTED & BURNED-IN	A * * C P EA	2	807	18-Jan-82			
6	303-0008	A IC, 74LS08	RI P 1 P B P EA	2	442	15-Jul-81			
5	308-0010	A IC, 74810, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	307-0010	A IC, 74810	RI P 1 P B P EA	1	807	18-Jan-82			
5	306-0020	A IC, 74LS20, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0020	A IC, 74LS20	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0021	A IC, 74LS21, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0021	A IC, 74LS21	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0032	A IC, 74LS32, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0032	A IC, 74LS32	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0051	A IC, 74LS51, TESTED & BURNED-IN	A * * C P EA	2	807	18-Jan-82			
6	303-0051	A IC, 74LS51	RI P 1 P B P EA	2	442	15-Jul-81			
5	306-0074	A IC, 74LS74, TESTED & BURNED-IN	A * * C P EA	2	807	18-Jan-82			
6	303-0074	A IC, 74LS74	RI P 1 P B P EA	2	442	15-Jul-81			
5	306-0086	A IC, 74LS86, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0086	A IC, 74LS86	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0138	A IC, 74LS138, TESTED & BURNED-IN	A * * C P EA	3	807	18-Jan-82			
6	303-0138	A IC, 74LS138	RI P 1 P B P EA	3	442	15-Jul-81			
5	306-0126	A IC, 74LS126, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0126	A IC, 74LS126	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0193	A IC, 74LS153, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0193	A IC, 74LS153	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0157	A IC, 74LS157, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0157	A IC, 74LS157	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0161	A IC, 74LS161, TESTED & BURNED-IN	A * * C P EA	4	807	18-Jan-82			
6	303-0161	A IC, 74LS161	RI P 1 P B P EA	4	442	15-Jul-81			
5	306-0174	A IC, 74LS174, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0174	A IC, 74LS174	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0139	A IC, 74LS139, TESTED & BURNED-IN	A * * C P EA	2	807	18-Jan-82			
6	303-0139	A IC, 74LS139	RI P 1 P B P EA	2	442	15-Jul-81			
5	306-0244	A IC, 74LS244, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0244	A IC, 74LS244	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0251	A IC, 74LS251, TESTED & BURNED-IN	A * * C P EA	1	807	18-Jan-82			
6	303-0251	A IC, 74LS251	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0257	A IC, 74LS257, TESTED & BURNED-IN	A * * C P EA	3	807	18-Jan-82			
6	303-0257	A IC, 74LS257	RI P 1 P B P EA	3	442	15-Jul-81			

BIL VERSION 29-Mar-79 PAGE 7

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:32

EFFECTIVITY DATE: ALL

INDEXED BILLS OF MATERIAL

INACT.F0.256K APPLE III UM: EA
 ERC: B SRCE CODE: A TYPE: * ABC: A
 ECN: A116 ASSEMBLY QTY: 1
 DATE: 08-Jul-82

PARENT PART: 610-8136

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR B T S A P CD C Y P B L UM	EXTENDED QTY PER	ECN CHO	START DATE	S/N	CLOSE DATE	S/N
5	306-0283	A IC, 74LS283, TESTED & BURNED-IN	A * C P EA	1	807	18-Jan-82			
6	305-0283	A IC, 74LS283	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0323	A IC, 74LS323, TESTED & BURNED-IN	A * C P EA	1	807	18-Jan-82			
6	305-0323	A IC, 74LS323	RI P 1 P B P EA	1	442	15-Jul-81			
5	306-0374	A IC, 74LS374, TESTED & BURNED-IN	A * C P EA	6	807	18-Jan-82			
6	305-0374	A IC, 74LS374	RI P 1 P B P EA	6	442	15-Jul-81			
5	306-0399	A IC, 74LS399, TESTED & BURNED-IN	A * C P EA	2	807	18-Jan-82			
6	305-0399	A IC, 74LS399	RI P 1 P B P EA	2	442	15-Jul-81			
6	307-0074	A IC, 74S74	A * C P EA	3	807	18-Jan-82			
5	308-0086	A IC, 74886, TESTED & BURNED-IN	RI P 1 P B P EA	3	C063	15-Jul-81			
6	307-0086	A IC, 74886	A * C P EA	2	807	18-Jan-82			
5	308-0151	A IC, 74S151, TESTED & BURNED-IN	RI P 1 P B P EA	2	C063	15-Jul-81			
6	307-0151	A IC, 74S151	A * C P EA	1	807	18-Jan-82			
5	308-0153	A IC, 74S153, TESTED & BURNED-IN	RI P 1 P B P EA	4	807	18-Jan-82			
6	307-0153	A IC, 74S153	A * C P EA	4	C063	15-Jul-81			
5	308-0175	A IC, 74S175, TESTED & BURNED-IN	RI P 1 P B P EA	1	807	18-Jan-82			
6	307-0175	A IC, 74S175	A * C P EA	1	C063	15-Jul-81			
5	308-0195	A IC, 74S195, TESTED & BURNED-IN	RI P 1 P B P EA	1	807	18-Jan-82			
6	307-0195	A IC, 74S195 OR 93800	A * C P EA	1	807	18-Jan-82			
5	308-0257	A IC, 74S257, TESTED & BURNED-IN	RI P 1 P B P EA	4	807	18-Jan-82			
6	307-0257	A IC, 74S257	A * C P EA	4	C063	15-Jul-81			
5	342-0032	A IC VIDEO CONTROL TST & BRN IN	RI P 1 P B P EA	1	831	18-Jan-82			
6	341-0032	ROM, VIDED CONTROL	RL P 1 P C P EA	1	449	10-Aug-81			
5	302-0166	A IC, 74166, TESTED & BURNED-IN	A * C P EA	1	807	18-Jan-82			
6	301-0166	A IC, 74166	RI P 1 P B P EA	1	C063	15-Jul-81			
5	369-6502	B IC MICRO 6502B 3MHZ TEST&BURN-IN	A * C P EA	1	A098	18-Jan-82			
6	368-6502	IC/MICROPROCESSOR 6502B 3MHZ	A * C P EA	1	458	31-Aug-81			
5	354-0380	A IC LM380 AUDIO PWR AMP TST & BRN IN	RI P 1 P C P EA	1	831	18-Jan-82			
6	353-0380	IC, LM380 AUDIO PWR AMP TST & BRN IN	A * C P EA	1	449	10-Aug-81			
5	342-0030	MA IC ROM SYNCHROM WITH BURN-IN	RI P 1 P C P EA	1	831	18-Jan-82			
6	341-0030	ROM, SYNCHROM	RL P 1 P C P EA	1	449	10-Aug-81			
5	342-0031	MA IC ROM BOOT WITH BURN-IN	RI P 1 P C P EA	1	831	18-Jan-82			
6	341-0031	ROM, BOOT	RL P 1 P C P EA	1	532	10-Aug-81			
5	338-6522	A IC 6522 I/F ADAPTER TEST&BURN-IN	RI P 1 P C P EA	1	877	18-Jan-82			
6	337-6522	IC, 6522 VERSATILE I/F ADAPTER	RL P 1 P B P EA	2	449	31-Aug-81			
5	338-0002	A ASYN COMM INT 6551A TST & BRN IN	A * C P EA	1	877	18-Jan-82			
6	337-0002	A IC, 6551A ASYN COMM I/F ADAP. SELE.	A * C P EA	1	736	31-Aug-81			
5	316-8304	O IC, 8304B 8-BIT TRI-ST. TST&BRN IN	RI P 1 P B P EA	1	831	18-Jan-82			
6	315-8304	IC, 8304B 8-BIT TRI-ST. TST&BRN IN	A * C P EA	1	449	10-Aug-81			
5	330-0556	A IC 556 DUAL TIMER TEST&BURN-IN	RI P 1 P C P EA	3	449	10-Aug-81			
6	329-0556	IC, 556 DUAL TIMER	RI P 1 P B P EA	3	802	18-Jan-82			
5	302-9334	B IC, 9334, TESTED & BURNED-IN	A * C P EA	3	C063	20-Jul-81			
6	301-9334	A IC, 9334	EX P 7 P B P EA	0	C063	20-Jul-81			
5	301-0259	A IC, 74259	RI P 1 P B P EA	1	831	18-Jan-82			
6	360-1488	O IC, 1488 QUAD LINE DRIVER TST&BRN-IN	A * C P EA	1	449	10-Aug-81			
5	359-1488	A IC, 1488 QUAD LINE DRIVER	RI P 1 P C P EA	1	831	18-Jan-82			
6	358-9708	A IC, 9708 6-CH 8-BIT TESTED & BRN IN	RI P 1 P C P EA	1	449	10-Aug-81			
5	355-9708	IC, 9708 6-CH 8-BIT A TO D	A * C P EA	1	831	18-Jan-82			
6	354-9708	IC, 9708 6-CH 8-BIT TESTED & BRN IN	RI P 1 P B P EA	1	449	10-Aug-81			

PRINTED 23-JUL-82 13:33

PERSONAL COMPUTER SYSTEMS

BIL VERSION 29-Mar-79 PAGE 8

EFFECTIVITY DATE: ALL

PARENT PART: 610-8156

INACT.F0,256K APPLE III

UM: EA

ECN: A116 ASSEMBLY QTY: 1

INDEXED BILLS OF MATERIAL

DATE: 08-JUL-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	ST	BA	P	BL	UM	EXTENDED QTY PER	ECN CHG	START DATE	S/N	CLOSE DATE	S/N
6	359-1489	O IC, 1489 LINE RECEIVER	RI	P	I	P	C	P	EA	449	10-AUG-81			
5	372-3904	A TRANSISTOR,NPN 8H, 1AHP, 2N3904	RP	P	I	P	C	P	EA	4026	18-JAN-82			
5	111-0003	O RES ARRAY D/A CONVERTER 7 RES 8 PIN	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	111-0025	A RESISTOR ARRAY, 8IP 7 RES, SPECIAL	P	*	*	*	*	*	EA	923	18-JAN-82			
5	135-9101	A CAP, 1uF +80-20% Z5U/V5V 50V	RP	P	I	P	C	P	EA	433	18-JAN-82			
5	151-3501	B CHOKER, 27uH 10%	RP	P	I	P	C	P	EA	479	18-JAN-82			
5	342-0028	A IC, PROM, STATE MACH. P6A W/BURN-IN	A	*	*	*	*	*	EA	831	18-JAN-82			
6	341-0028	A IC, PROM, STATE MACHINE, P6A	RL	A	I	P	C	P	EA	454	10-AUG-81			
7	335-0471	A PROM,748471	RL	A	I	P	C	P	EA					
5	334-0005	A IC 1024 X 4 STATIC RAM T8T & BRN IN	RL	A	*	*	*	*	EA	877	18-JAN-82			
6	334-2114	O IC, 1024 X 4 STATIC RAM 2114	RL	P	I	P	C	P	EA	449	31-AUG-81			
5	306-0133	A IC, 74LS133, TESTED & BURNED-IN	RL	A	*	*	*	*	EA	807	18-JAN-82			
6	305-0133	A IC, 74LS133	RI	P	I	P	C	P	EA	442	15-JUL-81			
5	820-0043	O PCB, MAIN LOGIC BD #2 A3	RP	P	7	P	A	P	EA	503	18-JAN-82			
5	101-4112	A RES, 1/4W 5% 1.1K OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	101-4104	A RES 1/4W 5% 100K OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	126-5102	O CAP, 10uF 16V	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	197-0001	B CRYSTAL, 14.318630 MHZ	RI	P	I	P	C	P	EA	487	18-JAN-82			
5	101-4102	A RES 1/4W 5% 1K OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	126-4102	O CAP, 1uF 50V	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	101-4225	A RES 1/4W 5% 2.2 MEG OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	131-3401	O CAP, 20PF 5% NPO 50V	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	372-4258	A TRANSISTOR, PNP HIGH BPD, 8H, 2N4258	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	101-4335	A RES 1/4W 5% 3.3 MEG OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	197-0004	O CRYSTAL, TUNING FORK 32.768 KHZ	RI	P	I	P	C	P	EA	449	18-JAN-82			
5	101-4302	A RES 1/4W 5% 3K OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	101-4470	A RES 1/4W 5% 47 OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	101-4474	A RES 1/4W 5% 470K OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	101-4473	A RES 1/4W 5% 47K OHM	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	515-0053	MB CONN, STRAIGHT HEADER 2 PIN	RP	P	I	P	C	P	EA	426	18-JAN-82			
5	371-4148	A DIODE, 1N4148	RP	P	I	P	C	P	EA	532	18-JAN-82			
5	519-0011	O CONNECTOR, 9 PIN D	RC	P	I	P	C	P	EA	731	18-JAN-82			
5	519-0038	A CONN, 15 PIN D RT ANGLE PC MOUNT	RC	P	I	P	C	P	EA	467	18-JAN-82			
5	515-0001	A JACK, PHONE RT ANG (MON) NTT 333-15	RP	P	I	P	C	P	EA	725	18-JAN-82			
5	515-0002	O JACK, PHONE	RP	P	I	P	C	P	EA	784	18-JAN-82			
5	519-0014	O CONNECTOR, HEADER 25 PIN	RP	P	I	P	C	P	EA	467	18-JAN-82			
5	519-0016	A CONN, 26 PIN HEADER W/O MTQ EARS	RC	P	I	P	C	P	EA	467	18-JAN-82			
5	519-0017	A CONN, 26 PIN HEADER W/MTQ EARS	RC	P	I	P	C	P	EA	523	18-JAN-82			
5	135-5102	A COIL, 10UH RADIAL	RC	P	I	P	C	P	EA	523	18-JAN-82			
5	519-0018	O CONNECTOR, 25 PIN D	RC	P	I	P	C	P	EA	DC44	18-JAN-82			
5	378-0001	B LED, RED	RP	P	I	P	C	P	EA	DC53	18-JAN-82			
5	511-1401	C SOCKET, IC 14 PIN	RP	P	I	P	C	P	EA	467	18-JAN-82			
5	511-1601	C SOCKET, IC 16 PIN	RP	P	I	P	C	P	EA	799	18-JAN-82			
5	511-1801	C SOCKET, IC 18 PIN	RP	P	I	P	C	P	EA	799	18-JAN-82			
5	511-2001	C SOCKET, IC 20 PIN	RP	P	I	P	C	P	EA	799	18-JAN-82			
5	511-2401	C SOCKET, IC 24 PIN	RP	P	I	P	C	P	EA	799	18-JAN-82			
5	511-2801	C SOCKET, IC 28 PIN	RP	P	I	P	C	P	EA	799	18-JAN-82			
5	511-4001	C SOCKET, IC 40 PIN	RP	P	I	P	C	P	EA	799	18-JAN-82			
5	342-0036	A IC, PROM, CAS865 1 WITH BURN-IN	A	*	*	*	*	*	EA	831	18-JAN-82			

BIL VERSION 29-Mar-79 PAGE 9

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:34

INDENTED BILL OF MATERIAL

EFFECTIVITY DATE: ALL
 PARENT PART: 610-8156
 INACT.FO.256K APPLE III
 ERC: B SRCE CODE: A TYPE: *

UM: EA
 ABC: A
 PROD CODE: U
 PLAN CODE: U

ECN: A116 ASSEMBLY QTY: 1
 DATE: 08-Jul-82

START DATE S/N
 CLOSE DATE S/N

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR S T B A P CD C Y P B L UM	EXTENDED QTY PER	ECN CHO	START DATE	S/N	CLOSE DATE	S/N
7	333-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL P 1 P B P EA	1	SARA 27-Oct-80				
5	342-0043	A IC PROM U174 WITH BURN-IN	RL A 1 P C P EA	1	831 18-Jan-82				
6	341-0043	O PROM, 1024 X 4 U174	RL A 1 P C P EA	1	449 10-Aug-81				
7	333-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL P 1 P B P EA	1	SARA				
5	342-0045	A IC PROM U176.2 WITH BURN-IN	RL A 1 P C P EA	1	831 18-Jan-82				
6	341-0045	O PROM, 1024 X 4 U176.2	RL A 1 P C P EA	1	449 10-Aug-81				
7	333-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL P 1 P B P EA	1	SARA				
5	342-0046	A IC PROM U180 WITH BURN-IN	RL A 1 P C P EA	1	831 18-Jan-82				
6	341-0046	O PROM, 1024 X 4 U180	RL A 1 P C P EA	1	449 10-Aug-81				
7	333-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL P 1 P B P EA	1	SARA				
5	342-0055	MA IC PROM U175-65 WITH BURN-IN	RL A 1 P C P EA	1	831 18-Jan-82				
6	341-0055	O PROM U175-65	RL A 1 P C P EA	1	503 10-Aug-81				
7	333-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL P 1 P B P EA	1	SARA 27-Oct-80				
5	513-5002	D CONNECTOR, EDGE 25/50 PIN (NO TABS)	RP P 1 P C P EA	4	760 18-Jan-82				
5	519-0103	A HEADER, RIGHT ANGLE 10 PINS	RP P 1 P C P EA	1	877 18-Jan-82				
5	306-0260	A IC, 74LS260, TESTED & BURNED-IN	P *	1	807 18-Jan-82				
6	305-0260	A IC, 74LS260	P *	1	442 15-Jul-81				
5	131-5101	O CAP, 10pF 10% NPO 50V	RI P 1 P B P EA	1	426 18-Jan-82				
5	126-6404	A CAP, 220UF 16V	RP P 1 P C P EA	1	C103 24-May-82				
5	151-5101	O CHOKER, 10uH 10%	RP P 1 P B P EA	2	442 18-Jan-82				
5	132-6401	O CAP, 220pF 10% Z5F 50V	RP P 1 P C P EA	4	426 18-Jan-82				
5	372-3906	A TRANSISTOR, PNP 8M &. 2N3906	RP P 1 P C P EA	1	C026 18-Jan-82				
5	101-4750	A RES 1/4W 5% 75 OHM	RP P 1 P C P EA	3	426 18-Jan-82				
5	101-4152	A RES 1/4W 5% 1.5K OHM	RP P 1 P C P EA	2	426 18-Jan-82				
5	101-4682	A RES 1/4W 5% 6.8K OHM	RP P 1 P C P EA	2	426 18-Jan-82				
5	519-0032	O CONN, 10 PIN FRICTION LOCK	RC P 1 P C P EA	1	442 18-Jan-82				
5	306-0132	A IC, 74LS132, TESTED & BURNED-IN	P *	1	807 18-Jan-82				
6	305-0132	A IC, 74LS132	P *	1	442 15-Jul-81				
5	375-0005	O DIODE, 1N5712 SCHOTTKY BARRIER	RI P 1 P B P EA	1	449 18-Jan-82				
5	111-0018	O RES ARRAY 7 X 330 OHM	RP P 1 P C P EA	1	426 18-Jan-82				
5	111-0017	O RES ARRAY 9 X 3.3K OHM	RP P 1 P C P EA	1	426 18-Jan-82				
5	101-4330	A RES 1/4W 5% 33 OHM	RP P 1 P C P EA	4	426 18-Jan-82				
5	101-4101	O RES 1/4W 5% 100 OHM	RP P 1 P C P EA	2	426 18-Jan-82				
5	111-0014	O RES ARRAY 9 X 1K OHM	RP P 1 P C P EA	2	426 18-Jan-82				
5	101-4472	A RES 1/4W 5% 4.7K OHM	RP P 1 P C P EA	2	426 18-Jan-82				
5	101-4153	A RES 1/4W 5% 15K OHM	RP P 1 P C P EA	1	426 18-Jan-82				
5	101-4241	A RES 1/4W 5% 240 OHM	RP P 1 P C P EA	4	426 18-Jan-82				
5	101-4105	A RES, 1/4W 5% 1M OHM	RP P 1 P C P EA	2	426 18-Jan-82				
5	101-4301	A RES, 1/4W 5% 300 OHM	RP P 1 P C P EA	3	426 18-Jan-82				
5	101-4471	A RES 1/4W 5% 470 OHM	RP P 1 P C P EA	3	426 18-Jan-82				
5	125-5401	A CAP, 22uF 16V	RP P 1 P C P EA	1	433 18-Jan-82				
5	511-0801	C SOCKET, IC 8 PIN	RP P 1 P C P EA	1	799 18-Jan-82				
5	132-6101	O CAP, 100pF 20% Z5F 50V	RP P 1 P C P EA	3	426 18-Jan-82				
5	101-4181	A RES, 1/4W 5% 180 OHM	RP P 1 P C P EA	1	426 18-Jan-82				
5	119-2101	MA CAP, .01uF 10% 100V	RP P 1 P C P EA	1	449 18-Jan-82				
5	101-4125	A RES 1/4W 5% 1.2 MEG OHM	RP P 1 P C P EA	1	426 18-Jan-82				
5	133-2401	B CAP, .022uF 20% Y5P 25V	RP P 1 P C P EA	2	433 18-Jan-82				
5	138-0002	MO CAP, VAR CERAMIC TRIMMER 3 B/4. OF	RP P 1 P C P EA	1	449 18-Jan-82				
5	111-0019	O RES ARRAY 5 X 1K OHM	RP P 1 P C P EA	3	426 18-Jan-82				

BIL VERSION 29-Mar-79 PAGE 10

PERSONAL COMPUTER SYSTEMS

PRINTED 23-JUL-82 13:34

INDENTED BILLS OF MATERIAL

EFFECTIVITY DATE: ALL

INACT.F0:256K APPLE III UM:EA
 ERC: B SRCE CODE: A TYPE: * ABC: A
 ECN: A116 ASSEMBLY QTY: 1
 DATE: 08-Jul-82 PLAN CODE: U

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	B	T	S	A	P	L	UM	EXTENDED QTY PER	ECN	START DATE	S/N	CLOSE DATE	S/N
6	307-0374	A IC, 748374	RI	P	1	P	B	P	E	A	1	C063	15-Jul-81			
5	308-0000	A IC, 74800, TESTED & BURNED-IN	A	*	C	P	E	A			1	807	18-Jan-82			
6	307-0000	A IC, 74800, QUAD NAND GATE	RI	P	5	P	B	P	E	A	1	C063	15-Jul-81			
5	342-0035	MA IC ROM KYBD ENC. A3 BRN-IN & TBT	A	*	C	P	E	A			1	886	18-Jan-82			
6	341-0035	B ROM, KEYBOARD ENCODER	RL	P	1	P	C	P	E	A	1	683	21-Sep-81			
5	111-0001	O REB ARRAY 7 X 1K OHMS	RP	P	1	P	C	P	E	A	3	426	18-Jan-82			
5	131-5701	O CAP, 47PF 5% N470 50V	RP	P	1	P	C	P	E	A	1	426	18-Jan-82			
5	112-0001	B RES, INDIV. NTRK. 68 OHM/5RES/10PIN	RP	P	1	P	C	P	E	A	1	609	18-Jan-82			
5	376-0001	A TRANSISTOR, MPS-051	RP	P	1	P	C	P	E	A	1	C111	18-Jan-82			
5	119-2701	A CAP, .047UF 10% 100V	RP	P	1	P	C	P	E	A	1	DC3	18-Jan-82			
5	101-4103	A RES 1/4W 5% 10K OHM	RP	P	1	P	C	P	E	A	2	426	18-Jan-82			
5	101-4270	A RES 1/4W 5% 27 OHM	RP	P	1	P	C	P	E	A	3	426	18-Jan-82			
5	101-4121	A RES 1/4W 5% 120 OHM	RP	P	1	P	C	P	E	A	4	426	18-Jan-82			
5	750-0006	B BUS BAR, CAPACITIVE .2UF	RP	P	1	P	C	P	E	A	7	503	18-Jan-82			
5	562-2405	A WIRE, #24 KYNAR GREEN SOLID	EX	X	7	P	C	P	E	A	4	426	18-Jan-82			
5	101-4432	A RES 1/4W 5% 4.3K OHM	RP	P	1	P	C	P	E	A	1	426	18-Jan-82			
5	111-0006	O RES, ARRAY SPECIAL BR. 9P	RP	P	1	P	C	P	E	A	1	426	18-Jan-82			
5	831-0100	A RIVET, PLASTIC .125	RP	X	*	C	P	E	A		8	778	18-Jan-82			
5	306-0014	A IC, 74LS14, TESTED & BURNED-IN	A	*	C	P	E	A			1	807	18-Jan-82			
6	305-0014	A IC, 74LS14	RI	P	1	P	B	P	E	A	1	442	15-Jul-81			
5	136-3101	MC CAP, 1UF 10% XR7 50V	RI	P	1	P	C	P	E	A	6	C082	18-Jan-82			
5	342-0061	A I.C. PROM RAS65 WITH BURN-IN	A	*	C	P	E	A			1	867	18-Jan-82			
6	341-0061	A IC/PROM, RAS65	RL	P	1	C	P	E	A		1	665	24-Aug-81			
5	342-0063	A IC, PROM CASB256 WITH BURN-IN	RL	P	1	C	P	E	A		1	867	18-Jan-82			
6	341-0063	A IC/PROM, CASB256	RL	P	1	C	P	E	A		1	665	24-Aug-81			
5	825-0233	A LABEL, PCB SERIALIZATION	P	*	*	U	E	A			1	A024	26-Apr-82			
5	942-0197	A BAG, POLY 12X18 ANTI-STAT	P	*	*	*	E	A			2	C132	26-Jul-82			
5	875-0005	A CONTACT PROTECTOR	P	*	*	*	E	A			4	426	18-Jan-82			
5	126-6402	O CAP, 220UF 16V	RP	P	1	P	C	P	E	A	1	714	18-Jan-82			
3	805-0099	A ASSY, PCB, TST. 5V MEMORY 256K A3	EX	X	7	P	C	P	E	A	1	972	18-Jan-82			
3	610-4256	B BOTTOM COVER, MAIN BOARD 2 A3	EX	X	7	P	C	P	E	A	1	559	07-Nov-81			
4	860-0018	A STANDOFF M3.5X0.6 BLD THD .200" LO	EX	X	7	P	C	P	E	A	6	661	18-Jan-82			
3	865-0007	A BUMPER, HEMISPHERE, 2" HIGH	EX	X	7	P	C	P	E	A	11	445	18-Jan-82			
3	830-0018	O CLAMP, C-TYPE, FLAT CABLE	EX	X	7	P	C	P	E	A	1	NONE	18-Jan-82			
3	000-0001	A NOT USED THIS ASSEMBLY	EX	X	7	P	C	P	E	A	0	661	18-Jan-82			
3	865-0006	A BUMPER, 80 .23"	EX	X	7	P	C	P	E	A	2	559	18-Jan-82			
3	825-0207	A LABEL, FCC NON COMPLIANCE APPLE 3	EX	X	7	P	C	P	E	A	1	793	18-Jan-82			
3	825-0312	A LABEL, WARNING SLOT A3	EX	X	7	P	C	P	E	A	1	948	18-Jan-82			
3	610-3103	C INACTIVE, SUBASSY, KEYBOARD APPLE III	UN	A	1	P	A	P	E	A	1	948	05-Jun-80			
4	610-4103	O INACTIVE, ASSY, PCB, TST KEYBOARD A3	UN	A	1	P	A	P	E	A	1	SARA				
5	610-0103	INACTIVE, ASSY, PCB, UNITST.KBD A3	RP	P	1	P	A	P	E	A	1	352				
6	820-0025	PCB, KEYBOARD (SARA)	EX	P	7	P	C	P	E	A	1	P150				
6	710-0001	O LAMP 02-1876-01	RP	P	1	P	C	P	E	A	1	442				
6	705-0009	A SWITCH, END ACTUATED PUSH BUTTON	RP	P	1	P	C	P	E	A	1	SARA				
6	515-0014	O CONN, 26 PIN HEADER	RP	P	1	P	C	P	E	A	1	SARA				
6	705-0011	SWITCH, DC51 ALT ACTION (SARA)	RP	P	1	P	C	P	E	A	1	SARA				
6	705-0012	SWITCH, DC51 DUAL ACTION (SARA)	RP	P	1	P	C	P	E	A	4	352				
6	705-0004	O SWITCH DC51-31	RP	P	1	P	C	P	E	A	67	441	05-Jun-80			
4	610-0197	O SUBASSY, KEYCAPS, COMPLETE SET A3	UN	N	1	P	B	P	E	A	1					

BIL VERSION 29-Mar-79 PAGE 11

PERSONAL COMPUTER SYSTEMS

INDENTED BILLS OF MATERIAL

PRINTED 23-Jul-82 13:35

EFFECTIVITY DATE: ALL

PARENT PART: 610-8156

INACT.FO.256K APPLE III
ERC: B BRCE CODE: A TYPE: *

UM: EA
ABC: A
PLAN CODE: U
PROD CODE: U

ECN: A116
DATE: 08-Jul-82
ASSEMBLY QTY: 1

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	S	T	B	A	P	UM	EXTENDED QTY PER	ECN	CHQ	START DATE	S/N	CLOSE DATE	S/N
5	815-0088	0 KEY CAP, APPLE 3, ESCAPE	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0089	0 KEY CAP, APPLE 3, / / 1	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0090	0 KEY CAP, APPLE 3, 2 / 2	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0091	0 KEY CAP, APPLE 3, 3 / 3	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0092	0 KEY CAP, APPLE 3, 4 / 4	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0093	0 KEY CAP, APPLE 3, 5 / 5	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0094	0 KEY CAP, APPLE 3, 6 / 6	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0095	0 KEY CAP, APPLE 3, 7 / 7	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0096	0 KEY CAP, APPLE 3, 8 / 8	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0097	0 KEY CAP, APPLE 3, 9 / 9	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0098	0 KEY CAP, APPLE 3, 0 / 0	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0099	0 KEY CAP, APPLE 3, - / -	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0100	0 KEY CAP, APPLE 3, = / =	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0101	0 KEY CAP, APPLE 3, / / \	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0102	0 KEY CAP, APPLE 3, 7	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0103	0 KEY CAP, APPLE 3, 8	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0104	0 KEY CAP, APPLE 3, 9	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0105	0 KEY CAP, APPLE 3, TAB	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0106	0 KEY CAP, APPLE 3, 0	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0107	0 KEY CAP, APPLE 3, W	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0108	0 KEY CAP, APPLE 3, E	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0109	0 KEY CAP, APPLE 3, R	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0110	0 KEY CAP, APPLE 3, T	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0111	0 KEY CAP, APPLE 3, Y	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0112	0 KEY CAP, APPLE 3, U	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0113	0 KEY CAP, APPLE 3, I	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0114	0 KEY CAP, APPLE 3, O	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0115	0 KEY CAP, APPLE 3, P	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0116	0 KEY CAP, APPLE 3, [/ [RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0117	0 KEY CAP, APPLE 3,] /]	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0118	0 KEY CAP, APPLE 3, ~ / ~	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0119	0 KEY CAP, APPLE 3, 4	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0120	0 KEY CAP, APPLE 3, 5 (W/SENSOR)	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0121	0 KEY CAP, APPLE 3, 6	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0122	0 KEY CAP, APPLE 3, CONTROL	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0123	0 KEY CAP, APPLE 3, A	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0124	0 KEY CAP, APPLE 3, B	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0125	0 KEY CAP, APPLE 3, D (W SENSOR)	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0126	0 KEY CAP, APPLE 3, F	EX	P	1				* P EA	1	441		05-Jun-80			
5	815-0127	0 KEY CAP, APPLE 3, G	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0128	0 KEY CAP, APPLE 3, H	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0129	0 KEY CAP, APPLE 3, J	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0130	0 KEY CAP, APPLE 3, K (W SENSOR)	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0131	0 KEY CAP, APPLE 3, L	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0132	0 KEY CAP, APPLE 3, / / ,	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0133	0 KEY CAP, APPLE 3, ' / ' ,	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0134	0 KEY CAP, APPLE 3, RETURN	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0135	0 KEY CAP, APPLE 3, 1	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0136	0 KEY CAP, APPLE 3, 2	RM	P	1				* P EA	1	441		05-Jun-80			
5	815-0137	0 KEY CAP, APPLE 3, 3	RM	P	1				* P EA	1	441		05-Jun-80			

BIL VERSION 29-Mar-79 PAGE 12

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:36

INDENTED BILLS OF MATERIAL

EFFECTIVITY DATE: ALL

PARENT PART: 610-8156 INACT.F0,256K APPLE III UM: EA PROD CODE: ECN: A116 ASSEMBLY QTY: 1
 ERC: B SRCE CODE: A TYPE: * ABC: A PLAN CODE: U DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	ST	TA	AP	UM	EXTENDED QTY PER	ECN	START DATE	B/N	CLOSE DATE	B/N
3	815-0138	0 KEY CAP, APPLE 3, BRIFT	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0139	0 KEY CAP, APPLE 3, Z	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0140	0 KEY CAP, APPLE 3, X	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0141	0 KEY CAP, APPLE 3, C	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0142	0 KEY CAP, APPLE 3, V	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0143	0 KEY CAP, APPLE 3, R	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0144	0 KEY CAP, APPLE 3, N	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0145	0 KEY CAP, APPLE 3, M	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0146	0 KEY CAP, APPLE 3, S	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0147	0 KEY CAP, APPLE 3, >	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0148	0 KEY CAP, APPLE 3, ?	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0149	0 KEY CAP, APPLE 3, BRIFT	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0150	0 KEY CAP, APPLE 3, ARROW UP	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0151	0 KEY CAP, APPLE 3, 0 (ZERO)	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0152	0 KEY CAP, APPLE 3, (DECIMAL)	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0153	0 KEY CAP, APPLE 3, ALPHA LOCK	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0154	0 KEY CAP, APPLE 3, APPLE 1	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0155	0 KEY CAP, APPLE 3, APPLE 2	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0156	0 KEY CAP, APPLE 3, SPACE BAR	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0157	0 KEY CAP, APPLE 3, ARROW LEFT	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0158	0 KEY CAP, APPLE 3, ARROW RIGHT	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0159	0 KEY CAP, APPLE 3, ARROW DOWN	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0160	0 KEY CAP, APPLE 3, - (MINUS)	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0161	0 KEY CAP, APPLE 3, ENTER	RM	P	1	*	P	EA	441	05-Jun-80			
3	815-0163	3 RESET KEY, APPLE III	RM	P	1	*	P	EA	427	05-Jun-80			
3	815-0179	1 SPACER, RESET SWITCH A3	RM	P	1	P	C	EA	427	05-Jun-80			
3	062-0032D	0 DHQ, KEY CAP 1 SPACE 17 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0033D	0 DHQ, KEY CAP 1 SPACE 10 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0034D	0 DHQ, KEY CAP 1 SPACE 10 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0036D	0 DHQ, KEY CAP 1 SPACE W/BUMP 5 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0037D	0 DHQ, KEY CAP 1-1/2 SPACE	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0038D	0 DHQ, KEY CAP 1-1/2 SPACE 17 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0039D	0 DHQ, KEY CAP 1-1/4 SPACE 17 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0040D	0 DHQ, KEY CAP 1-1/2 SPACE 5 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0041D	0 DHQ, KEY CAP 1-3/4 SPACE 10 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0042D	0 DHQ, KEY CAP 2 SPACE 10 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0043D	0 DHQ, KEY CAP 2 SPACE 17 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0044D	0 DHQ, KEY CAP 2-1/4 SPACE 17 DEG	EX	D	7	S	U	EA	441	05-Jun-80			
3	062-0049D	0 DHQ, KEY CAP 6-1/2 SPACE	EX	D	7	S	U	EA	441	05-Jun-80			
3	000-0003	A NOT USED THIS ASSEMBLY							NONE	26-Apr-82			
3	000-0004	A NOT USED THIS ASSEMBLY							NONE	26-Apr-82			
3	000-0005	A NOT USED THIS ASSEMBLY							NONE	26-Apr-82			
3	000-0006	A NOT USED THIS ASSEMBLY							NONE	26-Apr-82			
3	000-0007	A NOT USED THIS ASSEMBLY							NONE	26-Apr-82			
3	825-0354	A LABEL, NOMEX PAPER							A024	26-Apr-82			
3	914-0028	A LABEL, CARRIAGE LOCK							816	26-Apr-82			
3	825-0357	A LABEL, 256K							A024	26-Apr-82			
3	610-5156	A SUBASSY, A3 PAN, LOGIC & MEM BD 256K							A024	26-Apr-82			
3	610-5129D	A DHQ, SUBASSY A3 PAN, LOGIC & MEM BD							A024	26-Apr-82			
4	610-5129	A KEY, TEST LOGIC, END SW, PAM											

BIL VERSION 29-Mar-79 PAGE 13

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:37

EFFECTIVITY DATE: ALL
 PARENT PART: 610-8156
 INACT.FO.256K APPLE III
 ERC: B SRCE CODE: A TYPE: *
 I N D E N T E D B I L L S O F M A T E R I A L
 UM: EA
 ABC: A
 PROD CODE: U
 PLAN CODE: U
 ECN: A116
 ASSEMBLY QTY: 1
 DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR B T S A P CD C Y P B L U M	EXTENDED QTY PER	ECN CHG	START DATE	B/N	CLOSE DATE	S/N
5	610-0106	E ASSY,UNTST.A3 LOGIC FOR 5V RAM	A * A P EA	1	A091	18-Jan-82			
6	610-0106D	E DWG.ASSY.UNIT.A3 LOGIC FOR 5V RAM	* * * EA	0	A091	26-Jul-82			
6	050-0047	A SCHEMATIC,MN.LOGIC 8Q.A3/5V RAM	D * * U EA	0	923	18-Jan-82			
6	112-0102	B RES. ARRAY.47 OHM	RP P 1 P C P EA	5	611	18-Jan-82			
6	306-0011	A IC. 74LS11. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0011	A IC. 74LS11	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0125	A IC. 74LS125. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0125	A IC. 74LS125	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0000	A IC. 74LS000. TESTED & BURNED-IN	A * C P EA	2	607	18-Jan-82			
6	305-0000	A IC. 74LS000	RI P 1 P B P EA	2	442	15-Jul-81			
6	306-0002	A IC. 74LS02. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0002	A IC. 74LS02M	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0004	A IC. 74LS04. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0004	A IC. 74LS04	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0005	A IC. 74LS05. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0005	A IC. 74LS05	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0008	A IC. 74LS08. TESTED & BURNED-IN	A * C P EA	2	607	18-Jan-82			
6	305-0008	A IC. 74LS08	RI P 1 P B P EA	2	442	15-Jul-81			
6	308-0010	A IC. 74S10. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	307-0010	A IC. 74S10	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0020	A IC. 74LS20. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0020	A IC. 74LS20	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0021	A IC. 74LS21. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0021	A IC. 74LS21	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0032	A IC. 74LS32. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0032	A IC. 74LS32	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0051	A IC. 74LS51. TESTED & BURNED-IN	A * C P EA	2	607	18-Jan-82			
6	305-0051	A IC. 74LS51	RI P 1 P B P EA	2	442	15-Jul-81			
6	306-0074	A IC. 74LS74. TESTED & BURNED-IN	A * C P EA	2	607	18-Jan-82			
6	305-0074	A IC. 74LS74	RI P 1 P B P EA	2	442	15-Jul-81			
6	306-0086	A IC. 74LS86. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0086	A IC. 74LS86	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0138	A IC. 74LS138. TESTED & BURNED-IN	A * C P EA	3	607	18-Jan-82			
6	305-0138	A IC. 74LS138	RI P 1 P B P EA	3	442	15-Jul-81			
6	306-0126	A IC. 74LS126. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0126	A IC. 74LS126	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0153	A IC. 74LS153. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0153	A IC. 74LS153	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0157	A IC. 74LS157. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0157	A IC. 74LS157	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0161	A IC. 74LS161. TESTED & BURNED-IN	A * C P EA	4	607	18-Jan-82			
6	305-0161	A IC. 74LS161	RI P 1 P B P EA	4	442	15-Jul-81			
6	306-0174	A IC. 74LS174. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0174	A IC. 74LS174	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0139	A IC. 74LS139. TESTED & BURNED-IN	A * C P EA	2	607	18-Jan-82			
6	305-0139	A IC. 74LS139	RI P 1 P B P EA	2	442	15-Jul-81			
6	306-0244	A IC. 74LS244. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0244	A IC. 74LS244	RI P 1 P B P EA	1	442	15-Jul-81			
6	306-0251	A IC. 74LS251. TESTED & BURNED-IN	A * C P EA	1	607	18-Jan-82			
6	305-0251	A IC. 74LS251	RI P 1 P B P EA	1	442	15-Jul-81			

PRINTED 23-Jul-82 13:38

PERSONAL COMPUTER SYSTEMS

BIL VERSION 29-Mar-79 PAGE 14

INDENTED BILLS OF MATERIAL

EFFECTIVITY DATE: ALL PARENT PART: 610-8156 INACT.F0.256K APPLE III ECN: A116 ASSEMBLY QTY: 1
 ERC: B BRCE CODE: A TYPE: * UM: EA PROD CODE: U DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	B	T	A	P	L	UM	EXTENDED QTY PER	ECN	CHG	DATE	START	B/N	DATE	CLOSE	S/N
6	306-0257	A IC, 74LS257, TESTED & BURNED-IN	A	*					C	P	EA	807		18-Jan-82				
7	305-0257	A IC, 74LS257	RI	P	1	P	B	P	EA	3	442		15-Jul-81					
6	306-0283	A IC, 74LS283, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	303-0283	A IC, 74LS283	RI	P	1	P	B	P	EA	1	442		15-Jul-81					
6	306-0323	A IC, 74LS323, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	305-0323	A IC, 74LS323	RI	P	1	P	B	P	EA	1	442		15-Jul-81					
6	306-0374	A IC, 74LS374, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	6	807		18-Jan-82					
6	305-0374	A IC, 74LS374	RI	P	1	P	B	P	EA	6	442		15-Jul-81					
6	306-0399	A IC, 74LS399, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	2	807		18-Jan-82					
6	305-0399	A IC, 74LS399	RI	P	1	P	B	P	EA	2	442		15-Jul-81					
6	307-0074	A IC, 74874, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	3	807		18-Jan-82					
6	308-0086	A IC, 74874	RI	P	1	P	B	P	EA	3	807		18-Jan-82					
6	307-0086	A IC, 74886, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	3	807		18-Jan-82					
6	308-0151	A IC, 748151, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	2	807		18-Jan-82					
6	307-0151	A IC, 748151	RI	P	1	P	B	P	EA	2	807		18-Jan-82					
6	308-0153	A IC, 748153, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	4	807		18-Jan-82					
6	307-0153	A IC, 748153	RI	P	1	P	B	P	EA	4	807		18-Jan-82					
6	308-0175	A IC, 748175, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	307-0175	A IC, 748175	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	308-0195	A IC, 748195, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	307-0195	A IC, 748195 OR 93800	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	308-0257	A IC, 748257, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	307-0257	A IC, 748257	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	342-0032	A IC VIDEO CONTROL TST & BRN IN	RL	P	1	P	C	P	EA	1	449		10-Aug-81					
6	341-0032	O ROM, VIDEO CONTROL	RL	P	1	P	C	P	EA	1	831		18-Jan-82					
6	302-0166	A IC, 74166, TESTED & BURNED-IN	RI	P	1	P	B	P	EA	1	449		10-Aug-81					
6	301-0166	A IC, 74166	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	369-6502	B IC MICRO 6502B 3MHZ TEST&BURN-IN	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	368-6502	IC MICROPROCESSOR 6502B 3MHZ	RI	P	1	P	B	P	EA	1	807		18-Jan-82					
6	354-0380	A IC LM380 AUDIO PWR AMP TST & BRN IN	RI	P	1	P	C	P	EA	1	458		31-Aug-81					
6	353-0380	O IC, LM380 AUDIO POWER AMPLIFIER	RI	P	1	P	C	P	EA	1	831		18-Jan-82					
6	342-0030	MA IC ROM SYNCHROM WITH BURN-IN	RI	P	1	P	C	P	EA	1	449		10-Aug-81					
6	341-0030	O ROM, SYNCHROM	RI	P	1	P	C	P	EA	1	831		18-Jan-82					
6	342-0031	MA IC ROM BOOT WITH BURN-IN	RI	P	1	P	C	P	EA	1	449		10-Aug-81					
6	341-0031	I ROM, BOOT	RI	P	1	P	C	P	EA	1	831		18-Jan-82					
6	338-6522	A IC 6522 I/F ADAPTER TEST&BURN-IN	RI	P	1	P	C	P	EA	1	831		18-Jan-82					
6	337-6522	MO IC, 6522 VERSATILE I/F ADAPTER	RI	P	1	P	C	P	EA	1	532		10-Aug-81					
6	338-0002	A ASYN COMM INT 6551A TST & BRN IN	RI	P	1	P	B	P	EA	2	877		18-Jan-82					
6	337-0002	A IC, 6551A ASYN COMM I/F ADAP. SELE.	RI	P	1	P	B	P	EA	2	449		31-Aug-81					
6	316-8304	O IC, 8304B 8-BIT TRI-ST. TST&BRN IN	RI	P	1	P	B	P	EA	1	736		31-Aug-81					
6	330-0556	A IC, 8304B 8-BIT TRI-STATE	RI	P	1	P	B	P	EA	1	831		18-Jan-82					
6	329-0556	A IC, 556 DUAL TIMER TEST&BURN-IN	RI	P	1	P	B	P	EA	1	449		10-Aug-81					
6	302-9334	B IC, 9334, TESTED & BURNED-IN	RI	P	1	P	C	P	EA	3	831		18-Jan-82					
6	301-9334	A IC, 9334	RI	P	1	P	C	P	EA	3	449		10-Aug-81					
6	301-0259	A IC, 74259	RI	P	1	P	B	P	EA	3	802		18-Jan-82					
6	360-1488	A IC 1488 QUAD LINE DRIVER TST&BRN-IN	RI	P	1	P	B	P	EA	3	802		18-Jan-82					
6	359-1488	O IC, 1488 QUAD LINE DRIVER	EX	P	7	P	B	P	EA	0	802		18-Jan-82					
6	744-0708	A IC 9708 4-BIT TESTED & BURN IN	RI	P	1	P	C	P	EA	1	449		10-Aug-81					

PRINTED 23-JUL-82 13:38

PERSONAL COMPUTER SYSTEMS

BIL VERSION 29-Mar-79 PAGE 15

IDENTIFIED BILLS OF MATERIAL

EFFECTIVITY DATE: ALL PARENT PART: 610-8156 INACT.F0,256K APPLE IIII UM: EA ASSEMBLY QTY: 1
 ERC: B SRCE CODE: A TYPE: * ABC: A PLAN CODE: U DATE: 08-JUL-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	BT	SA	P	UM	EXTENDED -QTY PER	ECN	START DATE	CHG	B/N	CLOSE DATE	B/N
7	355-9708	0 IC, 9708 6-CH, 8-BIT A TO D	RI	P	1	P	B	P	EA	1	449	10-Aug-81		
6	360-1489	A IC 1489 LINE REC. TEST&BURN-IN	A	*	C	P	EA	1	831	18-Jan-82				
7	359-1489	0 IC, 1489 LINE RECEIVER	RI	P	1	P	C	EA	1	449	10-Aug-81			
6	372-3904	A TRANSISTOR,NPN 8M, 4AMP, 2N3904	RP	P	1	P	C	EA	4	C026	18-Jan-82			
6	111-0003	0 RES ARRAY D/A CONVERTER 7 RES 8 PIN	RP	P	1	P	C	EA	1	426	18-Jan-82			
6	111-0025	A RESISTOR ARRAY, 8IP, 7 RES, SPECIAL	P	*	P	EA		1	923	18-Jan-82				
6	135-9101	A CAP, 10UF +80-20% Z5U/Y5V 50V	RP	P	1	P	C	EA	41	433	18-Jan-82			
6	191-5501	B CHOKE, 27UH 10X	RP	P	1	P	C	EA	1	479	18-Jan-82			
6	342-0028	A IC, PROM, STATE MACH P&A W/BURN-IN	A	*	C	EA		1	831	18-Jan-82				
7	341-0028	1 IC, PROM, STATE MACHINE, P&A	RL	A	1	P	C	EA	1	454	10-Aug-81			
8	335-0471	A PROM, 748471	RL	A	1	P	C	EA	1					
6	334-0005	A IC 1024 X 4 STATIC RAM 1BT & BRN IN	A	*	C	EA		2	877	18-Jan-82				
7	334-2114	0 IC, 1024 X 4 STATIC RAM 2114	RL	P	1	P	C	EA	2	449	31-Aug-81			
6	306-0133	A IC, 74LS133, TESTED & BURNED-IN	RL	A	*	C	EA	2	807	18-Jan-82				
7	305-0133	A IC, 74LS133	RI	A	1	P	C	EA	2	442	15-Jul-81			
6	820-0043	0 PCB, MAIN LOGIC BD 82 A3	RP	P	7	P	A	EA	1	503	18-Jan-82			
6	101-4112	A RES, 1/4W 5% 1.1K OHM	RP	P	1	P	C	EA	2	426	18-Jan-82			
6	101-4104	A RES, 1/4W 5% 100K OHM	RP	P	1	P	C	EA	3	426	18-Jan-82			
6	126-5102	0 CAP, 10UF 16V	RP	P	1	P	C	EA	2	426	18-Jan-82			
6	197-0001	B CRYSTAL, 14.318630 MHz	RI	P	1	P	C	EA	1	487	18-Jan-82			
6	101-4102	0 CAP, 1UF 50V	RP	P	1	P	C	EA	15	426	18-Jan-82			
6	126-4102	0 CAP, 1UF 50V	RP	P	1	P	C	EA	3	426	18-Jan-82			
6	101-4225	A RES, 1/4W 5% 2.2 MEG OHM	RP	P	1	P	C	EA	1	426	18-Jan-82			
6	131-5401	0 CAP, 20pF, 5% NPO, 50V	RP	P	1	P	C	EA	1	426	18-Jan-82			
6	372-4258	A TRANSISTOR, PNP HIGH SPD, SM, 2N4258	RP	P	1	P	C	EA	2	C026	18-Jan-82			
6	101-4335	A RES, 1/4W 5% 3.3 MEG OHM	RP	P	1	P	C	EA	2	426	18-Jan-82			
6	197-0004	0 CRYSTAL, TUNING FORK, 32.768 MHz	RI	P	1	P	C	EA	1	449	18-Jan-82			
6	101-4302	A RES, 1/4W 5% 3K OHM	RP	P	1	P	C	EA	5	426	18-Jan-82			
6	101-4470	A RES, 1/4W 5% 47 OHM	RP	P	1	P	C	EA	3	426	18-Jan-82			
6	101-4474	A RES, 1/4W 5% 470K OHM	RP	P	1	P	C	EA	1	426	18-Jan-82			
6	101-4473	A RES, 1/4W 5% 47K OHM	RP	P	1	P	C	EA	2	426	18-Jan-82			
6	515-0053	MB CONN, STRAIGHT HEADER 2 PIN	RP	P	1	P	C	EA	1	532	18-Jan-82			
6	371-4148	A DIODE, 1N4148	RP	P	1	P	C	EA	5	731	18-Jan-82			
6	519-0011	0 CONNECTOR, 9 PIN D	RC	P	1	P	C	EA	2	467	18-Jan-82			
6	519-0038	A CONN, 15 PIN D RT ANGLE PC MOUNT	RC	P	1	P	C	EA	2	467	18-Jan-82			
6	519-0001	A JACK, PHONO RT ANG (MON) NTT 333-15	RP	P	1	P	C	EA	1	725	18-Jan-82			
6	515-0002	0 JACK, PHONE	RP	P	1	P	C	EA	1	784	18-Jan-82			
6	519-0014	0 CONNECTOR, HEADER 25 PIN	RC	P	1	P	C	EA	1	467	18-Jan-82			
6	519-0016	A CONN, 26 PIN HEADER W/O MTO EARS	RC	P	1	P	C	EA	2	467	18-Jan-82			
6	519-0017	A CONN, 26 PIN HEADER W/MTO EARS	RC	P	1	P	C	EA	2	523	18-Jan-82			
6	155-5102	A COIL, 10UH RADIAL	RC	P	1	P	C	EA	1	523	18-Jan-82			
6	519-0018	0 CONNECTOR, 25 PIN D	RC	P	1	P	C	EA	2	DC44	18-Jan-82			
6	378-0001	B LED, RED	RP	P	1	P	C	EA	1	467	18-Jan-82			
6	511-1401	C SOCKET, IC 14 PIN	RP	P	1	P	C	EA	1	DC53	18-Jan-82			
6	511-1601	C SOCKET, IC 16 PIN	RP	P	1	P	C	EA	30	799	18-Jan-82			
6	511-1801	C SOCKET, IC 18 PIN	RP	P	1	P	C	EA	36	799	18-Jan-82			
6	511-2001	C SOCKET, IC 20 PIN	RP	P	1	P	C	EA	9	799	18-Jan-82			
6	511-2401	C SOCKET, IC 24 PIN	RP	P	1	P	C	EA	11	799	18-Jan-82			
6	511-2801	C SOCKET, IC 28 PIN	RP	P	1	P	C	EA	4	799	18-Jan-82			
6	511-4001	C SOCKET, IC 40 PIN	RP	P	1	P	C	EA	1	799	18-Jan-82			

BIL VERSION 29-Mar-79 PAGE 16

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:39

INDENTED BILLS OF MATERIAL

EFFECTIVITY DATE: ALL

PARENT PART: 610-8156 INACT.FG.256K APPLE III UM: EA PROD CODE: U ECN: A116 ASSEMBLY QTY: 1
 SRC CODE: A TYPE: * ABC: A PLAN CODE: U DATE: 08-Jul-82

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR	S	T	B	A	P	UM	EXTENDED QTY PER	ECN	START DATE	S/N	CLOSE DATE	S/N
			CD	C	Y	P	B	L	UM		CHG	DATE		DATE	
6	342-0056	A IC. PROM, CABB65.1 WITH BURN-IN	A	*					C	P	EA	831	18-Jan-82		
7	341-0056	O IC. PROM CABB65.1	RL	A	1	P	C	P	EA		503	10-Aug-81			
8	335-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL	P	1	P	B	P	EA		BARA	27-Oct-80			
6	342-0043	A IC PROM U174 WITH BURN-IN	A	*					C	P	EA	831	18-Jan-82		
7	341-0043	O PROM, 1024 X 4 U174	RL	A	1	P	C	P	EA		449	10-Aug-81			
8	335-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL	P	1	P	B	P	EA		BARA	18-Jan-82			
6	342-0045	A IC. PROM, U176.2 WITH BURN-IN	A	*					C	P	EA	831	18-Jan-82		
7	341-0045	O PROM, 1024 X 4 U176.2	RL	A	1	P	C	P	EA		449	10-Aug-81			
8	335-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL	P	1	P	B	P	EA		BARA	18-Jan-82			
6	342-0046	A IC. PROM, U180 WITH BURN-IN	A	*					C	P	EA	831	18-Jan-82		
7	341-0046	O PROM, 1024 X 4 U180	RL	A	1	P	B	P	EA		449	10-Aug-81			
8	335-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL	P	1	P	B	P	EA		BARA	18-Jan-82			
6	342-0055	MA IC. PROM, U175-65 WITH BURN-IN	A	*					C	P	EA	831	18-Jan-82		
7	341-0055	O PROM, 1024 X 4 U180	RL	A	1	P	B	P	EA		503	10-Aug-81			
8	335-0003	O PROM, 1024 X 4 UNPROGRAMMED	RL	P	1	P	B	P	EA		BARA	27-Oct-80			
6	513-0002	D CONNECTOR, EDGE 25/50 PIN (NO TABS)	RP	P	1	P	C	P	EA	4	760	18-Jan-82			
6	515-0103	A HEADER, RIGHT ANGLE 10 PINS	RP	P	1	P	C	P	EA	1	877	18-Jan-82			
6	306-0260	A IC, 74LS260, TESTED & BURNED-IN	A	*					C	P	EA	807	18-Jan-82		
7	305-0260	A IC, 74LS260	RI	P	1	P	B	P	EA	1	442	15-Jul-81			
6	131-3101	O CAP, 10pF 10% NPO 50V	RP	P	1	P	C	P	EA	1	426	18-Jan-82			
6	126-6404	A CAP, 220uF 16V	A	*					C	P	EA	C103	24-May-82		
6	151-5101	O CHDME, 10uH 10%	RP	P	1	P	B	P	EA	2	442	18-Jan-82			
6	132-6401	O CAP, 220pF 10% Z5F 50V	RP	P	1	P	C	P	EA	4	426	18-Jan-82			
6	372-3906	A TRANSISTOR, PNP 8M & AMP, 2N3906	RP	P	1	P	C	P	EA	1	C026	18-Jan-82			
6	101-4152	A RES 1/4W 5% 1.5K OHM	RP	P	1	P	C	P	EA	3	426	18-Jan-82			
6	101-4682	A RES 1/4W 5% 6.8K OHM	RP	P	1	P	C	P	EA	2	426	18-Jan-82			
6	519-0032	O CONN, 10 PIN FRICTION LOCK	RC	P	1	P	C	P	EA	1	442	18-Jan-82			
6	306-0132	A IC, 74LS132, TESTED & BURNED-IN	A	*					C	P	EA	807	18-Jan-82		
7	305-0132	A IC, 74LS132	RI	P	1	P	B	P	EA	1	442	15-Jul-81			
6	375-0005	O DIODE, 1N5712 SCHOTTKY BARRIER	RP	P	1	P	C	P	EA	1	449	18-Jan-82			
6	111-0018	O RES ARRAY 7 X 330 OHM	RP	P	1	P	C	P	EA	1	426	18-Jan-82			
6	111-0017	O RES ARRAY 9 X 3.3K OHM	RP	P	1	P	C	P	EA	2	426	18-Jan-82			
6	101-4330	A RES 1/4W 5% 33 OHM	RP	P	1	P	C	P	EA	4	426	18-Jan-82			
6	101-4101	A RES 1/4W 5% 100 OHM	RP	P	1	P	C	P	EA	2	426	18-Jan-82			
6	111-0014	O RES ARRAY 9 X 1K OHM	RP	P	1	P	C	P	EA	2	426	18-Jan-82			
6	101-4472	A RES 1/4W 5% 4.7K OHM	RP	P	1	P	C	P	EA	1	426	18-Jan-82			
6	101-4153	A RES 1/4W 5% 15K OHM	RP	P	1	P	C	P	EA	4	426	18-Jan-82			
6	101-4241	A RES 1/4W 5% 240 OHM	RP	P	1	P	C	P	EA	1	426	18-Jan-82			
6	101-4105	A RES, 1/4W 5% 1M OHM	RP	P	1	P	C	P	EA	2	426	18-Jan-82			
6	101-4301	A RES, 1/4W 5% 300 OHM	RP	P	1	P	C	P	EA	3	426	18-Jan-82			
6	101-4471	A RES 1/4W 5% 470 OHM	RP	P	1	P	C	P	EA	3	426	18-Jan-82			
6	125-5401	A CAP, 22uF 16V	RP	P	1	P	C	P	EA	1	433	18-Jan-82			
6	511-0801	C SOCKET, 1C 8 PIN	RP	P	1	P	C	P	EA	1	799	18-Jan-82			
6	132-6101	O CAP, 100pF 20% Z5F 50V	RP	P	1	P	C	P	EA	3	426	18-Jan-82			
6	101-4181	A RES, 1/4W 5% 180 OHM	RP	P	1	P	C	P	EA	1	426	18-Jan-82			
6	119-2101	MA CAP, .01uF 10% 100V	RP	P	1	P	C	P	EA	1	449	18-Jan-82			
6	101-4125	A RES 1/4W 5% 1.2 MEG OHM	RP	P	1	P	C	P	EA	1	426	18-Jan-82			
6	133-2401	B CAP, .022uF 20% Y5P 25V	RP	P	1	P	C	P	EA	2	433	18-Jan-82			

PRINTED 23-JUL-82 13:39

PERSONAL COMPUTER SYSTEMS

BIL VERSION 29-Mar-79 PAGE 17

INDENTED BILLS OF MATERIAL

EFFECTIVITY DATE: ALL

INACT.F0, 256K APPLE III UM: EA
 ERC: B SRCE CODE: A TYPE: * ABC: A
 ECN: A116 ASSEMBLY QTY: 1
 DATE: 08-JUL-82 PLAN CODE: U

LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR ST B A P	UM: EA	EXTENDED QTY PER	ECN	START DATE	ECN	CHG	DATE	CLOSE DATE	S/N	S/N	B/N
6	111-0019	O RES ARRAY 5 X 1K OHM	RP P 1 P C P EA		3	426	18-Jan-82							
6	308-0374	A IC, 748374, TESTED & BURNED-IN	RI P 1 P B P EA		1	807	18-Jan-82							
6	308-0000	A IC, 74800, TESTED & BURNED-IN	RI P 1 P B P EA		1	807	18-Jan-82							
6	307-0000	A IC, 74800 QUAD NAND GATE	RI P 5 P B P EA		1	C063	15-Jul-81							
6	342-0035	MA IC ROM KYBD ENC. A3 BRN-IN & TBT	RI P 5 P B P EA		1	886	18-Jan-82							
6	341-0035	B ROM, KEYBOARD ENCODER	RL P 1 P C P EA		1	683	21-Sep-81							
6	111-0001	O RES ARRAY 7 X 1K OHM	RP P 1 P C P EA		3	426	18-Jan-82							
6	131-5701	O CAP, 47PF 5% N470 50V	RP P 1 P C P EA		1	426	18-Jan-82							
6	112-0001	B RES, INDIV. NTRK. A8 OHM/5RES/10PIN	RP P 1 P C P EA		1	609	18-Jan-82							
6	376-0001	A TRANSISTOR, MPB-US1	RP P 1 P C P EA		1	DC3	18-Jan-82							
6	119-2701	A CAP, .047UF 10% 100V	RP P 1 P C P EA		2	426	18-Jan-82							
6	101-4103	A RES 1/4W 5% 10K OHM	RP P 1 P C P EA		3	426	18-Jan-82							
6	101-4270	A RES 1/4W 5% 27 OHM	RP P 1 P C P EA		4	426	18-Jan-82							
6	101-4121	A RES 1/4W 5% 120 OHM	RP P 1 P C P EA		7	503	18-Jan-82							
6	750-0006	B BUS BAR, CAPACITIVE .2UF	RP P 1 P C P EA		4	503	18-Jan-82							
6	562-2405	A WIRE, #24 WYARN GREEN SOLID	EX X 7 P C P FT		4	426	18-Jan-82							
6	101-4432	A RES 1/4W 5% 4.3K OHM	RP P 1 P C P EA		1	426	18-Jan-82							
6	111-0006	O RES, ARRAY SPECIAL BR, 9P	RP P 1 P C P EA		1	426	18-Jan-82							
6	831-0100	A RIVET, PLASTIC .125	X * * C P EA		8	778	18-Jan-82							
6	306-0014	A IC, 74LS14, TESTED & BURNED-IN	RI P 1 P B P EA		1	442	15-Jul-81							
7	305-0014	A IC, 74LS14	RI P 1 P B P EA		1	442	15-Jul-81							
6	136-3101	MC CAP, .1UF 10% XR7 50V	RP P 1 P C P EA		4	C082	18-Jan-82							
6	342-0061	A I.C. PROM RA865 WITH BURN-IN	RL P 1 C P EA		6	867	18-Jan-82							
6	341-0061	A IC/PROM, RA865	RL P 1 C P EA		1	645	24-Aug-81							
6	342-0063	A IC/PROM, CAS8256 WITH BURN-IN	RL P 1 C P EA		1	867	18-Jan-82							
6	341-0063	A IC/PROM, CAS8256	RL P 1 C P EA		1	645	24-Aug-81							
6	825-0233	A LABEL, PCB SERIALIZATION	RL P 1 C P EA		1	923	18-Jan-82							
6	942-0197	A BAG, POLY 12X18 ANTI-STAT	P * * * U EA		1	A024	26-Apr-82							
6	879-0005	A CONTACT PROTECTOR	P * * * P EA		2	C132	26-Jul-82							
6	126-6402	O CAP, 220UF 16V	RP P 1 P C P EA		4	426	18-Jan-82							23-May-82
4	610-4256	A ASSY, PCB, TBT, 5V MEMORY 256K A3	EX X 7 C P EA		1	714	26-Apr-82							
4	865-0007	A BUMPER, HEMISPHERE .2" HIGH	EX X 7 C P EA		11	661	26-Apr-82							
4	805-0099	B BOTTOM COVER, MAIN BOARD 2 A3	EX X 7 C P EA		1	972	26-Apr-82							
5	860-0018	A STANDOFF M3.5X0.6 BLD THD .200" L9	EX X 1 P C P EA		6	559	07-Nov-81							
4	410-1410	B SCREW, M3.5 X .6 X 10MM PHMS	EX X 1 C P EA		6	C086	26-Apr-82							
4	860-0102	B SPACER, NYLON 125LX1401.D X.3100.D	EX X 7 C P EA		6	C095	26-Apr-82							
3	830-0031	A TIE WRAP, 3.800L0 WHITE	EX X 7 C P EA		1	701	26-Apr-82							
3	420-1001	B SCREW, M3.5X6X10	EX X 1 P C P EA		18	C086	18-Jan-82							26-Apr-82
3	610-5197	A SUB ASSY, LOOD/NAMEPLATE A3	EX X 7 P C P EA		1	A035	10-May-82							
4	825-0054	C LABEL, APPLE LOOD NAMEPLATE	EX X 7 P C P EA		1	A035	10-May-82							
4	825-0066	C LABEL, MODEL NAMEPLATE A3	EX X 7 P C P EA		1	A035	10-May-82							
2	590-0029	O CABLE, MODEM ELIMINATOR	RC P 1 P B P EA		1	526	18-Jan-82							10-May-82
3	520-0020	A HOUSING, CONN D-TYPE RECEPT 25 PIN	RP P 7 P D P EA		1	482	12-Dec-80							
3	515-0023	A CONTACT, SOCKET (D-TYPE)	RP P 1 P D P EA		9	482	12-Dec-80							
3	590-0070	A CABLE, ROUND B CONDUCTOR	RC P 1 P D P FT		1	526	12-Dec-80							
3	515-0032	A CONTACT, PIN (D-TYPE)	RP P 1 P D P EA		1	482	12-Dec-80							
3	520-0030	A HOUSING, CONN D-TYPE 25 PIN	EX P 7 P D P EA		1	DC53	12-Dec-80							
2	942-0047	B BAG, PLASTIC 8" X 12"	EX X 1 P C P EA		1	C137	18-Jan-82							08-Jul-82
2	942-0047	C INACT FORMER OF THIS QUANTITY BACKED	EX X 1 P C P EA		1	C137	18-Jan-82							

BIL VERSION 29-Mar-79 PAGE 18

PERSONAL COMPUTER SYSTEMS

PRINTED 23-Jul-82 13:40

EFFECTIVITY DATE: ALL

INDENTED BILLS OF MATERIAL

PARENT PART: 610-8156

INACT. FG. 256K APPLE III
ERC: B BRCE CODE: A TYPE: *

UM: EA PROD CODE: U
ABC: A PLAN CODE: U

ECN: A116 ASSEMBLY QTY: 1
DATE: 08-Jul-82

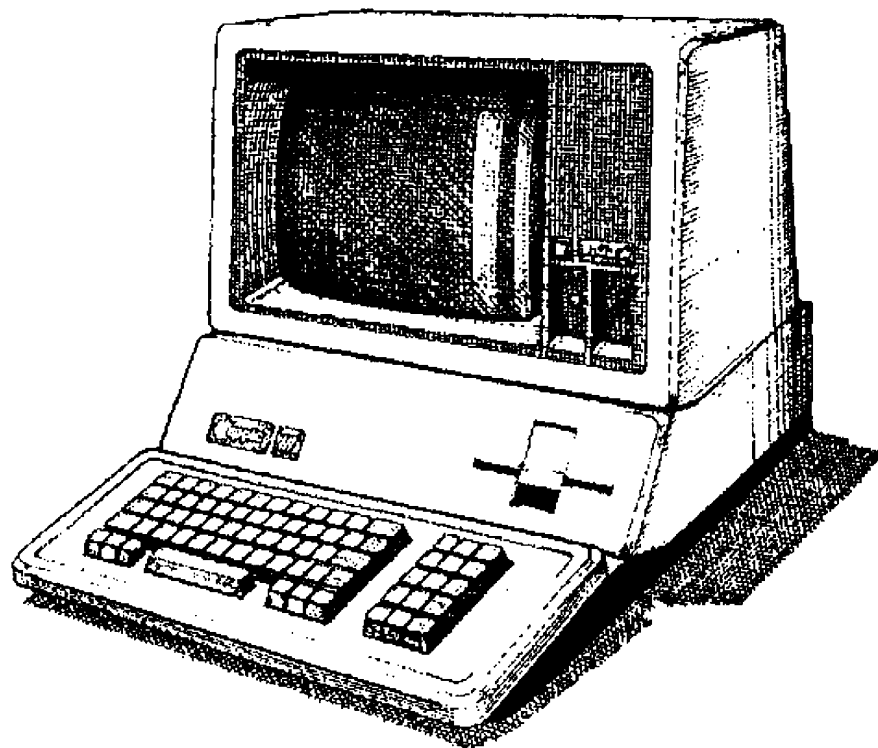
LEVEL	COMPONENT PART NUMBER	E PART DESCRIPTION	PR CD	T B C Y P B L U M	A P	EXTENDED QTY PER	ECN CHO	START DATE	B/N	CLOSE DATE	S/N		
2	942-0079	B INACT. BOX, FG APPLE III SYSTEM	EX	X	7	P	C	P	EA	1	A116	18-Jan-82	08-Jul-82
2	945-0008	A TAPE, SEALING GLASS WEB #341 A3	EX	X	7	P	B	P	RL	0.0100	474	18-Jan-82	08-Jul-82
2	945-0009	A TAPE, 3M #218 3/4" WIDE	EX	X	1	P	C	P	RL	0.0050	C145	18-Jan-82	08-Jul-82
2	030-0241	A CUSTOMER LETTER, A3 CLOCK CHIP	EX	X	*	P	*	P	EA	1	442	18-Jan-82	26-Apr-82
2	944-0059	O BAG, POLY, ANTI-STATIC, 24X36	EX	X	7	P	C	P	EA	1	442	26-Apr-82	08-Jul-82

END OF REPORT



Apple /// Computer Information

Apple /// Service Reference Manual



Section II of II • Servicing Information

Chapter 15 • Wire List

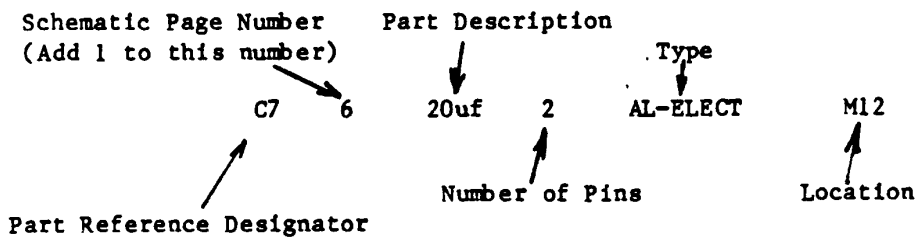
Written by Apple Computer • 1982



THE APPLE /// WIRE LIST

The following list is to be used for finding circuit components which are connected to each other.

INTEPRETING WIRE LIST NOMENCLATURE



Note: This wire list has schematic page numbers for the old Apple /// Schematic. Please add 1 to the schematic page number for the correct page.



PART REFERENCE DESIGNATOR
SYMBOLIC SHEET #3
PART DESCRIPTION
PIN LEADS

A/// WIRE LIST

B1 2 BRIDGE 2
 1 PCASO.3*
 P1-9 U3-12 B1-1
 2 RASO.3*
 J17-12 R58-2 B1-2
 C1 4 10U 2 AL-ELECT B1
 1 Q2-1 C1-1 R4-2
 2 GND
 C2 4 .1U 2 MON A3
 1 PWRDN*
 U72-23 Q3-3 C2-1 R6-2
 2 GND
 C3 4 3-30P 3 A3
 1 GND
 2 GND
 3 Y2-1 U72-10 C3-3
 C4 4 20P 2 CER A3
 1 C4-1 U72-11 Y2-2
 2 GND
 C5 5 47P 2 CER N9
 1 P3-9 R55-2 C5-1 L1-1
 2 GND
 C7 6 22U 2 AL-ELECT M12
 1 +5V
 2 DT1M
 R31-1 U164-10 C7-2 U96-2 U96-6
 C9 7 .022U 2 CER M10
 1 U105-6 R39-2 C9-1
 2 GND
 C10 7 1U 2 ELECT
 1 SUMSND
 R35-2 P9-1 C10-1 R78-2
 2 PEKTSFK
 U172-19 C10-2
 C12 7 10U 2 AL-ELECT
 1 +12V
 2 GND
 C13 7 1U 2 ELECT N10
 1 C13-1 U103-8
 2 GND
 C15 7 .01U 2 MYLAR M11



```

1          TCAP
          C15-1 U105-4
2          GND

C16 8 47P 2 CER H14
1          C16-1 U107-1
2          U107-2 C16-2 R47-2

C17 8 .1U 2 MON M7
1          U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1
          X3-1 C75-1
2          U164-6 C17-2 X5-2 C75-2

C18 8 .05U 2 CER M8
1          U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1
          X3-1 C75-1
2          GND

C20 8 .1U 2 MON
1          UPRST*
          C20-1 U113-6 U113-2 R44-2 X4-2 X6-1
2          GND

C21 8 .1U 2 MON A5
1          U113-8 R46-2 U113-12 C21-1
2          GND

C22 8 .1U 2 MON G12
1          U107-31 C22-1
2          GND

C23 9 .1U 2 MON B14
1          Q10-2 R90-2 R91-1 C23-1 R49-1
2          GND

C25 1 .1U 2 MON M7
1          +5FV
2          GNDF

C26 1 .1U 2 MON
1          -5FV
2          GNDF

C27 1 .1U 2 MON N11
1          +12FV
2          GNDF

C28 1 .1U 2 MON M10
1          -12FV
2          GNDF

C29 1 .1U 2 MON J2
    
```

+5FV

-5FV

+12FV

-12FV



	1		+12V
	2		GND
C30	1	.1U 2 MON	J2
	1		+5V
	2		GND
C31	1	.1U 2 MON	K1
	1		+12V
	2		GND
C32	1	.1U 2 MON	K2
	1		+12V
	2		GND
C33	1	.1U 2 MON	K4
	1		+12V
	2		GND
C34	1	.1U 2 MON	K6
	1		+12V
	2		GND
C35	1	.1U 2 MON	M1
	1		-12V
	2		GND
C36	1	.1U 2 MON	M1
	1		-5V
	2		GND
C37	1	.1U 2 MON	A2
	1		+5V
	2		GND
C38	1	.1U 2 MON	K8
	1		+5V
	2		GND
C39	1	.1U 2 MON	K10
	1		+5V
	2		GND
C40	1	.1U 2 MON	K12
	1		+5V
	2		GND
C41	1	.1U 2 MON	M6
	1		-12V
	2		GND
C42	1	.1U 2 MON	G4



	1		+5V
	2		GND
C43	1	.1U 2 MON	G12
	1		+5V
	2		GND
C44	1	.1U 2 MON	G12
	1		+5V
	2		GND
C45	1	.1U 2 MON	F7
	1		+5V
	2		GND
C46	1	.1U 2 MON	A12
	1		+5V
	2		GND
C47	1	.1U 2 MON	E14
	1		+5V
	2		GND
C48	1	.1U 2 MON	M6
	1		-5V
	2		GND
C50	1	.1U 2 MON	D7
	1		+5V
	2		GND
C51	1	.1U 2 MON	B8
	1		+5V
	2		GND
C52	1	.1U 2 MON	A6
	1		+5V
	2		GND
C53	1	.1U 2 MON	E4
	1		+5V
	2		GND
C54	1	.1U 2 MON	D4
	1		+5V
	2		GND
C55	1	.1U 2 MON	C2
	1		+5V
	2		GND
C56	1	.1U 2 MON	B10



	1		+5V	
	2		GND	
C57	1	.1U 2 MON	B12	
	1		+5V	
	2		GND	
C58	1	.1U 2 MON	A10	
	1		+5V	
	2		GND	
C59	9	10P 2 CER	A8	
	1		CS6522	
			R68-2 U73-24 C59-1 U97-24	
	2		GND	
C60	1	220U 2 AL-ELECT	N11	
	1		+5FV	
	2		GND	
C61	1	220U 2 AL-ELECT	N7	
	1		GND	
	2		-5FV	
C62	1	220U 2 AL-ELECT	N12	
	1		+12FV	
	2		GND	
C63	1	220U 2 AL-ELECT	M10	
	1		GND	
	2		-12FV	
C64	1	.1U 2 MON	F14	
	1		+5V	
	2		GND	
C65	1	.1U 2 MON	A4	
	1		+5V	
	2		GND	
C66	1	220P 2 CER	M5	
	1		R96-2 C66-1	
	2		GND	
C67	1	220P 2 CER	L1	
	1		R93-2 C67-1	
	2		GND	
C68	1	220P 2 CER	L1	
	1		R94-2 C68-1	
	2		GND	



C69 7 .1U 2 MON A8
 1 +5V
 2 C69-2 R75-2 U181-1 U181-2

C70 7 .1U 2 MON A8
 1 R77-2 U181-8 U181-12 C70-1
 2 GND

C71 8 1U 2 ELECT M14
 1 KRESET*
 U139-12 R80-1 C71-1 U179-15 J7-15
 2 GNDF

C73 9 .1U 2 MON B14
 1 +12V
 2 GND

C74 2 100P 2 CER D12
 1 R100-2 C74-1 U4-11
 2 GND

C75 8 .1U 2 MON M6
 1 U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1
 X3-1 C75-1
 2 U164-6 C17-2 X5-2 C75-2

C76 7 .1U 2 MON M8
 1 U105-8 R37-2 R38-1 C76-1
 2 GND

C77 1 220P 2 CER L1
 1 R95-2 C77-1
 2 GND

C78 7 100P 2 CER
 1 C78-1 U103-2 R34-2 R36-1
 2 GND

C79 1 .1U 2 MON
 1 +5V
 2 GND

C80 1 .1U 2 MON
 1 +5V
 2 GND

J1 1 26PIN 26 RIBBON
 1 GNDF
 2 DPHO
 J6-2 U166-9 J1-2
 3 GNDF
 4 DPH1



5	J6-4 U167-9 J1-4
6	GNDF
6	DPH2
7	J6-6 U166-8 J1-6
7	GNDF
8	DPH3
8	J6-8 U167-8 J1-8
9	-12FV
10	WRREQ
11	J6-10 U166-7 J1-10
11	+5FV
12	+5FV
13	+12FV
14	ENBL1.E*
14	J1-14 U166-5
15	+12FV
16	RDDATA
16	J6-16 U166-4 J1-16
17	+12FV
18	WRDATA
18	J6-18 U166-3 J1-18
19	+12FV
20	WRPROT
20	J6-20 U167-4 J1-20
21	ENBL3.E*
21	J1-21 U166-2
22	ENBL2.E*
22	J1-22 U167-6
23	AI1*
23	U167-5 J1-23 J6-23
24	SIDE2/1
24	J6-24 U167-7 J1-24
25	NO CONNECTION
26	EXT*
26	J1-26 U167-2

-12FV

J2 1 9PIN 9 9D

1	GNDF
2	+5FV
3	GNDF
4	XO
5	U169-9 J2-4
5	SW2
6	U169-7 J2-5
6	+12FV
7	GNDF
8	YO
8	U169-6 J2-8
9	SWO
9	U169-8 J2-9

J3 1 9PIN 9 9D



1	GNDF
2	+5FV
3	GNDF
4	X1/SER
	J3-4 U169-3
5	SW1/MGNSW
	J3-5 U169-2
6	+12FV
7	GNDF
8	Y1/XCO
	J3-8 U169-4
9	SW3/SCO
	J3-9 U169-5

J4 1 25PIN 25 25D

1	GNDF
2	TXD
	J4-2 U172-9
3	DATA IN
	J4-3 U172-8
4	RTS
	J4-4 U172-7
5	CTS
	J4-5 U172-6
6	DSR
	J4-6 U172-5
7	GNDF
8	DCD
	J4-8 U172-3
9	NO CONNECTION
10	NO CONNECTION
11	NO CONNECTION
12	NO CONNECTION
13	NO CONNECTION
14	NO CONNECTION
15	NO CONNECTION
16	NO CONNECTION
17	NO CONNECTION
18	NO CONNECTION
19	NO CONNECTION
20	DTR
	J4-20 U172-4
21	NO CONNECTION
22	NO CONNECTION
23	NO CONNECTION
24	NO CONNECTION
25	NO CONNECTION

J5 1 15PIN 15 15D

1	GNDF
2	XRGB4
	J5-2 P17-3 P18-3



3	XSYNC
	J5-3 P17-9 P18-6
4	PDINT*
	J5-4 U72-14
5	XRGB1
	J5-5 P17-7 P18-5
6	GNDF
7	-5FV
8	+12FV
9	XRGB2
	J5-9 P17-5 P18-4
10	XRGB8
	J5-10 P17-1 P18-2
11	B&WVID
	J5-11 R15-2 J10-1 P18-7
12	NTSC
	J5-12 R12-2 P18-8
13	GNDF
14	-12FV
15	+5FV

-5FV

J6 1 26PIN 26 RIBBON

1	GNDF
2	DPHO
	J6-2 U166-9 J1-2
3	GNDF
4	DPH1
	J6-4 U167-9 J1-4
5	GNDF
6	DPH2
	J6-6 U166-8 J1-6
7	GNDF
8	DPH3
	J6-8 U167-8 J1-8
9	-12FV
10	WRREQ
	J6-10 U166-7 J1-10
11	+5FV
12	+5FV
13	+12FV
14	ENBL1.I*
	J6-14 U166-6
15	+12FV
16	RDDATA
	J6-16 U166-4 J1-16
17	+12FV
18	WRDATA
	J6-18 U166-3 J1-18
19	+12FV
20	WRPROT
	J6-20 U167-4 J1-20
21	NO CONNECTION



22 NO CONNECTION
 23 AII*
 U167-5 J1-23 J6-23
 24 SIDE2/1
 J6-24 U167-7 J1-24
 25 NO CONNECTION
 26 INT*
 J6-26 U167-3

J7 1 26PIN 26 RIBBON

1 KY0
 J7-1 U107-17
 2 KY1
 J7-2 U107-18
 3 KVCC
 J7-3 Q9-2
 4 KY2
 J7-4 U107-19
 5 APPLEII*
 J7-5 P11-5 U106-12 U164-5
 6 KY3
 J7-6 U107-20
 7 APPLEI*
 J7-7 P11-4 U111-3
 8 KY4
 J7-8 U107-21
 9 CAPLCK*
 J7-9 P11-3 U109-13
 10 KY5
 J7-10 U107-22
 11 CONTROL*
 J7-11 U179-13 U107-28 U109-10 P11-2
 12 KY8
 J7-12 U107-25
 13 GNDP
 14 KX0
 J7-14 U171-9
 15 KRESET*
 U139-12 R80-1 C71-1 U179-15 J7-15
 16 KX2
 J7-16 U171-8
 17 KX7
 J7-17 U171-7
 18 KX1
 J7-18 U171-6
 19 KX5
 J7-19 U171-5
 20 KX3
 J7-20 U171-4
 21 KX4
 J7-21 U171-3
 22 KY9



23 J7-22 U107-26
 KY6
 J7-23 U107-23
 24 SHIFT*
 J7-24 P11-6 U107-29
 25 KY7
 J7-25 U107-24
 26 KX6
 J7-26 U171-2

J8 1 10PIN 10 PWR G1
 1 -12V
 2 GND
 3 GND
 4 GND
 5 GND
 6 -5V
 7 +5V
 8 +5V
 9 +12V
 10 +12V

J9 1 2PIN 2 MOLEX M11
 1 AUDIO
 J9-1 R69-1
 2 +5V

J10 1 2PIN 3 RCA
 1 B&WVID
 J5-11 R15-2 J10-1 P18-7
 2 GNDF
 3 GNDF

J11 1 3PIN 3 MINPHONE
 1 EXTSPK1
 J11-1 R34-1
 2 EXTSPK2
 J11-2 U172-2
 3 GNDF

J12 1 50PIN 50 EDG
 1 IOSEL1*
 U74-14 J12-1
 2 A0
 J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
 3 A1
 J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
 U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
 4 A2
 J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
 U177-2 U94-2 U75-2 U73-36 U63-14



5 A3
 J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
 U97-35 U177-3 U94-3 U75-3 U73-35 U63-12

6 A4
 J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
 U112-2 U63-9

7 A5
 J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
 U112-3 U63-7

8 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11

9 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3

10 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4

11 A9
 J15-11 J14-11 J13-11 J12-11 U5-4 U74-2 U71-5
 U67-7

12 A10
 J15-12 J14-12 J13-12 J12-12 U128-1 U74-3 U71-4
 U67-9

13 A11
 J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
 U150-13 U74-5 U67-12 U3-5 U174-5

14 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6

15 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
 U71-1 U70-7 U6-5 U3-6 U174-7

16 A14
 J15-16 J14-16 J13-16 J12-16 U3-7 U147-4 U70-9
 U6-16 U174-4

17 A15
 J15-17 J14-17 J13-17 J12-17 U3-4 U147-3 U70-12
 U174-16 U6-7

18 R/W*
 J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
 U176-6 U165-11 U160-11

19 PHO
 J15-19 U136-11^{HOA} J14-19 J13-19 J12-19 U124-5^{A11} U65-37
R96-1

20 IOSTRB*
 J15-20 U150-12 J14-20 J13-20 J12-20

21 RDY
 J15-21 U164-8 J14-21 J13-21 J12-21 P14-4 U65-2

22 TSADB*
 J15-22 J12-22 J14-22 J13-22 U163-3 P14-3

23 SPARE2



	J15-23	J12-23	J14-23	J13-23			
24	SPARE1						
	J15-24	J12-24	J14-24	J13-24			
25	+5V						
26	GND						
27	DMAOK						
	U144-8	J12-27	J15-27	J14-27	J13-27		
28	DMAI*						
	U176-2	J12-28	J15-28	J14-28	J13-28	U163-5	P14-2
	U153-9						
29	IONMI*						
	J15-29	U97-17	J14-29	J13-29	U139-13	J12-29	
30	IRQ1*						
	U101-14	J12-30	P2-7	U148-1			
31	IORESET*						
	J12-31	J13-31	J14-31	J15-31	P14-6	U164-2	U164-4
	U96-4	U94-15					
32	INH*						
	J15-32	J14-32	J13-32	J12-32	U176-4	U165-13	P14-5
33	-12V						
34	-5V						
35	SYNC						
	U65-7	U136-1	J15-35	U174-3	J14-35	J13-35	J12-35
36	C7M						
	J15-36	J14-36	J13-36	J12-36	U141-9	U119-2	U146-12
	U90-14						
37	Q3						
	J14-37	U85-3	J15-37	J13-37	J12-37	U117-12	R93-1
	U154-2						
38	PRE1M*						
	J12-38	J13-38	J14-38	J15-38	U123-6	R95-1	
39	CO2X*						
	J15-39	U77-13	J14-39	J13-39	J12-39		
40	PRE1M						
	J12-40	J13-40	J14-40	J15-40	U119-12	U73-25	U123-5
	U97-25	U139-10	R94-1				
41	DEVSEL1*						
	J12-41	U76-14					
42	D7						
	J15-42	J14-42	J13-42	J12-42	U68-12	J16-24	U111-12
	U101-5	U91-7	U69-12	P15-9			
43	D6						
	J15-43	J14-43	J13-43	J12-43	U68-13	J16-23	U111-9
	U91-13	U69-9	P15-8				
44	D5						
	J15-44	J14-44	J13-44	J12-44	U68-14	J16-22	U111-7
	U91-6	U69-7	P15-7				
45	D4						
	J15-45	J14-45	J13-45	J12-45	U68-15	J16-21	U111-4
	U91-14	U69-4	P15-6				
46	D3						
	J15-46	J14-46	J13-46	J12-46	U68-16	J16-20	U109-12



47 U91-5 U66-12 P15-5
 D2
 J15-47 J14-47 J13-47 J12-47 U68-17 J16-19 U109-9
 U91-15 U66-9 P15-4

48 D1
 J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7
 U91-4 U66-7 P15-3

49 D0
 J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4
 U91-16 U66-4 P15-2

50 +12V

J13 1 50PIN 50 EDG

1 IOSEL2*
 U74-13 J13-1

2 A0
 J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13

3 A1
 J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
 U101-10 U177-1 U94-1 U73-37 U75-1 U63-16

4 A2
 J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
 U177-2 U94-2 U75-2 U73-36 U63-14

5 A3
 J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
 U97-35 U177-3 U94-3 U75-3 U73-35 U63-12

6 A4
 J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
 U112-2 U63-9

7 A5
 J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
 U112-3 U63-7

8 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11

9 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3

10 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4

11 A9
 J15-11 J14-11 J13-11 J12-11 U5-4 U74-2 U71-5
 U67-7

12 A10
 J15-12 J14-12 J13-12 J12-12 U128-1 U74-3 U71-4
 U67-9

13 A11
 J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
 U150-13 U74-5 U67-12 U3-5 U174-5

14 A12



	J15-14	J14-14	J13-14	J12-14	U9-12	U155-11	U165-14
	U70-4	U71-2	U174-6				
15	A13						
	J15-15	J14-15	J13-15	J12-15	U128-4	U155-12	U165-15
	U71-1	U70-7	U6-5	U3-6	U174-7		
16	A14						
	J15-16	J14-16	J13-16	J12-16	U3-7	U147-4	U70-9
	U6-16	U174-4					
17	A15						
	J15-17	J14-17	J13-17	J12-17	U3-4	U147-3	U70-12
	U174-16	U6-7					
18	R/W*						
	J15-18	J14-18	J13-18	J12-18	U3-3	U136-9	U180-17
	U176-6	U165-11	U160-11				
19	PHO						
	J15-19	U136-11	J14-19	J13-19	J12-19	U124-5	U65-37
	R96-1						
20	IOSTRB*						
	J15-20	U150-12	J14-20	J13-20	J12-20		
21	RDY						
	J15-21	U164-8	J14-21	J13-21	J12-21	P14-4	U65-2
22	TSADB*						
	J15-22	J12-22	J14-22	J13-22	U163-3	P14-3	
23	SPARE2						
	J15-23	J12-23	J14-23	J13-23			
24	SPARE1						
	J15-24	J12-24	J14-24	J13-24			
25	+5V						
26	GND						
27	DMAOK						
	U144-8	J12-27	J15-27	J14-27	J13-27		
28	DMAI*						
	U176-2	J12-28	J15-28	J14-28	J13-28	U163-5	P14-2
	U153-9						
29	IONMI*						
	J15-29	U97-17	J14-29	J13-29	U139-13	J12-29	
30	IRQ2*						
	U101-15	J13-30	P2-6	U148-2			
31	IORESET*						
	J12-31	J13-31	J14-31	J15-31	P14-6	U164-2	U164-4
	U96-4	U94-15					
32	INH*						
	J15-32	J14-32	J13-32	J12-32	U176-4	U165-13	P14-5
33	-12V						
34	-5V						
35	SYNC						
	U65-7	U136-1	J15-35	U174-3	J14-35	J13-35	J12-35
36	C7M						
	J15-36	J14-36	J13-36	J12-36	U141-9	U119-2	U146-12
	U90-14						
37	Q3						
	J14-37	U85-3	J15-37	J13-37	J12-37	U117-12	R93-1



38 U154-2
PRE1M*
J12-38 J13-38 J14-38 J15-38 U123-6 R95-1

39 CO2X*
J15-39 U77-13 J14-39 J13-39 J12-39

40 PRE1M
J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5
U97-25 U139-10 R94-1

41 DEVSEL2*
J13-41 U76-13

42 D7
J15-42 J14-42 J13-42 J12-42 U68-12 J16-24 U111-12
U101-5 U91-7 U69-12 P15-9

43 D6
J15-43 J14-43 J13-43 J12-43 U68-13 J16-23 U111-9
U91-13 U69-9 P15-8

44 D5
J15-44 J14-44 J13-44 J12-44 U68-14 J16-22 U111-7
U91-6 U69-7 P15-7

45 D4
J15-45 J14-45 J13-45 J12-45 U68-15 J16-21 U111-4
U91-14 U69-4 P15-6

46 D3
J15-46 J14-46 J13-46 J12-46 U68-16 J16-20 U109-12
U91-5 U66-12 P15-5

47 D2
J15-47 J14-47 J13-47 J12-47 U68-17 J16-19 U109-9
U91-15 U66-9 P15-4

48 D1
J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7
U91-4 U66-7 P15-3

49 D0
J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4
U91-16 U66-4 P15-2

50 +12V

J14 1 50PIN 50 EDG

1 IOSEL3*
U74-12 J14-1

2 A0
J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
U101-11 U94-13 U73-38 U75-13 U63-18 U177-13

3 A1
J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
U101-10 U177-1 U94-1 U73-37 U75-1 U63-16

4 A2
J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
U177-2 U94-2 U75-2 U73-36 U63-14

5 A3
J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
U97-35 U177-3 U94-3 U75-3 U73-35 U63-12

6 A4



	J15-6	J14-6	J13-6	J12-6	U9-6	U77-1	U76-1
	U112-2	U63-9					
7	A5						
	J15-7	J14-7	J13-7	J12-7	U9-10	U76-2	U77-2
	U112-3	U63-7					
8	A6						
	J15-8	J14-8	J13-8	J12-8	U2-4	U77-3	U76-3
	U71-10	U63-5	U71-11				
9	A7						
	J15-9	J14-9	J13-9	J12-9	U13-10	U77-4	U76-6
	U71-7	U63-3					
10	A8						
	J15-10	J14-10	J13-10	J12-10	U67-4	U74-1	U71-6
	U2-12	U13-6	U13-4				
11	A9						
	J15-11	J14-11	J13-11	J12-11	U5-4	U74-2	U71-5
	U67-7						
12	A10						
	J15-12	J14-12	J13-12	J12-12	U128-1	U74-3	U71-4
	U67-9						
13	A11						
	J15-13	J14-13	J13-13	J12-13	U128-12	U6-6	U71-3
	U150-13	U74-5	U67-12	U3-5	U174-5		
14	A12						
	J15-14	J14-14	J13-14	J12-14	U9-12	U155-11	U165-14
	U70-4	U71-2	U174-6				
15	A13						
	J15-15	J14-15	J13-15	J12-15	U128-4	U155-12	U165-15
	U71-1	U70-7	U6-5	U3-6	U174-7		
16	A14						
	J15-16	J14-16	J13-16	J12-16	U3-7	U147-4	U70-9
	U6-16	U174-4					
17	A15						
	J15-17	J14-17	J13-17	J12-17	U3-4	U147-3	U70-12
	U174-16	U6-7					
18	R/W*						
	J15-18	J14-18	J13-18	J12-18	U3-3	U136-9	U180-17
	U176-6	U165-11	U160-11				
19	PHO						
	J15-19	U136-11	J14-19	J13-19	J12-19	U124-5	U65-37
	R96-1						
20	IOSTRB*						
	J15-20	U150-12	J14-20	J13-20	J12-20		
21	RDY						
	J15-21	U164-8	J14-21	J13-21	J12-21	P14-4	U65-2
22	TSADB*						
	J15-22	J12-22	J14-22	J13-22	U163-3	P14-3	
23	SPARE2						
	J15-23	J12-23	J14-23	J13-23			
24	SPARE1						
	J15-24	J12-24	J14-24	J13-24			
25	+5V						



26	GND						
27	DMAOK						
	U144-8	J12-27	J15-27	J14-27	J13-27		
28	DMAI*						
	U176-2	J12-28	J15-28	J14-28	J13-28	U163-5	P14-2
	U153-9						
29	IONMI*						
	J15-29	U97-17	J14-29	J13-29	U139-13	J12-29	
30	IRQ3*						
	U97-7	J14-30	P2-4	U148-4			
31	IORESET*						
	J12-31	J13-31	J14-31	J15-31	P14-6	U164-2	U164-4
	U96-4	U94-15					
32	INH*						
	J15-32	J14-32	J13-32	J12-32	U176-4	U165-13	P14-5
33	-12V						
34	-5V						
35	SYNC						
	U65-7	U136-1	J15-35	U174-3	J14-35	J13-35	J12-35
36	C7M						
	J15-36	J14-36	J13-36	J12-36	U141-9	U119-2	U146-12
	U90-14						
37	Q3						
	J14-37	U85-3	J15-37	J13-37	J12-37	U117-12	R93-1
	U154-2						
38	PRE1M*						
	J12-38	J13-38	J14-38	J15-38	U123-6	R95-1	
39	C02X*						
	J15-39	U77-13	J14-39	J13-39	J12-39		
40	PRE1M						
	J12-40	J13-40	J14-40	J15-40	U119-12	U73-25	U123-5
	U97-25	U139-10	R94-1				
41	DEVSEL3*						
	J14-41	U76-12					
42	D7						
	J15-42	J14-42	J13-42	J12-42	U68-12	J16-24	U111-12
	U101-5	U91-7	U69-12	P15-9			
43	D6						
	J15-43	J14-43	J13-43	J12-43	U68-13	J16-23	U111-9
	U91-13	U69-9	P15-8				
44	D5						
	J15-44	J14-44	J13-44	J12-44	U68-14	J16-22	U111-7
	U91-6	U69-7	P15-7				
45	D4						
	J15-45	J14-45	J13-45	J12-45	U68-15	J16-21	U111-4
	U91-14	U69-4	P15-6				
46	D3						
	J15-46	J14-46	J13-46	J12-46	U68-16	J16-20	U109-12
	U91-5	U66-12	P15-5				
47	D2						
	J15-47	J14-47	J13-47	J12-47	U68-17	J16-19	U109-9
	U91-15	U66-9	P15-4				



48 D1
 J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7
 U91-4 U66-7 P15-3
 49 D0
 J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4
 U91-16 U66-4 P15-2
 50 +12V

J15 1 50PIN 50 EDG

1 IOSEL4*
 U74-11 J15-1
 2 A0
 J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
 3 A1
 J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
 U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
 4 A2
 J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
 U177-2 U94-2 U75-2 U73-36 U63-14
 5 A3
 J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
 U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
 6 A4
 J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
 U112-2 U63-9
 7 A5
 J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
 U112-3 U63-7
 8 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11
 9 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3
 10 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4
 11 A9
 J15-11 J14-11 J13-11 J12-11 U5-4 U74-2 U71-5
 U67-7
 12 A10
 J15-12 J14-12 J13-12 J12-12 U128-1 U74-3 U71-4
 U67-9
 13 A11
 J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
 U150-13 U74-5 U67-12 U3-5 U174-5
 14 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6
 15 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15



	U71-1	U70-7	U6-5	U3-6	U174-7		
16	A14						
	J15-16	J14-16	J13-16	J12-16	U3-7	U147-4	U70-9
	U6-16	U174-4					
17	A15						
	J15-17	J14-17	J13-17	J12-17	U3-4	U147-3	U70-12
	U174-16	U6-7					
18	R/W*						
	J15-18	J14-18	J13-18	J12-18	U3-3	U136-9	U180-17
	U176-6	U165-11	U160-11				
19	PHO						
	J15-19	U136-11	J14-19	J13-19	J12-19	U124-5	U65-37
	R96-1						
20	IOSTRB*						
	J15-20	U150-12	J14-20	J13-20	J12-20		
21	RDY						
	J15-21	U164-8	J14-21	J13-21	J12-21	P14-4	U65-2
22	TSADB*						
	J15-22	J12-22	J14-22	J13-22	U163-3	P14-3	
23	SPARE2						
	J15-23	J12-23	J14-23	J13-23			
24	SPARE1						
	J15-24	J12-24	J14-24	J13-24			
25	+5V						
26	GND						
27	DMAOK						
	U144-8	J12-27	J15-27	J14-27	J13-27		
28	DMAI*						
	U176-2	J12-28	J15-28	J14-28	J13-28	U163-5	P14-2
	U153-9						
29	IONMI*						
	J15-29	U97-17	J14-29	J13-29	U139-13	J12-29	
30	IRQ4*						
	U97-6	J15-30	P2-3	U148-5			
31	IORESET*						
	J12-31	J13-31	J14-31	J15-31	P14-6	U164-2	U164-4
	U96-4	U94-15					
32	INH*						
	J15-32	J14-32	J13-32	J12-32	U176-4	U165-13	P14-5
33	-12V						
34	-5V						
35	SYNC						
	U65-7	U136-1	J15-35	U174-3	J14-35	J13-35	J12-35
36	C7M						
	J15-36	J14-36	J13-36	J12-36	U141-9	U119-2	U146-12
	U90-14						
37	Q3						
	J14-37	U85-3	J15-37	J13-37	J12-37	U117-12	R93-1
	U154-2						
38	PRE1M*						
	J12-38	J13-38	J14-38	J15-38	U123-6	R95-1	
39	C02X*						



40	J15-39 U77-13 J14-39 J13-39 J12-39 PRE1M J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5 U97-25 U139-10 R94-1
41	DEVSEL4* J15-41 U76-11
42	D7 J15-42 J14-42 J13-42 J12-42 U68-12 J16-24 U111-12 U101-5 U91-7 U69-12 P15-9
43	D6 J15-43 J14-43 J13-43 J12-43 U68-13 J16-23 U111-9 U91-13 U69-9 P15-8
44	D5 J15-44 J14-44 J13-44 J12-44 U68-14 J16-22 U111-7 U91-6 U69-7 P15-7
45	D4 J15-45 J14-45 J13-45 J12-45 U68-15 J16-21 U111-4 U91-14 U69-4 P15-6
46	D3 J15-46 J14-46 J13-46 J12-46 U68-16 J16-20 U109-12 U91-5 U66-12 P15-5
47	D2 J15-47 J14-47 J13-47 J12-47 U68-17 J16-19 U109-9 U91-15 U66-9 P15-4
48	D1 J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7 U91-4 U66-7 P15-3
49	D0 J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4 U91-16 U66-4 P15-2
50	+12V

J16 1 25PIN 25 LNGMOLEX

1	DB0 J16-1 U66-3 U80-13
2	DB1 J16-2 U66-6 U80-8
3	DB2 J16-3 U66-10 U80-14
4	DB3 J16-4 U66-13 U80-7
5	DB4 J16-5 U69-3 U80-17
6	DB5 J16-6 U69-6 U80-4
7	DB6 J16-7 U69-10 U80-18
8	DB7 J16-8 U69-13 U80-3
9	DA0 J16-9 U10-4 U84-13 U66-2
10	DA1



11	J16-10	U10-5	U84-8	U66-5			
	DA2						
12	J16-11	U10-12	U84-14	U66-11			
	DA3						
13	J16-12	U66-14	U84-7				
	DA4						
14	J16-13	U69-2	U84-17				
	DA5						
15	J16-14	U69-5	U84-4				
	DA6						
16	J16-15	U69-11	U84-18				
	DA7						
17	J16-16	U153-5	U84-3	U69-14			
	D0						
	J15-49	J14-49	J13-49	J12-49	U68-19	J16-17	U109-4
	U91-16	U66-4	P15-2				
18	D1						
	J15-48	J14-48	J13-48	J12-48	U68-18	J16-18	U109-7
	U91-4	U66-7	P15-3				
19	D2						
	J15-47	J14-47	J13-47	J12-47	U68-17	J16-19	U109-9
	U91-15	U66-9	P15-4				
20	D3						
	J15-46	J14-46	J13-46	J12-46	U68-16	J16-20	U109-12
	U91-5	U66-12	P15-5				
21	D4						
	J15-45	J14-45	J13-45	J12-45	U68-15	J16-21	U111-4
	U91-14	U69-4	P15-6				
22	D5						
	J15-44	J14-44	J13-44	J12-44	U68-14	J16-22	U111-7
	U91-6	U69-7	P15-7				
23	D6						
	J15-43	J14-43	J13-43	J12-43	U68-13	J16-23	U111-9
	U91-13	U69-9	P15-8				
24	D7						
	J15-42	J14-42	J13-42	J12-42	U68-12	J16-24	U111-12
	U101-5	U91-7	U69-12	P15-9			
25	GND						

J17 1 25PIN 25 LNGMOLEX

1	GND		
2	+5V		
3	+12V		
4	AR6		
	J17-4	U13-9	P16-5
5	AR5		
	R99-2	J17-5	
6	AR4		
	J17-6	U9-7	P16-6
7	AR3		
	J17-7	U5-9	P16-3
8	AR2		



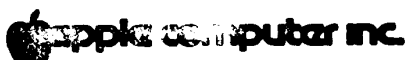
9 J17-8 U5-7 P16-2
 ARI
 R98-2 J17-9
 10 ARO
 R97-2 J17-10
 11 RAMR/W*
 J17-11 U150-8
 12 RAS0.3*
 J17-12 R58-2 B1-2
 13 RAS1.2*
 J17-13 R59-2
 14 RAS4.5*
 J17-14 R60-2
 15 RAS6.7*
 J17-15 R61-2
 16 -5V
 17 CAS0*
 J17-17 U4-2
 18 CAS1*
 J17-18 U4-5
 19 CAS2*
 J17-19 U4-6
 20 CAS3*
 J17-20 U4-9
 21 CAS4,6*
 J17-21 U4-12
 22 CAS5,7*
 U4-15 J17-22
 23 AX*
 U124-8 U2-2 U13-2 U9-2 U5-2 J17-23
 24 AR7
 U13-7 J17-24
 25 GND

J19 1 3PIN 3 MOLEX H14

1 UUTSUNK*
 J19-1 U118-9 U162-8 U120-9
 2 FIELDOUT
 U126-19 J19-2
 3 FIELDIN
 J19-3 U121-18 R63-1

J20 1 10PIN 10 MOLEX H1

1 SYNCH
 J20-1 P17-10 U126-5 P3-2 P4-6
 2 RGB2
 J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
 U89-10
 3 C3.5M
 U132-10 U141-10 U146-13 U119-7 J20-3
 4 RGB4
 P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7



```

    J20-4
5     RGB8
    J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
    U89-2
6     RGB1
    U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
    U90-12
7     FORCPAGE
    U87-6 P10-2 J20-7
8     COLRGATE
    J20-8 U147-2 U126-6
9     COLORKILL*
    U147-13 U87-11 J20-9
10    GND

J21 1 5PIN 5 MOLEX B14
    1     HPEDIS
    U135-2 J21-1 R54-1
    2     C14M*
    U141-13 U146-8 U124-11 J21-2
    3     UUTRST*
    U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
    U120-1 U118-1 U117-1
    4     UUTUPRST
    R92-2 U162-11 J21-4 U164-3
    5     GND

L1 5 27U 2 L1 M10
    1     P3-9 R55-2 C5-1 L1-1
    2     GNDF

L2 1 30U 2 FIL1 M7
    1     +5V
    2     +5FV

L3 1 10U 2 FIL2 L7
    1     -5V
    2     -5FV

L4 1 30U 2 FIL1 M7
    1     +12V
    2     +12FV

L5 1 10U 2 FIL2 L7
    1     -12V
    2     -12FV

Q1 4 3904 3 TR1 C1
    1     U72-24 Q1-1 X1-2
    2     Q1-2 R5-2
    3     +5V
    
```




Q2 4 3906 3 TRI C1
 1 Q2-1 C1-1 R4-2
 2 +5V
 3 Q3-2 Q2-3 R101-1

Q3 4 3904 3 TRI B1
 1 GND
 2 Q3-2 Q2-3 R101-1
 3 PWRDN*
 U72-23 Q3-3 C2-1 R6-2

Q4 5 3904 3 TRI M7
 1 R15-1 Q4-1 R16-1
 2 P4-1 Q4-2
 3 +5V

Q9 8 MPSU51 3 TRI L14
 1 +5FV
 2 KVCC
 3 J7-3 Q9-2
 Q9-3 R43-1 U111-10

Q10 9 4258 3 TRI B13
 1 Q11-1 R48-2 Q10-1
 2 Q10-2 R90-2 R91-1 C23-1 R49-1
 3 R52-1 Q10-3 Y1-1

Q11 9 4258 3 TRI B13
 1 Q11-1 R48-2 Q10-1
 2 Y1-2 Q11-2 R49-2
 3 Q11-3 R51-1 U146-10 U146-5 U146-2

Q12 5 3904 3 TRI L7
 1 R12-1 R13-1 Q12-1
 2 P3-1 Q12-2
 3 +5V

P1 2 1K 10 SIP10 C14
 1 +5V
 2 PCAS4,6*
 U4-13 U6-14 P1-2
 3 PCAS1*
 P1-3 U4-4 U6-12 U128-10
 4 PCAS2*
 P1-4 U4-7 U6-11
 5 PCAS5,7*
 U4-14 P1-5 U6-13
 6 PCAS3*
 U128-2 U128-13 P1-6 U3-11 U4-8
 7 PCAS0*



```

      P1-7 U4-3 U3-14
8      PUSELB
      U4-17 P1-8 U3-13
9      PCAS0,3*
      P1-9 U3-12 B1-1
10     NO CONNECTION

P2 4 1K 8 SIP8 J4
1      +5V
2      NO CONNECTION
3      IRQ4*
      U97-6 J15-30 P2-3 U148-5
4      IRQ3*
      U97-7 J14-30 P2-4 U148-4
5      NO CONNECTION
6      IRQ2*
      U101-15 J13-30 P2-6 U148-2
7      IRQ1*
      U101-14 J12-30 P2-7 U148-1
8      NO CONNECTION

P3 5 SP3 10 SIP10 L9
1      P3-1 Q12-2
2      SYNCH
      J20-1 P17-10 U126-5 P3-2 P4-6
3      RGB1
      U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
      U90-12
4      RGB2
      J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
      U89-10
5      RGB4
      P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7
      J20-4
6      RGB8
      J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
      U89-2
7      NTSCA
      P3-7 U90-7
8      NTSCB
      P3-8 U163-8
9      P3-9 R55-2 C5-1 L1-1
10     NO CONNECTION

P4 5 SP4 8 SIP8 M8
1      P4-1 Q4-2
2      RGB1
      U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
      U90-12
3      RGB2
      J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
      U89-10
    
```



4 RGB4
 P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7
 J20-4
 5 RGB8
 J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
 U89-2
 6 SYNCH
 J20-1 P17-10 U126-5 P3-2 P4-6
 7 NO CONNECTION
 8 +5V

P8 5 1K 6 SIP6 F4

1 +5V
 2 NO CONNECTION
 3 DX5
 U89-12 U86-5 P8-3
 4 DX4
 U89-13 U86-16 P8-4
 5 DX6
 U89-5 U86-19 P8-5
 6 DX7
 U89-4 U86-2 P8-6

P9 7 SP9 8 SIP8 C5

1 SUMSND
 R35-2 P9-1 C10-1 R78-2
 2 SND5
 P9-2 U97-15
 3 SND4
 P9-3 U97-14
 4 SND3
 P9-4 U97-13
 5 SND2
 P9-5 U97-12
 6 SND1
 P9-6 U97-11
 7 SNDO
 P9-7 U97-10
 8 +5V

P10 5 1K 8 SIP8 G4

1 +5V
 2 FORCPAGE
 U87-6 P10-2 J20-7
 3 RGB8
 J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
 U89-2
 4 RGB1
 U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
 U90-12
 5 NO CONNECTION
 6 RGB2



J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
 U89-10
 7 RGB4
 P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7
 J20-4
 8 NO CONNECTION

P11 8 1K 8 SIP8 L14
 1 +5V
 2 CONTROL*
 J7-11 U179-13 U107-28 U109-10 P11-2
 3 CAPLCK*
 J7-9 P11-3 U109-13
 4 APPLEI*
 J7-7 P11-4 U111-3
 5 APPLEII*
 J7-5 P11-5 U106-12 U164-5
 6 SHIFT*
 J7-24 P11-6 U107-29
 7 NO CONNECTION
 8 NO CONNECTION

P13 5 1K 6 SIP6 G3
 1 GND
 2 DX0
 U88-14 U86-12 P13-2
 3 DX2
 U88-5 U86-15 P13-3
 4 DX3
 U88-2 U86-6 P13-4
 5 DX1
 U88-11 U86-9 P13-5
 6 NO CONNECTION

P14 3-4-8 1K 6 SIP6 M1
 1 +5V
 2 DMAI*
 U176-2 J12-28 J15-28 J14-28 J13-28 U163-5 P14-2
 U153-9
 3 TSADB*
 J15-22 J12-22 J14-22 J13-22 U163-3 P14-3
 4 RDY
 J15-21 U164-8 J14-21 J13-21 J12-21 P14-4 U65-2
 5 INH*
 J15-32 J14-32 J13-32 J12-32 U176-4 U165-13 P14-5
 6 IORESET*
 J12-31 J13-31 J14-31 J15-31 P14-6 U164-2 U164-4
 U96-4 U94-15

P15 3 3.3K 10 SIP10 C3
 1 +5V
 2 DO



	J15-49	J14-49	J13-49	J12-49	U68-19	J16-17	U109-4
3	U91-16	U66-4	P15-2				
	D1						
	J15-48	J14-48	J13-48	J12-48	U68-18	J16-18	U109-7
4	U91-4	U66-7	P15-3				
	D2						
	J15-47	J14-47	J13-47	J12-47	U68-17	J16-19	U109-9
5	U91-15	U66-9	P15-4				
	D3						
	J15-46	J14-46	J13-46	J12-46	U68-16	J16-20	U109-12
6	U91-5	U66-12	P15-5				
	D4						
	J15-45	J14-45	J13-45	J12-45	U68-15	J16-21	U111-4
7	U91-14	U69-4	P15-6				
	D5						
	J15-44	J14-44	J13-44	J12-44	U68-14	J16-22	U111-7
8	U91-6	U69-7	P15-7				
	D6						
	J15-43	J14-43	J13-43	J12-43	U68-13	J16-23	U111-9
9	U91-13	U69-9	P15-8				
	D7						
	J15-42	J14-42	J13-42	J12-42	U68-12	J16-24	U111-12
10	U101-5	U91-7	U69-12	P15-9			
	NO CONNECTION						

P16 2 330 8 SIP8 F13

1	+5V			
2	AR2			
	J17-8	U5-7	P16-2	
3	AR3			
	J17-7	U5-9	P16-3	
4	ARO			
	P16-4	R97-1	U2-7	
5	AR6			
	J17-4	U13-9	P16-5	
6	AR4			
	J17-6	U9-7	P16-6	
7	AR5			
	R99-1	U9-9	P16-7	
8	AR1			
	P16-8	R98-1	U2-9	

P17 5 75 10 SIPI0 M9

1	XRGB8						
	J5-10	P17-1	P18-2				
2	RGB8						
	J20-5	P4-5	P17-2	P3-6	P10-3	U90-11	U90-3
	U89-2						
3	XRGB4						
	J5-2	P17-3	P18-3				
4	RGB4						
	P4-4	P17-4	P3-5	P10-7	U90-4	U90-10	U89-7



5 J20-4
 XRGB2
 J5-9 P17-5 P18-4
 6 RGB2
 J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
 U89-10
 7 XRGB1
 J5-5 P17-7 P18-5
 8 RGB1
 U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
 U90-12
 9 XSYNC
 J5-3 P17-9 P18-6
 10 SYNCH
 J20-1 P17-10 U126-5 P3-2 P4-6

P18 5 220P 8 SIP8 M9

1 GNDF
 2 XRGB8
 J5-10 P17-1 P18-2
 3 XRGB4
 J5-2 P17-3 P18-3
 4 XRGB2
 J5-9 P17-5 P18-4
 5 XRGB1
 J5-5 P17-7 P18-5
 6 XSYNC
 J5-3 P17-9 P18-6
 7 B&WVID
 J5-11 R15-2 J10-1 P18-7
 8 NTSC
 J5-12 R12-2 P18-8

P19 4 3.3K 10 SIP10 B6

1 +5V
 2 SEL1M
 U180-15 U174-2 U150-3 U73-9 P19-2
 3 IOEN
 U73-8 P19-3 U147-10
 4 SCRN
 U154-13 U73-7 P19-4
 5 RESETLK*
 U179-14 U73-6 P19-5 U139-5
 6 RWPR
 U73-5 P19-6 U180-16
 7 PRIMSTK
 U73-4 P19-7 U158-9
 8 ROMSEL2
 U73-3 U64-21 P19-8
 9 ROMSEL1
 U165-10 U73-2 P19-9
 10 PH2M



P19-10 U141-5 U65-39 U176-17

R4 4 100K 2 QW A2
 1 +5V
 2 Q2-1 C1-1 R4-2

R5 4 47K 2 QW C2
 1 +12V
 2 Q1-2 R5-2

R6 4 470K 2 QW A2
 1 +5V
 2 PWRDN*
 U72-23 Q3-3 C2-1 R6-2

R9 4 3.3M 2 QW A2
 1 R9-1 T1-1 X1-1 U105-10
 2 GND

R125 47 2 QW N6
 1 R12-1 R13-1 Q12-1
 2 NTSC
 J5-12 R12-2 P18-8

R13 5 75 2 QW N6
 1 R12-1 R13-1 Q12-1
 2 GND

R15 5 47 2 QW M6
 1 R15-1 Q4-1 R16-1
 2 B&WVID
 J5-11 R15-2 J10-1 P18-7

R16 5 75 2 QW N5
 1 R15-1 Q4-1 R16-1
 2 GND

R31 6 47K 2 QW N14
 1 DT1M
 R31-1 U164-10 C7-2 U96-2 U96-6
 2 +5V

R34 7 3K 2 QW N5
 1 EXTSPK1
 J11-1 R34-1
 2 C78-1 U103-2 R34-2 R36-1

R35 7 4.7K 2 QW G6
 1 AIISPKR
 R35-1 U173-9
 2 SUMSND
 R35-2 P9-1 C10-1 R78-2



R36 7 1K 2 QW M12
 1 C78-1 U103-2 R34-2 R36-1
 2 GND

R37 7 4.3K 2 QW L9
 1 +12V
 2 U105-8 R37-2 R38-1 C76-1

R38 7 1.1K 2 QW L9
 1 U105-8 R37-2 R38-1 C76-1
 2 GND

R39 7 1.2M 2 QW L9
 1 +12V
 2 U105-6 R39-2 C9-1

R40 8 1M 2 QW L11
 1 ANYKEY
 U107-4 R40-1 X2-2 U106-11 U109-3
 2 U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1
 X3-1 C75-1

R43 8 1K 2 QW G13
 1 Q9-3 R43-1 U111-10
 2 GND

R44 8 2.2M 2 QW A6
 1 +5V
 2 UPRST*
 C20-1 U113-6 U113-2 R44-2 X4-2 X6-1

R45 8 15K 2 QW A5
 1 +5V
 2 U113-13 R45-2 R46-1

R46 8 3.3M 2 QW A5
 1 U113-13 R45-2 R46-1
 2 U113-8 R46-2 U113-12 C21-1

R47 8 100K 2 QW H14
 1 U107-3 R47-1
 2 U107-2 C16-2 R47-2

R48 9 180 2 QW A13
 1 +12V
 2 Q11-1 R48-2 Q10-1

R49 9 47 2 QW A13
 1 Q10-2 R90-2 R91-1 C23-1 R49-1
 2 Y1-2 Q11-2 R49-2



R51 9 75 2 QW A13
 1
 2 Q11-3 R51-1 U146-10 U146-5 U146-2
 GND

R52 9 47 2 QW A12
 1 R52-1 Q10-3 Y1-1
 2 NO CONNECTION

R53 9 1K 2 QW C14
 1 +5V
 2 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1

R54 9 1K 2 QW C9
 1 HPEDIS
 U135-2 J21-1 R54-1
 2 GND

R55 5 1K 2 QW K8
 1 COLORBURST
 R55-1 U147-12
 2 P3-9 R55-2 C5-1 L1-1

R57 7 10K 2 QW L8
 1 +5V
 2 PDLOT*
 U101-13 U105-7 R57-2

R58 2 27 2 QW C14
 1 RRAS0.3*
 R58-1 U12-3
 2 RAS0.3*
 J17-12 R58-2 B1-2

R59 2 27 2 QW C14
 1 RRAS1.2*
 R59-1 U12-6
 2 RAS1.2*
 J17-13 R59-2

R60 2 27 2 QW E12
 1 RRAS4.5*
 R60-1 U12-8
 2 RAS4.5*
 J17-14 R60-2

R61 2 27 2 QW D12
 1 RRAS6.7*
 R61-1 U12-11



2 RAS6.7*
 J17-15 R61-2

 R63 9 3K 2 QW G11
 1 FIELDIN
 J19-3 U121-18 R63-1
 2 GND

 R65 6 1K 2 QW H9
 1 +5V
 2 MOTON*
 U179-1 U92-15 U178-15 R65-2 U164-12 U92-16

 R66 6 1K 2 QW L11
 1 U92-9 U91-9 R66-1
 2 GND

 R67 6 1K 2 QW L9
 1 +5V
 2 U161-11 U92-5 R67-2

 R68 9 3K 2 QW F7
 1 PCS6522
 R68-1 U180-14
 2 CS6522
 R68-2 U73-24 C59-1 U97-24

 R69 7 33 2 QW M11
 1 AUDIO
 J9-1 R69-1
 2 R69-2 U103-6

 R75 7 1MEG 2 QW A8
 1 +5V
 2 C69-2 R75-2 U181-1 U181-2

 R76 7 1.5K 2 QW A8
 1 +5V
 2 R76-2 R77-1 U181-13

 R77 7 6.8K 2 QW A8
 1 R76-2 R77-1 U181-13
 2 R77-2 U181-8 U181-12 C70-1

 R78 7 6.8K 2 QW A7
 1 U181-9 R78-1
 2 SUMSND
 R35-2 P9-1 C10-1 R78-2

 R79 4 3K 2 QW A4
 1 +5V
 2 CKIRQ*



```

                U72-13 R79-2 U97-40

R80 8 10K 2 QW M14
  1      KRESET*
        U139-12 R80-1 C71-1 U179-15 J7-15
  2      +5V

R81 1 240 2 QW N4
  1      X7-2 R81-1
  2      GND

R85 7 3K 2 QW N4
  1      +5V
  2      SW1/MGNSWUF
        R85-2 U169-19 U73-39 U101-3

R86 7 1.1K 2 QW J9
  1      R86-1 U160-6 U161-2
  2      GND

R87 7 15K 2 QW J1
  1      +12V
  2      PCTS
        R87-2 U172-15 U100-4

R88 7 15K 2 QW J1
  1      +12V
  2      PDSR
        R88-2 U172-16 U100-10

R89 7 15K 2 QW J2
  1      +12V
  2      PDCD
        R89-2 U172-18 U100-13

R90 9 1.5K 2 QW A13
  1      +12V
  2      Q10-2 R90-2 R91-1 C23-1 R49-1

R91 9 1.1K 2 QW A12
  1      Q10-2 R90-2 R91-1 C23-1 R49-1
  2      GND

R92 8 1K 2 QW A4
  1      TRESET
        U113-5 R92-1
  2      UUTUPRST
        R92-2 U162-11 J21-4 U164-3

R93 1 120 2 QW L1
  1      Q3
        J14-37 U85-3 J15-37 J13-37 J12-37 U117-12 R93-1

```



```

                U154-2
    2           R93-2 C67-1

R94 1 120 2 QW L1
    1           PRE1M
                J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5
                U97-25 U139-10 R94-1
    2           R94-2 C68-1

R95 1 120 2 QW L1
    1           PRE1M*
                J12-38 J13-38 J14-38 J15-38 U123-6 R95-1
    2           R95-2 C77-1

R96 1 120 2 QW M5
    1           PHO
                J15-19 U136-11 J14-19 J13-19 J12-19 U124-5 U65-37
                R96-1
    2           R96-2 C66-1

R97 2 33 2 QW E13
    1           ARO
                P16-4 R97-1 U2-7
    2           ARO
                R97-2 J17-10

R98 2 33 2 QW E13
    1           AR1
                P16-8 R98-1 U2-9
    2           AR1
                R98-2 J17-9

R99 2 33 2 QW F12
    1           AR5
                R99-1 U9-9 P16-7
    2           AR5
                R99-2 J17-5

R100 2 100 2 QW D12
    1           C14M
                U146-6 R100-1 U117-10 U119-9 U79-7 U85-11
    2           R100-2 C74-1 U4-11

R101 4 100K 2 QW A2
    1           Q3-2 Q2-3 R101-1
    2           GND

R102 6 470 2 QW L9
    1           +5V
    2           Q3*
                U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
                U131-5 R102-2 R103-1 U146-1
    
```



```

R103 6 300 2 QW L9
  1      Q3*
      U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
      U131-5 R102-2 R103-1 U146-1
  2      GND

R106 1 1K 2 QW J8
  1      +5V
  2      S5A
      R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
      U173-10

R107 1 1K 2 QW H12
  1      +5V
  2      S5B
      R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
      U114-7 U116-5

R108 1 1K 2 QW D5
  1      +5V
  2      S5C
      R108-2 U140-1 U140-4 U140-10 U140-13

R109 1 1K 2 QW A10
  1      +5V
  2      S5D
      R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7

T1 4 4V 2 BAT G14
  1      R9-1 T1-1 X1-1 U105-10
  2      GND

U1 2 LS283 16 E11
  1      SUM2
      U1-1 U9-5
  2      V4
      U87-19 U1-2 U154-4 U120-12 U1-11
  3      H4
      U121-5 U1-3 U116-14
  4      SUM1
      U1-4 U5-11
  5      H3
      U114-11 U1-5 U121-6
  6      V3
      U120-13 U1-15 U154-5 U1-6
  7      GND
  8      GND
  9      NO CONNECTION
 10     SUM4
      U1-10 U2-3
    
```



```

11      V4
        U87-19 U1-2 U154-4 U120-12 U1-11
12      GND
13      SUM3
        U1-13 U9-11
14      H5
        U121-4 U1-14 U116-13
15      V3
        U120-13 U1-15 U154-5 U1-6
16      +5V

U2 2 S153 16 F13
1      GND
2      AX*
        U124-8 U2-2 U13-2 U9-2 U5-2 J17-23
3      SUM4
        U1-10 U2-3
4      A6
        J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
        U71-10 U63-5 U71-11
5      H0
        U114-14 U2-5
6      A0
        J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
        U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
7      ARO
        P16-4 R97-1 U2-7
8      GND
9      AR1
        P16-8 R98-1 U2-9
10     A1
        J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
        U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
11     H1
        U114-13 U2-11 U155-9
12     A8
        J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
        U2-12 U13-6 U13-4
13     V1
        U121-22 U2-13 U118-11 U13-5 U13-3
14     AY*
        U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
        U3-1
15     GND
16     +5V

U3 2 7643 18 341-0056 C12
1      AY*
        U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
        U3-1
2      RDHIRES
        U175-1 U11-17 U3-2 U6-3 U126-9
    
```



3 R/W*
 J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
 U176-6 U165-11 U160-11

4 A15
 J15-17 J14-17 J13-17 J12-17 U3-4 U147-3 U70-12
 U174-16 U6-7

5 A11
 J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
 U150-13 U74-5 U67-12 U3-5 U174-5

6 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
 U71-1 U70-7 U6-5 U3-6 U174-7

7 A14
 J15-16 J14-16 J13-16 J12-16 U3-7 U147-4 U70-9
 U6-16 U174-4

8 Q1
 U6-8 U131-4 U3-8 U124-12 U117-14

9 GND

10 GND

11 PCAS3*
 U128-2 U128-13 P1-6 U3-11 U4-8

12 PCAS0,3*
 P1-9 U3-12 B1-1

13 PUSELB
 U4-17 P1-8 U3-13

14 PCAS0*
 P1-7 U4-3 U3-14

15 PRAS0.3
 U128-5 U11-14 U12-1 U3-15 U6-4

16 PRAS1.2
 U12-4 U11-13 U3-16

17 ABK2
 U6-1 U11-6 U10-7 U3-17

18 +5V

U4 2 S374 20 D13

1 GND

2 CAS0*
 J17-17 U4-2

3 PCAS0*
 P1-7 U4-3 U3-14

4 PCAS1*
 P1-3 U4-4 U6-12 U128-10

5 CAS1*
 J17-18 U4-5

6 CAS2*
 J17-19 U4-6

7 PCAS2*
 P1-4 U4-7 U6-11

8 PCAS3*
 U128-2 U128-13 P1-6 U3-11 U4-8

9 CAS3*



10 J17-20 U4-9
 GND
 11 R100-2 C74-1 U4-11
 CAS4,6*
 12 J17-21 U4-12
 PCAS4,6*
 13 U4-13 U6-14 P1-2
 PCAS5,7*
 14 U4-14 P1-5 U6-13
 CAS5,7*
 15 U4-15 J17-22
 USELB
 16 U4-16 U66-1 U69-1
 PUSELB
 17 U4-17 P1-8 U3-13
 18 NO CONNECTION
 19 NO CONNECTION
 20 +5V

US 2 S153 16 E12

1 GND
 2 AX*
 U124-8 U2-2 U13-2 U9-2 U5-2 J17-23
 3 V2*V5
 U121-8 U154-8 U5-3
 4 A9
 J15-11 J14-11 J13-11 J12-11 U5-4 U74-2 U71-5
 U67-7
 5 H2
 U121-7 U5-5 U114-12 U85-8
 6 A2
 J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
 U177-2 U94-2 U75-2 U73-36 U63-14
 7 AR2
 J17-8 U5-7 P16-2
 8 GND
 9 AR3
 J17-7 U5-9 P16-3
 10 A3
 J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
 U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
 11 SUM1
 U1-4 U5-11
 12 U5-12 U128-3
 13 MUX1
 U101-12 U175-14 U5-13
 14 AY*
 U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
 U3-1
 15 GND
 16 +5V



U6 2 7643 18 341-0042 C13

1	ABK2
	U6-1 U11-6 U10-7 U3-17
2	ABK1
	U6-2 U11-5 U10-2
3	RDHIRES
	U175-1 U11-17 U3-2 U6-3 U126-9
4	PRAS0.3
	U128-5 U11-14 U12-1 U3-15 U6-4
5	A13
	J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
	U71-1 U70-7 U6-5 U3-6 U174-7
6	A11
	J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
	U150-13 U74-5 U67-12 U3-5 U174-5
7	A15
	J15-17 J14-17 J13-17 J12-17 U3-4 U147-3 U70-12
	U174-16 U6-7
8	Q1
	U6-8 U131-4 U3-8 U124-12 U117-14
9	GND
10	GND
11	PCAS2*
	P1-4 U4-7 U6-11
12	PCAS1*
	P1-3 U4-4 U6-12 U128-10
13	PCAS5,7*
	U4-14 P1-5 U6-13
14	PCAS4,6*
	U4-13 U6-14 P1-2
15	AY*
	U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
	U3-1
16	A14
	J15-16 J14-16 J13-16 J12-16 U3-7 U147-4 U70-9
	U6-16 U174-4
17	ABK3
	U6-17 U11-7 U10-10
18	+5V

U9 2 S153 16 F12

1	GND
2	AX*
	U124-8 U2-2 U13-2 U9-2 U5-2 J17-23
3	MUX2
	U9-3 U175-13
4	U9-4 U128-11
5	SUM2
	U1-1 U9-5
6	A4
	J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
	U112-2 U63-9



7 AR4
 J17-6 U9-7 P16-6
 8 GND
 9 AR5
 R99-1 U9-9 P16-7
 10 A5
 J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
 U112-3 U63-7
 11 SUM3
 U1-13 U9-11
 12 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6
 13 MUX3
 U9-13 U175-12
 14 AY*
 U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
 U3-1
 15 GND
 16 +5V

U10 2 LS399 16 A9

1 U10-1 U153-6
 2 ABK1
 U6-2 U11-5 U10-2
 3 BKSW1
 U97-2 U10-3
 4 DAO
 J16-9 U10-4 U84-13 U66-2
 5 DA1
 J16-10 U10-5 U84-8 U66-5
 6 BKSW2
 U97-3 U10-6
 7 ABK2
 U6-1 U11-6 U10-7 U3-17
 8 GND
 9 CLKBK
 U136-8 U10-9
 10 ABK3
 U6-17 U11-7 U10-10
 11 BKSW3
 U97-4 U10-11
 12 DA2
 J16-11 U10-12 U84-14 U66-11
 13 S5D
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
 U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
 14 GND
 15 ABK4
 U11-15 U174-1 U10-15 U158-10
 16 +5V



```

U11 2 7643 18 341-0044 C11
 1      ZPAGE*
      U11-1 U174-15 U67-1 U131-8 U70-1
 2      PAB
      U11-2 U174-17 U67-3 U158-6
 3      AY*
      U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
      U3-1
 4      PA15
      U135-10 U11-4 U70-13 U65-25
 5      ABK1
      U6-2 U11-5 U10-2
 6      ABK2
      U6-1 U11-6 U10-7 U3-17
 7      ABK3
      U6-17 U11-7 U10-10
 8      GND
 9      GND
10     GND
11     PRAS6.7
      U12-12 U11-11
12     PRAS4.5
      U12-9 U11-12
13     PRAS1.2
      U12-4 U11-13 U3-16
14     PRAS0.3
      U128-5 U11-14 U12-1 U3-15 U6-4
15     ABK4
      U11-15 U174-1 U10-15 U158-10
16     RFSH
      U158-13 U11-16 U126-2
17     RDHIRES
      U175-1 U11-17 U3-2 U6-3 U126-9
18     +5V

U12 2 S00 14 D12
 1      PRAS0.3
      U128-5 U11-14 U12-1 U3-15 U6-4
 2      RAS
      U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
 3      RRAS0.3*
      R58-1 U12-3
 4      PRAS1.2
      U12-4 U11-13 U3-16
 5      RAS
      U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
 6      RRAS1.2*
      R59-1 U12-6
 7      GND
 8      RRAS4.5*
      R60-1 U12-8
 9      PRAS4.5
    
```



10 U12-9 U11-12
 RAS
 U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
 11 RRAS6.7*
 R61-1 U12-11
 12 PRAS6.7
 U12-12 U11-11
 13 RAS
 U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
 14 +5V

U13 2 S153 16 E13

1 GND
 2 AX*
 U124-8 U2-2 U13-2 U9-3 U5-2 J17-23
 3 V1
 U121-22 U2-13 U118-11 J13-5 U13-3
 4 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4
 5 V1
 U121-22 U2-13 U118-11 U13-5 U13-3
 6 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4
 7 AR7
 U13-7 J17-24
 8 GND
 9 AR6
 J17-4 U13-9 P16-5
 10 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3
 11 V0
 U121-23 U13-11 U118-12
 12 U13-12 U128-8
 13 PG2*
 U87-9 U13-13
 14 AY*
 U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
 U3-1
 15 GND
 16 +5V

U63 3 LS244 20 D6 - 1425279

1 U70-15 U163-4 U67-15 U63-19 U63-1
 2 PA0
 U64-8 U63-2 U65-9
 3 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3
 4 PA1



	U64-7	U63-4	U65-10				
5	A6						
	J15-8	J14-8	J13-8	J12-8	U2-4	U77-3	U76-3
	U71-10	U63-5	U71-11				
6	PA2						
	U64-6	U63-6	U65-11				
7	A5						
	J15-7	J14-7	J13-7	J12-7	U9-10	U76-2	U77-2
	U112-3	U63-7					
8	PA3						
	U64-5	U63-8	U65-12				
9	A4						
	J15-6	J14-6	J13-6	J12-6	U9-6	U77-1	U76-1
	U112-2	U63-9					
10	GND						
11	PA4						
	U64-4	U63-11	U65-13				
12	A3						
	J15-5	J14-5	J13-5	J12-5	U5-10	U109-1	U111-1
	U97-35	U177-3	U94-3	U75-3	U73-35	U63-12	
13	PA5						
	U64-3	U63-13	U65-14				
14	A2						
	J15-4	J14-4	J13-4	J12-4	U5-6	U101-9	U97-36
	U177-2	U94-2	U75-2	U73-36	U63-14		
15	PA6						
	U64-2	U63-15	U65-15				
16	A1						
	J15-3	J14-3	J13-3	J12-3	U2-10	U97-37	U98-14
	U101-10	U177-1	U94-1	U73-37	U75-1	U63-16	
17	PA7						
	U64-1	U63-17	U65-16				
18	A0						
	J15-2	J14-2	J13-2	J12-2	U2-6	U97-38	U98-13
	U101-11	U94-13	U73-38	U75-13	U63-18	U177-13	
19	U70-15	U163-4	U67-15	U63-19	U63-1		
20	+5V						

U64 3 2332 24 B9

1	PA7						
	U64-1	U63-17	U65-16				
2	PA6						
	U64-2	U63-15	U65-15				
3	PA5						
	U64-3	U63-13	U65-14				
4	PA4						
	U64-4	U63-11	U65-13				
5	PA3						
	U64-5	U63-8	U65-12				
6	PA2						
	U64-6	U63-6	U65-11				
7	PA1						



8	U64-7 U63-4 U65-10
	PA0
	U64-8 U63-2 U65-9
9	ID0
	U98-18 U65-33 U97-33 U72-15 U73-33 U64-9 U68-1
10	ID1
	U98-19 U65-32 U97-32 U72-16 U73-32 U64-10 U68-2
11	ID2
	U98-20 U65-31 U97-31 U72-17 U73-31 U64-11 U68-3
12	NO CONNECTION
13	ID3
	U98-21 U65-30 U97-30 U72-18 U73-30 U64-13 U68-4
14	ID4
	U98-22 U65-29 U97-29 U72-19 U73-29 U64-14 U68-5
15	ID5
	U98-23 U65-28 U97-28 U72-20 U73-28 U64-15 U68-6
16	ID6
	U98-24 U65-27 U97-27 U72-21 U73-27 U64-16 U68-7
17	ID7
	U98-25 U65-26 U97-26 U72-22 U73-26 U64-17 U68-8
18	PA11
	U64-18 U144-12 U65-20 U67-13 U135-4
19	PA10
	U64-19 U155-6 U65-19 U67-10
20	TROMSEL*
	U153-8 U162-13 U64-20
21	ROMSEL2
	U73-3 U64-21 P19-8
22	PA9
	U64-22 U155-5 U65-18 U67-6
23	PPA8
	U158-4 U152-4 U158-5 U65-17 U64-23 U132-5
24	+5V

U65 3 6502B 40 B8

1	GND
2	RDY
	J15-21 U164-8 J14-21 J13-21 J12-21 P14-4 U65-2
3	NO CONNECTION
4	IRQ*
	U97-21 U65-4 U98-26 U97-9 U73-21
5	NO CONNECTION
6	NMI*
	U139-6 U65-6 U155-2
7	SYNC
	U65-7 U136-1 J15-35 U174-3 J14-35 J13-35 J12-35
8	+5V
9	PA0
	U64-8 U63-2 U65-9
10	PA1
	U64-7 U63-4 U65-10
11	PA2



12	U64-6 U63-6 U65-11 PA3 U64-5 U63-8 U65-12
13	PA4 U64-4 U63-11 U65-13
14	PA5 U64-3 U63-13 U65-14
15	PA6 U64-2 U63-15 U65-15
16	PA7 U64-1 U63-17 U65-16
17	PPA8 U158-4 U152-4 U158-5 U65-17 U64-23 U132-5
18	PA9 U64-22 U155-5 U65-18 U67-6
19	PA10 U64-19 U155-6 U65-19 U67-10
20	PA11 U64-18 U144-12 U65-20 U67-13 U135-4
21	GND
22	PA12 U65-22 U135-9 U70-3
23	PA13 U65-23 U135-8 U70-6
24	PA14 U65-24 U135-11 U70-10
25	PA15 U135-10 U11-4 U70-13 U65-25
26	ID7 U98-25 U65-26 U97-26 U72-22 U73-26 U64-17 U68-8
27	ID6 U98-24 U65-27 U97-27 U72-21 U73-27 U64-16 U68-7
28	ID5 U98-23 U65-28 U97-28 U72-20 U73-28 U64-15 U68-6
29	ID4 U98-22 U65-29 U97-29 U72-19 U73-29 U64-14 U68-5
30	ID3 U98-21 U65-30 U97-30 U72-18 U73-30 U64-13 U68-4
31	ID2 U98-20 U65-31 U97-31 U72-17 U73-31 U64-11 U68-3
32	ID1 U98-19 U65-32 U97-32 U72-16 U73-32 U64-10 U68-2
33	ID0 U98-18 U65-33 U97-33 U72-15 U73-33 U64-9 U68-1
34	IR/W* U98-28 U97-22 U112-14 U73-22 U163-11 U65-34 U160-12
35	NO CONNECTION
36	NO CONNECTION
37	PHO J15-19 U136-11 J14-19 J13-19 J12-19 U124-5 U65-37
38	R96-1 GND



39 PH2M
 P19-10 U141-5 U65-39 U176-17
 40 RESET*
 U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
 U173-1

U66 3 S257 16 D3

1 USELB
 U4-16 U66-1 U69-1
 2 DAO
 J16-9 U10-4 U84-13 U66-2
 3 DB0
 J16-1 U66-3 U80-13
 4 DO
 J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4
 U91-16 U66-4 P15-2
 5 DA1
 J16-10 U10-5 U84-8 U66-5
 6 DB1
 J16-2 U66-6 U80-8
 7 DI
 J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7
 U91-4 U66-7 P15-3
 8 GND
 9 D2
 J15-47 J14-47 J13-47 J12-47 U68-17 J16-19 U109-9
 U91-15 U66-9 P15-4
 10 DB2
 J16-3 U66-10 U80-14
 11 DA2
 J16-11 U10-12 U84-14 U66-11
 12 D3
 J15-46 J14-46 J13-46 J12-46 U68-16 J16-20 U109-12
 U91-5 U66-12 P15-5
 13 DB3
 J16-4 U66-13 U80-7
 14 DA3
 J16-12 U66-14 U84-7
 15 EN257
 U176-14 U66-15 U69-15
 16 +5V

U67 3 S257 16 D8

1 ZPAGE*
 U11-1 U174-15 U67-1 U131-8 U70-1
 2 U67-2 U132-6
 3 PA8
 U11-2 U174-17 U67-3 U158-6
 4 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4
 5 Z1



	U72-6	U67-5	U73-11				
6	PA9						
	U64-22	U155-5	U65-18	U67-6			
7	A9						
	J15-11	J14-11	J13-11	J12-11	U5-4	U74-2	U71-5
	U67-7						
8	GND						
9	A10						
	J15-12	J14-12	J13-12	J12-12	U128-1	U74-3	U71-4
	U67-9						
10	PA10						
	U64-19	U155-6	U65-19	U67-10			
11	Z2						
	U72-7	U67-11	U73-12				
12	A11						
	J15-13	J14-13	J13-13	J12-13	U128-12	U6-6	U71-3
	U150-13	U74-5	U67-12	U3-5	U174-5		
13	PA11						
	U64-18	U144-12	U65-20	U67-13	U135-4		
14	Z3						
	U72-8	U67-14	U73-13				
15	U70-15	U163-4	U67-15	U63-19	U63-1		
16	+5V						

U68 3 8304 20 C3

1	ID0						
	U98-18	U65-33	U97-33	U72-15	U73-33	U64-9	U68-1
2	ID1						
	U98-19	U65-32	U97-32	U72-16	U73-32	U64-10	U68-2
3	ID2						
	U98-20	U65-31	U97-31	U72-17	U73-31	U64-11	U68-3
4	ID3						
	U98-21	U65-30	U97-30	U72-18	U73-30	U64-13	U68-4
5	ID4						
	U98-22	U65-29	U97-29	U72-19	U73-29	U64-14	U68-5
6	ID5						
	U98-23	U65-28	U97-28	U72-20	U73-28	U64-15	U68-6
7	ID6						
	U98-24	U65-27	U97-27	U72-21	U73-27	U64-16	U68-7
8	ID7						
	U98-25	U65-26	U97-26	U72-22	U73-26	U64-17	U68-8
9	EN8304						
	U176-13	U68-9					
10	GND						
11	IR*/W						
	U91-3	U68-11	U158-2	U163-10			
12	D7						
	J15-42	J14-42	J13-42	J12-42	U68-12	J16-24	U111-12
	U101-5	U91-7	U69-12	P15-9			
13	D6						
	J15-43	J14-43	J13-43	J12-43	U68-13	J16-23	U111-9
	U91-13	U69-9	P15-8				



14	D5	J15-44	J14-44	J13-44	J12-44	U68-14	J16-22	U111-7
		U91-6	U69-7	P15-7				
15	D4	J15-45	J14-45	J13-45	J12-45	U68-15	J16-21	U111-4
		U91-14	U69-4	P15-6				
16	D3	J15-46	J14-46	J13-46	J12-46	U68-16	J16-20	U109-12
		U91-5	U66-12	P15-5				
17	D2	J15-47	J14-47	J13-47	J12-47	U68-17	J16-19	U109-9
		U91-15	U66-9	P15-4				
18	D1	J15-48	J14-48	J13-48	J12-48	U68-18	J16-18	U109-7
		U91-4	U66-7	P15-3				
19	D0	J15-49	J14-49	J13-49	J12-49	U68-19	J16-17	U109-4
		U91-16	U66-4	P15-2				
20	+5V							

U69 3 S257 16 D2

1	USELB	U4-16	U66-1	U69-1				
2	DA4	J16-13	U69-2	U84-17				
3	DB4	J16-5	U69-3	U80-17				
4	D4	J15-45	J14-45	J13-45	J12-45	U68-15	J16-21	U111-4
		U91-14	U69-4	P15-6				
5	DA5	J16-14	U69-5	U84-4				
6	DB5	J16-6	U69-6	U80-4				
7	D5	J15-44	J14-44	J13-44	J12-44	U68-14	J16-22	U111-7
		U91-6	U69-7	P15-7				
8	GND							
9	D6	J15-43	J14-43	J13-43	J12-43	U68-13	J16-23	U111-9
		U91-13	U69-9	P15-8				
10	DB6	J16-7	U69-10	U80-18				
11	DA6	J16-15	U69-11	U84-18				
12	D7	J15-42	J14-42	J13-42	J12-42	U68-12	J16-24	U111-12
		U101-5	U91-7	U69-12	P15-9			
13	DB7	J16-8	U69-13	U80-3				
14	DA7	J16-16	U153-5	U84-3	U69-14			



15 EN257
 U176-14 U66-15 U69-15
 16 +5V

U70 3 S257 16 D7

1 ZPAGE*
 U11-1 U174-15 U67-1 U131-8 U70-1
 2 Z4
 U72-9 U70-2 U73-14
 3 PA12
 U65-22 U135-9 U70-3
 4 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6
 5 Z5
 U73-15 U70-5
 6 PA13
 U65-23 U135-8 U70-6
 7 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
 U71-1 U70-7 U6-5 U3-6 U174-7
 8 GND
 9 A14
 J15-16 J14-16 J13-16 J12-16 U3-7 U147-4 U70-9
 U6-16 U174-4
 10 PA14
 U65-24 U135-11 U70-10
 11 Z6
 U73-16 U70-11
 12 A15
 J15-17 J14-17 J13-17 J12-17 U3-4 U147-3 U70-12
 U174-16 U6-7
 13 PA15
 U135-10 U11-4 U70-13 U65-25
 14 Z7
 U73-17 U70-14
 15 U70-15 U163-4 U67-15 U63-19 U63-1
 16 +5V

U71 4 LS133 16 G7

1 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
 U71-1 U70-7 U6-5 U3-6 U174-7
 2 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6
 3 A11
 J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
 U150-13 U74-5 U67-12 U3-5 U174-5
 4 A10
 J15-12 J14-12 J13-12 J12-12 U128-1 U74-3 U71-4
 U67-9



5 A9
 J15-11 J14-11 J13-11 J12-11 U5-4 U74-2 U71-5
 U67-7

6 A8
 J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
 U2-12 U13-6 U13-4

7 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3

8 GND

9 U112-1 U71-9

10 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11

11 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11

12 AIISW*
 U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
 U155-3

13 C-FXXX
 U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1

14 C-FXXX
 U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1

15 C-FXXX
 U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1

16 +5V

U72 4 58167 24 B3

1 C07X*
 U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13

2 CLKRD
 U72-2 U158-3

3 U72-3 U112-12

4 NO CONNECTION

5 Z0
 U72-5 U132-4 U73-10

6 Z1
 U72-6 U67-5 U73-11

7 Z2
 U72-7 U67-11 U73-12

8 Z3
 U72-8 U67-14 U73-13

9 Z4
 U72-9 U70-2 U73-14

10 Y2-1 U72-10 C3-3

11 C4-1 U72-11 Y2-2

12 GND

13 CKIRQ*
 U72-13 R79-2 U97-40

14 PDINT*
 J5-4 U72-14



15	ID0	U98-18	U65-33	U97-33	U72-15	U73-33	U64-9	U68-1
16	ID1	U98-19	U65-32	U97-32	U72-16	U73-32	U64-10	U68-2
17	ID2	U98-20	U65-31	U97-31	U72-17	U73-31	U64-11	U68-3
18	ID3	U98-21	U65-30	U97-30	U72-18	U73-30	U64-13	U68-4
19	ID4	U98-22	U65-29	U97-29	U72-19	U73-29	U64-14	U68-5
20	ID5	U98-23	U65-28	U97-28	U72-20	U73-28	U64-15	U68-6
21	ID6	U98-24	U65-27	U97-27	U72-21	U73-27	U64-16	U68-7
22	ID7	U98-25	U65-26	U97-26	U72-22	U73-26	U64-17	U68-8
23	PWRDN*	U72-23	Q3-3	C2-1	R6-2			
24		U72-24	Q1-1	X1-2				

U73 4 6522 40 B6

1	GND							
2	ROMSEL1	U165-10	U73-2	P19-9				
3	ROMSEL2	U73-3	U64-21	P19-8				
4	PRIMSTK	U73-4	P19-7	U158-9				
5	RWPR	U73-5	P19-6	U180-16				
6	RESETLK*	U179-14	U73-6	P19-5	U139-5			
7	SCRN	U154-13	U73-7	P19-4				
8	IOEN	U73-8	P19-3	U147-10				
9	SEL1M	U180-15	U174-2	U150-3	U73-9	P19-2		
10	Z0	U72-5	U132-4	U73-10				
11	Z1	U72-6	U67-5	U73-11				
12	Z2	U72-7	U67-11	U73-12				
13	Z3	U72-8	U67-14	U73-13				
14	Z4	U72-9	U70-2	U73-14				
15	Z5	U73-15	U70-5					
16	Z6	U73-16	U70-11					



17	Z7								
	U73-17	U70-14							
18	SCO								
	U160-3	U73-18	U160-5	U101-1					
19	SER								
	U73-19	U160-8	U161-9						
20	+5V								
21	IRQ*								
	U97-21	U65-4	U98-26	U97-9	U73-21				
22	IR/W*								
	U98-28	U97-22	U112-14	U73-22	U163-11	U65-34	U160-12		
23	FFDX*								
	U112-5	U73-23	U148-9						
24	CS6522								
	R68-2	U73-24	C59-1	U97-24					
25	PRE1M								
	J12-40	J13-40	J14-40	J15-40	U119-12	U73-25	U123-5		
	U97-25	U139-10	R94-1						
26	ID7								
	U98-25	U65-26	U97-26	U72-22	U73-26	U64-17	U68-8		
27	ID6								
	U98-24	U65-27	U97-27	U72-21	U73-27	U64-16	U68-7		
28	ID5								
	U98-23	U65-28	U97-28	U72-20	U73-28	U64-15	U68-6		
29	ID4								
	U98-22	U65-29	U97-29	U72-19	U73-29	U64-14	U68-5		
30	ID3								
	U98-21	U65-30	U97-30	U72-18	U73-30	U64-13	U68-4		
31	ID2								
	U98-20	U65-31	U97-31	U72-17	U73-31	U64-11	U68-3		
32	ID1								
	U98-19	U65-32	U97-32	U72-16	U73-32	U64-10	U68-2		
33	ID0								
	U98-18	U65-33	U97-33	U72-15	U73-33	U64-9	U68-1		
34	RESET*								
	U65-40	U162-10	U98-4	U97-34	U73-34	U75-15	U79-9		
	U173-1								
35	A3								
	J15-5	J14-5	J13-5	J12-5	U5-10	U109-1	U111-1		
	U97-35	U177-3	U94-3	U75-3	U73-35	U63-12			
36	A2								
	J15-4	J14-4	J13-4	J12-4	U5-6	U101-9	U97-36		
	U177-2	U94-2	U75-2	U73-36	U63-14				
37	A1								
	J15-3	J14-3	J13-3	J12-3	U2-10	U97-37	U98-14		
	U101-10	U177-1	U94-1	U73-37	U75-1	U63-16			
38	A0								
	J15-2	J14-2	J13-2	J12-2	U2-6	U97-38	U98-13		
	U101-11	U94-13	U73-38	U75-13	U63-18	U177-13			
39	SW1/MCNSWUF								
	R85-2	U169-19	U73-39	U101-3					
40	IOIRQ								



U73-40 U155-10

U74 4 LS138 16 J6

- 1 AB
J15-10 J14-10 J13-10 J12-10 U67-4 U74-1 U71-6
U2-12 U13-6 U13-4
- 2 A9
J15-11 J14-11 J13-11 J12-11 U5-4 U74-2 U71-5
U67-7
- 3 A10
J15-12 J14-12 J13-12 J12-12 U128-1 U74-3 U71-4
U67-9
- 4 GPH1
U162-3 U76-4 U141-6 U74-4
- 5 A11
J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
U150-13 U74-5 U67-12 U3-5 U174-5
- 6 CXXX
U176-15 U74-6 U150-1 U147-8
- 7 C7XX*
U176-16 U74-7
- 8 GND
- 9 C6XX*
U176-7 U74-9
- 10 U74-10 U176-5
- 11 IOSEL4*
U74-11 J15-1
- 12 IOSEL3*
U74-12 J14-1
- 13 IOSEL2*
U74-13 J13-1
- 14 IOSEL1*
U74-14 J12-1
- 15 COXX*
U77-5 U74-15 U76-5
- 16 +5V

U75 4 9334 16 H6

- 1 A1
J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
- 2 A2
J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
U177-2 U94-2 U75-2 U73-36 U63-14
- 3 A3
J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
- 4 TEXT
U87-23 U75-4
- 5 MIX
U87-1 U75-5
- 6 PAGE2



7 U87-4 U75-6
 HIRES
 U87-2 U75-7
 8 GND
 9 PDLO
 U105-16 U75-9
 10 PDL2
 U160-4 U105-2 U75-10
 11 PDLEN
 U105-3 U75-11
 12 AXCO
 U161-5 U75-12 U105-1
 13 AO
 J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
 14 CO5X*
 U77-10 U75-14
 15 RESET*
 U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
 U173-1
 16 +5V

U76 4 LS138 16 K4

1 A4
 J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
 U112-2 U63-9
 2 A5
 J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
 U112-3 U63-7
 3 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11
 4 GPH1
 U162-3 U76-4 U141-6 U74-4
 5 COXX*
 U77-5 U74-15 U76-5
 6 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3
 7 SEL6551*
 U98-3 U76-7 U148-12
 8 GND
 9 DEVSEL6*
 U94-14 U76-9 U91-2
 10 DEVSEL5*
 U177-14 U76-10
 11 DEVSEL4*
 J15-41 U76-11
 12 DEVSEL3*
 J14-41 U76-12
 13 DEVSEL2*
 J13-41 U76-13



14 DEVSEL1*
 J12-41 U76-14
 15 NO CONNECTION
 16 +5V

U77 4 LS138 16 K7

1 A4
 J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
 U112-2 U63-9
 2 A5
 J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
 U112-3 U63-7
 3 A6
 J15-8 J14-8 J13-8 J12-8 U2-4 U77-3 U76-3
 U71-10 U63-5 U71-11
 4 A7
 J15-9 J14-9 J13-9 J12-9 U13-10 U77-4 U76-6
 U71-7 U63-3
 5 COXX*
 U77-5 U74-15 U76-5
 6 GPH2
 U150-2 U77-6 U162-4
 7 C07X*
 U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13
 8 GND
 9 C06X*
 U101-7 U77-9
 10 C05X*
 U77-10 U75-14
 11 C04X*
 U77-11 U181-6
 12 SPKR*
 U173-11 U77-12
 13 C02X*
 J15-39 U77-13 J14-39 J13-39 J12-39
 14 CLRSTRB*
 U106-1 U77-14
 15 KBD*
 U109-15 U111-15 U77-15
 16 +5V

U78 5 LS374 20 E3

1 ENHREG*
 U175-11 U78-1
 2 DC7
 U79-2 U78-2 U82-14 U136-4
 3 DV7
 U86-3 U78-3 U80-2 U84-2 U136-3 U136-2
 4 DV5
 U82-3 U86-4 U78-4 U81-3 U80-5 U84-5
 5 DC5
 U79-4 U78-5 U82-13



6	DC3
	U79-10 U78-6 U82-12
7	DV3
	U82-2 U86-7 U78-7 U81-2 U80-6 U84-6
8	DV1
	U82-1 U86-8 U78-8 U81-1 U80-9 U84-9
9	DC1
	U79-12 U78-9 U82-11
10	GND
11	Q0
	U119-13 U150-10 U117-15 U131-2 U78-11
12	DC0
	U79-14 U78-12 U81-14
13	DV0
	U82-17 U86-13 U78-13 U81-17 U80-12 U84-12
14	DV2
	U82-16 U86-14 U78-14 U81-16 U80-15 U84-15
15	DC2
	U79-11 U78-15 U81-13
16	DC4
	U79-5 U78-16 U81-12
17	DV4
	U82-15 U86-17 U78-17 U81-15 U80-16 U84-16
18	DV6
	U78-18 U86-18 U82-4 U81-4 U80-19 U84-19
19	DC6
	U78-19 U79-3 U81-11
20	+5V

U79 5 74166 16 F4

1	GND
2	DC7
	U79-2 U78-2 U82-14 U136-4
3	DC6
	U78-19 U79-3 U81-11
4	DC5
	U79-4 U78-5 U82-13
5	DC4
	U79-5 U78-16 U81-12
6	U153-11 U79-6
7	C14M
	U146-6 R100-1 U117-10 U119-9 U79-7 U85-11
8	GND
9	RESET*
	U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
	U173-1
10	DC3
	U79-10 U78-6 U82-12
11	DC2
	U79-11 U78-15 U81-13
12	DC1
	U79-12 U78-9 U82-11



13 BTO
 U88-3 U85-18 U162-5 U79-13 U83-4 U83-1
 14 DCO
 U79-14 U78-12 U81-14
 15 LDPS*
 U117-9 U79-15 U131-6
 16 +5V

U80 5 LS374 20 F2

1 VBEN
 U80-1 U132-11
 2 DV7
 U86-3 U78-3 U80-2 U84-2 U136-3 U136-2
 3 DB7
 J16-8 U69-13 U80-3
 4 DB5
 J16-6 U69-6 U80-4
 5 DV5
 U82-3 U86-4 U78-4 U81-3 U80-5 U84-5
 6 DV3
 U82-2 U86-7 U78-7 U81-2 U80-6 U84-6
 7 DB3
 J16-4 U66-13 U80-7
 8 DB1
 J16-2 U66-6 U80-8
 9 DV1
 U82-1 U86-8 U78-8 U81-1 U80-9 U84-9
 10 GND
 11 C1M
 U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11
 U84-11
 12 DVO
 U82-17 U86-13 U78-13 U81-17 U80-12 U84-12
 13 DB0
 J16-1 U66-3 U80-13
 14 DB2
 J16-3 U66-10 U80-14
 15 DV2
 U82-16 U86-14 U78-14 U81-16 U80-15 U84-15
 16 DV4
 U82-15 U86-17 U78-17 U81-15 U80-16 U84-16
 17 DB4
 J16-5 U69-3 U80-17
 18 DB6
 J16-7 U69-10 U80-18
 19 DV6
 U78-18 U86-18 U82-4 U81-4 U80-19 U84-19
 20 +5V

U81 5 2114 18 E5

1 DV1
 U82-1 U86-8 U78-8 U81-1 U80-9 U84-9



2	DV3	U82-2	U86-7	U78-7	U81-2	U80-6	U84-6
3	DV5	U82-3	U86-4	U78-4	U81-3	U80-5	U84-5
4	DV6	U78-18	U86-18	U82-4	U81-4	U80-19	U84-19
5	VA	U121-3	U175-5	U116-11	U82-5	U81-5	
6	VB	U121-2	U175-6	U118-14	U82-6	U81-6	
7	VC	U121-1	U175-7	U118-13	U83-15	U83-14	U82-7 U81-7
8	DHIRES	U83-10	U126-8	U82-8	U81-8	U87-17	
9	GND						
10	WE2114*	U82-10	U175-16	U81-10	U144-6		
11	DC6	U78-19	U79-3	U81-11			
12	DC4	U79-5	U78-16	U81-12			
13	DC2	U79-11	U78-15	U81-13			
14	DC0	U79-14	U78-12	U81-14			
15	DV4	U82-15	U86-17	U78-17	U81-15	U80-16	U84-16
16	DV2	U82-16	U86-14	U78-14	U81-16	U80-15	U84-15
17	DV0	U82-17	U86-13	U78-13	U81-17	U80-12	U84-12
18	+5V						

U82 5 2114 18 E4 .

1	DV1	U82-1	U86-8	U78-8	U81-1	U80-9	U84-9
2	DV3	U82-2	U86-7	U78-7	U81-2	U80-6	U84-6
3	DV5	U82-3	U86-4	U78-4	U81-3	U80-5	U84-5
4	DV6	U78-18	U86-18	U82-4	U81-4	U80-19	U84-19
5	VA	U121-3	U175-5	U116-11	U82-5	U81-5	
6	VB	U121-2	U175-6	U118-14	U82-6	U81-6	
7	VC	U121-1	U175-7	U118-13	U83-15	U83-14	U82-7 U81-7
8	DHIRES	U83-10	U126-8	U82-8	U81-8	U87-17	
9	NO CONNECTION						
10	WE2114*						



U82-10 U175-16 U81-10 U144-6
 11 DC1
 U79-12 U78-9 U82-11
 12 DC3
 U79-10 U78-6 U82-12
 13 DC5
 U79-4 U78-5 U82-13
 14 DC7
 U79-2 U78-2 U82-14 U136-4
 15 DV4
 U82-15 U86-17 U78-17 U81-15 U80-16 U84-16
 16 DV2
 U82-16 U86-14 U78-14 U81-16 U80-15 U84-15
 17 DVO
 U82-17 U86-13 U78-13 U81-17 U80-12 U84-12
 18 +5V

U83 5 S151 16 J3

1 BTO
 U88-3 U85-18 U162-5 U79-13 U83-4 U83-1
 2 BT1
 U88-6 U83-2 U85-19 U85-17
 3 BTO*
 U83-3 U162-6
 4 BTO
 U88-3 U85-18 U162-5 U79-13 U83-4 U83-1
 5 BTMUX
 U85-13 U83-5
 6 NO CONNECTION
 7 BL
 U97-16 U83-7 U85-7 U126-16
 8 GND
 9 AILLORES
 U83-9 U87-16
 10 DHIRES
 U83-10 U126-8 U82-8 U81-8 U87-17
 11 INV
 U83-11 U173-5
 12 GND
 13 GND
 14 VC
 U121-1 U175-7 U118-13 U83-15 U83-14 U82-7 U81-7
 15 VC
 U121-1 U175-7 U118-13 U83-15 U83-14 U82-7 U81-7
 16 +5V

U84 5 LS374 20 E2

1 VAEN
 U84-1 U132-3
 2 DV7
 U86-3 U78-3 U80-2 U84-2 U136-3 U136-2
 3 DA7



4	J16-16 U153-5 U84-3 U69-14 DA5
5	J15-14 U69-5 U84-4 DV5
6	U82-3 U86-4 U78-4 U81-3 U80-5 U84-5 DV3
7	U82-2 U86-7 U78-7 U81-2 U80-6 U84-6 DA3
8	J16-12 U66-14 U84-7 DA1
9	J16-10 U10-5 U84-8 U66-5 DV1
10	U82-1 U86-8 U78-8 U81-1 U80-9 U84-9 GND
11	C1M U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11 U84-11
12	DV0 U82-17 U86-13 U78-13 U81-17 U80-12 U84-12
13	DA0 J16-9 U10-4 U84-13 U66-2
14	DA2 J16-11 U10-12 U84-14 U66-11
15	DV2 U82-16 U86-14 U78-14 U81-16 U80-15 U84-15
16	DV4 U82-15 U86-17 U78-17 U81-15 U80-16 U84-16
17	DA4 J16-13 U69-2 U84-17
18	DA6 J16-15 U69-11 U84-18
19	DV6 U78-18 U86-18 U82-4 U81-4 U80-19 U84-19
20	+5V

U85 5 LS374 20 G2

1	GND
2	Q0* U152-12 U144-2 U85-2
3	Q3 J14-37 U85-3 J15-37 J13-37 J12-37 U117-12 R93-1 U154-2
4	C1M* U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2 U126-11 U85-4 U144-5 U132-12 U150-4
5	D1M* U85-5 U86-11
6	DBL U85-6 U88-15
7	SL U97-16 U83-7 U85-7 U126-16
8	H2



9 U121-7 U5-5 U114-12 U85-8
 DH2
 U132-9 U85-9
 10 GND
 11 C14M
 U146-6 R100-1 U117-10 U119-9 U79-7 U85-11
 12 BTMUXD
 U85-12 U89-1
 13 BTMUX
 U85-13 U83-5
 14 BT2
 U88-10 U85-14 U85-16
 15 BT3
 U88-13 U85-15
 16 BT2
 U88-10 U85-14 U85-16
 17 BT1
 U88-6 U83-2 U85-19 U85-17
 18 BT0
 U88-3 U85-18 U162-5 U79-13 U83-4 U83-1
 19 BT1
 U88-6 U83-2 U85-19 U85-17
 20 +5V

U86 5 LS374 20 F3

1 OE374
 U87-15 U86-1
 2 DX7
 U89-4 U86-2 P8-6
 3 DV7
 U86-3 U78-3 U80-2 U84-2 U136-3 U136-2
 4 DV5
 U82-3 U86-4 U78-4 U81-3 U80-5 U84-5
 5 DX5
 U89-12 U86-5 P8-3
 6 DX3
 U88-2 U86-6 P13-4
 7 DV3
 U82-2 U86-7 U78-7 U81-2 U80-6 U84-6
 8 DV1
 U82-1 U86-8 U78-8 U81-1 U80-9 U84-9
 9 DX1
 U88-11 U86-9 P13-5
 10 GND
 11 DIM*
 U85-5 U86-11
 12 DX0
 U88-14 U86-12 P13-2
 13 DVO
 U82-17 U86-13 U78-13 U81-17 U80-12 U84-12
 14 DV2
 U82-16 U86-14 U78-14 U81-16 U80-15 U84-15



15	DX2
	U88-5 U86-15 P13-3
16	DX4
	U89-13 U86-16 P8-4
17	DV4
	U82-15 U86-17 U78-17 U81-15 U80-16 U84-16
18	DV6
	U78-18 U86-18 U82-4 U81-4 U80-19 U84-19
19	DX6
	U89-5 U86-19 P8-5
20	+5V
U87 5 2316 24 341-0032 G5	
1	MIX
	U87-1 U75-5
2	HIRES
	U87-2 U75-7
3	AIISW*
	U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
	U155-3
4	PAGE2
	U87-4 U75-6
5	VBL
	U175-15 U87-5 U97-19 U97-18 U121-19 U154-6
6	FORCPAGE
	U87-6 P10-2 J20-7
7	+5V
8	GND
9	PG2*
	U87-9 U13-13
10	SEL374
	U87-10 U132-13 U132-1
11	COLORKILL*
	U147-13 U87-11 J20-9
12	GND
13	AHIRES
	U88-1 U87-13 U141-1
14	CH80*
	U139-9 U153-12 U87-14
15	OE374
	U87-15 U86-1
16	AILORES
	U83-9 U87-16
17	DHIRES
	U83-10 U126-8 U82-8 U81-8 U87-17
18	NO CONNECTION
19	V4
	U87-19 U1-2 U154-4 U120-12 U1-11
20	GND
21	+5V
22	V2
	U120-14 U154-10 U87-22



23 TEXT
 U87-23 U75-4
 24 +5V

U88 5 LS157 16 G3

1 AHIRES
 U88-1 U87-13 U141-1
 2 DX3
 U88-2 U86-6 P13-4
 3 BT0
 U88-3 U85-18 U162-5 U79-13 U83-4 U83-1
 4 U89-3 U88-4
 5 DX2
 U88-5 U86-15 P13-3
 6 BT1
 U88-6 U83-2 U85-19 U85-17
 7 U89-6 U88-7
 8 GND
 9 U89-11 U88-9
 10 BT2
 U88-10 U85-14 U85-16
 11 DX1
 U88-11 U86-9 P13-5
 12 U89-14 U88-12
 13 BT3
 U88-13 U85-15
 14 DX0
 U88-14 U86-12 P13-2
 15 DBL
 U85-6 U88-15
 16 +5V

U89 5 LS399 16 H3

1 BTMUXD
 U85-12 U89-1
 2 RGB8
 J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
 U89-2
 3 U89-3 U88-4
 4 DX7
 U89-4 U86-2 P8-6
 5 DX6
 U89-5 U86-19 P8-5
 6 U89-6 U88-7
 7 RGB4
 P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7
 J20-4
 8 GND
 9 CKDSP
 U141-11 U89-9
 10 RGB2
 J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13



11 U89-10
 U89-11 U88-9
 12 DX5
 U89-12 U86-5 P8-3
 13 DX4
 U89-13 U86-16 P8-4
 14 U89-14 U88-12
 15 RGB1
 U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
 U90-12
 16 +5V

U90 5 LS153 16 L8

1 GND
 2 C3.5M*
 U135-13 U119-6 U147-1 U90-2
 3 RGB8
 J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
 U89-2
 4 RGB4
 P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7
 J20-4
 5 RGB2
 J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
 U89-10
 6 RGB1
 U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
 U90-12
 7 NTSCA
 P3-7 U90-7
 8 GND
 9 NTSCB*
 U163-9 U90-9
 10 RGB4
 P4-4 P17-4 P3-5 P10-7 U90-4 U90-10 U89-7
 J20-4
 11 RGB8
 J20-5 P4-5 P17-2 P3-6 P10-3 U90-11 U90-3
 U89-2
 12 RGB1
 U89-15 P17-8 P4-2 P3-3 J20-6 P10-4 U90-6
 U90-12
 13 RGB2
 J20-2 P17-6 P4-3 P3-4 P10-6 U90-5 U90-13
 U89-10
 14 C7M
 J15-36 J14-36 J13-36 J12-36 U141-9 U119-2 U146-12
 U90-14
 15 GND
 16 +5V

U91 6 LS323 20 K10



1	U92-7 U91-1
2	DEVSEL6*
	U94-14 U76-9 U91-2
3	IR*/W
	U91-3 U68-11 U158-2 U163-10
4	D1
	J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7
	U91-4 U66-7 P15-3
5	D3
	J15-46 J14-46 J13-46 J12-46 U68-16 J16-20 U109-12
	U91-5 U66-12 P15-5
6	D5
	J15-44 J14-44 J13-44 J12-44 U68-14 J16-22 U111-7
	U91-6 U69-7 P15-7
7	D7
	J15-42 J14-42 J13-42 J12-42 U68-12 J16-24 U111-12
	U101-5 U91-7 U69-12 P15-9
8	U92-2 U91-8
9	U92-9 U91-9 R66-1
10	GND
11	DWRPROTT
	U167-17 U91-11
12	Q3*
	U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
	U131-5 R102-2 R103-1 U146-1
13	D6
	J15-43 J14-43 J13-43 J12-43 U68-13 J16-23 U111-9
	U91-13 U69-9 P15-8
14	D4
	J15-45 J14-45 J13-45 J12-45 U68-15 J16-21 U111-4
	U91-14 U69-4 P15-6
15	D2
	J15-47 J14-47 J13-47 J12-47 U68-17 J16-19 U109-9
	U91-15 U66-9 P15-4
16	D0
	J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4
	U91-16 U66-4 P15-2
17	NO CONNECTION
18	U92-8 U91-18
19	U92-6 U91-19
20	+5V

U92 6 S471 20 341-0028 K11

1	U93-15 U92-1
2	U92-2 U91-8
3	U92-3 U94-11
4	U163-13 U94-12 U92-4
5	U161-11 U92-5 R67-2
6	U92-6 U91-19
7	U92-7 U91-1
8	U92-8 U91-18
9	U92-9 U91-9 R66-1



10 GND
 11 U93-6 U92-11
 12 U93-14 U92-12
 13 U93-4 U92-13
 14 U93-3 U92-14
 15 MOTON*
 U179-1 U92-15 U178-15 R65-2 U164-12 U92-16
 16 MOTON*
 U179-1 U92-15 U178-15 R65-2 U164-12 U92-16
 17 U93-7 U92-17
 18 U93-5 U92-18
 19 DWRDATA
 U166-18 U92-19 U93-2
 20 +5V

U93 6 LS174 16 K12

1 S5B
 R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
 U114-7 U116-5
 2 DWRDATA
 U166-18 U92-19 U93-2
 3 U93-3 U92-14
 4 U93-4 U92-13
 5 U93-5 U92-13
 6 U93-6 U92-11
 7 U93-7 U92-17
 8 GND
 9 Q3*
 U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
 U131-5 R102-2 R103-1 U146-1
 10 U161-13 U93-10
 11 U93-12 U161-12 U93-11
 12 U93-12 U161-12 U93-11
 13 DRDATA
 U166-17 U93-13
 14 U93-14 U92-12
 15 U93-15 U92-1
 16 +5V

U94 6 9334 16 L13

1 A1
 J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
 U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
 2 A2
 J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
 U177-2 U94-2 U75-2 U73-36 U63-14
 3 A3
 J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
 U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
 4 VA1
 U166-12 U175-4 U94-4
 5 VB1



6	U167-12 U175-3 U94-5 VC1 U166-13 U175-2 U94-6
7	PDPH3 U167-13 U94-7
8	GND
9	MOTEN* U164-11 U94-9
10	DEXT* U167-18 U94-10 U178-2 U163-1
11	U92-3 U94-11
12	U163-13 U94-12 U92-4
13	A0 J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
14	DEVSEL6* U94-14 U76-9 U91-2
15	IORESET* J12-31 J13-31 J14-31 J15-31 P14-6 U164-2 U164-4 U96-4 U94-15
16	+5V

U96 6-8 556 14 L10	
1	NO CONNECTION
2	DT1M R31-1 U164-10 C7-2 U96-2 U96-6
3	NO CONNECTION
4	IORESET* J12-31 J13-31 J14-31 J15-31 P14-6 U164-2 U164-4 U96-4 U94-15
5	MOTON U164-13 U96-5
6	DT1M R31-1 U164-10 C7-2 U96-2 U96-6
7	GND
8	U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1 X3-1 C75-1
9	U96-9 U139-1
10	U96-10 U162-2 U139-2
11	NO CONNECTION
12	U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1 X3-1 C75-1
13	U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1 X3-1 C75-1
14	+5V

U97 7 6522 40 B5	
1	GND
2	BKSW1 U97-2 U10-3
3	BKSW2 U97-3 U10-6



4 BKS^W3
U97-4 U10-11

5 NO CONNECTION

6 IRQ4*
U97-6 J15-30 P2-3 U148-5

7 IRQ3*
U97-7 J14-30 P2-4 U148-4

8 AIIS^W*
U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
U155-3

9 IRQ*
U97-21 U65-4 U98-26 U97-9 U73-21

10 SNDO
P9-7 U97-10

11 SND1
P9-6 U97-11

12 SND2
P9-5 U97-12

13 SND3
P9-4 U97-13

14 SND4
P9-3 U97-14

15 SND5
P9-2 U97-15

16 BL
U97-16 U83-7 U85-7 U126-16

17 IONMI*
J15-29 U97-17 J14-29 J13-29 U139-13 J12-29

18 VBL
U175-15 U87-5 U97-19 U97-18 U121-19 U154-6

19 VBL
U175-15 U87-5 U97-19 U97-18 U121-19 U154-6

20 +5V

21 IRQ*
U97-21 U65-4 U98-26 U97-9 U73-21

22 IR/W*
U98-28 U97-22 U112-14 U73-22 U163-11 U65-34 U160-12

23 FFEX*
U97-23 U148-10 U112-6

24 CS6522
R68-2 U73-24 C59-1 U97-24

25 PRE1M
J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5
U97-25 U139-10 R94-1

26 ID7
U98-25 U65-26 U97-26 U72-22 U73-26 U64-17 U68-8

27 ID6
U98-24 U65-27 U97-27 U72-21 U73-27 U64-16 U68-7

28 ID5
U98-23 U65-28 U97-28 U72-20 U73-28 U64-15 U68-6

29 ID4
U98-22 U65-29 U97-29 U72-19 U73-29 U64-14 U68-5



```

30      ID3
        U98-21 U65-30 U97-30 U72-18 U73-30 U64-13 U68-4
31      ID2
        U98-20 U65-31 U97-31 U72-17 U73-31 U64-11 U68-3
32      ID1
        U98-19 U65-32 U97-32 U72-16 U73-32 U64-10 U68-2
33      ID0
        U98-18 U65-33 U97-33 U72-15 U73-33 U64-9 U68-1
34      RESET*
        U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
        U173-1
35      A3
        J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
        U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
36      A2
        J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
        U177-2 U94-2 U75-2 U73-36 U63-14
37      A1
        J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
        U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
38      A0
        J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
        U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
39      KBDINT*
        U106-6 U97-39
40      CKIRQ*
        U72-13 R79-2 U97-40
    
```

U98 7 6551 28 B2

```

1      GND
2      +5V
3      SEL6551*
        U98-3 U76-7 U148-12
4      RESET*
        U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
        U173-1
5      NO CONNECTION
6      ACIACK
        U98-6 U140-5
7      NO CONNECTION
8      RTS*
        U99-12 U98-8
9      CTS*
        U100-6 U98-9
10     TXD*
        U99-4 U98-10
11     DTR*
        U99-9 U98-11
12     RXD*
        U100-3 U98-12
13     A0
        J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
    
```



14 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
 AI
 J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
 U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
 15 +5V
 16 DCD*
 U100-11 U98-16
 17 DSR*
 U100-8 U98-17
 18 ID0
 U98-18 U65-33 U97-33 U72-15 U73-33 U64-9 U68-1
 19 ID1
 U98-19 U65-32 U97-32 U72-16 U73-32 U64-10 U68-2
 20 ID2
 U98-20 U65-31 U97-31 U72-17 U73-31 U64-11 U68-3
 21 ID3
 U98-21 U65-30 U97-30 U72-18 U73-30 U64-13 U68-4
 22 ID4
 U98-22 U65-29 U97-29 U72-19 U73-29 U64-14 U68-5
 23 ID5
 U98-23 U65-28 U97-28 U72-20 U73-28 U64-15 U68-6
 24 ID6
 U98-24 U65-27 U97-27 U72-21 U73-27 U64-16 U68-7
 25 ID7
 U98-25 U65-26 U97-26 U72-22 U73-26 U64-17 U68-8
 26 IRQ*
 U97-21 U65-4 U98-26 U97-9 U73-21
 27 C1M
 U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11
 U84-11
 28 IR/W*
 U98-28 U97-22 U112-14 U73-22 U163-11 U65-34 U160-12

U99 7 1488 14 J2

1 -12V
 2 NO CONNECTION
 3 NO CONNECTION
 4 TXD*
 U99-4 U98-10
 5 +5V
 6 PTXD
 U172-12 U99-6
 7 GND
 8 PDTR
 U172-17 U99-8
 9 DTR*
 U99-9 U98-11
 10 +5V
 11 PRTS
 U172-14 U99-11
 12 RTS*
 U99-12 U98-8



13 +5V
14 +12V

U100 7 1489 14 K2

1 PRXD
U172-13 U100-1
2 NO CONNECTION
3 RXD*
U100-3 U98-12
4 PCTS
R87-2 U172-15 U100-4
5 NO CONNECTION
6 CTS*
U100-6 U98-9
7 GND
8 DSR*
U100-8 U98-17
9 NO CONNECTION
10 PDSR
R88-2 U172-16 U100-10
11 DCD*
U100-11 U98-16
12 NO CONNECTION
13 PDCD
R89-2 U172-18 U100-13
14 +5V

U101 7 LS251 16 L7

1 SCQ
U160-3 U73-18 U160-5 U101-1
2 SW2UF
U101-2 U169-14
3 SW1/MCNSWUF
R85-2 U169-19 U73-39 U101-3
4 SWOUF
U101-4 U169-13
5 D7
J15-42 J14-42 J13-42 J12-42 U68-12 J16-24 U111-12
U101-5 U91-7 U69-12 P15-9
6 NO CONNECTION
7 C06X*
U101-7 U77-9
8 GND
9 A2
J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
U177-2 U94-2 U75-2 U73-36 U63-14
10 A1
J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
11 A0
J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
U101-11 U94-13 U73-38 U75-13 U63-18 U177-13



```

12      MUX1
        U101-12 U175-14 U5-13
13      PDLOT*
        U101-13 U105-7 R57-2
14      IRQ1*
        U101-14 J12-30 P2-7 U148-1
15      IRQ2*
        U101-15 J13-30 P2-6 U148-2
16      +5V

U103 7 380N-8N 8 M11
1      NO CONNECTION
2      C78-1 U103-2 R34-2 R36-1
3      GND
4      GND
5      GND
6      R69-2 U103-6
7      +12V
8      C13-1 U103-8

U105 7 9708 16 M9
1      AXCO
        U161-5 U75-12 U105-1
2      PDL2
        U160-4 U105-2 U75-10
3      PDLEN
        U105-3 U75-11
4      TCAP
        C15-1 U105-4
5      GND
6      U105-6 R39-2 C9-1
7      PDLOT*
        U101-13 U105-7 R57-2
8      U105-8 R37-2 R38-1 C76-1
9      NO CONNECTION
10     R9-1 T1-1 X1-1 U105-10
11     Y1/XCOUF
        U161-6 U169-17 U105-11
12     PX1/SER
        U105-12 U161-8 U160-9 U169-18
13     YOUF
        U105-13 U169-15
14     +5V
15     XOUF
        U105-15 U169-12
16     PDLO
        U105-16 U75-9

U106 8 LS74 14 H11
1      CLRSTRB*
        U106-1 U77-14
2      S5B

```

15.75



```

R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
U114-7 U116-5
3 U139-3 U106-3
4 S5B
R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
U114-7 U116-5
5 U106-5 U111-14
6 KBDINT*
U106-6 U97-39
7 GND
8 NO CONNECTION
9 KAPPLEII*
U106-9 X3-2 U111-6
10 S5B
R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
U114-7 U116-5
11 ANYKEY
U107-4 R40-1 X2-2 U106-11 U109-3
12 APPLEII*
J7-5 P11-5 U106-12 U164-5
13 S5B
R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
U114-7 U116-5
14 +5V

U107 8 AY3600 40 H14
1 C16-1 U107-1
2 U107-2 C16-2 R47-2
3 U107-3 R47-1
4 ANYKEY
U107-4 R40-1 X2-2 U106-11 U109-3
5 NO CONNECTION
6 KSHIFT*
U107-6 U109-6
7 ASCII7
U107-7 U111-13
8 ASCII6
U107-8 U111-11
9 ASCII5
U107-9 U111-5
10 ASCII4
U107-10 U111-2
11 ASCII3
U107-11 U109-14
12 ASCII2
U107-12 U109-11
13 ASCII1
U107-13 U109-5
14 ASCII0
U107-14 U109-2
15 GND
16 DTRDY*
    
```



```

17      U107-16 U162-1
        KY0
        J7-1 U107-17
18      KY1
        J7-2 U107-18
19      KY2
        J7-4 U107-19
20      KY3
        J7-6 U107-20
21      KY4
        J7-8 U107-21
22      KY5
        J7-10 U107-22
23      KY6
        J7-23 U107-23
24      KY7
        J7-25 U107-24
25      KY8
        J7-12 U107-25
26      KY9
        J7-22 U107-26
27      -12FV
28      CONTROL*
        J7-11 U179-13 U107-28 U109-10 P11-2
29      SHIFT*
        J7-24 P11-6 U107-29
30      +5FV
31      U107-31 C22-1
32      NO CONNECTION
33      KX7UF
        U171-14 U107-33
34      KX6UF
        U171-19 U107-34
35      KX5UF
        U171-16 U107-35
36      KX4UF
        U171-18 U107-36
37      KX3UF
        U171-17 U107-37
38      KX2UF
        U171-13 U107-38
39      KX1UF
        U171-15 U107-39
40      KX0UF
        U171-12 U107-40

U109 8 LS257 16 J12
1      A3
        J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
        U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
2      ASCII0
        U107-14 U109-2
    
```



```

3      ANYKEY
      U107-4 R40-1 X2-2 U106-11 U109-3
4      DO
      J15-49 J14-49 J13-49 J12-49 U68-19 J16-17 U109-4
      U91-16 U66-4 P15-2
5      ASCII1
      U107-13 U109-5
6      KSHIFT*
      U107-6 U109-6
7      D1
      J15-48 J14-48 J13-48 J12-48 U68-18 J16-18 U109-7
      U91-4 U66-7 P15-3
8      GND
9      D2
      J15-47 J14-47 J13-47 J12-47 U68-17 J16-19 U109-9
      U91-15 U66-9 P15-4
10     CONTROL*
      J7-11 U179-13 U107-28 U109-10 P11-2
11     ASCII2
      U107-12 U109-11
12     D3
      J15-46 J14-46 J13-46 J12-46 U68-16 J16-20 U109-12
      U91-5 U66-12 P15-5
13     CAPLCK*
      J7-9 P11-3 U109-13
14     ASCII3
      U107-11 U109-14
15     KBD*
      U109-15 U111-15 U77-15
16     +5V

U111 8 LS257 16 H12
1      A3
      J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
      U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
2      ASCII4
      U107-10 U111-2
3      APPLE1*
      J7-7 P11-4 U111-3
4      D4
      J15-45 J14-45 J13-45 J12-45 U68-15 J16-21 U111-4
      U91-14 U69-4 P15-6
5      ASCII5
      U107-9 U111-5
6      KAPPLEII*
      U106-9 X3-2 U111-6
7      D5
      J15-44 J14-44 J13-44 J12-44 U68-14 J16-22 U111-7
      U91-6 U69-7 P15-7
8      GND
9      D6
      J15-43 J14-43 J13-43 J12-43 U68-13 J16-23 U111-9
    
```



```

10      U91-13 U69-9 P15-8
11      Q9-3 R43-1 U111-10
11      ASCII6
11      U107-8 U111-11
12      D7
12      J15-42 J14-42 J13-42 J12-42 U68-12 J16-24 U111-12
12      U101-5 U91-7 U69-12 P15-9
13      ASCII7
13      U107-7 U111-13
14      U106-5 U111-14
15      KBD*
15      U109-15 U111-15 U77-15
16      +5V

U112 4 LS139 16 C8
1      U112-1 U71-9
2      A4
2      J15-6 J14-6 J13-6 J12-6 U9-6 U77-1 U76-1
2      U112-2 U63-9
3      A5
3      J15-7 J14-7 J13-7 J12-7 U9-10 U76-2 U77-2
3      U112-3 U63-7
4      FFCX*
4      U165-7 U112-4
5      FFDX*
5      U112-5 U73-23 U148-9
6      FFEX*
6      U97-23 U148-10 U112-6
7      NO CONNECTION
8      GND
9      NO CONNECTION
10     NO CONNECTION
11     NO CONNECTION
12     U72-3 U112-12
13     CO7X*
13     U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13
14     IR/W*
14     U98-28 U97-22 U112-14 U73-22 U163-11 U65-34 U160-12
15     IOSTOPD*
15     U123-9 U112-15 U180-3
16     +5V

U113 8 556 14 A5
1      NO CONNECTION
2      UPRST*
2      C20-1 U113-6 U113-2 R44-2 X4-2 X6-1
3      NO CONNECTION
4      +5V
5      TRESET
5      U113-5 R92-1
6      UPRST*
6      C20-1 U113-6 U113-2 R44-2 X4-2 X6-1
    
```



7 GND
 8 U113-8 R46-2 U113-12 C21-1
 9 FLASH
 U113-9 U136-5
 +5V
 11 NO CONNECTION
 12 U113-8 R46-2 U113-12 C21-1
 13 U113-13 R45-2 R46-1
 14 +5V

U114 9 LS161 16 F10

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 3 GND
 4 GND
 5 GND
 6 GND
 7 S5B
 R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
 U114-7 U116-5
 8 GND
 9 HPE*
 U135-3 U152-2 U114-9 U116-9 U116-12
 10 S5B
 R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10
 U114-7 U116-5
 11 H3
 U114-11 U1-5 U121-6
 12 H2
 U121-7 U5-5 U114-12 U85-8
 13 H1
 U114-13 U2-11 U155-9
 14 H0
 U114-14 U2-5
 15 U116-10 U114-15 U116-7
 16 +5V

U116 9 LS161 16 F11

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 3 GND
 4 GND
 5 S5B
 R107-2 U106-2 U106-4 U106-10 U106-13 U93-1 U114-10



6 U114-7 U116-5
 COMP
 U116-6 U121-17 U118-3
 7 U116-10 U114-15 U116-7
 8 GND
 9 HPE*
 U135-3 U152-2 U114-9 U116-9 U116-12
 10 U116-10 U114-15 U116-7
 11 VA
 U121-3 U175-5 U116-11 U82-5 U81-5
 12 HPE*
 U135-3 U152-2 U114-9 U116-9 U116-12
 13 H5
 U121-4 U1-14 U116-13
 14 H4
 U121-5 U1-3 U116-14
 15 U118-5 U116-15 U118-10 U118-7
 16 +5V

U117 9 S195 16 D10

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 Q3*
 U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
 U131-5 R102-2 R103-1 U146-1
 3 Q3*
 U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
 U131-5 R102-2 R103-1 U146-1
 4 S5D
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
 U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
 5 S5D
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
 U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
 6 S5D
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
 U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
 7 S5D
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
 U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
 8 GND
 9 LDPS*
 U117-9 U79-15 U131-6
 10 C14M
 U146-6 R100-1 U117-10 U119-9 U79-7 U85-11
 11 Q3*
 U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
 U131-5 R102-2 R103-1 U146-1
 12 Q3
 J14-37 U85-3 J15-37 J13-37 J12-37 U117-12 R93-1
 U154-2



13 NO CONNECTION
 14 Q1
 U6-8 U131-4 U3-8 U124-12 U117-14
 15 Q0
 U119-13 U150-10 U117-15 U131-2 U78-11
 16 +5V

U118 9 LS161 16 G11

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 3 COMP
 U116-6 U121-17 U118-3
 4 GND
 5 U118-5 U116-15 U118-10 U118-7
 6 S5060
 U120-3 U121-15 U118-6
 7 U118-5 U116-15 U118-10 U118-7
 8 GND
 9 UUTSUNK*
 J19-1 U118-9 U162-8 U120-9
 10 U118-5 U116-15 U118-10 U118-7
 11 V1
 U121-22 U2-13 U118-11 U13-5 U13-3
 12 VO
 U121-23 U13-11 U118-12
 13 VC
 U121-1 U175-7 U118-13 U83-15 U83-14 U82-7 U81-7
 14 VB
 U121-2 U175-6 U118-14 U82-6 U81-6
 15 U120-5 U118-15 U120-4 U120-10 U120-7
 16 +5V

U119 9 S175 16 B12

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 C7M
 J15-36 J14-36 J13-36 J12-36 U141-9 U119-2 U146-12
 U90-14
 3 C7M*
 U119-3 U153-13 U119-4
 4 C7M*
 U119-3 U153-13 U119-4
 5 U119-5 U146-11
 6 C3.5M*
 U135-13 U119-6 U147-1 U90-2
 7 C3.5M
 U132-10 U141-10 U146-13 U119-7 J20-3



8 GND
 9 C14M
 U146-6 R100-1 U117-10 U119-9 U79-7 U85-11
 10 CIM
 U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11
 U84-11
 11 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 12 PRE1M
 J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5
 U97-25 U139-10 R94-1
 13 QO
 U119-13 U150-10 U117-15 U131-2 U78-11
 14 RAS
 U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
 15 NO CONNECTION
 16 +5V

U120 9 LS161 16 G12

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 3 S5060
 U120-3 U121-15 U118-6
 4 U120-5 U118-15 U120-4 U120-10 U120-7
 5 U120-5 U118-15 U120-4 U120-10 U120-7
 6 GND
 7 U120-5 U118-15 U120-4 U120-10 U120-7
 8 GND
 9 UUTSUNK*
 J19-1 U118-9 U162-8 U120-9
 10 U120-5 U118-15 U120-4 U120-10 U120-7
 11 V5
 U120-11 U154-9
 12 V4
 U87-19 U1-2 U154-4 U120-12 U1-11
 13 V3
 U120-13 U1-15 U154-5 U1-6
 14 V2
 U120-14 U154-10 U87-22
 15 U162-9 U120-15
 16 +5V

U121 9 2316 24 341-0030 G9

1 VC
 U121-1 U175-7 U118-13 U83-15 U83-14 U82-7 U81-7
 2 VB
 U121-2 U175-6 U118-14 U82-6 U81-6



3 VA
 U121-3 U175-5 U116-11 U82-5 U81-5
 4 H5
 U121-4 U1-14 U116-13
 5 H4
 U121-5 U1-3 U116-14
 6 H3
 U114-11 U1-5 U121-6
 7 H2
 U121-7 U5-5 U114-12 U85-8
 8 V2*V5
 U121-8 U154-8 U5-3
 9 RSYNCH
 U121-9 U126-4
 10 RCOLRGT
 U121-10 U126-7
 11 RTCWRT
 U126-13 U121-11
 12 GND
 13 RBL
 U121-13 U154-12
 14 RRFSH
 U121-14 U126-3
 15 S5060
 U120-3 U121-15 U118-6
 16 RFIELD
 U121-16 U126-18
 17 COMP
 U116-6 U121-17 U118-3
 18 FIELDIN
 J19-3 U121-18 R63-1
 19 VBL
 U175-15 U87-5 U97-19 U97-18 U121-19 U154-6
 20 GND
 21 +5V
 22 V1
 U121-22 U2-13 U118-11 U13-5 U13-3
 23 V0
 U121-23 U13-11 U118-12
 24 +5V

U123 9 S74 14 D11

1 UUTRST*
 U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
 U120-1 U118-1 U117-1
 2 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 3 AX
 U124-9 U123-3
 4 SSD
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4



```

5      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
      PRE1M
      J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5
      U97-25 U139-10 R94-1
6      PRE1M*
      J12-38 J13-38 J14-38 J15-38 U123-6 R95-1
7      GND
8      NO CONNECTION
9      IOSTOPD*
      U123-9 U112-15 U180-3
10     SSD
      R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
11     CIM
      U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11
      U84-11
12     FSPACE*
      U180-1 U176-1 U165-12 U123-12 U148-8
13     SSD
      R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
14     +5V

U124 9 S74 14 A11
1      SSD
      R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
2      U124-2 U152-8
3      U124-3 U146-3
4      SSD
      R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
5      PHO
      J15-19 U136-11 J14-19 J13-19 J12-19 U124-5 U65-37
      R96-1
6      NO CONNECTION
7      GND
8      AX*
      U124-8 U2-2 U13-2 U9-2 U5-2 J17-23
9      AX
      U124-9 U123-3
10     SSD
      R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
      U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
11     C14M*
      U141-13 U146-8 U124-11 J21-2
12     Q1
      U6-8 U131-4 U3-8 U124-12 U117-14
13     UUTRST*
      U124-13 U119-1 U123-1 R53-2 U114-1 J21-3 U116-1
      U120-1 U118-1 U117-1
14     +5V
    
```



```

U126 9 LS374 20 G10
1      GND
2      RFSH
      U158-13 U11-16 U126-2
3      RRFSH
      U121-14 U126-3
4      RSYNCH
      U121-9 U126-4
5      SYNCH
      J20-1 P17-10 U126-5 P3-2 P4-6
6      COLRGATE
      J20-8 U147-2 U126-6
7      RCOLRGT
      U121-10 U126-7
8      DHIRES
      U83-10 U126-8 U82-8 U81-8 U87-17
9      RDHIRES
      U175-1 U11-17 U3-2 U6-3 U126-9
10     GND
11     CIM*
      U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
      U126-11 U85-4 U144-5 U132-12 U150-4
12     TCWRT
      U126-12 U144-4
13     RTCWRT
      U126-13 U121-11
14     U126-14 U152-1 U154-11
15     U158-12 U126-15
16     BL
      U97-16 U83-7 U85-7 U126-16
17     U126-17 U152-3
18     RFIELD
      U121-16 U126-18
19     FIELDOUT
      U126-19 J19-2
20     +5V

U128 2 S86 14 E10
1      A10
      J15-12 J14-12 J13-12 J12-12 U128-1 U74-3 U71-4
      U67-9
2      PCAS3*
      U128-2 U128-13 P1-6 U3-11 U4-8
3      U5-12 U128-3
4      A13
      J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
      U71-1 U70-7 U6-5 U3-6 U174-7
5      PRAS0.3
      U128-5 U11-14 U12-1 U3-15 U6-4
6      U128-9 U128-6
7      GND
    
```



8	U13-12 U128-8
9	U128-9 U128-6
10	PCAS1*
	P1-3 U4-4 U6-12 U128-10
11	U9-4 U128-11
12	All
	J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
	U150-13 U74-5 U67-12 U3-5 U174-5
13	PCAS3*
	U128-2 U128-13 P1-6 U3-11 U4-8
14	+5V

U131	3-9 LS51	14	B11
1	U152-6 U144-10 U131-1		
2	Q0		
	U119-13 U150-10 U117-15 U131-2 U78-11		
3	U131-3 U135-5		
4	Q1		
	U6-8 U131-4 U3-8 U124-12 U117-14		
5	Q3*		
	U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12		
	U131-5 R102-2 R103-1 U146-1		
6	LDPS*		
	U117-9 U79-15 U131-6		
7	GND		
8	ZPAGE*		
	U11-1 U174-15 U67-1 U131-8 U70-1		
9	DMA1		
	U131-11 U163-6 U131-10 U131-9 U160-13		
10	DMA1		
	U131-11 U163-6 U131-10 U131-9 U160-13		
11	DMA1		
	U131-11 U163-6 U131-10 U131-9 U160-13		
12	U135-6 U131-12		
13	U144-9 U131-13 U155-4		
14	+5V		

U132	3-5-7 LS86	14	D4
1	SEL374		
	U87-10 U132-13 U132-1		
2	C1M		
	U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11		
	U84-11		
3	VAEN		
	U84-1 U132-3		
4	Z0		
	U72-5 U132-4 U73-10		
5	PPA8		
	U158-4 U152-4 U158-5 U65-17 U64-23 U132-5		
6	U67-2 U132-6		
7	GND		
8	U140-3 U132-8		

apple computer inc.

```

9          DH2
          U132-9 U85-9
10         C3.5M
          U132-10 U141-10 U146-13 U119-7 J20-3
11         VBEN
          U80-1 U132-11
12         CIM*
          U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
          U126-11 U85-4 U144-5 U132-12 U150-4
13         SEL374
          U87-10 U132-13 U132-1
14         +5V

U135 3-9 LS260 14 C9
1         CIM
          U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11
          U84-11
2         HPEDIS
          U135-2 J21-1 R54-1
3         HPE*
          U135-3 U152-2 U114-9 U116-9 U116-12
4         PA11
          U64-18 U144-12 U65-20 U67-13 U135-4
5         U131-3 U135-5
6         U135-6 U131-12
7         GND
8         PA13
          U65-23 U135-8 U70-6
9         PA12
          U65-22 U135-9 U70-3
10        PA15
          U135-10 U11-4 U70-13 U65-25
11        PA14
          U65-24 U135-11 U70-10
12        Q3*
          U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
          U131-5 R102-2 R103-1 U146-1
13        C3.5M*
          U135-13 U119-6 U147-1 U90-2
14        +5V

U136 2-5 LS51 14 H4
1         SYNC
          U65-7 U136-1 J15-35 U174-3 J14-35 J13-35 J12-35
2         DV7
          U86-3 U78-3 U80-2 U84-2 U136-3 U136-2
3         DV7
          U86-3 U78-3 U80-2 U84-2 U136-3 U136-2
4         DC7
          U79-2 U78-2 U82-14 U136-4
5         FLASH
          U113-9 U136-5
    
```



6 U173-2 U136-6
 7 GND
 8 CLKBK
 U136-8 U10-9
 9 R/W*
 J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
 U176-6 U165-11 U160-11
 10 IND*
 U136-10 U147-5 U174-12
 11 PHO
 J15-19 U136-11 J14-19 J13-19 J12-19 U124-5 U65-37
 R96-1
 12 Q3*
 U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
 U131-5 R102-2 R103-1 U146-1
 13 NO CONNECTION
 14 +5V

U139 5-8 LS132 14 H10

1 U96-9 U139-1
 2 U96-10 U162-2 U139-2
 3 U139-3 U106-3
 4 U139-4 U139-11
 5 RESETLK*
 U179-14 U73-6 P19-5 U139-5
 6 NMI*
 U139-6 U65-6 U155-2
 7 GND
 8 U139-8 U154-1
 9 CH80*
 U139-9 U153-12 U87-14
 10 PRE1M
 J12-40 J13-40 J14-40 J15-40 U119-12 U73-25 U123-5
 U97-25 U139-10 R94-1
 11 U139-4 U139-11
 12 KRESET*
 U139-12 R80-1 C71-1 U179-15 J7-15
 13 IONMI*
 J15-29 U97-17 J14-29 J13-29 U139-13 J12-29
 14 +5V

U140 7-9 S74 14 D5

1 S5C
 R108-2 U140-1 U140-4 U140-10 U140-13
 2 U140-6 U140-2
 3 U140-3 U132-8
 4 S5C
 R108-2 U140-1 U140-4 U140-10 U140-13
 5 ACIACK
 U98-6 U140-5
 6 U140-6 U140-2
 7 GND



Apple Computer Inc.

8 NO CONNECTION
 9 U152-9 U140-9
 10 SSC
 R108-2 U140-1 U140-4 U140-10 U140-13
 11 U152-10 U152-11 U140-11
 12 PHASEN
 U140-12 U180-13
 13 SSC
 R108-2 U140-1 U140-4 U140-10 U140-13
 14 +5V

U141 4-5 LS00 14 H2

1 AHIRE5
 U88-1 U87-13 U141-1
 2 U141-2 U141-8
 3 U141-12 U141-3
 4 U141-4 U150-6
 5 PH2M
 P19-10 U141-5 U65-39 U176-17
 6 GPPI
 U162-3 U76-4 U141-6 U74-4
 7 GND
 8 U141-2 U141-8
 9 C7M
 J15-36 J14-36 J13-36 J12-36 U141-9 U119-2 U146-12
 U90-14
 10 C3.5M
 U132-10 U141-10 U146-13 U119-7 J20-3
 11 CKDSP
 U141-11 U89-9
 12 U141-12 U141-3
 13 C14M*
 U141-13 U146-8 U124-11 J21-2
 14 +5V

U144 3-5 LS20 14 B10

1 ENCWRT
 U177-10 U144-1
 2 Q0*
 U152-12 U144-2 U85-2
 3 NO CONNECTION.
 4 TCWRT
 U126-12 U144-4
 5 CIM*
 U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
 U126-11 U85-4 U144-5 U132-12 U150-4
 6 WE2114*
 U82-10 U175-16 U81-10 U144-6
 7 GND
 8 DMAOK
 U144-8 J12-27 J15-27 J14-27 J13-27
 9 U144-9 U131-13 U155-4



10 U152-6 U144-10 U131-1
 11 NO CONNECTION
 12 PA11
 U64-18 U144-12 U65-20 U67-13 U135-4
 13 TROMSEL
 U162-12 U144-13
 14 +5V

U146 9 S86 14 B13

1 Q3*
 U93-9 U136-12 U91-12 U117-11 U117-2 U117-3 U135-12
 U131-5 R102-2 R103-1 U146-1
 2
 Q11-3 R51-1 U146-10 U146-5 U146-2
 3 U124-3 U146-3
 4 SSD
 R109-2 U124-1 U124-4 U124-10 U10-13 U146-4 U123-4
 U123-10 U123-13 U117-4 U117-5 U117-6 U117-7
 5
 Q11-3 R51-1 U146-10 U146-5 U146-2
 6 C14M
 U146-6 R100-1 U117-10 U119-9 U79-7 U85-11
 7 GND
 8 C14M*
 U141-13 U146-8 U124-11 J21-2
 9 GND
 10
 Q11-3 R51-1 U146-10 U146-5 U146-2
 11 U119-5 U146-11
 12 C7M
 J15-36 J14-36 J13-36 J12-36 U141-9 U119-2 U146-12
 U90-14
 13 C3.5M
 U132-10 U141-10 U146-13 U119-7 J20-3
 14 +5V

U147 4-5 LS11 14 K8

1 C3.5M*
 U135-13 U119-6 U147-1 U90-2
 2 COLRGATE
 J20-8 U147-2 U126-6
 3 A15
 J15-17 J14-17 J13-17 J12-17 U3-4 U147-3 U70-12
 U174-16 U6-7
 4 A14
 J15-16 J14-16 J13-16 J12-16 U3-7 U147-4 U70-9
 U6-16 U174-4
 5 IND*
 U136-10 U147-5 U174-12
 6 C-FXXX
 U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1
 7 GND



```

8      CXXX
      U176-15 U74-6 U150-1 U147-8
9      C-FXXX
      U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1
10     IOEN
      U73-8 P19-3 U147-10
11     U147-11 U155-13
12     COLORBURST
      R55-1 U147-12
13     COLORKILL*
      U147-13 U87-11 J20-9
14     +5V

U148 4 LS21 14 J4
1      IRQ1*
      U101-14 J12-30 P2-7 U148-1
2      IRQ2*
      U101-15 J13-30 P2-6 U148-2
3      NO CONNECTION
4      IRQ3*
      U97-7 J14-30 P2-4 U148-4
5      IRQ4*
      U97-6 J15-30 P2-3 U148-5
6      U155-8 U148-6
7      GND
8      FSPACE*
      U180-1 U176-1 U165-12 U123-12 U148-8
9      FFDX*
      U112-5 U73-23 U148-9
10     FFEX*
      U97-23 U148-10 U112-6
11     NO CONNECTION
12     SEL6551*
      U98-3 U76-7 U148-12
13     C07X*
      U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13
14     +5V

U150 2-4 S10 14 E9
1      CXXX
      U176-15 U74-6 U150-1 U147-8
2      GPH2
      U150-2 U77-6 U162-4
3      SELIM
      U180-15 U174-2 U150-3 U73-9 P19-2
4      CIM*
      U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
      U126-11 U85-4 U144-5 U132-12 U150-4
5      C07X*
      U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13
6      U141-4 U150-6
7      GND
    
```



```

8      RAMR/W*
      J17-11 U150-8
9      RAS
      U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
10     Q0
      U119-13 U150-10 U117-15 U131-2 U78-11
11     WRAMEN
      U180-12 U150-11
12     IOSTRB*
      J15-20 U150-12 J14-20 J13-20 J12-20
13     A11
      J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
      U150-13 U74-5 U67-12 U3-5 U174-5
14     +5V

U152 3-9 LS00 14 E7
1      U126-14 U152-1 U154-11
2      HPE*
      U135-3 U152-2 U114-9 U116-9 U116-12
3      U126-17 U152-3
4      PPA8
      U158-4 U152-4 U158-5 U65-17 U64-23 U132-5
5      U158-8 U152-5
6      U152-6 U144-10 U131-1
7      GND
8      U124-2 U152-8
9      U152-9 U140-9
10     U152-10 U152-11 U140-11
11     U152-10 U152-11 U140-11
12     Q0*
      U152-12 U144-2 U85-2
13     RAS
      U152-13 U150-9 U119-14 U12-2 U12-5 U12-10 U12-13
14     +5V

U153 2-4-5 LS08 14 E8
1      U158-11 U153-1 U180-4
2      CIM*
      U153-2 U123-2 U119-11 U120-2 U118-2 U116-2 U114-2
      U126-11 U85-4 U144-5 U132-12 U150-4
3      AY*
      U13-14 U11-3 U9-14 U5-14 U2-14 U153-3 U6-15
      U3-1
4      S399
      U174-14 U153-4
5      DA7
      J16-16 U153-5 U84-3 U69-14
6      U10-1 U153-6
7      GND
8      TROMSEL*
      U153-8 U162-13 U64-20
9      DMAI*
    
```



	U176-2	J12-28	J15-28	J14-28	J13-28	U163-5	P14-2
	U153-9						
10	ROMSEL*						
	U153-10	U165-9	U176-3				
11	U153-11	U79-6					
12	CH80*						
	U139-9	U153-12	U87-14				
13	C7M*						
	U119-3	U153-13	U119-4				
14	+5V						
U154	5-9	LS08	14	F8			
1	U139-8	U154-1					
2	Q3						
	J14-37	U85-3	J15-37	J13-37	J12-37	U117-12	R93-1
	U154-2						
3	U173-3	U154-3					
4	V4						
	U87-19	U1-2	U154-4	U120-12	U1-11		
5	V3						
	U120-13	U1-15	U154-5	U1-6			
6	VBL						
	U175-15	U87-5	U97-19	U97-18	U121-19	U154-6	
7	GND						
8	V2*V5						
	U121-8	U154-8	U5-3				
9	V5						
	U120-11	U154-9					
10	V2						
	U120-14	U154-10	U87-22				
11	U126-14	U152-1	U154-11				
12	RBL						
	U121-13	U154-12					
13	SCRN						
	U154-13	U73-7	P19-4				
14	+5V						
U155	3-4-8	LS02	14	D9			
1	U155-1	U164-1					
2	NMI*						
	U139-6	U65-6	U155-2				
3	AIISW*						
	U165-6	U71-12	U97-8	U178-1	U178-13	U177-15	U87-3
	U155-3						
4	U144-9	U131-13	U155-4				
5	PA9						
	U64-22	U155-5	U65-18	U67-6			
6	PA10						
	U64-19	U155-6	U65-19	U67-10			
7	GND						
8	U155-8	U148-6					
9	H1						



10 U114-13 U2-11 U155-9
 IOIRQ
 U73-40 U155-10
 11 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6
 12 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
 U71-1 U70-7 U6-5 U3-6 U174-7
 13 U147-11 U155-13
 14 +5V

U158 3-4-9 LS32 14 J8
 1 C07X*
 U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13
 2 IR*/W
 U91-3 U68-11 U158-2 U163-10
 3 CLKRD
 U72-2 U158-3
 4 PPA8
 U158-4 U152-4 U158-5 U65-17 U64-23 U132-5
 5 PPA8
 U158-4 U152-4 U158-5 U65-17 U64-23 U132-5
 6 PA8
 U11-2 U174-17 U67-3 U158-6
 7 GND
 8 U158-8 U152-5
 9 PRIMSTK
 U73-4 P19-7 U158-9
 10 ABK4
 U11-15 U174-1 U10-15 U158-10
 11 U158-11 U153-1 U180-4
 12 U158-12 U126-15
 13 RFSH
 U158-13 U11-16 U126-2
 14 +5V

U160 3-7 LS125 14 J9
 1 ENSEL
 U161-4 U160-1 U177-11 U161-1
 2 PSW3/SCO
 U169-16 U161-3 U160-2
 3 SCO
 U160-3 U73-18 U160-5 U101-1
 4 PDL2
 U160-4 U105-2 U75-10
 5 SCO
 U160-3 U73-18 U160-5 U101-1
 6 R86-1 U160-6 U161-2
 7 GND
 8 SER
 U73-19 U160-8 U161-9



9 PX1/SER
 U105-12 U161-8 U160-9 U169-18
 10 ENSIO
 U160-10 U177-12 U161-10
 11 R/W*
 J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
 U176-6 U165-11 U160-11
 12 IR/W*
 U98-28 U97-22 U112-14 U73-22 U163-11 U65-34 U160-12
 13 DMA1
 U131-11 U163-6 U131-10 U131-9 U160-13
 14 +5V

U161 6-7 LS126 14 J10

1 ENSEL
 U161-4 U160-1 U177-11 U161-1
 2 R86-1 U160-6 U161-2
 3 PSW3/SCO
 U169-16 U161-3 U160-2
 4 ENSEL
 U161-4 U160-1 U177-11 U161-1
 5 AXCO
 U161-5 U75-12 U105-1
 6 Y1/XCOUF
 U161-6 U169-17 U105-11
 7 GND
 8 PX1/SER
 U105-12 U161-8 U160-9 U169-18
 9 SER
 U73-19 U160-8 U161-9
 10 ENSIO
 U160-10 U177-12 U161-10
 11 U161-11 U92-5 R67-2
 12 U93-12 U161-12 U93-11
 13 U161-13 U93-10
 14 +5V

U162 3-4-5-8-9 LS04 14 H8

1 DTRDY*
 U107-16 U162-1
 2 U96-10 U162-2 U139-2
 3 GPH1
 U162-3 U76-4 U141-6 U74-4
 4 GPH2
 U150-2 U77-6 U162-4
 5 BTO
 U88-3 U85-18 U162-5 U79-13 U83-4 U83-1
 6 BTO*
 U83-3 U162-6
 7 GND
 8 UUTSUNK*
 J19-1 U118-9 U162-8 U120-9



```

9          U162-9 U120-15
10         RESET*
          U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
          U173-1
11         UUTUPRST
          R92-2 U162-11 J21-4 U164-3
12         TROMSEL
          U162-12 U144-13
13         TROMSEL*
          U153-8 U162-13 U64-20
14         +5V

U163 3-5-6 LS04 14 K9
1         DEXT*
          U167-18 U94-10 U178-2 U163-1
2         PINT*
          U167-19 U163-2 U178-5
3         TSADB*
          J15-22 J12-22 J14-22 J13-22 U163-3 P14-3
4         U70-15 U163-4 U67-15 U63-19 U63-1
5         DMAI*
          U176-2 J12-28 J15-28 J14-28 J13-28 U163-5 P14-2
          U153-9
6         DMA1
          U131-11 U163-6 U131-10 U131-9 U160-13
7         GND
8         NTSCB
          P3-8 U163-8
9         NTSCB*
          U163-9 U90-9
10        IR*/W
          U91-3 U68-11 U158-2 U163-10
11        IR/W*
          U98-28 U97-22 U112-14 U73-22 U163-11 U65-34 U160-12
12        DWRREQ
          U166-14 U163-12
13        U163-13 U94-12 U92-4
14        +5V

U164 2-6-8 LS05 14 H9
1         U155-1 U164-1
2         IORESET*
          J12-31 J13-31 J14-31 J15-31 P14-6 U164-2 U164-4
          U96-4 U94-15
3         UUTUPRST
          R92-2 U162-11 J21-4 U164-3
4         IORESET*
          J12-31 J13-31 J14-31 J15-31 P14-6 U164-2 U164-4
          U96-4 U94-15
5         APPLEII*
          J7-5 P11-5 U106-12 U164-5
6         U164-6 C17-2 X5-2 C75-2
    
```




7 GND
 8 RDY
 J15-21 U164-8 J14-21 J13-21 J12-21 P14-4 U65-2
 9 PRDY
 U164-9 U174-13
 10 DTIM
 R31-1 U164-10 C7-2 U96-2 U96-6
 11 MOTEN*
 U164-11 U94-9
 12 MOTON*
 U179-1 U92-15 U178-15 R65-2 U164-12 U92-16
 13 MOTON
 U164-13 U96-5
 14 +5V

U165 4 LS133 16 J7
 1 C-FXXX
 U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1
 2 S5A
 R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
 U173-10
 3 S5A
 R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
 U173-10
 4 S5A
 R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
 U173-10
 5 S5A
 R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
 U173-10
 6 AIISW*
 U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
 U155-3
 7 FFCX*
 U165-7 U112-4
 8 GND
 9 ROMSEL*
 U153-10 U165-9 U176-3
 10 ROMSEL1
 U165-10 U73-2 P19-9
 11 R/W*
 J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
 U176-6 U165-11 U160-11
 12 FSPACE*
 U180-1 U176-1 U165-12 U123-12 U148-8
 13 INH*
 J15-32 J14-32 J13-32 J12-32 U176-4 U165-13 P14-5
 14 A12
 J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
 U70-4 U71-2 U174-6
 15 A13
 J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15



16 U71-1 U70-7 U6-5 U3-6 U174-7
+5V

U166 6 FILTER 20 N13

1 GNDF
2 ENBL3.E*
J1-21 U166-2
3 WRDATA
J6-18 U166-3 J1-13
4 RDDATA
J6-16 U166-4 J1-16
5 ENBL1.E*
J1-14 U166-5
6 ENBL1.I*
J6-14 U166-6
7 WRREQ
J6-10 U166-7 J1-10
8 DPH2
J6-6 U166-8 J1-6
9 DPHO
J6-2 U166-9 J1-2
10 GNDF
11 GNDF
12 VA1
U166-12 U175-4 U94-4
13 VC1
U166-13 U175-2 U94-6
14 DWRREQ
U166-14 U163-12
15 PENBL1.I*
U166-15 U178-4
16 PENBLE*
U166-16 U178-7
17 DRDATA
U166-17 U93-13
18 DWRDATA
U166-18 U92-19 U93-2
19 PENBL3.E*
U166-19 U179-7
20 GNDF

U167 6 FILTER 20 M13

1 GNDF
2 EXT*
J1-26 U167-2
3 INT*
J6-26 U167-3
4 WRPROT
J6-20 U167-4 J1-20
5 AII*
U167-5 J1-23 J6-23
6 ENBL2.E*



7 J1-22 U167-6
 SIDE2/1
 J6-24 U167-7 J1-24
 8 DPH3
 J6-8 U167-8 J1-8
 9 DPH1
 J6-4 U167-9 J1-4
 10 GNDF
 11 GNDF
 12 VB1
 U167-12 U175-3 U94-5
 13 PDPH3
 U167-13 U94-7
 14 PSIDE
 U167-14 U177-7
 15 PENBL2,E*
 U167-15 U179-6
 16 PAII
 U178-12 U167-16
 17 DWRPROTT
 U167-17 U91-11
 18 DEXT*
 U167-18 U94-10 U178-2 U163-1
 19 PINT*
 U167-19 U163-2 U178-5
 20 GNDF

U169 7 FILTER 20 N10
 1 GNDF
 2 SW1/MGNSW
 J3-5 U169-2
 3 X1/SER
 J3-4 U169-3
 4 Y1/XCO
 J3-8 U169-4
 5 SW3/SCO
 J3-9 U169-5
 6 YO
 U169-6 J2-8
 7 SW2
 U169-7 J2-5
 8 SWO
 U169-8 J2-9
 9 XO
 U169-9 J2-4
 10 GNDF
 11 GNDF
 12 XOUF
 U105-15 U169-12
 13 SWOUF
 U101-4 U169-13
 14 SW2UF

15.100



15 U101-2 U169-14
 YOUF
 U105-13 U169-15
 16 PSW3/SCO
 U169-16 U161-3 U160-2
 17 Y1/XCOUF
 U161-6 U169-17 U105-11
 18 PX1/SER
 U105-12 U161-8 U160-9 U169-18
 19 SW1/MGNSWUF
 R85-2 U169-19 U73-39 U101-3
 20 GNDF

U171 8 FILTER 20 K13

1 GNDF
 2 KX6
 J7-26 U171-2
 3 KX4
 J7-21 U171-3
 4 KX3
 J7-20 U171-4
 5 KX5
 J7-19 U171-5
 6 KX1
 J7-18 U171-6
 7 KX7
 J7-17 U171-7
 8 KX2
 J7-16 U171-8
 9 KX0
 J7-14 U171-9
 10 GNDF
 11 GNDF
 12 KXOUF
 U171-12 U107-40
 13 KX2UF
 U171-13 U107-38
 14 KX7UF
 U171-14 U107-33
 15 KX1UF
 U171-15 U107-39
 16 KX5UF
 U171-16 U107-35
 17 KX3UF
 U171-17 U107-37
 18 KX4UF
 U171-18 U107-36
 19 KX6UF
 U171-19 U107-34
 20 GNDF

U172 7 FILTER 20 N3



1 GNDF
 2 EXTSPK2
 J11-2 U172-2
 3 DCD
 J4-8 U172-3
 4 DTR
 J4-20 U172-4
 5 DSR
 J4-6 U172-5
 6 CTS
 J4-5 U172-6
 7 RTS
 J4-4 U172-7
 8 DATA IN
 J4-3 U172-8
 9 TXD
 J4-2 U172-9
 10 GNDF
 11 GNDF
 12 PTXD
 U172-12 U99-6
 13 PRXD
 U172-13 U100-1
 14 PRTS
 U172-14 U99-11
 15 PCTS
 R87-2 U172-15 U100-4
 16 PDSR
 R88-2 U172-16 U100-10
 17 PDTR
 U172-17 U99-8
 18 PDCD
 R89-2 U172-18 U100-13
 19 PEXTSPK
 U172-19 C10-2
 20 GNDF

U173 5-7 LS74 14 H7

1 RESET*
 U65-40 U162-10 U98-4 U97-34 U73-34 U75-15 U79-9
 U173-1
 2 U173-2 U136-6
 3 U173-3 U154-3
 4 S5A
 R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
 U173-10
 5 INV
 U83-11 U173-5
 6 NO CONNECTION
 7 GND
 8 U173-8 U173-12
 9 AIISPKR



```

10      R35-1 U173-9
        SSA
        R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
        U173-10
11      SPKR*
        U173-11 U77-12
12      U173-8 U173-12
13      SSA
        R106-2 U165-2 U165-3 U165-4 U165-5 U173-4 U173-13
        U173-10
14      +5V

U174 2 7643 18 341-0043 C10
1      ABK4
        U11-15 U174-1 U10-15 U158-10
2      SELIM
        U180-15 U174-2 U150-3 U73-9 P19-2
3      SYNC
        U65-7 U136-1 J15-35 U174-3 J14-35 J13-35 J12-35
4      A14
        J15-16 J14-16 J13-16 J12-16 U3-7 U147-4 U70-9
        U6-16 U174-4
5      A11
        J15-13 J14-13 J13-13 J12-13 U128-12 U6-6 U71-3
        U150-13 U74-5 U67-12 U3-5 U174-5
6      A12
        J15-14 J14-14 J13-14 J12-14 U9-12 U155-11 U165-14
        U70-4 U71-2 U174-6
7      A13
        J15-15 J14-15 J13-15 J12-15 U128-4 U155-12 U165-15
        U71-1 U70-7 U6-5 U3-6 U174-7
8      GND
9      GND
10     GND
11     NO CONNECTION
12     IND*
        U136-10 U147-5 U174-12
13     PRDY
        U164-9 U174-13
14     S399
        U174-14 U153-4
15     ZPAGE*
        U11-1 U174-15 U67-1 U131-8 U70-1
16     A15
        J15-17 J14-17 J13-17 J12-17 U3-4 U147-3 U70-12
        U174-16 U6-7
17     PA8
        U11-2 U174-17 U67-3 U158-6
18     +5V

U175 2 7643 18 341-0055 F9
1      RDHIRES
    
```



2 U175-1 U11-17 U3-2 U6-3 U126-9
 VC1
 U166-13 U175-2 U94-6
 3 VBI
 U167-12 U175-3 U94-5
 4 VA1
 U166-12 U175-4 U94-4
 5 VA
 U121-3 U175-5 U116-11 U82-5 U81-5
 6 VB
 U121-2 U175-6 U118-14 U82-6 U81-6
 7 VC
 U121-1 U175-7 U118-13 U83-15 U83-14 U82-7 U81-7
 8 GND
 9 GND
 10 GND
 11 ENHREG*
 U175-11 U78-1
 12 MUX3
 U9-13 U175-12
 13 MUX2
 U9-3 U175-13
 14 MUX1
 U101-12 U175-14 U5-13
 15 VBL
 U175-15 U87-5 U97-19 U97-18 U121-19 U154-6
 16 WE2114*
 U82-10 U175-16 U81-10 U144-6
 17 SCR
 U177-9 U175-17
 18 +5V

U176 4 7643 18 341-0045 F5
 1 FSPACE*
 U180-1 U176-1 U165-12 U123-12 U148-8
 2 DMAI*
 U176-2 J12-28 J15-28 J14-28 J13-28 U163-5 P14-2
 U153-9
 3 ROMSEL*
 U153-10 U165-9 U176-3
 4 INH*
 J15-32 J14-32 J13-32 J12-32 U176-4 U165-13 P14-5
 5 U74-10 U176-5
 6 R/W*
 J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
 U176-6 U165-11 U160-11
 7 C6XX*
 U176-7 U74-9
 8 GND
 9 GND
 10 GND
 11 RAMEN



12 U180-7 U176-11
 NO CONNECTION
 13 EN8304
 U176-13 U68-9
 14 EN257
 U176-14 U66-15 U69-15
 15 CXXX
 U176-15 U74-6 U150-1 U147-8
 16 C7XX*
 U176-16 U74-7
 17 PH2M
 P19-10 U141-5 U65-39 U176-17
 18 +5V

U177 6 9334 16 L12

1 A1
 J15-3 J14-3 J13-3 J12-3 U2-10 U97-37 U98-14
 U101-10 U177-1 U94-1 U73-37 U75-1 U63-16
 2 A2
 J15-4 J14-4 J13-4 J12-4 U5-6 U101-9 U97-36
 U177-2 U94-2 U75-2 U73-36 U63-14
 3 A3
 J15-5 J14-5 J13-5 J12-5 U5-10 U109-1 U111-1
 U97-35 U177-3 U94-3 U75-3 U73-35 U63-12
 4 EXT0
 U179-2 U177-4
 5 EXT1
 U179-3 U177-5
 6 INTON
 U177-6 U178-3
 7 PSIDE
 U167-14 U177-7
 8 GND
 9 SCR
 U177-9 U175-17
 10 ENCWRT
 U177-10 U144-1
 11 ENSEL
 U161-4 U160-1 U177-11 U161-1
 12 ENSIO
 U160-10 U177-12 U161-10
 13 A0
 J15-2 J14-2 J13-2 J12-2 U2-6 U97-38 U98-13
 U101-11 U94-13 U73-38 U75-13 U63-18 U177-13
 14 DEVSEL5*
 U177-14 U76-10
 15 AIISW*
 U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
 U155-3
 16 +5V

U178 3-6 LS257 16 L11



```

1      AIISW*
      U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
      U155-3
2      DEXT*
      U167-18 U94-10 U178-2 U163-1
3      INTON
      U177-6 U178-3
4      PENBL1,I*
      U166-15 U178-4
5      PINT*
      U167-19 U163-2 U178-5
6      NENBLE*
      U179-5 U178-6
7      PENBLE*
      U166-16 U178-7
8      GND
9      NO CONNECTION
10     NO CONNECTION
11     NO CONNECTION
12     PAII
      U178-12 U167-16
13     AIISW*
      U165-6 U71-12 U97-8 U178-1 U178-13 U177-15 U87-3
      U155-3
14     GND
15     MOTON*
      U179-1 U92-15 U178-15 R65-2 U164-12 U92-16
16     +5V

U179 6-8 LS139 16 J11
1      MOTON*
      U179-1 U92-15 U178-15 R65-2 U164-12 U92-16
2      EXTO
      U179-2 U177-4
3      EXT1
      U179-3 U177-5
4      NO CONNECTION
5      NENBLE*
      U179-5 U178-6
6      PENBL2,E*
      U167-15 U179-6
7      PENBL3,E*
      U166-19 U179-7
8      GND
9      NO CONNECTION
10     NO CONNECTION
11     X6-2 U179-11
12     NO CONNECTION
13     CONTROL*
      J7-11 U179-13 U107-28 U109-10 P11-2
14     RESETLK*
      U179-14 U73-6 P19-5 U139-5
    
```



```

,      KRESET*
      U139-12 R80-1 C71-1 U179-15 J7-15
16     +5V

, 9 7643 18 341-0046 F7
1      FSPACE*
      U180-1 U176-1 U165-12 U123-12 U148-8
2      C-FXXX
      U71-15 U180-2 U71-14 U71-13 U147-6 U147-9 U165-1
3      IOSTOPD*
      U123-9 U112-15 U180-3
4      U158-11 U153-1 U180-4
5      CIM
      U132-2 U135-1 U119-10 U123-11 U180-5 U98-27 U80-11
      U84-11
6      CO7X*
      U180-6 U77-7 U112-13 U158-1 U72-1 U150-5 U148-13
7      RAMEN
      U180-7 U176-11
8      GND
9      GND
10     GND
11     NO CONNECTION
12     WRAMEN
      U180-12 U150-11 .
13     PHASEN
      U140-12 U180-13
14     PCS6522
      R68-1 U180-14
15     SEL1M
      U180-15 U174-2 U150-3 U73-9 P19-2
16     RWPR
      U73-5 P19-6 U180-16
17     R/W*
      J15-18 J14-18 J13-18 J12-18 U3-3 U136-9 U180-17
      U176-6 U165-11 U160-11
18     +5V

```

```

J181 7 556 14 A7
1      C69-2 R75-2 U181-1 U181-2
2      C69-2 R75-2 U181-1 U181-2
3      NO CONNECTION
4      +5V
5      U181-5 U181-10
6      CO4X*
      U77-11 U181-6
7      GND
8      R77-2 U181-8 U181-12 C70-1
9      U181-9 R78-1
10     U181-5 U181-10
11     NO CONNECTION
12     R77-2 U181-8 U181-12 C70-1

```



```

13          R76-2 R77-1 U181-13
14          +5V

X1 4 SCHOTTKY 2 DIODE A2
1          R9-1 T1-1 X1-1 U105-10
2          U72-24 Q1-1 X1-2

X2 3 1N4148 2 DIODE L10
1          U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1
          X3-1 C75-1
2          ANYKEY
          U107-4 R40-1 X2-2 U106-11 U109-3

X3 8 1N4148 2 DIODE L11
1          U96-13 U96-12 U96-8 R40-2 X2-1 C18-1 C17-1
          X3-1 C75-1
2          KAPPLEII*
          U106-9 X3-2 U111-6

X4 8 1N4148 2 DIODE A5
1          +5V
2          UPRST*
          C20-1 U113-6 U113-2 R44-2 X4-2 X6-1

X5 8 1N4148 2 DIODE M7
1          GND
2          U164-6 C17-2 X5-2 C75-2

X6 8 1N4148 2 DIODE A6
1          UPRST*
          C20-1 U113-6 U113-2 R44-2 X4-2 X6-1
2          X6-2 U179-11

X7 1 LED 2 LED M4
1          +5V
2          X7-2 R81-1

Y1 9 14MHZ 2 XTAL1 A12
1          R52-1 Q10-3 Y1-1
2
          Y1-2 Q11-2 R49-2

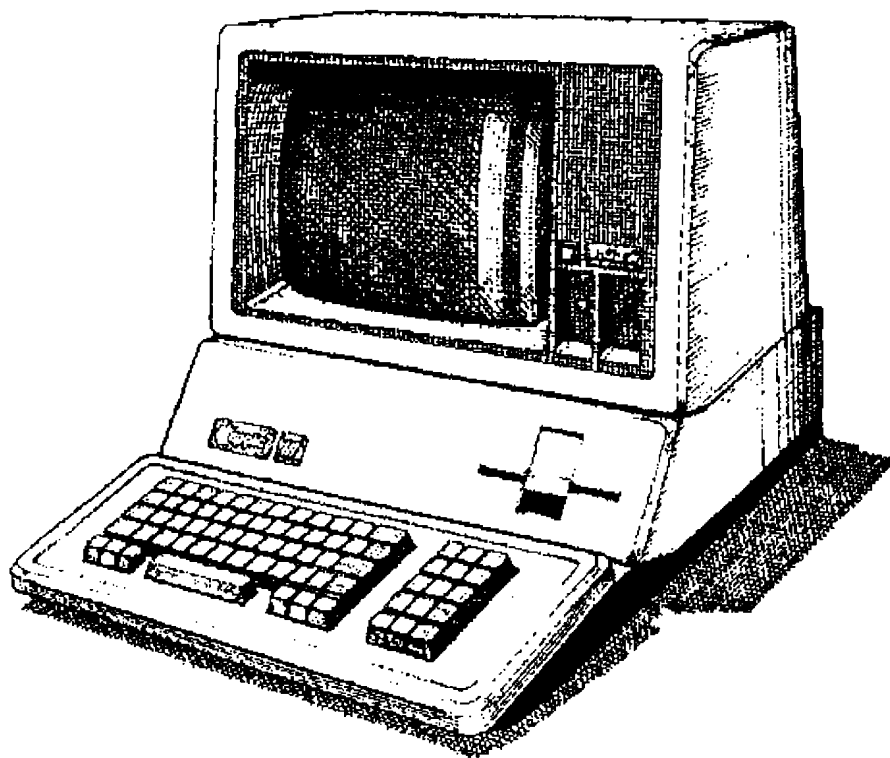
Y2 4 32KHZ 2 XTAL2 A3
1          Y2-1 U72-10 C3-3
2          C4-1 U72-11 Y2-2
    
```

END OF DATA



Apple /// Computer Information

Apple /// Service Reference Manual



Section II of II • Servicing Information

Chapter 16 • Module Replacement Procedures

Written by Apple Computer • 1982



APPLE III

MODULE REPLACEMENT PROCEDURES

TABLE OF CONTENTS

PAGE

I.	Peripheral Logic Access Cover Removal	2-35
II.	Peripheral Card Removal/Installation	2-37
III.	Keyboard Replacement	2-39
IV.	Analog Board Replacement — Disk Assembly	2-42
V.	Disk Mechanical Assembly Replacement—Disk Assembly	2-46
VI.	Power Supply Replacement	2-48
VII.	Logic Assembly Removal	2-51
VIII.	Logic Assembly Replacement	2-54
	A) Memory Board Removal	
	B) Memory Board Installation	
	C) Encoder Board Removal	
	D) Encoder Board Installation	
	E) Main Logic Board Replacement	

 **apple computer inc.**

16.2



I. PERIPHERAL LOGIC ACCESS COVER REMOVAL

1. Power down the Apple ///. Disconnect the AC power cord from the source and then from the power supply receptacle of the Apple ///.
2. Disconnect all external cables. Refer to Figure 1.1.
3. Lift up the front edge of the Apple and tilt it up 90 degrees so that it rests on the back side of the casting.
4. Locate the 1/4 turn locking screw on each side of the Apple and, with a flat blade screw driver, turn each one 1/4 turn counterclockwise to loosen. Do not attempt to remove these screws as they are self capturing and will not normally come out. Refer to Figure 1.2 item A.
5. Lower the Apple /// to the operating position and with a hand on each side of the access cover lift up and pull forward to remove. Refer to Figure 1.3.
6. To replace the cover reverse the procedure as outlined in steps 1 through 5.

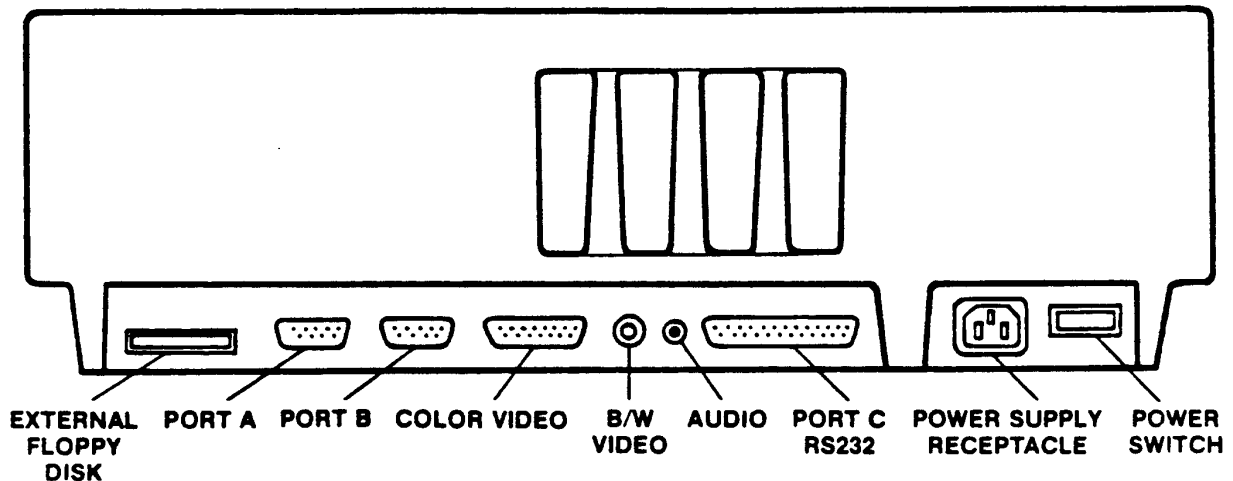


FIGURE 1.1

Apple Computer Inc.

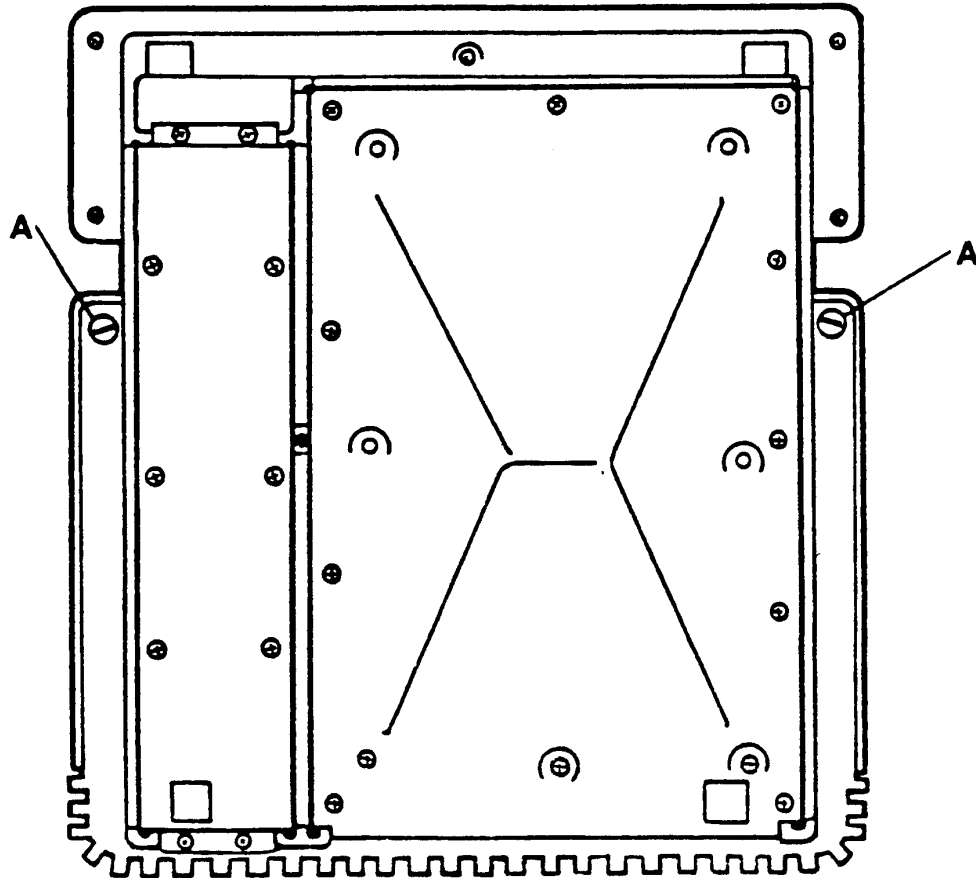


FIGURE 1.2

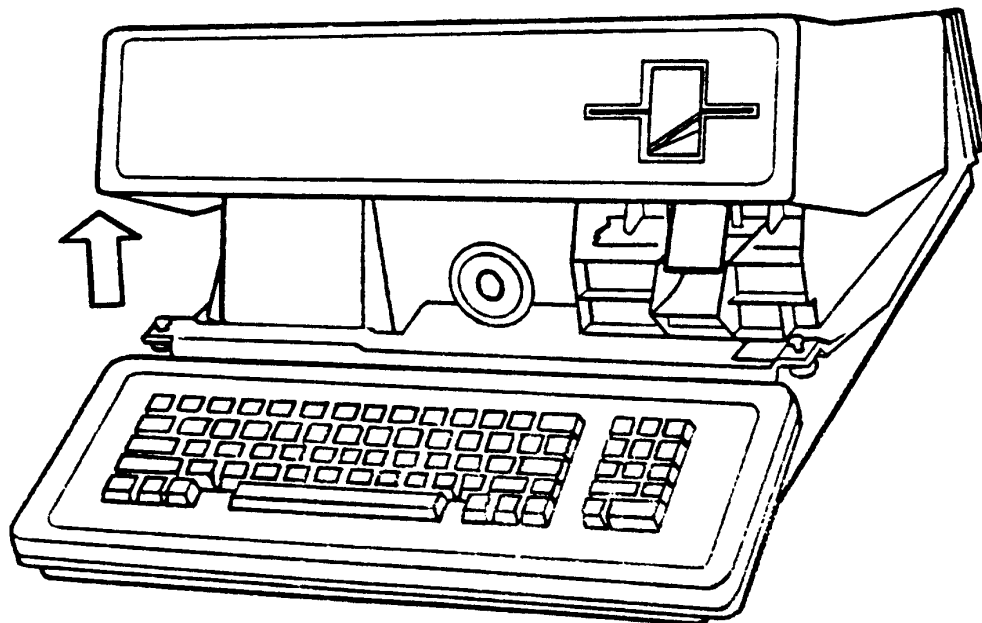


FIGURE 1.3

16.4



II. PERIPHERAL CARD REMOVAL/INSTALLATION

1. Power down the Apple ///. Disconnect the AC power cord from the source and then the power supply receptacle of the Apple ///.
2. Remove the peripheral logic access cover. Refer to Procedure I.
3. Locate the desired peripheral card. Refer to Figure 2.1 for slot number assignment. Disconnect all cords or cables connected to the peripheral card or cards to be removed.
4. Grasping the card firmly with both hands (using thumbs and forefingers) gently pull straight up on the card to free it from the connector and guide slots. Refer to Figure 2.2.

NOTE: If the card is too firmly captured to allow removal using just the fingers, a metal hook in the pilot hole near the top rear of the card may be used to gain a better grip on the card. Be careful not to tilt or rotate the card, or damage to the card and/or connector may occur.

CAUTION: Never remove or install any card or device with the power on or catastrophic shorting of signal to power supplies may occur.

5. To replace a peripheral card reverse the procedure as outlined in steps 1 through 4.
6. To install a new peripheral card, remove the RFI shield card (dummy card) from the desired slot and follow the detailed procedure enclosed with the new peripheral card.

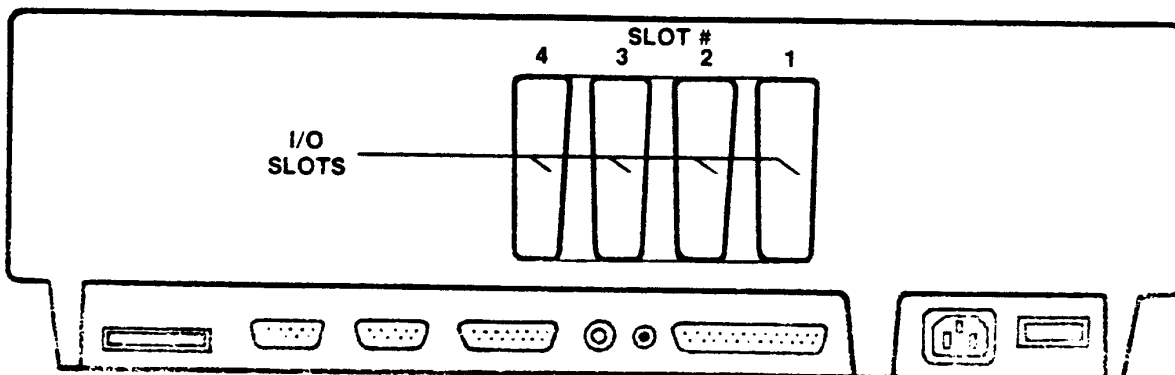


FIGURE 2.1

16.5

 apple computer inc.

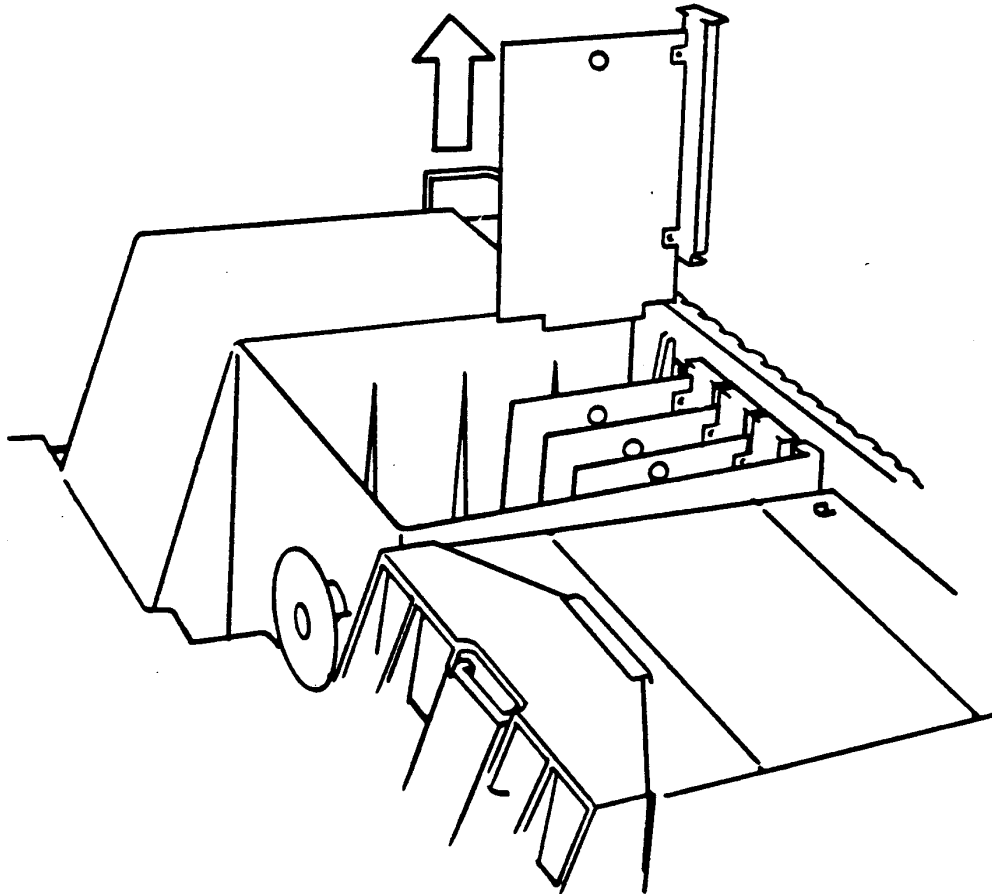


FIGURE 2.2

16.6



III. KEYBOARD REPLACEMENT

1. Power down the Apple ///. Disconnect the AC power cord from the source and then from the power supply receptacle of the Apple ///.
2. Place the Apple on its right side with the bottom facing you.
3. Locate, remove and retain the five (5) keyboard cover mounting screws located two each on the right and left ends and one in the front center. Refer to Figure 3.1 item A.
4. Remove the keyboard cover.
5. Place the Apple back into its normal operating position.
6. Locate and remove the two (2) retaining screws on the left end of the keyboard. Refer to Figure 3.2 item A. Loosen the right two (2) retaining screws. Refer to Figure 3.2 item B.
7. Remove the keyboard by lifting the left end and sliding the right end from under the loosened screws. Refer to Figure 3.3.
8. Disconnect the keyboard cable, located on the middle rear edge of the exposed keyboard. Refer to Figure 3.2 item C. Do not pull on the cable. Disconnect by using a screwdriver to push on the tab on the cable connector. Refer to Figure 3.2 item D.
9. Install the replacement keyboard by reversing the steps as outlined in 1 through 8 above. Observe that the keyboard cable makes a tight turn where it wraps to the underside of the keyboard. This is necessary if the keyboard cover is to fit properly.

Caution: Exercise care when tightening the five (5) keyboard cover screws to keep from stripping the threads in the cover.

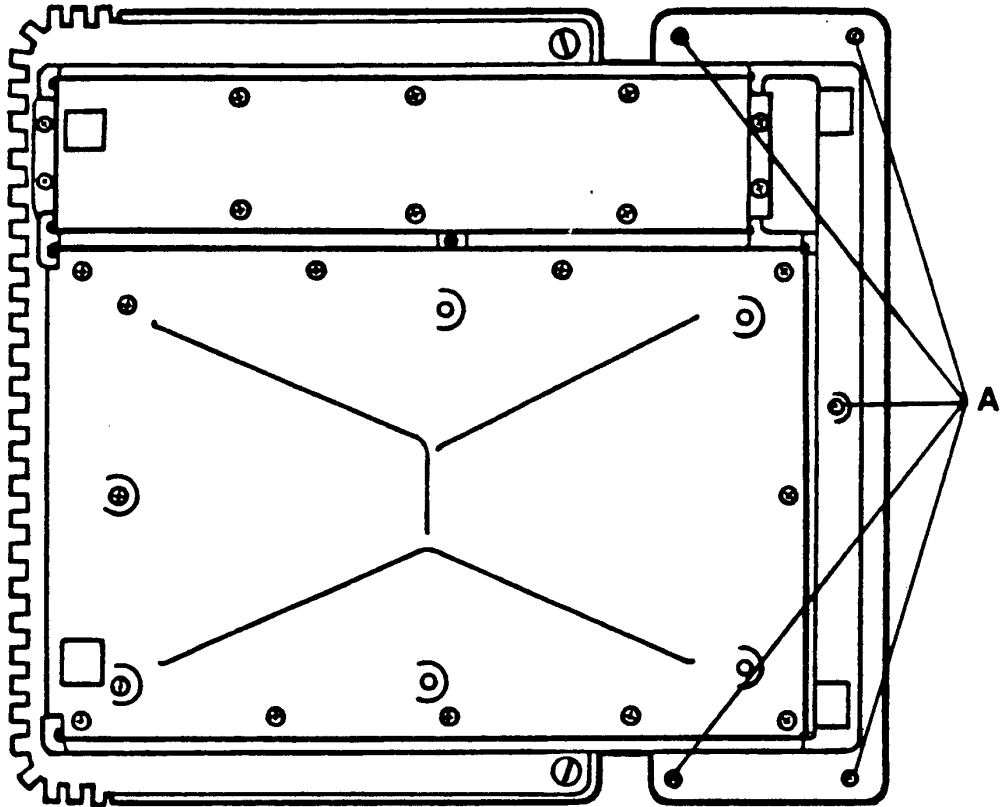
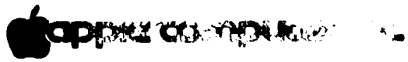


FIGURE 3.1

16.8

 apple computer inc.

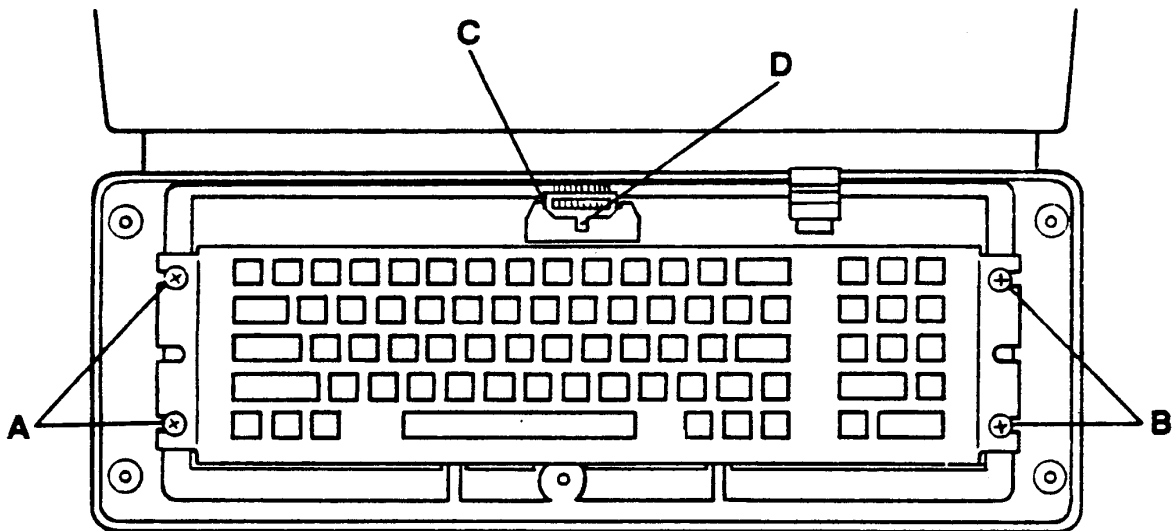


FIGURE 3.2

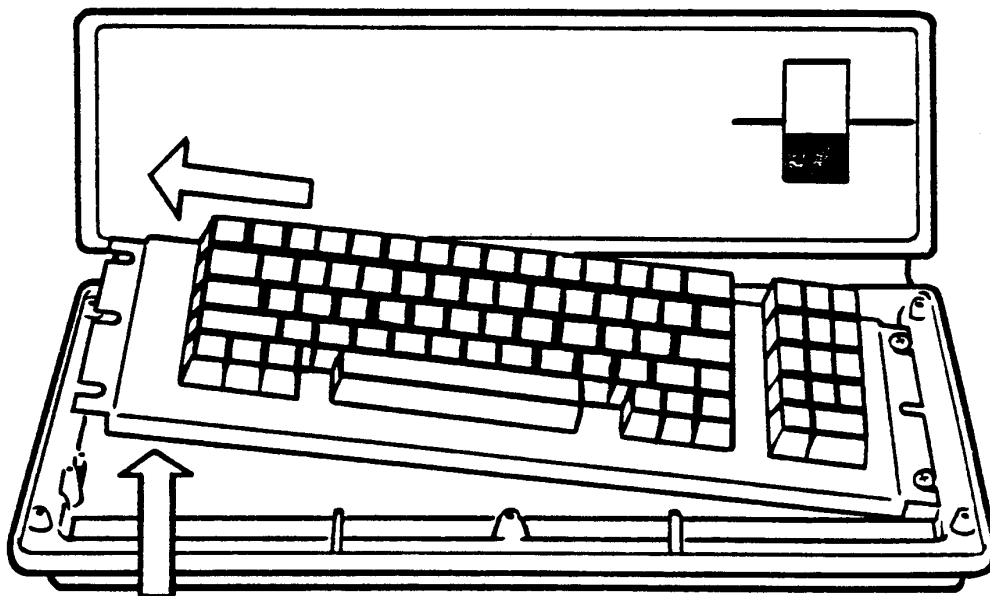


FIGURE 3.3

16.9



IV. ANALOG BOARD REPLACEMENT - DISK ASSEMBLY*

1. Power down the Apple ///. Disconnect the AC power cord from the source and then from the power supply receptacle of the Apple ///.
2. Remove the peripheral logic access cover. Refer to Procedure I.
3. Locate the two Tinnerman retaining clips which hold down the Disk Assembly shield. Refer to Figure 4.1 item A.
4. Using the blade of a screwdriver, slide the clips forward until the enlarged slots of the clips are around the mounting posts.
5. Remove and retain the clips.
6. Remove and retain the Disk Assembly shield by flexing the side out (Figure 4.1 item B) and lifting up on the shield. NOTE: The shield is only retained by the spring tension of the sides and four dimples which fit into depressions of the disk casting.
7. Disconnect the disk ribbon cable by pushing on the center tab of the plug with a small screwdriver. Do not pull it out by the cable. Refer to Figure 4.2 item A. Disconnect the read/write head cable. Refer to Figure 4.2 item B. Do not attempt to remove motor control cable yet. Refer to Figure 4.2 item D.
8. Remove and retain the two small Phillips head mounting screws which hold the Analog board at the front of the casting. Refer to Figure 4.2 item C.
9. To remove the Analog board, gently slide the left front of the board forward until it clears the guide holding the left edge. Then slide the right rear of the board to the left until it clears the guide holding the right edge. Refer to Figure 4.3. Tilt up the right rear of the board and lift clear.
10. At this time disconnect the motor control cable. Refer to Figure 4.2 item D. Note there are four nylon locking pawls which engage two holes in the board from both the top and bottom. These must be disengaged before the connector can be disconnected.
11. Install the replacement Analog board by reversing the steps as outlined in 1 through 10 above.

* The Disk Assembly is comprised of two modules, the Analog Board and the Disk Mechanical Assembly.

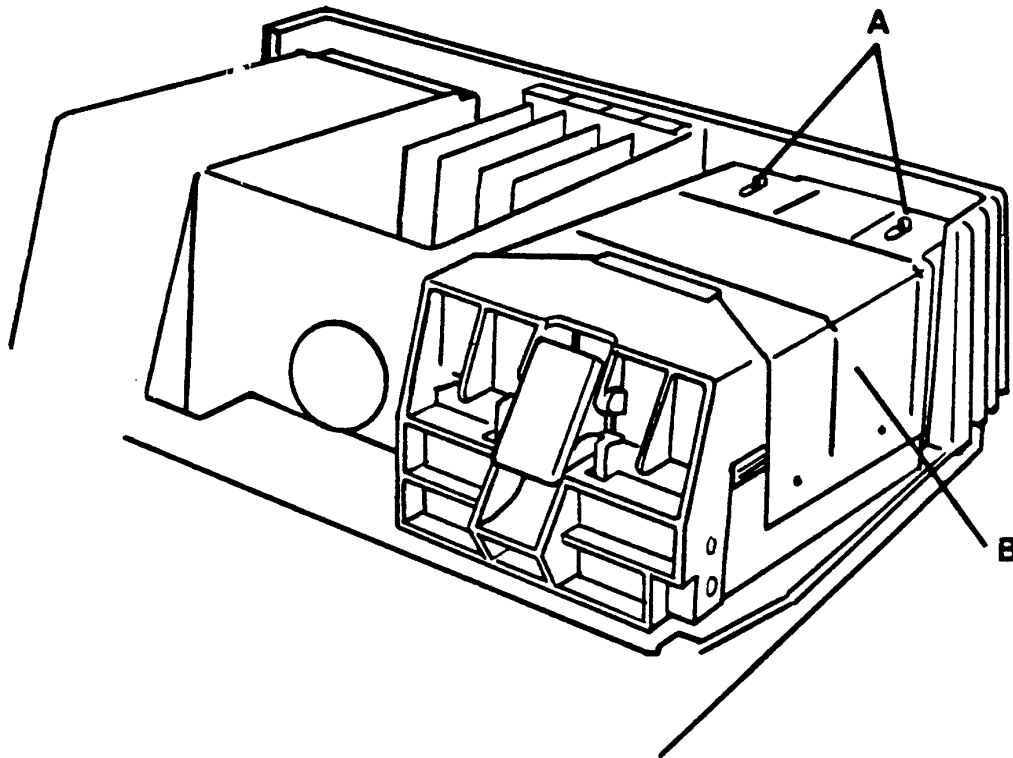


FIGURE 4.1

16.11

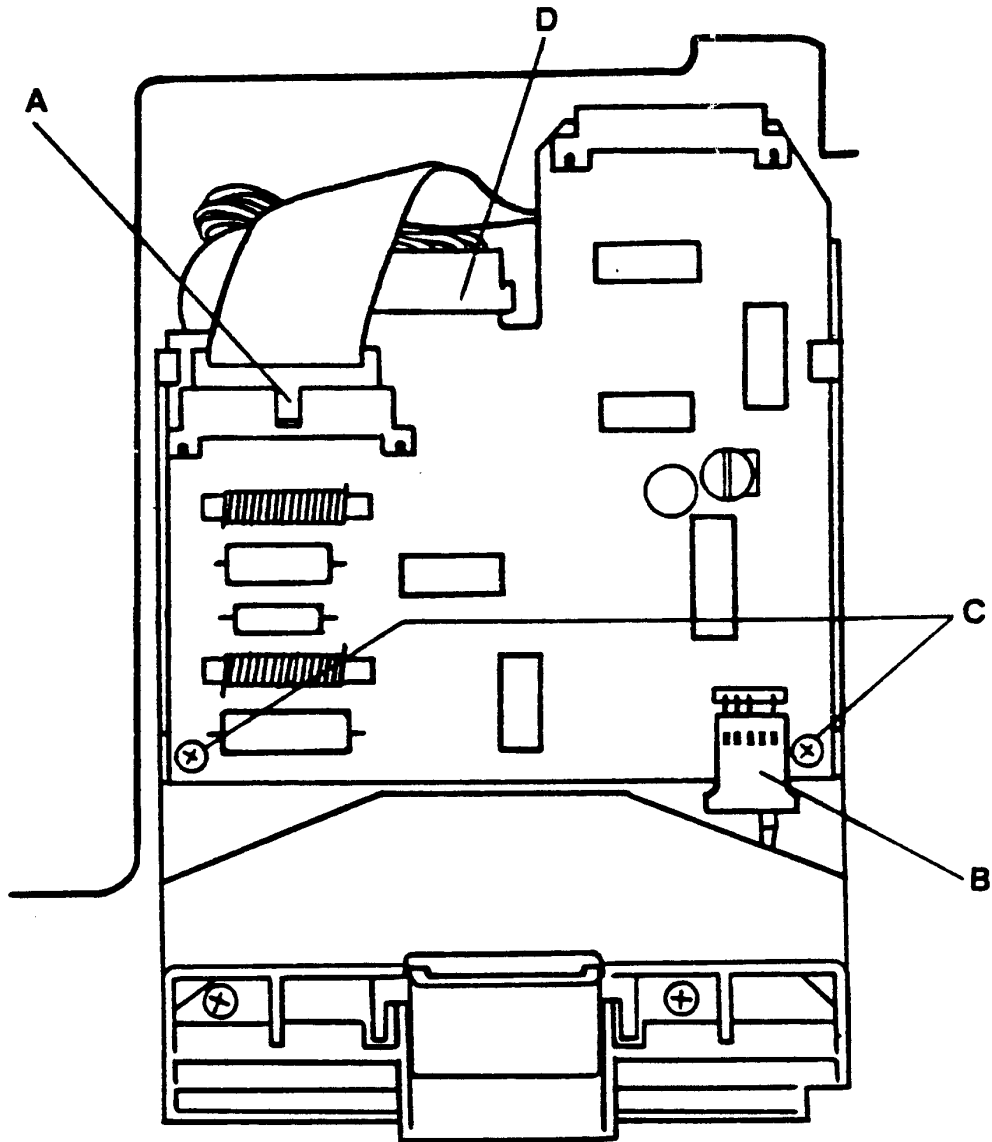


FIGURE 4.2

16.12

 apple computer inc.

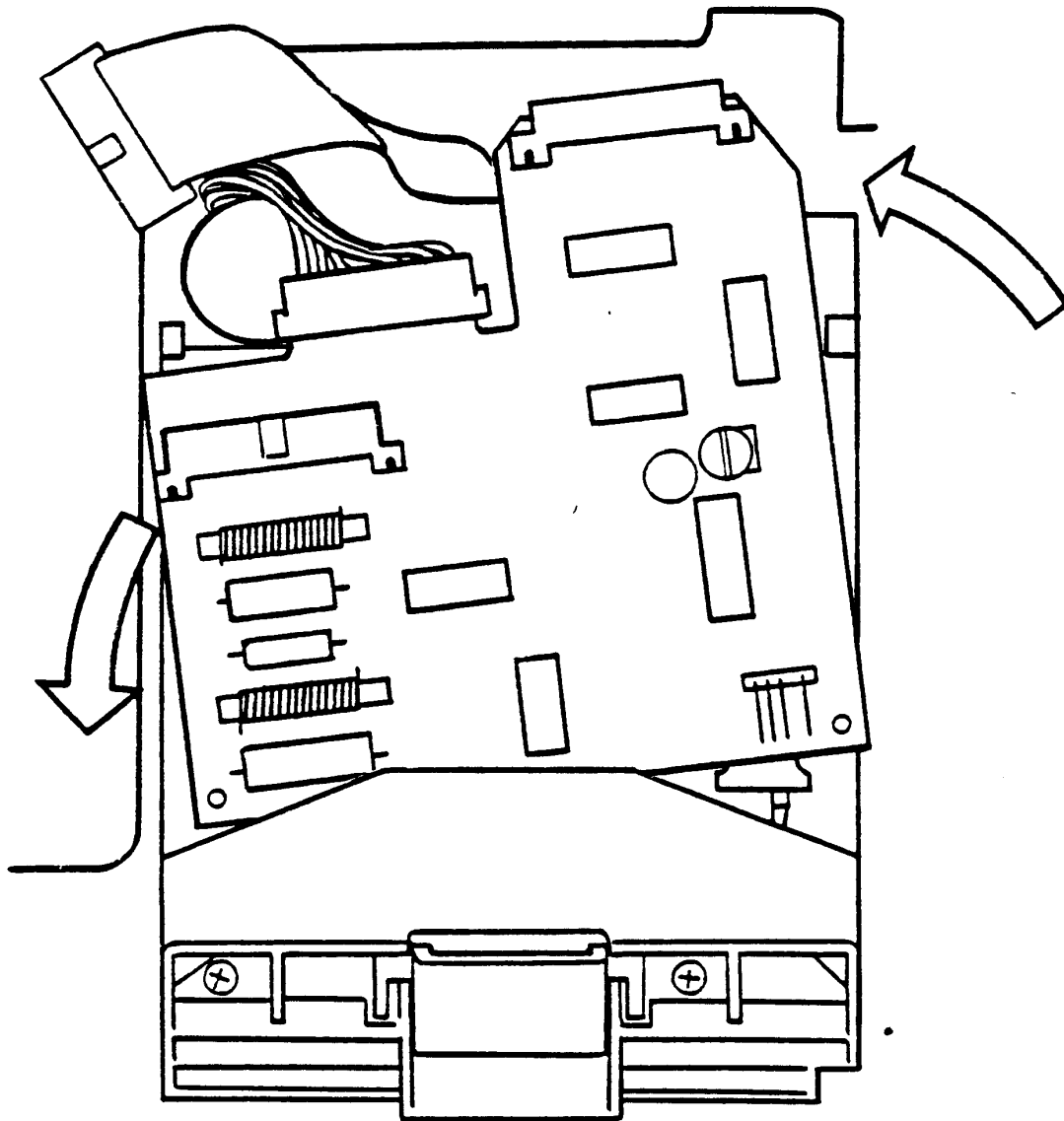


FIGURE 4.3

16.13



V. DISK MECHANICAL ASSEMBLY REPLACEMENT-DISK ASSEMBLY*

1. Power down the Apple ///. Disconnect the AC power cord from the source and then from the power supply receptacle of the Apple ///.
2. Remove the peripheral logic access cover. Refer to Procedure I.
3. Remove the disk shield. Refer to steps 3 through 6 in Procedure IV.
4. Disconnect the disk ribbon cable by pushing on the center tab of the plug with a small screwdriver. Refer to Figure 5.1 item A. Do not pull on the cable.
5. Scribe a line on the Apple /// chassis along the front (Figure 5.1 item B) and left side (Figure 5.1 item C) of the Disk Assembly bezel. This line will provide a location reference when the Disk Assembly is re-installed.
6. Locate the two Phillips head screws which mount the Disk Assembly to the Apple chassis. They can be seen by looking down through the front diskette guide and door assembly. Refer to Figure 5.1 item D.
7. Completely loosen the two mounting screws but let them remain sitting where they are.
8. Loosen (Don't Remove!) the Phillips head screw through the retaining clip which holds the lower left rear edge of the Disk casting. Refer to Figure 5.1 item E.
9. Remove the Disk Assembly by sliding it forward until it clears the retaining spring clip and then lift it from the chassis.
10. Recover the two front screws from the Disk Assembly.
11. Separate the Analog Board from the Disk Mechanical Assembly as outlined in Procedure IV step 9.
12. Install the replacement Disk Mechanical Assembly by reversing the steps as outlined in steps 1 through 11 above. Use the reference mark made in step 5 to insure proper alignment of the Disk Assembly.

* The Disk Assembly is comprised of two modules, the Analog Board and the Disk Mechanical assembly.

 apple computer inc.

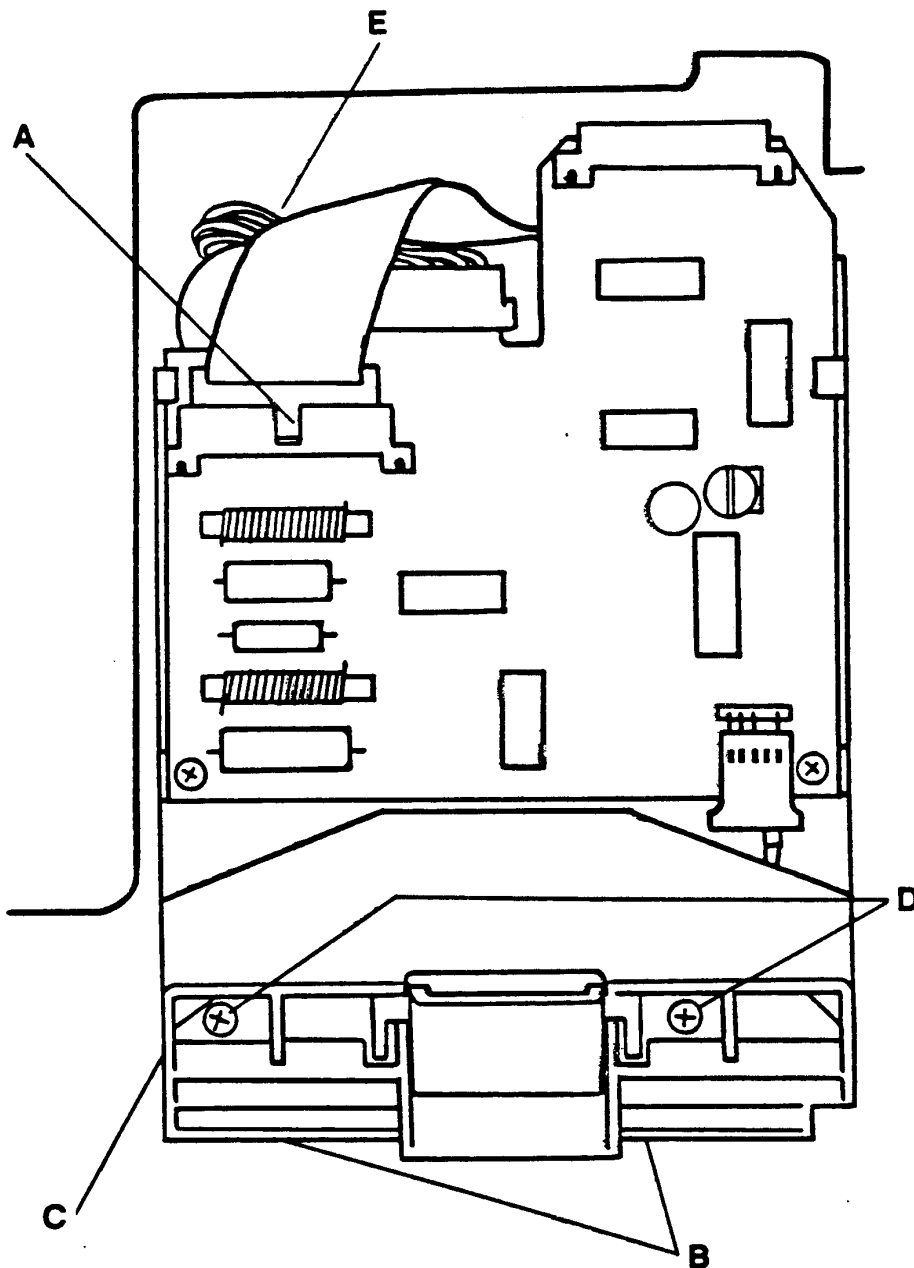


FIGURE 5.1

16.15



VI. POWER SUPPLY REPLACEMENT

1. Power down the Apple ///. Disconnect the AC power cord from the source and then remove it from the power supply receptacle of the Apple ///.
2. Disconnect all external cables.
3. Turn the Apple /// upside down with the keyboard facing you and place on a soft pad.

NOTE: You may want to place a foam block under the keyboard to keep the unit from rocking while it is upside down.

4. Loosen (Don't Remove!) the two Phillips head screws located on the rear edge of the power supply bottom cover, near the on/off switch and power supply receptacle. Refer to Figure 6.1 item A.
5. Locate and loosen the eight (8) screws securing the power supply bottom to the chassis. Refer to Figure 6.1 item B. These screws may be captured and if so should not come free of the assembly.
6. Lift up the front edge and slide the power supply forward until the rear edge clears the two rear mounting screws. Gently lift up the power supply assembly to gain access to the electrical connector. Refer to Figure 6.2.
7. Disconnect the power supply connector by pressing in the tabs while gently pulling. Refer to Figure 6.2 item A. If the leads are secured to the power supply by a wire tie, cut the tie. The power supply can now be removed.
8. Prior to replacing or re-installing the supply replace the wire tie, if one was removed.
9. When re-installing the power supply, insert the cover under the two rear most screws and lower the power supply into place.
10. Tighten all screws.

CAUTION: When re-installing the securing screws use only enough torque to rotate the screw. These screws will strip out the chassis, if excessive torque is applied. Also, be certain that the screw is not starting at an angle to avoid cross-threading. If it appears the screw is cross-threading, back it out and try again.

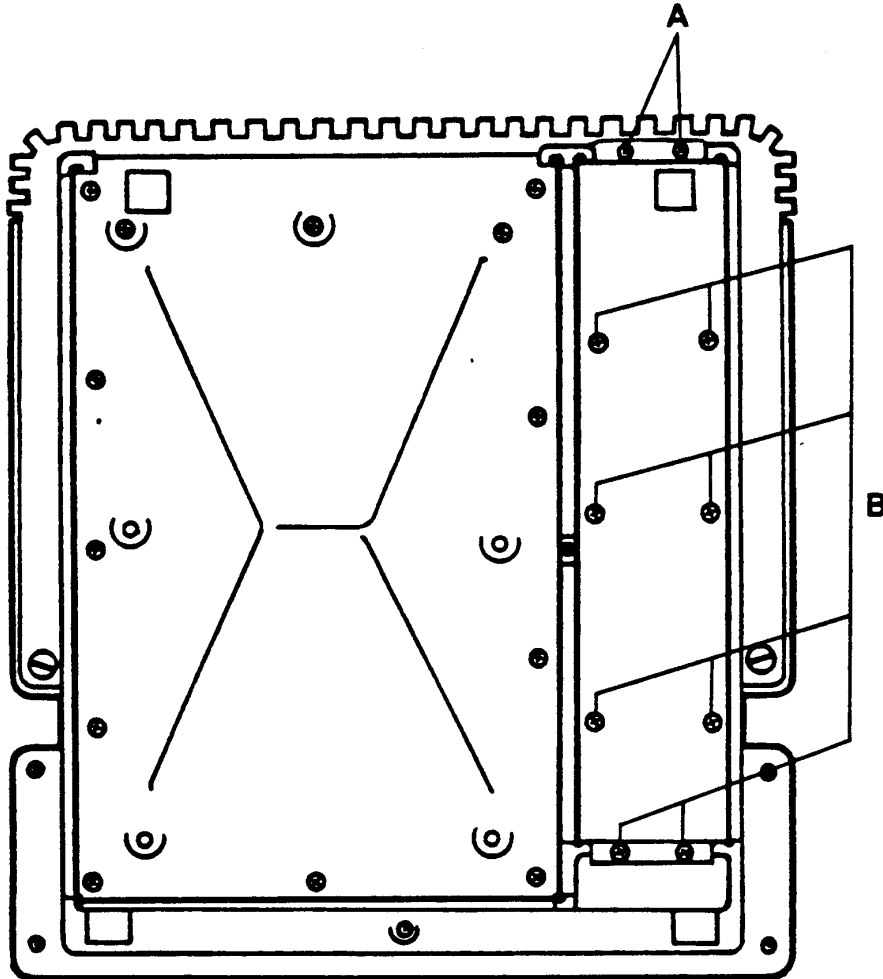


FIGURE 6.1

16.17

 apple computer inc.

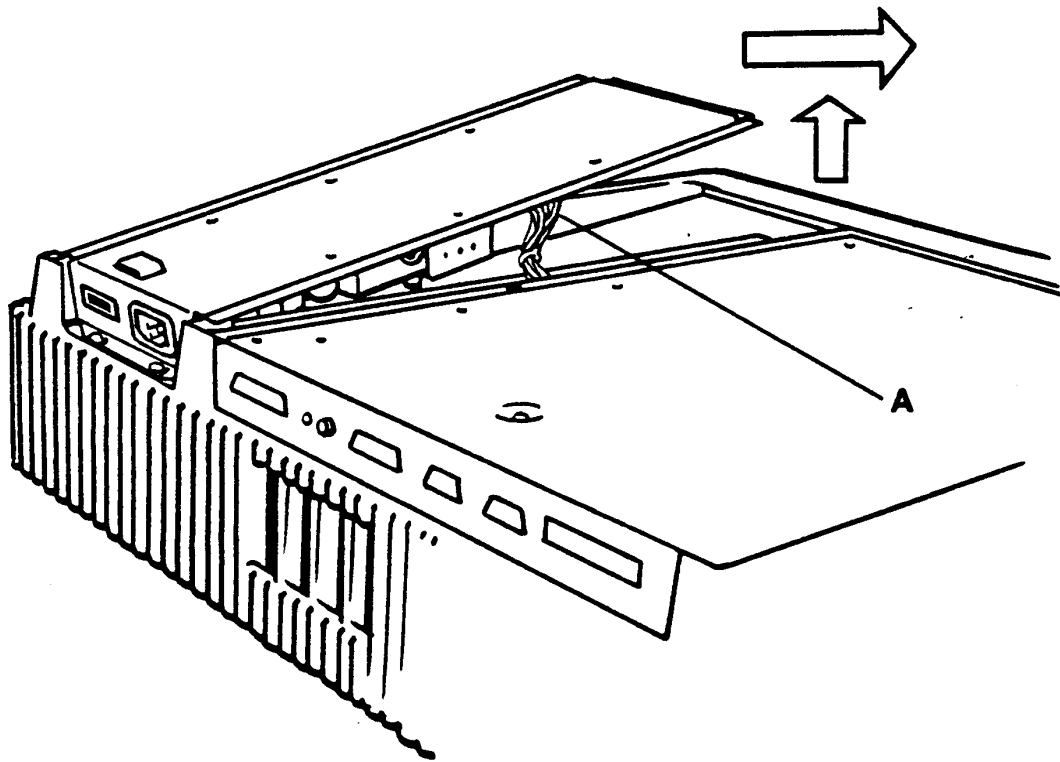


FIGURE 6.2

16.18



VII. LOGIC ASSEMBLY REMOVAL*

1. Power down the Apple ///. Disconnect the AC power cord from the source and then from the power supply receptacle of the Apple ///.
2. Disconnect all external cables.
3. Remove the peripheral logic access cover and all peripheral cards, or RFI shield cards. Refer to Procedure I and II.
4. Replace the access cover to protect disk bezel.
5. Place the Apple upside down on a soft pad. The rear of the Apple should face you.
6. Locate the ten (10) Phillips screws around the edge of the Logic access panel. Refer to Figure 7.1 item A. Locate the two (2) additional recessed screws that are about one and one-half inches in from the rear edge of the panel. Refer to Figure 7.1 item B.
7. Loosen these twelve (12) securing screws. These screws may be captured and if so should not come free of the access panel.

CAUTION: The logic board is attached to the access panel, and is still connected electrically to the keyboard, disk, speaker and the power supply. Cable travel allows the access panel to be tilted from the chassis about 45 degrees.

8. Slowly tilt up the access panel from the right side. Allow the panel to remain resting on its edge nearest the power supply. Refer to Figure 7.2.
9. Note the orientation and routing of the cables. While supporting the logic board from the underside remove the speaker cable (Figure 7.2 item A), the keyboard cable (Figure 7.2 item B), the disk cable (Figure 7.2 item C), and the power supply cable (Figure 7.2 item D).
10. The logic assembly is now free from the Apple and can be accessed for testing and repair.
11. To re-install the logic assembly reverse the procedure as outlined in steps 1 through 10 above.

CAUTION: Make sure cables are installed correctly and are not crimped or punctured by the mounting hardware when re-installing the logic assembly.



* The Logic Assembly is comprised of either three or four major elements, depending upon the time of manufacture. The earlier Logic Assemblies have four major elements: the access panel, the Main Logic board module, the Memory board module and the Encoder board module. The later Logic Assemblies have the Encoder board incorporated into the Main Logic board, and therefore, have three major elements: the access panel, the Main Logic board module and the Memory board module.

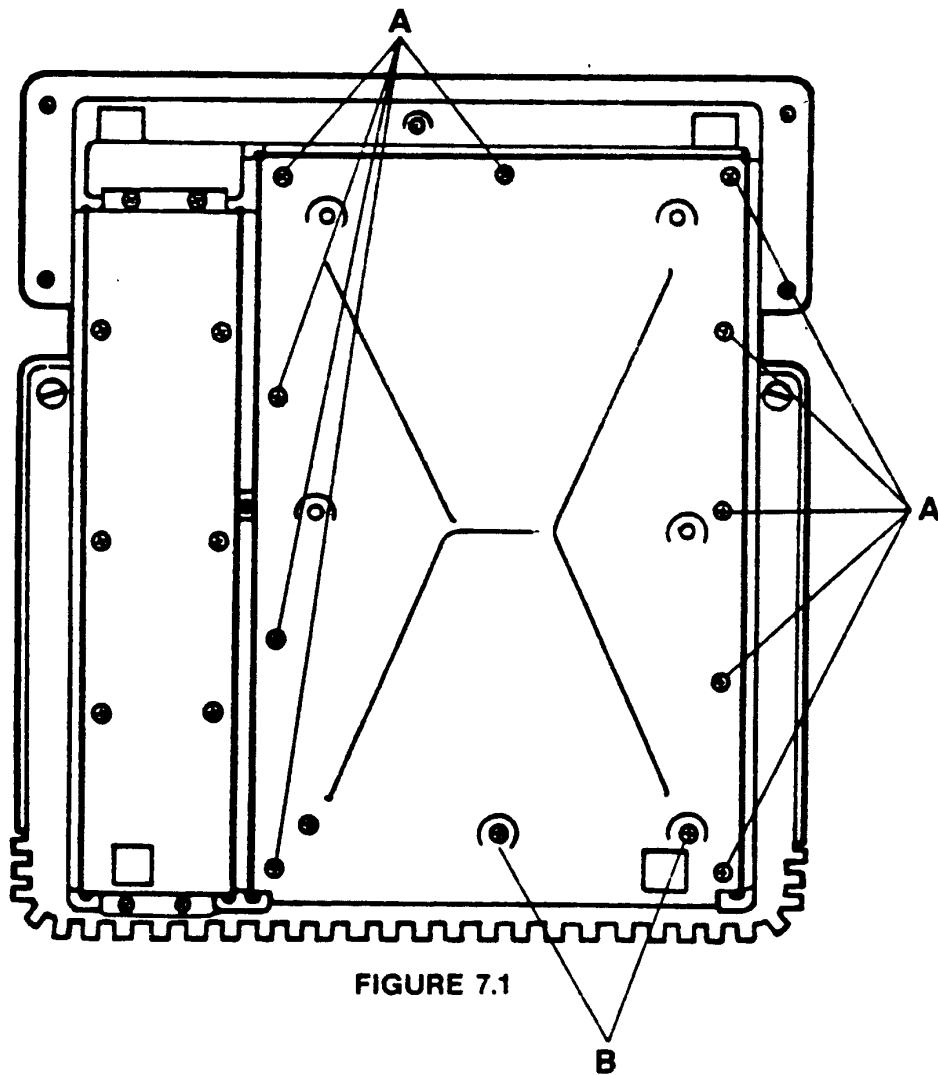


FIGURE 7.1

16.20

 apple computer inc.

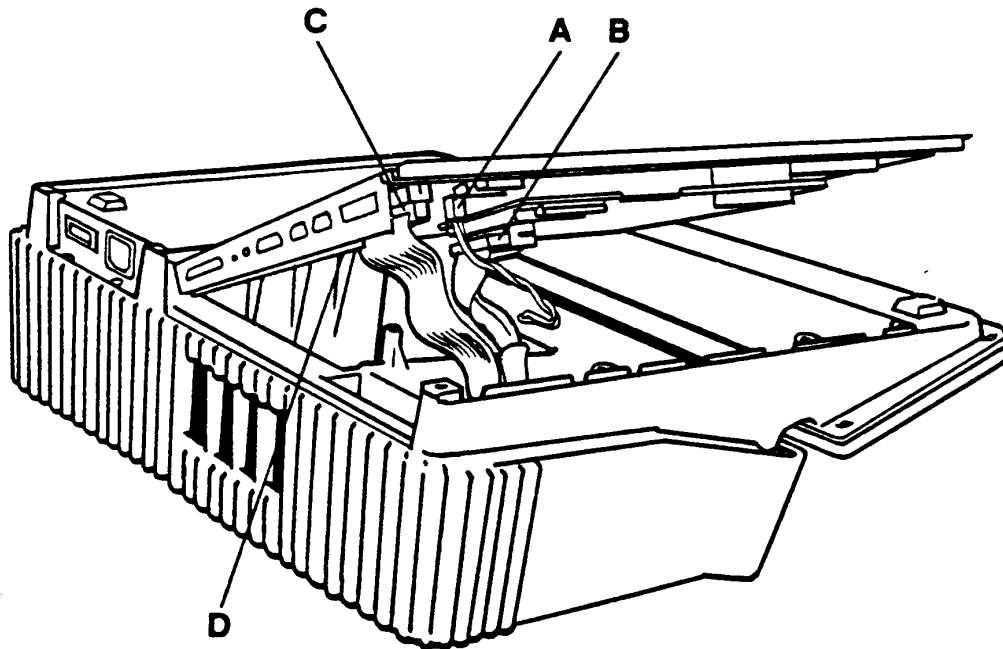


FIGURE 7.2

16.21



VIII. LOGIC ASSEMBLY REPLACEMENT

Before implementing any of this procedure the Logic Assembly must first be removed from the Apple. Refer to Procedure VII.

A. MEMORY BOARD REMOVAL

1. With the Logic Assembly placed flat on the work surface, use both hands to lift off the Memory board (Figure 8.1 item A) from the Main Logic board.

NOTE: The mechanical connection is also the electrical connection. Take care and lift straight up, or bending/breaking of the male connector pins mounted in the Main Logic board will occur.

B. MEMORY BOARD INSTALLATION

1. Align the connectors of the replacement Memory board over the connectors of the Main Logic board. The best way to do this is to tilt the Memory board and align the first pins on each side and lower the raised edge slowly, starting the next pins on each side as it is lowered.

CAUTION: Make sure that the board is properly oriented. The reference notches on the Memory board IC should face to the rear of the Logic Assembly (towards the output connectors).

2. Check that all the male pins are started correctly into the female connector of the Memory board. If any of the male pins are not properly started, lift up the Memory board slightly and "wiggle" it until the pins are aligned.

3. Once the pins of the connectors are all aligned, gently push straight down on the connectors on both sides of the Memory board until the connectors are fully seated.



C. ENCODER BOARD REMOVAL (early version of Logic Assembly only)

1. Locate Encoder board mount standoff. Refer to Figure 8.1 item B.

NOTE: The Encoder board connector on the Main Logic board has been re-formed for clearance purposes. This repositioning does not allow the standoff to match the pilot hole of the Encoder board. Therefore, the standoff can be removed, if you choose.

2. Remove the Encoder board from its connector.

D. ENCODER BOARD INSTALLATION (early version of logic assembly only)

1. Engage the replacement Encoder board on the connector and press into place.

E. MAIN LOGIC BOARD REPLACEMENT

1. Remove and retain the Memory board as detailed in Procedure VIII section A.
2. Remove and retain the Encoder board as outlined in Procedure VIII section C. (early version of logic assembly only)
3. Locate the retaining screw. Refer to Figure 8.1 item C. Remove and retain. Slide the board out from the peripheral connector opening of the access panel.
4. Lift off and set aside the Main Logic board.

CAUTION: Make sure that the insulating mylar shield (or substitute insulator) located between the board and the access panel remains in place before replacing the Main Logic board.

5. Place the replacement Main Logic board into the access panel, making sure that the peripheral connectors are aligned into their respective cutouts in the rear of the access panel.
6. Replace the retaining screw.
7. Replace the Encoder board. Refer to Procedure VIII section D.
8. Replace the Memory board. Refer to Procedure VIII section B.

 apple computer inc.

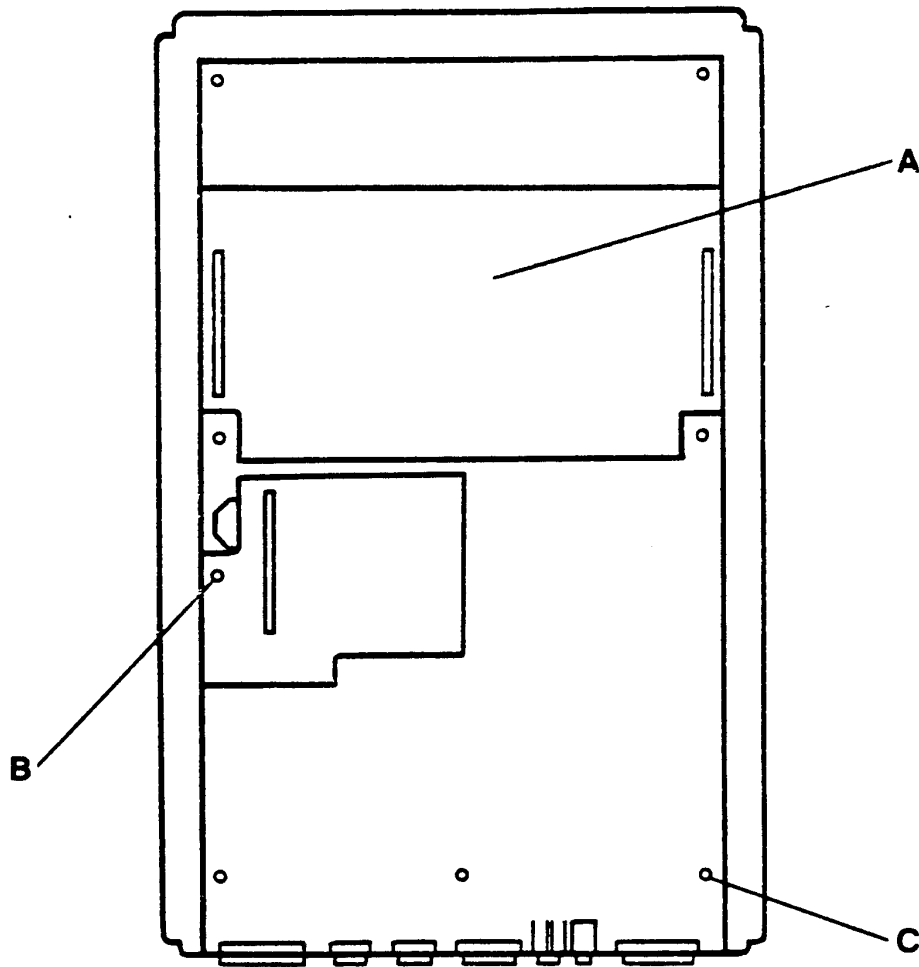


FIGURE 8.1

16.24



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

INTRODUCTION

The DIAGNOSTIC disk (part #652-0327), used in conjunction with this document will allow you to diagnose Apple /// failure modes at the modular level. Additionally, RAM failures may be diagnosed to the chip level. The descriptions of the tests that follow contain information relative to the specific test environment that you are using. For instance, it will be necessary to evaluate test results differently when using color verses B&W monitors.

EQUIPMENT REQUIRED

- Apple /// computer.
 - External Disk /// drives native to system under test.
 - B&W monitor w/ cable.
 - Apple /// Diagnostic diskette.
 - Apple /// External Test diskettes as required. *
- * See MAKE TEST DISKS.

OPTIONAL EQUIPMENT

- RGB Color Monitor, NTSC Color Monitor, or Color Receiver.
- Sup'r'mod'II, for Color Receiver.
- NTSC adapter, for Color Receiver and NTSC Color Monitor.
- Cables.

EQUIPMENT SETUP

As ALWAYS, insure that the Apple /// POWER is OFF BEFORE CONNECTING OR DISCONNECTING ANYTHING from the Apple /// or any equipment connected to the Apple ///.

- a. Connect external drives native to system under test.
- b. Connect B&W Monitor to J10 or,
- c. Where available: connect Color Monitor, NTSC adapter, and Sup'r'mod II to J5, as required.
- d. Connect power sources to system under test.
- e. Turn on monitor power.
- f. Install Diagnostic diskette in UUT internal drive.
- g. Install External Drive Test diskettes in external drives as required.

RUNNING DIAGNOSTICS

There are three critical operations that happen in order to run Apple /// Diagnostics.

With proper test setup configured as above and Diagnostic diskette installed in internal drive, turn on Apple /// power.

1. Power on internal diagnostic.

This fast internal diagnostic is described further in Apple /// owners guide. Refer to that document with questions on the start-up diagnostic. Let it suffice to say that these tests must be passed before the disk boot process may begin.
2. Boot

Disk boot is a several stage process that begins with the execution of a code block contained in ROM. On a successful



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

boot, control is passed to code loaded from disk. Several loads and transfers of control are made resulting in the loading and execution of the Diagnostic program.

3. Diagnostic

Having come this far is a vote of confidence for correct system operation. Now, under control of the Diagnostic, a thorough investigation of system hardware resources may begin.

DIAGNOSTIC MENU

After a successful boot load of the diagnostic program, the following menu will be presented.

TEST ALL

VIDEO	(NOT TESTED)
SOUND	(NOT TESTED)
RAM MAP	(NOT TESTED)
DISK	(NOT TESTED)
KEYBOARD	(NOT TESTED)
ROM	(NOT TESTED)

RAM TEST
MAKE TEST DISKS

CHOOSE:

ESC(APE
A(CCEPT S(KIP

The menu you actually see on the screen will show the 'TEST ALL' option in inverse video. This is to indicate that if the 'A' or A(CCEPT key is depressed, the inverse menu option will be selected for execution. Individual tests or the two special functions may be made candidates to A(CCEPT by S(KIPing through the list of menu options, with the 'S' or S(KIP key. As a menu option is S(KIPped, it is returned to normal video, and the next logical menu option is inverted or highlighted.

Finally, the Diagnostic program itself may be exited by depressing the 'ESC', or ESC(APE key. Actually the TAB key performs this function as well. This feature allows diagnostics running on an Apple][that use the same menu selection technique to 'feel' the same. ESC(CAPing the Diagnostic tests will prompt the user to reboot.



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

TEST ALL

A(CCEPTing this option will sequentially perform all of the diagnostic modules below:

- VIDEO
- SOUND
- RAM MAP
- DISK
- KEYBOARD
- ROM

Each of the Diagnostic modules operates in a manner identical to that encountered if each were A(CCEPTed individually. For a detailed description of the various Diagnostic modules, consult the following sections for each module by name.

VIDEO

The VIDEO Diagnostic module will test all of the various screen and color modes available on the Apple ///. You will be asked to make a subjective evaluation of each of the video mode tests. Of course if you are not using a Color Monitor, you will not be able to verify that the colors used in the test are actually present. Users of B&W Monitors will only be able to observe the different colors used as 16 shades of grey. An additional inconvenience that B&W Monitor users will have to put up with is that many B&W Monitors are not capable of displaying 16 linearly arranged shades of grey with a single setting of the monitor controls. In the description of each of the video mode tests, a warning will be given if this problem is anticipated.

The responses used for all of the video mode tests uses the format below:

- SPACE BAR TEST PASSES
- RETURN KEY TEST FAILS
- ESCAPE KEY LEAVE VIDEO TESTS
- LEFT ARROW KEY RETRY THE TEST

Except for the text mode and 16 color tests, each of the tests will display the same pattern: A picture of Winston Churchill will appear in the upper left corner, and a grid of diagonal lines will appear in the upper right corner, followed by a prompt field at the bottom of the screen.

1. HIRES MODE 1 - B&W pattern.
2. HIRES MODE 2 - B&W pattern.
3. 280 x 192 COLOR HIRES MODE 1 - Will appear as a negative image. A color monitor will show red and black.
4. 280 x 192 COLOR HIRES MODE 2 - Will appear as a green and white/ or yellow pattern.



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

5. SUPER HIRES MODE 1 - B&W pattern as in 1.
6. SUPER HIRES MODE 2 - B&W pattern as in 1.
7. AHIRE TEST 1 - On this and test 8, the screen will be divided into 4 horizontal sections, each a different color. The top half of Winston and the diagonal pattern are blue. The first two lines of the message are green, and the last two lines of the message are gold or orange. You may expect to have difficulty resolving the gray scale differences that represent these colors on a B&W Monitor on this and test 8.
8. AHIRE TEST 2 - pattern as in 7.
9. COLOR BAR & GRAY SCALE TEST - will display vertical bars of different colors. The border is blue. The colors, in order from left to right are: white, aqua, yellow, green, pink, grey, orange, brown, light blue, medium blue, grey, dark green, light purple, dark blue, magenta, and black. When viewed on a B&W Monitor, these colors will appear as a grey scale darkening from: white on left to black on right. You may expect to have difficulty resolving the gray scale differences that represent these colors on a B&W Monitor on this test. Although you will not be prompted for a response on this test, you must respond in the manner defined above. (That is... press space bar if test passes, etc.)
10. APPLE II TEXT MODE 1 - This screen will display:


```

                THE QUICK BROWN FOX JUMPS OVER THE LAZY DOGS

                abcdefghijklmnopqrstuvwxyz 0123456789
                                         (inverse)

                ...
                (flashing)
            
```
11. APPLE II TEXT MODE 2 - will display:


```

                22222222222222222222
                22222222222222222222
                22222222222222222222
                222
                222
                222
                22222222222222222222
                22222222222222222222
                22222222222222222222
                222
                222
                222
                22222222222222222222
                22222222222222222222
                22222222222222222222
            
```
12. APPLE /// 40 COLUMN TEXT MODE - display will be divided into sixteen



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

blocks each of one of the sixteen colors white through black. Each block will have a text phrase in a complementary color that describes the background color. You may expect to have difficulty resolving the gray scale differences that represent these colors on a B&W Monitor on this test. Although you will not be prompted for a response on this test, you are required to respond in the manner described above.

13. APPLE /// 80 COLUMN TEXT MODE - will display characters that are much smaller. In fact, you may expect to have difficulty reading these characters on a NTSC Color Monitor or Color Receiver due to the inherent 3.8 mhz bandwidth limitation of these types of displays. B&W composite video and RGB Color monitors will display this test screen in full splendor.

SOUND

1. SOFT BELL - The speaker will beep on and off. Press the space bar if you hear the sound. Press the return key if you do not.

2. HARD BELL - Same as 1. above but at a different pitch.

3. DAC OUTPUT - The digital to analog converter will produce a sound at the speaker output that periodically is of zero amplitude that grows in amplitude to be cut off again to zero amplitude. The change in amplitude should be regular and of constant pitch (except for the cut off of sound at max amplitude).

RAM MAP

This test does a cursory test of the ram to determine the configuration in the unit under test. The results will be displayed as a message describing the ram map determined in the test. eg:

RAM MAP GOOD FOR A 128K SYSTEM

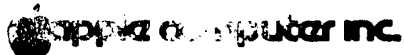
If the message does not correspond to the physical configuration determined by inspection, press return key to reject test. If the message describes the true configuration, press the space bar.

DISK

At this time you will be prompted to remove the Apple /// Diagnostic Diskette, and to insert the Internal Drive Test Diskette. Previous to this point, you should have inserted any External Drive Test Diskettes in external drives as required. This setup is acknowledged by depressing the return key. You will then be required to enter the number of external drives connected to the unit under test.

At this point the test runs automatically, terminated with a disk test summary for each drive specified above.

Upon completion of disk tests, you will be prompted to reinsert the Apple /// Diagnostic Diskette. You will acknowledge this operation by depressing the return key.



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

KEYBOARD

1. ALPHANUMERIC - depress each of the keys on the alphanumeric keyboard and verify that its representation on the display is removed.

Three keys require special key stroke sequences to test their function. These sequences are described at the top of the keyboard test screen. If, after depressing all alphanumeric keys, some key representations remain on the test display, then type the sequence CTRL-S.

2. SPECIAL FUNCTION KEYS - Depress the ALPHA LOCK key and note that the state of the ALPHA LOCK key changes. Leave the ALPHA LOCK key in the UP state.

Depress OPEN APPLE key and note that its state is reported as down.

Depress any key, (except SPACE or RETURN), and hold. Note the REPEAT SPEED. Anything from 5 to 15 / sec is acceptable. While holding a key down in repeat mode, depress the CLOSED APPLE key. Note that the REPEAT SPEED will increase to approximately double the normal rate.

Accept or reject this test as prompted at the bottom of the screen.

The SOLID APPLE key test will prompt you with a key stroke sequence that must be followed exactly. Perform the indicated operations as the test proceeds automatically.

3. NUMERIC KEYPAD - same as in 1. above.

ROM

This test is a straight forward go, no-go test, that reports the message: ROM PASSES... / ROM FAILS...

At this point, if you had selected the TEST ALL option, the main menu will appear with the test summary to the right of each of the test options.



APPLE /// DEALER SERVICE DIAGNOSTICS REFERENCE

OTHER DIAGNOSTIC OPTIONS

The Diagnostic Diskette menu contains two other options that may be accepted for execution. These options are separated from the options described above because they are 'one way' trips, in that upon completion of the execution of these modules, you will be required to reboot the unit under test. These modules are described below.

RAM TEST

This test performs a thorough exercising of all ram contained in the system. This test takes several seconds to complete and should not be interrupted. On invocation of this module however, you are given the chance to return to main menu.

Successful completion of this test module is indicated by the display of a diagnostic message on the upper left hand corner of the display. This message will contain a matrix of dots (.) and ones (1), if ram errors are encountered. These are placed in the matrix in a logical fashion. Ram chip failures may be determined by comparing ones (1) found in the displayed matrix against the ram chip locator matrix below.

```

B9 B8 B7 B6 B5 B4 B3 B2 *
B17 B16 B15 B14 B13 B12 B11 B10 *
B9 B8 B7 B6 B5 B4 B3 B2
B17 B16 B15 B14 B13 B12 B11 B10
C17 C16 C15 C14 C13 C12 C11 C10
D9 D8 D7 D6 D5 D4 D3 D2
D17 D16 D15 D14 D13 D12 D11 D10
C9 C8 C7 C6 C5 C4 C3 C2

```

* DISREGARD THESE TWO ROWS ON 96K SYSTEMS

NOTE: All other diagnostic messages displayed shall be disregarded.
No prompt will be given to reboot the system under test.

MAKE TEST DISKS

You will use this module to prepare Drive Test Diskettes used in the DISK test module described above.

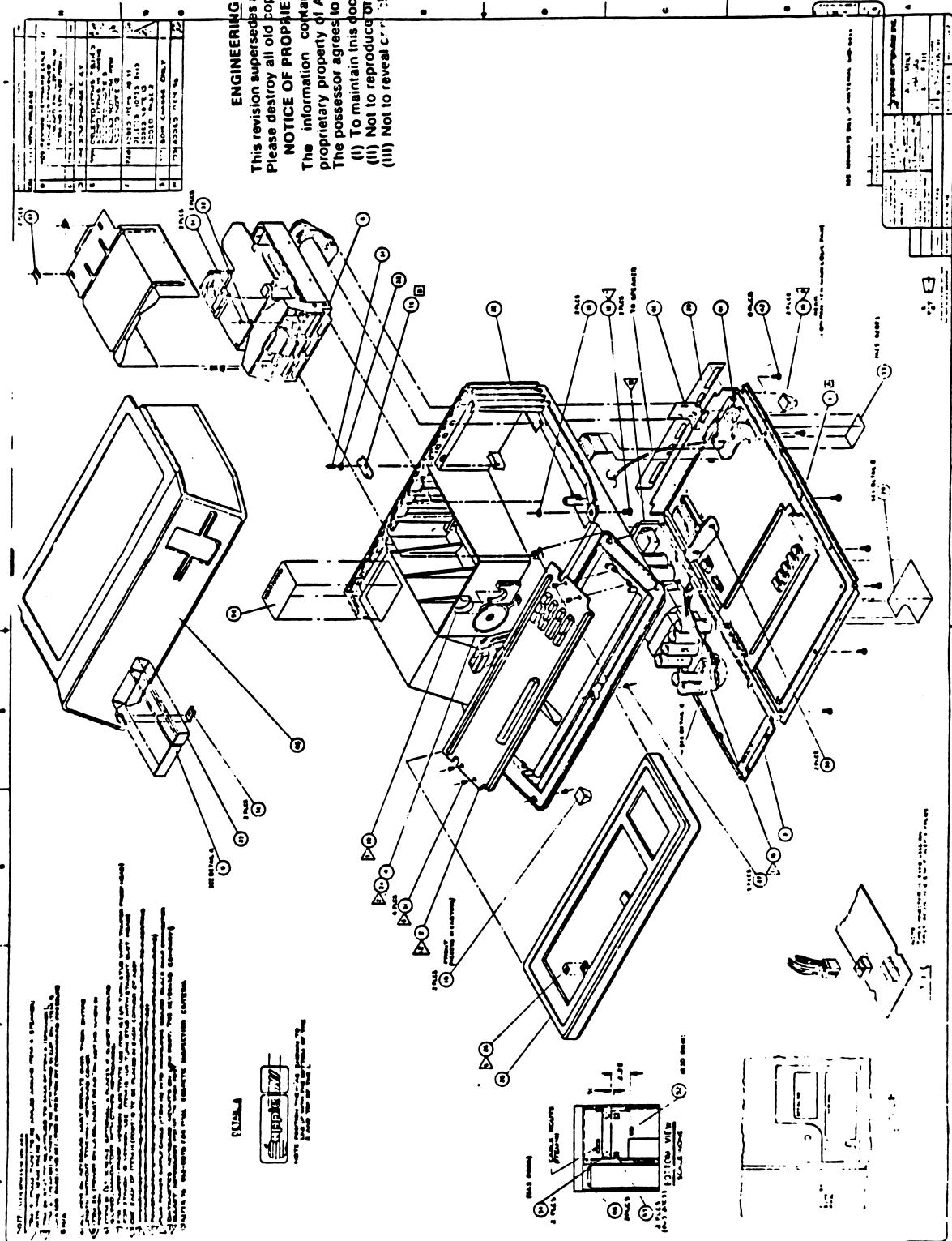
Upon accepting this module you will be prompted to remove the Diagnostic Diskette.

You will then be prompted to enter the drive number corresponding to the test diskette you wish to make. Enter this number <1 = internal , 2..4 = external>, followed by depressing RETURN.

You will be asked to insert a blank diskette into internal drive and acknowledge by depressing RETURN.

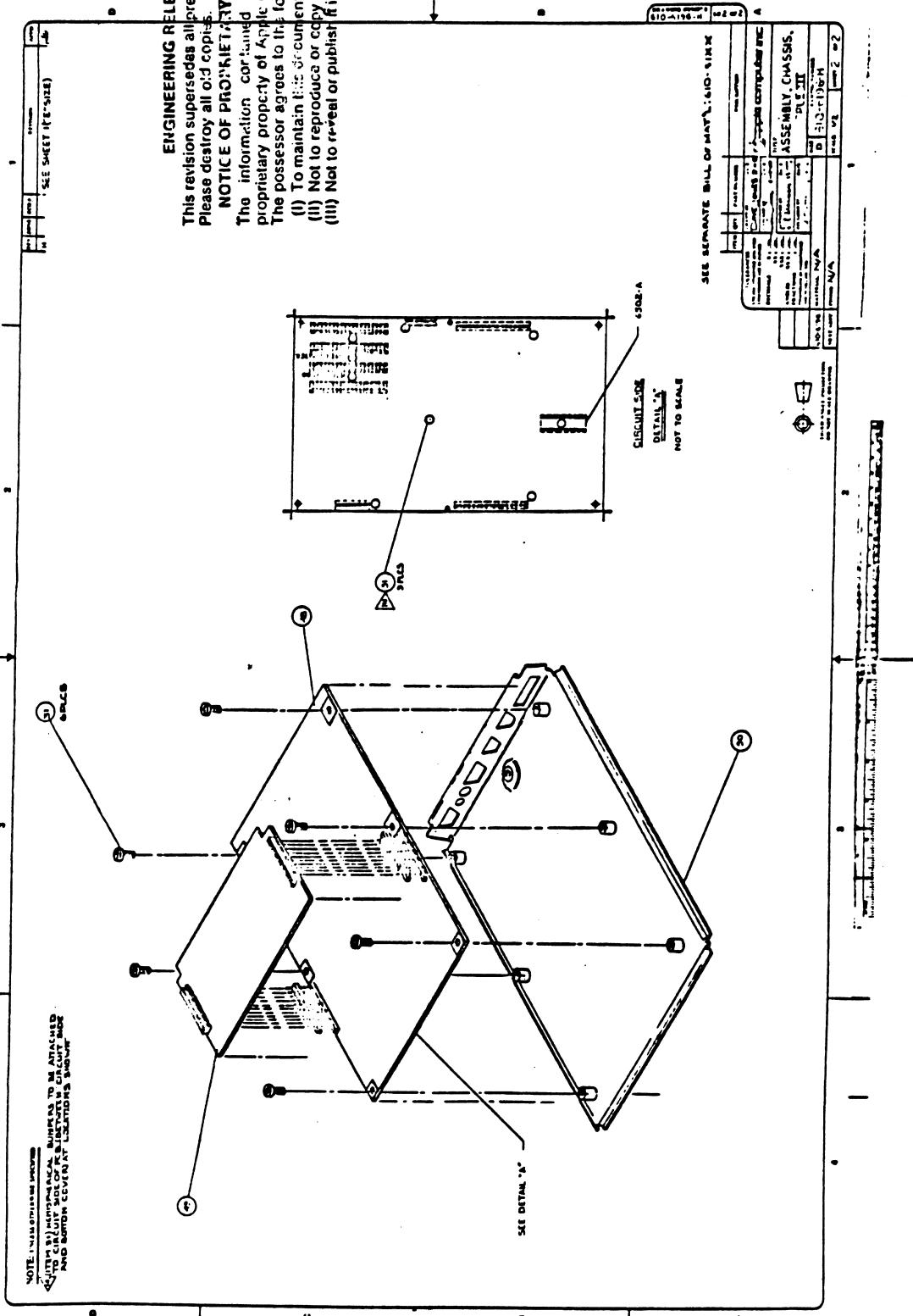
The utility will inform you that it is creating the test diskette for the drive that you specified in the operation above.

A 'DONE message' will be displayed on completion. You will be asked if you want to make another diskette. If you reply 'y' the above process will be repeated, otherwise you will be prompted to reboot.



ENGINEERING RELEASE
 This revision supersedes all previous versions.
 Please destroy all old copies.
NOTICE OF PROPRIETARY PROPERTY
 The information contained herein is the
 proprietary property of Apple Computer, Inc.
 The possessor agrees to the following:
 (i) To maintain this document in confidence
 (ii) Not to reproduce or copy it.
 (iii) Not to reveal its contents in whole or part.

REV.	DATE	DESCRIPTION	BY
1	1982.01.15	INITIAL RELEASE	...
2	1982.02.01
3	1982.03.10
4	1982.04.20
5	1982.05.15
6	1982.06.01
7	1982.07.10
8	1982.08.01
9	1982.09.15
10	1982.10.01
11	1982.11.10
12	1982.12.01
13	1983.01.15
14	1983.02.01
15	1983.03.10
16	1983.04.01
17	1983.05.15
18	1983.06.01
19	1983.07.10
20	1983.08.01
21	1983.09.15
22	1983.10.01
23	1983.11.10
24	1983.12.01
25	1984.01.15
26	1984.02.01
27	1984.03.10
28	1984.04.01
29	1984.05.15
30	1984.06.01
31	1984.07.10
32	1984.08.01
33	1984.09.15
34	1984.10.01
35	1984.11.10
36	1984.12.01
37	1985.01.15
38	1985.02.01
39	1985.03.10
40	1985.04.01
41	1985.05.15
42	1985.06.01
43	1985.07.10
44	1985.08.01
45	1985.09.15
46	1985.10.01
47	1985.11.10
48	1985.12.01
49	1986.01.15
50	1986.02.01
51	1986.03.10
52	1986.04.01
53	1986.05.15
54	1986.06.01
55	1986.07.10
56	1986.08.01
57	1986.09.15
58	1986.10.01
59	1986.11.10
60	1986.12.01
61	1987.01.15
62	1987.02.01
63	1987.03.10
64	1987.04.01
65	1987.05.15
66	1987.06.01
67	1987.07.10
68	1987.08.01
69	1987.09.15
70	1987.10.01
71	1987.11.10
72	1987.12.01
73	1988.01.15
74	1988.02.01
75	1988.03.10
76	1988.04.01
77	1988.05.15
78	1988.06.01
79	1988.07.10
80	1988.08.01
81	1988.09.15
82	1988.10.01
83	1988.11.10
84	1988.12.01
85	1989.01.15
86	1989.02.01
87	1989.03.10
88	1989.04.01
89	1989.05.15
90	1989.06.01
91	1989.07.10
92	1989.08.01
93	1989.09.15
94	1989.10.01
95	1989.11.10
96	1989.12.01
97	1990.01.15
98	1990.02.01
99	1990.03.10
100	1990.04.01



ENGINEERING RELEASE
 This revision supersedes all previous versions.
 Please destroy all old copies.

NOTICE OF PROPRIETARY PROPERTY
 The information contained herein is the
 proprietary property of Apple Computer, Inc.
 The possessor agrees to the following:
 (i) To maintain this document in confidence.
 (ii) Not to reproduce or copy it.
 (iii) Not to reveal or publish it in whole or part.

DATE	REV	DESCRIPTION
10/1/81	1	INITIAL RELEASE
10/1/81	2	REVISION
10/1/81	3	REVISION
10/1/81	4	REVISION
10/1/81	5	REVISION
10/1/81	6	REVISION
10/1/81	7	REVISION
10/1/81	8	REVISION
10/1/81	9	REVISION
10/1/81	10	REVISION
10/1/81	11	REVISION
10/1/81	12	REVISION
10/1/81	13	REVISION
10/1/81	14	REVISION
10/1/81	15	REVISION
10/1/81	16	REVISION
10/1/81	17	REVISION
10/1/81	18	REVISION
10/1/81	19	REVISION
10/1/81	20	REVISION
10/1/81	21	REVISION
10/1/81	22	REVISION
10/1/81	23	REVISION
10/1/81	24	REVISION
10/1/81	25	REVISION
10/1/81	26	REVISION
10/1/81	27	REVISION
10/1/81	28	REVISION
10/1/81	29	REVISION
10/1/81	30	REVISION
10/1/81	31	REVISION
10/1/81	32	REVISION
10/1/81	33	REVISION
10/1/81	34	REVISION
10/1/81	35	REVISION
10/1/81	36	REVISION
10/1/81	37	REVISION
10/1/81	38	REVISION
10/1/81	39	REVISION
10/1/81	40	REVISION
10/1/81	41	REVISION
10/1/81	42	REVISION
10/1/81	43	REVISION
10/1/81	44	REVISION
10/1/81	45	REVISION
10/1/81	46	REVISION
10/1/81	47	REVISION
10/1/81	48	REVISION
10/1/81	49	REVISION
10/1/81	50	REVISION
10/1/81	51	REVISION
10/1/81	52	REVISION
10/1/81	53	REVISION
10/1/81	54	REVISION
10/1/81	55	REVISION
10/1/81	56	REVISION
10/1/81	57	REVISION
10/1/81	58	REVISION
10/1/81	59	REVISION
10/1/81	60	REVISION
10/1/81	61	REVISION
10/1/81	62	REVISION
10/1/81	63	REVISION
10/1/81	64	REVISION
10/1/81	65	REVISION
10/1/81	66	REVISION
10/1/81	67	REVISION
10/1/81	68	REVISION
10/1/81	69	REVISION
10/1/81	70	REVISION
10/1/81	71	REVISION
10/1/81	72	REVISION
10/1/81	73	REVISION
10/1/81	74	REVISION
10/1/81	75	REVISION
10/1/81	76	REVISION
10/1/81	77	REVISION
10/1/81	78	REVISION
10/1/81	79	REVISION
10/1/81	80	REVISION
10/1/81	81	REVISION
10/1/81	82	REVISION
10/1/81	83	REVISION
10/1/81	84	REVISION
10/1/81	85	REVISION
10/1/81	86	REVISION
10/1/81	87	REVISION
10/1/81	88	REVISION
10/1/81	89	REVISION
10/1/81	90	REVISION
10/1/81	91	REVISION
10/1/81	92	REVISION
10/1/81	93	REVISION
10/1/81	94	REVISION
10/1/81	95	REVISION
10/1/81	96	REVISION
10/1/81	97	REVISION
10/1/81	98	REVISION
10/1/81	99	REVISION
10/1/81	100	REVISION

16.33

apple computer inc.

DOCUMENT NO. 064-0167

PAGE 4 OF 14

FIG. 1

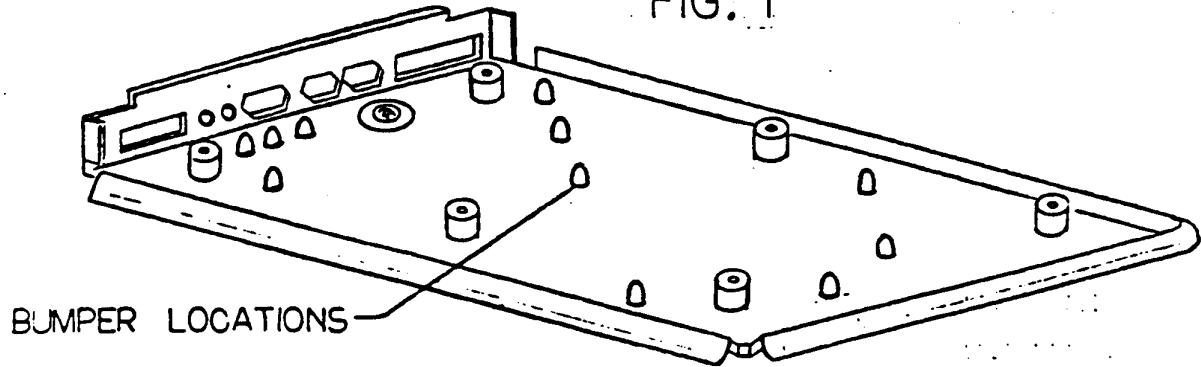
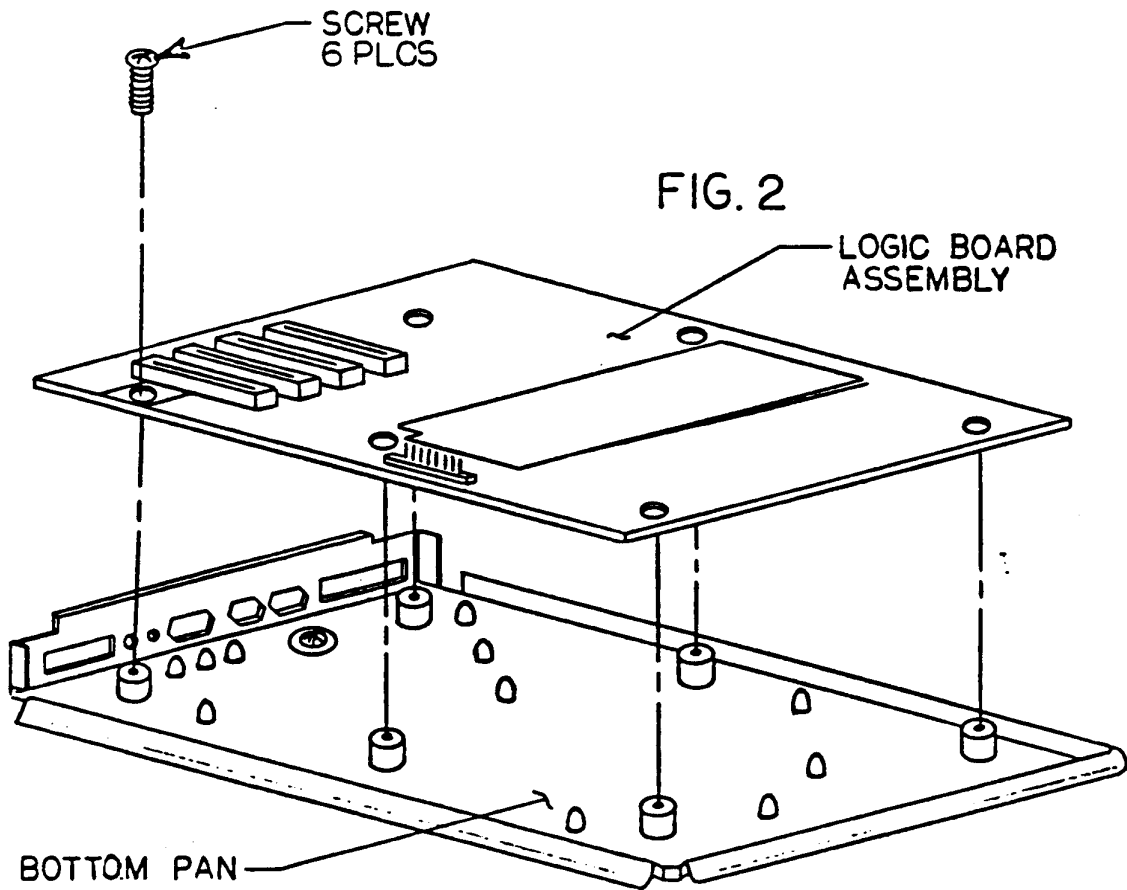


FIG. 2



DRAWN BY
M. Castillo

OPERATION NO.
A20

TITLE
LOGIC BOARD PAN ASSY.

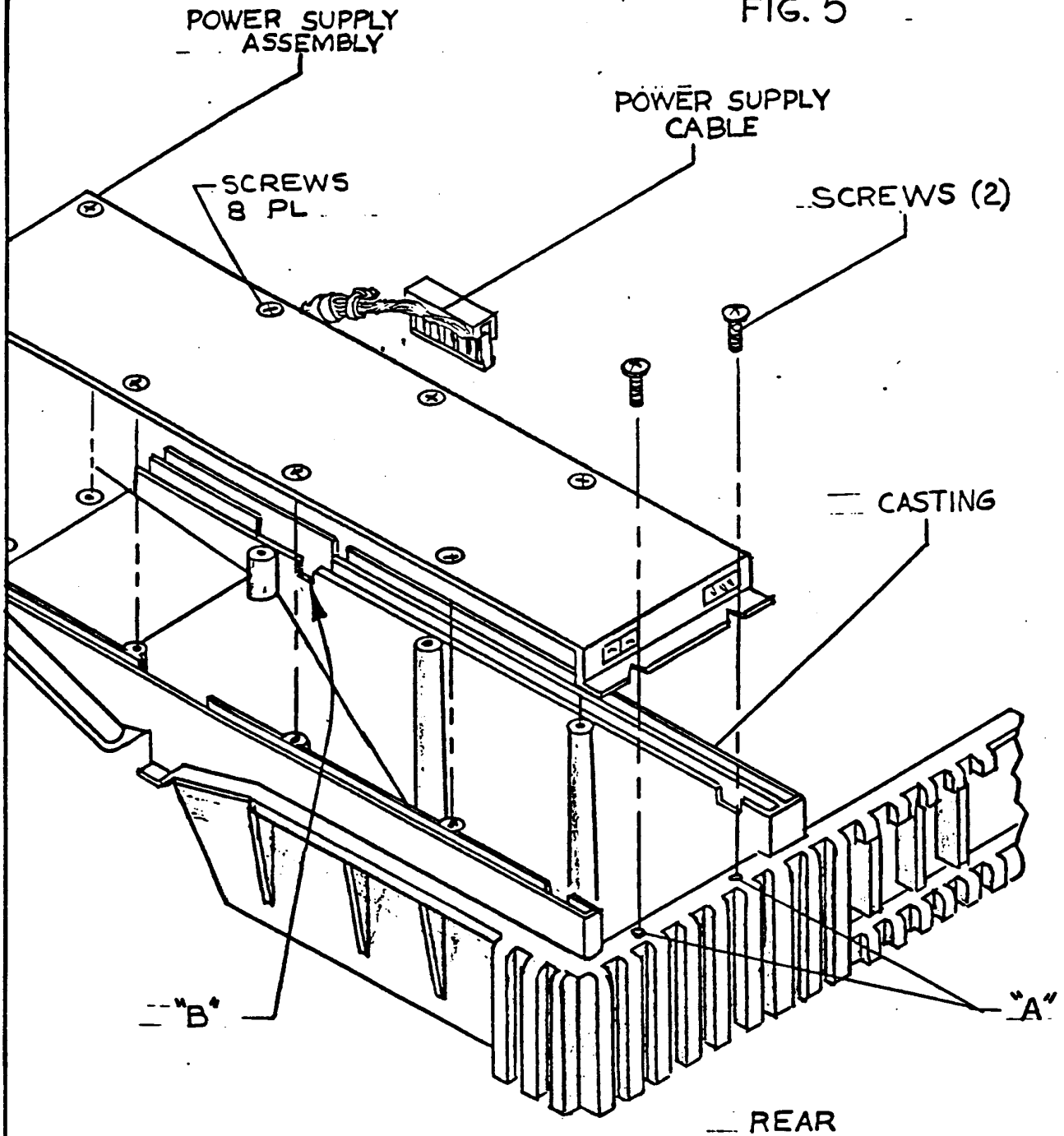
16.34



DOCUMENT NO. 064-0156

PAGE 11 OF 14

FIG. 5



DRAWN BY

OPERATION NO.

TITLE

040

INSTALL POWER SUPPLY

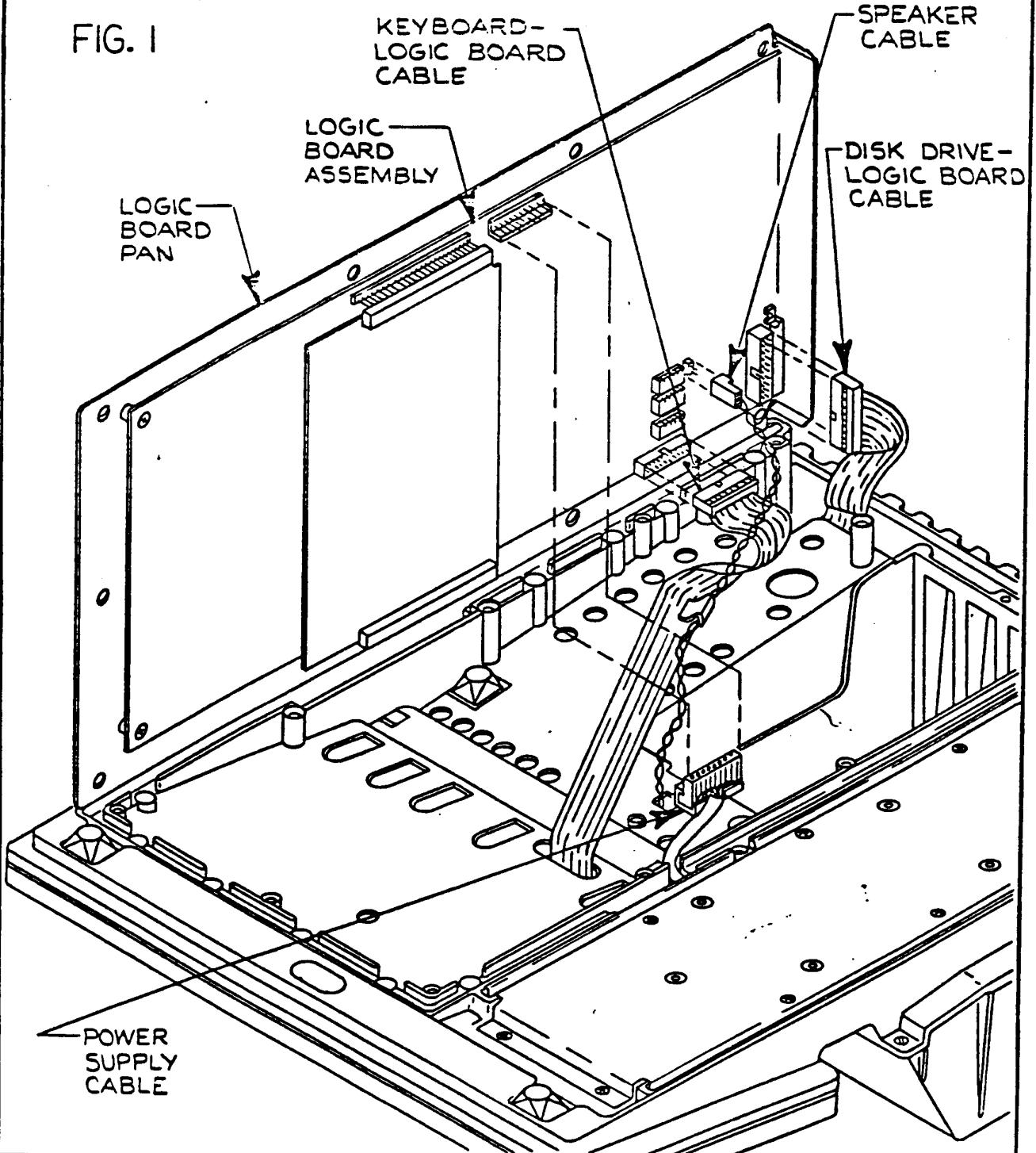
16.35

Apple Computer Inc.

NO. 064-0022

PAGE 3 OF 6

FIG. 1



DRAWN BY
M. Basilio

OPERATION NO.
050

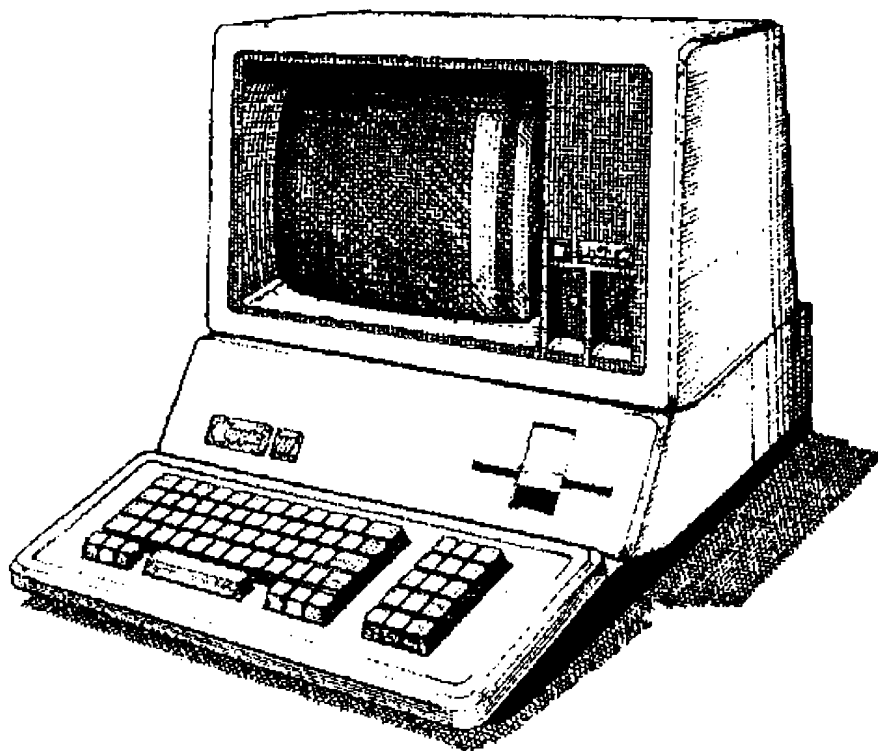
TITLE
INSTALL LOGIC BOARD ASSY.

16.36



Apple /// Computer Information

Apple /// Service Reference Manual



Section II of II • Servicing Information

Chapter 17 • General Appendix

Written by Apple Computer • 1982



APPLE /// SYSTEM OVERVIEW

FEATURING:

- [] EXTENDED DISPLAY
- [] EXPANDED RAM
- [] NEW PROCESSOR DESIGN
- [] BUILT IN I/O
- [] APPLE II EMULATOR

STARRING:

THE APPLE /// PROCESSOR

- * 6502 INSTRUCTION SUBSET
- * RELOCATABLE BASE REGISTER PAGE
- * RELOCATABLE STACK
- * 256K BYTE ADDRESS RANGE

VIDEO

- * NTSC COLOR COMPOSITE VIDEO
- * NTSC B/W COMPOSITE VIDEO
- * SYNC
- * 4 PRIMARY INDEPENDENT VIDEO LINES
- * MIX TO FROM RGB APPLE COLORS
- THREE LINES CAN DRIVE TTL RGB MONITOR
- FOUR INDEPENDENT VIDEO OUTPUTS CAN BE GENERATED

DISPLAY MODES

- * GRAY SCALE ON B/W OUTPUT
- * RAM CHARACTER GENERATOR (128 CHAR.)
- * 40 X 24 CHARACTER B/W TEXT (2K BYTES RAM)
- * 80 X 24 CHARACTER B/W TEXT
- * 40 X 24 CHARACTER COLOR TEXT
(16 BACKGROUND, 16 TEXT COLORS)
- * 280 X 192 B/W HIRES (8K RAM)
- * 560 X 192 B/W HIRES
- * 140 X 192 16-COLOR HIRES
- * 280 X 192 16-COLOR HIRES WITH 40 X 192
BACKGROUND/BACKGROUND RESOLUTION

I/O

- * FOUR APPLE II BUS PERIPHERAL SLOTS
- * ONE BUILT IN DISK DRIVE
- * CONTROLLER FOR THREE ADDITIONAL DRIVES
- * RS 232 PORT (COMPLETE)
- * SILENTYPE PORT
- * TWO PADDLE PORTS (A/D INPUTS)
- * EXTERNAL SOUND JACK (SIX BIT AUDIO,
HARDWARE BEEPER
- * EXPANDED VIDEO LINES



- * CLOCK/CALENDER

I/O BLOCK TRANSFER

- * PERMITS UP TO 1 PAGE (256 BYTE) FAST I/O TRANSFER WITHOUT DMA HARDWARE ON PERIPHERAL
- * TRANSFERS AT UP TO THE RAM CYCLE RATE

APPLE II EMULATION RESTRICTIONS

- * NO LANGUAGE OR ROM CARD
- * PADDLES ARE DIFFERENT
- * ENTER WITH SOFTWARE BUT ONLY RESET WILL EXIT



6.1 Development Monitor

The current version of the diagnostic/boot ROM includes a monitor that may be useful for debugging. THE MONITOR IS NOT A PART OF THE SUPPORTED Apple III SOFTWARE AND IT MAY BE CHANGED OR DELETED IN FUTURE VERSIONS OF THE BOOT ROM. MONITOR SUBROUTINES MUST NOT BE CALLED FROM DRIVERS OR INTERPRETERS.

6.2 Monitor Commands

The monitor commands are listed below. Commands are read directly from the keyboard; blanks are not used except as data separators in a {byte list}. Command lines may be up to 79 characters long and are always terminated by a RETURN. Multiple commands may be entered on the same line using a slant (/) as a command separator. Numeric data is always entered in hex. ASCII strings may be entered by enclosing them in single or double quotes. If single quotes (') are used, bit 7 of each byte will be clear; if double quotes (") are used, bit 7 will be set.

```

{address} ::= numeric value 0..FFFF
{address range} ::= {address} | {address}.{address}
{byte} ::= numeric value 0..FF | Single byte string
{byte list} ::= one or more bytes separated by blanks
{block num} ::= numeric value 0..117

{address range}           Memory dump
{address} : {byte list}   Memory store
{byte} < {address range} S Memory search
{address} < {address range} M Memory move
{address} < {address range} V Memory verify
{block num} < {address range} R Read disk
{block num} < {address range} W Write disk
{address} G               Subroutine call (JSR)
{address} J               Execute code (JMP)
U                          Call user subroutine (JSR $3F8)
X                          Repeat command line
RETURN                    Continue memory dump

```

When dumping memory, the output can be suspended then stepped by pressing the space bar; pressing any other key will resume normal output. Pressing the TAB key will terminate processing of the command line.

6.3 Escape Commands

Escape commands may be used during command input to move the cursor or control the display. Escape mode is entered by typing an ESCAPE; the cursor is changed to a flashing plus sign (+) to identify escape mode. Any character that is not an escape command will terminate escape mode.

```

↑ or Kc   Move cursor up
↓ or Jc   Move cursor down
← or Hc   Move cursor left
→ or Uc   Move cursor right

```

17.3



Apple III I/O System Programmer's Guide

L	Clear to end of line
P	Clear to end of page
S	Home cursor and clear screen
4	Set 40 column display
8	Set 80 column display

6.4 Zero Page Locations

58	Window left
59	Window right
5A	Window top
5B	Window bottom
5C	Horizontal cursor position
5D	Vertical cursor position
74,75	A1
76,77	A2
78,79	A3
7A,7B	A4

6.5 Entering Addresses

adr1	A1, A2, A := adr1		
adr1.adr2	A1, A3 := adr1	A2 := adr2	
adr3<adr1.adr2	A1, A3 := adr1	A2 := adr2	A4 := adr3

6.6 Pointer Usage

Memory dump:	(A1)
Memory store:	(A3)
Memory move:	(A4) := (A1)
Memory verify:	(A4) : (A1)
Memory search:	A4 : (A1)



APPLE /// LOGIC SIGNAL SOURCE

The location of the signal source is described first by the page number of the schematic followed by its coordinates.

SIGNAL	LOCATION	MEANING
6551sel*	5-4a	ACIA SEL
a0-a7	4-4d	ADDRESS BUS (EXTERNAL)
a8-a11	4-4c	
a12-a15	4-4b	
abk1	3-2c	ADDRESS BANK 1
abk2	3-2c	ADDRESS BANK 2
abk3	3-2c	ADDRESS BANK 3
abk4	3-2c	ADDRESS BANK 4
ahires	6-4a	AHIRES
alllores	6-4a	APPLE II LORES
altstk*	5-1b	ALTERNATE STACK
apple i*	9-4b	APPLE SWITCH 1
apple ii*	9-4b	APPLE SWITCH 2
ar0	3-3b	RAM ADDRESS X
anyky	9-4c	ANYKEY (DEPRESSED)
ar0	3-3d	
ar1	3-3d	
ar2	3-3c	
ar3	3-3c	
ar4	3-3c	
ar5	3-3b	
ar6	3-3b	
audio	8-1b	
ax,ax*	10-3a	ADDRESS MUX SELECT
axco	5-2a	
bcksw1,3	8-4b	BANKSWITCH
bl	10-1b	BLANKING
c08xn-c0Fxn	5-4a	ADDRESS DECODE
c0xxn-c7xxn	5-4b	ADDRESS DECODE
clm,clm*	5-4b	CLOCK 1 MEGHERTZ
cl4m,*	10-4c	CLOCK 14 MEGHERTZ
c3,5m,c3,5m*	10-4c	
c7m,c7m*	10-4b	
caplk*	9-4b	CAPS LOCK SWITCH
cas0*	3-2c	COLUMN ADDRESS SELECT (RAM)
cas1*	3-2c	
cas2*	3-2c	
cas3*	3-2c	
cas4,7*	3-2c	
cas5,6*	3-2c	
ch80*	6-4a	CHARACTER (80 COLUMN)
clkbatt	5-2d	CLOCK BATTERY
clken80	6-2d	CLOCK ENABLE 80 (COLUMN)



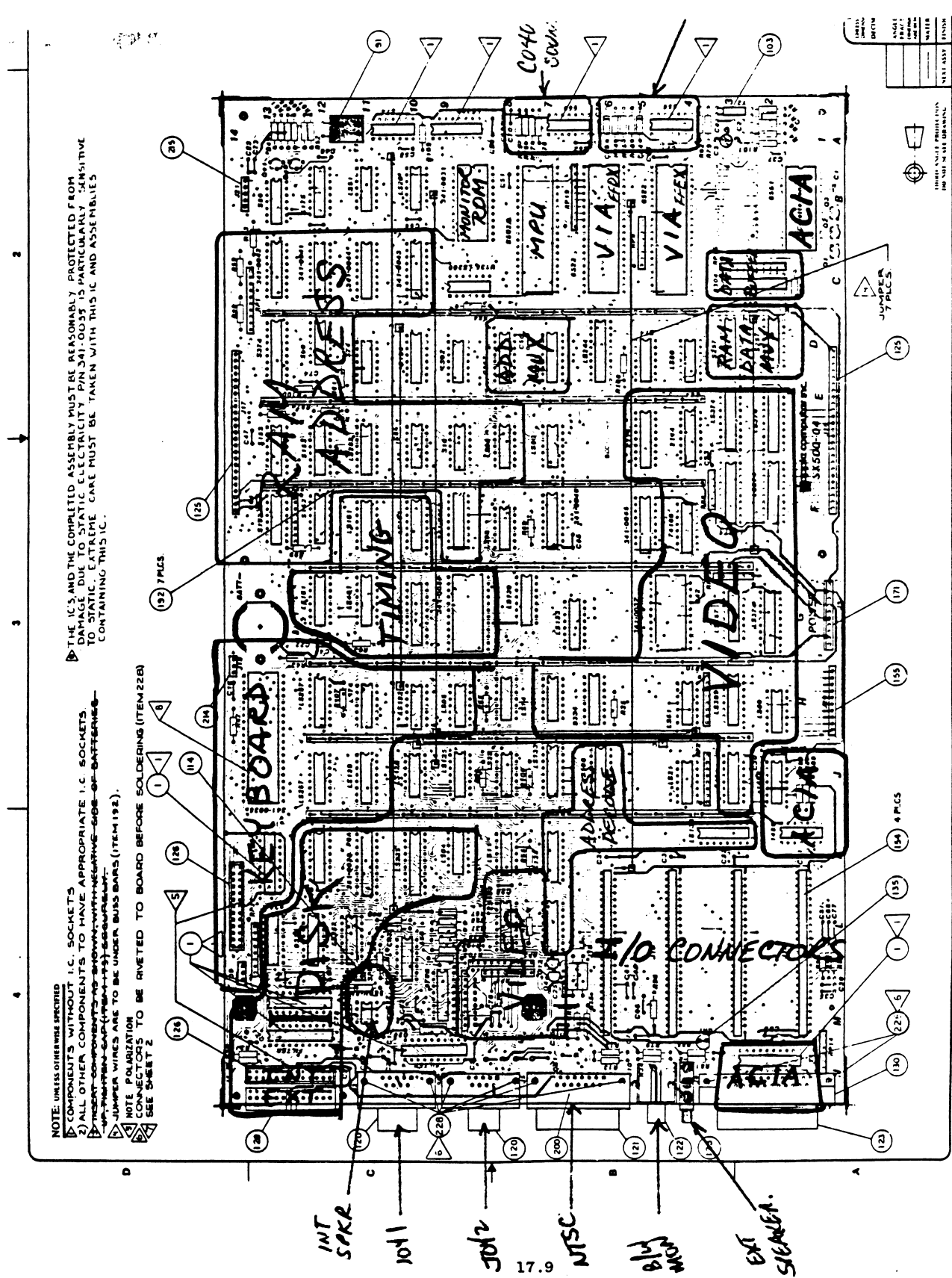
clkirq*	5-1c	CLOCK INTERRUPT
clrkl*	6-4a	
clrstrb*	6-3a	CLEAR STROBE (KEYBOARD)
colrgate	10-1b	COLOR GATE
comp	10-1b	COMPENSATE
control*	9-4b	CONTROL KEY
cs6522	10-4a	
cts	8-1c	CLEAR TO SEND (EIA)
c00x-c07x	5-3a	ADDRESS DECODE
cxxx	5-4b	
c-fxxx	5-4c	
d0-d7	4-2c	
da0,da7	10-3b	
dataIn	8-1c	EIA
db0-db7	10-4b	
dc0,dc7	6-3d	VIDEO BUS (CHAR GEN OUT)
dcd	8-1c	DATA CARRIER DETECT
devsell,6*	5-4a	DEVICE SELECT
dhires	5-4a	DHIRES
dma1*	4-4d	DIRECT MEMORY ACCESS IN
dmaok	4-4a	DMA OK
dph0,dph3	7-4b	DISK (MOTOR) PHASE X
dsply	10-1b	DISPLAY
dsr	8-1c	DATA SET READY (EIA)
dtrdy,*	9-4c	DATA READY (KEYBOARD)
dtr	8-1c	DATA TERMINAL READY
dv0,dv7	6-4d	VIDEO BUS (RAM LATCH OUT)
dxo,dx7	6-3a	VIDEO BUS (COLOR LATCH OUT)
en257	5-3d	ENABLE RAM BUS TO DATA BUS
en8304	5-3d	ENABLE MPU XCVR
enb11,e*	7-2c	ENABLE DISK 1 EXTERNAL
enb11,i*	7-2c	ENABLE DISK 1 INTERNAL
enb12,e*	7-2b	ENABLE DISK 2 EXTERNAL
enb13,i	7-2b	ENABLE DISK 3 EXTERNAL
encwrt	7-2c	ENABLE CHARACTER RAM WRITE
enhreg*	3-3c	ENABLE GRAPHICS REGISTER (LATCH)
ensel	7-2c	ENABLE EXT PRINTER SELECT
ensio	7-2c	ENABLE SERIAL I/O
PDL0	5-2a	PADDLE ADDRESS 0
ext*	7-4b	EXTERNAL (DRIVE)
extspk	8-1b	EXTERNAL SPEAKER
ffcx*	5-3c	
ffdx*	5-3c	
ffex*	5-3c	
fieldout	10-1b	
fspace*	5-3b	
gph1	5-3c	
gph2	5-3c	
h0,h3	10-2d	VIDEO STATE
h4,h5	10-2d	VIDEO STATE
hires	5-2a	
hpe*	10-2d	
id0-id7	4-3c	INTERNAL DATA BUS
ind*	3-3a	



inh*	5-4c	INHIBIT (ROM)
int*	7-2d	INTERNAL (DRIVE)
blankin	8-3c	
ioen	5-1b	I/O (DEVICES) ENABLE
ionmi*	9-2b	I/O NON MASKABLE INTERRUPT
iosell,4*	5-4b	I/O SELECT
iostopd*	10-4b	I/O STOPPED
iostrb*	5-4b	I/O STROBE
io sync	4-3b	I/O SYNC
irq1,4*	5-3c	
i r*/w	4-2c	INTERNAL READ/WRITE
i r/w*	4-3c	
kbint*	9-2c	KEYBOARD INTERRUPT
kbo*	5-3a	KEYBOARD STROBE(ENABLE KEY TO DATA)
kreset*	9-4b	RESET KEY
kvcc	9-4b	KEYBOARD VCC
ldps*	10-4c	LOAD PARARELL SHIFT REGISTER
mix	5-2a	
muxl	3-3c	
nmi*	9-1b	
ntsca,b	6-2b	
oe374	6-4a	ENABLE COLOR LATCH OUTPUTS
pa8	4-4b	PROCESSOR ADDRESS 8
pa15	4-4b	PROCESSOR ADDRESS 15
page 2	5-2a	
pcas0*	3-2d	PROM CAS X
pcas1*	3-2d	
pcas2*	3-2d	
pcas3*	3-2c	
pdint*	5-1c	POWER DOWN INTERRUPT (CLOCK)
pd12	5-2a	PADDLE ADDRESS 2
pdlen	5-2a	PADDLE ENABLE (ADC RAMPSTART)
pdlot*	8-3a	PADDLE OUT (RAMPSTART)
pg2*	6-4a	
ph0	10-3a	PROCESSOR 0
ponrst	9-3a	POWER ON RESET
pras0,3*	3-2b	PROM RAS X
pras1,2*	3-2b	
pras4,5*	3-2b	
pras6,7*	3-2b	
preim	10-4b	PRE 1 MEGAHERTZ
pwrwn*	5-2d	POWER DOWN
q0,q3	10-3c	Q STATE (ZMEG)
ram r/w	3-1b	
ras, ras*	10-4b	RAM ROW ADDRESS SELECT
ras0,3*	3-1b	
ras1,2*	3-1b	
ras4,5*	3-1b	
ras6,7*	3-1b	
rbl	10-1c	ROM BLANK
relkpwr	5-2d	CLOCK POWER
rcolrgt	10-1c	ROM COLORGATE
rddata	7-2d	READ DATA (DISK)
rdy	3-3a	READY



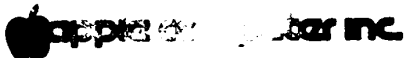
rdy*	3-3a	
reset*	9-2b	
resetlk*	5-1b	RESET LOCK
rffield	10-1b	
rfsh	10-1b	REFRESH
rgb1,rgb8	6-3b	RED, GREEN, BLUE
romsel*	5-3b	ROM SELECT
romsel1	5-1b	
romsel2	5-1b	
rrfsh	10-1b	ROM REFRESH
rtcwrt	10-1c	ROM TIME CHARACTER RAM WRITE
rts	8-1c	REQUEST TO SEND
rsync	10-1c	ROM SYNC
rwpr	5-1b	
rwrprot	10-1b	
r/w	4-3c	READ WRITE
s399	3-3a	
s50/60	10-1b	SELECT 50 HZ/60 HR
sco	5-1b	SERIAL CLOCK
scr	7-2c	SCROLL
scrn	5-1b	SCREEN
sel2m*	5-1b	SELECT 2 MEG
sel374	6-4a	SELECT ORDER OF 374'S to dvx bus
ser	5-1b	SERIAL DATE
shift*	9-3c	SHIFT KEY
spkr*	5-3a	SPEAKER (STROBE)
sum4	3-3d	
sum3	3-3d	
sum2	3-3d	
sum1	3-3d	
synch	10-1b	
tcwrt	10-1b	TIME CHARACTER RAM WRITE
text	5-2a	
tromsel	4-3a	
tromsel*	5-3d	
tsadb*	4-4d	
txd	8-1c	
uselb	3-2c	MICROPROCESSOR SELECT B RAM BUS
v0,v1	10-2c	
v2,v5	10-2b	
va	10-2c	
vb,vc	10-2c	
vbl	10-1c	
we2114	6-4d	WRITE ENABLE CHARACTER RAM
wrdata	7-2d	WRITE DATE (DISK)
wramen	10-4a	WRITE RAM ENABLE
wrprot	7-2d	WRITE PROTECT
wrreq	7-2d	WRITE REQUEST
x0,x7	9-4c	KEY SCAN
y0,y9	9-4c	KEY SCAN
z0,z7	5-1b	ZERO PAGE ADDRESS BITS
zpage*	4-4b	ZERO PAGE



NOTE: UNLESS OTHERWISE SPECIFIED
 1) COMPONENTS WITHOUT I.C. SOCKETS
 2) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 3) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 4) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 5) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 6) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 7) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 8) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 9) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 10) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 11) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 12) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 13) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 14) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 15) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 16) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 17) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 18) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 19) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 20) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 21) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 22) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 23) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 24) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS

NOTE: UNLESS OTHERWISE SPECIFIED
 1) COMPONENTS WITHOUT I.C. SOCKETS
 2) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 3) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 4) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 5) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 6) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 7) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 8) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 9) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 10) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 11) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 12) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 13) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 14) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 15) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 16) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 17) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 18) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 19) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 20) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 21) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 22) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 23) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS
 24) ALL OTHER COMPONENTS TO HAVE APPROPRIATE I.C. SOCKETS

ITEM	QUANTITY	DESCRIPTION
1	1	...
2	1	...
3	1	...
4	1	...
5	1	...
6	1	...
7	1	...
8	1	...
9	1	...
10	1	...
11	1	...
12	1	...
13	1	...
14	1	...
15	1	...
16	1	...
17	1	...
18	1	...
19	1	...
20	1	...
21	1	...
22	1	...
23	1	...
24	1	...



PROM 341-0043

A=A11
B=A12
C=A13
D=A14
E=IOSYNC
F=SEL2M'
G=ABK4
H=PAB
I=A15
J=ZPAGE'
DO=S399
D1=PRDY'
D2=IND'

S399=ZPAGE*PAB'*A15'*A14'*A13'*A12*A11

PRDY'=IOSYNC*SEL2M*ABK4

IND'=(ABK4*(ZPAGE*PAB'))'



PROM 341-0042

A=A13
 B=A11
 C=A15
 D=IND'
 E=DHIRES
 F=ABK1
 G=ABK2
 H=ABK3
 I=A14
 J=AY'
 DO=PCAS4,7'
 D1=PCAS5,6'
 D2=PCAS1'
 D3=PCAS2'

PCAS4,7'=(AY*IND'*ABK3'*A15'*(ABK1*ABK2'+ABK1'*ABK2)*(A14'*A13+A14*A13')+AY*IND*ABK3'*(ABK2'*ABK1'*A15+ABK2'*ABK1+ABK2*ABK1'*A15')*A14'+AY*IND'*ABK1*ABK2*ABK3'*(A15'*A14'*A13+A15*A14'*A13')+AY*IND*ABK3'*ABK2*(A15'*ABK1+A15*ABK1')*(A14'*A13'+A14*A13))'

PCAS5,6'=(AY*IND'*ABK3'*(ABK1*ABK2'+ABK1'*ABK2)*(A15'*A14*A13+A15*A14'*A13')+AY*IND*ABK3'*(ABK2'*ABK1'*A15+ABK2'*ABK1+ABK2*ABK1'*A15')*A14+AY*IND'*ABK1*ABK2*ABK3'*(A15'*A14)+AY*IND*ABK3'*ABK2*(A15'*ABK1+A15*ABK1')*(A14'*A13+A14*A13'))'

PCAS1'=(DHIRES *AY'+AY*(ABK1'*ABK2'*ABK3'*IND'+ABK1*ABK2*ABK3)*(A15'*A14'*A13+A15*A14'*A13')+AY*IND*ABK1'*ABK2'*ABK3'*A15'*(A14'*A13'+A14*A13))'

PCAS2'=(DHIRES *AY'+AY*(ABK1'*ABK2'*ABK3'*IND'+ABK1*ABK2*ABK3)*(A15'*A14)+AY*IND*ABK1'*ABK2'*ABK3'*A15'*(A14'*A13+A14*A13'))'



ROM 341-0032

A=A

B=B

C=S4

D=VBL

E=PAGE2

F=AIISW'

G=HIRES

H=MIX

I=TEXT

J=V2

K=V4

D0=PG2

D1=SEL374

D2=COLRKL'

D3=AHIRES

D4=CH80'

D5=OE374

D6=AIILORES

D7=DHIRES

PG2=AIISW'*(HIRES*MIX'*TEXT ')*HIRES*PAGE2*S4*VBL'

SEL374=(VBL'*(AIISW*(PAGE2'*(TEXT +MIX*V2*V4)'+PAGE2*(TEXT +MIX*V2*V4)))+
AIISW'*(HIRES*(PAGE2*S4)'+HIRES'*(PAGE2*S4))))'

COLRKL'=(AIISW*TEXT +AIISW'*(HIRES'*(MIX+TEXT ')+HIRES*TEXT '))'

AHIRES=AIISW'*(HIRES*MIX*TEXT)

CH80'=(AIISW'*MIX)'

OE374=(AIISW*HIRES'*(TEXT +MIX*V2*V4)'+AIISW'*MIX'*TEXT)'

AIILORES=AIISW*HIRES'*(TEXT +MIX*V2*V4)'+AIISW'*HIRES*MIX*TEXT

DHIRES=(AIISW*HIRES*(TEXT +MIX*V2*V4)'+AIISW'*HIRES)*VBL'



PROM 341-0046 U180

A=C1M
B=C07X'
C=RAMEN
D=DSPLY
E=IOSTOPD'
F=C-FXXX
G=FSPACE'
H=R/WN
I=RWRPROT
J=SEL2M'
DO=PCS6522
D1=PHASEN
D2=WRAMEN

PCS6522=(IOSTOPD'*C1M+FSPACE'*C1M')'

PHASEN=((SEL2M'*C07X')'*IOSTOPD'*FSPACE+FSPACE*C1M'+C1M'*SEL2M'+C1M'*DSPLY*RAMEN)'

WRAMEN=RAMEN*(RWRPROT*C-FXXX)*R/WN*(DSPLY*C1M')'



PROM 341-0056

A=A11
 B=A13
 C=A14
 D=A15
 E=R/WN
 F=DHIRES
 G=AY'
 H=ABK2
 I=PRAS1,2
 J=PRAS0,3
 DO=PCAS0'
 D1=PUSELB
 D2=PCAS0,3
 D3=PCAS3'

PCAS0'=(PRAS0,3 *(DHIRES'*AY'+AY*(A15'*A14'*A13'*A11'*R/WN'+A15'*A14'*A13'*
 R/WN+A15*A14'*A13+A15*A14*A13'*A11)))'

PUSELB =PRAS0,3 *(A15'*A14'*A13'*A11+A15*A14*A13'*A11'+A15*A14*A13)+PRAS0,3 *
 PRAS1,2 *(A15'*A14'*A13+A15*A14'*A13')+PRAS0,3 *PRAS1,2 *(A15'*A14'*A13+A15'*
 A14*A13')+PRAS0,3 '*PRAS1,2 *(A14'*A13'+A14*A13)+PRAS0,3 '*PRAS1,2 '*A14'

PCAS0,3 =(PRAS0,3 *(DHIRES'*AY'+AY*(A15'*A14'*A13'*A11+A15*A14*A13'*A11'+A15*
 A14*A13))+PRAS0,3 *(DHIRES'*AY'+AY*(A15'*A14'*A13'*A11'*R/WN'+A15'*A14'*A13'*
 R/WN+A15*A14'*A13+A15*A14*A13'*A11)))'

PCAS3'=(PRAS0,3 *(DHIRES'*AY'+AY*(A15'*A14'*A13'*A11+A15*A14*A13'*A11'+A15*
 A14*A13)))'



PROM 341-0044 *R4S*

A=ABK1
 B=ABK2
 C=ABK3
 D=PA15
 E=AY'
 F=PA8
 G=ZPAGE'
 H=DHIRES
 I=RFSH
 J=ABK4
 DO=PRAS0,3
 D1=PRAS1,2
 D2=PRAS4,5
 D3=PRAS6,7

PRAS0,3 =AY'*(DHIRES'+RFSH)+((ABK4*(ZPAGE*PA8')')'+ABK1*ABK2*ABK3)*AY

PRAS1,2 =AY'*(DHIRES+RFSH)+AY*(ABK1'*ABK2'*ABK3'*(ABK4*(ZPAGE*PA8')'*PA15)'+
 ABK1*ABK2*ABK3)+AY*ABK3'*(ABK1'*ABK2*ABK4*(ZPAGE*PA8')'*PA15+ABK1*ABK2*(ABK4*(
 ZPAGE*PA8')'*PA15)')

PRAS4,5 =RFSH*AY'+AY*ABK2'*ABK3'*(ABK1'*ABK4*(ZPAGE*PA8')'*PA15+ABK1*(ABK4*(
 ZPAGE*PA8')'*PA15)')

PRAS6,7 =RFSH*AY'+AY*ABK3'*(ABK1*ABK2'*ABK4*(ZPAGE*PA8')'*PA15+ABK1'*ABK2*(
 ABK4*(ZPAGE*PA8')'*PA15)')



PROM 341-0045 U176

A=C5XXN
B=R/WN
C=C6XXN
D=INH'
E=ROMSEL'
F=DMAI'
G=FSPACE'
H=PH2M
I=C7XXN
J=CXXX
D0=EN257
D1=EN8304
D3=RAMEN

EN257=(PH2M*R/WN*ROMSEL'*INH'*FSPACE'*(C5XXN *C6XXN*C7XXN*CXXX)')'

EN8304=(PH2M*FSPACE'*ROMSEL'*DMAI')'

RAMEN=ROMSEL'*INH'*FSPACE'*(C5XXN *C6XXN*C7XXN*CXXX)'

Apple Computer Inc.

PROM 341-0055 *U175.1*

A=VA
 B=VB
 C=VC
 D=VA1
 E=VB1
 F=VC1
 G=DHIRES
 H=SCR
 I=WE2114'
 J=VBL
 DO=MUX1
 D1=MUX2
 D2=MUX3
 D3=ENHREG'

$MUX1 = ((DHIRES' + SCR * (VA * VA1' + VA' * VA1) + SCR' * VA) * VBL' + VBL)' + VBL * WE2114' * SCR * VC1$

$MUX2 = (DHIRES * (SCR * (VA * VA1 * (VB * VB1' + VB' * VB1)' + (VA * VA1)' * (VB * VB1' + VB' * VB1))) + VB * SCR')'$

$MUX3 = DHIRES * (SCR * ((VA * VA1 * (VB + VB1) + VB * VB1) * (VC * VC1' + VC' * VC1)' + (VA * VA1 * (VB + VB1) + VB * VB1)' * (VC * VC1' + VC' * VC1)) + VC * SCR')$

$ENHREG' = DHIRES' * WE2114'$



ASCII Conversion Tables

ASCII Conversion Tables

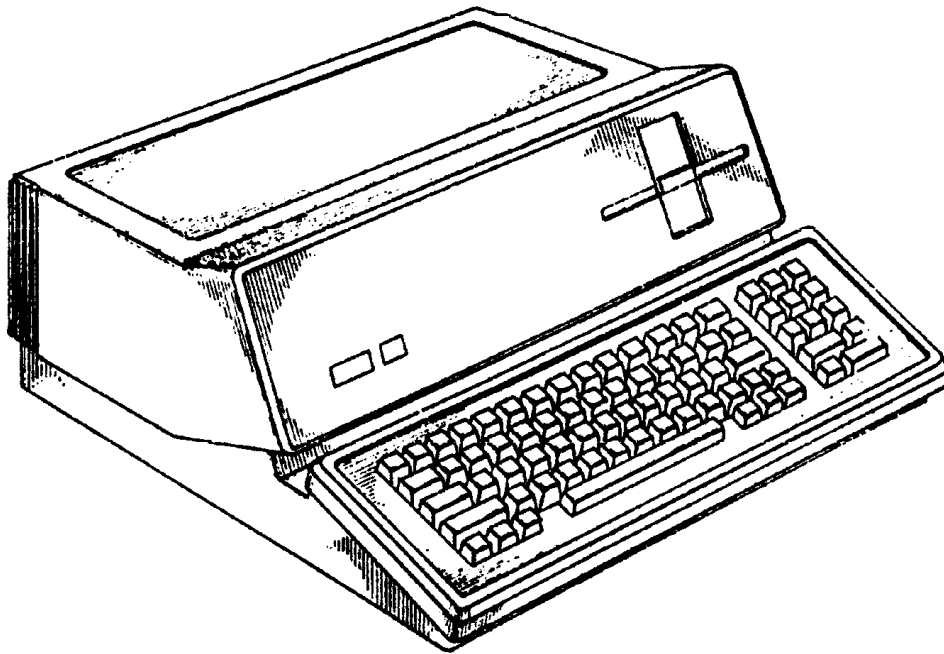
DEC	ASCII	OCTAL	HEX	BINARY	DEC	ASCII	OCTAL	HEX	BINARY
				76543210					76543210
0	NUL	000	00	00000000	64	@	100	40	01000000
1	SOH	001	01	00000001	65	A	101	41	01000001
2	STX	002	02	00000010	66	B	102	42	01000010
3	ETX	003	03	00000011	67	C	103	43	01000011
4	EOT	004	04	00000100	68	D	104	44	01000100
5	ENQ	005	05	00000101	69	E	105	45	01000101
6	ACK	006	06	00000110	70	F	106	46	01000110
7	BEL	007	07	00000111	71	G	107	47	01000111
8	BS	010	08	00001000	72	H	110	48	01001000
9	HT	011	09	00001001	73	I	111	49	01001001
10	LF	012	0A	00001010	74	J	112	4A	01001010
11	VT	013	0B	00001011	75	K	113	4B	01001011
12	FF	014	0C	00001100	76	L	114	4C	01001100
13	CR	015	0D	00001101	77	M	115	4D	01001101
14	SO	016	0E	00001110	78	N	116	4E	01001110
15	SI	017	0F	00001111	79	O	117	4F	01001111
16	DLE	020	10	00010000	80	P	120	50	01010000
17	DC1	021	11	00010001	81	Q	121	51	01010001
18	DC2	022	12	00010010	82	R	122	52	01010010
19	DC3	023	13	00010011	83	S	123	53	01010011
20	DC4	024	14	00010100	84	T	124	54	01010100
21	NAK	025	15	00010101	85	U	125	55	01010101
22	SYN	026	16	00010110	86	V	126	56	01010110
23	ETB	027	17	00010111	87	W	127	57	01010111
24	CAN	030	18	00011000	88	X	130	58	01011000
25	EM	031	19	00011001	89	Y	131	59	01011001
26	SUB	032	1A	00011010	90	Z	132	5A	01011010
27	ESC	033	1B	00011011	91	[133	5B	01011011
28	FS	034	1C	00011100	92		134	5C	01011100
29	GS	035	1D	00011101	93]	135	5D	01011101
30	RS	036	1E	00011110	94		136	5E	01011110
31	US	037	1F	00011111	95	_	137	5F	01011111



<u>DEC</u>	<u>ASCII</u>	<u>OCTAL</u>	<u>HEX</u>	<u>BINARY</u>	<u>DEC</u>	<u>ASCII</u>	<u>OCTAL</u>	<u>HEX</u>	<u>BINARY</u>
32	SP	040	20	00100000	96	,	140	60	01100000
33	!	041	21	00100001	97	a	141	61	01100001
34	"	042	22	00100010	98	b	142	62	01100010
35	#	043	23	00100011	99	c	143	63	01100011
36	\$	044	24	00100100	100	d	144	64	01100100
37	%	045	25	00100101	101	e	145	65	01100101
38	&	046	26	00100110	102	f	146	66	01100110
39	'	047	27	00100111	103	g	147	67	01100111
40	(050	28	00101000	104	h	150	68	01101000
41)	051	29	00101001	105	i	151	69	01101001
42	*	052	2A	00101010	106	j	152	6A	01101010
43	+	053	2B	00101011	107	k	153	6B	01101011
44	,	054	2C	00101100	108	l	154	6C	01101100
45	-	055	2D	00101101	109	m	155	6D	01101101
46	.	056	2E	00101110	110	n	156	6E	01101110
47	/	057	2F	00101111	111	o	157	6F	01101111
48	0	060	30	00110000	112	p	160	70	01110000
49	1	061	31	00110001	113	q	161	71	01110001
50	2	062	32	00110010	114	r	162	72	01110010
51	3	063	33	00110011	115	s	163	73	01110011
52	4	064	34	00110100	116	t	164	74	01110100
53	5	065	35	00110101	117	u	165	75	01110101
54	6	066	36	00110110	118	v	166	76	01110110
55	7	067	37	00110111	119	w	167	77	01110111
56	8	070	38	00111000	120	x	170	78	01111000
57	9	071	39	00111001	121	y	171	79	01111001
58	:	072	3A	00111010	122	z	172	7A	01111010
59	;	073	3B	00111011	123		173	7B	01111011
60	<	074	3C	00111100	124		174	7C	01111100
61	=	075	3D	00111101	125		175	7D	01111101
62	>	076	3E	00111110	126		176	7E	01111110
63	?	077	3F	00111111	127	DEL	177	7F	01111111



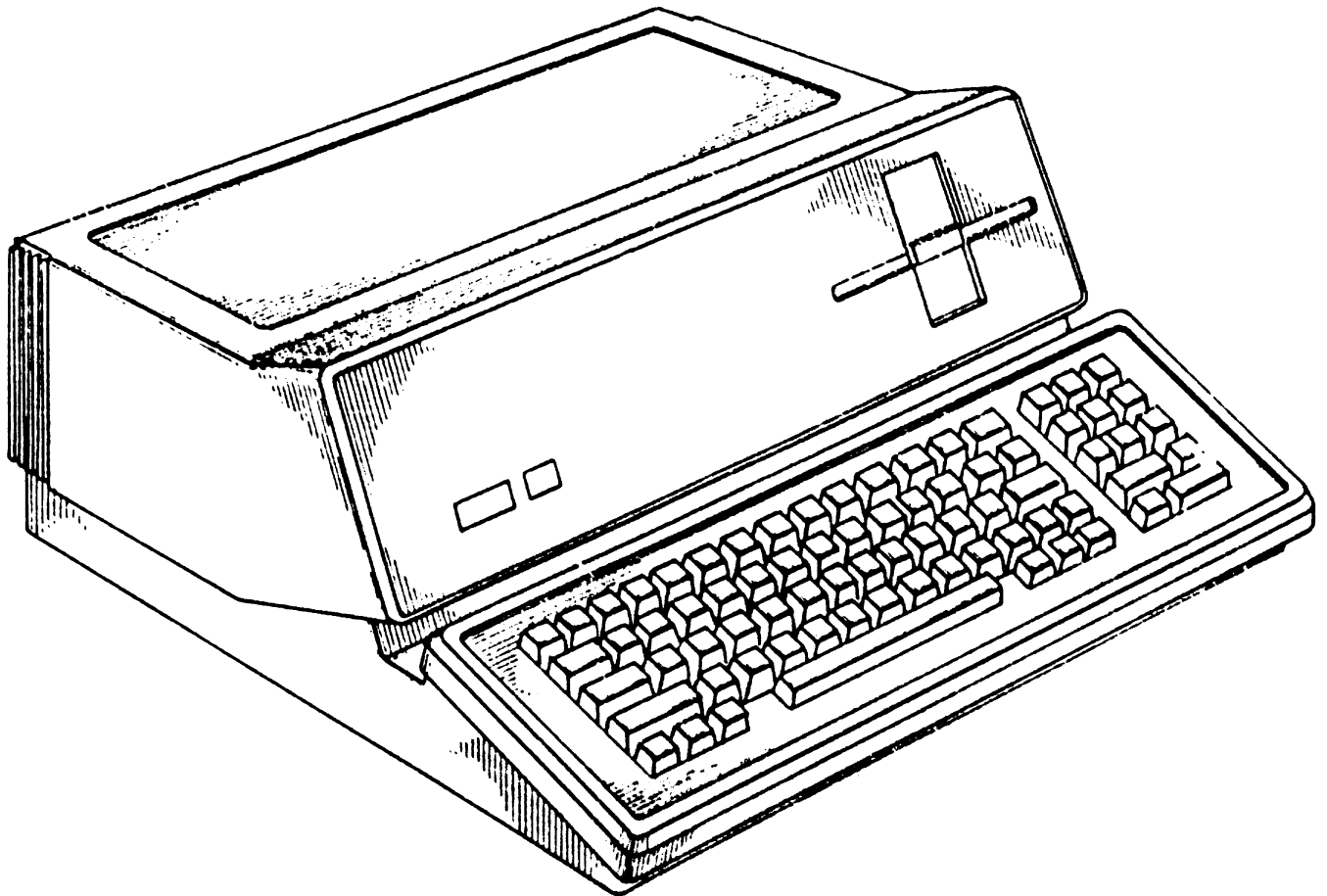
Apple /// Computer Information



APPLE /// PICTURES

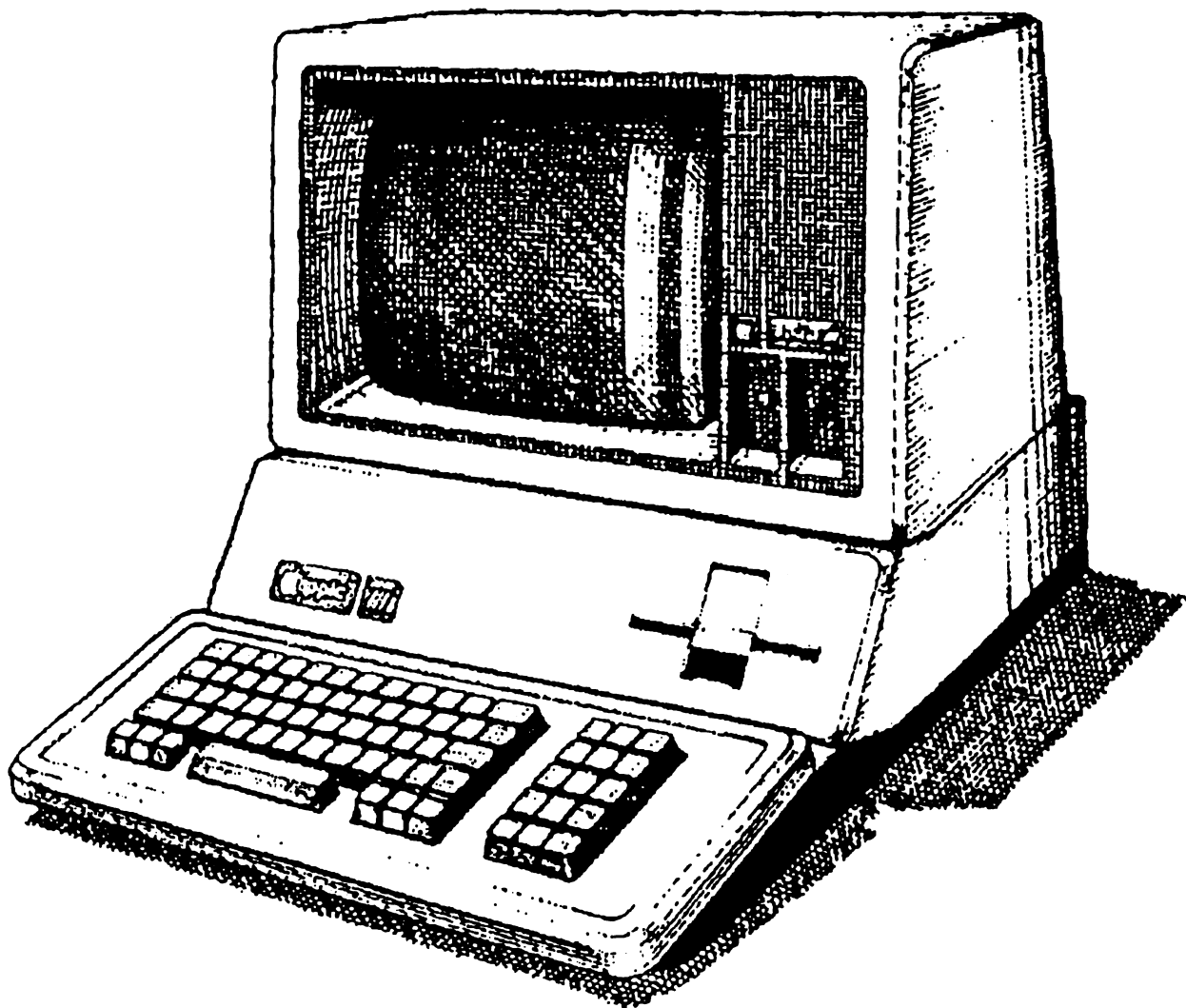
ADDED BY DAVID T CRAIG • 2006





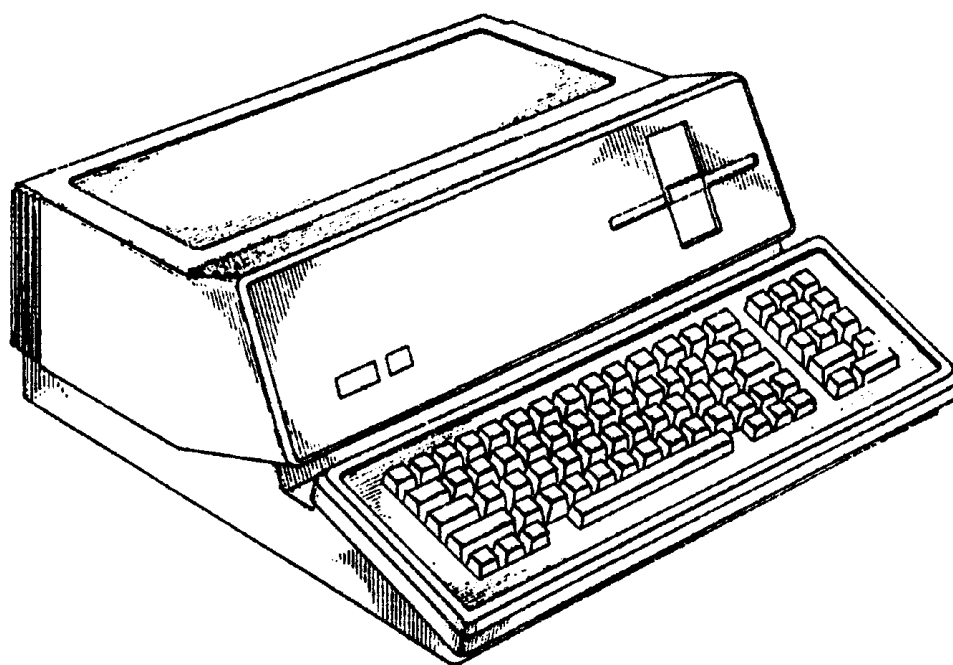


Apple III Computer





Apple /// Computer Information

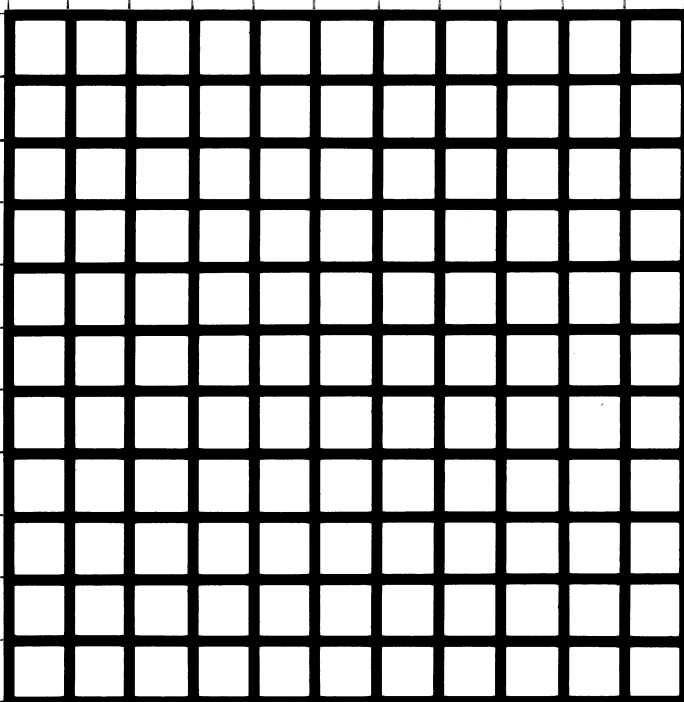


APPLE /// SYSTEM DATA SHEET

ADDED BY DAVID T CRAIG • 2006

Apple ///

System Data Sheet



EX LIBRIS: David T. Craig
736 Edgewater
[# _____] Wichita, Kansas 67230 (USA)

The Apple ///

The Most Powerful Personal Computer In Its Class

Too much information? Not enough time? The Apple /// was created to meet the information-handling needs of decision makers at all levels, in every size and kind of company. And the Apple /// can grow with you, so as your responsibilities increase, your ability to handle them stays one step ahead.

You can use the power of your Apple /// to create financial forecasts, budgets, and reports; for accounting, resource management, and project scheduling; in electronic communications, software development, and computer-assisted training. Over 400 business programs are available today for the Apple /// — plus the extensive library of CP/M® business software (with the Apple SoftCard™ ///). And most Apple II Plus programs will run in the Apple ///'s "emulation" mode.

The Apple ///: the personal computer for business.



Powerful features for professional needs.

The Apple /// is ready to go as soon as you unpack it, connect a monitor, and provide power. No interface cards are required, and you don't have to open the computer. The Apple /// already has a built-in disk drive, video outputs for color and monochromatic displays, and a numeric keypad.

Other built-in features include:

Large User Memory. The Apple ///'s 256K of internal memory means you can work with sophisticated programs and large financial and text documents, quickly and efficiently.

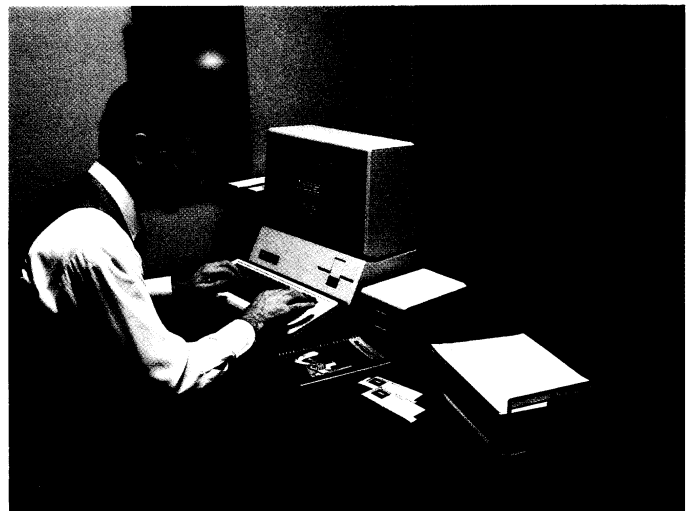
Color Graphics. The 16-color graphics capability of the Apple /// allows you to grasp the meaning of charts and graphs quickly. If you're not using a color monitor, your information is displayed in 16 shades, so the facts still stand out clearly.

High-Resolution Video. The Apple /// displays 107, 520 points of information on the screen (560 horizontal x 192 vertical) in text and monochromatic graphics modes. While text is normally presented in an 80-column by 24-line monochromatic format, it can be switched to 40-column monochromatic or color-on-color.

Accessory Connectors. The most common accessories plug right into the Apple ///. Connectors and interfacing hardware are already built in for the Apple Daisy Wheel Printer (or other serial printer), the Apple Silentype Printer, external floppy disk drives, color and monochromatic video displays (NTSC, RGB, and composite), a modem, and hand controls. The Apple /// also has four inside expansion slots for additional accessories.

Apple /// Sophisticated Operating System: it does it all for you.

Today . . . you can bring financial models into reports, insert names into form letters automatically, and turn numbers into charts, because the Apple ///'s Sophisticated Operating System (SOS) treats all your files identically. And, since applications programs written for the Apple /// are all based on this common SOS formatting, you can combine them on a ProFile™ mass storage system and move freely from one to another. The uniformity of SOS also provides an ideal environment for software development.



Tomorrow . . . you can expand your Apple /// elegantly. Because SOS controls all communications with accessories, you don't have to figure out how to make the computer work with a new printer, disk drive, or modem. SOS does this for you by using special files known as "device drivers." Apple /// programs come with the most commonly-used device drivers, and you can make programs compatible with new equipment by copying a driver file for the new device onto a program disk. Your software can just as easily be revised to take advantage of SOS upgrades, and of hardware enhancements to the computer itself.

Installation's easy. Learning is, too.

Because the Apple /// already has a built-in disk drive and video connector, the computer is ready to work as soon as you connect a monitor and provide power. Then, Apple makes it just as easy to learn how to use it. A comprehensive Owner's Guide gets you started, and a System Demonstration disk introduces you to the computer's text editing and graphics capabilities. Reference manuals and SOS utilities disks are included for more advanced needs, and additional tutorials on the computer and its programs are also available.



Durable. Dependable. Reliable.

The Apple /// is dependable, inside and out. Outside, it has a rugged die-cast aluminum chassis. Inside, electronics based on advanced microprocessor circuitry assure reliable operation. The system also meets UL and CSA standards.

Every time the computer is powered up, it performs a brief self-diagnostic routine. Should problems arise, help is close at hand, because of Apple's extensive dealer/service network. Average turnaround time on Apple /// servicing is less than one day.



Standard Features

- 256K internal memory (RAM)
- Built-in disk drive
- Custom microprocessor circuitry
- High-resolution color graphics (16 colors)
- 80-column, 24-line text display, upper and lower case
- Contoured typewriter-style keyboard; 61 keys; all 128 ASCII codes; auto-repeat on all keys
- Numeric keypad (13 keys)
- Special-purpose keys: Up-Arrow, Down-Arrow, Left-Arrow, Right-Arrow; programmable Open-Apple and Solid-Apple; TAB; SHIFT; ALPHA LOCK; CONTROL; RETURN; ENTER; ESCAPE
- Quick-connect plugs for disk drives, video and audio devices, serial printers, modems, and hand controls
- Four expansion slots for accessory interface cards
- Apple // Plus emulation mode
- High-quality sound generation
- Lockable case
- Self-testing diagnostics on powerup

Optional Accessories

- Monitor /// or color monitor
- Apple Daisy Wheel Printer
- Apple Dot Matrix Printer
- Apple Silentype Printer
- Disk /// floppy-disk drives
- ProFile hard-disk systems
- Apple SoftCard /// System (for CP/M capability)
- Parallel Card ///
- Serial Card ///
- Programming languages (Business BASIC, Pascal, COBOL)
- Cursor /// joysticks

Technical Specifications

■ **Video Display:**

Text and graphics may be displayed simultaneously. Graphics modes:
 —280 x 192, 16 colors (with some limitations);
 —280 x 192, monochromatic;
 —140 x 192, 16 colors;
 —560 x 192, monochromatic;
 —All Apple II modes (in emulation)

Graphics commands allow either of two screen buffers to be displayed.

Text modes:

- 80-column, 24-line monochromatic;
- 40-column, 24-line, 16-color foreground and background;
- 40-column, 24-line monochromatic.

All text modes have a software-definable, 128-character set (upper and lower-case), with normal or inverse display.

■ **Central Processing Unit (CPU):**

The custom-designed microprocessor circuitry of the Apple III utilizes the 6502B as one of its major components. Other circuitry provides extended addressing capability, relocatable stack, zero page, and memory mapping.

Type:

6502B.

Clock Speed:

1.4 MHz average; 1.8 MHz maximum.

Operations Per Second (8-bit):

Up to 750,000.

Data Bus:

Two 8-bit formats, combined for extended addressing.

Address Bus:

19 bits.

Address Range:

262,144 bytes (256K).

Registers:

Accumulator (A); Index Registers (X,Y); Stack Pointer (S); Program Counter (PC); Environmental Register (E); Bank (B); Zero Page (Z); Processor Status (P).

■ **Memory:**

256K dynamic RAM;
 4K ROM (initialization and self-test diagnostics).

■ **SOS (Sophisticated Operating System):**

Handles all system I/O;
 Can be configured to handle standard or custom I/O devices and peripherals by adding or deleting "device drivers";
 All languages and application programs access data through the SOS file system.

■ **Inputs and Outputs:**

Keyboard:

- 61 keys on main keyboard;
- 13 keys on numeric keypad;
- Full 128-character, ASCII encoded;
- All keys have automatic repeat;
- Four directional-arrow keys with two-speed repeat;
- Two user-definable Apple keys;
- Seven other special keys: SHIFT, CONTROL, ALPHA LOCK, TAB, ESCAPE, RETURN, ENTER.

Storage Devices:

- One 5.25-inch floppy disk drive built in, 140K (143, 360) bytes per diskette;
- Three additional drives can be connected by daisy-chain cable (Total: 560K bytes on-line storage);
- Up to four ProFile hard-disk drives (5 megabytes each) may be added with plug-in interface cards.

Video Output:

- RCA phono connector for NTSC monochromatic composite video;
- DB-15 connector for:
 NTSC color composite video;
 NTSC monochromatic composite video;
 RGB color video;
 Composite sync signal;
 Power supply voltages.
- Color signals appear as 16-level grey scale on monochromatic displays.

Audio Output:

- Built-in two-inch speaker; miniature phono jack on back panel;
- Driven by 6-bit D/A converter or fixed-frequency "beep" generator.

Serial (Printer/Modem) Port:

- RS-232C compatible, DB-25 female connector;
- Software-selectable baud rate and duplex mode.

One port may be used for the Silentype printer.

One port may be used for the Silentype printer.

Expansion:

- Four 50-pin expansion slots (fully buffered, with interrupt and DMA priority structure).

Joystick/Silentype Ports:

- Two DB-9 connectors.

■ **Languages Available:**

Apple Business BASIC, Apple II Pascal, Apple III COBOL.

■ **Emulation Mode:**

Provides hardware emulation of 48K byte Apple II Plus. Allows most Apple II programs, with the exception of Pascal and FORTRAN, to run without modification.

■ **Electrical Specifications:**

The Apple III's power cord should be plugged into a three-wire 110-120 volt outlet.

■ **Physical Specifications:**

- Height: 4.8 inches (12.20 cm)
- Depth: 18.2 inches (46.22 cm)
- Width: 17.5 inches (44.45 cm)
- Weight: 26 lbs. (11.8 kg)

The Apple III meets the following agency regulations:

- UL 114 — Office Appliances and Business Equipment.
- CSA 22.2, No. 154 — Data Processing Equipment.

The Apple III Personal Computer System Package

U.S. Order Number A3S0256

With your order for an Apple III

System you will receive:

- 256K Apple III;
- Power cord;
- Monitor cable;
- System Demonstration disk;
- System Utilities disk;
- System Utilities Data disk (contains device driver files, character sets, and keyboard layouts);
- Apple II Plus Emulation disk;
- Owner's Guide;
- Standard Device Drivers Manual;
- Warranty and service information.

Specifications or products may change without notice.

Apple, the Apple logo, ProFile, and Silentype are trademarks of Apple Computer, Inc. SoftCard is a trademark of MicroSoft Corporation.

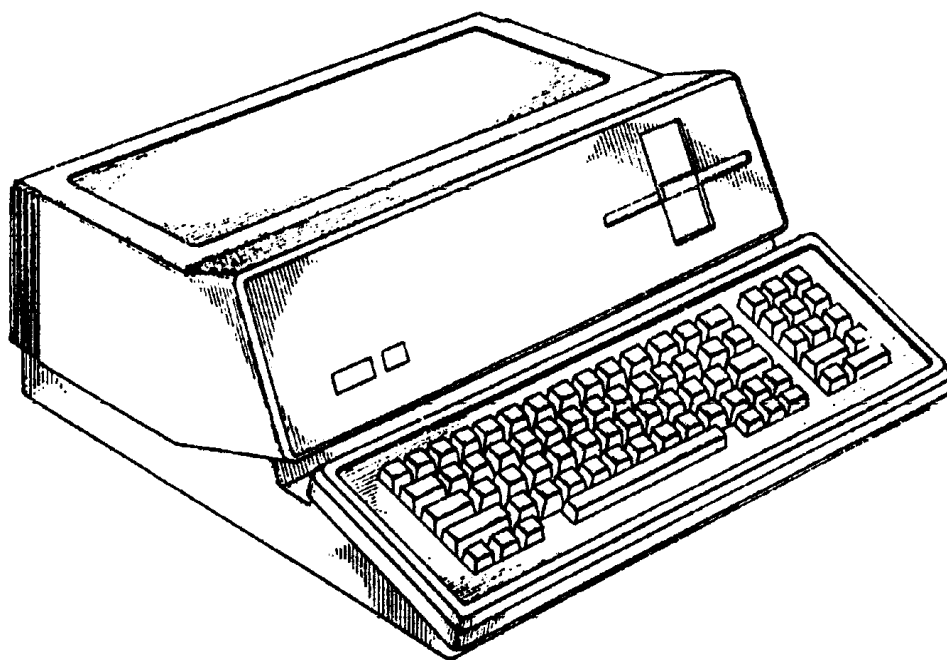
CP/M is a trademark of Digital Research, Inc.



20525 Mariani Avenue
 Cupertino, California 95014
 (408) 996-1010
 TLX 171-576



Apple /// Computer Information



APPLE /// SYSTEM DIAGRAM

ADDED BY DAVID T CRAIG • 2006

Apple III computer system diagram

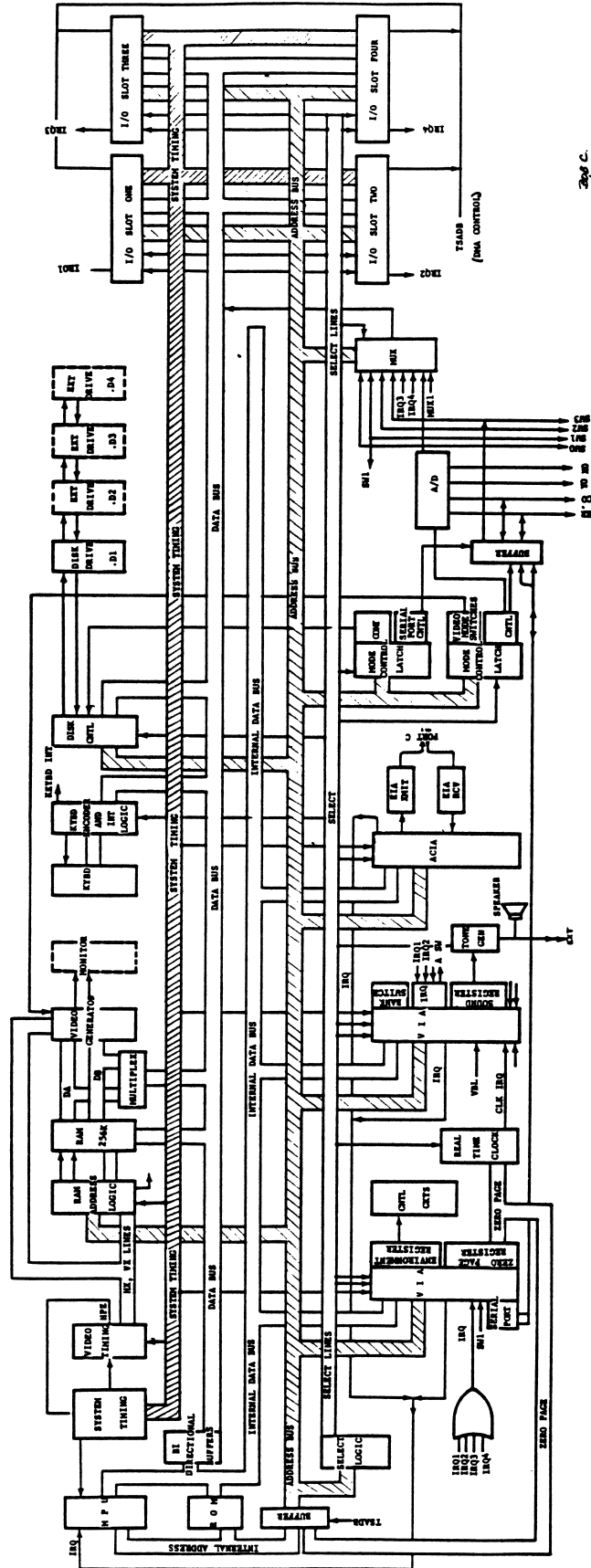
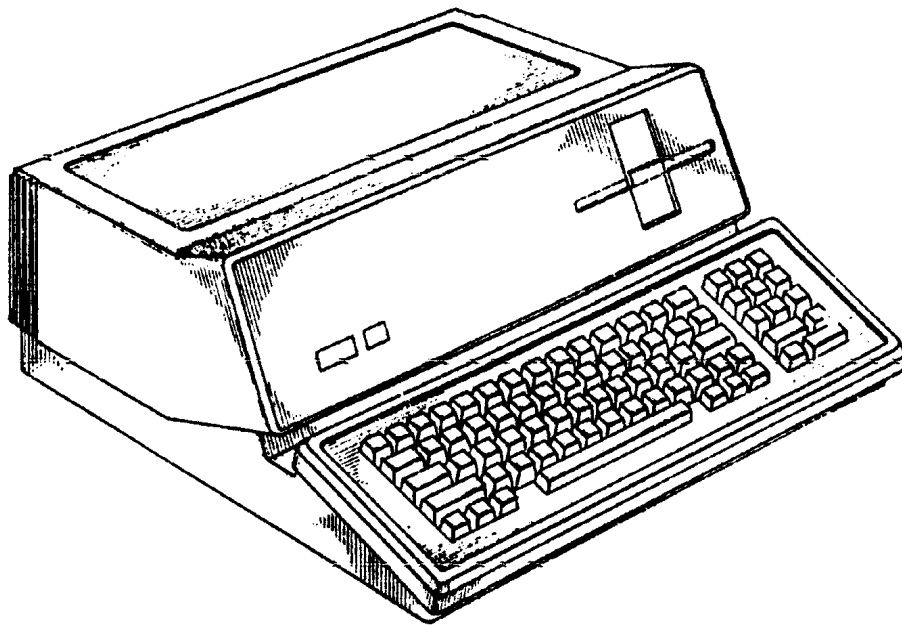


Fig. C.
Apple



Apple /// Computer Information



DESIGN PATENT

Patent # 268,584 -- Apple /// Computer

ADDED BY DAVID T CRAIG • 2006

United States Patent [19]

[11] **Des. 268,584**

Jobs et al.

[45] **** Apr. 12, 1983**

[54] **PERSONAL COMPUTER**

[56]

References Cited

[75] Inventors: **Steven P. Jobs**, Los Gatos; **Jerrold C. Manock**, Palo Alto; **Dean A. Hovey**, Los Altos; **David M. Kelley**, Palo Alto, all of Calif.

U.S. PATENT DOCUMENTS

D. 218,933 10/1970 Cook D14/106
 D. 229,945 1/1974 Santulli D14/106
 D. 252,086 6/1979 Calverly D14/106

[73] Assignee: **Apple Computer, Inc.**, Cupertino, Calif.

Primary Examiner—Susan J. Lucas
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[**] Term: **14 Years**

[57]

CLAIM

The ornamental design for a personal computer, substantially as shown.

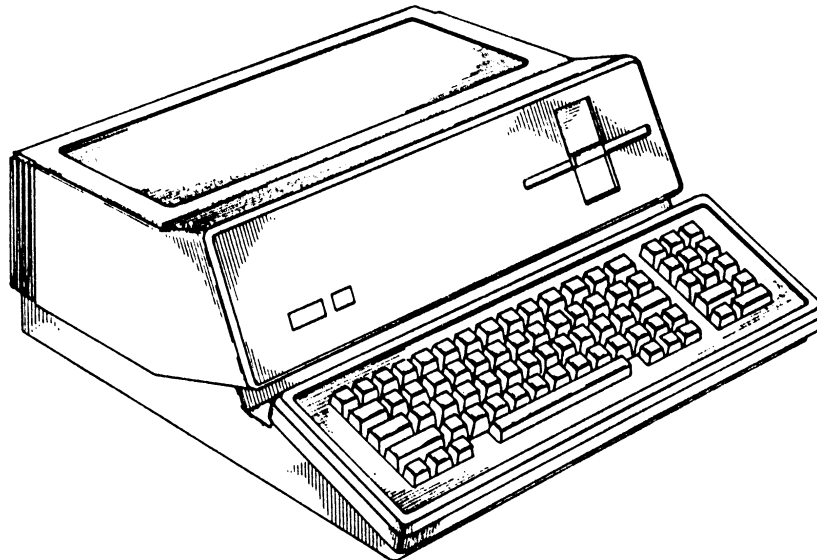
[21] Appl. No.: **203,502**

DESCRIPTION

[22] Filed: **Nov. 3, 1980**

FIG. 1 is a perspective view of the personal computer showing our new design;
 FIG. 2 is a top view thereof;
 FIG. 3 is a front elevational view thereof;
 FIG. 4 is a right side view thereof;
 FIG. 5 is a left side view thereof;
 FIG. 6 is a rear elevational view thereof; and,
 FIG. 7 is a bottom view thereof.

[51] Int. Cl. D14-02
 [52] U.S. Cl. D14/106
 [58] Field of Search D14/100, 101, 102, 103, D14/105, 106, 107, 111, 113, 114; 364/419, 708, 709, 900; 340/365 R; D18/7



Apple /// Computer

U.S. Patent

Apr. 12, 1983

Sheet 1 of 3

Des. 268,584

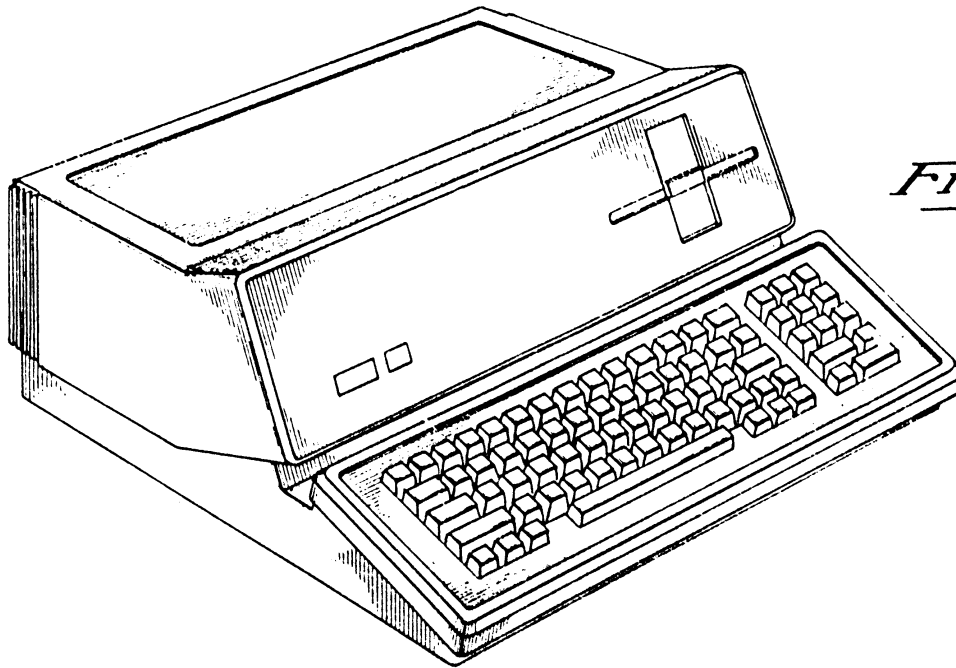


Fig. 1

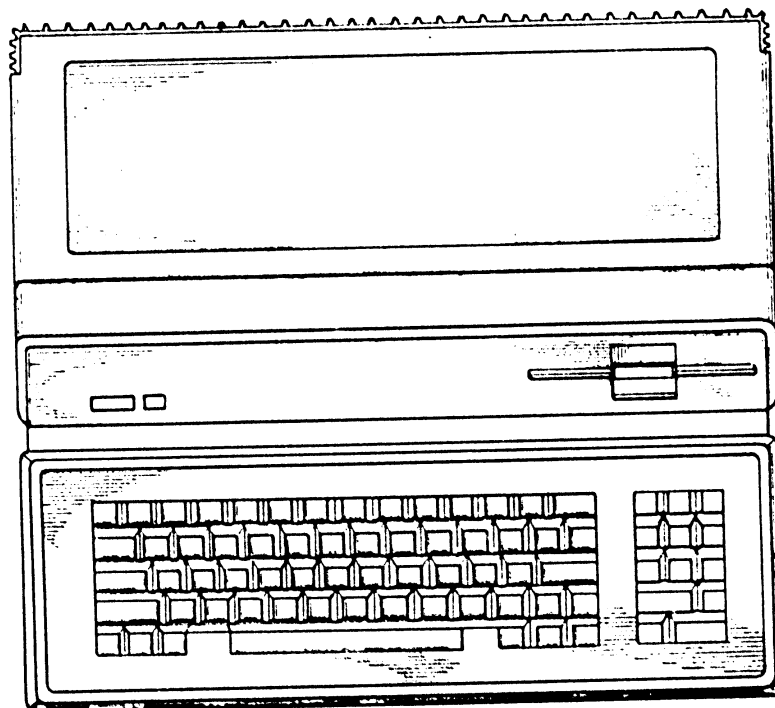


Fig. 2

U.S. Patent

Apr. 12, 1983

Sheet 2 of 3

Des. 268,584

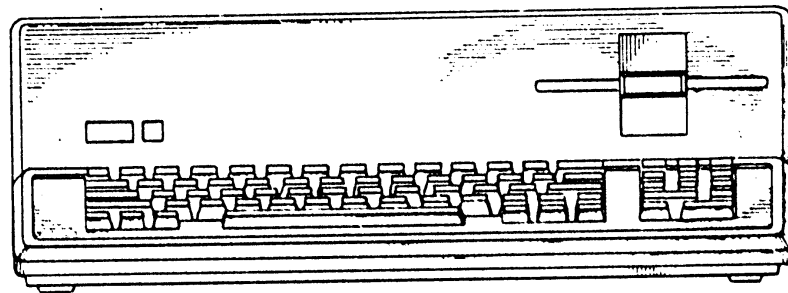


Fig. 3

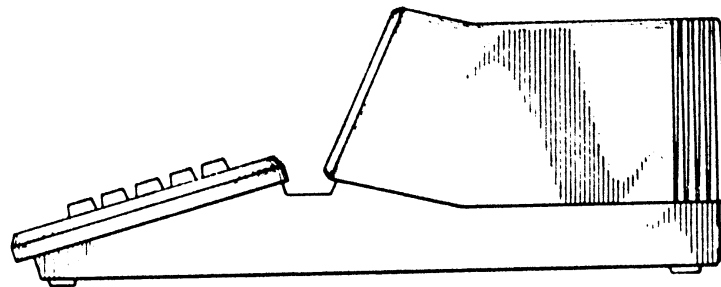


Fig. 4

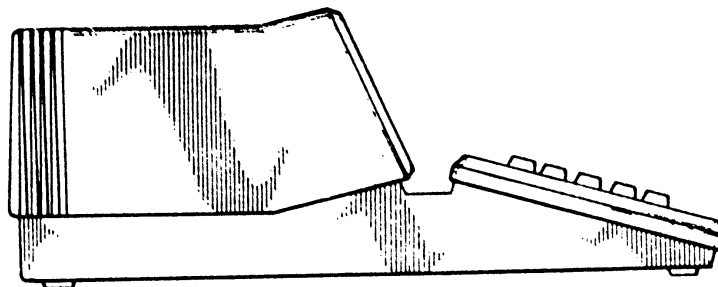


Fig. 5

U.S. Patent Apr. 12, 1983 Sheet 3 of 3 Des. 268,584

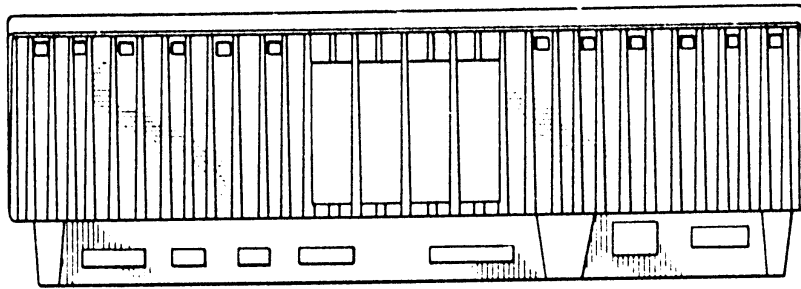


Fig. 6

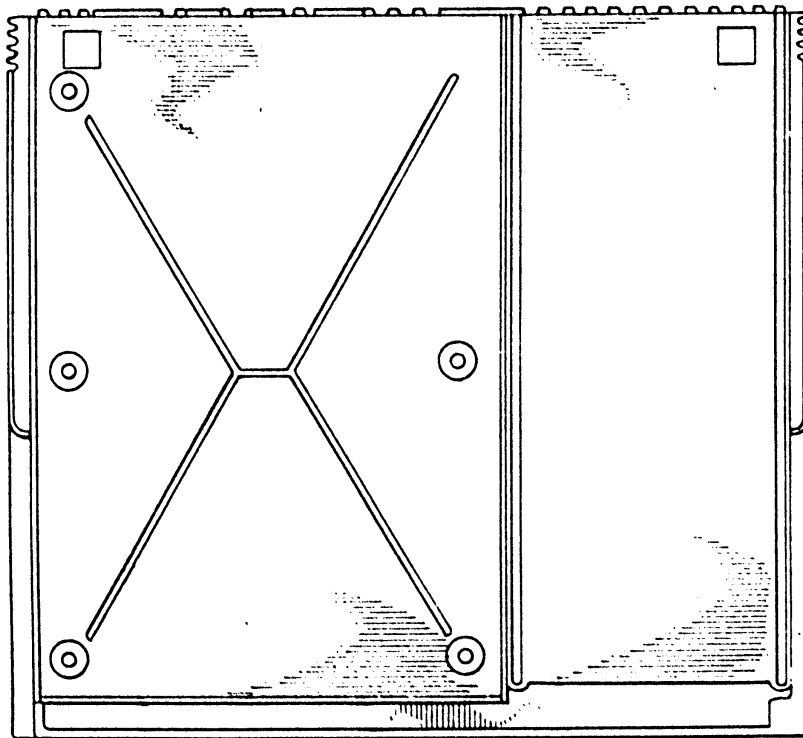
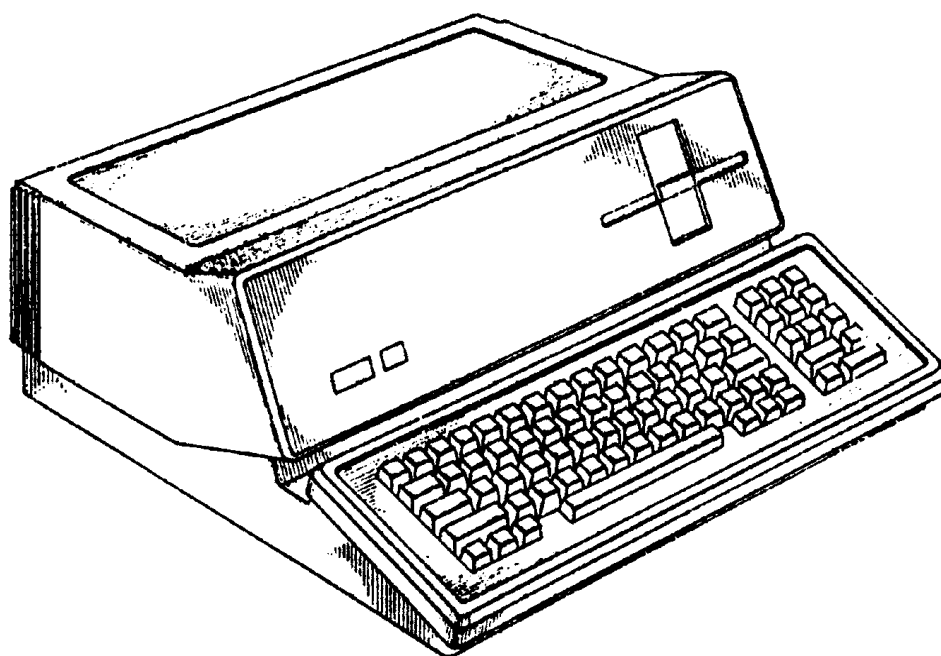


Fig. 7



Apple /// Computer Information



DESIGN PATENT

Patent # 273,191 -- Monitor ///

ADDED BY DAVID T CRAIG • 2006

United States Patent [19]
Oyama et al.

[11] **Des. 273,191**
[45] ** **Mar. 27, 1984**

[54] **COMPUTER DATA DISPLAY MONITOR**

[75] Inventors: **Terrell A. Oyama, San Jose; Loren D. Stirling, Pleasanton, both of Calif.**

[73] Assignee: **Apple Computer, Inc., Cupertino, Calif.**

[**] Term: **14 Years**

[21] Appl. No.: **321,235**

[22] Filed: **Nov. 13, 1981**

[52] U.S. Cl. **D14/113**

[58] Field of Search **D14/100-117;**
340/365 R, 700, 711, 720, 750

[56] **References Cited**

U.S. PATENT DOCUMENTS

D. 250,020 10/1978 **Pycha** **D14/113**

OTHER PUBLICATIONS

Computer Design, 2/80, p. 17, Hitachi, Ltd., Color Monitor.

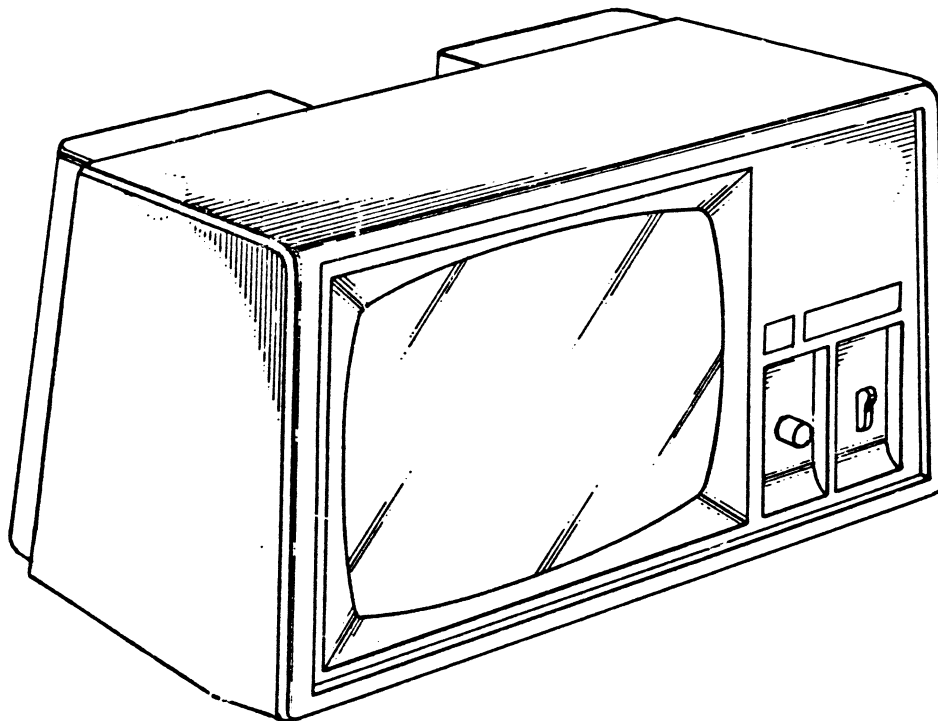
Primary Examiner—Susan J. Lucas
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **CLAIM**

The ornamental design for a computer data display monitor, substantially as shown and described.

DESCRIPTION

FIG. 1 is a perspective view of a computer data display monitor showing our new design; FIG. 2 is a front view thereof; FIG. 3 is a top view thereof; FIG. 4 is a right side view thereof; FIG. 5 is a rear view thereof; and, FIG. 6 is a bottom view thereof.



Apple 3 "Monitor III"

U.S. Patent

Mar. 27, 1984

Sheet 1 of 2

Des. 273,191

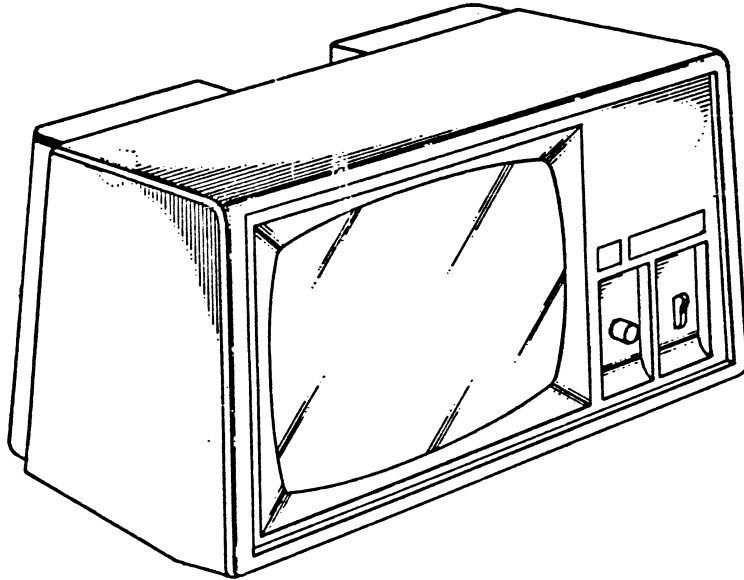


Fig. 1

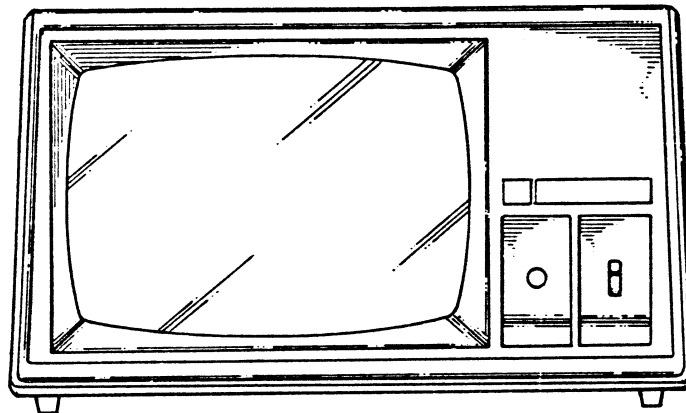


Fig. 2

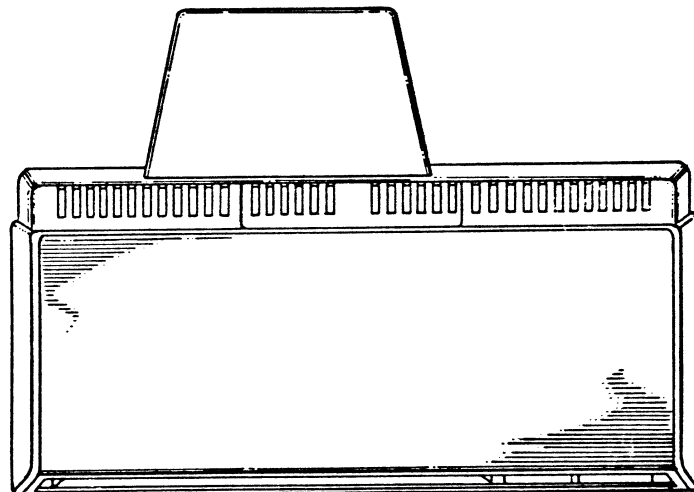


Fig. 3

U.S. Patent

Mar. 27, 1984

Sheet 2 of 2

Des. 273,191

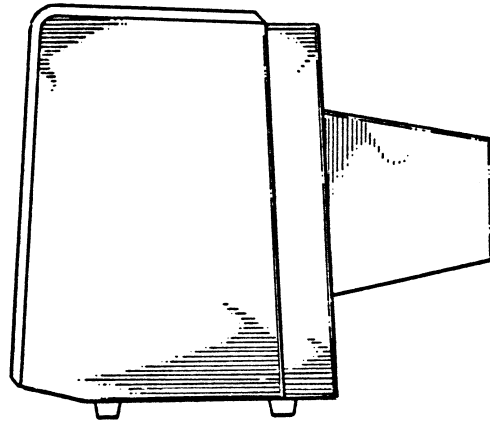


Fig. 4

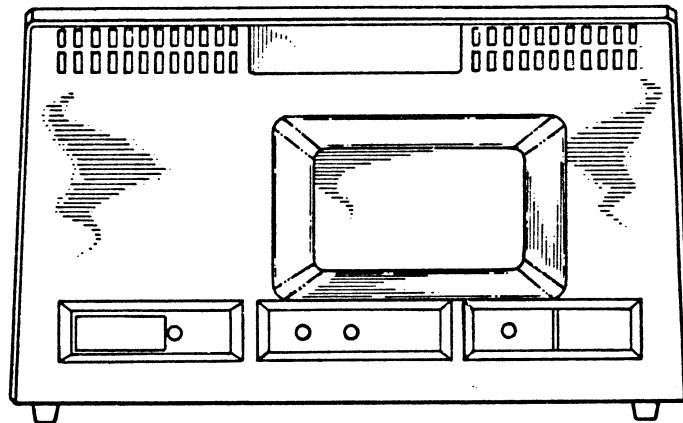


Fig. 5

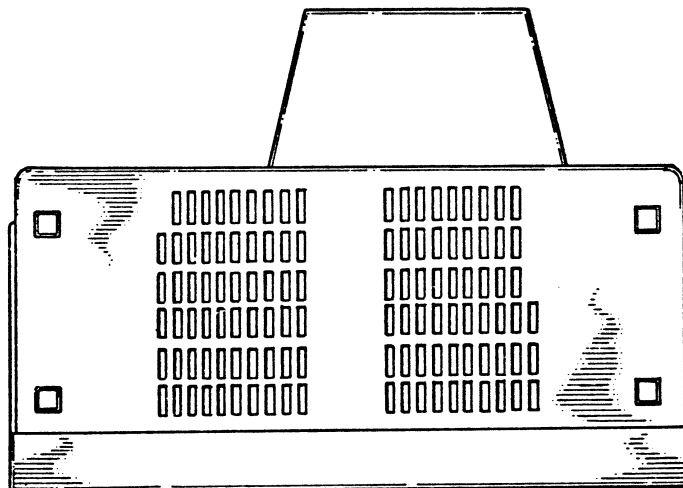
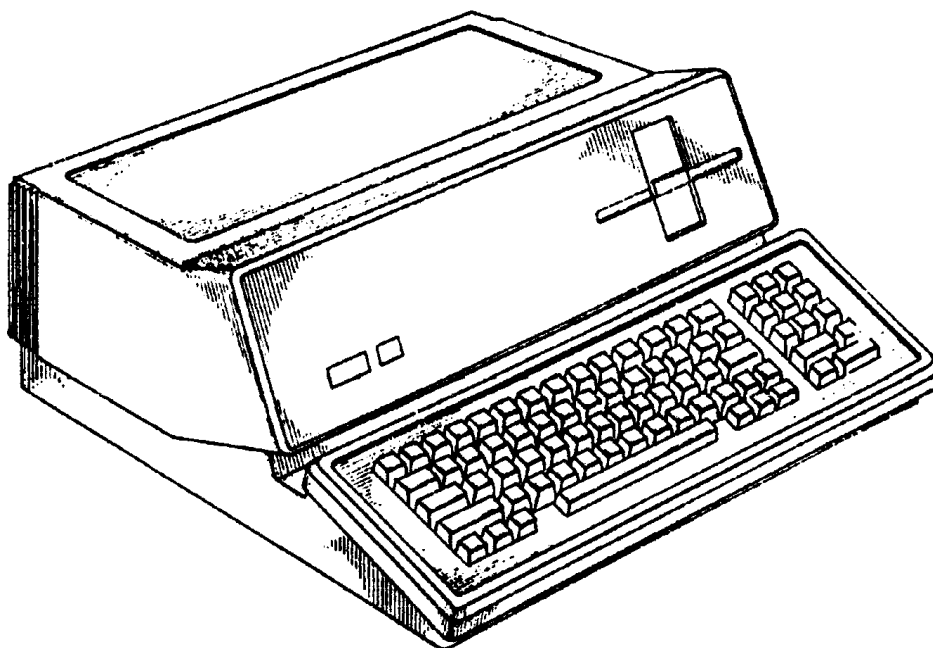


Fig. 6



Apple /// Computer Information



DESIGN PATENT

Patent # 273,295 -- Profile Hard Drive

ADDED BY DAVID T CRAIG • 2006

United States Patent [19]
Stewart

[11] **Des. 273,295**

[45] **** Apr. 3, 1984**

[54] **HARD DISK DRIVE**

[75] **Inventor:** James R. Stewart, San Jose, Calif.

[73] **Assignee:** Apple Computer, Inc., Cupertino, Calif.

[**] **Term:** 14 Years

[21] **Appl. No.:** 322,922

[22] **Filed:** Nov. 19, 1981

[52] **U.S. Cl.** D14/109

[58] **Field of Search** D14/100-116;
D13/41; 361/380, 390, 394; 360/97, 98, 133;
364/900

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,789,273 1/1974 O'Brien 360/97 X
3,899,794 8/1975 Brown, Jr. D14/109 X

OTHER PUBLICATIONS

North Star Computers, Inc., Catalog, 5-1980, p. 8, Hard Disk Drive.

Radio Shack TRS-80™ Microcomputer Catalog RSC-3, ©1979, p. 10, VOXBOX™ Housing.

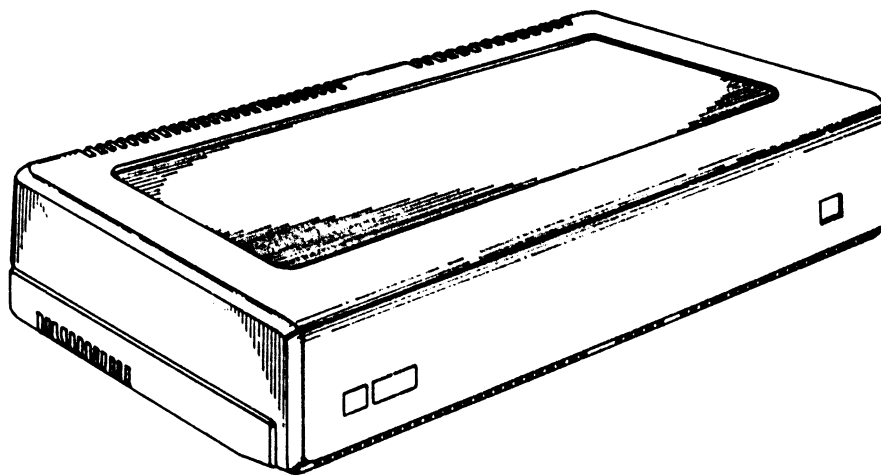
Primary Examiner—Susan J. Lucas
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **CLAIM**

The ornamental design for a hard disk drive, substantially as shown and described.

DESCRIPTION

FIG. 1 is a perspective view of the hard disk drive showing my new design;
FIG. 2 is a top view thereof;
FIG. 3 is a front view thereof;
FIG. 4 is a rear view thereof;
FIG. 5 is a right side view thereof; and
FIG. 6 is a left side view thereof.



Apple "ProFile" hard disk
(for Apple 3 and Lisa)

U.S. Patent

Apr. 3, 1984

Sheet 1 of 2

Des. 273,295

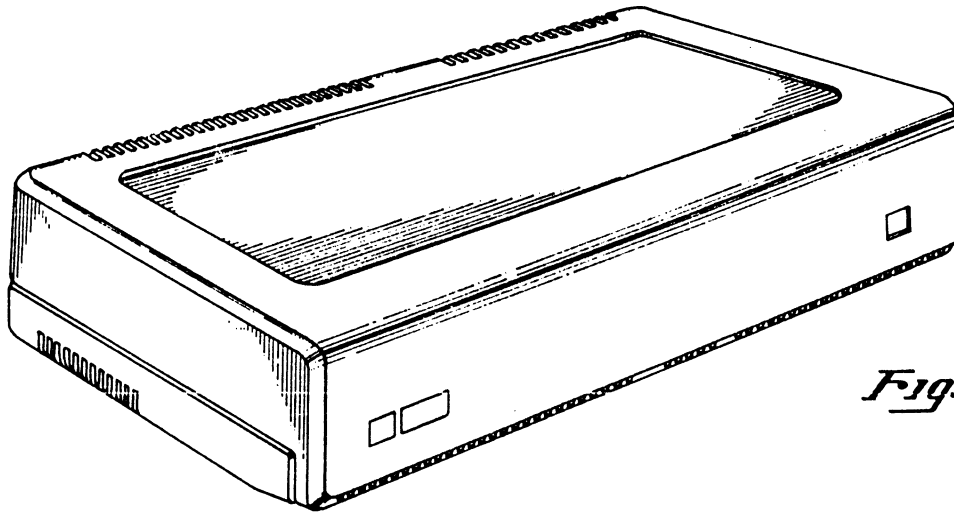


Fig. 1

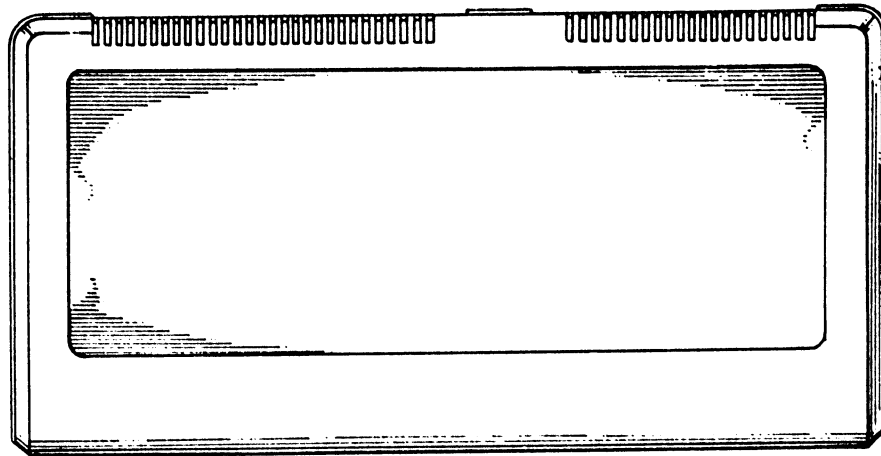


Fig. 2

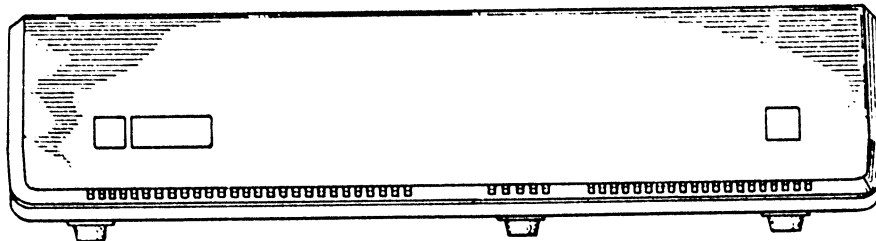


Fig. 3

U.S. Patent

Apr. 3, 1984

Sheet 2 of 2

Des. 273,295

Fig. 4

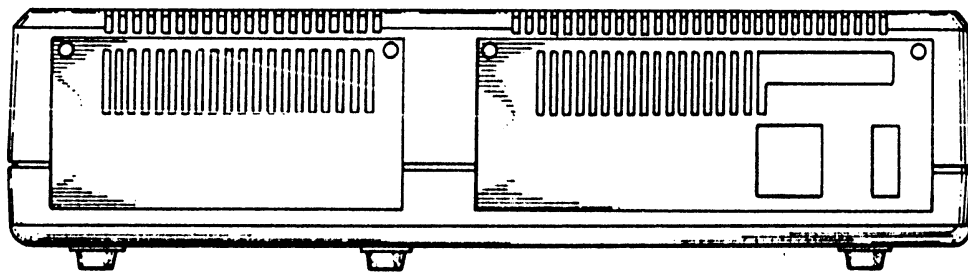


Fig. 5

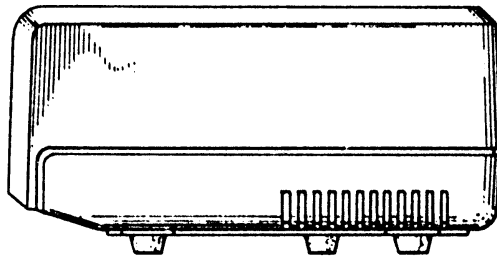
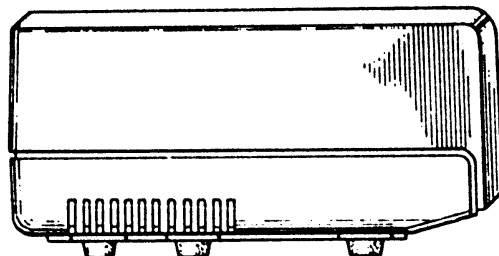
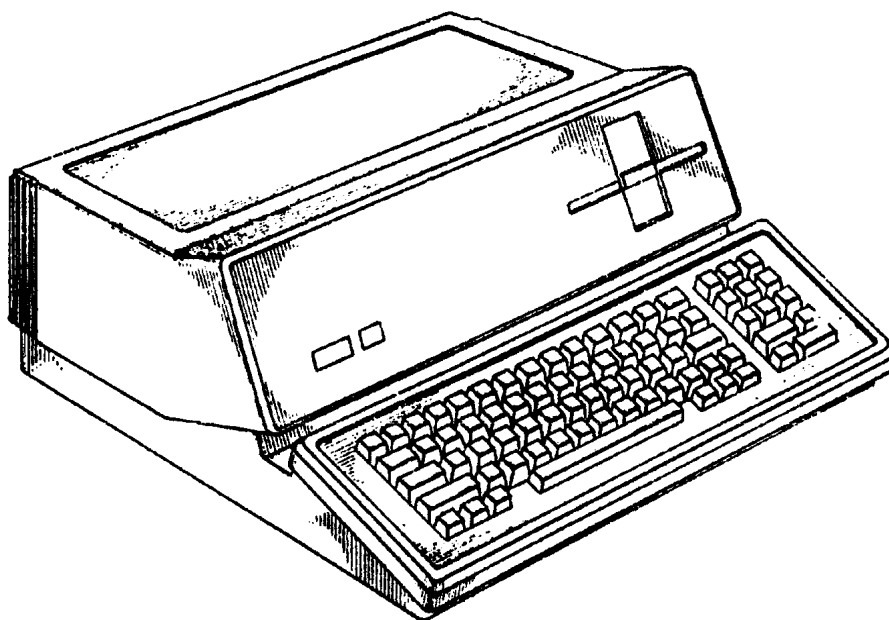


Fig. 6





Apple /// Computer Information



APPLE /// PATENT

Patent # 4,383,296 -- 10 May 1983

ADDED BY DAVID T CRAIG • 2006

United States Patent [19]

[11] **4,383,296**

Sander

Best Available Copy

[45] **May 10, 1983**

[54] **COMPUTER WITH A MEMORY SYSTEM FOR REMAPPING A MEMORY HAVING TWO MEMORY OUTPUT BUSES FOR HIGH RESOLUTION DISPLAY WITH SCROLLING OF THE DISPLAYED CHARACTERS**

[75] Inventor: **Wendell B. Sander, San Jose, Calif.**

[73] Assignee: **Apple Computer, Inc., Cupertino, Calif.**

[21] Appl. No.: **150,630**

[22] Filed: **May 16, 1980**

[51] Int. Cl.³ **G06F 13/06; G06F 3/14**

[52] U.S. Cl. **364/200; 340/726; 340/799**

[58] Field of Search ... **364/200 MS File, 900 MS File; 340/726, 798, 799; 358/17**

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,821,730	6/1974	Carey et al.	340/799 X
3,893,075	7/1975	Orban et al.	340/799 X
3,903,510	9/1975	Zobel	340/726 X
3,980,992	9/1976	Levy et al.	364/200
4,136,359	1/1979	Wozniak	358/17
4,150,364	4/1979	Baltzer	364/900 X

FOREIGN PATENT DOCUMENTS

1351590	5/1974	United Kingdom .
1482819	8/1977	United Kingdom .
1496563	12/1977	United Kingdom .
1524873	9/1978	United Kingdom .

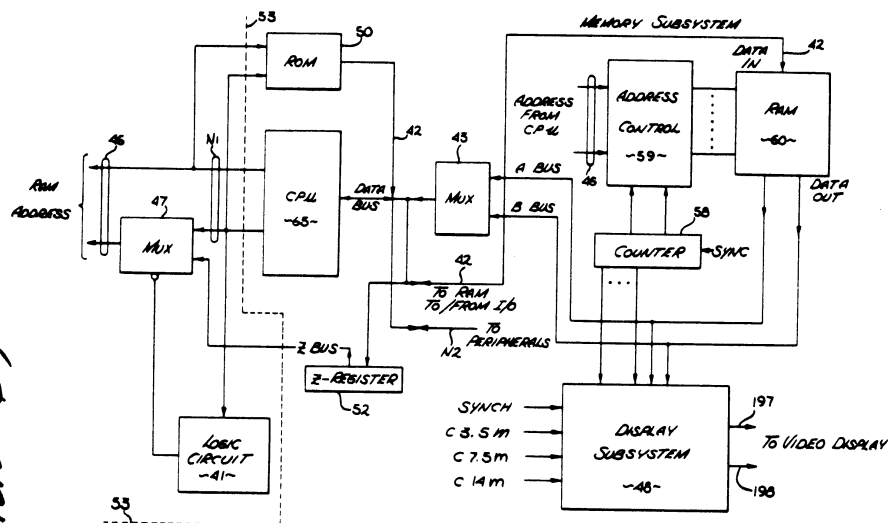
Primary Examiner—Raulfe B. Zache
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **ABSTRACT**

A microcomputer system with video display capability, particularly suited for small business applications and home use is described. The CPU performance is enhanced by permitting zero page data to be stored throughout the memory. The circuitry permitting this capability also provides a pointer for improved direct memory access. Through unique circuitry resembling "bank switching" improved memory mapping is obtained. Four-bit digital signals are converted to an AC chroma signal and a separate luminance signal for display modes. Display modes include high resolution modes, one of which displays 80 characters per line.

22 Claims, 9 Drawing Figures

CONTAINS BOOT ROM LISTING (REVISION Ø)



Apple /// Computer

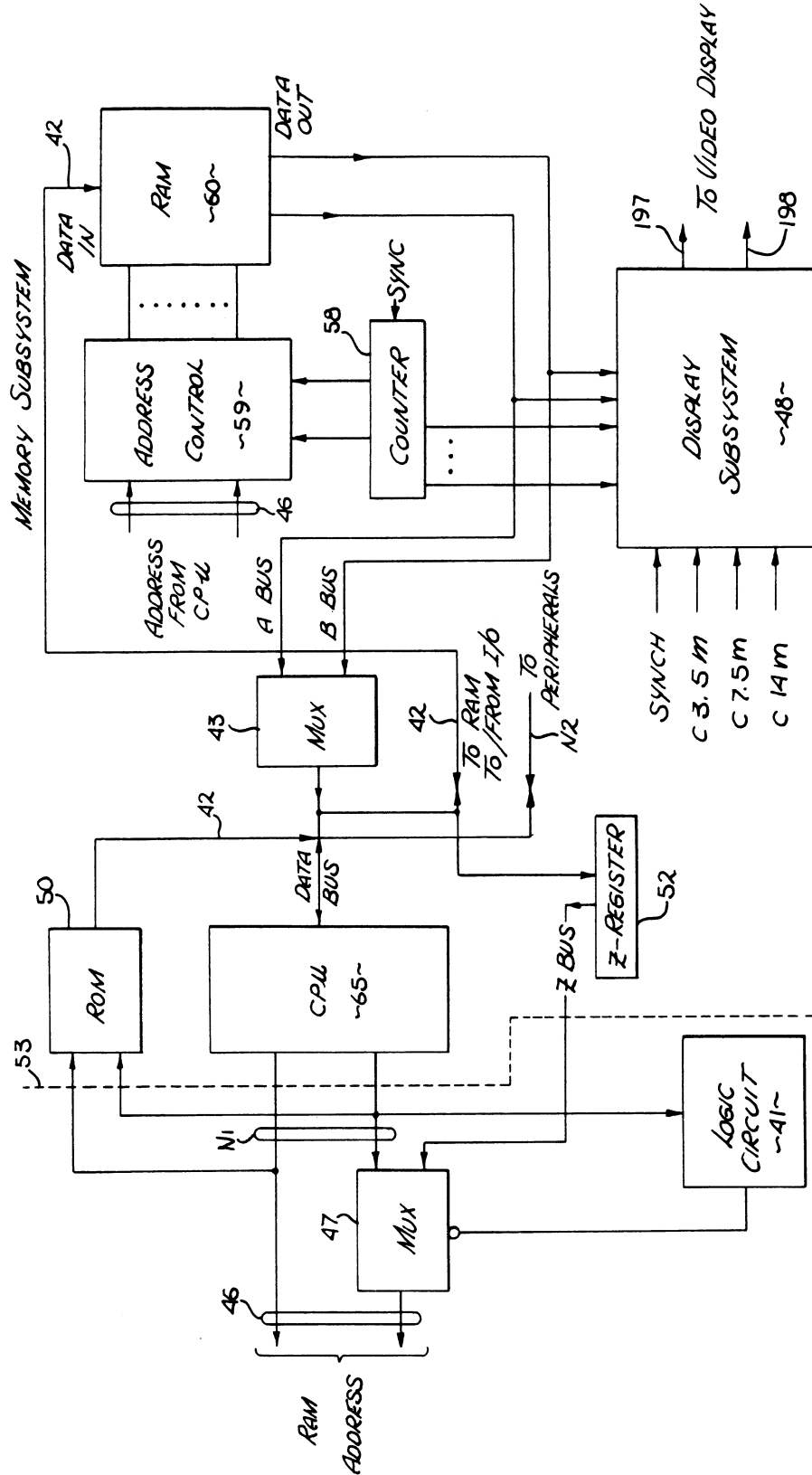


Fig. 1

U.S. Patent May 10, 1983

Sheet 2 of 8

4,383,296

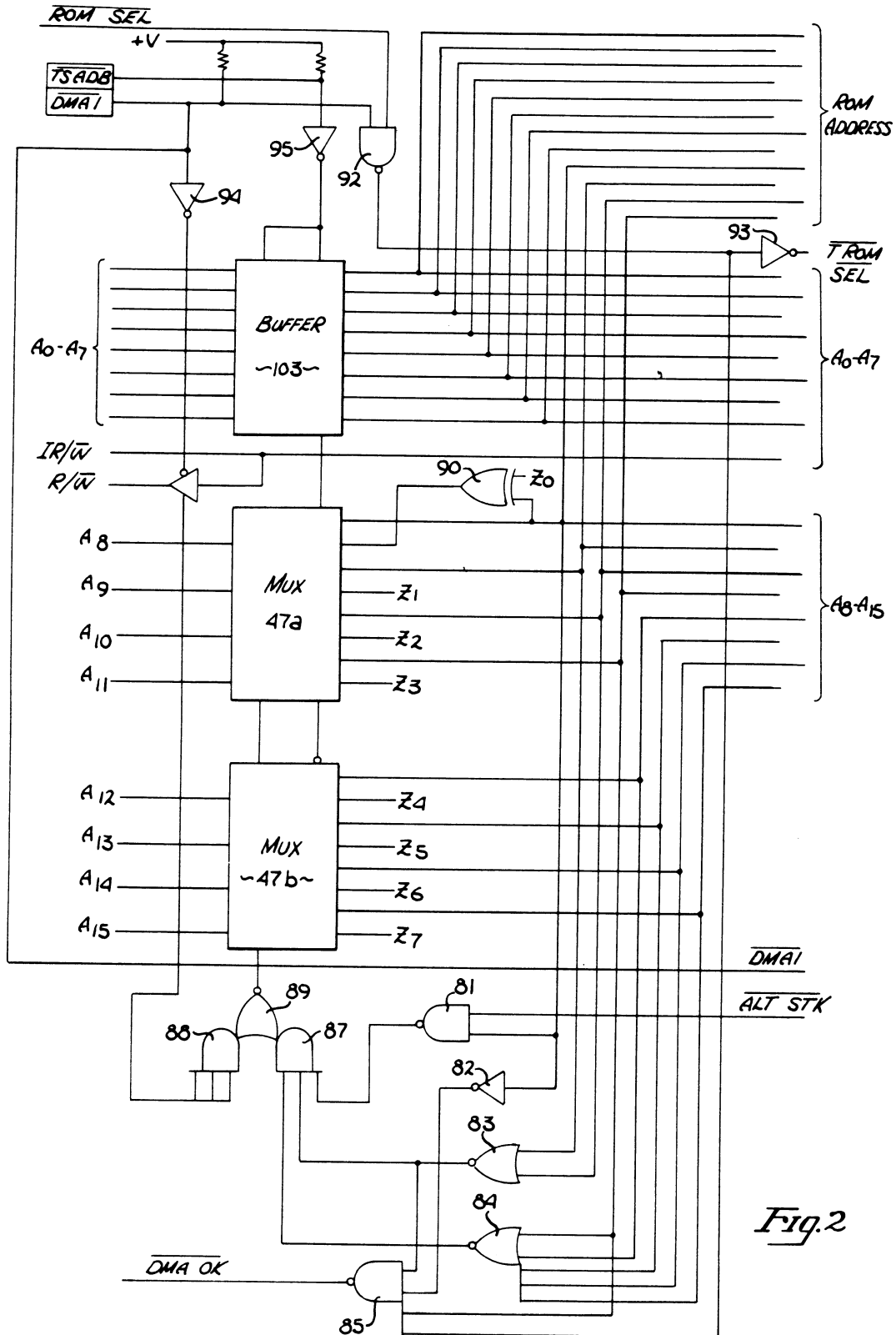


Fig. 2

U.S. Patent May 10, 1983

Sheet 3 of 8

4,383,296

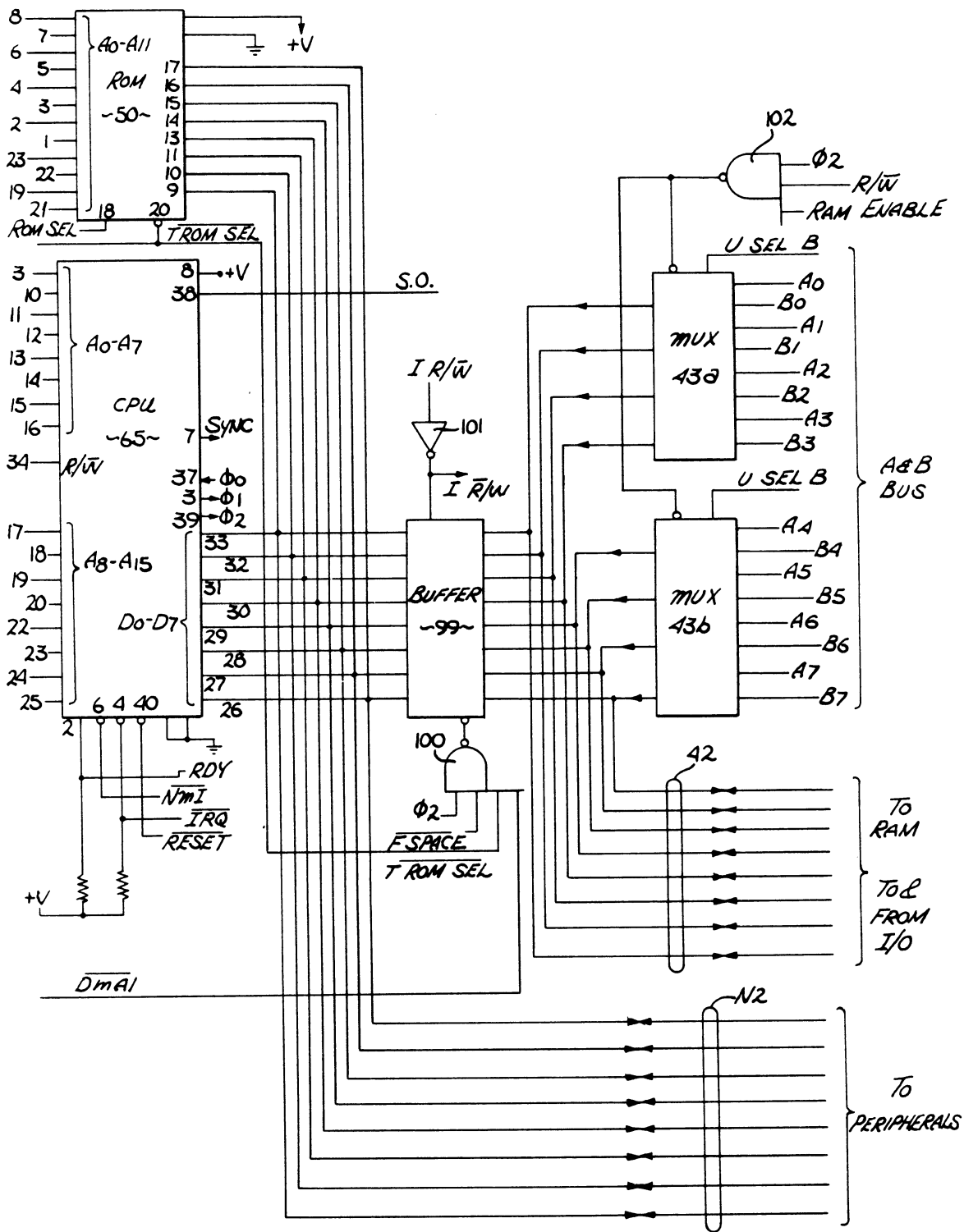


Fig. 3

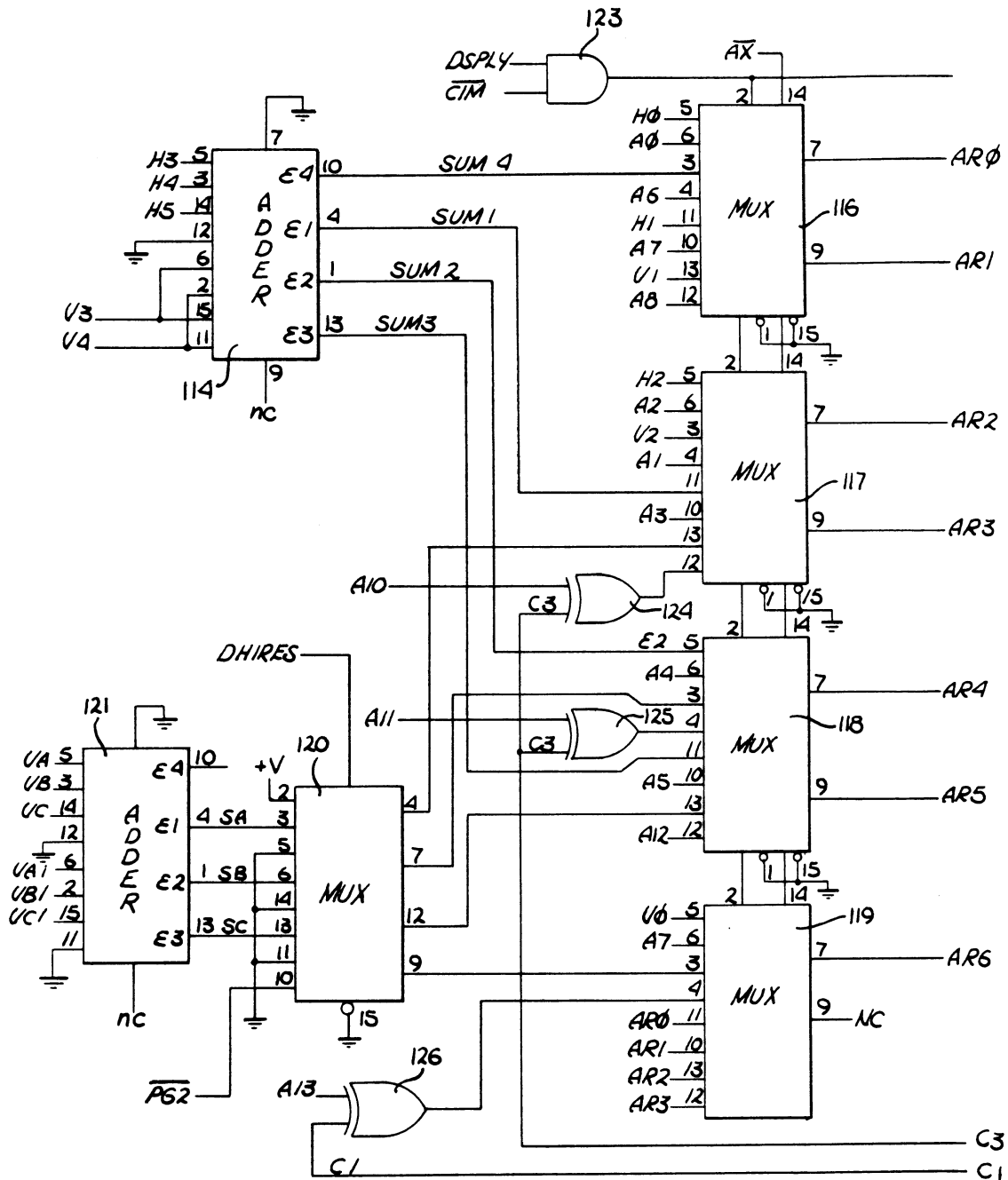
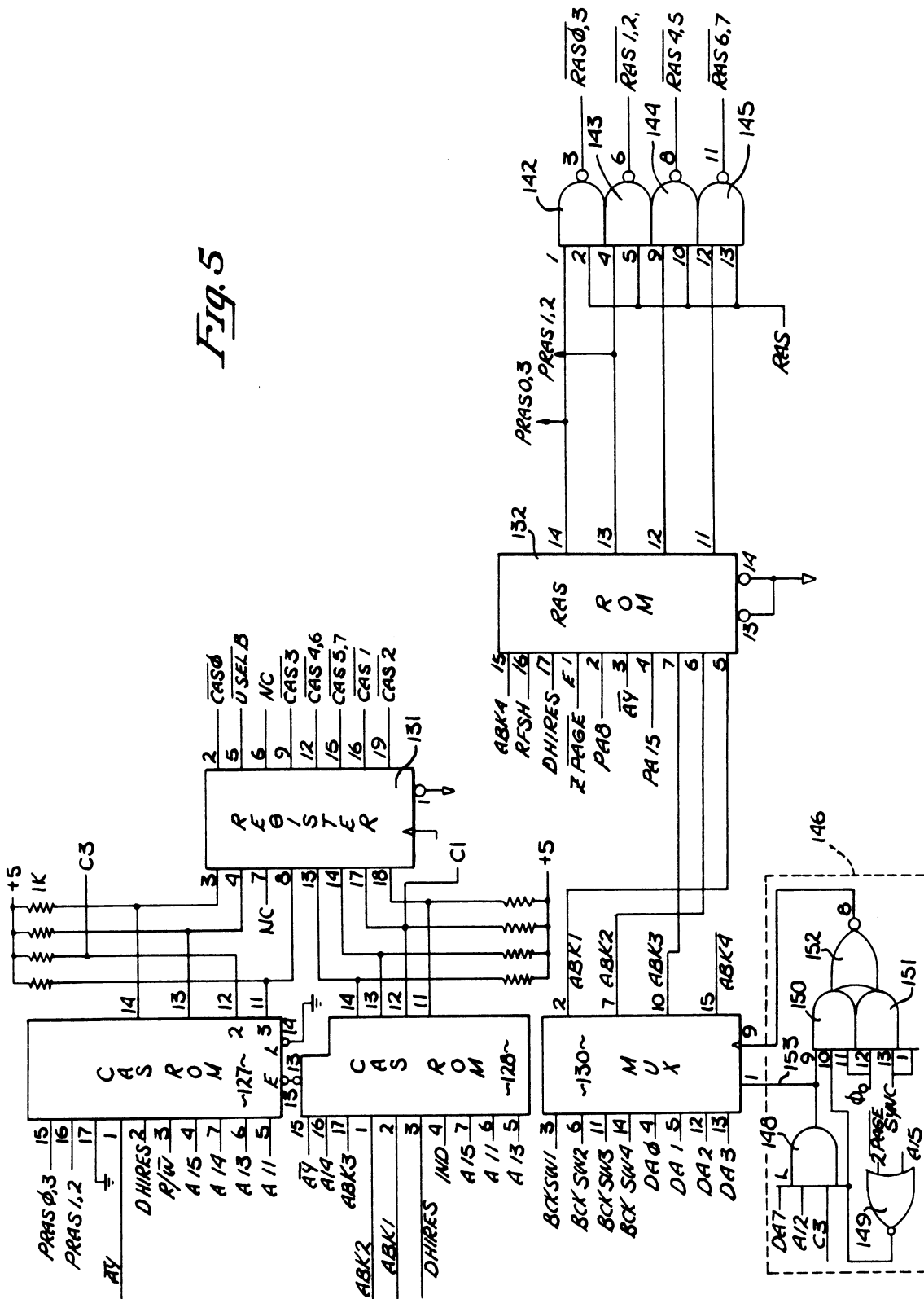


Fig. 4

FIG. 5



U.S. Patent May 10, 1983

Sheet 6 of 8

4,383,296

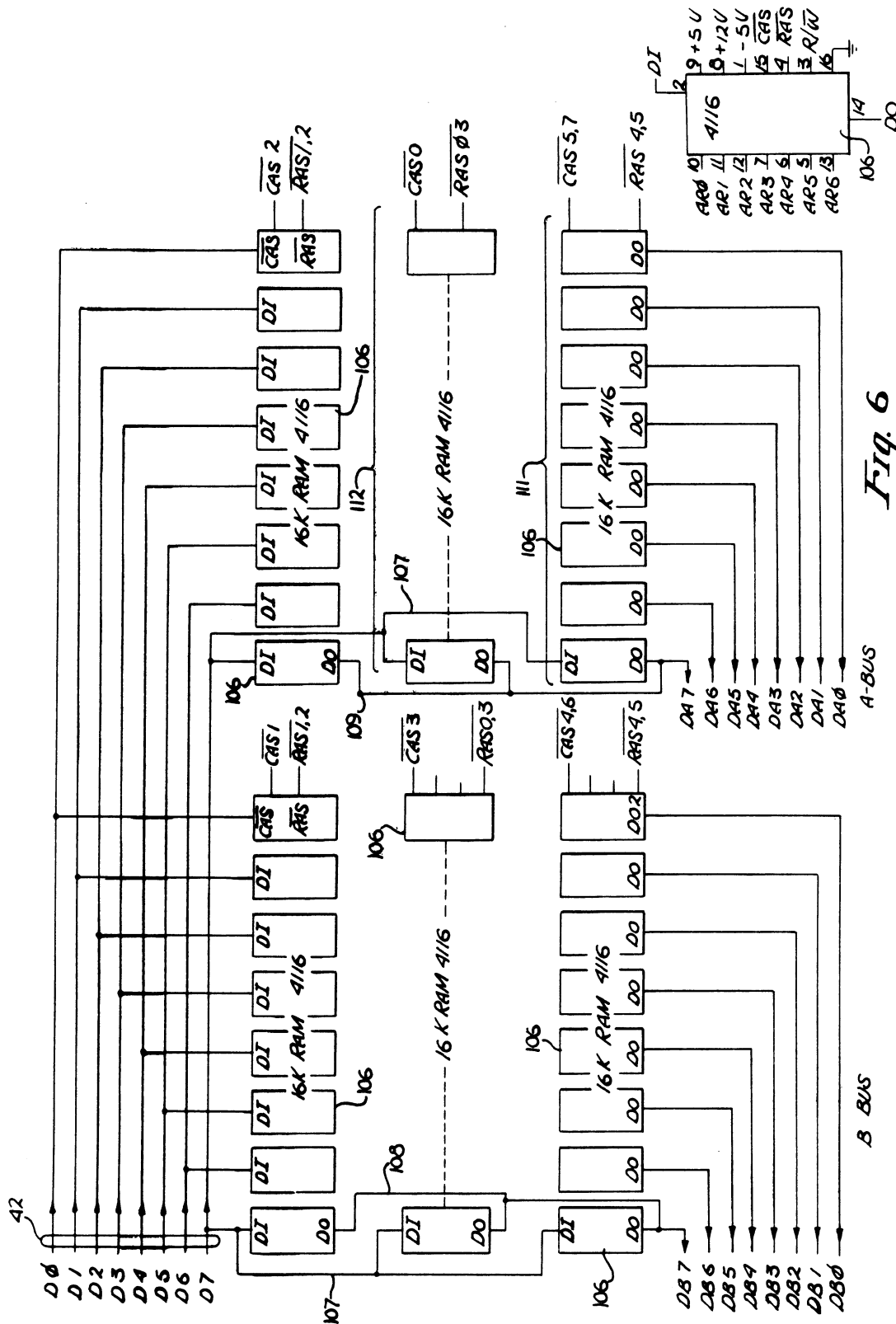


Fig. 6

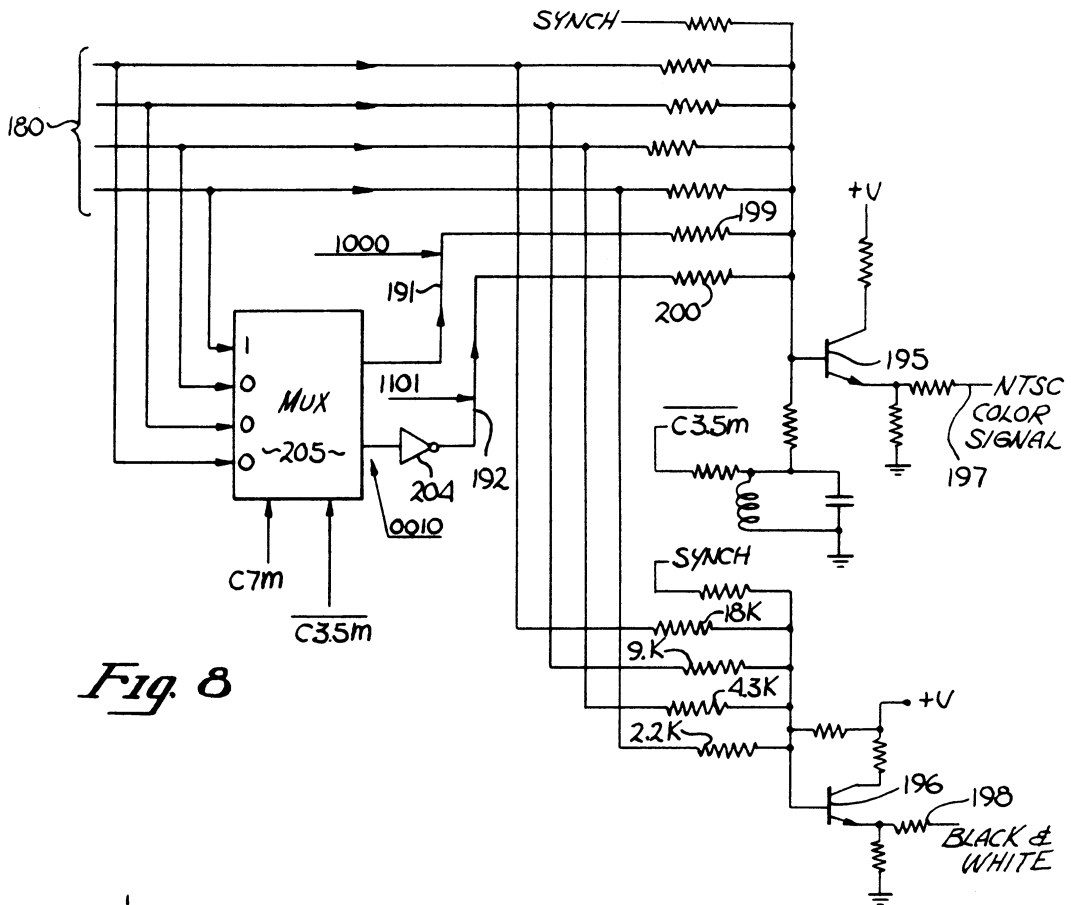


Fig. 8

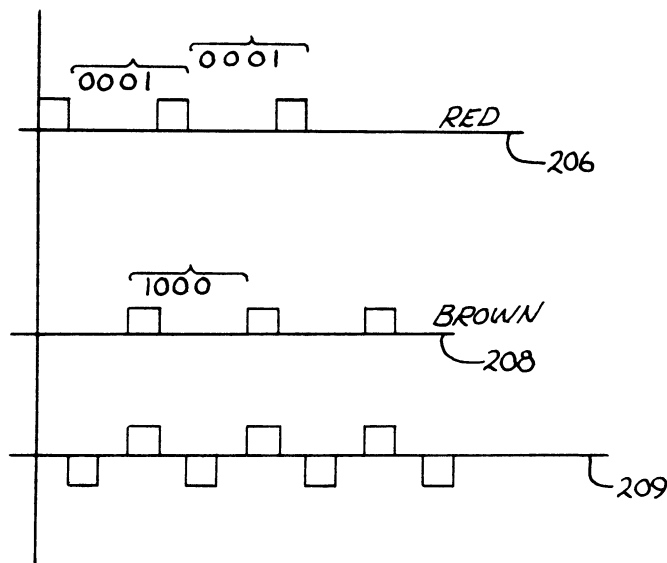


Fig. 9

4,383,296

1

**COMPUTER WITH A MEMORY SYSTEM FOR
REMAPPING A MEMORY HAVING TWO
MEMORY OUTPUT BUSES FOR HIGH
RESOLUTION DISPLAY WITH SCROLLING OF
THE DISPLAYED CHARACTERS**

BACKGROUND OF THE INVENTION

The invention relates to the field of digital computers, particularly microcomputers, having video display capabilities.

Prior Art

In the last few years, there has been rapid growth in the use of digital computers in homes by hobbyists, for small business and for routine engineering and scientific application. For the most part, these needs have been met with self-contained, relatively inexpensive microcomputers or microprocessors with essential peripherals, including disc drives and with relatively easy to manage computer programs. The design for computers for these needs requires considerable ingenuity since each computer must meet a wide range of applications and because this market is particularly cost conscious.

A home or small business computer must, for example, operate with a number of different program languages, including those requiring relatively large memories, such as Pascal. The computer should interface with a standard raster scanned display and provide a wide range of display capabilities, such as high density alpha-numeric character displays needed for word processing in addition to high resolution graphics displays.

To meet these specialize computer needs, generally requires that a relatively inexpensive microprocessor be used and that the capability of the processor be enhanced through circuit techniques. This reduces the overall cost of the computer by reducing, for example, power needs, bus structures, etc. Another important consideration is that the new computers be capable of using programs developed for earlier models.

As will be seen, the presently described microcomputer is ideally suited for home and small business applications. It provides a wide range of capabilities including advanced display capabilities not found in comparable prior art computers.

The closest prior art computer known to applicant is commercially available under the trademark, Apple-II. Portions of that computer are described in U.S. Pat. No. 4,136,359.

SUMMARY OF THE INVENTION

A digital computer which includes a central processing unit (CPU) and a random-access memory (RAM) with interconnecting address bus and data bus is described. One aspect of the present invention involves the increased capability of the CPU by allowing base page or zero page data to be stored throughout the memory. Alternate stack locations and an improved direct memory access capability are also provided by the same circuitry. Detection means are used for detecting a predetermined address range such as the zero page. This detection means causes a special register (Z-register) to be coupled into the address bus. The contents of this Z-register provide, for example, a pointer during direct memory access, or alternate stack locations for storing data normally stored on page one.

The memory of the invented computer is organized in an unusual manner to provide compatibility with the

2

8-bit data bus and yet provide high data rates (16-bits/MHz) needed for high resolution displays. A first plurality of memory devices are connected to a first memory output bus; these memory devices are also connected to the data bus. The memory includes a second plurality of memory devices which are also connected to the data bus; however, the outputs of these second devices are coupled to a second output memory bus. First switching means permit the first and second memory buses to be connected to the display for high data rate transfers. Second switching means permit either one of the memory buses to be connected to the data bus during non-display modes.

The addressing capability of the memory is greatly enhanced not only through bank switching, but through a novel remapping which does not require the CPU control associated with bank switching. In effect, the "unused" bits from one of the first and second memory buses are used for remapping purposes. This mode of operation is particularly useful for providing toggling between two separate portions of the memory.

The display subsystem of the described computer generates video color signal in a unique manner. A 4-bit color code as used in the prior art, is also used with the described display subsystem. However, this code is used to generate an AC chrominance signal and a separate DC luminance signal. This provides enhanced color capability over similar prior art color displays.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram showing the major components and subsystems of the invented and described microcomputer system.

FIGS. 2 and 3 together show the central processing unit (CPU) and the architecture associated with this CPU, particularly the address bus and data bus.

FIG. 2 is a circuit diagram primarily showing the address bus and the logic means associated with this bus.

FIG. 3 is a circuit diagram primarily showing the data bus and its interconnection with the memory buses (A bus and B bus), bootstrap read-only memory, and input/output ports.

FIGS. 4, 5 and 6 show the memory subsystem.

FIG. 4 is a circuit diagram primarily showing the circuitry for selecting between address signals from the address bus and display counter signals.

FIG. 5 is a circuit diagram primarily showing the generation of various "select" signals for the memory devices.

FIG. 6 is a circuit diagram showing the organization of the random-access memory and its interconnection with the data bus and memory output buses.

FIGS. 7 and 8 illustrate the display subsystem of the invented computer.

FIG. 7 is a circuit diagram showing the circuitry for generating the digital signals used for the video display.

FIG. 8 is a circuit diagram of the circuitry used to convert the digital signals to analog video signals.

FIG. 9 is a graph of several waveforms used to describe a prior art circuit and the circuit of FIG. 8.

**DETAILED DESCRIPTION OF THE
INVENTION**

A microcomputer system capable of driving a raster scanned video display is disclosed. In the following description, numerous specific details such as specific

4,383,296

3

part numbers, clock rates, etc, are set forth to provide a thorough understanding of the present invention. However, it will be obvious to one skilled in the art that the inventive concepts described in this patent may be practiced without these specific details. In other instances, well-known circuits have been shown in block diagram form in order not to obscure the present invention in unnecessary detail.

Referring first to FIG. 1, in general the described computer includes a central processing unit (CPU) 65, its associated data bus 42, address bus 46, a memory subsystem and a display subsystem 58.

The address bus 46 from the CPU is coupled to the memory subsystem to permit the selection of locations in memory. Some of the address signals pass through a multiplexer 47. For some modes of operation, signals from a register 52 are coupled through the multiplexer 47 onto the bus 46. The register 52 is identified as the Z-register and is coupled to the multiplexer 47 by the Z bus. The general description of the multiplexer 47 and its control by the logic circuit 41 are described in detail in conjunction with FIG. 2. In general, the circuitry shown to the left of the dotted line 53 is included in FIG. 2 while the CPU 65, memory 50, data bus 42 and multiplexer 43 are shown in detail in FIG. 3.

The address bus N1 is coupled to the read-only memory 50. The output of this memory is coupled to the computer's data bus 42. The read-only memory (ROM) 50, as will be described, stores test routines, and other data of a general bootstrap nature for system initialization.

The data bus 42 couples data to the random-access memory (RAM) 60 and to and from I/O ports. This bus also couples data to the Z-register 52 and other commonly used registers not illustrated. The data bus 42 receives data from the RAM 60 through the A bus and B bus which are selected by multiplexer 43. The peripheral Bus N2 is used, as is better illustrated in FIG. 3, for coupling to peripherals.

The memory subsystem is shown in detail in FIGS. 4, 5 and 6. The address control means which receives addresses on bus 46, makes the final selection of memory locations within the RAM 60. Bank switching, addressing for display purposes, scrolling and other memory mapping is controlled by the address control means 59 as will be described in greater detail in conjunction with FIGS. 4 and 5. The PAM 60 is shown in detail in FIG. 6. The counter 58 which is synchronized with the horizontal and vertical display signals, provides signals both to the address control means 59 and to the display subsystem 48.

The display subsystem receives data from the RAM 60 on the A bus and B bus and converts these digital signals to video signals which control a standard raster scanned display. A standard NTSC color signal is generated on line 197 and a black and white video signal on line 198. The same signals used to generate these video signals can be used to generate separate red, green, blue (RGB) video signals. The display subsystem 48 receives numerous timing signals including the standard color reference signal shown as 3.5 MHz (C3.5M). This subsystem is described in detail in FIGS. 7 and 8.

COMPUTER ARCHITECTURE

In the presently preferred embodiment, the CPU 65 (microprocessor) employed with the described computer is a commercially available component, the 6502A. This 8-bit processor (8-bit data bus) which has a

4

16-bit address bus is shown in FIG. 3 with its interconnections to the remainder of the computer. The pin number for each interconnection is shown adjacent to the corresponding line. In many cases, the nomenclature associated with the 6502A (CPU 65) is used in this application. For example, pin 6 receives the nonmaskable interrupt signal (\overline{NMI}), and pin 4 is coupled to receive the interrupt request signal (\overline{IRQ}). Some of the signals employed with the CPU 65, which are well-known in the art, and which are not necessary for the understanding of the present invention are not described in detail in this application, such as the various synchronization signals and clocking signals. The address signals from the CPU 65 are identified as A₀-A₇ and A₈-A₁₅. The data signals associated with the CPU 65 are shown as D₀-D₇. As will be apparent to one skilled in the art, the inventive concepts described in this application may be employed with other microprocessors.

Referring now to FIGS. 2 and 3, the general architecture, particularly the architecture associated with the CPU 65 can best be seen. The address signals A₀-A₇ are coupled to a buffer 103 by the bus shown primarily in FIG. 2. These address signals are also coupled to the ROM 50. The signals A₀-A₇ after passing through the buffer 103 are coupled to the memory subsystem. The address signals A₈-A₁₅ (higher order address bits) are coupled through lines shown in FIG. 2 to the multiplexers 47a and 47b. The contents of the Z-register 52 of FIG. 1 is also connected to the multiplexers 47a and 47b through the Z-bus (Z₁-Z₇). The multiplexers 47a and 47b allow the selection of either the signals A₈-A₁₅ from the CPU 65 or the contents of the Z-register (Z₁-Z₇) for addressing the RAM 60. The output of these multiplexers are shown as A₈-A₁₅; this designation is used even when the Z-bus is selected. Note in the case of the Z₀ signal, this signal is coupled to the multiplexer 47a through the exclusive OR gate 90 for reasons which are explained later. The address signals A₈-A₁₁ are also coupled to the ROM 50, thus the signals A₀-A₁₁ are used for addressing the ROM 50. The signals A₈-A₁₅ are connected to the logic circuit shown in the lower left-hand corner of FIG. 2; this logic circuit corresponds to the logic circuit 41 of FIG. 1.

The input and output data signals from the CPU 65 are coupled by a bidirectional bus to the bidirectional buffer 99 (FIG. 3). This buffer is selectively disabled by gate 100 to allow the output of ROM 50 to be communicated to CPU 65 and during other times not pertinent to the present discussion. The direction of flow through the buffer 99 is controlled by a read/write signal coupled to the buffer through inverter 101. Data from the CPU 65 is coupled through the buffer 99 and bus 42 to the RAM 60 or to I/O ports. Data from the RAM 60 is communicated to CPU 65 or bus N2 from the A bus and B bus through the buffer 99. The 4 lines of the A bus and 4 lines of the B bus are coupled to the multiplexer 43a. Similarly, the other 4 lines of the A and B buses are coupled to the multiplexer 43b. Multiplexers 43a and 43b select the 8 lines of the A bus or B bus and communicate the data through to buffer 99 and bus 42. These multiplexers are selectively disabled (for example, during writing) by gate 102. As will be described later, the 16 lines of the A bus and B bus permits the reading of 16-bits from the RAM at one time. This provides a data rate of 16-bits/MHz which is necessary, for example, for an 80 character per line display. The data is loaded into the RAM 60, 8-bits at a time.

4,383,296

5

The ROM 50, as mentioned, stores test programs, data needed to initialize various registers, character generation data (for RAM 162 of FIG. 7) and other related data. Specific programs employed in the presently preferred embodiment of the computer are set forth in Table 1. The ROM 50 is selected by control signals coupled to its pins 18 and 20, identified as signals ROM SEL and TROM SEL. Any one of a plurality of commercially available read-only memories may be used for the ROM 50. In the presently preferred embodiment, commercially available Part No. SY2333 is used.

Referring now to this logic circuit (lower left-hand corner of FIG. 2), the NAND gate 81 receives the address signal A_8 and also the alternate stack signal identified as $\overline{\text{ALT STK}}$. The output of this gate provides one input to the AND gate 87. The A_8 signal is also coupled through the inverter 82 to one input terminal of the NAND gates 85 and 86. The address signals A_9 and A_{10} are coupled to the input terminals of the NOR gate 83. The output of this gate is coupled to one input terminal of the NAND gates 85 and 86 and the AND gate 87. The address signals A_{11-15} are coupled to the input terminals of the NOR gate 84. The signal A_{11} is also coupled to an input terminal of the NAND gate 85.

The outputs of the AND gates 87 and 88 (through NOR gate 89), controls the multiplexers 47a and 47b. When the output of gate 89 is low the Z-bus is selected, otherwise the address signals from the CPU 65 are selected.

The logic circuit above-described, along with the Z-bus and Z-register provide enhanced performance for the computer. First, this circuit permits the zero page or base page data to be stored throughout the RAM 60 rather than just on zero page. Secondly, this circuit enables addressing of alternate stack locations (other than page one). Lastly, this circuit through the Z-register provides a RAM pointer for direct memory access (DMA).

Assume for purposes of discussion that the CPU 65 is addressing the zero page of memory. That is, the higher order address bits A_8-A_{15} are all zeros. The zeros for A_9-A_{15} are detected by the gates 83 and 84. If all the inputs to these gates are zeros, the outputs of these gates are high which condition is communicated to the gate 87. A_8 which is also low, insures that the output of gate 81 will be high. Thus, all the inputs to gate 87 are high, causing the signal at the output of the gate 89 to drop. When this occurs, the Z-bus is selected. Instead of all the binary zeros from the CPU being coupled to the main memory (RAM 60), the contents of the Z-register form part of the address for the memory. Therefore, even though the CPU 65 has selected the zero page, nonetheless data may be written into or from any location of RAM 60 (including the zero page). This enhances the performance of the CPU, since for example, the time consumed in shifting data to and from a single zero page is minimized.

Normally, the CPU 65 selects page one for stack locations. This occurs when A_8 is high and A_9-A_{15} are low. Assume first that the alternate stack locations have not been selected. Both inputs to gate 81 are high and its output is low. The low input to the gate 87 prevents the selection of the Z-bus. Thus, for these conditions the address signals A_0-A_7 select stack locations on page one.

6

Next assume that page one has been selected by the CPU and that the $\overline{\text{ALT STK}}$ signal is low, indicating the alternate stack locations are to be selected. (A flag is set by the CPU to change the $\overline{\text{ALT STK}}$ signal). Since the $\overline{\text{ALT STK}}$ signal is low and A_8 is high, a high output occurs from the gate 81. All the inputs to gates 83 and 84 are low, therefore, high outputs occur from both these gates. The conditions of gate 87 are met, causing a high output from this gate and lowering the output from the gate 89. The Z-bus is thus selected by the multiplexers 47a and 47b. This allows the contents of the Z-register to be used as alternate locations. Non-zero page locations are assured by inverting A_8 . The exclusive OR gate 90 acts as a selective inverter. If A_8 is high and Z_0 is low, then A_8 at the output of the multiplexer 47a will be low. Note that during zero page selection when A_8 is low, the Z_0 signal is directly communicated through gate 90 to the output of multiplexer 47a.

Thus, the logic circuits along with the $\overline{\text{ALT STK}}$ signal allows alternate stack locations to be selected through the Z-bus. This further enhances the performance of the CPU which would otherwise be limited to page one for stack locations.

The logic circuit of FIG. 2 is also used along with the Z-register to provide a pointer during direct memory access (DMA). Assume that direct access to the computer's memory is required by a peripheral apparatus. To initiate the DMA mode the CPU provides an address between F800 and R8FF. Through a logic circuit not illustrated in FIGS. 2 and 3, the ROM SEL signal is brought low for addresses between F000 and FFFF. This signal is communicated to gate 93 and causes the output of gate 92 to rise (DMA I is high at this time). This rise in potential is communicated to one input of the gate 85. Additionally, gate 85 senses that the address bits A_8, A_9 and A_{10} are low. This information is coupled to gate 85 through the inverter 82 and the NOR gate 83 as high signals. Also the fact that A_{11} is high is directly communicated to gate 85. Thus, with the address between F800 and F8FF the $\overline{\text{DMA OK}}$ signal drops in potential. This is sensed by the peripheral apparatus which in turn causes the DMA I signal to drop and provides a ready signal to the CPU 65. With the completion of this handshake, data may begin to be transferred to the RAM.

The DMA I signal through gate 93 and inverter 93 forces the $\overline{\text{TROM SEL}}$ signal low. This signal in addition to being communicated to the ROM 50, is coupled to the buffer 99 through gate 100, disabling this buffer (during the reading of ROM 50). Also, the ready signal causes the CPU to come to a hard stop. Importantly, the DMA I signal, after passing through the inverter 94 and the gates 88 and 89, assures the selection of the Z-register. The contents of the Z-register are fixed and provide a pointer to a page in the RAM.

Under the above conditions, the CPU increments the lower 8-bits of the address signal. The ROM 50 furnishes the instructions for incrementing the address, specifically SBC #1 and BEQ. The peripheral apparatus provides the data or receives the data in synchronization with the CPU operation. The peripheral also furnishes a read/write signal to indicate which operation is to occur. Data is then written into RAM via bus N2 and bus 42, or read from RAM via the A and B buses and bus N2.

Importantly, with the above DMA arrangement, addresses from the peripheral apparatus are not neces-

sary and the Z-register is used to provide a pointer to a page in RAM 60.

MEMORY SUBSYSTEM

The memory subsystem shown in FIG. 1 as the address control means 59 and RAM 60 is illustrated in detail in FIGS. 4, 5 and 6 as mentioned. In FIGS. 4 and 5, the memory control means is shown, while in FIG. 6 the memory devices and their organization are illustrated. The address control means of FIGS. 4 and 5 receives the address signals from the CPU 65 (A₀-A₁₅), the count in the vertical and horizontal counters (counter 58 of FIG. 1) which are used during display modes, control signals from the CPU and other signals. In general, this control means develops the address signals which are coupled to the RAM of FIG. 6 including the column address and row address signals, commonly referred to as $\overline{\text{CAS}}$ and $\overline{\text{RAS}}$. Other related functions are also shown in FIGS. 4 and 5, such as the circuitry which provides display scrolling, indirect RAM addressing and memory mapping.

The CPU of FIG. 3 provides a 16-bit address for addressing the memory. Under ordinary circumstances this address limits the memory capacity to 64K bytes. This size memory is insufficient in many applications, as for example, to effectively use the Pascal program language. As will be described in greater detail, the address control means of FIGS. 4 and 5 enable the use of a memory having a 96K byte or 128K byte capacity. One well-known technique which is used with the present invention for increasing this capacity is bank switching; this switching occurs under the control of the CPU. In addition, the address control means uses a unique indirect addressing mode which provides the benefits of bank switching, however, this mode does not require CPU control. This greatly enhances CPU operation with the larger memory (as will be described) when compared to the CPU controlled bank switching.

Referring first to FIG. 6, the RAM configuration is illustrated for a capacity of 96K bytes. The memory is organized into six rows, each of which includes eight 16K memory devices such as rows 111 and 112. In the presently preferred embodiment, Part No. 4116 MOS dynamic RAMs are used. (The pin designations and signal designations refer to this memory device.) Obviously, other memory devices may be employed.

Input data to these memory devices 106 is provided from the bus 42. Each line in the bus 42 is connected to the data input terminal of one device 106 in each row. The interconnection of this bus with each of the memory devices is not shown in FIG. 6 in order to overcomplicate this drawing. By way of example, however, line 107 connects the data bit D7 to the data input terminal of one of the memory devices in each of the six rows.

Three rows of devices 106 have their output terminals coupled to the A bus, and three rows are similarly coupled to the B bus. By way of example, line 108 connects three output terminals of devices 106 to the DB7 line of the B bus while line 109 connects three output terminals of the devices 106 to the DA7 line of the A bus.

The described memory devices 106 are each organized as a 16KX1 memory. Thus, each device receives a 14-bit address which is time multiplexed into two, 7-bit addresses. This multiplexing occurs under the control of the $\overline{\text{CAS}}$ and $\overline{\text{RAS}}$ signals as is well-known. The lines coupling the address signals to each of the devices in FIG. 6 are not illustrated. However, in the

lower right-hand corner of FIG. 6, the various signals applied to each device (including the address signals), along with the corresponding pin numbers are shown. Other circuitry not illustrated is the refresh control circuitry which operates in a well-known manner in conjunction with the $\overline{\text{CAS}}$, $\overline{\text{RAS}}$ and address signals to refresh the dynamic devices.

Each row of memory devices 106 receives a unique combination of $\overline{\text{CAS}}$ and $\overline{\text{RAS}}$ signals. For example, row 111 receives $\overline{\text{CAS}}$ 5, 7 and $\overline{\text{RAS}}$ 4, 5; similarly, row 112 receives $\overline{\text{CAS}}$ 0 and $\overline{\text{RAS}}$ 0, 3. The generation of these $\overline{\text{CAS}}$ and $\overline{\text{RAS}}$ signals is described in conjunction with FIG. 5. These signals (along with the 14-bit address signals) permit the selection of a single 8-bit location in the 96K byte memory (for writing) and also the selection (for reading) of 16-bit locations.

The memory of FIG. 6 may be expanded to a 128K byte memory by using 32K memory devices, such as Part No. 4132. In this case, four rows of eight, 32K memory devices are used with each row receiving two $\overline{\text{CAS}}$ and $\overline{\text{RAS}}$ signals.

Before reviewing FIG. 4, a general understanding of the organization of the display is helpful. The display, during certain modes, is organized into 80 horizontal segments and 24 vertical segments for a total of 1920 blocks. 11-bits of the counter 58 of FIG. 1 are used as part of the address signals for the memory to access data for displaying during these modes. These counter signals are shown in FIG. 4 as H₀-H₅ and V₀-V₄. During other display modes each horizontal segment is further divided into 8 segments (e.g. for displaying 80 alpha numeric characters per line). This requires 3 additional vertical timing signals shown as V_A, V_B and V_C in FIGS. 4 and 7.

Often in the prior art, two separate counters are used to supply the timing/address signals for accessing a memory when the data in the memory is displayed. The count in one counter represents the horizontal lines of the screen (vertical count) and the other the position along each line, (horizontal or dot count). In many prior art displays the most significant bit of the dot counter is used to increment the line counter. Data in memory intended for display is mapped with a one-to-one correlation to the counts in these counters. In another prior art system (implemented in the Apple-II computer sold by Apple Computer, Inc.) this one-to-one correlation is not used. Rather, to conserve on circuitry, a single counter is employed and a more dispersed mapping is used in the memory. (Note that where a maximum horizontal count of 80 is used, this number cannot be represented by all ones in a digital counter and thus the vertical counter cannot easily be incremented by the most significant bit in the horizontal counter.) Since this more dispersed mapping technique is part of the prior art and not critical to an understanding of the present invention, it shall not be described in detail. However, the manner in which it is implemented shall be discussed in conjunction with the adder 114 of FIG. 4. For purposes of discussion, the signals from the counter 58 of FIG. 1 are designated as either vertical (V) or horizontal (H).

Referring now to FIG. 4, the selection of either the counter signals on the address signals from the CPU is made by the multiplexers 116, 117, 118 and 119. Each of these commercially available multiplexers (Part No. 153) couples one of four input lines to an output line. There are eight inputs to multiplexers 116, 117 and 118 and the outputs of these multiplexers provide the ad-

dress signals for the memories (AR0 through AR5). The multiplexer 119 has four inputs on its pins 3, 4, 5, 6 and provides a single output on pin 7, the AR6 address signal. (The signals supplied to pins 11, 12 and 13 of multiplexer 119 are for clamping purposes only.)

The AX signal is applied to the pin 14 of each of the multiplexers. The signal on this line and the signal applied to pin 2, determines which of the four inputs is coupled to each of the outputs of the multiplexers. The AX signal is a RAM timing signal for clocking the first 7 bits and second 7 bits of the multiplexed 14-bit address applied to each of the memory devices 106. The other control signal to the multiplexers is developed through the AND gate 123. The inputs to this gate are the display signal (DSPLY) which indicates that the computer is in a display mode and a clocking signal, specifically a 1MHz timing signal (CIM). The output of the AND gate 123 determines whether the address signals from the CPU or the signals associated with the counter 58 of FIG. 1 are selected.

Assume for purposes of discussion that the display has not been selected, and thus, the output of gate 123 is low. The AX signal then selects for pin 7 of multiplexer 116 first the address signal A₀ and then A₆. Likewise, each of the multiplexers selects an address signal (except for those associated with exclusive OR gates 124 and 125 which shall be discussed). If the display signal is high and an output is present from the gate 123, then, by way of example, the AX signal first causes the H₁ signal and then the V₁ signal to be connected to the AR1 address line. Similarly, signals corresponding to the vertical and horizontal count are coupled to the other address lines during display modes.

The adder 114 is an ordinary digital adder for adding two 4-bit digital nibbles and for providing a digital sum signal. A commercially available adder (Part No. 283) is employed. The carry-in terminal (pin 7) is grounded and no carry-outs occur since one of the inputs (pin 12) is grounded. The adder sums the digital signal corresponding to H₃, H₄ and H₅ with the digital signal corresponding to V₃, V₄, V₃, V₄. The resultant sum signal is coupled to the multiplexers 116, 117 and 118 as illustrated. The summing of these horizontal and vertical counter signals is used to provide the more dispersed mapping as previously discussed.

The adder 121 is identical to adder 114 and is coupled to sum the three least significant vertical counter bits from the counter 58 (FIG. 2) with the signals VA1, VB1 and VC1. The sum is selected by the multiplexer 120 during the high resolution display modes and also during scrolling as will be described. These sum signals are coupled to the multiplexers 117, 118 and 119. During the low resolution display modes, the multiplexer 120 couples ground signals or the page 2 signal (PG2) to the multiplexers 117, 118 and 119. (The PG2 signal is used for special mapping purposes, not pertinent to the present invention.) During the high resolution modes when the display is not being scrolled, the VA1, VB2 and VB3 signals are at ground potential and thus no summing occurs within adder 121 and the VA, VB and VC signals are coupled directly to the multiplexers 117, 118 and 119.

The address signals A₁₀, A₁₁, and A₁₃ from the CPU are coupled to the multiplexers 117, 118 and 119, respectively, through exclusive OR gates 124, 125, and 126, respectively. The other input terminals to gates 124 and 125 receive the C₃ signal, while the other input terminal of the gate 126 receives the C₁ signal. (The

development of the C₁ and C₃ signals is illustrated in FIG. 5.) The gates 124, 125 and 126 provide mapping compensation within the memory. As the computer and memory are presently implemented, the sequence in which the various portions of the display are generated is not the same as the sequence in which the data is removed from memory for display. These gates provide compensating addresses and, in effect, cause a remapping so that the proper sequence is maintained when data is read from the memory for the display. These gates are shown to provide a complete disclosure of the presently preferred embodiment, however, they are not critical to the present invention.

In operation, the circuitry of FIG. 4, as mentioned, selects the address signals which are applied to each of the memory devices, either from the CPU or counter if the display mode is selected. It should be noted that not all of the address bits from the CPU are coupled to the multiplexers 116 through 119. Some of these address bits, as will be described in conjunction with FIG. 5, are used to develop the various CAS and RAS signals and thus select different memory rows within the memory of FIG. 6.

The scrolling operation which is used is somewhat unusual in that each line of the display is separately moved up (line-by-line) with one line of data in memory being moved for each frame. This technique provides a uniform, esthetically pleasing, scroll. Scrolling the screen one line per frame can be achieved by moving all the data in the memory into a new position for each frame. This would be very time consuming and impractical. With the described technique, only one-eighth of the data in the memory is moved for each new frame.

Referring to the adder 121, as mentioned, the signals V_A, V_B, V_C are the three least significant vertical counter bits from the counter 58. These bits or counts, by way of example, represent the 8 horizontal lines of each character. In adder 12, a 3-bit digital signal, VA1, VB1 and VC1, is added to the count from counter 58. This 3-bit signal is constant during each frame, however, it is incremented for each new frame.

During a first frame, 000 is added to the vertical count. During a second frame, 001 is added; and during a third frame, 010 is added, and so on. By adding this digital signal to the count from counter 58, the addresses to the memory are changed in the vertical sense. During the first frame when 000 is added, the display remains unaffected. During the next frame, when 001 is added to the vertical count, instead of first displaying the first line of a character, the second line of each character is displayed at the top of each character space and each subsequent line of the character is likewise moved up one line. If data in memory is not moved, the first line of the character would appear at the bottom of each character. Note when 001 is added to 111 from the counter, 000 results. Thus, the first line of characters would be addressed when the beam is scanning the eighth line of characters. To prevent this, the data corresponding to the first line of each character is moved in memory for this frame. The first line of one character is moved up and becomes the bottom line of the character directly above it. When 010 is added, the process is again repeated. For example, the third line of each character is first displayed in each character space and the second line of each character is moved up to become the bottom line of the character directly above it. This process is repeated to scroll the data. The movement of data in memory is controlled by the CPU in a well-known manner.

11

Thus, through use of adder 121, an even, continuous scroll is obtained without moving all the data in memory for each frame. Rather, only 1/8th of the data is moved for each frame.

Referring now to FIG. 5, the circuitry used to extend the addressing from the CPU is illustrated. In general, the CAS signals are generated by the ROMs 127 and 128. The RAS signals are generated by the ROM 132. The multiplexer 130 allows the selection of either the bank switching signals, or the unique indirect addressing mode when "bank switching" occurs without direct commands from the CPU.

The CAS ROM 127 receives as an address the following signals: PRAS,φ3, PRAS 1,2 AY, DHIRES, R/W, A11, A13, A14, and A15. As the PRASφ, 3 and PRAS 1, 2 represent the RAS signals being used. These signals are high when the respective RAS signal is active.

As previously mentioned, the AY signal is high for display modes and the DHIRES signal is high for high resolution display modes. The CAS ROM 128 receives as address signals the ABK1, ABK2, and ABK3 signals and also DHIRES, AY, IND, A11, A13, A14, and A15.

The ROMs 127 and 128 are programmed to implement the following equations.

$$PCAS0 = (PRAS0,3 \cdot (DHIRES \cdot AY + AY \cdot (A15 \cdot A14 \cdot A13 \cdot R/WN + A15 \cdot A14 \cdot A13 \cdot R/WN + A15 \cdot A14 \cdot A13 \cdot \bar{A}11))) \quad (1)$$

$$PCAS2 = (DHIRES \cdot AY + AY \cdot (ABK1 \cdot ABK2 \cdot ABK3 \cdot \bar{A}15 \cdot A14 \cdot A13 + A14 \cdot A13)) \quad (2)$$

$$PCAS3 = PRAS0,3 \cdot (DHIRES \cdot AY + AY \cdot (A15 \cdot A14 \cdot A13 \cdot A11 + A15 \cdot A14 \cdot A13 \cdot \bar{A}11)) \quad (3)$$

$$PCAS4,6 = (AY \cdot \bar{IND} \cdot ABK3 \cdot A15 \cdot (ABK1 \cdot ABK2 + ABK1 \cdot ABK2) \cdot (A14 \cdot A13 + A14 \cdot A13) + AY \cdot \bar{IND} \cdot ABK3 \cdot (ABK1 \cdot A15 + ABK2 \cdot ABK1 + ABK2 \cdot ABK1 \cdot A15) \cdot A14 + AY \cdot \bar{IND} \cdot ABK1 \cdot ABK2 \cdot ABK3 \cdot (A15 \cdot A14 \cdot A13 + A15 \cdot \bar{IND} \cdot ABK3 \cdot ABK2 \cdot (A15 \cdot ABK1 + A15 \cdot ABK1) \cdot (A14 \cdot A13 + A14 \cdot A13))) \quad (4)$$

$$PCAS5,7 = (AY \cdot \bar{IND} \cdot ABK3 \cdot (ABK1 \cdot ABK2 + ABK1 \cdot ABK2) \cdot (A15 \cdot A14 \cdot A13 + A15 \cdot A14 \cdot A13) + AY \cdot \bar{IND} \cdot ABK3 \cdot (ABK2 \cdot ABK1 \cdot A15 + ABK2 \cdot ABK1 + ABK2 \cdot ABK1 \cdot A15) \cdot A14 + AY \cdot \bar{IND} \cdot ABK1 \cdot ABK2 \cdot ABK3 \cdot (A15 \cdot A14) + AY \cdot \bar{IND} \cdot ABK3 \cdot ABK2 \cdot (A15 \cdot ABK1 + A15 \cdot ABK1) \cdot (A14 \cdot A13 + A14 \cdot A13)) \quad (5)$$

In effect, these ROMs are programmed to allow selection of predetermined rows in the memory, based on the address signals A10, A13, A14 and A15, (ignoring for a moment the contribution of the RAS signals and the other signals appearing in the equations).

The outputs of the CAS ROMs 127 and 128 are coupled to the register 131. Register 131 is a commercially available register which permits the enabling of output signals (Part No. 374). During accessing of the memory the various CAS signals (CAS 0 through CAS 7) are coupled to the memory of FIG. 6 to permit selection of the appropriate memory devices. The signal USELB from CAS ROM 127 through register 131 selects either the A bus or B bus. This signal is coupled to the multiplexers 43a and 43b of FIG. 3.

During normal operation, the multiplexer 130 selects the bank switching signals BCKSW 1 through BCKSW

12

4. These four signals (or alternatively four signals from the A bus) provide four of the inputs (address signals) to the ROM 132. The other inputs to this ROM are the DHIRES, Z PAGE, PA8, PA15, RFSH (refresh), and AY signals. These address signals select the RAS 0, 3; RAS 1, 2; RAS 4, 5 and RAS 6, 7 signals. The ROM 132 is programmed to implement the following four equations.

$$PRAS0,3 = \bar{AY} \cdot (DHIRES + RFSH) + (ABK4 \cdot (Z \text{ Page} \cdot \bar{PA8}) + ABK1 \cdot ABK2 \cdot ABK3) \cdot AY \quad (6)$$

$$PRAS1,2 = \bar{AY} \cdot (DHIRES + RFSH) + AY \cdot (ABK1 \cdot ABK2 \cdot ABK3 \cdot (ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15 + ABK1 \cdot ABK2 \cdot ABK3) + AY \cdot ABK3 \cdot (ABK1 \cdot ABK2 \cdot ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15 + ABK1 \cdot ABK2 \cdot (ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15))) \quad (7)$$

$$PRAS4,5 = RFSH \cdot AY + AY \cdot ABK2 \cdot ABK3 \cdot (ABK1 \cdot ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15 + ABK1 \cdot (ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15)) \quad (8)$$

$$PRAS6,7 = RFSH \cdot AY + AY \cdot ABK3 \cdot (ABK1 \cdot ABK2 \cdot ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15 + ABK1 \cdot ABK2 \cdot (ABK4 \cdot (Z \text{ PAGE} \cdot \bar{PA8}) \cdot PA15)) \quad (8)$$

Thus, the bank switching signals (along with the other input signals to ROM 132) select predetermined rows in memory in conjunction with the CAS signals.

The output signals of the ROM 132 are coupled through the NAND gates 142, 143, 144 and 145 to the memory. The other input terminals of these gates receive the RAS timing signal. In this manner, the output signals of the ROM 132 are clocked through the gates 142 through 145 to provide the RAS signals shown in FIGS. 5 and 6.

An important feature to the presently described computer is provided by the circuitry shown within the dotted line 146. The AND gate 148 receives, at its input terminals, the DA7, A12, and C3 signals. The NOR gate 149 receives the zero page and A15 signal. The output of gate 149 provides one input to the gate 148 and also one input to the AND gate 150. The output of gate 148 provides another input signal to gate 150 and this signal (line 153) is one of the two control signals coupled to the multiplexer 130. The AND gates 150 and 151 also receive a SYNC signal and the φ0 signal. The output of the gates 150 and 151 are coupled to a NOR gate 152 with the output of the gate 152 (line 154) coupled to the other control terminal of the multiplexer 130.

The gates 150, 151 and 152 effectively form a clock for multiplexer/register 130 (multiplexer 130 is a commercial part, Part No. 399, which effectively is a register/multiplexer). This selects the lower four input lines to the multiplexer 130. However, because of the synchronization signal applied to gate 151, the multiplexer 130 selects the bank switching signals each time an OP code is fetched by the CPU.

To understand the operation of the circuit shown within the dotted line 146 it should be recalled that the memory of FIG. 6 provides a 16-bit output. As mentioned, during certain display modes, 16-bits/msec. are needed for display purposes. In nondisplay modes, only 8-bits are required, particularly for interaction with the CPU. When the memory is addressed by the CPU during the indirect addressing modes the data on the A bus is not ordinarily used. However, with the circuitry shown within the dotted line 146, this otherwise "un-

used" data is put to use to provide the equivalent of the bank switching signals through multiplexer 130.

Whenever the CPU selects a predetermined range of addresses, the multiplexer 130 selects the equivalent of the bank switching signals from the A bus provided DA7 is high. (This occurs when addressing as zero page address space -1800 through 1FFF.) Once the signal on line 153 is high it is latched through gates 150, 151 and 152 causing the multiplexer 130 to select the four bits from the A bus (assuming the timing signals are high). Even if the next reference from the CPU is not to this special address range, the multiplexer 130 nonetheless remains latched with the four bits from the data bus. Once the SYN pulse drops, however, which is an indication that an OP code is being fetched, the signal on line 154 rises in potential, causing the multiplexer to switch back to the bank switching signals.

Effectively, what occurs is that when the CPU selects this special address range, (and provided DA7 is high) the bits DA0 through DA3 which are stored in memory, cause a remapping, that is, the address from the CPU accesses a different part of the memory. With the fetching of each OP code, the mapping automatically returns to the bank switching signals. Importantly, the remapping, which occurs is controlled by the bits stored in the RAM (DA ϕ through DA3). Thus, with the remapping information stored in RAM, toggling can occur between different portions of the memory without requiring bank switching signals, or the like from the CPU. This enhances the CPU's performance since CPU time is not used for remapping. Additionally, it provides an easy tool for programming.

For some program languages it is desirable to separate data and the program into separate portions of the memory. For example, the 128K memory can be divided into two 64K memories, one for program and one for data. Switching can occur between these memory portions without the generation of bank switching signals by the CPU with the above described circuit. This arrangement is particularly useful when using the Pascal program language.

DISPLAY SUBSYSTEM

The display subsystem 48 of FIG. 1 receives data from the A bus and B bus and converts the data into video signals which may be used for displaying alphanumeric characters or other images on a standard raster scanned cathode ray tube display. The display subsystem 48 specifically generates on line 197, a standard NTSC color video signal and a video black and white video signal on line 198 (FIG. 8). This display subsystem, in addition to other inputs, receives a synchronization signal, and several clocking signals. For sake of simplicity, the standard color reference signal of 3.579545 MHz is shown as C3.5M. Twice this frequency and four times this frequency are shown as C7M and C14M, respectively.

Before describing the details of the display subsystem 48, a discussion of a prior art display system will be helpful in understanding the present display subsystem. In U.S. Pat. No. 4,136,359, a video display system is described which is implemented in a commercially available computer, Apple-II, sold by Apple Computer, Inc., of Cupertino, Calif. In this system, 4-bit digital words are shifted in parallel into a shift register. These words are then circulated in the shift register at 14 MHz to define a waveform having components at 3.5 MHz. Referring to FIG. 9, line 206, assume that the digital

word 0001 is placed in the shift register and circulated at a rate of 14 MHz. The resultant signal which has a component of 3.5 MHz is shown on line 206. The phase relationship of this component to the 3.5 MHz reference signal determines the color of the resultant video signal. This relationship is changed by changing the 4-bit word placed in the shift register. As explained in the above-referenced patent, if the signal 1000 is placed in the register and circulated, the resultant phase relationship of the 3.5 MHz component results in the color brown, this signal is shown on line 208. With this prior art technique, the luminance was determined by the DC component of the signals such as shown on lines 206 and 208.

The display subsystem 48 of FIG. 1 also uses 4-bit words to generate the various color signals in a manner somewhat similar to the above-described system. Referring to FIG. 8, 4-bit words representative of colors (16 possible colors) are coupled to the bus 180. (The generation of these words shall be described in detail in conjunction with FIG. 7.) Instead of using a shift register which circulates the 4-bit work, the same result is achieved by using a multiplexer 205 which sequentially selects each of the lines of the bus 180. The signals on bus 180 also provide a luminance signal and a black and white video signal with a gray scale.

The 4 lines of the bus 180 are coupled to multiplexer 205; this multiplexer also receives the C7M and the C3.5M timing signals. These two timing signals cause each of the four lines to be sequentially selected and coupled to line 191. (Note that the order in which each of the lines of the bus 180 is selected does not change.)

In effect, the multiplexer operates to serialize the parallel signal from bus 180. Assume for sake of explanation that the digital signals on bus 180 are 1000 as indicated in FIG. 8. The signal on line 191 will then be 10001000 The output of the multiplexer 205 coupled to the input of the inverter 204 also receives in a sequential order, the signals from bus 180, however, in a different order. For the example shown, the input to inverter 204 is 00100010 After inversion, this results in the signal 11011101 . . . on line 192. Effectively, the signals on lines 191 and 192 are added by resistors 199 and 200. The resultant waveform is an AC signal (no DC component) shown in FIG. 9 on line 209. Thus, with the described circuit, a chroma signal is generated, having a predetermined phase relationship to the 3.5 MHz color reference signal. This phase relationship which is varied by changing the signals on bus 180 determines the color of the video signal on line 197.

In the prior art display discussed above, the DC component of the color signal determines the luminance. In the present invention, the signals on bus 180 are coupled to the base of transistor 195, consists of an AC signal from resistors 199 and 200, and the luminance level also determined by the signals on bus 180. These inputs to transistor 195, along with the C3.5M signal, generate a NTSC color signal on line 197 of improved quality when compared to the discussed prior art system.

In some cases, the signals on bus 180 are all binary ones or all binary zeros. When this occurs, there is no AC component from resistors 199 and 200 (no color signal) and the resultant signal on line 197 is either "black" or "white."

The lines of bus 180 are also coupled through resistors to the base of a transistor 196. Each of these resistors have a different value to provide a "weighting" to the binary signal.

4,383,296

15

This weighting is used for non-color displays to provide "gray" shades as opposed to having a display with only black and white. The binary signals on bus 180 drive the transistor 196 to provide a video signal on line 198. RGB is generated with weighted sums of these same five signals.

Referring now to FIG. 7, data from memory is coupled from the A bus and B bus to registers 159 and 158, respectively. These registers are clocked by the 1 MHz clocking signal and its complement, thus permitting the sequential transfer of 8-bit words every 0.5 msec. As will be described, in some display modes the data is transferred at the 2 MHz rate, and in other display modes, at a 1 MHz rate.

The registers 158 and 159 are coupled to an 8 line display bus 160. This display bus transfers data to registers 164 and 173, and also addresses to a memory 162. The registers 164 and 173 and memory 162 are enabled during specific display modes as will be apparent.

The character memory 162, in the presently preferred embodiment, is a random-access memory which stores patterns representative of alpha-numeric characters. Each time the computer is powered up, the character information is transferred from the ROM 50 into the character memory 162 during an initialization period. During character display modes, the signals from the display bus 160 are addresses, identifying particular alpha-numeric characters stored within the character memory 160. The vertical counter signals V_A , V_B , and V_C (previously discussed in conjunction with adder 121 of FIG. 4) identify the particular line in each character which is to be displayed. Thus, the generation of the digital signals representative of each of the characters occurs in an ordinary manner. The 7-bit signal representative of each line of each character (memory output) is coupled to the shift register 167. Through timing signals not shown, either the register 164 or the character memory 162 is selected to allow the shift register 167 to receive either data directly from the A bus or B bus, or alpha-numeric character information from the memory 162.

The 7-bits of information from either memory 162 or register 164 are serialized by the shift register 167 either at a 7 MHz rate or 14 MHz rate, depending upon the display mode. The serialized data is coupled by line 185 to the multiplexer 169, pins 1 and 4. The inverse of this data is also coupled to multiplexer 169, pin 3. Line 185 is also coupled as one input to the multiplexer 166 and to the register 170 (input 1).

The output 1 of register 170 (line 186) is coupled to the multiplexer 169, pin 1; to register 170 (input 2); and to multiplexer 166. Output 2 of register 170 (line 187) is coupled to input 3 of register 170 and also to multiplexer 166. Output 3 of register 170 (line 187) provides a third input to the multiplexer 166. Input 4 of the register 170 receives the output of the multiplexer 169 (line 189). Output 4 of register 120 (line 190) provides one control signal for the multiplexer 171.

The multiplexer 171 selects either the four lines of bus 183 or the four lines of bus 184. The output of multiplexer 171, bus 180, provides the 4-bit signal discussed in conjunction with FIG. 8. During one of the high resolution display modes (AHIRES), the multiplexer 171 is controlled by a timing signal from the output of the gate 178.

The multiplexer 166 selects either the lines of bus 181 or bus 182. The output of this multiplexer provides the signals for the bus 184. In all but the AHIRES display

16

mode, multiplexer 166 selects bus 181. Thus, typically, the multiplexer 171 receives the signals from bus 174.

For purposes of description above, and also for purposes of explaining for some of the display modes below a simplifying assumption has been made. The signals coupled to the bus 180 by multiplexer 171, for most modes, are controlled by the serialized signal on line 190. This serialized signal is in synchronization with the C7M or C14M clocking signals. The multiplexer 205 of FIG. 8, which as described above, does the "spinning" for the parallel digital signal on bus 180, operates in synchronization with the multiplexer 171. In the description above, and except when otherwise noted below, it is assumed that, by way of example, if the multiplexer 171 is coupling all binary ones and zeros onto bus 180, the signal on line 191 will be either ones or zeros. Also for this condition the signal on line 192 will be all binary zeros or ones, and thus, no AC signal is generated at the base of transistor 195. However, as actually implemented, there is a "phase" difference between the clocking of the multiplexer 171 when compared to the sampling of the signals from bus 180 by the multiplexer 205. This results in a first constant AC signal on the gate of transistor 195 even when it appears that all binary ones are on bus 180, and a second constant AC signal when all binary zeros are on the bus 180. Thus, in this specification, when it states that "black" or "white" signals are being generated, instead, as currently implemented, two constant colors are generated on a color display. Where a true black and white is desired, color suppression is introduced such as through the color burst signal.

The circuit of FIG. 7, along with the circuit of FIG. 8, provides the capability for several distinct display modes. The first of these modes provides a display consisting of 40 characters (or spaces) per horizontal line. This requires a data rate of 8-bits/MHz or half the data rate the memory is capable of delivering. In this mode, data is loaded from the A bus during every other 0.5 μ sec period. (B bus is not used during this mode.) This data addresses the character memory 162, and along with the signals V_A , V_B and V_C , provides the appropriate character line (7-bits) to the shift register 167. During this mode, registers 164 and 173 are disabled. The shift register 167 for this mode shifts the data at a data rate of 7 MHz (note $\overline{CH80}$ is high, allowing the 7 MHz signal from gate 175 to control the shift register 167). Each 7-bit signal is shifted serially onto line 185 and then to line 189 since multiplexer 169 selects pin 4. The data is shifted through the register 170 onto line 190. The serial binary signal on line 190 causes the selection of buses 183 or 184.

The four lines of bus 183 during this mode are coupled to +V (register 173 is disabled); therefore the selection of bus 184 provides four binary ones. The selection of bus 184 provides four binary zeros through bus 181. Thus, the serial binary signal on line 190 provides either all binary ones or all binary zeros to bus 180. As discussed, the circuit of FIG. 8 will provide a black and white display with 40 characters per line.

If the inverse and flashing timing means 172 is selected, each time the shift register 167 is loaded, multiplexer 169 shifts between pins 3 and 4. This causes the characters to change from white characters on a black background to black characters on a white background, and so on.

During the 80 character per line display mode, the registers 158 and 159 are each loaded during sequential

4,383,296

17

0.5 μ sec periods (this utilizes the 2 MHz cycle rate previously discussed). The shift register 167 shifts the character data from memory 162 at a 14 MHz rate. The serialized data at the 14 MHz rate is shifted through the register 170 and again controls the multiplexer 171 as previously described. (Note that register 170 is always clocked at the 14 MHz rate.) Flashing again can be obtained as previously discussed.

In another alpha-numeric character display mode, the background of each character may be in one color and the character itself (foreground) in another color. This mode provides 40 characters per line. The character identification (address for RAM 162), is furnished on the A bus to register 159 at a frequency of 1 MHz. The color information (background color and foreground color) is furnished on the B bus as two 4-bit words to register 158. In the manner previously described, the address from register 159 selects the appropriate character from memory 162 and provides this information to shift register 167. The color information from the B bus is transferred to register 173. For purposes of explanation, assume that the 4-bits identifying the color red for the background are on bus 184 (from register 173 and multiplexer 166) and that 4-bits representing the color blue for the foreground are on bus 183. (Note that when register 173 is enabled, the signals from the register override the binary ones and zeros which otherwise appear on the lines of bus 174.) The serial binary signal representative of the character itself on line 190, selects either the color blue from bus 183 for the character itself or the color red from bus 184 for the background. The digital signals representative of these colors are transferred to bus 180 and provide the color data to the circuit of FIG. 8. For black and white displays, a "gray" scale is provided through the weighting circuit associated with transistor 196 of FIG. 8. Again, the multiplexer 169 may, through the timing means 172, alternate between the signal of line 185 and its inverse, which will have the effect of interchanging the foreground and background colors.

During the high resolution graphics modes, the character memory 162 is not used, but rather, data from the memory directly provides pattern information for display. This requires more mapping of data from within the main memory since new data is required for each line of the display. (Note that when characters are displayed, the character memory 162 provides the different signals required for the 8 lines of each character row.) During these high resolution modes, the register 164 is enabled and the character memory 162 is disabled. Thus, the data from the A bus and B bus is shifted into the shift register 167. In these modes, the "HRES" signal to multiplexer 169 causes this multiplexer to select between pins 1 and 2. Pin 2 provides the signal directly from the shift register 167 while the signal on pin 1 is effectively the signal on line 185 delayed by one period of the C14M signal. This delay occurs through the register 170 from input 2 to output 2 since register 170 is clocked at C14M.

18

During a first graphics mode, data from the display bus 160 is loaded into shift register 167 at the rate of 7-bits/MHz. The data is serialized on line 185 and in the manner previously described for displaying characters, controls the selection of all binary ones and all binary zeros through the multiplexer 171. Note, as mentioned before, in the presently preferred embodiment, unless color suppression is used, this will not result in a black and white display, but rather a two-color display. If a high bit is present on line 140 of the display bus, the inverse and flashing timing means 172 causes the multiplexer 169 to alternate between pins 1 and 2. This switching occurs at a 1 MHz rate and provides a phase shift for every other 7-bits of data coupled to the multiplexer 171 on line 190. This results in an additional color being generated on the display for every other 7-bits of data.

For the above-described graphics modes when shift register 161 is shifting at a 7 MHz rate, 8-bits may be coupled to the bus 160 during each period. Specifically, as in the case of the differing background and foreground colors for the 40 character per line display mode, two 4-bit color words are shifted into register 173 at a rate of 1 MHz. Then, the multiplexer 171 selects between two predetermined colors on buses 183 and 184. Note these colors can be changed at a 1 MHz rate.

In an additional color mode identified as "AHIRES," multiplexer 171 operates under the control of gates 176, 177 and 178. In effect, multiplexer 171 selects bus 184 and latches the signals on this bus every four cycles of the C14M clock. Data is shifted into the shift register 167 from the A bus and B bus every 0.5 μ sec the register 167 operates under the control of the C14M signal. Each data bit on line 185 is shifted first to line 186, then to line 187 and finally to line 188. These lines are coupled to the multiplexer 171 through multiplexer 166 which selects bus 182 since AHIRES is high. In effect, what occurs is that 4-bit color words are serialized onto line 185 and then brought back into parallel on bus 182. Since multiplexer 171 latches the signals on bus 184 every four cycles of the C14M signal, a new color word is generated at a 3.5 MHz rate on the bus 180. The resultant display is 140 by 192 colored blocks wherein each block can be any one of 16 colors.

In the last display mode, typically used with color suppression, data is shifted into the shift register 167 from the display bus at the rate of 14-bits/MHz. The data is serialized onto line 185 and controls the selection of either all binary ones or all zeros through multiplexer 171. This provides the highest resolution graphics display for the system.

Thus, a microcomputer with video display capability has been described. The computer is fabricated from commercially available parts and provides high utilization of these parts. Numerous existing programs including many of those which operate on the Apple-II computer, may be employed in the above-described computer.

60

65

T A B L E I

F000:	13	*****		
F000:	14	*	CRITICAL TIMING	*
F000:	15	*	REQUIRES PAGE BOUND	*
F000:	16	*	CONSIDERATIONS FOR	*
F000:	17	*	CODE AND DATA	*
F000:	18	*	-----CODE-----	*
F000:	19	*	VIRTUALLY THE ENTIRE	*
F000:	20	*	'WRITE' ROUTINE	*
F000:	21	*	MUST NOT CROSS	*
F000:	22	*	PAGE BOUNDARIES	*
F000:	23	*	CRITICAL BRANCHES IN	*
F000:	24	*	THE 'WRITE', 'READ',	*
F000:	25	*	AND READ ADDR SUBRS	*
F000:	26	*	WHICH MUST NOT CROSS	*
F000:	27	*	PAGE BOUNDARIES ARE	*
F000:	28	*	NOTED IN COMMENTS	*
F000:	29	*		*
F000:	30	*****		
F000:	31	*		*
F000:	32	*	EQUATES	*
F000:	33	*		*
0200:	34	NBUF1	EQU \$200	
0302:	35	NBUF2	EQU \$302	(ZERO PAGE AT \$300)
F000:	36	*		
00B0:	37	HRDERRS	EQU \$80	
00E0:	38	DVMOT	EQU \$E0	
F000:	39	*		
00E1:	40	IBSLOT	EQU \$81	
00B2:	41	IBPRN	EQU IBSLOT+1	
00B3:	42	IBTRK	EQU IBSLOT+2	
00B4:	43	IBSECT	EQU IBSLOT+3	
00B5:	44	IDBUFF	EQU IBSLOT+4	; 85
00B7:	45	IBCMD	EQU IBSLOT+6	
00B8:	46	IBSTAT	EQU IBSLOT+7	
00B9:	47	IBSMOD	EQU IBSLOT+8	
00B9:	48	CSUM	EQU IBSMOD	USED ALSO FOR ADDRESS HEADER CKSUM
00BA:	49	IOBPDN	EQU IBSLOT+9	
00BB:	50	IMASK	EQU IBSLOT+\$A	
00BC:	51	CURTRK	EQU IBSLOT+\$B	
00B5:	52	DRVOTRK	EQU CURTRK-7	
F000:	53	;SLOT 4, DRIVE 1		
F000:	54	;SLOT 4, DRIVE 2		
F000:	55	;SLOT 5, DRIVE 1		
F000:	56	;SLOT 5, DRIVE 2		
F000:	57	;SLOT 6, DRIVE 1		
F000:	58	;SLOT 6, DRIVE 2		
0093:	59	RETRYCNT	EQU IBSLOT+\$12	
0094:	60	SEEKCNT	EQU IBSLOT+\$13	
009B:	61	BUF	EQU IBSLOT+\$1A	
009F:	62	ENVTEMP	EQU IBSLOT+\$1E	
F000:	63	*IBSLOT+\$1F	NOT USED.	
F000:	64	*		
F000:	66	*****		
F000:	67	*		*
F000:	68	*	-----READADR-----	*
F000:	69	*		*
F000:	70	*****		
0095:	71	COUNT	EQU IBSLOT+\$14 ; 'MUST FIND' COUNT.	
0095:	72	LAST	EQU IBSLOT+\$14 ; 'ODD BIT' NIBLS.	
0096:	73	CKSUM	EQU IBSLOT+\$15 ; CHECKSUM BYTE.	
0097:	74	CSSTV	EQU IBSLOT+\$16 ; FOUR BYTES,	
F000:	75	*	CHECKSUM, SECTOR, TRACK, AND VOLUME.	
F000:	76	*		
F000:	77	*****		
F000:	78	*		*

APPLE III BOOT ROM LISTING
 REVISION D ROM (See addr F1B9)

	21	4,383,296	22
F000:	79	* -----WRITE-----	*
F000:	80	*	*
F000:	81	* USES ALL NBUFS	*
F000:	82	* AND 32-BYTE	*
F000:	83	* DATA TABLE 'NIBL'	*
F000:	84	*	*
F000:	85	*****	
F000:	86	*	
F000:	87	*****	
F000:	88	*	*
F000:	89	* -----READ-----	*
F000:	90	*	*
F000:	91	* USES ALL NBUFS	*
F000:	92	* USES LAST 54 BYTES	*
F000:	93	* OF A CODE PAGE FOR	*
F000:	94	* SIGNIFICANT BYTES	*
F000:	95	* OF DNIBL TABLE.	*
F000:	96	*	*
F000:	97	*****	
F000:	98	*	
F000:	99	*****	
F000:	100	*	*
F000:	101	* ---- SEEK ----	*
F000:	102	*	*
F000:	103	*****	
0095:	104	TRKCNT EQU COUNT ; HALFTRKS MOVED COUNT.	
009D:	105	PRIOR EQU IB SLOT+\$1C	
009E:	106	TRKN EQU IB SLOT+\$1D	
F000:	107	*	
F000:	108	*****	
F000:	109	*	*
F000:	110	* ---- MSWAIT ----	*
F000:	111	*	*
F000:	112	*****	
0099:	113	MONTIMEL EQU CSSTV+2 ; MOTOR-ON TIME	
009A:	114	MONTIMEH EQU MONTIMEL+1 ; COUNTERS.	
F000:	115	*	
F000:	117	*****	
F000:	118	*	*
F000:	119	* DEVICE ADDRESS	*
F000:	120	* ASSIGNMENTS	*
F000:	121	*	*
F000:	122	*****	
C080:	123	PHASEOFF EQU \$C080 ; STEPPER PHASE OFF.	
C081:	124	PHASEON EQU \$C081 ; STEPPER PHASE ON.	
C08C:	125	Q6L EQU \$C08C ; Q7L, Q6L=READ	
C08D:	126	Q6H EQU \$C08D ; Q7L, Q6H=SENSE WPROT	
C08E:	127	Q7L EQU \$C08E ; Q7H, Q6L=WRITE	
C08F:	128	Q7H EQU \$C08F ; Q7H, Q6H=WRITE STORE	
FFEF:	129	INTERUPT EQU \$FFEF	
FFDF:	130	ENVIRON EQU \$FFDF	
0080:	131	ONEMEG EQU \$80	
007F:	132	TWOMEG EQU \$7F	
F000:	133	*****	
F000:	134	*	
F000:	135	* EQUATES FOR RWTS AND BLOCK	
F000:	136	*	
F000:	137	*****	
C088:	138	MOTOROFF EQU \$C088	

4,383,296

23

24

COB9:	139	MOTORON	EQU	\$COB9	
COBA:	140	DRVOEN	EQU	\$COBA	
COBB:	141	DRV1EN	EQU	\$COBB	
COB1:	142	PHASON	EQU	\$COB1	
COB0:	143	PHSOFF	EQU	\$COB0	
0097:	144	TEMP	EQU	CSSTV	; PUT ADDRESS INFO HERE
0097:	145	CSUM1	EQU	TEMP	
0098:	146	SECT	EQU	CSUM1+1	
0099:	147	TRACK	EQU	SECT+1	
0099:	148	TRKN1	EQU	TRACK	
009A:	149	VOLUME	EQU	TRACK+1	
0083:	150	IBRERR	EQU	HRDERRS+3	
0082:	151	IBDERR	EQU	HRDERRS+2	
0081:	152	IBWPER	EQU	HRDERRS+1	
0080:	153	IBNODRV	EQU	HRDERRS	
F000:	155	*****			
F000:	156	*			
F000:	157	* HEAD WRITE A		*	
F000:	158	* TRACK AND SECTOR		*	
F000:	159	*		*	
F000:	160	*****			
F000:	161	*			
F000 A9 01	162	REGRWTS	LDY #1		; RETRY COUNT
F002 A6 81	163		LDX 1BSLOT		; GET SLOT # FOR THIS OPERATION
F004 84 94	164		STY SFEKCNT		; ONLY ONE RECALIBRATE PER CALL
F006 08	165		PHP		; DETERMINE INTERRUPT STATUS
F007 68	166		PLA		
F008 6A	167		ROR A		
F009 6A	168		ROR A		; GET INTERRUPT FLAG INTO BIT 7
F00A 6A	169		ROR A		
F00B 6A	170		ROP A		
F00C 85 8B	171		STA IMASK		
F00E AD DF FF	172		LDA ENVIRON		; PRESERVE ENVIRONMENT
F011 85 9F	173		STA ENVTEMP		
F013:	174	*			
F013:	175	* NOW CHECK IF THE MOTOR IS ON. THEN START IT			
F013:	176	*			
F013 20 2B F1	177		JSP CH-DRV		; SET ZERO FLAG IF MOTOR STOPPED
F016 08	178		PHP		; SAVE TEST RESULTS
F017 A5 85	179		LDA IBBUFF		; MOVE OUT POINTER TO BUFFER INTO ZPAGE
F019 85 9B	180		STA BUF		
F01B A5 8A	181		LDA IBBUFF+1		
F01D 85 9C	182		STA BUF+1		
F01F A5 80	183		LDA #DVMOT		
F021 85 9A	184		STA MONTIMEH		
F023 A5 82	185		LDA IBDRVN		; DETERMINE DRIVE ONE OR TWO
F025 C5 8A	186		CMPI IOBPDN		; SAME DRIVE USED BEFORE?
F027 85 8A	187		STA IOBPDN		; SAVE IT FOR NEXT TIME
F029 08	188		PHP		; KEEP RESULTS OF COMPARE
F02A 8A	189		ROR A		; GET DRIVE NUMBER INTO CARRY
F02B 8D 89 00	190		LDA MOTORON.X		; TURN ON THE DRIVE
F02E 90 01	191		BCC DRIVSEL		; BRANCH IF DRIVE 1 SELECTED
F030 E8	192		INX		; SELECT DRIVE 2
F031 8D 8A 00	193	DRIVSEL	LDA DRVOEN.X		
F034 20 40 FF	194		JSR SETIMEG		; INSURE ONE MEGAHERTZ OPERATION
F037 28	195		PLP		; WAS IT SAME DRIVE?
F038 F0 0A	196		BEQ OK		
F03A 28	197		PLP		; MUST INDICATE DRIVE OFF BY SETTING ZERO
F03B A0 07	198		LDY #7		; DELAY 150 MS BEFORE STEPPING (FLAG)
F03D 20 56 F1	199	DRVWAIT	JSR MSWAIT		; (ON RETURN A=0)
F040 86	200		DEY		
F041 00 FA	201		BNE DRWAIT		
F043 08	202		PHP		; NOW ZERO FLAG SET
F044 A5 83	203	OK	LDA IBTRK		; GET DESTINATION TRACK
F046 A6 81	204		LDX 1BSLOT		; RESTORE PROPER X (SLOT*16)
F048 20 05 F1	205		JSR MYSEEK		; AND GO TO IT
F04B:	206	*NOW AT THE DESIRED TRACK			; WAS THE MOTOR
F04B:	207	* ON TO START WITH?			
F04B 28	208		PLP		; WAS MOTOR ON?
F04C D0 17	209		BNE TRYTRK		; IF SO, DON'T DELAY, GET IT TODAY!
F04E:	210	*			

```

F04E      211 * MOTOR WAS OFF, WAIT FOR IT TO SPEED UP
F04E      212 *
F04E:A0 12 213 MOTOF   LDY  #*12           ; WAIT EXACTLY 100 US FOR EACH COUNT
F050:88      214 CONWAIT DEY                ; IN MONTIME
F051:D0 F1 215           BNE  CONWAIT
F053:E4 89 216           INC  MONTIMEH ; COUNT UP TO 0000
F055:D4 F1 217           BNE  MOTOF
F057:C4 9A 218           INC  MONTIMEH
F059:D4 F0 219           BNE  MOTOF
F05B:      221 *****
F05B:      222 *
F05B:      223 * MOTOR SHOULD BE UP TO SPEED
F05B:      224 * IF IT STILL LOOKS STOPPED THEN
F05B:      225 * THE DRIVE IS NOT PRESENT
F05B:      226 *
F05B:      227 *****
F05B:20 2B F1 228           JSR  CHKDRV   ; IS DRIVE PRESENT?
F05E:D0 05 229           BNE  TRYTRK   ; YES, CONTINUE
F060:A9 80 230 MDRIVEPR PLA #1BNDRV ; NO, GET TELL EM NO DRIVE
F062:4C EB F1 231           JMP  HNDLERR
F065:      232 *
F065:      233 * NOW CHECK IF IT IS NOT THE FORMAT DISK COMMAND
F065:      234 * LOCATE THE CORRECT SECTOR FOR THIS OPERATION
F065:      235 *
F065:A5 87 236 TRYTRK  LDA  IBCMD   ; GET COMMAND CODE #
F067:F0 77 237           BEQ  ALLDONE  ; IF NULL COMMAND, GO HOME TO BED
F069:C9 03 238           CMP  #3         ; COMMAND IN RANGE?
F06B:8C 73 239           BCS  ALLDONE  ; NO, DO NOTHING!
F06D:6A      240           ROR  A         ; SET CARRY=1 FOR READ, 0 FOR WRITE
F06E:8C 0B 241           ROR  TRYTRK2 ; MUST PREINIBLIZE FOR WRITE
F070:AD DF F1 242           LLA  ENVIRON  ;
F073:29 7E 243           AND  #1WMEG  ; SHIFT TO HIGH SPEED!
F075:8D DF F1 244           STA  ENVIRON  ;
F078:20 C6 F1 245           JSR  PREINIB1 ;
F07B:A0 7E 246 TRYTRK2 LDY  #127     ; ONLY 127 RETRIES OF ANY KIND
F07D:84 93 247           DEC  CTR     ;
F07F:A6 81 248 TRYTRK2  LDA  IBCMD   ; GET SLOT NUM INTO X-REG
F081:20 BD F1 249           BEQ  #ADR16  ; READ NEXT ADDRESS FIELD
F084:90 21 250           BCC  RDRIGHT ; IF READ IT RIGHT, HURRAH!
F086:24 8B 251 TRYADR2  BIT  IMASK   ; SHOULD INTERRUPTS BE ALLOWED?
F088:3C 01 252           DMI  NOINTR1 ; NO, DON'T ALLOW THEM.
F08A:58      253           CLE  A         ; RE-ENABLED AFTER READ/READADR/WRITE
F08C:04 93 254 TRYTRK2  BEQ  RETRYCNT ; ANOTHER MISTAKE!! FAILURE
F08E:13 F0 255           BPL  TRYADR  ; WELL, LET IT GO THIS TIME
F08F:A5 8C 256           LDA  CURTRK  ;
F091:48      257           PHA                ; SAVE TRACK WE REALLY WANT
F092:C6 94 258           DEC  SEEKCNT ; ONLY RECALIBRATE ONCE!
F094:D0 4F 259           BNE  DRVERR  ; TRIED TO RECALIBRATE A SECOND TIME,
F096:A4 89 260           LDA  #*60   ; RECALIBRATE ALL OVER AGAIN! ERROR!
F098:20 27 F1 261           JSR  SETTRK  ; PRETEND TO BE ON TRACK 80
F09B:A9 00 262           LDA  #*00   ;
F09D:20 05 F1 263           JSR  MYSEEK  ; MOVE TO TRACK 00
FOA0:68      264 GDRAL1  PLA                ;
FOA1:21 05 F1 265 GDRAL  JSR  MYSEEK  ; GO TO CORRECT TRACK THIS TIME!
FOA4:4C 7B F1 266           JMP  TRYTRK2 ; LOOP BACK, TRY AGAIN ON THIS TRACK
FOA7:      267 *
FOA7:      268 * HAVE NOW READ AN ADDRESS FIELD CORRECTLY.
FOA7:      269 * MAKE SURE THIS IS THE TRACK, SECTOR, AND VOLUME DESIRED.
FOA7:A4 99 270 RDRIGHT LDY  TRACK   ; ON THE RIGHT TRACK?
FOA9:C4 8C 271           CPY  CURTRK  ;
FOAB:F0 0E 272           BEQ  RETTRK  ; IF SO, GOOD
FOAD:      273 * RECALIBRATING FROM THIS TRACK
FOAD:A5 8C 274           LDA  CURTRK  ; PRESERVE DESTINATION TRACK
FOAF:48      275           PHA                ;
FOB0:98      276           TYA                ;
FOB1:20 25 F1 277           JSR  SETTRK  ;
FOB4:68      278           PLA                ;
FOB5:20 09 F1 279           JSR  MYSEEK  ;
FOB8:4C 86 F0 280           JMP  TRYADR2  ; GO AHEAD AND RECALIBRATE
FOBB:      282 *
FOBB:      283 * DRIVE IS ON RIGHT TRACK, CHECK VOLUME MISMATCH
FOBB:      284 *
FOBB:A5 9A 285 RTTRK  LDA  VOLUME  ; GET ACTUAL VOLUME HERE
FOBD:85 89 286           STA  IBSMOD  ; TELL OPSYS WHAT VOLUME WAS THERE
    
```

4,383,296

27

28

F0BF: A5 98	287	CORRECTVOL	LDA SECT	;	CHECK IF THIS IS THE RIGHT SECTOR
F0C1: C5 84	288		CMP IBSECT		
F0C3: F0 02	289		BEQ CORRECTSECT	;	IF SO, DO WHATEVER WANTED
F0C5: D0 BF	290		BNE TRYADR2	;	NO, TRY ANOTHER SECTOR
F0C7: A5 87	291	CORRECTSECT	LDA IBCMD	;	READ OR WRITE?
F0C9: 4A	292		LSR A	;	THE CARRY WILL TELL
F0CA: 90 2D	293		BCC WRIT	;	CARRY WAS SET FOR READ OPERATION.
F0CC: 20 4B F1	294		JSR READ16	;	CLEARED FOR WRITE
F0CF: 80 B5	295		BCS TRYADR2	;	CARRY SET UPON RETURN IF BAD READ
F0D1: AD DF FF	296		LDA ENVIRON		
F0D4: 29 7F	297		AND #TWOMEG		
F0D6: 95 DF FF	298		STA ENVIRON	;	SET TWO MEGAHERTZ MODE
F0D9: 20 11 F3	299		JSR POSTNIB16	;	DO PARTIAL POSTNIBBLE CONVERSION
F0DC: 80 AB	300		BCS TRYADR2	;	CHEKSUM ERROR
F0DE: A5 81	301		LDX IBSLOT	;	RESTORE SLOTNUM INTO X
F0E0: 18	302	ALLDONE	CLC		
F0E1: A9 00	303		LDA #0	;	NO ERROR
F0E3: 90 04	304		BCC ALDONE1	;	SKIP OVER NEXT BYTE WITH B11 OPCODE
F0E5: 68	305	DRVERR	PLA	;	REMOVE CURTRK
F0E6: A9 82	306		LDA #IBDERR	;	BAD DRIVE
F0E8: 38	307	HNDLERR	SEC	;	INDICATE AN ERROR
F0E9: 85 88	308	ALDONE1	STA IBSTAT	;	GIVE HIM ERROR#
F0EB: 8D 88 00	309		LDA MOTOROFF,X	;	TURN IT OFF
F0EE: 24 88	310		BIT IMASK	;	SHOULD INTERRUPTS BE ENABLED?
F0F0: 30 01	311		BMI NOINTR2	;	BRANCH IF NOT
F0F2: 58	312		CLI		
F0F3: A5 9F	313	NOINTR2	LDA ENVTEMP	;	RESTORE ORIGINAL ENVIRONMENT
F0F5: 8D DF FF	314		STA ENVIRON		
F0F8: 50	315		RTS		
F0F9: 20 19 F1	316	WRIT	JSR WRITE16	;	WRITE NYBBLES NOW
F0FC: 90 E2	317		BCC ALDONE	;	IF NO ERRORS
F0FE: A9 81	318		LDA #IBWPER	;	DISK IS WRITE PROTECTED!
F100: 50 E6	319		BVC HNDLERR	;	TAKEN IF TRULY WRITE PROTECT ERROR
F102: 40 84 FF	320		JMP TRYADR2	;	OTHERWISE ASSUME AN INTERRUPT MESSD
F105:	321	*			THINGS UP
F105:	322	*			
F105:	323	*			SEEK ROUTINE
F105:	324	*			SEEKS TRACK 'N' IN SLOT #X/*10
F105:	325	*			IF DRIVNO IS NEGATIVE, ON DRIVE 0
F105:	326	*			IF DRIVNO IS POSITIVE, ON DRIVE 1
F105:	327	*			
F105: 0A	327	MYSEEK	ASL A	;	ASSUME TWO PHASE STEPPER
F106: 85 99	328	SEV1	STA TRKN1	;	SAVE DESTINATION TRACK(*2)
F108: 20 19 F1	329		JSR ALLOFF	;	TURN ALL PHASES OFF TO BE SURE
F10B: 20 3E F1	330		JSR DRVINDX	;	GET INDEX TO PREVIOUS TRACK FOR CURRENT DRIVE
F10E: 85 85	331		LDA DRVOTRK,X		
F110: 85 80	332		STA CURTRK	;	THIS IS WHERE I AM
F112: A5 99	333		LDA TRKN1	;	AND WHERE I'M GOING TO
F114: 95 85	334		STA DRVOTRK,X		
F116: 20 00 F4	335	GOSEEK	JSR SEEK	;	GO THERE!
F119: A0 03	336	ALLOFF	LDY #3	;	TURN OFF ALL PHASES BEFORE RETURNING
F11B: 98	337	NXOFF	TYA	;	(SEND PHASE IN ACC.)
F11C: 20 4A F4	338		JSR CLRPHASE	;	CARRY IS CLEAR, PHASES SHOLD BE TURNED OFF
F11F: 86	339		DEY		
F120: 10 F9	340		BPL NXOFF		
F122: 46 BC	341		LSR CURTRK	;	DIVIDE BACK DOWN
F124: 60	342		RTS	;	ALL OFF... NOW IT'S DARK
F125:	344	*			
F125:	345	*			THIS SUBROUTINE SETS THE SLOT DEPENDENT TRACK
F125:	346	*			LOCATION
F125:	347	*			
F125: 20 3E F1	348	SETTRK	JSR DRVINDX	;	GET INDEX TO DRIVE NUMBER.
F128: 95 85	349		STA DRVOTRK,X		
F12A: 60	350		RTS		
F12B:	351	*			*****
F12B:	352	*			
F12B:	353	*			SUBR TO TELL IF MOTOR IS STOPPED
F12B:	354	*			
F12B:	355	*			IF MOTOR IS STOPPED, CONTROLLER'S
F12B:	356	*			SHIFT REG WILL NOT BE CHANGING.
F12B:	357	*			
F12B:	358	*			RETURN Y=0 AND ZERO FLAG SET IF IT IS STOPPED.

4,383,296

29

30

```

F12B:          359 *
F12B:          360 *****
F12B: A0 00    361 CHKDRV LDY #0          ; INIT LOOP COUNTER
F12D: BD BC CO 362 CHKDRV1 LDA Q6L, X    ; READ THE SHIFT REG
F130: 20 3D F1 363          JSR CKDRTS    ; DELAY
F132: 48      364          PHA
F134: 68      365          PLA          ; MORE DELAY
F136: 0E 80 C0 366          CMP Q6L, X    ; HAS SHIFT REG CHANGED?
F138: D0 03    367          BNE CKDRTS    ; YES, MOTOR IS MOVING
F13A: 88      368          DEY          ; NO, DEC RETRY COUNTER
F13B: D0 F0    369          BNE CHKDRV1   ; AND TRY 256 TIMES
F13C: 80      370 CKDRTS  RTS          ; THEN RETURN
F13E:          371 *
F13F: 48      372 DRVINDX PHA          ; PRESERVE ACC.
F13F: 8A      373          TXA          ; GET SLOT(**10)/8
F140: 4A      374          LSR A
F141: 4A      375          LSR A
F142: 4A      376          LSR A
F143: 05 8D    377          ORA IBDRVN    ; FOR DRIVE 0 OR 1
F145: AA      378          TAX          ; INTO X FOR INDEX TO TABLE
F146: 68      379          PLA          ; RESTORE ACC.
F147: 60      380          RTS
F148:          381 *****
F148:          382 *
F148:          383 * NOTE: FORMATTING ROUTINES
F148:          384 *         NOT INCLUDED FOR SOS
F148:          385 *
F148:          386 *****
F148:          388 *****
F148:          389 *
-----
F148:          390 *         READ SUBROUTINE *
F148:          391 *         (16-SECTOR FORMAT) *
F148:          392 *
-----
F148:          393 *****
F148:          394 *
-----
F148:          395 *         READS ENCODED BYTES *
F148:          396 *         INTO NBUF1 AND NBUF2 *
F148:          397 *
-----
F148:          398 *         FIRST READS NBUF2 *
F148:          399 *         HIGH TO LOW, *
F148:          400 *         THEN READS NBUF1 *
F148:          401 *         LOW TO HIGH. *
-----
F148:          402 *
-----
F148:          403 *         ---- ON ENTRY ---- *
F148:          404 *
-----
F148:          405 *         X-REG: SLOTNUM *
F148:          406 *         TIMES *10. *
F148:          407 *
-----
F148:          408 *         READ MODE (Q6L, Q7L) *
F148:          409 *
-----
F148:          410 *         ---- ON EXIT ---- *
F148:          411 *
-----
F148:          412 *         CARRY SET IF ERROR. *
F148:          413 *
-----
F148:          414 *         IF NO ERROR: *
F148:          415 *         A-REG HOLDS $AA. *
F148:          416 *         X-REG UNCHANGED. *
F148:          417 *         Y-REG HOLDS $00. *
F148:          418 *         CARRY CLEAR. *
F148:          419 *         ---- CAUTION ---- *
F148:          420 *

```

4,383,296

		31			32
F148:		421 *	OBSERVE		*
F148:		422 *	'NO PAGE CROSS'		*
F148:		423 *	WARNINGS ON		*
F148:		424 *	SOME BRANCHES!!		*
F148:		425 *			*
F148:		426 *	---- ASSUMES ----		*
F148:		427 *			*
F148:		428 *	1 USEC CYCLE TIME		*
F148:		429 *			*
F148:		430	*****		*
F148	AO 20	431	READ16 LDY ##20		'MUST FIND' COUNT.
F14A	8B	432	RSYNC DEY		IF CAN'T FIND MARKS
F14B	FO 6B	433	BEG RDERR		THEN EXIT WITH CARRY SET
F14D	BD BC CO	434	RD1 LDA Q6L,X		READ NIBL.
F150:	10 FB	435	BPL RD1		*** NO PAGE CROSS! ***
F152:	49 D5	436	RSYNC1 EOR ##D5		DATA MARK 1?
F154:	DO F4	437	BNE RSYNC		LOOP IF NOT.
F156	EA	438	NOP		DELAY BETWEEN NIBLS.
F157	BD BC CO	439	RD2 LDA Q6L,X		
F15A	10 FB	440	BPL RD2		*** NO PAGE CROSS! ***
F15C:	C9 AA	441	CMP ##AA		DATA MARK 2?
F15E:	DO F2	442	BNE RSYNC1		(IF NOT, IS IT DM1?)
F160:	AO 55	443	LDY ##55		INIT NBUF2 INDEX.
F162:		444 *			(ADDED NIBL DELAY)
F162	BD BC CO	445	RD3 LDA Q6L,X		
F165	10 FB	446	BPL RD3		*** NO PAGE CROSS! ***
F167	C9 AD	447	CMP ##AD		DATA MARK 3?
F169	DO E7	448	BNE RSYNC1		(IF NOT, IS IT DM1?)
F16B:		449 *			(CARRY SET IF DM3)
F16B	BD BC CO	450	RD4 LDA Q6L,X		
F16E	10 FB	451	BPL RD4		*** NO PAGE CROSS! ***
F170	99 02 03	452	STA NBUF2,Y		STORE BYTES DIRECTLY
F173	AD EF FF	453	LDA INTERRUPT		POLL INTERRUPT LINE
F176	05 8B	454	ORA IMASK		(THIS MAY BE USED TO INVALIDATE POLL
F17B	10 40	455	BPL GOSERV		
F17A	8B	456	DEY		INDEX TO NEXT
F17B	10 EE	457	BPL RD4		
F17D:	C9	458	RD5 INY		(FIRST TIME Y=0)
F17E	BD BC CO	459	RD5A LDA Q6L,X		GET ENCODED BYTES OF NBUF1
F181	10 FB	460	BPL RD5A		
F183	99 00 03	461	STA NBUF1,Y		
F186	AD EF FF	462	LDA INTERRUPT		POLL INTERRUPT LINE
F189	05 8B	463	ORA IMASK		(THIS MAY BE USED TO INVALIDATE POLL
F18B	10 2D	464	DPL GOSERV		
F18D	CO E4	465	CPY ##E4		WITHIN 1 MS OF COMPLETION?
F18F	DO EC	466	BNE RD5		
F191	C9	467	INY		
F192	BD BC CO	468	RD6 LDA Q6L,X		NO POLL FROM NOW ON
F195	10 FB	469	BPL RD6		
F197	99 00 03	470	STA NBUF1,Y		
F19A	C8	471	INY		FINISH OUT NBUF1 PAGE
F19B	DO F5	472	BNE RD6		
F19D	BD BC CO	473	RD6SUM LDA Q6L,X		GET CHECKSUM BYTE
F1A0	10 FB	474	BPL RD6SUM		
F1A2	85 96	475	STA CKSUM		
F1A4	EA	476	NOP		EXTRA DELAY BETWEEN BYTES
F1A5	BD BC CO	477	RD7 LDA Q6L,X		
F1A8	10 FB	478	BPL RD7		*** NO PAGE CROSS! ***
F1AA	C9 DE	479	CMP ##DE		FIRST BIT SLIP MARK?
F1AC	DO OA	480	BNE RDERR		(ERR IF NOT)
F1AE	EA	481	NOP		DELAY BETWEEN NIBLS.
F1AF	BD BC CO	482	RDB LDA Q6L,X		
F1B2	10 FB	483	BPL RDB		*** NO PAGE CROSS! ***
F1B4	C9 AA	484	CMP ##AA		SECOND BIT SLIP MARK?
F1B6	FO 5F	485	BEG RDEXIT		(DONE IF IT IS)
F1B8	39	486	RDERR SEC		INDICATE 'ERROR EXIT'
F1B9	60	487	RTS		RETURN FROM READ16 OR RDADR16.
F1BA		488 *			
F1BA	4C B3 F2	489	GOSERV JMP SERVICE		GO SERVICE INTERRUPT



ADDRESS F1B9 SPECIFICS
 ROM REVISION: 60
 REV 0 — 60
 REV 1 — AD

4,383,296

33

34

```

F1BD: 491 *****
F1BD: 492 *
F1BD: 493 * READ ADDRESS FIELD *
F1BD: 494 * SUBROUTINE *
F1BD: 495 * (16-SECTOR FORMAT) *
F1BD: 496 *
F1BD: 497 *****
F1BD: 498 *
F1BD: 499 * READS VOLUME, TRACK *
F1BD: 500 * AND SECTOR *
F1BD: 501 *
F1BD: 502 * ---- ON ENTRY ---- *
F1BD: 503 *
F1BD: 504 * XREG: SLOTNUM TIMES $10 *
F1BD: 505 *
F1BD: 506 * READ MODE (Q6L Q7L) *
F1BD: 507 *
F1BD: 508 * ---- ON EXIT ---- *
F1BD: 509 *
F1BD: 510 * CARRY SET IF ERROR *
F1BD: 511 *
F1BD: 512 * IF NO ERROR *
F1BD: 513 * A-REG HOLDS $AA *
F1BD: 514 * Y-REG HOLDS $00. *
F1BD: 515 * X-REG UNCHANGED. *
F1BD: 516 * CARRY CLEAR. *
F1BD: 517 *
F1BD: 518 * CSSTV HOLDS CHKSUM, *
F1BD: 519 * SECTOR, TRACK, AND *
F1BD: 520 * VOLUME READ. *
F1BD: 521 *
F1BD: 522 * USES TEMPS COUNT, *
F1BD: 523 * LAST, CSUM, AND *
F1BD: 524 * 4 BYTES AT CSSTV *
F1BD: 525 *
F1BD: 526 * ---- EXPECTS ---- *
F1BD: 527 *
F1BD: 528 * ORIGINAL 10-SECTOR *
F1BD: 529 * NORMAL DENSITY NIBLS *
F1BD: 530 * (4-BIT), ODD BITS, *
F1BD: 531 * THEN EVEN *
F1BD: 532 *
F1BD: 533 * ---- CAUTION ---- *
F1BD: 534 *
F1BD: 535 * OBSERVE *
F1BD: 536 * 'NO PAGE CROSSES' *
F1BD: 537 * WARNINGS ON *
F1BD: 538 * SOME BRANCHES!! *
F1BD: 539 *
F1BD: 540 * ---- ASSUMES ---- *
F1BD: 541 *
F1BD: 542 * 1 USEC CYCLE TIME *
F1BD: 543 *
F1BD: 544 *****
F1BD: A0 FC 545 RDADR16 LDY #$FC
F1BF: 84 95 546 STY COUNT ; 'MUST FIND' COUNT.
    
```


4,383,296

35

36

F101 C8	547 RDASYN	INX		
F102 D0 04	548	BNE RDA1		LOW ORDER OF COUNT
F104 E6 95	549	INC	COUNT	(2K NIBLS TO FIND
F106 FC F0	550	BEQ	RDERR	ADR MARK, ELSE ERR)
F108 BD BC 00	551 RDA1	LDA	Q6L,X	READ NIBL
F10B 10 FB	552	BPL	RDA1	*** NO PAGE CROSS! ***
F10C 09 5	553 RDASN1	CMR	#\$D5	ADR MARK 1?
F10F D0 F0	554	BNE	RDASYN	(LOOP IF NOT)
F1D1 EA	555	NOP		ADDED NIBL DELAY
F1D2 BD BC 00	556 RDA2	LDA	Q6L,X	
F1D5 10 FB	557	BPL	RDA2	*** NO PAGE CROSS! ***
F1D7 09 AA	558	CMR	#\$AA	ADR MARK 2?
F1D9 00 F2	559	BNE	RDASN1	(IF NOT, IS IT AM)
F1DB A0 03	560	LDY	#\$3	INDEX FOR 4-BYTE READ
F1DD	561 *			(ADDED NIBL DELAY)
F1DD BD BC 00	562 RDA3	LDA	Q6L,X	
F1E0 10 FB	563	BPL	RDA3	*** NO PAGE CROSS! ***
F1E2 09 94	564	CMR	#\$94	ADR MARK 3?
F1E4 D0 E7	565	BNE	RDASN1	(IF NOT, IS IT AM)
F1E6	566 *			(LEAVES CARRY SET!)
F1E6 A9 00	567	LDA	#\$0	INIT CHECKSUM
F1E8 B5 B9	568 RDAFLD	STA	CSUM	
F1EA BD BC 00	569 RDA4	LDA	Q6L,X	READ (ODD BIT) NIBL
F1ED 10 FB	570	BPL	RDA4	*** NO PAGE CROSS! ***
F1EF 0A	571	ROL	A	ALIGN ODD BITS, 17 INTO LSB
F1F0 B5 95	572	STA	LAST	(SAVE THEM)
F1F2 BD BC 00	573 RDA5	LDA	Q6L,X	READ (EVEN BIT) NIBL
F1F5 10 FB	574	BPL	RDA5	*** NO PAGE CROSS! ***
F1F7 15 95	575	ANR	LAST	MERGE ODD AND EVEN BITS
F1F9 99 97 00	576	STA	CSUM	STORE DATA BYTE
F1FB 45 39	577	ENR	CSUM	
F1FE B8	578	DEY		
F1FF 10 E7	579	BPL	RDAFLD	LOOP ON 4 DATA BYTES
F201 A8	580	TAY		IF FINAL CHECKSUM
F202 B5 B4	581	ROL	RDERR	(NONZERO, THEN ERROR)
F204 B0 B1 00	582 RDA6	LDA	Q6L,X	FIRST BIT-SLIP NIBL
F207 10 FB	583	BPL	RDA6	*** NO PAGE CROSS! ***
F209 09 E7	584	CMR	#\$D0	
F20B D0 AB	585	BNE	RDERR	ERROR IF NONMATCH
F20D 78	586	SET		DELAY (NO INTERRUPTS FROM NOW ON)
F20E BD BC 00	587 RDA7	LDA	Q6L,X	SECOND BIT-SLIP NIBL
F211 10 FB	588	BPL	RDA7	*** NO PAGE CROSS! ***
F213 09 AA	589	CMR	#\$AA	
F215 D0 A1	590	BNE	RDERR	ERROR IF NONMATCH
F217 18	591 RDEXT	CLC		CLEAR CARRY ON
F218 50	592 WEX10	RTS		NORMAL READ EXITS
F219	593	ENR	RWT82	
F219:	2	*****		
F219:	3	*		*
F219	4	*	WRITE SUBR	*
F219	5	*	(16-SECTOR FORMAT)	*
F219	6	*		*
F219	7	*****		
F219	8	*		*
F219	9	*	WRITES DATA FROM	*
F219	10	*	NBUF1 AND NBUF2	*
F219	11	*		*
F219	12	*	FIRST NBUF2.	*
F219	13	*	HIGH TO LOW.	*
F219	14	*	THEN NBUF1.	*
F219	15	*	LOW TO HIGH.	*
F219	16	*		*
F219	17	*	---- ON ENTRY ----	*
F219	18	*		*
F219	19	*	X-REG SLOTNUM	*
F219	20	*	TIMES \$10.	*
F219	21	*		*
F219	22	*		*
F219	23	*	---- ON EXIT ----	*
F219	24	*		*
F219	25	*	CARRY SET IF ERROR.	*

F219	26 *	(W PROT VIOLATION) *	
F219	27 *		
F219	28 *	IF NO ERROR.	
F219	29 *		
F219	30 *	A-REG UNCERTAIN	
F219	31 *	X-REG UNCHANGED	
F219	32 *	Y-REG HOLDS \$00.	
F219	33 *	CARRY CLEAR	
F219	34 *		
F219	35 *	----- ASSUMES -----	
F219	36 *		
F219	37 *	1 USEC CYCLE TIME	
F219	38 *		
F219	39	*****	
F219 38	40	WRITE16 SEC	ANTICIPATE WPROT ERR.
F21A 88	41	CLV	TO INDICATE WRITE PROTECT ERROR INSTEAD OF
F21B 8D 8D C0	42	LDA Q6H, X	INTERUPT
F21E 8D 8E C0	43	LDA Q7L, X	SENSE WPROT FLAG
F221 30 F5	44	BMI WEXIT	BRANCH IF NOT WRITE PROTECTED
F223 A9 FF	45	WRT1 LDA #\$FF	SYNC DATA.
F225 9D 8F C0	46	STA Q7H, X	(5) GOTO WRITE MODE
F228 1D 8C C0	47	ORA Q6L, X	(4)
F22B A0 04	48	LDY #\$4	(2) FOR FIVE NIBLS
F22D EA	49	NOP	(2)
F22E 48	50	PHA	(4)
F22F 68	51	PLA	(3)
F230 48	52	WSYNC PHA	(4) EXACT TIMING
F231 68	53	PLA	(3) EXACT TIMING
F23C 20 8D F2	54	JSR WNIBL7	(13, 9, 6) WRITE SYNC
F235 88	55	DEY	(2)
F236 D0 F5	56	BNE WSYNC	(2*) MUST NOT CROSS PAGE!
F238 A9 D5	57	LDA #\$D5	(2) 1ST DATA MARK
F23A 20 8C F2	58	JSR WNIBL9	(15, 9, 6)
F23D A9 AA	59	LDA #\$AA	(2) 2ND DATA MARK
F23F 20 DC F2	60	JBR WNIBL9	(15, 9, 6)
F242 A9 AD	61	LDA #\$AD	(2) 3RD DATA MARK.
F244 20 8C F2	62	JSR WNIBL9	(15, 9, 6)
F247 A0 55	63	LDY #\$55	(2) NBUF2 INDEX
F249 EA	64	NOP	(2) FOR TIMING
F24A EA	65	NOP	(2)
F24B EA	66	NOP	(2)
F24C D0 08	67	BNE VRYFRST	(3) BRANCH ALWAYS
F24E AD EF FF	68	WINTRPT LDA INTERRUPT	(4) POLL INTERRUPT LINE
F251 05 8B	69	ORA IMASK	(3)
F253 EA	70	NOP	(2)
F254 10 5D	71	BPL SERVICE	(2) BRANCH IF INTERRUPT HAS OCCURED
F256 30 00	72	VRYFRST BMI WRTFRST	(3) FOR TIMING
F258 B9 02 03	73	WRTFRST LDA NBUF2, Y	(4)
F25B 9D 8D C0	74	STA Q6H, X	(5) STORE ENCODED BYTE
F25E 8D 8C C0	75	LDA Q6L, X	(4) TIME MUST = 32 US PER BYTE!
F261 88	76	DEY	(2)
F262 10 EA	77	BPL WINTRPT	(3) (2 IF BRANCH NOT TAKEN)
F264 98	78	TYA	(2) INSURE NO INTERRUPT THIS BYTE.
F265 30 03	79	BMI WMIDLE	(3) BRANCH ALWAYS.
F267 AD EF FF	80	WNTRPT1 LDA INTERRUPT	(4) POLL INTERRUPT LINE
F26A 05 8B	81	WMIDLE ORA IMASK	(3)
F26C EA	82	NOP	(2)
F26D 30 02	83	BMI WDATA2	(3) BRANCH IF NO INTERRUPT
F26F 10 42	84	BPL SERVICE	GO SERVICE INTERRUPT.
F271 C8	85	WDATA2 INY	(2)
F272 B9 00 02	86	LDA NBUF1, Y	(4)
F275 9D 8D C0	87	STA Q6H, X	(5) STORE ENCODED BYTE
F278 8D 8C C0	88	LDA Q6L, X	(4)
F27B C0 E4	89	CPY #\$E4	(2) WITHIN 1 MS OF COMPLETION?
F27D D0 E8	90	BNE WNTRPT1	(3) (2) NO KEEP WRITTING AND POLLING.
F27F EA	91	NOP	(2)
F280 C8	92	INY	(2)
F281 EA	93	WDATA3 NOP	(2)
F282 EA	94	NOP	(2)
F283 48	95	PHA	(4)
F284 68	96	PLA	(3)
F285 B9 00 02	97	LDA NBUF1, Y	(4) WRITE LAST OF ENCODED BYTES

4,383,296

39

40

F288: 9D BD C0	98	STA Q6H, X	(5) WITHOUT POLLING INTERRUPTS.
F28B: BD BC C0	99	LDA Q6L, X	(4)
F28E: A5 96	100	LDA CKSUM	(3) NORMALLY FOR TIMING
F290: C8	101	INY	(2)
F291: D0 EE	102	BNE WDATA3	(3) (2)
F293: F0 00	103	BEQ WRCKSUM	(3) BRANCH ALWAYS
F295: 20 BD F2	104	WRCKSUM JSR WNIBL7	(13, 9, 6) GO WRITE CHECK SUM
F298: A9 DE	105	LDA #SDE	(2) DM4, BIT SLIP MARK
F29A: 20 BC F2	106	JSR WNIBL9	(15, 9, 6) WRITE IT
F29D: A9 AA	107	LDA #SAA	(2) DM5, BIT SLIP MARK
F29F: 20 BC F2	108	JSR WNIBL9	(15, 9, 6) WRITE IT
F2A2: A9 EB	109	LDA #SEB	(2) DM6, BIT SLIP MARK
F2A4: 20 BC F2	110	JSR WNIBL9	(15, 9, 6) WRITE IT
F2A7: A9 FF	111	LDA #SFF	(2) TURN-OFF BYTE
F2A9: 20 BC F2	112	JSR WNIBL9	(15, 9, 9) WRITE IT
F2AC: BD BE C0	113	NOWRITE LDA Q7L, X	OUT OF WRITE MODE
F2AF: BD BC C0	114	LDA Q6L, X	TO READ MODE.
F2B2: 60	115	RTS	RETURN FROM WRITE.
F2B3:	116	*	
F2B3 38	117	SERVICE SEC	TREAT INTERRUPTION AS ERROR
F2B4: 20 54 F3	118	BIT SEV	SET VFLAG TO INDICATE INTERRUPT
F2B7: 20 AC F2	119	JSR NOWRITE	TAKE IT OUT OF WRITE MODE
F2BA 58	120	CLI	COULD NOT HAVE GOT HERE WITHOUT CLI OK
F2BB 60	121	RTS	
F2BC	122	*****	
F2BC	123	*	
F2BC	124	* 7-BIT NIBL WRITE SUBRS	*
F2BC	125	*	
F2BC	126	* A-REG ORD PRIOR EXIT	*
F2BC	127	* CARRY CLEARED	*
F2BC	128	*	
F2BC	129	*****	
F2BC 18	130	WNIBL9 CLC	(2) 9 CYCLES, THEN WRITE
F2BD 48	131	WNIBL7 PHA	(3) 7 CYCLES, THEN WRITE
F2BF 60	132	PLA	(4)
F2C0: 9D BD C0	133	WNIBL LTA Q6H, X	(5) NIBL WRITE SUB
F2C2: 10 BC C0	134	LRA Q6L, X	(4) CLOBBERS ACC NOT CARR
F2C5 60	135	RTS	
F2C6	136	*	
F2C6	138	*****	
F2C6	139	*	
F2C6	140	* PRENIBLIZE SUBR	*
F2C6	141	* (16-SECTOR FORMAT)	*
F2C6	142	*	
F2C6	143	*****	
F2C6	144	*	
F2C6	145	* CONVERTS 256 BYTES OF	*
F2C6	146	* USER DATA IN (BUF) INTO	*
F2C6	147	* ENCODED BYTES TO BE	*
F2C6	148	* WRITTEN DIRECTLY TO DISK	*
F2C6	149	* ENCODED CHECK SUM IN	*
F2C6	150	* ZERO PAGE 'CKSUM'	*
F2C6	151	*	
F2C6	152	* ---- ON ENTRY ----	*
F2C6	153	*	
F2C6	154	* BUF IS 2-BYTE POINTER	*
F2C6	155	* TO 256 BYTES OF USER	*
F2C6	156	* DATA.	*
F2C6	157	*	
F2C6	158	* ---- ON EXIT ----	*
F2C6	159	*	
F2C6	160	* A-REG CHECK SUM	*
F2C6	161	* X-REG UNCERTAIN	*
F2C6	162	* Y-REG HOLDS 0.	*
F2C6	163	* CARRY SET.	*
F2C6	164	*	
F2C6	165	*****	
F2C6 A2 02	166	PRENIB16 LDX #S2	START NBUF2 INIEX
F2C8: A0 00	167	LDY #0	START USER BUF INDEX.
F2CA: 88	168	PRENIB1 DEY	NEXT USER BYTE
F2CB: B1 9B	169	LDA (BUF), Y	
F2CD 4A	170	LSR A	SHIFT TWO BITS OF
F2CE 3E 01 03	171	ROL NBUF2-1, X	CURRENT USER BYTE

4,383,296

		41			42
F2D1	4A	172	LSR	A	INTO CURRENT NBUF2
F2D2	3E 01 03	173	ROL	NBUF2-1,X	BYTE
F2D3	99 01 02	174	STA	NBUF1+1,Y	(6 BITS LEFT)
F2D8	EB	175	INX		FROM 0 TO \$55
F2D9	E0 56	176	CPX	#\$56	
F2DB	90 ED	177	RCC	PRENIB1	BR IF NO WRAPAROUND
F2DD	A2 00	178	LDX	*0	RESET NBUF2 INDEX
F2DF	98	179	TYA		USER BUF INDEX
F2E0	D0 E8	180	DNE	PRENIB1	(DONE IF ZERO)
F2E2	A0 56	181	LDY	#\$56	(ACC=0 FOR CHECK SUM)
F2E4	59 00 03	182	PRENIB3	EOR NBUF2-2,Y	COMBINE WITH PREVIOUS
F2E7	29 3F	183	PRENIB2	AND #\$3F	STRIP GARBAGE BITS
F2E9	AA	184	TAX		TO FORM RUNNING CHECK SUM
F2EA	BD 55 F3	185	LDA	NIBL,X	GET ENCODED EQUIV
F2ED	99 01 03	186	STA	NBUF2-1,Y	REPLACE PREVIOUS
F2F0	B9 00 03	187	LDA	NBUF2-2,Y	RESTORE ACTUAL PREVIOUS
F2F3	88	188	DEY		
F2F4	D0 EE	189	DNE	PRENIB3	LOOP UNTIL ALL OF NBUF2 IS CONVERTED
F2F6	29 3F	190	AND	#\$3F	
F2F8	59 01 02	191	PRENIB4	EOR NBUF1+1,Y	NOW DO THE SAME FOR
F2FB	AA	192	TAX		NIBBLE BUFFER 1
F2FC	BD 55 F3	193	LDA	NIBL,X	TO DO ANY BACK TRACKING (NBUF1-1)
F2FF	99 00 02	194	STA	NBUF1,Y	
F302	B9 01 02	195	LDA	NBUF1+1,Y	RECOVER THAT WHICH IS NOW PREVIOUS
F305	C8	196	INY		
F306	D0 F0	197	BNE	PRENIB4	
F308	AA	198	TAX		USE LAST AS CHECK SUM
F309	BD 55 F3	199	LDA	NIBL,X	
F30C	85 96	200	STA	CKSUM	
F30E	4C 4C F3	201	JMP	SETIMEG	ALL DONE.
F311		203	*****		
F311		204	*	*	
F311		205	*	POSTNIBLIZE SUBR	*
F311		206	*	16-SECTOR FORMAT	*
F311		207	*	*	*
F311		208	*****		
F311		209	*	*	
F311	A0 55	210	POSTNIB16	LDY #\$55	FIRST CONVERT TO 6 BIT NIBBLES
F313	A9 00	211	LDA	*0	INIT CHECK SUM
F315	BE 02 03	212	PNIBL1	LDX NBUF2,Y	GET ENCODED BYTE
F318	5D 00 F3	213	EOR	DNIBL,X	
F31B	99 02 03	214	STA	NBUF2,Y	REPLACE WITH 6 BIT EQUIV
F31E	88	215	DEY		
F31F	10 F4	216	BPL	PNIBL1	LOOP UNTIL DONE WITH NIBBLE BUFFER 2
F321	C8	217	INY		NOW Y=0
F322	BE 00 02	218	PNIBL2	LDX NBUF1,Y	DO THE SAME WITH
F325	5D 00 F3	219	EOR	DNIBL,X	
F328	99 00 02	220	STA	NBUF1,Y	NIBBLE BUFFER 1
F32B	C8	221	INY		DO ALL 256 BYTES
F32C	D0 F4	222	BNE	PNIBL2	
F32E	A6 96	223	LDX	CKSUM	MAKE SURE CHECK SUM MATCHES
F330	5D 00 F3	224	EOR	DNIBL,X	BETTER BE ZERO
F333	38	225	SEC		ANTICIPATE ERROR
F334	D0 16	226	BNE	POSTERR	BRANCH IF IT IS
F336	A2 56	227	LDX	#\$56	INIT NBUF2 INDEX
F338	CA	228	DEX		NBUF IDX \$55 TO \$0.
F339	30 FB	229	BMI	POST1	WRAPAROUND IF NEG
F33B	B9 00 02	230	LDA	NBUF1,Y	
F33E	5E 02 03	231	LSR	NBUF2,X	SHIFT 2 BITS FROM
F341	2A	232	ROL	A	CURRENT NBUF2 NIBL
F342	5E 02 03	233	LSR	NBUF2,X	INTO CURRENT NBUF1
F345	2A	234	ROL	A	NIBL
F346	91 9B	235	STA	(BUF),Y	BYTE OF USER DATA
F348	C8	236	INY		NEXT USER BYTE
F349	D0 ED	237	DNE	POST2	
F34B	18	238	CLC		GOOD DATA
F34C		239	POSTERR	EQU *	
F34C	AD DF FF	240	SETIMEG	LDA ENVIRON	
F34F	09 80	241	ORA	#ONEMEG	SET TO ONE MEGAHERTZ CLOCK RATE
F351	8D DF FF	242	STA	ENVIRON	
F354	60	243	SEV	RTS	(SEV USED TO SET VFLAG)

4,383,296

43

44

F355:	245	*****		
F355:	246	*		*
F355:	247	*	6-BIT TO 7-BIT	*
F355:	248	*	NIBL CONVERSION TABLE	*
F355:	249	*		*
F355:	250	*****		
F355:	251	*		*
F355:	252	*	CODES WITH MORE THAN	*
F355:	253	*	ONE PAIR OF ADJACENT	*
F355:	254	*	ZEROES OR WITH NO	*
F355:	255	*	ADJACENT ONES (EXCEPT	*
F355:	256	*	37) ARE EXCLUDED.	*
F355:	257	*		*
F355:	258	*****		
F355: 96 97 9A	259	NIBL	DFB \$96, \$97, \$9A	
F358: 9B 9D 9E	260		DFB \$9B, \$9D, \$9E	
F35B: 9F A6 A7	261		DFB \$9F, \$A6, \$A7	
F35E: AB AC AD	262		DFB \$AB, \$AC, \$AD	
F361: AE AF B2	263		DFB \$AE, \$AF, \$B2	
F364: B3 B4 B5	264		DFB \$B3, \$B4, \$B5	
F367: B6 B7 B9	265		DFB \$B6, \$B7, \$B9	
F36A: BA BB BC	266		DFB \$BA, \$BB, \$BC	
F36D: BD BE BF	267		DFB \$BD, \$BE, \$BF	
F370: CB CD CE	268		DFB \$CB, \$CD, \$CE	
F373: CF D3 D6	269		DFB \$CF, \$D3, \$D6	
F376: D7 D9 DA	270		DFB \$D7, \$D9, \$DA	
F379: DB DC DD	271		DFB \$DB, \$DC, \$DD	
F37C: DE DF E5	272		DFB \$DE, \$DF, \$E5	
F37F: E6 E7 E9	273		DFB \$E6, \$E7, \$E9	
F382: EA EB EC	274		DFB \$EA, \$EB, \$EC	
F385: ED EE EF	275		DFB \$ED, \$EE, \$EF	
F388: F2 F3 F4	276		DFB \$F2, \$F3, \$F4	
F38B: F5 F6 F7	277		DFB \$F5, \$F6, \$F7	
F38E: F9 FA FB	278		DFB \$F9, \$FA, \$FB	
F391: FC FD FE	279		DFB \$FC, \$FD, \$FE	
F394: FF	280		DFB \$FF	
F395:	282	*****		
F395:	283	*		*
F395:	284	*	7-BIT TO 6-BIT	*
F395:	285	*	DENIBLIZE TABL	*
F395:	286	*	(16-SECTOR FORMAT)	*
F395:	287	*		*
F395:	288	*	VALID CODES	*
F395:	289	*	\$96 TO \$FF ONLY.	*
F395:	290	*		*
F395:	291	*		*
F395:	292	*	CODES WITH MORE THAN	*
F395:	293	*	ONE PAIR OF ADJACENT	*
F395:	294	*	ZEROES OR WITH NO	*
F395:	295	*	ADJACENT ONES (EXCEPT	*
F395:	296	*	BIT 7) ARE EXCLUDED	*
F395:	297	*****		
F395: 00	298		BRK ONE BYTE LEFT OVER	
F300:	299	DNIBL	EQU REGRWTS+\$300	
F396: 00 01 98	300		DFB \$00, \$01, \$98	
F399: 99 02 03	301		DFB \$99, \$02, \$03	

4,383,296

45

46

```

F39C 9C 04 05 302
F39F 06 A0 A1 303
F3A2 A2 A3 A4 304
F3A5 A5 07 08 305
F3A8 AB A9 AA 306
F3AB 09 0A 0B 307
F3AE 0C 0D 0E 308
F3B1 B1 0E 0F 309
F3B4 10 11 12 310
F3B7 13 B8 14 311
F3BA 15 16 17 312
F3BD 18 19 1A 313
F3C0 C0 C1 C2 314
F3C3 C3 C4 C5 315
F3C6 C6 C7 C8 316
F3C9 C9 CA 1B 317
F3CC CC 1C 1D 318
F3CF 1E D0 D1 319
F3D2 D2 1F D4 320
F3D5 D5 20 21 321
F3D8 D8 22 23 322
F3DB 24 25 26 323
F3DE 27 28 E0 324
F3E1 E1 E2 E3 325
F3E4 E4 29 2A 326
F3E7 2B EB 2C 327
F3EA 2D 2E 2F 328
F3ED 30 31 32 329
F3F0 F0 F1 33 330
F3F3 34 35 36 331
F3F6 37 38 FB 332
F3F9 39 3A 3B 333
F3FC 3C 3D 3E 334
F3FF 3F 335
F400: 337 *****
F400: 338 *
F400: 339 * FAST SEEK SUBROUTINE *
F400: 340 *
F400: 341 *****
F400: 342 *
F400: 343 * ---- ON ENTRY ---- *
F400: 344 *
F400: 345 * X-REG HOLDS SLOTNUM *
F400: 346 * TIMES $10. *
F400: 347 *
F400: 348 * A-REG HOLDS DESIRED *
F400: 349 * HALFTRACK. *
F400: 350 * (SINGLE PHASE) *
F400: 351 *
F400: 352 * CURTRK HOLDS CURRENT *
F400: 353 * HALFTRACK. *
F400: 354 *
F400: 355 * ---- ON EXIT ---- *
F400: 356 *
F400: 357 * A-REG UNCERTAIN. *
F400: 358 * Y-REG UNCERTAIN. *
F400: 359 * X-REG UNDISTURBED. *
F400: 360 *
F400: 361 * CURTRK AND TRKN HOLD *
F400: 362 * FINAL HALFTRACK. *
    
```

```

DFB $9C, $04, $05
DFB $06, $A0, $A1
DFB $A2, $A3, $A4
DFB $A5, $07, $08
DFB $AB, $A9, $AA
DFB $09, $0A, $0B
DFB $0C, $0D, $0E
DFB $B1, $0E, $0F
DFB $10, $11, $12
DFB $13, $B8, $14
DFB $15, $16, $17
DFB $18, $19, $1A
DFB $C0, $C1, $C2
DFB $C3, $C4, $C5
DFB $C6, $C7, $C8
DFB $C9, $CA, $1B
DFB $CC, $1C, $1D
DFB $1E, $D0, $D1
DFB $D2, $1F, $D4
DFB $D5, $20, $21
DFB $D8, $22, $23
DFB $24, $25, $26
DFB $27, $28, $E0
DFB $E1, $E2, $E3
DFB $E4, $29, $2A
DFB $2B, $EB, $2C
DFB $2D, $2E, $2F
DFB $30, $31, $32
DFB $F0, $F1, $33
DFB $34, $35, $36
DFB $37, $38, $FB
DFB $39, $3A, $3B
DFB $3C, $3D, $3E
DFB $3F
    
```

4,383,296

47

48

```

F400: 363 *
F400: 364 * PRIOR HOLDS PRIOR *
F400: 365 * HALFTRACK IF SEEK *
F400: 366 * WAS REQUIRED. *
F400: 367 *
F400: 368 * MONTIMEL AND MONTIMEH *
F400: 369 * ARE INCREMENTED BY *
F400: 370 * THE NUMBER OF *
F400: 371 * 100 USEC QUANTUMS *
F400: 372 * REQUIRED BY SEEK *
F400: 373 * FOR MOTOR ON TIME *
F400: 374 * OVERLAP. *
F400: 375 *
F400: 376 * --- VARIABLES USED --- *
F400: 377 *
F400: 378 * CURTRK, TRKN, COUNT, *
F400: 379 * PRIOR, SLOTTEMP *
F400: 380 * MONTIMEL, MONTIMEH *
F400: 381 *
F400: 382 *****
F400: B5 9E 383 SEEK STA TRKN ;SAVE TARGET TRACK
F402: C5 8C 384 CMP CURTRK ;ON DESIRED TRACK?
F404: F0 42 385 BEQ SETPHASE ;YES, ENERGIZE PHASE AND RETURN
F406: A9 00 386 LDA #$0
F408: B5 95 387 STA TRKCNT ;HALFTRACK COUNT.
F40A: A5 8C 388 SEEK2 LDA CURTRK ;SAVE CURTRK FOR
F40C: B5 9D 389 STA PRIOR ;DELAYED TURNOFF.
F40E: 38 SEC
F40F: E5 9E 391 SBC TRKN ;DELTA-TRACKS.
F411: F0 31 392 BEQ SEEKEND ;BR IF CURTRK=DESTINATION
F413: B0 06 393 BCS OUT (MOVE OUT, NOT IN)
F415: A9 FF 394 EOR #$FF CALC TRKS TO GO
F417: E6 8C 395 INC CURTRK INCR CURRENT TRACK (IN)
F419: 90 04 396 BCC MINTST (ALWAYS TAKEN).
F41B: 69 FE 397 OUT ADC #$FF ;CALC TRKS TO GO
F41D: C6 8C 398 DEC CURTRK ;DECR CURRENT TRACK (OUT)
F41F: C5 95 399 MINTST CMP TRKCNT
F421: 90 02 400 BCC MAXTST AND 'TRKS MOVED'
F423: A5 95 401 LDA TRKCNT
F425: C9 09 402 MAXTST CMP #$9
F427: B0 02 403 BCS STEP2 ;IF TRKCNT:$8 LEAVE Y ALONE (Y=$8)
F429: A8 404 STOP TAY ;ELSE SET ACCELERATION INDEX IN Y
F42A: 38 SEC
F42B: 20 48 F4 406 STEP2 JSR SETPHASE
F42E: B9 67 F4 407 LDA ONTABLE, Y ;FOR 'ONTIME'
F431: 20 56 F4 408 JSR MSWAIT ;(100 USEC INTERVALS)
F434: A5 9D 409 LDA PRIOR
F436: 18 410 CLC ;FOR PHASEOFF
F437: 20 4A F4 411 JSR CLRPHASE ;TURN OFF PRIOR PHASE
F43A: B9 70 F4 412 LDA OFFTABLE, Y ;THEN WAIT 'OFFTIME'
F43D: 20 56 F4 413 JSR MSWAIT (100 USEC INTERVALS)
F440: E6 95 414 INC TRKCNT ('TRACKS MOVED' COUNT.
F442: D0 C6 415 BNE SEEK2 (ALWAYS TAKEN)
F444: 20 56 F4 416 SEEKEND JSR MSWAIT ;SETTLE 25 MSEC
F447: 18 417 CLC ;SET FOR PHASE OFF
F448: A5 8C 418 SETPHASE LDA CURTRK ;GET CURRENT TRACK
F44A: 29 03 419 CLRPHASE AND #3 ;MASK FOR 1 OF 4 PHASES
F44C: 2A 420 ROL A ;DOUBLE FOR PHASEON/OFF INDEX
F44E: C1 91 421 ORA IBLSLOT
F44F: AA 422 TAX
F450: BD 80 C0 423 LDA PHASEOFF, X ;TURN ON/OFF ONE PHASE
F453: A6 81 424 LDX IBLSLOT ;RESTORE X-REG
F455: 60 425 SEEKRTS RTS ;AND RETURN
F456: 427 *****
F456: 428 *
F456: 429 * MSWAIT SUBROUTINE *
F456: 430 *

```

4,383,296

49

50

F456	431	*****		
F456	432	*		*
F456	433	*	DELAYS A SPECIFIED	*
F456	434	*	NUMBER OF 100 USEC	*
F456	435	*	INTERVALS FOR NOTICE	*
F456	436	*	ON-TIME	*
F456	437	*		*
F456	438	*	----- ON ENTRY -----	*
F456	439	*		*
F456	440	*	A-REG HOLDS NUMBER	*
F456	441	*	OF 100 USEC	*
F456	442	*	INITIALS TO	*
F456	443	*	DELAY	*
F456	444	*		*
F456	445	*	---- ON EXIT ----	*
F456	446	*		*
F456	447	*	A-REG HOLDS #00	*
F456	448	*	A-REG HOLDS #01	*
F456	449	*	A-REG HOLDS #02	*
F456	450	*	CARRY SET	*
F456	451	*		*
F456	452	*	MONITIME1 MONITIME1	*
F456	453	*	MONITIME2 MONITIME2	*
F456	454	*	MONITIME3 MONITIME3	*
F456	455	*	MONITIME4 MONITIME4	*
F456	456	*	MONITIME5 MONITIME5	*
F456	457	*	MONITIME6 MONITIME6	*
F456	458	*	MONITIME7 MONITIME7	*
F456	459	*	MONITIME8 MONITIME8	*
F456	460	*	MONITIME9 MONITIME9	*
F456	461	*	MONITIME10 MONITIME10	*
F456	462	*	MONITIME11 MONITIME11	*
F456	463	*	MONITIME12 MONITIME12	*
F456	464	*	MONITIME13 MONITIME13	*
F456	465	*	MONITIME14 MONITIME14	*
F456	466	*	MONITIME15 MONITIME15	*
F456	467	*	MONITIME16 MONITIME16	*
F456	468	*	MONITIME17 MONITIME17	*
F456	469	*	MONITIME18 MONITIME18	*
F456	470	*	MONITIME19 MONITIME19	*
F456	471	*	MONITIME20 MONITIME20	*
F456	472	*	MONITIME21 MONITIME21	*
F456	473	*	MONITIME22 MONITIME22	*
F456	474	*****		
F456	475	*		*
F456	476	*	PHASE ON-, OFF-TIME	*
F456	477	*	TABLES IN 100-USEC	*
F456	478	*	INTERVALS (SEEK)	*
F456	479	*		*
F456	480	*****		
F467: 01 30 28	481	ONTABLE	DFB 1, \$30, \$28	
F46A: 24 20 1E	482		DFB \$24, \$20, \$1E	
F46D: 1D 1C 1C	483		DFB \$1D, \$1C, \$1C	
F470: 70 2C 26	484	OFFTABLE	DFB \$70, \$2C, \$26	
F473: 22 1F 1E	485		DFB \$22, \$1F, \$1E	
F476: 1D 1C 1C	486		DFB \$1D, \$1C, \$1C	

4,383,296

51

52

```

F479: 86 83      488 BLOCKIO STX  IBTRK
F47B: A0 05      489          LDY  #5
F47D: 48         490          PHA
F47E: 0A         491 TRKSEC ASL  A
F47F: 26 83      492          ROL  IBTRK
F481: 88         493          DEY
F482: D0 FA      494          BNE  TRKSEC
F484: 68         495          PLA
F485: 29 07      496          AND  #7
F487: A8         497          TAY
F488: B9 A0 F4    498          LDA  SECTABL, Y
F48B: 85 84      499          STA  IBSECT
F48D: 20 00 F0    500          JSR  REGRWTS
F490: B0 0B      501          BCS  QUIT
F492: E6 86      502          INC  IBBUFP+1
F494: E6 84      503          INC  IBSECT
F496: E6 84      504          INC  IBSECT
F498: 20 00 F0    505          JSR  REGRWTS
F49B: C6 86      506          DEC  IBBUFP+1
F49D: A5 88      507 QUIT LDA  IBSTAT
F49F: 60         508          RTS
F4A0:           509 *
F4A0:           510 SECTABL EQU *
F4A0: 00 04 08    511          DFB  $0, $4, $8
F4A3: 00 01 05    512          DFB  $0, $1, $5
F4A6: 09 0D      513          DFB  $9, $D
F4A8:           514 *
F4A8:           516 * * * * *
F4A8:           517 *
F4A8:           518 * JOYSTICK READ ROUTINE
F4A8:           519 *
F4A8:           520 * * * * *
F4A8:           521 * ENTRY ACC= COUNT DOWN HIGH
F4A8:           522 * X&Y= DON'T CARE
F4A8:           523 *
F4A8:           524 * EXIT ACC= TIMER HIGH BYTE
F4A8:           525 * Y= TIMER LOW BYTE
F4A8:           526 * CARRY CLEAR
F4A8:           527 *
F4A8:           528 * IF CARRY SET, ROUTINE
F4A8:           529 * WAS INTERRUPTED &
F4A8:           530 * ACC & Y ARE INVALID
F4A8:           531 * * * * *
F4A8:           532 *
FFD9:           533 TIMLATCH EQU $FFD9
FFD8:           534 TIMER1L EQU $FFD8
FFD9:           535 TIMER1H EQU $FFD9
C066:           536 JOYRDY EQU $C066
F4A8:           537 *
F4A8:           538 ANALOG EQU * ; CARRY SHOULD BE SET!
F4A8: 8D D9 FF      539 STA TIMLATCH ; START THE TIMER!
F4A8: AD EF FF      540 ANLOG1 LDA INTERRUPT
F4A8: 2D 86 C2      541 AND JOYRDY ; WAIT FOR ONE OR THE OTHER TO GO LOW
F4B1: 20 F8      542 BMI ANLOG1
F4B3: AD 86 C0      543 LDA JOYRDY ; WAY IT REALLY THE JOYSTICK?
F4B4: 30 C2      544 BMI GOODTIME ; NOPE, FORGET IT
F4B8: 18         545 CLC ; TIME'S A SLIP SLIDIN AWAY
F4B9: AD D9 FF      546 LDA TIMER1H ; NOW, WHAT TIME IS IT?
F4BC: AD D8 FF      547 LDY TIMER1L
F4B7: 10 C3      548 BPL GOODTIME ; TIME WAS VALID!
F4C1: AD D9 FF      549 LDA TIMER1H ; HI BYTE CHANGED
F4C4: 60         550 GOODTIME RTS
*** SUCCESSFUL ASSEMBLY. NO ERRORS
    
```

4,383,296

53

54

FOE9 ALDONE1	FOE0 ALLDONE	F119 ALLOFF	?F4A8 ANALDG
F4AB ANLDG1	?F479 BLOCKID	9B BUF	F12D CHKDRV1
F12B CHKDRV	F13D CKDRTS	96 CKSUM	F44A CLRPHASE
FO50 CONWAIT	FOC7 CORRECTSECT	?FOBF CORRECTVCL	95 COUNT
97 CSSTV	97 CSUM1	89 CSUM	8C CURTRK
F300 DNIBL	FO31 DRIVSEL	CO8A DRVOEN	85 DRVOTRK
?CO8B DRV1EN	FOE5 DRVERR	F13E DRVINDX	FO3D DRVWAIT
E0 DVMOT	FFDF ENVIRON	9F ENVTEMP	?FOAO GDCAL1
?FOA1 GDCAL	F4C4 GOODTIME	?F116 G0SEEK	F1BA GOSERV
FOE8 HNDLERR	90 HRDERRS	85 IBBUFP	87 IBCMD
82 IBDERR	82 IBDRVN	80 IBNODRV	? 83 IBRERR
84 IBSECT	81 IBSL0T	89 IBSMOD	88 IBSTAT
83 IBTRK	81 IBWPER	8B IMASK	FFEF INTERRUPT
8A IOBPDN	CO66 JOYRDY	95 LAST	F425 MAXTST
F41F MINTST	9A MONTIMEH	99 MONTIMEL	FO4E MOTOF
CO88 MOTOROFF	CO89 NOTORON	F458 MSW1	F461 MSW2
F456 MSWAIT	F105 MYSEEK	O200 NBUF1	O302 NBUF2
F355 NIBL	?FO60 NODRIVERR	FO8D NOINTR1	FOF3 NOINTR2
F2AC NOWRITE	F11B NXOFF	F470 OFFTABLE	FO44 OK
80 ONEMEG	F467 ONTABLE	F41B OUT	CO80 PHASEOFF
?CO81 PHASEON	?CO81 PHASON	?CO80 PHSOFF	F315 PNIBL1
F322 PNIBL2	F336 POST1	F338 POST2	F34C POSTERR
F311 POSTNIB16	F2CA PRENIB1	F2C6 PRENIB16	?F2E7 PRENIB2
F2E4 PRENIB3	F2F8 PRENIB4	9D PRIOR	CO8D Q6H
CO8C Q6L	CO8F Q7H	CO8E Q7L	F49D QUIT
F14D RD1	F157 RD2	F162 RD3	F16B RD4
F17E RDSA	F17D RD5	F192 RD6	F1A5 RD7
F1AF RDB	F1C8 RDA1	F1D2 RDA2	F1DD RDA3
F1EA RDA4	F1F2 RDA5	F204 RDA6	F20E RDA7
F1BD RDADR1&	F1E8 RDAFLD	F1CD RDASN1	F1C1 RDASYN
F19D RDCKSUM	F188 RDERR	F217 RDEXIT	FOA7 RDRIGHT
F148 READ16	FO00 REGRWTS	93 RETRYCNT	F152 RSYNC1
F14A RSYNC	FO8B RTTRK	F4A0 SECTABL	98 SECT
?F106 SEEK1	F40A SEEK2	94 SEEKCNT	F400 SEEK
F444 SEEKEND	?F455 SEEKRTS	F2B3 SERVICE	F34C SETIMEG
F448 SETPHASE	F125 SETTRK	F354 SEV	F42B STEP2
?F429 STEP	97 TEMP	FFD9 TIMER1H	FFD8 TIMER1L
FFD9 TIMLATCH	99 TRACK	95 TRKCNT	9E TRKN
99 TRKN1	F47E TRKSEC	FO86 TRYADR2	FO7F TRYADR
FO7B TRYTRK2	FO65 TRYTRK	7F TWOMEG	9A VOLUME
F256 VRYFRST	F271 WDATA2	F281 WDATA3	F218 WEXIT
F24E WINTRPT	F26A WMIDLE	?F2BF WNIBL	F2BD WNIBL7
F2BC WNIBL9	F267 WNTRPT1	F295 WRCKSUM	F219 WRITE16
FOF9 WRIT	?F223 WRT1	F258 WTRFRST	F230 WSYNC
7F TWOMEG	80 IBNODRV	80 HRDERRS	80 ONEMEG
81 IBSL0T	81 IBWPER	82 IBDERR	82 IBDRVN
83 IBRERR	83 IBTRK	84 IBSECT	85 DRVOTRK
85 IBBUFP	87 IBCMD	88 IBSTAT	89 CSUM
89 IBSMOD	8A IOBPDN	8B IMASK	8C CURTRK
93 RETRYCNT	94 SEEKCNT	95 LAST	95 TRKCNT
95 COUNT	96 CKSUM	97 CSSTV	97 CSUM1
97 TEMP	98 SECT	99 MONTIMEL	99 TRKN1
99 TRACK	9A MONTIMEH	9A VOLUME	9B BUF
9D PRIOR	9E TRKN	9F ENVTEMP	E0 DVMOT
O200 NBUF1	O302 NBUF2	CO66 JOYRDY	?CO80 PHSOFF
CO80 PHASEOFF	?CO81 PHASON	?CO81 PHASEON	CO88 MOTOROFF
CO89 MOTORON	CO8A DRVOEN	?CO8B DRV1EN	CO8C Q6L
CO8D Q6H	CO8E Q7L	CO8F Q7H	FO00 REGRWTS
FO31 DRIVSEL	FO3D DRVWAIT	FO44 OK	FO4E MOTOF
FO50 CONWAIT	?FO60 NODRIVERR	FO65 TRYTRK	FO7B TRYTRK2
FO7F TRYADR	FO86 TRYADR2	FO8B NOINTR1	?FOAO GDCAL1
?FOA1 GDCAL	FOA7 RDRIGHT	FO8B RTTRK	?FOBF CORRECTVCL
FOC7 CORRECTSECT	FOE0 ALLDONE	FOE5 DRVERR	FOE8 HNDLERR
FOE9 ALDONE1	FOF3 NOINTR2	FOF9 WRIT	F105 MYSEEK
?F106 SEEK1	?F116 G0SEEK	F119 ALLOFF	F11B NXOFF
F125 SETTRK	F12B CHKDRV	F12D CHKDRV1	F13D CKDRTS

4,383,296

55

56

F13E DRVINDX
 F152 RSYNC1
 F17D RD5
 F1A5 RD7
 F1BD RDADR16
 F1D2 RDA2
 F1F2 RDA5
 F218 WEXIT
 F24E WINTRPT
 F26A WMIDDLE
 F2AC NQWRITE
 ?F2BF WNIBL
 ?F2E7 PRENIB2
 F315 PNIBL1
 F34C POSTERR
 F400 SEEK
 F425 MAXTST
 F448 SETPHASE
 F458 MSW1
 ?F479 BLOCKIO
 ?F4AB ANLOG1
 FFD9 TIMNLATCH

F148 READ16
 F157 RD2
 F17E RD5A
 F1AF RDB
 F1C1 RDASYN
 F1DD RDA3
 F204 RDA6
 F219 WRITE16
 F256 VRYFRST
 F271 WDATA2
 F2B3 SERVICE
 F2C6 PRENID16
 F2F8 PRENID4
 F322 PNIBL2
 F34C SETIMEG
 F40A SEEK2
 ?F429 STEP
 F44A CLRPHASE
 F461 MSW2
 F47E TRKSEC
 F4AB ANLOG1
 FFD9 TIMER1H

F14A RSYNC
 F162 RD3
 F192 RD6
 F1DB RDERR
 F1C8 RDA1
 F1E8 RDAFLD
 F20E RDA7
 ?F223 WRT1
 F258 WRTFRST
 F281 WDATA3
 F2BC WNIBL9
 F2CA PRENID1
 F300 DNIDL
 F336 POST1
 F354 SEV
 F41D OUT
 F42B STEP2
 ?F455 SEEKRTS
 F467 ONTABLE
 F49D GUIT
 F4C4 GOODTIME
 FFD9 ENVIRON

F14D RD1
 F16B RD4
 F19D RDCKSUM
 F1BA GOSERV
 F1CD RDASN1
 F1EA RDA4
 F217 RDEXIT
 F230 WSYNC
 F267 WINTRPT1
 F295 WRCKSUM
 F2BD WNIBL7
 F2E4 PRENIB3
 F311 POSTNIB16
 F338 POST2
 F355 NIDL
 F41F MINTST
 F444 SEEKEND
 F456 MSWAIT
 F470 OFFTABLE
 F4A0 SECTABL
 FFD8 TIMER1L
 FFEF INTERRUPT

```

0000 2 *****
0000 3 *
0000 4 *SARA DIAGNOSTIC TEST ROUTINES
0000 5 *
0000 6 *DECEMBER 19, 1979
0000 7 * BY
0000 8 *N. BROEDNER & R. LASHLEY
0000 9 *
0000 10 *COPYRIGHT 1979 BY APPLE COMPUTER, INC
0000 11 *
0000 12 *****
0001: 13 ROM EQU $1 FOR RAM VERSION: 1 IF TRIPLE ROM
0000: 14 ZRPG EQU $0
0010: 15 ZRPG1 EQU $10
0018: 16 PTRLO EQU ZRPG1+8
0019: 17 PTRHI EQU ZRPG1+9
001A: 18 INK EQU ZRPG1+$A
00B7: 19 IBCMD EQU $B7
00B5: 20 IBRUFF EQU $B5
0091: 21 PREVTRK EQU $91
F479: 22 BLOCKIO EQU $F479
005D: 23 CV EQU $5D
00FF: 24 STR0 EQU $FF
1419: 25 INK EQU $1400+PTRHI
1810: 26 PHP EQU $1800+ZRPG1
C000: 27 KYBD EQU $C000
C008: 28 KEYBD EQU $C008
C010: 29 KBDSTRB EQU $C010
C05B: 30 PLEN EQU $C05B
C047: 31 ADRS EQU $C047
C050: 32 GRMD EQU $C050
C051: 33 TXCMD EQU $C051
C066: 34 ADTO EQU $C066
C0D0: 35 DISKOFF EQU $C0D0
C0F1: 36 AC1AST EQU $C0F1
C0F2: 37 AC1ACM EQU $C0F2
C0F3: 38 AC1ACN EQU $C0F3
C100: 39 SLT1 EQU $C100
C200: 40 SLT2 EQU $C200
C300: 41 SLT3 EQU $C300
C400: 42 SLT4 EQU $C400
CFFF: 43 EXPROM EQU $CFFF
FFD0: 44 ZPREG EQU $FFD0
FFDF: 45 SYSD1 EQU $FFDF
    
```

4,383,296

57

58

```

FFD2: 46 SYSD2 EQU $FFD2
FFD3: 47 SYSD3 EQU $FFD3
FFE0: 48 SYSE0 EQU $FFE0
FFE1: 49 BNKSW EQU $FFE1
FFE2: 50 SYSE2 EQU $FFE2
FFE3: 51 SYSE3 EQU $FFE3
FC25: 52 COUT EQU $FC25
FD07: 53 CROUT1 EQU $FD07
FD0F: 54 KEYIN EQU $FD0F
FBC7: 55 SETCVH EQU $FBC7
FD98: 56 CLDSTRT EQU $FD98
FD9D: 57 SETUP EQU $FD9D
F901: 58 MONITOR EQU $F901
0000: 59 *
    
```

```

----- NEXT OBJECT FILE NAME IS DIAG. OBJ
F4C5: 60 ORG $F4C5
F4C5:00 B1 B2 61 RAMTBL DFB $0, $B1, $B2, $BA, $B9, $10, $0, $13
F4C8: DA B9 10
F4CB: 00 13
F4CD: 62 CHPG EQU *
F4CD:52 41 CD 63 DCI 'RAM'
F4D0:52 4F CD 64 DCI 'RGM'
F4D3:56 49 C1 65 DCI 'VIA'
F4D6:41 43 49 66 DCI 'ACIA'
F4D9:C1
F4DA:41 2F C4 67 DCI 'A/D'
F4DD:44 49 41 68 DCI 'DIAGNOSTIC'
F4E0:47 4E 4F
F4E3:53 54 49
F4E6:C3
F4E7:5A D0 69 DCI 'ZF'
F4E9:52 45 54 70 DCI 'RETRY'
F4EC:52 D9
F4EE: 71 *
F4EE: 72 * SETUP SYSTEM
F4EE: 73 *
F4EE: 74 *
F4EE:A9 53 75 LDA ##52+ROM TURN OFF SCREEN, SET 2MHZ SPEED
F4F0:8D DF FF 76 STA SYSD1 AND RUN OFF ROM
F4F3:A2 00 77 LDX ##00 SET BANK SWITCH TO ZERO
F4F5:8E E0 FF 78 STX SYSE0
F4F8:8E EF FF 79 STX BNKSW
F4FB:8E D0 FF 80 STX ZPREG AND SET ZERO PAGE SAME
F4FE:CA 81 DEX
F4FF:8E D2 FF 82 STX SYSD2 PROGRAM DDR'S
F502:8E D3 FF 83 STX SYSD3
F505:9A 84 TXS
F506:EB 85 INX
F507:A9 0F 86 LDA ##0F
F509:8D E3 FF 87 STA SYSE3
F50C:A9 3F 88 LDA ##3F
F50E:8D E2 FF 89 STA SYSE2
F511:A0 06 90 LDY ##06
F513:D9 D0 C0 91 DISK1 LDA DISKOFF, Y
F516:88 92 DEY
F517:88 93 DEY
F518:10 F9 94 BPL DISK1
F51A:AD 08 C0 95 LDA KEYBD
F51D:29 04 96 AND ##04
F51F:D0 03 97 BNE NXBYT
F521:4C 89 F6 98 JMP RECON
F524: 99 *
F524: 100 * VERIFY ZERO PAGE
F524: 101 *
    
```

4,383,296

59		60	
F524: A9 01	102 NXBYT	LDA ##01	ROTATE A 1 THROUGH
F526: 95 00	103 NXBIT	STA ZRPG,X	EACH BIT IN THE 0 PG
F528: D5 00	104	CMP ZRPG,X	TO COMPLETELY TEST
F52A: D0 FE	105 NOGOOD	BNE NOGOOD	THE PAGE. HANG IF NOGOOD
F52C: 0A	106	ASL A	TRY NEXT BIT OF BYTE
F52D: D0 F7	107	BNE NXBIT	UNTIL BYTE IS ZERO.
F52F: E8	108	INX	CONTINUE UNTIL PAGE
F530: D0 F2	109	BNE NXBYT	IS DONE.
F532:	110 *		
F532: 8A	111 CNTWR	TXA	PUSH A DIFFERENT
F533: 48	112	PHA	BYTE ONTO THE
F534: E8	113	INX	STACK UNTIL ALL
F535: D0 FB	114	BNE CNTWR	STCK BYTES ARE FULL.
F537: CA	115	DEX	THEN PULL THEM
F538: 86 18	116	STX PTRLO	OFF AND COMPARE TO
F53A: 68	117 PULBT	PLA	THE COUNTER GOING
F53B: C5 18	118	CMP PTRLO	BACKWARDS. HANG IF
F53D: D0 E8	119	BNE NOGOOD	THEY DON'T AGREE.
F53F: C6 18	120	DEC PTRLO	GET NEXT COUNTER BYTE
F541: D0 F7	121	BNE PULBT	CONTINUE UNTIL STACK
F543: 68	122	PLA	IS DONE. TEST LAST BYTE
F544: D0 E4	123	BNE NOGOOD	AGAINST ZERO.
F546:	124 *		
F546:	125 * SIZE THE MEMORY		
F546:	126 *		
F546: A2 08	127	LDX ##08	ZERO THE BYTES USED TO DISPLAY
F548: 95 10	128 NOMEM	STA ZRPG1,X	THE DAD RAM LOCATIONS
F54A: CA	129	DEX	EACH BYTE= A CAS LINE
F54D: 10 FB	130	BPL NOMEM	ON THE SARA BOARD.
F54D:	131 *		
F54D: A2 02	132	LDX ##02	STARTING AT PAGE 2
F54F: 86 19	133 NMEM1	STX PTRHI	TEST THE LAST BYTE
F551: A9 00	134	LDA ##00	IN EACH MEM PAGE TO
F553: A0 FF	135	LDY ##FF	SEE IF THE CHIPS ARE
F555: 91 18	136	STA (PTRLO),Y	THERE. (AVOID 0 & STK PAGES)
F557: D1 18	137	CMP (PTRLO),Y	CAN THE BYTE BE 0'D?
F559: F0 07	138	BEG NMEM2	
F55B: 20 48 F7	139	JSR RAM	NO, FIND WHICH CAS IT IS.
F55E: 94 10	140	STY ZRPG1,X	SET CORRES. BYTE TO FF
F560: A6 19	141	LDX PTRHI	RESTORE X REGISTER
F562: E8	142 NMEM2	INX	AND INCREMENT TO NEXT
F563: E0 C0	143	CPX ##C0	PAGE UNTIL I/O IS REACHED.
F565: D0 E8	144	BNE NMEM1	
F567: A2 20	145	LDX ##20	THEN RESET TO PAGE 20
F569: EE EF FF	146	INC BNKSW	AND GOTO NEXT BANK TO
F56C: AD EF FF	147	LDA BNKSW	CONTINUE (MASK INPUTS
F56F: 29 0F	148	AND ##0F	FROM BANKSWITCH TO SEE
F571: C9 03	149	CMP ##03	WHAT SWITCH IS SET TO)
F573: D0 DA	150	BNE NMEM1	CONTINUE UNTIL BANK '3'
F575:	151 *		
F575:	152 * SETUP SCREEN		
F575: 20 9D FD	153 ERRLP	JSR SETUP	CALL SCRNM SETUP ROUTINE
F578: A2 00	154	LDX ##00	SETUP I/O AGAIN
F57A: 8E E0 FF	155	STX SYSEO	FOR VIA TEST
F57D: CA	156	DEX	PROGRAM DATA DIR
F57E: 8E D2 FF	157	STX SYSD2	REGISTERS
F581: 8E D3 FF	158	STX SYSD3	
F584: A9 3F	159	LDA ##3F	
F586: 8D E2 FF	160	STA SYSE2	
F589: A9 0F	161	LDA ##0F	
F58B: 8D E3 FF	162	STA SYSE3	
F58E: A2 10	163	LDX ##10	HEADING OF 'DIAGNOSTICS' WITH
F590: 20 38 F7	164	JSR STRWT	THIS SUBROUTINE
F593: A2 00	165 ERRLP1	LDX ##00	PRINT 'RAM'
F595: 86 5D	166	STX CV	SET CURSOR TO 2ND LINE
F597: A9 04	167	LDA ##04	SPACE CURSOR OUT 3

4,383,296

		61			62
F599:	20 C7 F8	168	JSR	SETCVH	(X STILL=0 ON RETURN)
F59C:	20 38 F7	169	JSR	STRWT	THE SAME SUBROUTINE
F59F:	A2 07	170	LDX	#\$07	FOR BYTES 7 - 0 IN
F5A1:		171	RAMWT1	EQU	*
F5A1:	85 10	172	LDA	ZRPG1,X	OUT EACH BIT AS A
F5A3:	A0 08	173	LDY	#\$08	' ' OR '1' FOR INDICATE BAD OR MISSING
F5A5:	0A	174	RAMWT2	ASL	A
F5A6:	48	175	PHA		CHIPS SUBROUTINE 'RAM' RAM
F5A7:	A9 AE	176	LDA	#\$AE	SETS UP THESE BYTES
F5A9:	90 02	177	BCC	RAMWT4	LOAD A ' ' TO ACC
F5AB:	A9 31	178	LDA	#\$31	AND PRINT IT
F5AD:	20 25 FC	179	RAMWT4	JSR	COUT
F5D0:	68	180	PLA		RESTORE BYTE
F5B1:	88	181	DEY		AND ROTATE ALL B
F5B2:	D0 F1	182	BNE	RAMWT2	TIMES
F5B4:	20 07 FD	183	JSR	CRDUT1	CLEAR TO END OF LINE.
F5B7:	CA	184	DEY		
F5B8:	10 E7	185	BPL	RAMWT1	
F5BA:		186	*		
F5BA:		187	*	ZPG&STK TEST	
F5BA:		188	*		
F5BA:	9A	189	TXS		
F5BB:	8C EF FF	190	STY	BNKSW	
F5BE:	98	191	ZP1	TYA	
F5BF:	8D D0 FF	192	STA	ZPREG	
F5C2:	85 FF	193	STA	STK0	
F5C4:	C8	194	INY		
F5C5:	98	195	TYA		
F5C6:	48	196	PHA		
F5C7:	68	197	PLA		
F5C8:	C8	198	INY		
F5C9:	C0 20	199	CPY	#\$20	
F5CB:	D0 F1	200	BNE	ZP1	
F5CD:	A0 C0	201	LDY	#\$C0	
F5CF:	8C D0 FF	202	STY	ZPREG	
F5D2:	86 18	203	STX	PTRLO	
F5D4:	EB	204	ZP2	INX	
F5D5:	86 19	205	STX	PTRHI	
F5D7:	8A	206	TXA		
F5D8:	D1 18	207	CMP	PTRLO,Y	
F5DA:	D0 06	208	BNE	ZP3	
F5DC:	E0 1F	209	CPX	#\$1F	
F5DE:	D0 F4	210	BNE	ZP2	
F5E0:	F0 05	211	BEG	ROMTST	
F5E2:		212	ZP3	EQU	*
F5E2:	A2 1A	213	LDX	#\$1A	CHIP IS THERE, BAD ZERO AND STACK
F5E4:	20 7B F7	214	JSR	MESSERR	SO PRINT 'ZP' MESSAGE
F5E7:		215	*		& SET FLAG (2MHZ MODE)
F5E7:		216	*	ROM TEST ROUTINE	
F5E7:		217	*		
F5E7:	A9 00	218	ROMTST	LDA	#\$00
F5E9:	A8	219	TAY		SET POINTERS TO
F5EA:	A2 F0	220	LDY	#\$F0	\$F000
F5EC:	85 18	221	STA	PTRLO	
F5EE:	86 19	222	STX	PTRHI	SET X TO \$FF
F5F0:	A2 FF	223	LDX	#\$FF	FOR WINDOWING I/O
F5F2:	81 18	224	ROMTST1	EOR	(PTRLO),Y COMPUTE CHKSUM ON
F5F4:	E4 19	225	CPY	PTRHI	EACH ROM BYTE.
F5F6:	D0 06	226	BNE	ROMTST2	WINDOW OUT
F5F8:	C0 BF	227	CPY	#\$BF	RANGES FFC0-FFEF
F5FA:	D0 02	228	BNE	ROMTST2	
F5FC:	A0 EF	229	LDY	#\$EF	
F5FE:	C8	230	ROMTST2	INY	
F5FF:	D0 F1	231	BNE	ROMTST1	
F601:	E6 19	232	INC	PTRHI	
F603:	D0 ED	233	BNE	ROMTST1	
F605:	A8	234	TAY		TEST ACC. FOR 0
F606:	F0 05	235	BEG	VIAST	YES, NEXT TEST
F608:	A2 03	236	LDX	#\$03	PRINT 'ROM' AND
F60A:	20 7B F7	237	JSR	MESSERR	SET ERROR
F60D:		238	*		

4,383,296

63

64

Address	Hex	Disassembly	Comments
F60D:	239	* VIA TEST ROUTINE	
F60D:	240	*	
F60D: 18	241	VIATST CLC	SET UP FOR ADDING BYTES
F60E: D8	242	CLD	
F60F: AD E0 FF	243	LDA SYSEO	MASK OFF INPUT BITS
F612: 29 3F	244	AND #\$3F	AND STORE BYTE IN
F614: 85 18	245	STA PTRLO	TEMPOR. LOCATION
F616: AD EF FF	246	LDA BNKSW	MASK OFF INPUT BITS
F619: 29 4F	247	AND #\$4F	AND ADD TO STORED
F61B: 65 18	248	ADC PTRLO	BYTE IN TEMP. LOC.
F61D: 6D D0 FF	249	ADC ZPREG	ADD REMAINING
F620: 85 18	250	STA PTRLO	REGISTERS OF THE
F622: AD DF FF	251	LDA SYSD1	VIA'S
F625: 29 5F	252	AND #\$5F	(MASK THIS ONE)
F627: 65 18	253	ADC PTRLO	AND TEST
F629: 6D D2 FF	254	ADC SYSD2	TO SEE
F62C: 6D D3 FF	255	ADC SYSD3	IF THEY AGREE
F62E: 6D E2 FF	256	ADC SYSE2	WITH THE RESET
F632: 6D E3 FF	257	ADC SYSE3	CONDITION.
F635: C9 E1	258	CMP #\$E0+ROM	=E1?
F637: F0 05	259	BEQ ACIA	YES, NEXT TEST
F639: A2 06	260	LDX #\$06	NO, PRINT 'VIA' MESS.
F63B: 20 7B F7	261	JSR MESSERR	AND SET ERROR FLAG
F63E:	262	*	
F63E:	263	* ACIA TEST ROUTINE	
F63E:	264	*	
F63E: 18	265	ACIA CLC	SETUP FOR ADDITION
F63F: A9 9F	266	LDA #\$9F	MASK INPUT BITS
F641: 2D F1 C0	267	AND ACIAST	FROM STATUS REG
F644: 6D F2 C0	268	ADC ACIACM	AND ADD DEFAULT STATES
F647: 6D F3 C0	269	ADC ACIACN	OF CONTROL AND COMMND
F64A: C9 10	270	CMP #\$10	REGS. =10?
F64C: F0 05	271	BEQ ATD	YES, NEXT TEST
F64E: A2 09	272	LDX #\$09	NO, 'ACIA' MESSAGE AND
F650: 20 7B F7	273	JSR MESSERR	THEN SET ERROR FLAG
F653:	274	*	
F653:	275	* A/D TEST ROUTINE	
F653:	276	*	
F653: A9 C0	277	ATD LDA #\$C0	
F655: 8D DC FF	278	STA \$FFDC	
F658: AD 5A C0	279	LDA PDLEN+2	
F65B: AD 5E C0	280	LDA PDLEN+6	
F65E: AD 5C C0	281	LDA PDLEN+4	
F661: A0 20	282	LDY #\$20	
F663: 88	283	ADCTST1 DEY	WAIT FOR 40 USEC
F664: D0 FD	284	BNE ADCTST1	
F666: AD 5D C0	285	LDA PDLEN+5	SET A/D RAMP
F669: C8	286	ADCTST3 TNY	COUNT FOR CONVERSION
F66A: F0 0A	287	BEQ ADCERR	(255=ERROR)
F66C: AD 66 C0	288	LDA ADT0	IF BIT 7 =1?
F66F: 30 FB	289	BMI ADCTST3	YES, CONTINUE
F671: 98	290	TYA	NO, MOVE COUNT TO ACC
F672: 29 E0	291	AND #\$E0	ACC<32?
F674: F0 05	292	BEQ KEYPLUG	
F676:	293	ADCERR EQU *	NO,
F676: A2 0D	294	LDX #\$0D	PRINT 'A/D' MESS
F678: 20 7B F7	295	JSR MESSERR	AND SET ERROR FLAG
F67B:	296	*	
F67B:	297	* KEYBOARD PLUGIN TEST	
F67B:	298	*	
F67B: AD 08 C0	299	KEYPLUG LDA KEYBD	IS KYBD PLUGGED IN?
F67E: 0A	300	ASL A	(IS LIGHT CURRENT

4,383,296

65

66

F67F: 10 41	301	BPL	SEX	PRESENT?) NO, BRANCH
F681: AD DF FF	302	LDA	SYSD1	IS ERROR FLAG SET?
F684: 10 03	303	BPL	RECON	(2MHZ MODE) NO, BRANCH
F686: 4C 93 F3	304	JMP	ERRLP1	ERROR, HANG
F689:	305 *			
F689:	306 *			RECONFIGURE SYSTEM
F689:	307 *			
F689:	308	RECON	EGU	*
F689: A9 77	309	LDA	#\$77	TURN ON SCREEN
F68B: 8D DF FF	310	STA	SYSD1	
F68E: 20 98 FD	311	JSR	CLDSTRT	INITIALIZE MONITOR AND DEFAULT CHARACTER SET
F691: A9 10	312	LDA	#\$10	TEST FOR "APPLE 1"
F693: 2D 08 C0	313	AND	KEYBD	
F696: D0 09	314	BNE	BOOT	NO, DO REGULAR BOOT
F698: 2C 10 C0	315	BIT	KBDSTRB	CLEAR KEYBOARD
F69B: AD 50 C0	316	LDA	GRMD	
F69E: 20 01 FF	317	JSR	MONITOR	AND NEVER COME BACK.
F6A1: A2 01	318	BOOT	LDX	#1 READ BLOCK 0
F6A3: 86 B7	319	STX	IBCMD	
F6A5: CA	320	DEX		
F6A6: 86 85	321	STX	IBBUF	INTO RAM AT \$A000
F6A8: A9 A0	322	LDA	#\$A0	
F6AA: B5 86	323	STA	IBBUF+1	
F6AC: 4A	324	LSR	A	, FOR TRACK 80
F6AD: B5 91	325	STA	PREVTRK	MAKE IT RECALIBRATE TOO!
F6AF: 8A	326	TXA		
F6B0: 20 79 F4	327	JSR	BLOCKIO	
F6B3: 90 0A	328	BCC	GOBOOT	IF WE'VE SUCCEEDED, DO IT UP
F6B5: A2 1C	329	LDX	#\$1C	
F6B7: 20 38 F7	330	JSR	STRWT	RETRY
F6BA: 20 0F FD	331	JSR	KEYIN	
F6BD: B0 E2	332	BCS	BOOT	
F6BF: 4C 00 A0	333	GOBOOT	JMP	\$A000 GO TO IT FOOL...
F6C2:	334 *			
F6C2:	335 *			SYSTEM EXERCISER
F6C2:	336 *			
F6C2: A0 7F	337	SEX	LDY	#\$7F TRYFROM
F6C4: 98	338	SEX1	TYA	7F TO 0
F6C9: 29 FE	339		AND	#\$FE ADD =
F6C7: 49 4E	340		EOR	#\$4E 4EOR4F?
F6C9: FC 03	341		BEG	SEX2 YES, SKP
F6CB: B9 00 C0	342	LDA	KEYBD, Y	NO, CONT
F6CE: 89	343	DEX		NXT ADD
F6CF: D0 F3	344	BNE	SEX1	
F6D1: AD 51 C0	345	LDA	TXTMD	SET TXT
F6D4: B9 00 C1	346	SEX3	LDA	SLT1, Y EXERCSE
F6D7: B9 00 C2	347	LDA	SLT2, Y	ALL
F6DA: B9 00 C3	348	LDA	SLT3, Y	SLOTS
F6DD: B9 00 C4	349	LDA	SLT4, Y	
F6E0: AD FF CF	350	LDA	EXPROM	DISABLE EXPANSION ROM AREA
F6E3: C8	351	INY		
F6E4: D0 EE	352	BNE	SEX3	
F6E6:	353 *			
F6E6:	354 *			RAM TEST ROUTINE
F6E6:	355 *			
F6E6: A9 73	356	USRETRY	LDA	#\$72+ROM
F6E8: 8D DF FF	357	STA	SYSD1	
F6EB: A9 18	358	LDA	#\$18	
F6ED: 8D D0 FF	359	STA	ZPREG	
F6F0: A9 00	360	LDA	#\$00	
F6F2: A2 07	361	LDX	#\$07	
F6F4: 95 10	362	RAMTST0	STA	ZRPG1, X
F6F6: CA	363	DEX		
F6F7: 10 FB	364	BPL	RAMTST0	
F6F9: 20 84 F7	365	JSR	RAMSET	
F6FC: 08	366	PHP		
F6FD: 20 F7 F7	367	RAMTST1	JSR	RAMWT
F700: 20 F7 F7	368	JSR	RAMWT	

		4,383,296		
		67	68	
F703:	28	369	PLP	
F704:	6A	370	ROR	A
F705:	08	371	PHP	
F706:	20 A1 F7	372	JSR	PTRINC
F709:	D0 F2	373	BNE	RAMTST1
F70B:	20 84 F7	374	JSR	RAMSET
F70E:	08	375	PHP	
F70F:	20 FB F7	376	JSR	RAMRD
F712:	48	377	PHA	
F713:	A9 00	378	LDA	##00
F715:	91 18	379	STA	(PTRLO),Y
F717:	68	380	PLA	
F718:	28	381	PLP	
F719:	6A	382	ROR	A
F71A:	08	383	PHP	
F71B:	20 A1 F7	384	JSR	PTRINC
F71E:	D0 EF	385	BNE	RAMTST4
F720:		386	*	
F720:		387	*	RETURN TO START
F720:		388	*	
F720:	A9 00	389	LDA	##00
F722:	8D EF FF	390	STA	BNKSW
F725:	8D D0 FF	391	STA	ZPREG
F72B:	A2 07	392	LDX	##07
F72A:	8D 10 18	393	RAMTST6	LDA PHP, X
F72D:	95 10	394	STA	ZRPG1, X
F72F:	CA	395	DEX	
F730:	10 FB	396	BPL	RAMTST6
F732:	20 7E F7	397	JSR	ERROR
F735:	4C 75 F5	398	JMP	ERRLP
F738:		399	*****	
F738:		400	*	SARA TEST SUBROUTINES
F738:		401	*****	
F738:		402	*	
F738:		403	*	SUBROUTINE STRING WRITE
F738:		404	*	
F738:	BD 1D F4	405	STRWT	LDA CHPG, X
F738:	48	406	PHA	
F73C:	09 8D	407	ORA	##80 NORMAL VIDED
F73E:	20 2E FF	408	JSR	COUT & PRNT
F741:	ED	409	INX	NXT
F742:	5B	410	PLA	CHR
F743:	10 FB	411	BPL	STRWT
F745:	4C 07 FD	412	JMP	CROUT1 CLR TO END OF LINE
F748:		413	*	
F748:		414	*	SUBROUTINE RAM
F748:		415	*	
F748:	48	416	RAM	PHA SV ACC
F749:	8A	417	TXA	CONVRT
F74A:	4A	418	LSR	A ADD TO
F74B:	4A	419	LSR	A USE FOR
F74C:	4A	420	LSR	A B ENTRY
F74D:	4A	421	LSR	A
F74E:	08	422	PHP	
F74F:	4A	423	LSR	A
F750:	28	424	PLP	
F751:	AA	425	TAX	LOOKUP
F752:	BD 03 F4	426	LDA	RAMTBL, X IF VAL

			4,383,296			
			69			
					70	
F755	10	14	427	BPL	RAM0	NO GET
F757	48		428	FHA		WHICH
F758	AD	EF FF	429	LDA	BANKW	
F75B	29	0F	430	AND	##0F	
F75D	AA		431	TAX		
F75E	68		432	PLA		
F75F	E0	00	433	CPX	##00	
F761	F0	13	434	BEG	RAM1	BANK
F763	4A		435	LSR	A	SET
F764	4A		436	LSR	A	PROPER
F765	4A		437	LSR	A	RAM
F766	0A		438	BEQ		VALUE
F767	D0	00	439	ENB	RAM1	
F769	29	00	440	ADD	##05	CONVE
F76B	D0	19	441	RAND	RAM1	TO VAL
F76D	8A		442	TXA		
F76E	F0	02	443	BEG	RAM00	
F770	A9	03	444	TXA	#3	
F772	90	02	445	RAM00	RAM1	
F774	49	03	446	FOR	#3	
F776	29	07	447	RAM1	AND	BANKW
F778	AA		448	TAX		
F779	68		449	PLA		
F77A	60		450	RTS		
F77B			451	*		
F77E			452	* SUBROUTINE	ERRPR	
F77B			453	*		
F77B	20	08 FF	454	MESSAGE	USR	STRWT
F77E	A9	F3	455	ERROR	LDA	##F2+ROM
F780	8D	DF FF	456		STA	9YSD1
F780	60		457		RTS	MHZ 10
F784			458	*		
F784			459	* SUBROUTINE	RAMSET	
F784			460	*		
F784	A2	01	461	RAMSET	LDX	##01
F786	86	1A	462		STX	BNK
F788	A0	00	463		LDY	##00
F78A	A9	AA	464		LDA	##AA
F78C	38		465		SEC	
F78E	48		466	RAMSET	FHA	
F78E	02		467		PHP	
F78E	A5	1A	468		LDA	BNK
F791	09	80	469		ORA	##80
F793	8D	19 14	470		STA	IDNK
F796	A9	02	471		LDA	##02
F798	85	19	472		STA	PTRHI
F79A	A2	00	473		LDX	##00
F79C	86	18	474		STX	PTRLO
F79E	28		475		PLP	
F79F	68		476		PLA	
F7A0	60		477		RTS	
F7A1			478	*		
F7A1			479	* SUBROUTINE	PTRINC	
F7A1			480	*		
F7A1	48		481	PTRINC	PHA	

			71	4,383,296	72
F7A2	E6	18	482	INC	PTRLO
F7A4	D0	1D	483	BNE	RETS
F7A6	A5	1A	484	LDA	BNK
F7A8	10	0E	485	BPL	PINCL
F7AA	A5	19	486	LDA	PTRHI
F7AC	09	13	487	CMP	##13
F7AE	F0	06	488	BEG	PINC2
F7B0	09	17	489	CMP	##17
F7B2	D0	14	490	BNE	PINCL
F7B4	0A	17	491	INC	PTRHI
F7B6	06	1F	492	PLA	PTRHI
F7B8	E6	19	493	PINCL	INC
F7BA	D0	07	494	BNE	RETS
F7BC	0E	1A	495	BEF	BNK
F7BE	06	1A	496	LDY	BNK
F7C0	21	0D	497	DBF	RAMSET1
F7C2	08		498	PRCL	PLA
F7C4	A6	1A	499	LDA	BNK
F7C6	E0	FD	500	LRA	##FD
F7C8	50		501	RTS	
F7CA			502	*	
F7CC			503	* SUBROUTINE	RAMERR
F7CE			504	*	
F7D0	4E		505	RAMERR	PLA
F7D2	A6	19	506	LDY	PTRHI
F7D4	A4	1A	507	LDY	BNK
F7D6	00		508	PLA	
F7D8	0A	19	509	LDY	RAMERR4
F7DA	0A		510	LDY	
F7DC	30	1D	511	BPL	RAMERR5
F7DE	18		512	PLC	
F7E0	59	20	513	ADC	##20
F7E2	90	EF	514	RAMERR2	STY
F7E4	AA		515	TAX	
F7E6	20	4B	516	RAMERR3	JSR
F7E8	68	F7	517	PLA	RAM
F7EA	48		518	PLA	
F7EC	AD	0E	519	LDY	##00
F7EE	51	18	520	DBF	(PTRLO), Y
F7F0	15	10	521	ORA	ZRPG1, X
F7F2	95	10	522	STA	ZRPG1, X
F7F4	68		523	PLA	
F7F6	60		524	RTS	
F7F8	A9	0E	525	RAMERR4	LDA
F7FA	50	EF	526	STA	##00
F7FC	F0	EA	527	BEG	BNKSW
F7FE	38		528	RAMERR5	SEC
F800	E9	60	529	SBC	##60
F802	0B		530	INY	
F804	D0	00	531	BNE	RAMERR3
F806			532	*	
F808			533	* SUBROUTINE	RAMWT
F80A			534	*	
F80C	49	FF	535	RAMWT	EOR
F80E	07	18	536	STA	(PTRLO), Y

4,383,296

73

74

0000 01 18
0000 00 0A
0000 00 00

537 RAMRD
538
539

0000 00 00
0000 00 00
0000 00 00

*** SUCCESSFUL ASSEMBLY NO ERRORS

COF3 ACIACN	F63E ACIA	COF2 ACIACM	COF1 ACIAST
F676 ADCERR	F663 ADCTST1	F669 ADCTST3	?C047 ADRS
C066 ADTO	F653 ATD	F479 BLOCKIO	1A BNK
FFEF BNKSW	F6A1 BOOT	F4CD CHPG	FD98 CLDSTRT
F532 CNTWR	FC25 COUT	FD07 CROUT1	5D CV
F513 DISK1	C0D0 DISKOFF	F575 ERRLP	F593 ERRLP1
F77E ERROR	CFFF EXPROM	F6BF GOBOOT	C050 GRMD
85 IBBUFF	87 IBCMD	1419 IBNK	C010 KBDSTRB
C008 KEYBD	FD0F KEYIN	F67D KEYPLUG	C000 KYBD
F77B MESSERR	F901 MONITOR	F54F NMEM1	F562 NMEM2
F52A NOGOOD	F548 NOMEM	F526 NXBIT	F524 NXBYT
C058 PDLEN	1810 PHP	F7B8 PINC1	F7B6 PINC2
91 PREVTRK	19 PTRHI	F7A1 PTRINC	18 PTRLD
F53A PULBT	F772 RAM00	F776 RAM1	F748 RAM
F76B RAMO	F7DB RAMERR3	F7EA RAMERR4	F7C9 RAMERR
F7D7 RAMERR2	F7F1 RAMERR5	F7FB RAMRD	F784 RAMSET
F78D RAMSET1	F4C5 RAMTBL	F6F4 RAMTST0	F6FD RAMTST1
F70F RAMTST4	F72A RAMTST6	F5A1 RAMWT1	F5AD RAMWT4
F7F7 RAMWT	F5A5 RAMWT2	F689 RECON	F7C3 RETS
F5F2 ROMTST1	F5FE ROMTST2	F5E7 ROMTST	01 ROM
FBC7 SETCVH	FD9D SETUP	F6C4 SEX1	F6C2 SEX
F6CE SEX2	F6D4 SEX3	C100 SLT1	C200 SLT2
C300 SLT3	C400 SLT4	FF STKO	F738 STRWT
FFDF SYSD1	FFD2 SYSD2	FFD3 SYSD3	FFE0 SYSEO
FFE2 SYSE2	FFE3 SYSE3	C051 TXTMD	?F6E6 USRENTY
F60D VIATST	F5BE ZP1	F5D4 ZP2	F5E2 ZP3
FFD0 ZPRG	10 ZRPG1	00 ZRPG	
00 ZRPG	01 ROM	10 ZRPG1	18 PTRLD
19 PTRHI	1A BNK	3D CV	85 IBBUFF
87 IBCMD	91 PREVTRK	FF STKO	1419 IBNK
1810 PHP	C000 KYBD	C008 KEYBD	C010 KBDSTRB
?C047 ADRS	C050 GRMD	C051 TXTMD	C058 PDLEN
C066 ADTO	C0D0 DISKOFF	COF1 ACIAST	COF2 ACIACM
COF3 ACIACN	C100 SLT1	C200 SLT2	C300 SLT3
C400 SLT4	CFFF EXPROM	F479 BLOCKIO	F4C5 RAMTBL
F4CD CHPG	F513 DISK1	F524 NXBYT	F526 NXBIT
F52A NOGOOD	F532 CNTWR	F53A PULBT	F548 NOMEM
F54F NMEM1	F562 NMEM2	F575 ERRLP	F593 ERRLP1
F5A1 RAMWT1	F5A5 RAMWT2	F5AD RAMWT4	F5BE ZP1
F5D4 ZP2	F5E2 ZP3	F5E7 ROMTST	F5F2 ROMTST1
F5FE ROMTST2	F60D VIATST	F63E ACIA	F693 ATD
F663 ADCTST1	F669 ADCTST3	F676 ADCERR	F678 KEYPLUG
F689 RECON	F6A1 BOOT	F6BF GOBOOT	F6C2 SEX
F6C4 SEX1	F6CE SEX2	F6D4 SEX3	?F6E6 USRENTY
F6F4 RAMTST0	F6FD RAMTST1	F70F RAMTST4	F72A RAMTST6
F738 STRWT	F748 RAM	F76D RAMO	F772 RAM00
F776 RAM1	F77B MESSERR	F77E ERROR	F784 RAMSET
F78D RAMSET1	F7A1 PTRINC	F7D6 PINC2	F7B8 PINC1
F7C3 RETS	F7C9 RAMERR	F7D7 RAMERR2	F7DB RAMERR3
F7EA RAMERR4	F7F1 RAMERR5	F7F7 RAMWT	F7FB RAMRD
F901 MONITOR	FBC7 SETCVH	FC25 COUT	FD07 CROUT1
FD0F KEYIN	FD98 CLDSTRT	FD9D SETUP	FFD0 ZPRG
FFD2 SYSD2	FFD3 SYSD3	FFDF SYSD1	FFE0 SYSEO
FFE2 SYSE2	FFE3 SYSE3	FFEF BNKSW	

----- NEXT OBJECT FILE NAME IS MON.OBJ

F7FF: 2 ORG *F7FF

F7FF: 3 *

75

4,383,296

76

E7FF		4 *		
E7FF	50	5	RETJ	RTS
F800	E9 01	6		SBC #1
F802	F0 FB	7		BEG RETJ
F804	E9 01	8		SBC #1
F806	F0 F7	9		BEG RETJ
F808	E9 01	10		SBC #1
F80A	F0 F3	11		BEG RETJ
F80C	E9 01	12		SBC #1
F80E	F0 EF	13		BEG RETJ
F810	E9 01	14		SBC #1
F812	F0 EB	15		BEG RETJ
F814	F0 E7	16		SBC #1
F816	F0 F7	17		BEG RETJ
F818	E9 01	18		SBC #1
F81A	F0 E3	19		BEG RETJ
F81C	E9 01	20		SBC #1
F81E	F0 CF	21		BEG RETJ
F820	E9 01	22		SBC #1
F822	F0 DB	23		BEG RETJ
F824	E9 01	24		SBC #1
F826	F0 D7	25		BEG RETJ
F828	E9 01	26		SBC #1
F82A	F0 DB	27		BEG RETJ
F82C	E9 01	28		SBC #1
F82E	F0 D7	29		BEG RETJ
F830	E9 01	30		SBC #1
F832	F0 CB	31		BEG RETJ
F834	E9 01	32		SBC #1
F836	F0 C7	33		BEG RETJ
F838	E9 01	34		SBC #1
F83A	F0 C3	35		BEG RETJ
F83C	E9 01	36		SBC #1
F83E	F0 BF	37		BEG RETJ
F840	E9 01	38		SBC #1
F842	F0 B3	39		BEG RETJ
F844	E9 01	40		SBC #1
F846	F0 B7	41		BEG RETJ
F848	E9 01	42		SBC #1
F84A	F0 B3	43		BEG RETJ
F84C	E9 01	44		SBC #1
F84E	F0 A7	45		BEG RETJ
F850	E9 01	46		SBC #1
F852	F0 A3	47		BEG RETJ
F854	E9 01	48		SBC #1
F856	F0 A7	49		BEG RETJ
F858	E9 01	50		SBC #1
F85A	F0 A3	51		BEG RETJ
F85C	E9 01	52		SBC #1
F85E	F0 BF	53		BEG RETJ
F860	E9 01	54		SBC #1
F862	F0 9B	55		BEG RETJ
F864	E9 01	56		SBC #1
F866	F0 97	57		BEG RETJ
F868	E9 01	58		SBC #1

77

4,383,296

78

F86A: F0 93	59	BEG	RET1
F86C: E9 01	60	SBC	#1
F86E: F0 8F	61	BEG	RET1
F870: E9 01	62	SBC	#1
F872: F0 8B	63	BEG	RET1
F874: E9 01	64	SBC	#1
F876: F0 87	65	BEG	RET1
F878: E9 01	66	SBC	#1
F87A: F0 83	67	BEG	RET1
F87C: E9 01	68	SBC	#1
F87E: F0 02	69	BEG	RET3
F880: E9 01	70	SBC	#1
F882: F0 7C	71	BEG	RET2
F884: E9 01	72	SBC	#1
F886: F0 78	73	BEG	RET2
F888: E9 01	74	SBC	#1
F88A: F0 74	75	BEG	RET2
F88C: E9 01	76	SBC	#1
F88E: F0 70	77	BEG	RET2
F890: E9 01	78	SBC	#1
F892: F0 6C	79	BEG	RET2
F894: E9 01	80	SBC	#1
F896: F0 68	81	BEG	RET2
F898: E9 01	82	SBC	#1
F89A: F0 64	83	BEG	RET2
F89C: E9 01	84	SBC	#1
F89E: F0 60	85	BEG	RET2
F8A0: E9 01	86	SBC	#1
F8A2: F0 5C	87	BEG	RET2
F8A4: E9 01	88	SBC	#1
F8A6: F0 58	89	BEG	RET2
F8A8: E9 01	90	SBC	#1
F8AA: F0 54	91	BEG	RET2
F8AC: E9 01	92	SBC	#1
F8AE: F0 50	93	BEG	RET2
F8B0: E9 01	94	SBC	#1
F8B2: F0 4C	95	BEG	RET2
F8B4: E9 01	96	SBC	#1
F8B6: F0 48	97	BEG	RET2
F8B8: E9 01	98	SBC	#1
F8BA: F0 44	99	BEG	RET2
F8BC: E9 01	100	SBC	#1
F8BE: F0 40	101	BEG	RET2
F8C0: E9 01	102	SBC	#1
F8C2: F0 3C	103	BEG	RET2
F8C4: E9 01	104	SBC	#1
F8C6: F0 38	105	BEG	RET2
F8C8: E9 01	106	SBC	#1
F8CA: F0 34	107	BEG	RET2
F8CC: E9 01	108	SBC	#1
F8CE: F0 30	109	BEG	RET2
F8D0: E9 01	110	SBC	#1
F8D2: F0 2C	111	BEG	RET2
F8D4: E9 01	112	SBC	#1

RET3

4,383,296

79

80

F8D0: F0 28	113	BEG	RET2
F8D8: E9 01	114	SBC	#1
F8EA: F0 24	115	BEG	RET2
F8EC: E9 01	116	SBC	#1
F8EE: F0 20	117	BEG	RET2
F8F0: E9 01	118	SBC	#1
F8F2: F0 1C	119	BEG	RET2
F8F4: E9 01	120	SBC	#1
F8F6: F0 18	121	BEG	RET2
F8F8: E9 01	122	SBC	#1
F8FA: F0 14	123	BEG	RET2
F8FC: E9 01	124	SBC	#1
F8FE: F0 10	125	BEG	RET2
F900: E9 01	126	SBC	#1
F902: F0 0C	127	BEG	RET2
F904: E9 01	128	SBC	#1
F906: F0 08	129	BEG	RET2
F908: E9 01	130	SBC	#1
F90A: F0 04	131	BEG	RET2
F90C: E9 01	132	SBC	#1
F90E: F0 00	133	BEG	RET2
F910: E9 01	134	SBC	#1
F912: F0 20	134	BEG	RET2
F901:	135	LNH	MUN9A
F901:	2 *		
F901:	3 *		
005B:	4	SCRNLDC EQU	\$5B
F901:	5 *		
005B:	6	LMARGIN EQU	SCRNLDC
0059:	7	RMARGIN EQU	SCRNLDC+1
005A:	8	WINTOP EQU	SCRNLDC+2
005B:	9	WINETM EQU	SCRNLDC+3
005C:	10	CH EQU	SCRNLDC+4
005D:	11	CV EQU	SCRNLDC+5
005E:	12	BAS4L EQU	SCRNLDC+6
005F:	13	BAS4H EQU	SCRNLDC+7
0060:	14	BAS8L EQU	SCRNLDC+8
0061:	15	BAS8H EQU	SCRNLDC+9
0062:	16	TBAS4L EQU	SCRNLDC+\$A
0063:	17	TBAS4H EQU	SCRNLDC+\$B
0064:	18	TBAS8L EQU	SCRNLDC+\$C
0065:	19	TBAS8H EQU	SCRNLDC+\$D
0066:	20	FORGND EQU	SCRNLDC+\$E
0067:	21	BKGND EQU	SCRNLDC+\$F
0068:	22	MODES EQU	SCRNLDC+\$10
0069:	23	CURSOR EQU	SCRNLDC+\$11
006A:	24	STACK EQU	SCRNLDC+\$12
006B:	25	PROMPT EQU	SCRNLDC+\$13
006C:	26	TEMPX EQU	SCRNLDC+\$14
006D:	27	TEMPY EQU	SCRNLDC+\$15
006E:	28	CSWL EQU	SCRNLDC+\$16
006F:	29	CSWH EQU	SCRNLDC+\$17
0070:	30	KSWL EQU	SCRNLDC+\$18
0071:	31	KSWH EQU	SCRNLDC+\$19
0072:	32	PCL EQU	SCRNLDC+\$1A
0073:	33	PCH EQU	SCRNLDC+\$1B
0074:	34	A1L EQU	SCRNLDC+\$1C
0075:	35	A1H EQU	A1L+1
0076:	36	A2L EQU	A1L+2
0077:	37	A2H EQU	A1L+3
0078:	38	A3L EQU	A1L+4
0079:	39	A3H EQU	A1L+5

4,383,296

81

82

007A:	40 A4L	EQU	A1L+6	
007B:	41 A4H	EQU	A1L+7	
007C:	42 STATE	EQU	A1L+8	
007D:	43 YSAV	EQU	A1L+9	
007E:	44 INBUF	EQU	A1L+#A	; AND #B
0080:	45 TEMP	EQU	A1L+#C	
0069:	46 MASK	EQU	CURSOR	
F901:	47 *			
C000:	48 KBD	EQU	%C000	
C010:	49 KBDSTRB	EQU	%C010	
F901:	50 *			
03F8:	51 USERADR	EQU	%3F8	
F479:	52 BLOCKIO	EQU	%F479	
F689:	53 RECON	EQU	%F689	AS OF 12/20/79
F4EE:	54 DIAGN	EQU	%F4EE	
0050:	55 INBUFLEN	EQU	%50	; ONLY 80 BYTES (%3A0-3EF)
0081:	56 IBSLOT	EQU	%81	
0082:	57 IBDRVN	EQU	IBSLOT+1	
0085:	58 IBBUFP	EQU	IBSLOT+4	
0087:	59 IBCMD	EQU	IBSLOT+6	
F901	60 *			
F901:	61 ENTRY	EQU	*	
F901'DA	62	TSX		
F902 86 6A	63	STX	STACK	
F904	64 *			
F904: D8	65 MON	CLD		; MUST BE HEX MODE
F905: 20 3A FC	66	JSR	DELL	
F908 A6 6A	67 MONZ	LDX	STACK	; RESTORE STACK TO ORIGINAL LOCATION
F90A 9A	68	TXS		
F90B: A9 DF	69	LDA	#\$DF	; PROMPT (APPLE) FOR SARA MONITOR
F90D 85 6B	70	STA	PROMPT	
F90F: 20 D5 FC	71	JSR	GETLNZ	; GET A LINE OF INPUT
F912: 20 67 F9	72 SCAN	JSR	ZSTATE	; SET REGULAR SCAN
F915: 20 2C F9	73 NXTINP	JSR	GETNUM	; ATTEMPT TO READ HEX BYTE
F918 84 7D	74	STY	YSAV	; STORE CURRENT INPUT POINTER
F91A A0 11	75	LDY	#\$11	; 17 COMMANDS
F91C 8B	76 CMDSRCH	BEY		
F91D: 30 E5	77	BMI	MON	; GIVE UP IF UNRECOGNIZABLE
F91F: D9 6C F9	78	CMP	CMDTAB.Y	; FOUND?
F922: D0 F8	79	BNE	CMDSRCH	; NO KEEP LOOKING
F924: 20 5E F9	80	JSR	TOSUB	; PERFORM FUNCTION
F927 A4 7D	81	LDY	YSAV	; GET NEXT POINTER
F929 4C 15 F9	82	JMP	NXTINP	; DO NEXT COMMAND
F92C:	83 *			
F92C: A2 00	84 GETNUM	LDX	#0	; CLEAR A2
F92E 86 76	85	STX	A2L	
F930 86 77	86	STX	A2H	
F932 B1 7E	87 NXTCHR	LDA	(INBUF).Y	
F934 C8	88	INY		; BUMP INDEX FOR NEXT TIME
F935 49 D0	89	EOR	#\$B0	
F937 C9 0A	90	CMP	#\$A	; TEST FOR DIGIT
F939 90 06	91	BCC	DIGIT	; SAVE IT IF 1-9
F93B 69 8B	92	ADC	#\$B8	; TEST FOR HEX A-F
F93D C9 FA	93	CMP	#\$FA	
F93F 90 2A	94	BCC	DIGRET	
F941: A2 03	95 DIGIT	LDX	#3	
F943: 0A	96	ASL	A	
F944: 0A	97	ASL	A	
F945: 0A	98	ASL	A	
F946: 0A	99	ASL	A	
F947: 0A	100 NXTBIT	ASL	A	; SHIFT HEX DIGITS INTO A2
F948: 26 76	101	ROL	A2L	
F94A: 26 77	102	ROL	A2H	
F94C: CA	103	DEX		; SHIFTED ALL YET?
F94D: 10 F8	104	BPL	NXTBIT	
F94F: A5 7C	105 NXTBAS	LDA	STATE	
F951: D0 06	106	BNE	NXTBS2	; IF ZERO THEN COPY TO A1D3

4,383,296

83

84

F953: B5 77	107	LDA	A2H, X	
F955: 95 75	108	STA	A1H, X	
F957: 95 79	109	STA	A3H, X	
F959: EB	110	NXTBS2	INX	
F95A: F0 F3	111		BEQ	NXTBAS
F95C: D0 D4	112		BNE	NXTCHR
F95E:	113	*		
F95E: A9 FA	114	TOSUB	LDA	#ASCII ; PUSH ADDRESS OR FUNCTION
F960: 4B	115		PHA	; AND RETURN TO IT.
F961: B9 7C F9	116		LDA	CMDVEC, Y
F964: 4B	117		PHA	
F965: A5 7C	118		LDA	STATE ; PASS MODE VIA ACC.
F967: A0 00	119	ZSTATE	LDY	#0
F969: 84 7C	120		STY	STATE ; RESET STATE OF SCAN
F96B: 60	121	DIGRET	RTS	
F96C:	122	*		
F96C:	123	CMDTAB	EGU	*
F96C: 00	124		DFB	\$0 ; G =GO (CALL) SUBROUTINE
F96D: 03	125		DFB	\$3 ; J =JUMP (CONT) PROGRAM
F96E: 06	126		DFB	\$6 ; M =MOVE MEMORY
F96F: EB	127		DFB	\$EB ; R =READ DISK BLOCK
F970: EE	128		DFB	\$EE ; U =USER FUNCTION
F971: EF	129		DFB	\$EF ; V =VERIFY MEMORY BLOCKS
F972: F0	130		DFB	\$F0 ; W =WRITE DISK BLOCK
F973: F1	131		DFB	\$F1 ; X =REPEAT LINE OF COMMANDS
F974: 99	132		DFB	\$99 ; SP =SPACE (DYTE SEPARATOR)
F975: 9B	133		DFB	\$9B ; " =ASCII (HI BIT ON)
F976: A0	134		DFB	\$A0 ; ' =ASCII (HI BIT OFF)
F977: 93	135		DFB	\$93 ; : =SET STORE MODE
F978: A7	136		DFB	\$A7 ; . =RANGE SEPARATOR
F979: AB	137		DFB	\$AB ; / =COMMAND SEPARATOR
F97A: 95	138		DFB	\$95 ; < =DEST/SOURCE SEPARATOR
F97B: C6	139		DFB	\$C6 ; CR =CARRAGE RETURN
F97C:	140	*		
F97C:	141	CMDVEC	EGU	*
F97C: 7C	142		DFB	GO-1
F97D: 7A	143		DFB	JUMP-1
F97E: 2B	144		DFB	MOVE-1
F97F: DF	145		DFB	READ-1
F980: 77	146		DFB	USER-1
F981: 3A	147		DFB	VRFY-1
F982: C2	148		DFB	WRTE-1
F983: 18	149		DFB	REPEAT-1
F984: A3	150		DFB	SPCE-1
F985: 06	151		DFB	ASCII-1
F986: 06	152		DFB	ASCII0-1
F987: 27	153		DFB	SETMODE-1
F988: B7	154		DFB	SETMODE-1
F989: 99	155		DFB	SEP-1
F98A: 90	156		DFB	DEST-1
F98B: 25	157		DFB	CRMON-1
F98C:	158	*		
F98C:	159	*		
F98C: E6 7A	160	NXTA4	INC	A4L ; BUMP 16 BIT POINTERS
F98E: D0 02	161		BNE	NXTA1
F990: E6 7B	162		INC	A4H
F992: E6 74	163	NXTA1	INC	A1L ; BUMP A1
F994: 20 05	164		BNE	TSTA1
F996: E6 75	165		INC	A1H
F998: 3E	166		SEC	; IN CASE OF ROLL OVER.
F999: F0 10	167		BEQ	RETA1
F99B: A5 74	168	TSTA1	LDA	A1L ; TEST A1DA2
F99D: 3E	169		SEC	
F99E: E5 76	170		SBC	A2L
F9A0: 85 80	171		STA	TEMP

4,383,296

85

86

F9A2: A5 75	172	LDA	A1H	
F9A4: E5 77	173	SBC	A2H	
F9A5: 09 30	174	ORA	TEMP	
F9A8: D0 01	175	BNE	RETA1	; IF A1 LESS THAN OR EQUAL TO A2
F9AA: 18	176	CLC		; THEN CARRY CLEAR ON RETURN
F9AB: 60	177	RETA1	RTS	
F9AC:	178	*		
F9AC:	179	*		
F9AC: 48	180	PRBYTE	PHA	; SAVE LOW NIBBLE
F9AD: 4A	181	LSR	A	
F9AE: 4A	182	LSR	A	; SHIFT HI NIBBLE TO PRINT.
F9AF: 4A	183	LSR	A	
F9B0: 4A	184	LSR	A	
F9B1: 20 B7 F9	185	JSR	PRHEXZ	
F9B4: 6B	186	PLA		
F9B5: 29 0F	187	PRHEX	AND	##0F ; STRIP HI NIBBLE
F9B7: 09 B0	188	PRHEXZ	ORA	##B0 ; MAKE IT NUMERIC
F9B9: C9 BA	189	CMP	##BA	; IS IT '>'9'
F9BB: 90 02	190	BCC	PRHEX2	
F9BD: 69 06	191	ADC	##6	; MAKE IT 'A'-'F'
F9BF: 4C 25 FC	192	PRHEX2	JMP	CDUT
F9C2:	193	*		
F9C2: 20 AC F9	194	PRBYCOL	JSR	PRBYTE
F9C5:	195	*		
F9C5: A9 BA	196	PRCOLON	LDA	##BA ; PRINT A COLON
F9C7: D0 F6	197	BNE	PRHEX2	; BRANCH ALWAYS
F9C9:	198	*		
F9C9: A9 07	199	TSTBOWID	LDA	#7 ; ANTICIPATE
F9CB: 24 68	200	BIT	MODES	; TEST FOR 80
F9CD: 50 02	201	BVC	SVMASK	
F9CF: A9 0F	202	LDA	##F	
F9D1: 85 69	203	SVMASK	STA	MASK
F9D3: 60	204		RTS	
F9D4:	205	*		
F9D4: 8A	206	A1PC	TXA	; TEST FOR NEW PC
F9D5: F0 07	207	BEG	OLDPC	
F9D7: B5 74	208	A1PC1	LDA	A1L, X
F9D9: 95 72	209		STA	PCL, X
F9DB: CA	210		DEX	
F9DC: 10 F9	211		BPL	A1PC1
F9DE: 60	212	OLDPC	RTS	
F9DF:	213	*		
F9DF: 85 69	214	ASCII1	STA	MASK ; SAVE HI BIT STATUS
F9E1: A4 7D	215	ASCII2	LDY	YSAV ; MOVE ASCII TO MEMORY
F9E3: B1 7E	216		LDA	(INBUF), Y
F9E5: E6 7D	217		INC	YSAV ; BUMP FOR NEXT THING.
F9E7: A0 00	218		LDY	#0
F9E9: C9 A2	219		CMP	##A2 ; ASCII " ?
F9EB: D0 05	220		BNE	ASCII3 ; NOPE, CONTINUE.
F9ED: A5 69	221		LDA	MASK
F9EF: 10 20	222		BPL	BITON ; HE'S CHANGED MODES.
F9F1: 60	223		RTS	; NO, HE'S DONE.
F9F2: C9 A7	224	ASCII3	CMP	##A7 ; ASCII ' ?
F9F4: D0 05	225		BNE	CRCHK ; NO, TEST FOR EOL.
F9F6: A5 69	226		LDA	MASK
F9F8: 30 1B	227		BMI	BITOFF ; CHANGE MODES.
F9FA: 60	228		RTS	
F9FB: C9 8D	229	CRCHK	CMP	##8D ; END OF LINE?
F9FD: F0 07	230		BEG	ASCDONE ; YES, FINISHED
F9FF: 25 69	231		AND	MASK
FA01: 20 AF FA	232		JSR	STOR1 ; GO STORE IT!
FA04: D0 DB	233		BNE	ASCII2 ; DO NEXT.

4,383,296

87

88

FA06: 60	234	ABCDONE	RTS		
FA07:	235	*			
FA07: 38	236	ASCII	SEC		INDICATE HI ON.
FA08: 90	237		DFB	\$90	(BCC - NEVER TAKEN)
FA09: 18	238	ASCII0	CLC		INDICATE HI OFF
FA0A: AA	239	CKMDE	TAX		SAVE STATE
FA0B: 86 7C	240		STX	STATE	RETAIN STATE
FA0D: 49 BA	241		EOR	#\$BA	ARE WE IN STORE MODE?
FA0F: D0 7D	242		DNE	ERROR	
FA11: A7 FF	243	DITON	LDA	#\$FF	SET HI BIT UNMASKED
FA13: D0 CA	244		BCS	ASCII1	
FA15: A9 7F	245	DITOFF	LDA	#\$7F	MASK HI BIT
FA17: 10 C6	246		BPL	ASCII1	ALWAYS
FA19: 20 00 C0	247	REPEAT	BIT	KBD	REPEAT UNTIL KEYPRESS
FA1C: 10 03	248		BPL	REPEAT1	
FA1E: 40 0F FD	249		JMP	KEYIN	
FA21: 68	250	REPEAT1	PLA		CLEAN UP STACK
FA22: 68	251		PLA		
FA23: 4C 12 F9	252		JMP	SCAN	
FA26:	253	*			
FA26:	254	*			
FA26: 20 A0 FA	255	CRMON	JSR	BL1	
FA28: 4C 08 F9	256		JMP	MONZ	
FA2C:	257	*			
FA2C: 20 9B F9	258	MOVE	JSR	TSTA1	DON'T MOVE ANYTHING IF ILLEGAL INPUT
FA2F: B0 5D	259		BCS	ERROR	
FA31: B1 74	260	MOVNXT	LDA	(A1L),Y	MOVE A BYTE
FA33: 91 7A	261		STA	(A4L),Y	
FA35: 20 9C F9	262		JSR	NXTA4	DUMP BOTH A1 AND A4
FA38: 90 F7	263		BCC	MOVNXT	
FA3A: 60	264		RTS		ALL DONE WITH MOVE
FA3B:	265	*			
FA3B:	266	*			
FA3E: 20 9B F9	267	VRFY	JSR	TSTA1	TEST VALID RANGE
FA3E: B0 4E	268		BCS	ERROR	
FA40: B1 74	269	VRFY1	LDA	(A1L),Y	COMPARE BYTE FOR BYTE
FA42: D1 7A	270		CMP	(A4L),Y	MATCH?
FA44: F0 06	271		BEG	VRFY2	YES, DO NEXT.
FA46: 20 52 FA	272		JSR	MISMATCH	PRINT BOTH BYTES
FA49: 20 EF FC	273		JSR	CROUT	GOTO NEWLINE
FA4C: 20 8C F9	274	VRFY2	JSR	NXTA4	DUMP BOTH A1 AND A4
FA4F: 90 EF	275		BCC	VRFY1	
FA51: 60	276		RTS		VERIFY DONE.
FA52:	277	*			
FA52: A5 7B	278	MISMATCH	LDA	A4H	PRINT ADDRESS OF A4
FA54: 20 AC F9	279		JSR	PRBYTE	
FA57: A5 7A	280		LDA	A4L	
FA59: 20 C2 F9	281		JSR	PRBYCOL	OUTPUT A COLON FOR SEPARATOR
FA5C: D1 7A	282		LDA	(A4L),Y	AND THE DATA...
FA5E: 20 70 FA	283		JSR	PRBYTSP	PRINT THE BYTE AND A SPACE
FA61: 20 73 FA	284	PRINTA1	JSR	PRSPC	LEAD WITH A SPACE
FA64: A5 75	285		LDA	A1H	OUTPUT ADDRESS A1
FA66: 20 AC F9	286		JSR	PRBYTE	
FA69: A5 74	287		LDA	A1L	
FA6B: 20 C2 F9	288		JSR	PRBYCOL	SEPARATE WITH A COLEN
FA6E: B1 74	289	PRA1BYTE	LDA	(A1L),Y	PRINT BYTE POINTED TO BY A1
FA70: 20 AC F9	290	PRBYTSP	JSR	PRBYTE	
FA73: A5 A0	291	PRSPC	LDA	#\$A0	PRINT A SPACE
FA75: 4C 25 FC	292		JMP	COUT	END VIA OUTPUT ROUTINE.
FA78:	293	*			
FA78: 4C F8 05	294	USER	JMP	USERADR	
FA7B:	295	*			
FA7B: 68	296	JUMP	PLA		
FA7C: 68	297		PLA		LEAVE STACK WITH NOTHING ON IT
FA7D: 20 D4 F9	298	GD	JSR	A1PC	STUFF PROGRAM COUNTER
FA80: 6C 72 0C	299		JMP	(PCL)	JUMP TO USER PROG
FA83:	300	*			
FA83:	301	HWERROR	EQU	*	PRINT ERROR NUMBER

4,383,296

89

90

FAB2	20	AC	F9	302	JSR	RRBYTE	PRINT THE OFFENDER
FAB6	A9	A1		303	LDA	##A1	FOLLOWED BY A "!"
FAB8	20	25	FC	304	JSR	COUT	
FABB	20	07	FD	305	JSR	NOSTOP	OUTPUT A CARRAGE RETURN AND STOPLIST
FABE	4C	04	F9	306	JMP	MON	
FAP1				307	*		
FAP1	A5	7A		308	DEST	LDA	A2L
FA93	B5	7A		309		STA	A4L
FA95	A5	77		310		LDA	A2H
FA97	B5	7B		311		STA	A4H
FA99	B0			312		RTS	
FA9A				313	*		
FA9A	B0	44	FA	314	SEP	JSR	SPACE
FA9D	98			315		TYA	SEPARATOR TEST STORE MODE OR DUMP
FA9E	FD	1D		316		BEQ	ZERO MODE
FAA0				317	*		BRANCH ALWAYS
FAA2	15	1D		318	DL1	DEC	YSAV
FAA3	FD	15		319		BEQ	NO LINE
FAA4	CA			320	SPACE	DEY	IF NO LINE, GIVEN A ROW OF BYTES
FAA5	D0	16		321		BNE	TEST IF AFTER ANOTHER SPACE
FAA7	C9	DA		322		CMF	SETMDZ
FAA9	D0	4B		323		BNE	STORE MODE?
FAAB	B5	7E		324	STOP	STA	KEEP IT IN STORE STATE
FAAD	A5	7A		325		LDA	KEEP BYTE TO BE STORED
FAAF	91	7B		326	STOP1	SEA	PUT IT IN MEMORY
FAP1	E6	7B		327		INC	BUMP POINTER
FAB1	D0	02		328		BNE	DUMMY
FAB5	E6	79		329		INC	ACH
FAP1	B0			330	DUMMY	RTS	ALSO USED FOR CLEAR MODE
FA				331			
FAP1	A4	1D		332	MODE	LDY	LINE INPUT CHARACTER
FAP1	1F			333		DEY	
FAB8	B1	7E		334		LDA	(INBUFF) TO SET MODE
FABE	B0	7C		335	SETMDZ	STA	STATE
FABF	B0			336		RTS	
FA				337	*		
FAC1	A7	01		338	READ	LDA	#1
FAC2	20			339		DFB	SET DISK COMMAND TO READ
FAC3	A7	02		340	WRTE	LDA	#2
FAC5	B5	B7		341	SAVEMD	STA	DUMMY BIT TO SKIP 2 BYTES
FAC7	A5	74		342	RWLOOP	LDA	SET DISK COMMAND TO WRITE
FAC9	B5	B5		343		STA	IBCMD
FAD1	A5	75		344		LDA	COMMAND FORMAT IS
FAD1	B5	B6		345		STA	IBBUFF
FAD3	A5	75		346		LDA	BLOCKNUMBER ADDRESS AND ADDRESS
FAD5	B5	B6		347		STA	IBBUFF+1
FAD6	A5	7E		348		LDX	SEND BLOCK NUMBER VIA X & A
FAD7	A5	7A		349		LDA	A4L
FAD7	B0	AA		350		SEI	NO INTERRUPTS WHILE IN MONITOR
FAD9	E6	7A		351		JSR	BLOCKIO
FABB	D0	02		352		BCS	DO DISK FEVER
FABD	E6	7B		353		INC	GIVE UP IF ERROR ENCOUNTERED
FABF	E6	75		354	NONVER	INC	DUMP BLOCK NUMBER
FAC1	E6	75		355		INC	NONVER
FAC3	20	9B	F9	356		JSR	NONVER
FAC6	90	DF		357		DCC	TEST FOR FINISHED
FAC8	B0			358		RTS	NOT DONE, DO NEXT BLOCK
FAP1				359	*		
FAC9				360		CHN	MON9B
FAC9				1	DUMPO	EGU	* OUTPUT 1 ROW OF BYTES
FACB	B5	77		2		LDA	A1H
FACD	20	09	F9	3		STA	A2H
FAD0	05	74		4		JSR	TSTROWID
FAD2	B5	76		5		ORA	SET WIDTH MASK INTO ACC
FAD4	D0	06		6		STA	A2L
FAD6				7		BNE	DUMPO
FAD6				8	*		BRANCH ALWAYS
FAD6	4A			9	TSTDUMP	LSR	A DUMP?
FAD7	B0	95		10	ERROR1	BCS	ERROR

4,383,296

91

92

FB49 20 09 FF	11 DUMP	JSR	TSTBOWID	TEST FOR EITHER 80 OR 40 COLUMNS
FB4F 45 74	12 DUMP	LDA	A1L	USE A4 FOR ASCII DUMP
FB4E 85 7A	13	STA	A4L	
FB00 45 75	14	LDA	A1H	
FB02 95 7B	15	STA	A4H	
FB04 20 9B FF	16	JSR	TSTAI	TEST FOR VALID RANGE
FB07 80 EE	17	BCS	ERROR1	
FB07 20 81 FF	18 DUMP	JSR	PRINTAI	PRINT ADDRESS AND FIRST BYTE
FB0C 20 92 FF	19 DUMP	JSR	NXTAI	
FB0F 80 10	20	BCS	DUMPASC	END WITH ASCII
FB11 45 74	21	LDA	A1L	TEST END OF LINE
FB17 25 59	22	AND	MASK	FOR 40/80 COLUMN
FB15 20 05	23	BNE	DUMP3	
FB17 20 21 FF	24	JSR	DUMPASC	
FB1A 80 ED	25	BNE	DUMP1	BRANCH ALWAYS
FB1C 20 9E FA	26 DUMP	JSR	PRAI:BYTE	GO PRINT NEXT BYTE AND A SPACE
FB1E 80 EB	27	BNE	DUMP2	ALWAYS (ACC JUST PULLED AS #A0)
FB01	28 *			
FB01 45 7A	29 DUMPASC	LDA	A4L	RESET TO BEGINING OF LINE
FB03 25 74	30	STA	A1L	
FB05 45 7B	31	LDA	A4H	
FB07 85 75	32	STA	A1H	
FB09 20 73 FF	33	JSR	PRSPC	PRINT AN EXTRA SPACE
FB0C 40 00	34 ASCII	LDX	#0	TO INDEX MEMORY INDIRECT
FB0E 81 71	35	LDA	VAL:BY	
FB0E 09 80	36	ORA	#80	SET NORMAL VIDEO
FB32 09 A0	37	CMP	#\$A0	TEST FOR CONTROL CHARACTERS
FB34 80 02	38	BCS	ASC2	OK TO PRINT NON CONTROLS
FB35 A9 AC	39	LDA	#\$AE	OTHERWISE PRINT A SPACE
FB36 20 05 FF	40 ASCII	JSR	CRIT	PUT IT OUT
FB37 21 30 FF	41	JSR	DX:CA4	BRNCH BOTH A1 AND A4
FB38 80 06	42	BCS	ASC3	FINISHED
FB3A 45 74	43	LDA	A1L	TEST END OF LINE
FB3C 25 59	44	AND	MASK	
FB3E 20 05	45	BNE	ASCII	NOT DONE. PRINT NEXT
FB3E 40 00	46 ASCII	JMP	CROUT	
FB49	47 *			
FB49	48 *			
FB49	49 *			
FB49 38	50 COL80	SEC		INDICATE 80 COLUMNS DESIRED
FB4A AD 53 CO	51	LDA	#\$053	GOTO 80 COLUMN MODE
FB4B 80 04	52	BCS	SET80	BRANCH ALWAYS
FB4F	53 *			
FB4F 18	54 COL40	CLC		INDICATE 40 COLUMNS DESIRED
FB50 AD 52 CO	55	LDA	#\$052	GOTO 40 COLUMN MODE
FB53 A5 68	56 SET80	LDA	MODES	
FB55 09 40	57	ORA	#\$40	ASSUME 80
FB57 80 02	58	BCS	SET80A	AND BRANCH IF IT IS
FB59 29 BF	59	AND	#\$BF	BUT FIX FOR 40 IF NOT
FB5B 85 68	60 SET80A	STA	MODES	
FB5D 09 7F	61	ORA	#\$7F	ISOLATE BIT 7
FB5F 29 A0	62	AND	#\$A0	(BIT 7 SETS NORMAL/INVERSE)
FB61 85 66	63	STA	FORGND	
FB63 80 02	64	BCS	SET80B	AGAIN ASSUMES 80 COLUMNS
FB65 A9 F0	65	LDA	#\$F0	IF NOT, SET FOR/BACKGROUND COLOR
FB67 85 67	66 SET80B	STA	BKGND	
FB69	67 *			
FB69 A5 58	68 CLSCRN	LDA	LMARGIN	SET CURSOR TO TOP LEFT OF WINDOW
FB6B 80 5C	69	STA	CH	
FB6D A5 5A	70	LDA	WINTOP	
FB6F 85 5D	71	STA	CV	NOW DROP INTO CLEAR END OF PAGE
FB71	72 *			
FB71 A5 5C	73 CLEOP	LDA	CH	SAVE CURRENT CURSOR POSITION
FB73 48	74	PHA		
FB74 A5 5D	75	LDA	CV	
FB75 48	76	PHA		
FB77 20 D1 FB	77	JSR	SETCV	
FB7A 20 8E FB	78 CLEOP1	JSR	CLEOL	CLEAR TO END OF FIRST LINE
FB7D A5 58	79	LDA	LMARGIN	
FB7F 85 5C	80	STA	CH	
FB81 20 C9 FB	81	JSR	CURDOWN	GOTO NEXT LINE

FBB4 90 F4	82	BCC	CLEOP1		
FBB6 68	83	PLA			
FBB7 A8	84	TAY			
FBB8 68	85	PLA		.RESTORE CURSOR POSITION	
FBB9 85 5C	86	STA	CH		
FBBB 98	87	TYA		.GET OLD CV IN ACC AGAIN	
FBBC B0 23	88	BCS	SETCV	.BRANCH ALWAYS	
FBBE	89 *				
FBBE A5 5C	90	CLEOL	LDA	CH	.CLEAR TO END OF LINE FIRST
FBD0 4C B9 FC	91		JMP	CLEOL1	
FBD3	92 *				
FBD3 C9 80	93	CONTROL	CMP	##80	
FBD5 90 65	94	BCC	DISPLAYX		.IF INVERSE
FBD7 C9 8D	95	TSTCR	CMP	##8D	.IF CARRAGE RETURN THEN NEW LINE
FBD9 D0 3A	96	BNE	TSTBACK		
FBD9 20 8E FB	97	CARRAGE	JSR	CLEOL	.FIRST CLEAR TO THE END OF THIS LINE
FBD9 20 C3 FB	98	JSR	SETCHZ		.RESET CURSOR AND GOTO NEXT LINE (CARRY
FBA1 4C D2 FC	99	JMP	NXTLIN		.THEN GOTO THE NEXT LINE. IS SET)
FBA4	100 *				
FBA4	101 *				
FBA4 A5 5D	102	CURUP	LDA	CV	.TEST FOR TOP OF SCREEN
FBA6 C6 5D	103	DEC	CV		.ANTICIPATE 'NOT' TOP
FBA8 C5 5A	104	CMP	WINTOP		
FBA8 D0 02	105	BNE	CURUP1		.IT'S NOT TOP, CONTINUE
FBA8 A5 5B	106	LDA	WINBTM		.WRAP AROUND TO BOTTOM
FBAE 38	107	CURUP1	SEC		.DECREMENT BY ONE
FDAF E9 01	108	SBT	##		
FDB1 85 5D	109	SETCV	STA	CH	.SAVE NEW VERTICAL LINE
FDB3	110	BASCALC	BPL	BASCALC	
FDB3	111	CURDNI	EQU	*	
FDB3 A5 5D	112	LDA	CV		.GET VALUES FOR FIRST PAGE (\$400)
FDB5 10 4E	113	BPL	BASCALC		.ALWAYS
FDB7	114 *				
FDB7 24 68	115	CURLEFT	BIT	MODE5	.TEST FOR 80 OR 40
FDB9 70 02	116	BMI	LEFT80		
FDBB E6 5C	117	BPL	CH		
FDBD E6 5C	118	RIGHT	BIT	CH	.DUMP CURSOR HORIZONTAL
FDBF A5 5C	119	BPL	CH		.TEST FOR NEW LINE
FBC1 C5 5A	120	JSR	RMARGIN		
FBC3 A5 5B	121	SETCHZ	LDA	LMARGIN	.JUST IN CASE WE HAVE
FBC5 90 5D	122	BIT	CURLEFT		
FBC7 85 5C	123	SETCVH	STA	CH	.CURSOR AT START OF NEXT LINE
FBC9	124	*DEP INTO	FROM	FOR WRAP AROUND	
FBC9	125 *				
FBC9 E6 5D	126	CURDOWN	IN	CV	.MOVE CURSOR DOWN ONE LINE
FBCB A5 5D	127	LDA	CV		.ANTICIPATE NOT BOTTOM
FBCD C5 5B	128	CMP	WINBTM		.TEST FOR BOTTOM
FBCF 80 82	129	BCC	CURDNI		
FBD1 A5 5A	130	LDA	WINTOP		
FBD3 20 0C	131	BCS	SETCV		.BRANCH ALWAYS
FBD5	132 *				
FBD5 C9 88	133	BACKSP	CMP	##88	.BACKSPACE?
FBD7 D0 30	134	BNE	TSTBELL		
FBD9 24 58	135	CURLEFT	BIT	MODE5	.TEST FOR FORTY OR EIGHTY MODE
FDBB 70 02	136	BVS	LEFT80		
FDBD C6 5C	137	DEC	CH		
FDBF C6 5C	138	LEFT80	DEC	CH	
FDE1 30 06	139	BMI	LEFTUP		
FDE3 A5 5	140	LDA	CH		.TEST FOR WRAP AROUND
FDE5 C5 5B	141	CMP	LMARGIN		
FDE7 10 3B	142	BPL	CTRLRET		
FDE9 20 A4 FB	143	LEFTUP	JSR	CURUP	
FDEB A5 39	144	LDA	RMARGIN		
FDEF 25 5C	145	STA	CH		.SAVE NEW CURSOR POSITION
FDF1 D0 87	146	BNE	CURLEFT		.BRANCH ALWAYS
FDF3	147 *				
FDF3 C9 A0	148	COUT2	CMP	##A0	.IS IT CONTROL CHARACTER
FDF5 90 9D	149	BCC	CONTROL		
FDF7 24 83	150	BIT	MODE5		.TEST FOR INVERSE
FDF9 20 02	151	BMI	DISPLAYX		.AND PUT IT OUT

4,383,296

95

96

FC0A 20 19	152	AND	##7F	STRIP HI BIT
FC0C 20 1D FC	153	DISPLAYX	USR	DISPLAY
FC0E	154	*		
FC0F 20 B7 FB	155	INCHOR	USR	CURRIGHT
FC10 20 57	156	BYFLIN	BND	SCROLL
FC11	157	RTS		RESET CH UNL
FC12	158	*		
FC13	159	BASCALC1	PHA	CALC BASE ADR IN BAS4LH
FC14	160		PHA	
FC15	161	LSR	A	FOR GIVEN LINE NO.
FC16	162	AND	##03	LOC=LINE NO. C=#17
FC17 20 54	163	ORA	##04	ARG=000ABCDE, GENERATE
FC0C 85 5F	164	STA	BAS4H	BAS4H=000001CD
FC0E 49 0C	165	EOR	##C	
FC11 85 51	166	STA	BAS8H	
FC12 68	167	PLA		AND
FC13 29 18	168	AND	##18	BAS4L=EABAB0C0
FC15 20 02	169	BCC	BSCLC2	
FC17 29 7F	170	ADC	##7F	
FC19 85 5E	171	BSCLC2	STA	BAS4L
FC1B 0A	172	ASL	A	
FC1C 0A	173	ASL	A	
FC1D 00 5E	174	ORA	BAS4L	
FC1F 85 5E	175	STA	BAS4L	
FC21 85 60	176	STA	BAS8L	SAME FOR PAGE 2
FC23 29	177	PLP		
FC24 00	178	CTRLRET	RTS	
FC25	179	*		
FC26 40	180	OUTH	PHA	SAVE CHARACTER
FC28 24 6D	181	STY	TEMPY	
FC2B 26 6C	182	STX	TEMPX	
FC2A 20 33 FC	183	USR	COUT1	
FC2E A4 6D	184	LDY	TEMPY	
FC2F A4 6C	185	LDX	TEMPX	
FC31	186	PLA		
FC32 60	187	RTS		
FC33	188	*		
FC33 60 6E 00	189	COUT1	JMP	(CSWL) NORMALLY COUT1
FC34	190	*		
FC36 09 87	191	TSFRELL	CMP	##87
FC3B 00 12	192	BNE	LNFD	NO TEST FOR FORM FEED
FC3A	193	*		
FC3A A0 10	194	BELL	LDX	##10
FC3C 8A	195	TXA		
FC3D A8	196	BELL1	TAY	
FC3E 00 18 00	197	BELL2	BIT	##FD8
FC41 00 18	198	BELL2	BCQ	BELL2
FC43 00 08 FF	199	BELL3	BIT	##FD8
FC46 00 FR	200	BNE	BELL0	
FC48 60	201	DEY		
FC4F 00 23	202	BNE	BL 12	
FC4E 20 20 00	203	BIT	#000	
FC4E 60	204	PLP		
FC4F D0 EC	205	BNE	BELL1	
FC51 50	206	RTS		
FC52	207	*		
FC53 00 10	208	LNFD	##5A	LINE FEED?
FC54	209	CTRLRET	CTRLRET	
FC55 00 10	210	USR	CURDOWN	MOVE CURSOR DOWN A LINE
FC57 20 10	211	BCC	CTRLRET	BRANCH IF NO SCROLL NECESSARY
FC5E	212	*		
FC5B A0 5A	213	SCROLL	LDA	WINTOP
FC5E 41	214		PHA	SAVE IT FOR NOW
FC5E 00 10 10	215		USR	SETCV
FC5E 00 10 10	216	CTRL1	LDX	#3
FC5E 00 10 10	217	SCRL2	LDA	BAS4L, X
FC5E 00 10 10	218		STA	TBAS4L, X
FC5E 00 10 10	219		DEX	(TEMPORARY BASE ADDR.)
FC5E 00 10 10	220		BPL	SCRL2
FC5E 00 10 10	221		PLA	GET DESTINATION LINE

FC6B: 10	222	CLC		
FC6C: 09 01	223	ADC #1		CALCULATE SOURCE LINE
FC6E: C9 58	224	CMF WINDTM		IS IT THE LAST LINE?
FC70: D0 15	225	BCS LASTLN		YES: CLEAR IT
FC72: 48	226	PHA		SAVE AS NEXT DESTINATION LINE
FC73: 20 B1 FB	227	JSR SETCV		GET BASE ADDR FOR SOURCE LINE
FC76: A5 59	228	LDA RMARGIN		MOVE SOURCE TO DESTINATION
FC78: 4A	229	LSR A		DIVIDE BY 2
FC79: A8	230	TAY		
FC7A: 88	231	SCRL3 DEY		DONE YET?
FC7B: 30 E4	232	BMI SCRL1		YES: DO NEXT LINE
FC7D: B1 5E	233	LDA (BAS4L).Y		
FC7F: 91 62	234	STA (TBAS4L).Y		
FC81: B1 60	235	LDA (BAS8L).Y		MOVE BOTH PAGES
FC83: 91 64	236	STA (TBAS8L).Y		
FC84: 90 F3	237	BCC SCRL3		BRANCH ALWAYS
FC87: A5 58	238	LASTLN LDA RMARGIN		BLANK FILL THE LAST LINE
FC89: 4A	239	CLEOL1 LSR A		DIVIDE BY 2
FC8A: A8	240	TAY		
FC8B: B0 04	241	BCS CLEOL2		
FC8D: A5 66	242	LDA FORGND		(NORMALLY A SPACE)
FC8F: 91 5E	243	STA (BAS4L).Y		
FC91: A5 67	244	CLEOL2 LDA SPGND		(IF 80 COLUMNS) ALSO A SPACE
FC92: 91 60	245	STA (BAS8L).Y		
FC95: C8	246	INY		
FC96: 98	247	TYA		TEST FOR END OF LINE
FC97: 0A	248	ASL A		MULT BY 2 AGAIN
FC98: C5 59	249	CMF RMARGIN		
FC9A: 90 ED	250	BCS CLEOL1		CONTINUE IF MORE TO DO
FC9C: 60	251	RTS		ALL DONE.
FC9D	252 *			
FC9D: 24 68	253	DISPLAY BIT MODE'S		TEST FOR 40 OR 80
FC9F: 7C 0C	254	BCS DSPL80		STORE THE SINGLE CHARACTER AND RETURN
FCA1: 45 5C	255	LSR CH		INSURE PROPER 40 COLUMN DEL. LAY
FCA3: 06 5C	256	ASL CH		BY DROPPING BIT 0
FCA5: 20 AD FC	257	JSR DSPL80		DISPLAY IN \$400 PAGE.
FCA8: A5 67	258	LDA BKOND		ALSO SET BACKGROUND COLOR
FCAA: 91 60	259	DSPBKOND STA (BASEL).Y		
FCAE: 60	260	RTS		
FCAE: 48	261 *			
FCAE: A5 5C	262	DSPL80 PHA		PRESERVE CHARACTER
FCB0: 4A	263	LDA CH		DETERMINE WHICH PAGE
FCB1: A8	264	LSR A		
FCB2: 68	265	TAY		
FCB3: B0 F5	266	PLA		
FCB5: 91 5E	267	BCS DSPBKOND		BRANCH IF \$800 PAGE
FCB7: 60	268	STA (BAS4L).Y		
FCB8	269	RTS		
FCB8	270 *			
FCB8: B1 7E	271	NOTCR LDA (INBUF).Y		ECHO CHARACTER
FCBA: 20 25 FC	272	JSR COUT		
FCBD: C9 88	273	CMF ##88		BACKSPACE?
FCDF: F0 1D	274	BEG BKSPCE		
FCE1: C9 98	275	CMF ##98		CANCEL?
FCE3: F0 08	276	BEG CANCEL		
FCE5: E6 80	277	INC TEMP		
FCE7: A5 80	278	LDA TEMP		
FCE9: C9 90	279	CMF #INBUFLN		
FCEB: D0 17	280	BNE NXTCHAR		NO WRAP AROUND ALLOWED.
FCEB: A9 DC	281	CANCEL LDA ##DC		OUTPUT BACKSLASH
FCEE: 20 25 FC	282	JSR COUT		
FCD2: 20 EF FC	283	JSR CROUT		
FCD5:	284	GETLNZ EGU *		
FCD5: A5 68	285	GETLN LDA PROMPT		
FCD7: 20 25 FC	286	JSR COUT		
FCD9: A0 01	287	LDY #1		
FCD9: B4 80	288	STY TEMP		START AT BEGINNING OF INBUF
FCD9: A4 80	289	BKSPCE LDY TEMP		
FCE0: F0 F3	290	BEG GETLN		
FCE2: C6 80	291	DEC TEMP		BACK UP INPUT BUFFER
FCE4: 20 60 FD	292	NXTCHAR JSR RDCHAR		GET INPUT
FCE7: A4 80	293	LDY TEMP		

4,383,296

99

100

FCE9: 91 7E	294	STA	(INBUF).Y	
FCEB: C9 8D	295	CMP	##8D	
FCED: D0 C9	296	BNE	NOTCR	
FCEF:	297	CRDUT	EGU	*
FCEf: 2C 00 C0	298	BIT	KBD	; TEST FOR START/STOP
FCF2: 10 13	299	BPL	NOSTOP	
FCF4: 20 2E FD	300	JSR	KEYIN3	; READ KBD
FCF7: C9 A0	301	CMP	##A0	; IS IT A SPACE?
FCF9: F0 07	302	BEG	STOPLST	; YES, PAUSE TIL NEXT KEYPRESS.
FCFB: C9 8D	303	CMP	##8D	; QUIT THIS OPERATION?
FCFD: D0 08	304	BNE	NOSTOP	; NO, IGNORE THIS KEY.
FCFF: 4C 8B FA	305	JMP	ENDCR2	; YES, RESTART
FD02: AD 00 C0	306	STOPLST	LDA	KBD
FD05: 10 FB	307	BPL	STOPLST	
FD07: A9 8D	308	NOSTOP	LDA	##FD
FD09: 4C 25 FC	309	JMP	COUT	
FD0C:	310	*		
FD0E: 6C 70 00	311	RDKEY	JMP	(KSWL)
FD0F:	312	*		
FD0F: A9 7F	313	KEYIN	LDA	##7F
FD11: 85 63	314	STA	TBAS4H	
FD13: 20 88 FD	315	JSR	PICK	; GO READ SCREEN
FD16: 48	316	KEYIN1	PHA	; SAVE CHR AT CURSOR POSITION
FD17: 20 35 FD	317	JSR	KEYWAIT	; TEST FOR KEYPRESS
FD1A: B0 08	318	BCS	KEYIN2	; GO GET IT
FD1C: A5 69	319	LDA	CURSCR	; GIVE THEM AN UNDERSCORE FOR A TIME
FD1E: 20 9D FC	320	JSR	DISPLAY	
FD21: 20 35 FD	321	JSR	KEYWAIT	; GO SEE IF KEYPRESSED
FD24: 68	322	KEYIN2	PLA	
FD25: 08	323	PHP		; SAVE KEYPRESS STATUS
FD26: 48	324	PHA		
FD27: 20 9D FC	325	JSR	DISPLAY	
FD2A: 68	326	PLA		
FD2B: 28	327	PLP		
FD2C: 90 E8	328	RCC	KEYIN1	
FD2E: AD 00 C0	329	KEYIN3	LDA	KBD
FD31: 2C 10 C0	330	KEYIN4	BIT	KBDSTR2
FD34: 60	331	RTS		
FD35: E6 62	332	KEYWAIT	INC	TBAS4L
FD37: D0 09	333	BNE	KWAIT2	
FD39: E6 63	334	INC	TBAS4H	
FD3B: A9 7F	335	LDA	##7F	; TEST FOR DONE
FD3E: 18	336	CLC		
FD3E: 25 63	337	AND	TBAS4H	
FD40: F0 05	338	BEG	KEYRET	; RETURN IF TIMED OUT
FD42: 0E 00 C0	339	KWAIT2	ASL	KBD
FD45: 90 EE	340	DCC	KEYWAIT	
FD47: 60	341	KEYRET	RTS	
FD48:	342	*		
FD48:	343	*		
FD48:	344	ESC3	EGU	*
FD48: 20 77 FD	345	JSR	GOESC	
FD4B: A5 68	346	ESCAPE	LDA	MODES
FD4D: 29 B0	347	AND	##80	
FD4F: 49 AB	348	EOR	##AD	
FD51: 85 69	349	STA	CURSOR	
FD53: 20 0C FD	350	ESC1	JSR	RDKEY
FD56: A0 08	351	LDY	#8	; TEST FOR ESCAPE COMMAND
FD58: D9 F0 FF	352	ESC2	CMP	ESCTABL.Y
FD5B: F0 EB	353	BEG	ESC3	
FD5D: 88	354	DEY		
FD5E: 10 FB	355	BPL	ESC2	; LOOP TIL FOUND OR DONE
FD60:	356	*		
FD60: A9 80	357	RDCHAR	LDA	##80
FD62: 25 68	358	AND	MODES	
FD64: 85 69	359	STA	CURSOR	; SAVE STANDARD CURSOR.
FD66: 20 0C FD	360	JSR	RDKEY	
FD69: C9 9B	361	CMP	##9B	; ESCAPE CHARACTER?
FD6B: F0 DE	362	BEG	ESCAPE	
FD6D: C9 95	363	CMP	##95	; FORWARD COPY?
FD6F: D0 D6	364	BNE	KEYRET	
FD71: 20 88 FD	365	JSR	PICK	; GET CHARACTER FROM SCREEN

FC68: 10	222	CLC		
FC6C: 49 01	223	ADC	#1	CALCULATE SOURCE LINE
FC6E: C9 58	224	CMF	WINDTM	IS IT THE LAST LINE?
FC70: D0 15	225	BCC	LASTLN	YES, CLEAR IT
FC72: 48	226	PHA		SAVE AS NEXT DESTINATION LINE
FC73: 20 B1 FB	227	JSR	SETCV	GET BASE ADDR FOR SOURCE LINE
FC76: A5 59	228	LDA	RNMARGIN	MOVE SOURCE TO DESTINATION
FC78: 4A	229	LSR	A	DIVIDE BY 2
FC79: A8	230	TAY		
FC7A: 88	231	SCRL3	DEY	DONE YET?
FC7B: 30 E4	232	BMI	SCRL1	YES, DO NEXT LINE
FC7D: B1 5E	233	LDA	(BAS4L).Y	
FC7F: 91 62	234	STA	(TBAS4L).Y	
FC81: B1 60	235	LDA	(BAS8L).Y	MOVE BOTH PAGES
FC83: 91 64	236	STA	(TBAS8L).Y	
FC84: 90 F3	237	BCC	SCRL3	BRANCH ALWAYS
FC87: A5 58	238	LASTLN	LDA	RNMARGIN
FC89: 4A	239	CLEOL1	LSR	A
FC8A: A8	240	TAY		DIVIDE BY 2
FC8B: B0 04	241	BCC	CLEOL2	
FC8D: A5 66	242	LDA	FORGND	(NORMALLY A SPACE)
FC8F: 91 5E	243	STA	(BAS4L).Y	
FC91: A5 67	244	CLEOL2	LDA	BKOPD
FC93: 91 60	245	STA	(BAS8L).Y	(IF 80 COLUMNS, ALSO A SPACE)
FC95: C8	246	INY		
FC96: 98	247	IYA		TEST FOR END OF LINE
FC97: 0A	248	ASL	A	MULT BY 2 AGAIN
FC98: C9 59	249	CMF	RNMARGIN	
FC9A: 90 ED	250	BCC	CLEOL1	CONTINUE IF MORE TO DO
FC9C: 60	251	RTS		ALL DONE.
FC9D	252	*		
FC9D: 24 68	253	DISPLAY	BIT	MODE'S
FC9F: 7C 0C	254	BVS	DISPL60	TEST FOR 40 OR 80
FCA1: 46 5C	255	JSR	CH	STORE THE SINGLE CHARACTER AND RETURN
FCA3: 06 5C	256	ASL	CH	INSURE PROPER 40 COLUMN DEL. LAY
FCA5: 20 AD FC	257	JSR	DISPL60	BY DROPPING BIT 0
FCA8: A5 67	258	LDA	BKOND	DISPLAY IN \$400 PAGE.
FCAA: 91 60	259	DSPBKOND	STA	(BAS8L).Y
FCAE: 60	260	RTS		ALSO SET BACKGROUND COLOR
FCAE: 48	261	*		
FCAE: 48	262	DISPL60	PHA	PRESERVE CHARACTER
FCAE: A5 5C	263	LDA	CH	DETERMINE WHICH PAGE
FCB0: 4A	264	LSR	A	
FCB1: A8	265	TAY		
FCB2: 68	266	PLA		
FCB3: B0 F5	267	BCC	DSPBKOND	BRANCH IF \$800 PAGE.
FCB5: 91 5E	268	STA	(BAS4L).Y	
FCB7: 60	269	RTS		
FCB8	270	*		
FCB8: B1 7E	271	NOTCR	LDA	(INBUF).Y
FCBA: 20 25 FC	272	JSR	COUT	ECHO CHARACTER
FCBD: C9 88	273	CMF	#\$88	BACKSPACE?
FCBF: F0 1D	274	BEG	BKSPCE	
FCC1: C9 98	275	CMF	#\$98	CANCEL?
FCC3: F0 08	276	BEG	CANCEL	
FCC5: E6 80	277	INC	TEMP	
FCC7: A5 80	278	LDA	TEMP	
FCC9: C9 50	279	CMF	#INBUFLN	
FCCB: D0 17	280	BNE	NXTCHAR	NO WRAP AROUND ALLOWED.
FCCD: A9 DC	281	CANCEL	LDA	#\$DC
FCCF: 20 25 FC	282	JSR	COUT	OUTPUT BACKSLASH
FCD2: 20 EF FC	283	JSR	CROUT	
FCD5	284	GETLNZ	EGU	*
FCD5: A5 6B	285	GETLN	LDA	PROMPT
FCD7: 20 25 FC	286	JSR	COUT	
FCD9: A0 01	287	LDY	#1	
FCD9: 84 80	288	STY	TEMP	START AT BEGINNING OF INBUF
FCD9: A4 80	289	BKSPCE	LDY	TEMP
FCE0: F0 F3	290	BEG	GETLN	
FCE2: C6 80	291	DEC	TEMP	BACK UP INPUT BUFFER
FCE4: 20 60 FD	292	NXTCHAR	JSR	RDCHAR
FCE7: A4 80	293	LDY	TEMP	GET INPUT

FCE9: 91 7E	294	STA	(INBUF),Y	
FCEB: C9 8D	295	CMP	##8D	
FCED: D0 C9	296	BNE	NOTCR	
FCEF:	297	CROUT	EQU	*
FCEf: 2C 00 C0	298	BIT	KBD	; TEST FOR START/STOP
FCF2: 10 13	299	BPL	NOSTOP	
FCF4: 20 2E FD	300	JSR	KEYIN3	; READ KBD
FCF7: C9 A0	301	CMP	##A0	; IS IT A SPACE?
FCF9: F0 07	302	BEG	STOPLST	; YES, PAUSE TIL NEXT KEYPRESS.
FCFB: C9 8D	303	CMP	##8D	; QUIT THIS OPERATION?
FCFD: D0 08	304	BNF	NOSTOP	; NO, IGNORE THIS KEY.
FCFF: 4C 8B FA	305	JMP	ERR0R2	; YES, RESTART
FD02: AD 00 C0	306	STOPLST	LDA	KBD
FD05: 10 FB	307	BPL	STOPLST	
FD07: A9 8D	308	NOSTOP	LDA	##8D
FD09: 4C 25 FC	309	JMP	COOT	
FD0C:	310	*		
FD0C: 6C 70 00	311	RDKEY	JMP	(KSWL)
FD0F:	312	*		
FD0F: A9 7F	313	KEYIN	LDA	##7F ; MAKE SURE FIRST IS CURSOR
FD11: 85 63	314	STA	TBAS4H	
FD13: 20 88 FD	315	JSR	PICK	; GO READ SCREEN
FD16: 48	316	KEYIN1	PHA	; SAVE CHR AT CURSOR POSITION
FD17: 20 35 FD	317	JSR	KEYWAIT	; TEST FOR KEYPRESS
FD1A: B0 08	318	BCS	KEYIN2	; GO GET IT
FD1C: A5 69	319	LDA	CURSCR	; GIVE THEM AN UNDERSCORE FOR A TIME
FD1E: 20 9D FC	320	JSR	DISPLAY	
FD21: 20 35 FD	321	JSR	KEYWAIT	; GO SEE IF KEYPRESSED
FD24: 68	322	KEYIN2	PLA	
FD25: 08	323	PHP		; SAVE KEYPRESS STATUS
FD26: 48	324	PHA		
FD27: 20 9D FC	325	JSR	DISPLAY	
FD2A: 68	326	PLA		
FD2B: 28	327	PLP		
FD2C: 90 E8	328	BCC	KEYIN1	
FD2E: AD 00 C0	329	KEYIN3	LDA	KBD ; READ KEYBOARD
FD31: 2C 10 C0	330	KEYIN4	BIT	KBDSTR2 ; CLEAR KEYBOARD STROBE
FD34: 60	331	RTS		
FD35: E6 62	332	KEYWAIT	INC	TBAS4L ; JUST KEEP COUNTING
FD37: D0 09	333	BNF	KWAIT2	
FD39: E6 63	334	INT	TBAS4H	
FD3B: A9 7F	335	LDA	##7F	; TEST FOR DONE
FD3D: 18	336	CLC		
FD3E: 25 63	337	AND	TBAS4H	
FD40: F0 05	338	BEG	KEYRET	; RETURN IF TIMED OUT
FD42: 0E 00 C0	339	KWAIT2	ASL	KBD
FD45: 90 EE	340	BCC	KEYWAIT	
FD47: 60	341	KEYRET	RTS	
FD48:	342	*		
FD48:	343	*		
FD48:	344	ESC3	EQU	*
FD48: 20 77 FD	345	JSR	GOESC	
FD4B: A5 68	346	ESCAPE	LDA	MODES ; SET TO + SIGN FOR CURSOR MOVE
FD4D: 29 80	347		AND	##80
FD4F: 49 AB	348	EOR	##AD	
FD51: 85 69	349	STA	CURSOR	
FD53: 20 0C FD	350	ESC1	JSR	RDKEY ; READ NEXT CHARACTER
FD56: A0 08	351	LDY	##8	; TEST FOR ESCAPE COMMAND
FD58: D9 F0 FF	352	ESC2	CMP	ESCTABL,Y
FD5B: F0 EB	353	BEG	ESC3	
FD5D: 88	354	DEY		
FD5E: 10 FB	355	BPL	ESC2	; LOOP TIL FOUND OR DONE
FD60:	356	*		
FD60: A9 80	357	RDCHAR	LDA	##80 ; GO READ A CHARACTER
FD62: 25 68	358	AND	MODES	
FD64: 85 69	359	STA	CURSOR	; SAVE STANDARD CURSOR
FD66: 20 0C FD	360	JSR	RDKEY	
FD69: C9 9B	361	CMP	##9B	; ESCAPE CHARACTER?
FD6B: F0 DE	362	BEG	ESCAPE	
FD6D: C9 95	363	CMP	##95	; FORWARD COPY?
FD6F: D0 D6	364	BNE	KEYRET	
FD71: 20 88 FD	365	JSR	PICK	; GET CHARACTER FROM SCREEN

4,383,296

101

102

```

FD74 09 80      366      ORA  ##80      ;SET TO NORMAL ASCII
FD76: 60      367      RTS
FD77:      368 *
FD77: A9 FB    369 GOESC   LDA  #<CLSCRN
FD79: 48      370      PHA
FD7A B9 7F FD  371      LDA  ESCVECT, Y
FD7D 48      372      PHA
FD7E 60      373      RTS
FD7F:      374 *
FD7F 8D      375 ESCVECT DFB  CLEOL-1
FD80 70      376      DFB  CLEOP-1
FD81 60      377      DFB  CLSCRN-1
FD82 40      378      DFB  COL40-1
FD83 48      379      DFB  COL80-1
FD84 D8      380      DFB  CURLEFT-1
FD85 B6      381      DFB  CURRIGHT-1
FD86 CB      382      DFB  CURDOWN-1
FD87 A3      383      DFB  CURUP-1
FD88:      384 *
FD88 A5 5C    385 PICK    LDA  CH      ;GET A CHARACTER AT CURRENT CURSOR POSITION
FD8A 4A      386      LSR  A      ;DETERMINE WHICH PAGE
FD8B AB      387      TAY
FD9C 24 68    388      BIT  MODES   ;AND IF 80 COLUMN MODE
FD9E 50 05    389      BVC  PICK40  ;FORGET CARRY IF 40 COLUMNS
FD9F 90 00    390      BCC  PICK40  ;GET CHARACTER FROM #400 PAGE
FD92 B1 60    391      LDA  (BASBL), Y
FD94 60      392      RTS
FD95 B1 5E    393 PICK40  LDA  (BAS4L), Y
FD97 60      394      RTS
FD98:      395 *
FD98:      2 CLDSTRT EQU  *
FD98: A9 03    3      LDA  ##3
FD9A: 8D D0 FF  4      STA  $FFD0   ;ZERO PAGE IS ON 3!
FD9D:      5 SETCF  EQU  *
FD9D: D8      6      CLD      ;OF COURSE!
FD9E: A2 03    7      LDX  #3
FDA0: 86 7F    8      STX  INBUF+1
FDA2: BD BC FF  9  SETUP1  LDA  NMIRG, X
FDA5: 9D CA FF  10     STA  $FFCA, X
FDA8: BD B4 FF  11     LDA  HOOKS, X
FDAB: 95 6E    12     STA  CSWL, X
FDAD: BD B8 FF  13     LDA  VBOUNDS, X
FDB0: 95 58    14     STA  LMARGIN, X
FDB2: CA      15     DEX
FDB3: 10 ED    16     BPL  SETUP1
FDB5: 85 82    17     STA  IBDRVN
FDB7: A9 A0    18     LDA  #4A0   ;INPUT BUFFER AT $3A0
FDB9: 85 7E    19     STA  INBUF
FDBB: A9 60    20     LDA  ##60
FDBD: 85 81    21     STA  IBSLOT
FDBF: A9 FF    22     LDA  ##FF
FDC1: 85 68    23     STA  MODES
FDC3: 20 4F FB  24     JSR  COL40   ;SET 40 COLUMNS, CLEAR SCREEN
FDC6:      25 *
00A0:      27 ADR    EQU  $A0
00A0:      28 CPORTL EQU  ADR
00A1:      29 CPORTH EQU  ADR+1
00A2:      30 CTEMP  EQU  ADR+2
00A3:      31 CTEMP1 EQU  ADR+3
00A4:      32 YTEMP  EQU  ADR+4
00B4:      33 ROWTEMP EQU  ADR+20
C0DB:      34 CWRTON  EQU  #C0DB
C0DA:      35 CWRTOFF EQU  #C0DA
FFEC:      36 CTEMP  EQU  $FFEC
FFED:      37 CTEMP  EQU  $FFED
FDC6:      38 *
FDC6:      39 *
FDC6 A9 78    40 GENENTR LDA  ##78   ;INIT SCREEN INDX LOCATIONS
FDC8 B5 A0    41     STA  CPORTL
FDCA A9 0B    42     LDA  ##8B
FDCC B5 A1    43     STA  CPORTH
    
```

4,383,296

103

104

FDCE A9 F0	44	LDA	#240	SET UP INDEX TO CHRSET
FDD0 B5 A4	45	STA	YTEMP	
FDD2 A9 00	46	LDA	#0	
FDD4 A4	47	PHX		
FDD5 95 B4	48	LDI TEMP	ROWTEMP X	
FDD7 E2	49	INY		
FDD8 E0 20	50	DEX	##20	
FDDA D0 57	51	BNE	ZIPTEMPS	
FDDC A9 05	52	LDA	#5	FAKE THE FIRST BIT PATTERN
FDDF 10	53	PHX		PHANTOM 9TH BIT SHIFTS AS BIT 0
FDE0 08	54	PHX		
FDE1 4E	55	PHX		
FDE1 B6 A2	56	GENASC	CTEMP	GENERATE THE ASCII
FDE2 A7 07	57	CASCI1	LDY #7	CODES FOR THE FIRST PASS
FDE5 A6 4E	58	CASCI2	LDX CTEMP	
FDE7 0A	59	CASCI3	PHX	
FDE8 93 A7	60	LDA	(PORTL)	\$XXE=CHR 0 4
FDEA 96	61	DEX		\$XXE=CHR 1 5
FDEB 06	62	DEY		\$XXD=CHR 2 6
FDEC 06 00	63	BMI	CASCI4	\$XXC=CHR 3 7
FDED 00 00	64	BY	##5	\$XXB=CHR 0 4
FDEE 10 00	65	PL	CASCI3	\$XXE=CHR 1 5
FDEE 00 00	66	TRG	CASCI2	\$XXD=CHR 2 6
FDEA 00 00	67	TRG	WPORT	\$XXB=CHR 3 7
FDE7 00 00	68	BCS	CBYTES	DO DECODE CHARACTER TABLE
FDE7 19 0A	69	JMP	##A	SECOND SET OF 4
FDE8 00 00	70	BNE	CASCI1	
FDE8 00 00	71	PH	##20	
FDE8 00 00	72	BNE	GENASC	BRANCH AHEAD
FDE8 00 00	73	LDX	CBYTES	RESTORE BIT PATTERNS
FDE8 28	74	PLP		
FDE8 A2 12	75	LDX	##23	14 CHARACTERS (OF 6 ROWS)
FDE8 A2 05	76	LDY	##5	(FIVE COLUMNS)
FDE8 00 00	77	TRG	ROWTEMP+4	BREAK BYTE INTO
FDE8 00 00	78	TRG	A	5 BIT GROUPS
FDE8 00 00	79	DBE	WPORT	BRANCH IF NONE BITS IN THIS BYTE
FDE8 04 00	80	STY	CTEMP	
FDE8 00 00	81	DEC	YTEMP	(NOTE CARRY IS SET)
FDE8 00 00	82	BEG	DONE	BRANCH IF ALL DONE
FDE8 00 00	83	LDY	YTEMP	GET CHARACTER TABLE INDEX
FDE8 00 00	84	LDA	CHRSET-LDY	
FE17 2A	85	ROL	A	(CARRY KEEPS BYTE NON-ZERO UNTIL ALL 8
FE18 A4 A2	86	LDY	CTEMP	RESTORE COLUMN COUNT
FE1A 88	87	SHFTCNT	DEY	(GOT ALL FIVE BITS)
FE1B D0 6A	88	BNE	CSHIFT	NO, DO NEXT
FE1D 0A	89	DEX		(ALL ROWS DONE)
FE1E 10 E5	90	DPL	CCOLMS	NO, DO NEXT
FE20 08	91	PHX		SAVE REMAINING BIT PATTERN AND CARRY
FE21 48	92	PHA		
FE22 20 28 FE	93	JSR	STORCHRS	MOVE EM TO NON DISPLAYED VIDEO AREA
FE23 40 01 FE	94	JMP	CBYTES	
FE24	95	DONE	EQU *	
FE28 0A 1F	97	STORCHRS	LDX ##1F	MOVE CHARACTER PATTERNS TO VIDEO AREA
FE2A A0 00	98	STORSET	LDY #0	
FE2C 85 B4	99	STOROW	LDA ROWTEMP X	
FE2E 0A	100	APL	A	SHIFT TO CENTER
FE2F 0F 3E	101	AND	##3E	STRIP EXTRA GARBAGE
FE31 91 A0	102	STA	(CPORTL),Y	
FE33 0A	103	DEX		
FE34 08	104	INY		
FE35 00 09	105	CPY	##8	THIS GROUP DONE
FE37 00 03	106	BNE	STOROW	NO, NEXT ROW
FE39 20 99 FE	107	JSR	NXTPORT	
FE3C 09 08	108	CMP	##8	
FE3E 0A 04	109	BEG	GENDONE	ALL ROWS STORED
FE40 8A	110	TXA		
FE41 10 E7	111	DPL	STORSET	
FE43 60	112	PHS		(PARIAL SET (\$478-\$5FF))
FE44	113	*		

4,383,296

105

106

```

FE44:A9 01      114 GENDONE LDA #1          ;SET NORMAL MODE
FE46:B5 A2      115          STA CTEMP
FE48:A9 60      116 GEN1   LDA #0        ;PREPARE TO SEND BYTES TO CHARACTER
FE4A:20 DB C0   117          BIT CURTCN  ;GENERATOR RAM
FE4D:20 AE FF   118          JRP VRETRCE ;WAIT FOR NEXT VERTICAL RETRACE
FE50:A9 20      119          LDA #020    ;WAIT AGAIN
FE52:20 AE FE   120          JSR VRETRCE
FE55:20 DA C0   121          BIT CWRTOFF  ;CHARACTERS ARE NOW LOADED
FE58:20 88 FE   122          JSR ALTCHR  ;REPEAT THIS SET FOR OTHER 64 CHARACTERS
FE5B:06 A2      123          DBC CTEMP  ;HAVE WE DONE ALTERNATES YET?
FE5D:10 16      124          BPL GEN2    ;NO, DO IT!
FE5F:A9 08      125          LDA #17     ;DUMP ASCII VALUES FOR NEXT SET
FE61:85 A1      126          STA CPORTH
FE63:A0 07      127 NXTASCI LDY #7      ;THE USUAL COUNTDOWN
FE65:B1 A0      128 NXTASCI LDA (CPORTL),Y
FE67:18         129          DEC C
FE68:69 06      130          ADC #8
FE6A:01 00      131          STA (CPORTL),Y
FE6C:88         132          DEY
FE6D:10 16      133          BPL NXTASCI
FE6F:20 99 FF   134          JSR NXTPORT
FE72:90 0F      135          BCC NXTASCI
FE74:00         136          RTS
FE76:A0 00      137 GEN2   LDY #33     ;SETUP ALTERNATE WITH UNDERLINES
FE77:A9 7F      138          LDA #7F
FE79:99 FC 05   139 UNDER  STA $5FC,Y
FE7C:99 FC 07   140          STA $7FC,Y
FE7F:98         141          DEY
FE80:10 16      142          BPL UNDER
FE82:A0 00      143          LDA #8
FE84:01 A1      144          STA CPORTH
FE86:00 00      145          BNE GEN1
FE88:         146 *
FE8A:00 01      147 ALTCHR LDY #7      ;ADJUST ASCII FOR ALTERNATE SET
FE8C:B1 A0      148 ALTCHI LDA (CPORTL),Y
FE8E:49 20      149          EOR #20    ;$20-->0 $40-->$60
FE8E:91 A0      150          STA (CPORTL),Y
FE90:80         151          DEY
FE91:10 F7      152          BPL ALTCHI ;ADJUST THEM ALL
FE93:20 99 FE   153          JSR NXTPORT
FE96:90 F0      154          BCC ALTCHR
FE98:60         155          RTS
FE99:         156 *
FE99:A5 A0      157 NXTPORT LDA CPORTL  ;CONVERT $78->$FB OR $FB->$78
FE9B:49 B0      158          EOR #80
FE9D:85 A0      159          STA CPORTL
FE9F:30 02      160          BMI NOHIGH
FEA1:E6 A1      161          INC CPORTH ;IF =C THEN =4
FEA3:A5 A1      162 NOHIGH  LDA CPORTH
FEA5:09 0C      163          CMP #4C
FEA7:D0 04      164          BNE PORTDN
FEA9:A9 04      165          LDA #4
FEAB:85 A1      166          STA CPORTH
FEAD:60         167 PORTDN  RTS
FEAE:         168 *
FEAE:         169 *
FEAE:85 A3      170 VRETRCE STA CTEMP1  ;SAVE BITS TO BE STORED
FEB0:AD EC FF   171          LDA CB2CTRL ;CONTROL PORT FOR 'CB2'
FEB3:29 3F      172          AND #3F    ;RESET HI BITS TO 0
FEB5:05 A3      173          ORA CTEMP1
FEB7:8D EC FF   174          STA CB2CTRL
FEBA:A9 08      175          LDA #8     ;TEST VERTICAL RETRACE
FEBC:8D ED FF   176          STA CB2INT
FEBF:2C ED FF   177 VWAIT  BIT CB2INT  ;WAIT FOR RETRACE
FEC2:F0 FB      178          BEQ VWAIT
FEC4:60         179          RTS
FEC5:         180 *
FEC5:         181 CHRSET  EQU *
    
```

107	4,383,296	108
FEC5 F0 01 82 182	DFB \$F0, \$01, \$82, \$18	
FEC8: 18		
FEC9: 40 84 81 183	DFB \$40, \$84, \$81, \$2F	
FEC0: 2F		
FECD: 58 44 81 184	DFB \$58, \$44, \$81, \$29	
FED0: 29		
FED1: 02 1E 01 185	DFB \$02, \$1E, \$01, \$91	
FED4: 91		
FED5: 7C 1F 49 186	DFB \$7C, \$1F, \$49, \$30	
FED8: 30		
FED9: 8A 08 43 187	DFB \$8A, \$08, \$43, \$14	
FEDC: 14		
FEDD: 31 2A 22 188	DFB \$31, \$2A, \$22, \$13	
FEE0: 13		
FEE1: E3 F7 C4 189	DFB \$E3, \$F7, \$C4, \$91	
FEE4: 91		
FEE5: 48 A2 DA 190	DFB \$48, \$A2, \$DA, \$24	
FEE8: 24		
FEE9: C6 4A 62 191	DFB \$C6, \$4A, \$62, \$8C	
FEEC: 8C		
FEED: 24 C6 F8 192	DFB \$24, \$C6, \$F8, \$63	
FEF0: 63		
FEF1: 8C 01 46 193	DFB \$8C, \$01, \$46, \$17	
FEF4: 17		
FEF5: 52 8A AF 194	DFB \$52, \$8A, \$AF, \$16	
FEFB: 16		
FEF9: 14 E3 33 195	DFB \$14, \$E3, \$33, \$31	
FEFC: 31		
FEFD: C6 F8 DC 196	DFB \$C6, \$F8, \$DC, \$73	
FEFD: 73		
FF01: 3F 46 17 197	DFB \$3F, \$46, \$17, \$62	
FF04: 62		
FF05: 8C 21 E6 198	DFB \$8C, \$21, \$E6, \$18	
FF08: 18		
FF09: 6A 8D 61 199	DFB \$6A, \$8D, \$61, \$CF	
FF0C: 0F		
FF0D: 18 62 74 200	DFB \$18, \$62, \$74, \$D1	
FF10: D1		
FF11: B9 18 49 201	DFB \$B9, \$18, \$49, \$4C	
FF14: 4C		
FF15: 91 C0 F3 202	DFB \$91, \$C0, \$F3, \$09	
FF18: 09		
FF19: 2C 91 C0 203	DFB \$2C, \$91, \$C0, \$14	
FF1C: 14		
FF1D: 1D 8C EF 204	DFB \$1D, \$8C, \$EF, \$07	
FF20: 07		
FF21: 17 48 82 205	DFB \$17, \$48, \$82, \$31	
FF24: 31		
FF25: 84 1E DF 206	DFB \$84, \$1E, \$DF, \$0B	
FF28: 0B		
FF29: 31 84 F8 207	DFB \$31, \$84, \$F8, \$FE	
FF2C: FE		
FF2D: 72 3E 3E 208	DFB \$72, \$3E, \$3E, \$17	
FF30: 17		
FF31: 62 8C FD 209	DFB \$62, \$8C, \$FD, \$C7	
FF34: C7		
FF35: 50 E3 0B 210	DFB \$50, \$E3, \$0B, \$51	

				4,383,296					
				109					110
FF39	51								
FF39	05	EB	08	211	DFB	\$05, \$E8, \$08, \$73			
FF3C	7D								
FF3D	18	0C	42	212	DFB	\$18, \$0C, \$42, \$3E			
FF40	3E								
FF41	01	02	20	213	DFB	\$01, \$02, \$20, \$42			
FF44	43								
FF45	3E	40	10	214	DFB	\$3E, \$41, \$18, \$8C			
FF48	0C								
FF49	08	00	70	215	DFB	\$08, \$00, \$70, \$EE			
FF4C	EE								
FF4D	00	11	10	216	DFB	\$00, \$11, \$11, \$21			
FF50	21								
FF51	11	02	00	217	DFB	\$11, \$02, \$E0, \$3C			
FF54	00								
FF55	21	31	00	218	DFB	\$21, \$31, \$02, \$E0			
FF58	E0								
FF59	1C	00	00	219	DFB	\$1C, \$00, \$08, \$B9			
FF5C	89								
FF5D	80	62	10	220	DFB	\$80, \$62, \$14, \$1F			
FF60	1F								
FF61	46	A2	DE	221	DFB	\$46, \$A2, \$DE, \$43			
FF64	43								
FF65	2C	04	80	222	DFB	\$2C, \$04, \$88, \$BE			
FF68	BF								
FF69	FF	CE	70	223	DFB	\$FF, \$CE, \$7D, \$37			
FF6C	37								
FF6D	49	88	95	224	DFB	\$49, \$88, \$95, \$18			
FF70	12								
FF71	85	09	60	225	DFB	\$98, \$09, \$62, \$D1			
FF74	01								
FF75	44	E8	80	226	DFB	\$44, \$E8, \$88, \$FB			
FF78	FB								
FF79	02	90	40	227	DFB	\$02, \$90, \$40, \$00			
FF7C	00								
FF7D	10	E0	00	228	DFB	\$10, \$E0, \$03, \$02			
FF80	02								
FF81	00	40	00	229	DFB	\$00, \$40, \$00, \$00			
FF84	00								
FF85	08	00	00	230	DFB	\$08, \$00, \$00, \$2B			
FF88	24								
FF89	10	42	40	231	DFB	\$10, \$42, \$44, \$25			
FF9C	25								
FF8D	82	88	2F	232	DFB	\$82, \$88, \$2F, \$48			
FF90	48								
FF91	25	44	10	233	DFB	\$25, \$44, \$10, \$82			
FF94	82								
FF95	02	00	2F	234	DFB	\$02, \$00, \$2F, \$5A			
FF98	5A								
FF99	40	45	02	235	DFB	\$40, \$45, \$02, \$8E			
FF9C	8E								
FF9D	64	50	90	236	DFB	\$64, \$50, \$90, \$01			
FFA0	01								
FFA1	3E	26	42	237	DFB	\$3E, \$26, \$42, \$80			
FFA4	80								
FFA5	21	80	00	238	DFB	\$21, \$80, \$00, \$05			
FFA8	05								
FFA9	00	FB	80	239	DFB	\$00, \$FB, \$80, \$00			
FFAC	00								
FFAD	05	08	FB	240	DFB	\$05, \$08, \$FB, \$80			
FFB0	80								
FFB1	28	05	88	241	DFB	\$28, \$05, \$88			
FFB4				242 *					
FFB4				243 HOOKS	EGU	*			
FFB4	F2	FB		244	DW	COU2			
FFB6	0F	FD		245	DW	KEYIN			
FFB8				246 *					

4,383,296

111

112

FFB8	247	VBOUNDS	EGU	*	
FFB8 00 50 00	248		DFB	\$0, \$50, 0, \$18	
FFB8 1B					
FFBC	249	*			
FFBC 4C 89 F8	250	NMIHQ	JMP	RECON	IN DIAGNOSTICS
FFBF 40	251		RTI		
FFC0: C3 CF D6	252		ASC	'COPYRIGHT JANUARY, 1980	APPLE COMPUTER INC. 'URH'
FFC3 D9 D2 C9					
FFC6 C7 C8 D4					
FFC9 A0 CA C1					
FFCC CE D5 C1					
FFCF D2 D9 AC					
FFD2: A0 B1 B9					
FFD5: B8 B0 A0					
FFD8: A0 C1 D0					
FFDB: D0 C0 C5					
FFDE: A0 C3 CF					
FFE1: CD D0 D5					
FFE4: D4 C5 D2					
FFE7: A0 C9 CE					
FFEA: C3 AE AE					
FFED: CA D2 C8					
FFF0:	253		CHN	MONVECT	
FFF0:	1	*			
FFF0: CC	2	ESCTABL	DFB	\$CC	
FFF1: D0	3		DFB	\$D0	
FFF2: D3	4		DFB	\$D3	
FFF3: B4	5		DFB	\$B4	
FFF4: B8	6		DFB	\$B8	
FFF5: B8	7		DFB	\$B8	
FFF6: 95	8		DFB	\$95	
FFF7: 8A	9		DFB	\$8A	
FFFB: B8	10		DFB	\$B8	
FFFY: 00	11		DFB	\$00	NOTHING
FFFA:	12	*			
FFFA: CA FF	13	NMI	DW	\$FFCA	
FFFC: EE F4	14	RESET	DW	DIAGN	FIRST DIAGNOSTICS
FFFE: CD FF	15	IRQ	DW	\$FFCD	
0000	16	*			

*J. Richard (Dick) Huston
(also worked on Apple III SOS)*

*** SUCCESSFUL ASSEMBLY: NO ERRORS

75 A1H	74 A1L	F9D4 A1PC	F9D7 A1PC1
77 A2H	76 A2L	79 A3H	78 A3L
7B A4H	7A A4L	A0 ADR	FEBA ALTC1
FE8B ALTCHR	FB2C ASC1	FB38 ASC2	FB46 ASC3
FA06 ASCDONE	FA09 ASCI10	F9DF ASCI11	F9E1 ASCI12
FA07 ASCII	F9F2 ASCI13	5F BAS4H	5E BAS4L
61 BAS8H	60 BAS8L	FC05 BASCALC1	?FBB3 BASCALC
FC3D BELL1	FC3E BELL2	FC43 BELL3	FC3A BELL
FA15 BITOFF	FA11 BITON	67 BKOND	FCDE BKSPCE
FAA0 BL1	F479 BLOCKIO	FC19 BSCLC2	FCCD CANCEL
?FB9B CARRAGE	FFEC CB2CTRL	FFED CB2INT	FE01 CBYTES
FE05 CCOLMS	5C CH	FEC5 CHRSET	?FA0A CKMDE
?FD9B CLDSTRT	FC89 CLEOL1	FB8E CLEOL	FC91 CLEOL2
FB71 CLEOP	FB7A CLEOP1	FB69 CLSCRN	F91C CMDSRCH
F96C CMDTAB	F97C CMDVEC	FB4F COL40	FB49 COL80
FB93 CONTROL	FC33 COUT1	FBF2 COUT2	FC25 COUT
A1 CPORTH	A0 CPORTL	F9FB CRCHK	FA26 CRMON
FCEF CROUT	FE07 CSHFT	? 6F CSWH	6E CSWL
A3 CTEMP1	A2 CTEMP	FC24 CTRLRET	FBB3 CURDN1
FBC9 CURDOWN	FBF7 CURIGHT	FBD9 CURLEFT	69 CURSOR

4,383,296

113

FBAE	CURUP1
CODB	CWRTON
F96D	DIGRET
FCAA	DSPBKGND
FB09	DUMP1
FB21	DUMPASC
FABE	ERROR
FD48	ESC3
66	FORGND
FDF4	GASC14
FE44	GENDONE
F92C	GETNUM
85	IDBUF
50	INBUFLEN
FA7D	JUMP
FD24	KEYIN2
FD47	KEYRET
FD42	KWAIT2
58	LMARGIN
68	MODES
FA31	MOVNXT
FD07	NOSTOP
F98C	NXTA4
F947	NXTBIT
F915	NXTINP
? 73	PCH
FEAD	PORTDN
FA70	PRBYTSP
?F9B5	PRHEX
FD60	RDCHAR
FA19	REPEAT
F900	RET2
59	RMARGIN
?FAC5	SAVCMD
FC7A	SCRL3
FB5B	SETBOA
FBB1	SETCV
?FD9D	SETUP
6A	STACK
FE28	STORCHRS
F9D1	SVMASK
64	TBASBL
F95E	TOSUB
FC36	TSTBELL
Q3FB	USERADR
FA4C	VRFY2
5B	WINBTM
A4	YTEMP
50	INBUFLEN
5A	WINTOP
5E	BAS4L
62	TBAS4L
66	FORGND
69	CURSOR
6D	TEMPY
? 71	KSWH
75	A1H
79	A3H
7D	YSAV
82	IDDRVN
A0	ADR
A4	YTEMP
C010	KBDSTRB
F4EE	DIAGN
F900	RET2

114

5D	CV
F4EE	DIAGN
FC9D	DISPLAY
FAB7	DUMMY
FB1C	DUMP3
?F901	ENTRY
?FD53	ESC1
FFF0	ESCTABL
FDE5	GASC12
FE75	GEN2
FCD5	GETLN
FA7D	GO
82	IDDRVN
?FBFF	INCHORZ
C000	KBD
?FD31	KEYIN4
? 71	KSWH
FBDF	LEFT80
69	MASK
F908	MONZ
?FFFA	NMI
FADF	NOVER
FE63	NXTASCI
FCE4	NXTCHAR
FE99	NXTPORT
FD95	PICK40
F9C2	PRBYCOL
F9BF	PRHEX2
6D	PRDPT
FAC0	READ
?FFFC	RESET
F9AD	RETA1
FAB3	RWERROR
FC61	SCRL1
FC5B	SCROLL
FB67	SETBOB
FADD	SETMDZ
FE1A	SHTCNT
FD02	STOPLST
FE2A	STORSET
62	TBAS4L
80	TEMP
F99B	TSTA1
FAF6	TSTDUMP
FFB8	VBDUNDS
FA40	VRFY1
FAC3	WRTE
F967	ZETATE
5B	LMARGIN
5C	CH
60	BASBL
64	TBASBL
68	MODES
6B	PROMPT
? 6F	CSWH
? 73	PCH
77	A2H
7B	A4H
80	TEMP
87	IBCMD
A2	CTEMP
Q3FB	USERADR
CODB	CWRTON
F7FF	RET1
F904	MON

CODA	CWRTOFF
F941	DIGIT
FE28	DONE
FAFC	DUMPO
FAE9	DUMPB
FAB8	ERROR2
FD58	ESC2
FD7F	ESCVECT
FDE7	GASC13
FDE1	GENASC
FCD5	GETLNZ
FFB4	HOOKS
81	IBSLOT
?FFFE	IRQ
FD16	KEYIN1
FDOF	KEYIN
70	KSWL
FBE9	LEFTUP
FA52	MISMATCH
FA2C	MOVE
FEA3	NOHIGH
F992	NXTA1
F94F	NXTBAS
F932	NXTCHR
F9DE	OLDPC
FD88	PICK
F9AC	PRBYTE
F9B7	PRHEXZ
FA73	PRSPC
F689	RECON
F7FF	RET1
FDBD	RIGHT1
FAC7	RWLOOP
FC63	SCRL2
FA9A	SEP
FBC3	SETCHZ
FAB8	SETMODE
FAA4	SPCE
FAAF	STOR1
?FAAD	STOR
? 65	TBAS8H
6D	TEMPY
FBD5	TSTBACK
FE79	UNDER
FEAE	VRTRCE
FEBF	VWAIT
7D	YSAV
59	RMARGIN
5D	CV
61	BAS8H
? 65	TBAS8H
69	MASK
6C	TEMPX
70	KSWL
74	A1L
78	A3L
7C	STATE
81	IBSLOT
A0	CPORTL
A3	CTEMP1
C000	KBD
F479	BLOCKID
F882	RET3
F908	MONZ

4,383,296

115

116

F912 SCAN	F915 NXTINP	F91C CMDSRCH	F92C GETNUM
F932 NXTCHR	F941 DIGIT	F947 NXTBIT	F94F NXTBAS
F959 NXTBS2	F95E TOSUB	F967 ZSTATE	F95B DIGRET
F96C CMDTAB	F97C CMDVEC	F98C NXTA4	F992 NXTA1
F99B TSTA1	F9AB RETA1	F9AC PRBYTE	?F9B5 PRHEX
F9B7 PRHEX2	F9BF PRHEX2	F9C2 PRBYCOL	?F9C5 PROOLON
F9C9 TSTBOWID	F9D1 SVMASK	F9D4 A1PC	F9D7 A1PC1
F9DE OLDPC	F9DF ASCII1	F9E1 ASCII2	F9F2 ASCII3
F9FB CRCHK	FA06 ASCDDNE	FA07 ASCII	FA09 ASCII0
?FA0A CKMDE	FA11 BITON	FA15 BITOFF	FA19 REPEAT
FA21 REPEAT1	FA26 CRMON	FA2C MOVE	FA31 MOVNXT
FA3B VRFY	FA40 VRFY1	FA4C VRFY2	FA52 MISMATCH
FA61 PRINTA1	FA6E PRA1BYTE	FA70 PRBYTSP	FA73 PRSPC
FA7B USER	FA7B JUMP	FA7D GO	FAB3 RWERROR
FAB8 ERROR2	FABE ERROR	FA91 DEST	FA9A SEP
FAA0 DL1	FAA4 SPCE	?FAAB STOR	FAAF STOR1
FAB7 DUMMY	FAB8 SETMODE	FABD SETMDZ	FAC0 READ
FAC3 WRTE	?FAC5 SAVCMD	FAC7 RWLOOP	FADF NOVER
FAE9 DUMPS	FAF6 TSTDUMP	FAF7 ERROR1	?FAF9 DUMP
FAFC DUMPO	FB09 DUMP1	FB0C DUMP2	FB1C DUMPS
FB21 DUMPASC	FB2C ASC1	FB3B ASC2	FB46 ASC2
FB49 COL80	FB4F COL40	FB53 SETB0	FB5B SETB0A
FB67 SETB0B	FB69 CLSCRN	FB71 CLEDP	FB7A CLEDP1
FB8E CLEDL	FB93 CONTROL	?FB97 TSTCR	?FB9B CARRAGE
FBA4 CURUP	FBAE CURUP1	FBB1 SETCV	FBB3 CURDN1
?FBB3 BASCALC	FBB7 CURIGHT	FBD0 RIGHT1	FBC3 SETCHE
?FBC7 SETCVH	FBC9 CURDOWN	FBD5 TSTBACK	FBD9 CURLEFT
FBD5 LEFTB0	FBE9 LEFTUP	FBF2 COUT2	FBFC DISPLAYX
?FBEF INCHORZ	FC02 NXTLIN	FC05 BASCALC1	FC19 BSCLC2
FC24 CTRLRET	FC25 COUT	FC33 COUT1	FC36 TSTBELL
FC3A BELL	FC3D BELL1	FC3E BELL2	FC4B BELLS
FC52 LNFD	FC5B SCROLL	FC61 SCRL1	FC6B SCRL2
FC7A SCRL3	FC87 LASTLN	FC89 CLEDL1	FC91 CLEDL2
FC9D DISPLAY	FCAA DSPBKOND	FCAD DSPL80	FCBB NOTOP
FCDD CANCEL	FCD5 GETLN	FCD5 GETLNZ	FCDE BKSPCE
FCF4 NXTCHAR	CFEF CROUT	FD02 STOPLST	FD07 NOSTOP
FD0C RDKEY	FD0F KEYIN	FD16 KEYIN1	FD24 KEYIN2
FD2E KEYIN3	?FD31 KEYIN4	FD35 KEYWAIT	FD42 KWAIT2
FD47 KEYRET	FD48 ESC3	FD4B ESCAPE	?FD53 ESC1
FD5B ESC2	FD60 RDCHAR	FD77 GDESC	FD7F ESCVESC
FD8B PICK	FD95 PICK40	?FD98 CLDSTR	?FD9D SETUP
FDA2 SETUP1	?FDC6 GENENTR	FDD5 ZIPTEMPS	FDE1 GENASC
FDE3 GASCII	FDE5 GASC12	FDE7 GASC13	FDFA GASC14
FE01 CBYTES	FE05 CCOLMS	FE07 CSHFT	FE1A SHFTCNT
FE2B DONE	FE2B STORCHRS	FE2A STORSET	FE2C STORCW
FE44 GENDONE	FE4B GEN1	FE43 NXTASCI	FE65 NXTASC2
FE75 GEN2	FE79 UNDER	FE8B ALTCHR	FE8A ALTCL
FE99 NXTPORT	FEA3 NDRHIGH	FEAD PORTDN	FEAE VRETRCE
FEBF VWAIT	FEC5 CHRSET	FFB4 HOOKS	FFBB VBOUNDS
FFBC NMIRG	FFEC CB2CTRL	FFED CB2INT	FFFO ESCTABL
?FFFA NMI	?FFFC RESET	?FFFE IRQ	

I claim:

1. In a digital computer which includes a central processing unit (CPU), a random-access memory (RAM), an address bus interconnecting said CPU and RAM such that said CPU addresses locations in said RAM and a data bus interconnecting said CPU and RAM, said CPU for certain functions addressing predetermined locations in said RAM with a predetermined range of address signals, an improvement comprising:

detection means for detecting said predetermined range of address signals, coupled to said address bus;

register means for storing digital signals, coupled to said data bus, and;

switching means for coupling said digital signals stored in said register means to said address bus when said detection means detects said predetermined range of said address signals;

whereby data for said certain functions normally stored by said CPU in said predetermined locations may be stored elsewhere in said RAM, thereby enhancing the performance of said computer.

2. The improvement defined by claim 2 wherein said detection means detects all binary zeros.

3. The improvement defined by claim 1 wherein said switching means comprises a multiplexer controlled by said detection means for selecting said register means.

4. The improvement defined by claim 1 including a read-only memory coupled to said address bus and said data bus.

5. The improvement defined by claim 4 wherein said stored signals in said register means provide a pointer for locations in said RAM during a direct memory access transfer.

6. The improvement defined by claim 5 wherein said read-only memory in response to signals on said address bus provides instructions to said CPU causing it to increment address signals during said direct memory access transfer.

7. In a digital computer which includes a central processing unit (CPU), a random-access memory (RAM), an address bus having a first plurality and a second plurality of lines for coupling said CPU with said RAM, and a data bus interconnecting said CPU and RAM, said CPU for certain operations addressing predetermined locations in said RAM with address signals on said first plurality of lines by coupling a predetermined address on said second plurality of lines, an improvement comprising:

register means for storing signals, coupled to said data bus;

multiplexing means coupled to said second plurality of lines and said register means for selecting signals from one of said second plurality of lines and said register means;

logic means coupled to said second plurality of lines and said multiplexing means for causing said multiplexing means to select signals from said register means when said CPU couples said predetermined address on said second plurality of lines;

whereby said signals from said register means provide alternate locations in RAM for storage associated with said certain operations.

8. The improvement defined by claim 7 wherein said predetermined address is all binary zeros.

9. The improvement defined by claim 7 including a read-only memory coupled to said address bus and said data bus.

10. The improvement defined by claim 8 wherein said stored signal in said register means provides a pointer for locations in said RAM during a direct memory access transfer.

11. The improvement defined by claim 9 wherein said read-only memory in response to signals on said address bus provides instructions to said CPU causing it to increment address signals during said direct memory access transfer.

12. In a digital processor used in conjunction with a display, said processor including a data bus and an address bus, a memory comprising:

a first plurality of memory devices for storing data, coupled to receive data from said data bus;

a first memory output bus coupled to receive data from said first plurality of memory device;

a second plurality of memory devices for storing data coupled to receive data from said data bus;

a second memory output bus coupled to receive data from said second plurality of memory devices;

addressing means coupled to said address bus for providing address signal for addressing said first and second plurality of memory devices;

first switching means for selecting data from one of said first and second memory buses for coupling to said data bus, said first switching means coupled to said first and second memory bus and said data bus;

second switching means for selecting data from said first and second memory buses for coupling to said display, said second switching means coupled to said first and second memory buses and said display; and,

circuit means for coupling one of a selected said first and second memory buses to said addressing means such that data from said selected one of said buses provides addressing information for selecting subsequent locations in said memory devices when said data bus is receiving data from the other of said memory buses,

whereby said memory provides data for a high resolution display and whereby some data stored in said memory is used for remapping locations in said memory.

13. The memory defined by claim 12 wherein said circuit means comprises a multiplexer, said multiplexer selecting between said data from said selected one of said buses and bank switching signals coupled to said multiplexer.

14. The memory defined by claim 13 wherein said multiplexer is controlled by a logic circuit which is coupled to said address bus and said selected one of said buses.

15. The memory defined by claim 14 wherein said logic circuit causes said multiplexer to select said bank switching signals each time said processor switches an OP code.

16. In a digital computer with a memory, which is used in conjunction with a raster scanned display, said display including a digital counter which provides a vertical count representative of the horizontal line scanned by the beam for said display, said memory providing data for displaying rows of characters, an

addressing means coupled to said memory for scrolling displayed characters, comprising:

an adder having a first and a second input terminal, the output of said adder providing a portion of an address signal for said memory, said first terminal of said adder being coupled to receive the lesser significant bits of said vertical count;

said computer providing a periodically repeated sequence of digital numbers coupled to said second terminal of said adder, said sequence of digital numbers provided by said computer having a maximum value equal to the number of scanned lines in each of said rows,

whereby the characters on said display are scrolled with a minimum of movement of data within said memory.

17. The addressing means defined by claim 16 wherein said sequence of digital numbers is incremented for each displayed frame.

18. In a digital computer which includes a single chip central processing unit (CPU), a random-access memory (RAM), an address bus interconnecting said CPU and RAM such that said CPU addresses locations in said RAM, and a data bus coupled to said CPU and RAM, said CPU for certain functions addressing the zero page in said RAM by providing binary zeroes on certain lines of said address bus; an improvement comprising:

a detection circuit for detecting said binary zeroes on said certain lines of said address bus;

a register for storing digital signals, said register coupled to said data bus for receiving digital signals from said data bus; and,

a multiplexer for selecting between said digital signals stored in said register and said certain lines of said address bus, said multiplexer being controlled by said detection circuit so as to select said register when said binary zeroes are detected on said certain lines of said address bus;

whereby data for said certain functions normally stored on page one of said RAM, may be stored elsewhere in said RAM, and still easily addressed by said CPU.

19. The improvement defined by claim 18 wherein one of said stored signals from said register is coupled to said multiplexer through an exclusive OR gate, said gate being coupled to one of said certain lines of said address bus.

20. The improvement defined by claim 18 or 19 wherein said computer provides an alternate stack sig-

nal and wherein said detection circuit also detects addresses for page one on said address bus, and said multiplexer selects said register if said page one addresses are detected and said alternate stack signal is in a predetermined state.

21. In a digital computer which includes a central processing unit (CPU), a random-access memory (RAM), an address bus interconnecting said CPU and RAM such that said CPU addresses locations in said RAM and a data bus interconnecting said CPU and RAM, said CPU for certain functions addressing predetermined locations in said RAM with a predetermined range of address signals, an improvement comprising:

detection means for detecting said predetermined range of address signals, coupled to said address bus;

register means for storing digital signals, coupled to said data bus, and;

switching means for coupling said digital signals stored in said register means to said address bus when said detection means detects said predetermined range of said address signals, said switching means also for coupling said digital signals stored in said register means to said address bus when a certain direct memory access (DMA) signal is in a predetermined state;

a read-only memory (ROM) coupled between said address bus and said data bus, said ROM in response to signals on said address bus providing instructions to said CPU on said data bus to cause said CPU to increment address signals when said DMA signal is in said predetermined state;

said register providing a pointer for locations in said RAM when said DMA signal is in said predetermined state, and said register providing RAM address signals when said certain functions are selected by said CPU,

whereby data for said certain functions normally stored by said CPU in said predetermined locations may be stored elsewhere in said RAM, thereby enhancing the performance of said computer.

22. The improvement defined by claim 21 wherein said switching means comprise a multiplexer which selects said register when said detection means detects all binary zeroes or when said DMA signal is in said predetermined state.

* * * * *

55

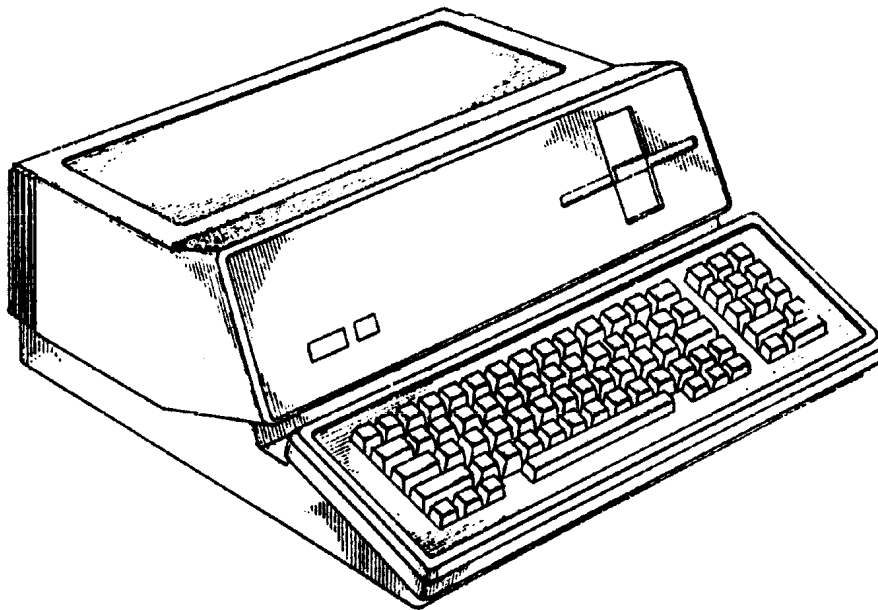
60

65

FINIS



Apple /// Computer Information



APPLE /// Plus PATENT

Patent # 4,533,909 -- 06 August 1983

ADDED BY DAVID T CRAIG • 2006

United States Patent [19]
Sander

[11] **Patent Number:** 4,533,909
 [45] **Date of Patent:** Aug. 6, 1985

- [54] **COMPUTER WITH COLOR DISPLAY**
- [75] **Inventor:** Wendell B. Sander, San Jose, Calif.
- [73] **Assignee:** Apple Computer, Inc., Cupertino, Calif.
- [21] **Appl. No.:** 560,529
- [22] **Filed:** Dec. 12, 1983

4,310,838 1/1982 Juso et al. 340/701
 4,360,804 11/1982 Ohura 340/703

Primary Examiner—David L. Trafton
Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

Related U.S. Application Data

- [60] Continuation of Ser. No. 394,801, Jul. 2, 1982, abandoned, which is a division of Ser. No. 150,630, May 16, 1980, Pat. No. 4,383,296.
- [51] **Int. Cl.³** G09F 9/30
- [52] **U.S. Cl.** 340/703; 340/803; 340/802
- [58] **Field of Search** 340/701, 703
- [56] **References Cited**

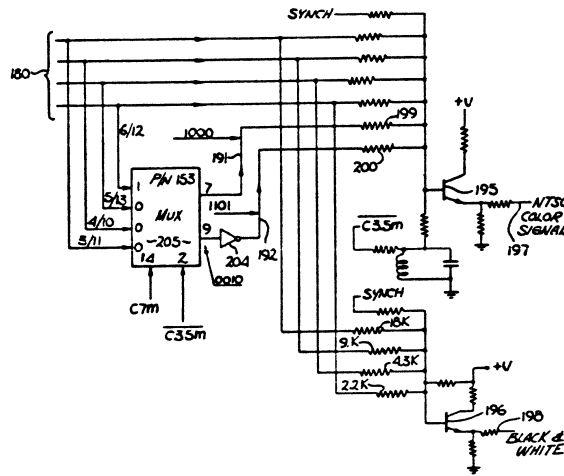
U.S. PATENT DOCUMENTS

4,136,359 1/1979 Wozniak 358/17

[57] **ABSTRACT**

A microcomputer system with video display capability, particularly suited for small business applications and home use is described. The CPU performance is enhanced by permitting zero page data to be stored throughout the memory. The circuitry permitting this capability also provides a pointer for improved direct memory access. Through unique circuitry resembling "bank switching" improved memory mapping is obtained. 4-bit digital signals are converted to an AC chroma signal and a separate luminance signal for display modes. Display modes include high resolution modes, one of which displays 80 characters per line.

11 Claims, 9 Drawing Figures



Apple /// Plus Computer

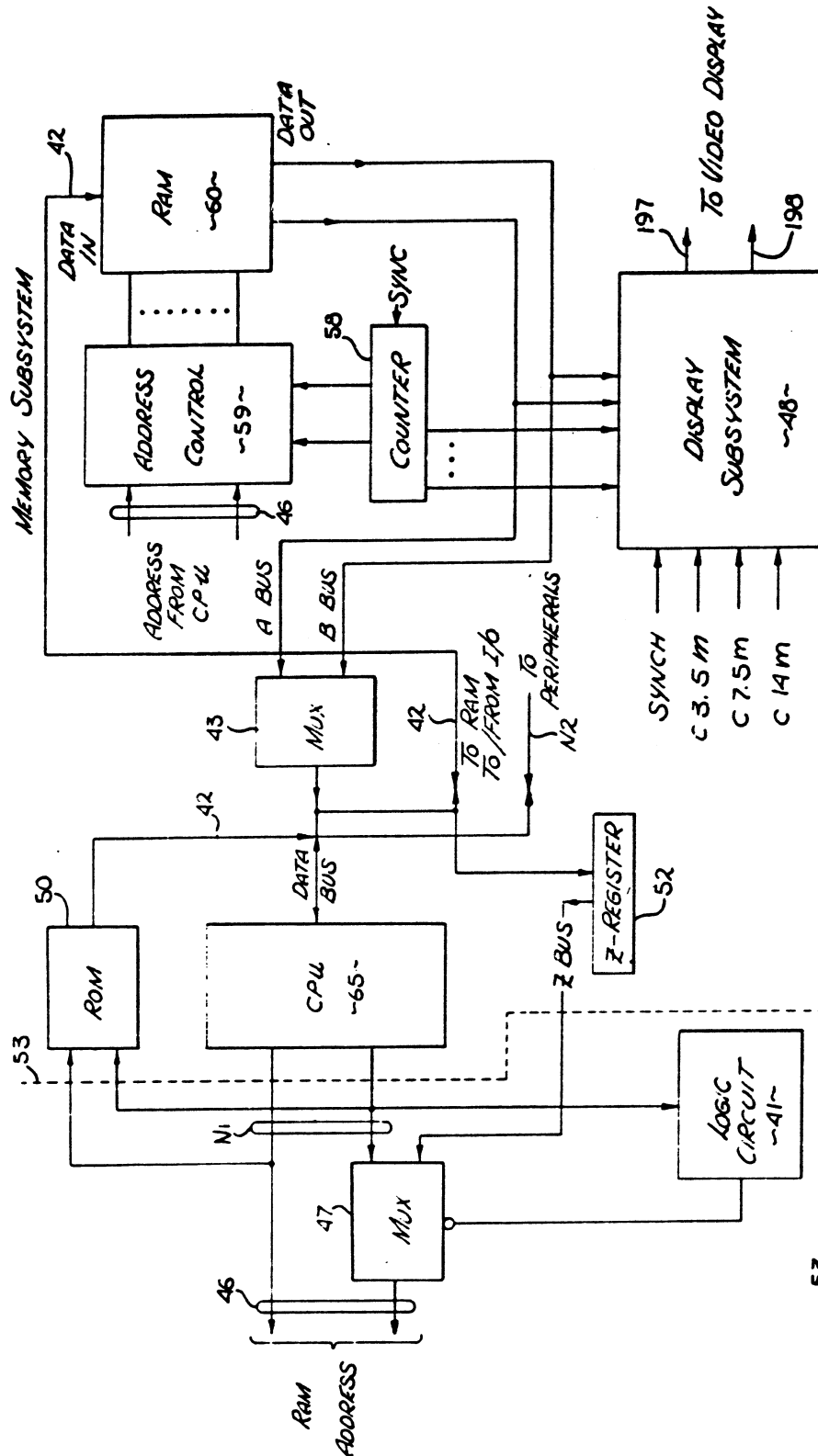


Fig. 1

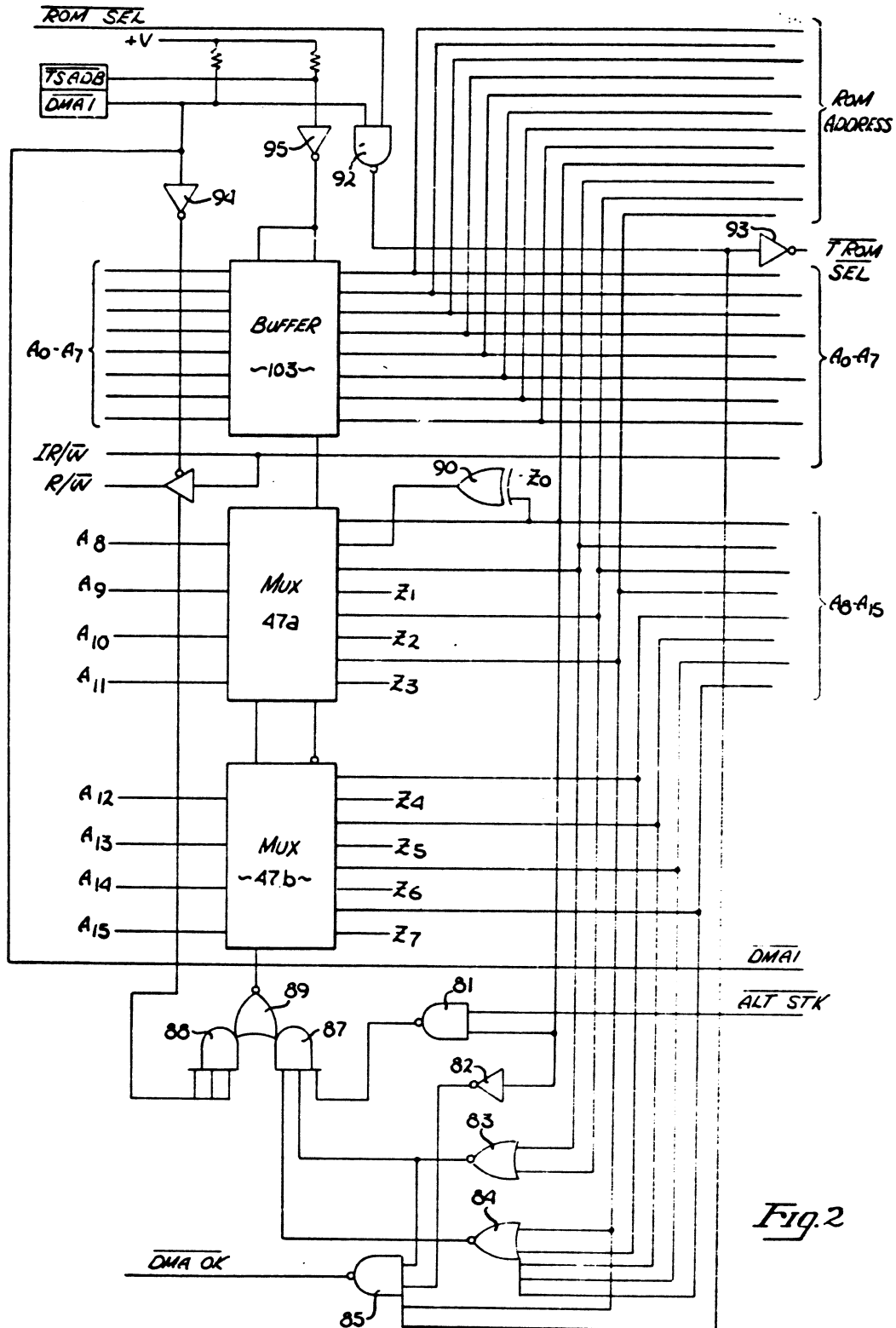


Fig. 2

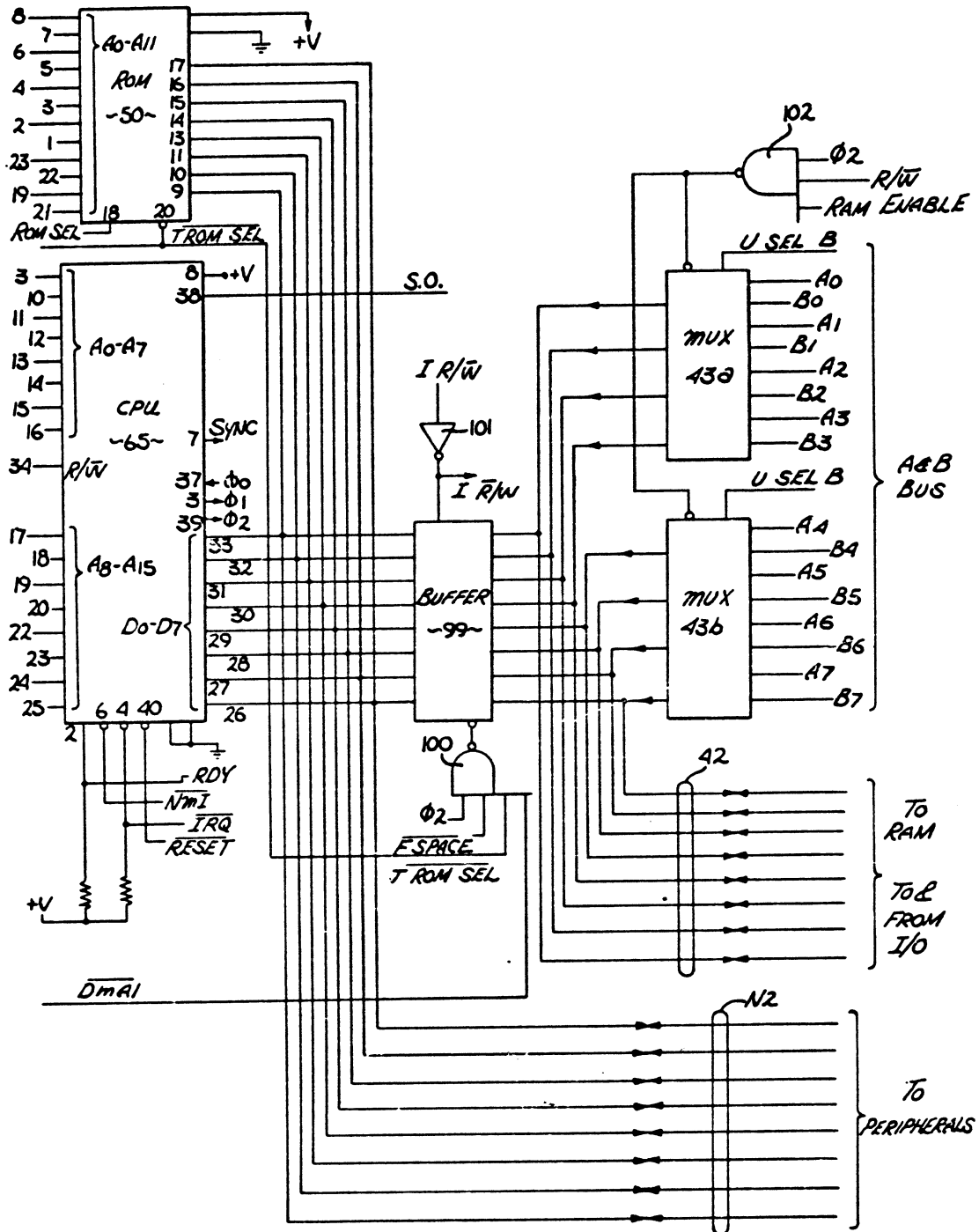


Fig. 3

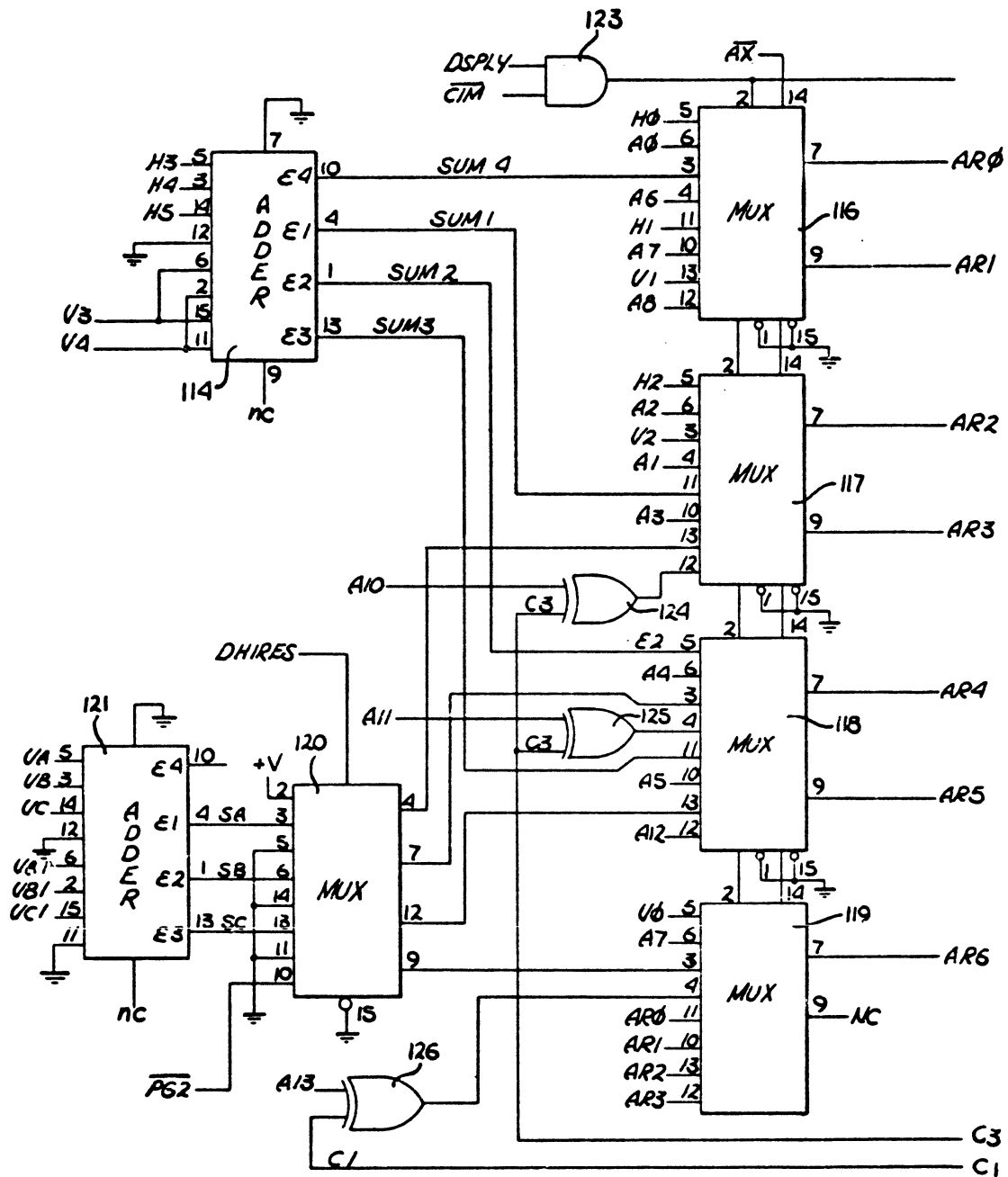
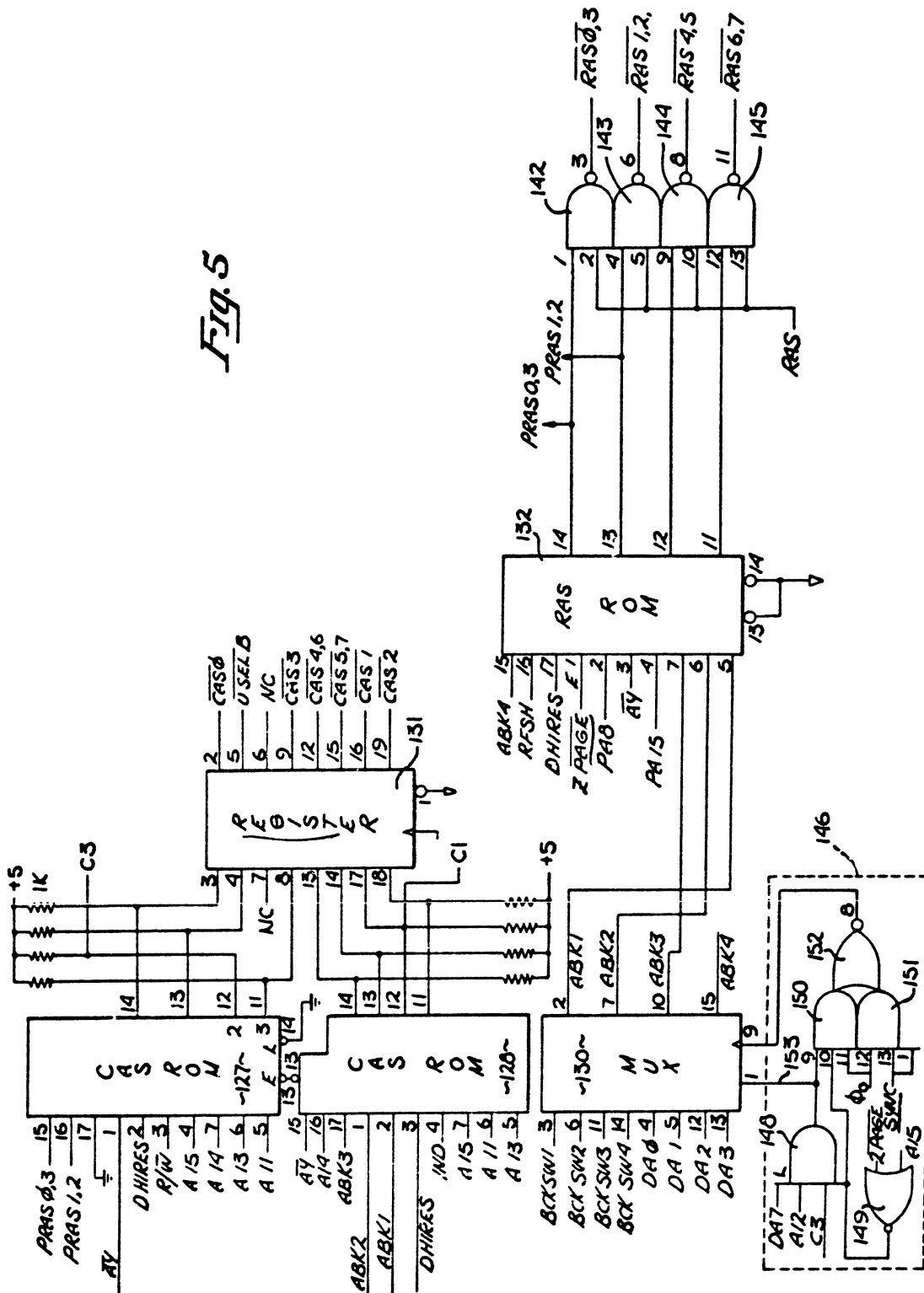


Fig. 4

Fig. 5



U.S. Patent Aug. 6, 1985

Sheet 6 of 8

4,533,909

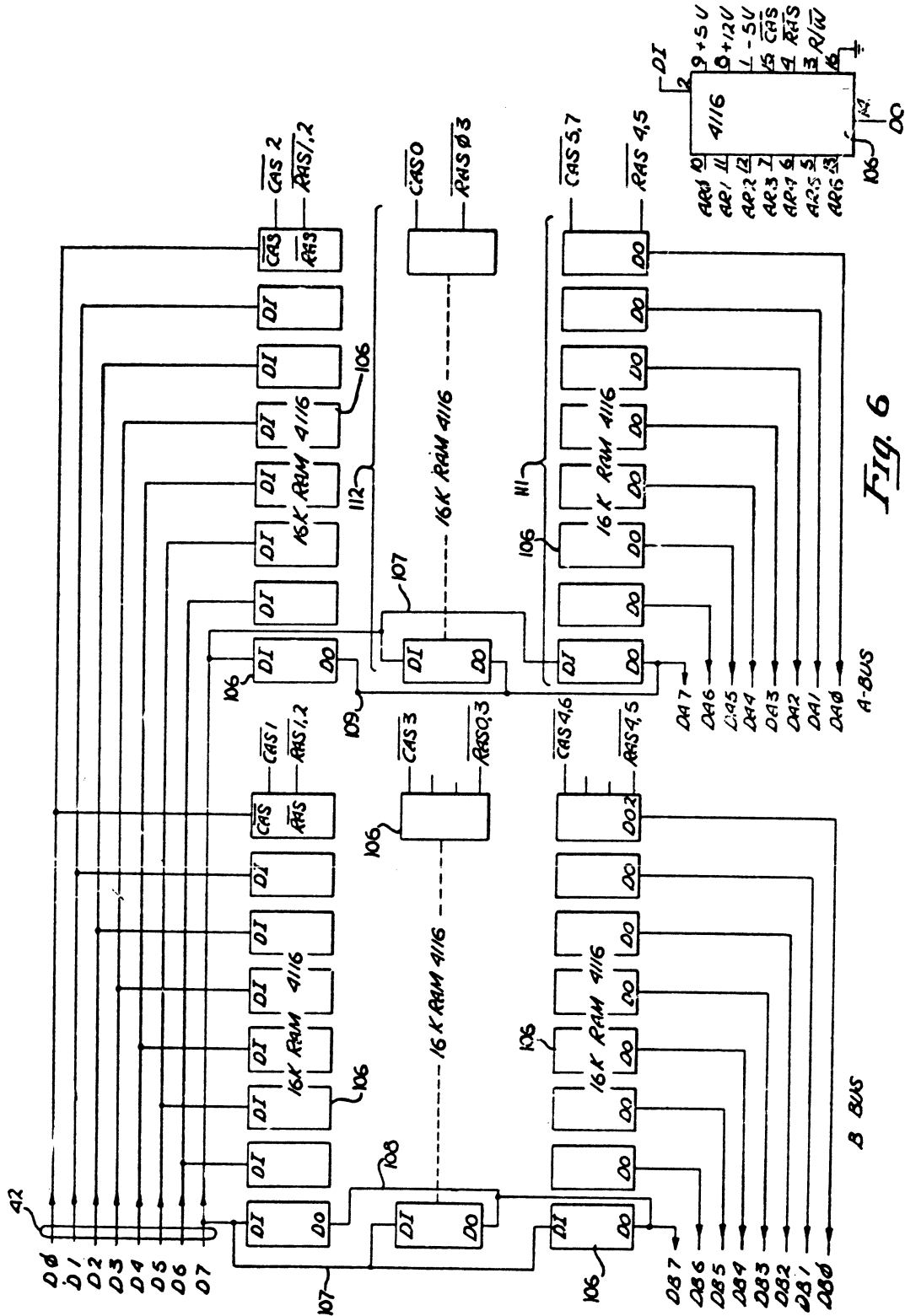


Fig. 6

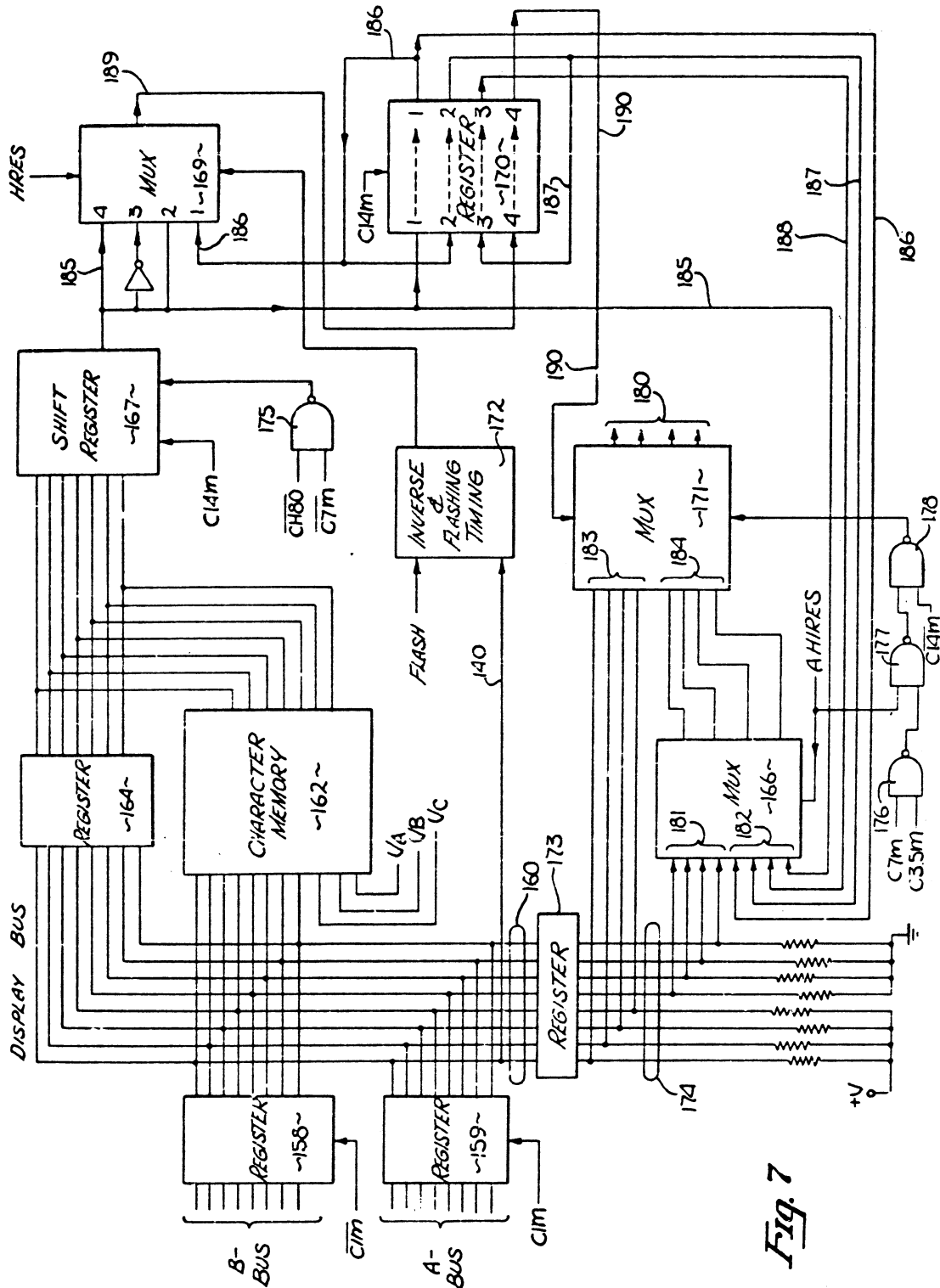


Fig. 7

U.S. Patent Aug. 6, 1985

Sheet 8 of 8

4,533,909

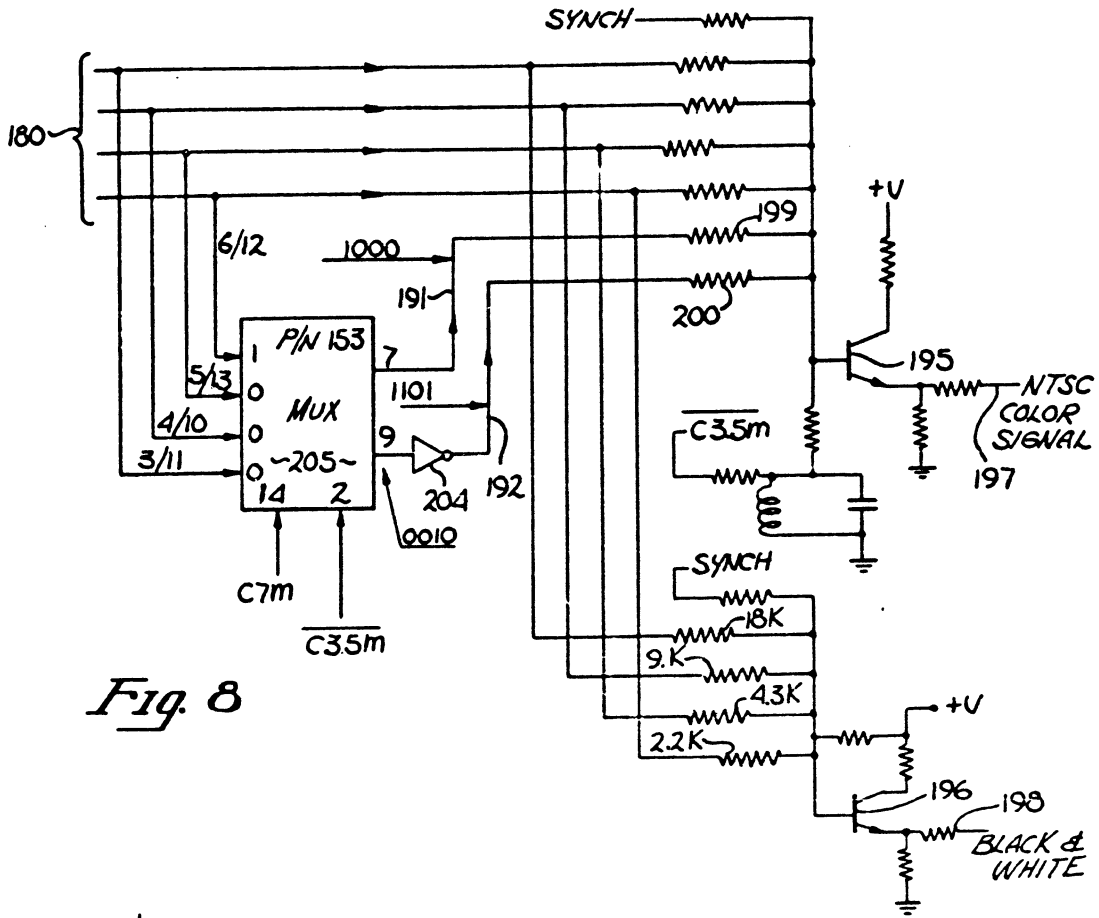


Fig. 8

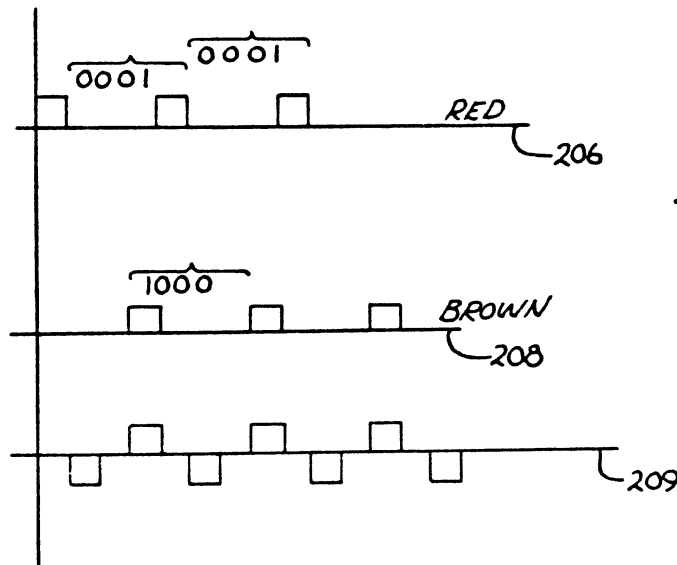


Fig. 9

4,533,909

1

COMPUTER WITH COLOR DISPLAY

This is a continuation of application Ser. No. 394,801 filed July 2, 1982, now abandoned, which is a divisional of application Ser. No. 150,630 filed May 16, 1980, now U.S. Pat. No. 4,383,296.

BACKGROUND OF THE INVENTION

The invention relates to the field of digital computers, particularly microcomputers, having video display capabilities.

Prior Art

In the last few years, there has been rapid growth in the use of digital computers in homes by hobbyists, for small business and for routine engineering and scientific application. For the most part, these needs have been met with self-contained, relatively inexpensive microcomputers or microprocessors with essential peripherals, including disc drives and with relatively easy to manage computer programs. The design of computers for these needs requires considerable ingenuity since each computer must meet a wide range of applications and because this market is particularly cost conscious.

A home or small business computer must, for example, operate with a number of different program languages, including those requiring relatively large memories, such as Pascal. The computer should interface with a standard raster scanned display and provide a wide range of display capabilities, such as high density alpha-numeric character displays needed for word processing in addition to high resolution graphics displays.

To meet these specialized computer needs, generally requires that a relatively inexpensive microprocessor be used and that the capability of the processor be enhanced through circuit techniques. This reduces the overall cost of the computer by reducing, for example, power needs, bus structures, etc. Another important consideration is that the new computers be capable of using programs developed for earlier models.

As will be seen, the presently described microcomputer is ideally suited for home and small business applications. It provides a wide range of capabilities including advanced display capabilities not found in comparable prior art computers.

The closest prior art computer known to applicant is commercially available under the trademark, Apple-II. Portions of that computer are described in U.S. Pat. No. 4,136,359.

SUMMARY OF THE INVENTION

A digital computer which includes a central processing unit (CPU) and a random-access memory (RAM) with interconnecting address bus and data bus is described. One aspect of the present invention involves the increased capability of the CPU by allowing base page or zero page data to be stored throughout the memory. Alternate stack locations and an improved direct memory access capability are also provided by the same circuitry. Detection means are used for detecting a predetermined address range such as the zero page. This detection means causes a special register (Z-register) to be coupled into the address bus. The contents of this Z-register provide, for example, a pointer during direct memory access, or alternate stack locations for storing data normally stored on page one.

2

The memory of the invented computer is organized in an unusual manner to provide compatibility with the 8-bit data bus and yet provide high data rates (16-bits/MHz) needed for high resolution displays. A first plurality of memory devices are connected to a first memory output bus; these memory devices are also connected to the data bus. The memory includes a second plurality of memory devices which are also connected to the data bus; however, the outputs of these second devices are coupled to a second output memory bus. First switching means permit the first and second memory buses to be connected to the display for high data rate transfers. Second switching means permit either one of the memory buses to be connected to the data bus during non-display modes.

The addressing capability of the memory is greatly enhanced not only through bank switching, but through a novel remapping which does not require the CPU control associated with bank switching. In effect, the "unused" bits from one of the first and second memory buses are used for remapping purposes. This mode of operation is particularly useful for providing toggling between two separate portions of the memory.

The display subsystem of the described computer generates video color signal in a unique manner. A 4-bit color code as used in the prior art, is also used with the described display subsystem. However, this code is used to generate an AC chrominance signal and a separate DC luminance signal. This provides enhanced color capability over similar prior art color displays.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram showing the major components and subsystems of the invented and described microcomputer system.

FIGS. 2 and 3 together show the central processing unit (CPU) and the architecture associated with this CPU, particularly the address bus and data bus. FIG. 2 is a circuit diagram primarily showing the address bus and the logic means associated with this bus. FIG. 3 is a circuit diagram primarily showing the data bus and its interconnection with the memory buses (A bus and B bus), bootstrap read-only memory, and input/output ports.

FIGS. 4, 5 and 6 show the memory subsystem. FIG. 4 is a circuit diagram primarily showing the circuitry for selecting between address signals from the address bus and display counter signals. FIG. 5 is a circuit diagram primarily showing the generation of various "select" signals for the memory devices. FIG. 6 is a circuit diagram showing the organization of the random-access memory and its interconnection with the data bus and memory output buses.

FIGS. 7 and 8 illustrate the display subsystem of the invented computer. FIG. 7 is a circuit diagram showing the circuitry for generating the digital signals used for the video display. FIG. 8 is a circuit diagram of the circuitry used to convert the digital signals to analog video signals.

FIG. 9 is a graph of several waveforms used to describe a prior art circuit and the circuit of FIG. 8.

DETAILED DESCRIPTION OF THE INVENTION

A microcomputer system capable of driving a raster scanned video display is disclosed. In the following description, numerous specific details such as specific part numbers, clock rates, etc., are set forth to provide

4,533,909

3

a thorough understanding of the present invention. However, it will be obvious to one skilled in the art that the inventive concepts described in this patent may be practiced without these specific details. In other instances, well-known circuits have been shown in block diagram form in order not to obscure the present invention in unnecessary detail.

Referring first to FIG. 1, in general the described computer includes a central processing unit (CPU) 65, its associated data bus 42, address bus 46 a memory subsystem and a display subsystem 58.

The address bus 46 from the CPU is coupled to the memory subsystem to permit the selection of locations in memory. Some of the address signals pass through a multiplexer 47. For some modes of operation, signals from a register 52 are coupled through the multiplexer 47 onto the bus 46. The register 52 is identified as the Z-register and is coupled to the multiplexer 47 by the Z bus. The general description of the multiplexer 47 and its control by the logic circuit 41 are described in detail in conjunction with FIG. 2. In general, the circuitry shown to the left of the dotted line 53 is included in FIG. 2 while the CPU 65, memory 50, data bus 42 and multiplexer 43 are shown in detail in FIG. 3.

The address bus N1 is coupled to the read-only memory 50. The output of this memory is coupled to the computer's data bus 42. The read-only memory (ROM) 50, as will be described, stores test routines, and other data of a general bootstrap nature for system initialization.

The data bus 42 couples data to the random-access memory (RAM) 60 and to and from I/O ports. This bus also couples data to the Z-register 52 and other commonly used registers not illustrated. The data bus 42 receives data from the RAM 60 through the A bus and B bus which are selected by multiplexer 43. The peripheral bus N2 is used, as is better illustrated in FIG. 3, for coupling to peripherals.

The memory subsystem is shown in detail in FIGS. 4, 5 and 6. The address control means which receives addresses on bus 46, makes the final selection of memory locations within the RAM 60. Bank switching, addressing for display purposes, scrolling and other memory mapping is controlled by the address control means 59 as will be described in greater detail in conjunction with FIGS. 4 and 5. The RAM 60 is shown in detail in FIG. 6. The counter 58 which is synchronized with the horizontal and vertical display signals, provides signals both to the address control means 59 and to the display subsystem 48.

The display subsystem receives data from the RAM 60 on the A bus and B bus and converts these digital signals to video signals which control a standard raster scanned display. A standard NTSC color signal is generated on line 197 and a black and white video signal on line 198. The same signals used to generate these video signals can be used to generate separate red, green, blue (RGB) video signals. The display subsystem 48 receives numerous timing signals including the standard color reference signal shown as 3.5 MHz (C3.5M). This subsystem is described in detail in FIGS. 7 and 8.

COMPUTER ARCHITECTURE

In the presently preferred embodiment, the CPU 65 (microprocessor) employed with the described computer is a commercially available component, the 6502A. This 8-bit processor (8-bit data bus) which has a 16-bit address bus is shown in FIG. 3 with its intercon-

4

nections to the remainder of the computer. The pin number for each interconnection is shown adjacent to the corresponding line. In many cases, the nomenclature associated with the 6502A (CPU 65) is used in this application. For example, pin 6 receives the nonmaskable interrupt signal (\overline{NMI}), and pin 4 is coupled to receive the interrupt request signal (\overline{IRQ}). Some of the signals employed with the CPU 65, which are well-known in the art, and which are not necessary for the understanding of the present invention are not described in detail in this application, such as the various synchronization signals and clocking signals. The address signals from the CPU 65 are identified as A_0 - A_7 and A_8 - A_{15} . The data signals associated with the CPU 65 are shown as D_0 - D_7 . As will be apparent to one skilled in the art, the inventive concepts described in this application may be employed with other microprocessors.

Referring now to FIGS. 2 and 3, the general architecture, particularly the architecture associated with the CPU 65 can best be seen. The address signals A_0 - A_7 are coupled to a buffer 103 by the bus shown primarily in FIG. 2. These address signals are also coupled to the ROM 50. The signals A_0 - A_7 after passing through the buffer 103 are coupled to the memory subsystem. The address signals A_8 - A_{15} (higher order address bits) are coupled through lines shown in FIG. 2 to the multiplexers 47a and 47b. The contents of the Z-register 52 of FIG. 1 is also connected to the multiplexers 47a and 47b through the Z-bus (Z_1 - Z_7). The multiplexers 47a and 47b allow the selection of either the signals A_8 - A_{15} from the CPU 65 or the contents of the Z-register (Z_1 - Z_7) for addressing the RAM 60. The output of these multiplexers are shown as A_8 - A_{15} ; this designation is used even when the Z-bus is selected. Note in the case of the Z_0 signal, this signal is coupled to the multiplexer 47a through the exclusive OR gate 90 for reasons which are explained later. The address signals A_8 - A_{11} are also coupled to the ROM 50, thus the signals A_0 - A_{11} are used for addressing the ROM 50. The signals A_8 - A_{15} are connected to the logic circuit shown in the lower left-hand corner of FIG. 2; this logic circuit corresponds to the logic circuit 41 of FIG. 1.

The input and output data signals from the CPU 65 are coupled by a bidirectional bus to the bidirectional buffer 99 (FIG. 3). This buffer is selectively disabled by gate 100 to allow the output of ROM 50 to be communicated to CPU 65 and during other times not pertinent to the present discussion. The direction of flow through the buffer 99 is controlled by a read/write signal coupled to the buffer through inverter 101. Data from the CPU 65 is coupled through the buffer 99 and bus 42 to the RAM 60 or to I/O ports. Data from the RAM 60 is communicated to CPU 65 or bus N2 from the A bus and B bus through the buffer 99. The 4 lines of the A bus and 4 lines of the B bus are coupled to the multiplexer 43a. Similarly, the other 4 lines of the A and B buses are coupled to the multiplexer 43b. Multiplexers 43a and 43b select the 8 lines of the A bus or B bus and communicate the data through to buffer 99 and bus 42. These multiplexers are selectively disabled (for example, during writing) by gate 102. As will be described later, the 16 lines of the A bus and B bus permits the reading of 16-bits from the RAM at one time. This provides a data rate of 16-bits/MHz which is necessary, for example, for an 80 character per line display. The data is loaded into the RAM 60, 8-bits at a time.

4,533,909

5.

The ROM 50, as mentioned, stores test programs, data needed to initialize various registers, character generation data (for RAM 162 of FIG. 7) and other related data. Specific programs employed in the presently preferred embodiment of the computer are set forth in Table 1 of U.S. Pat. No. 4,383,296. The ROM 50 is selected by control signals coupled to its pins 18 and 20, identified as signals ROM SEL and $\overline{\text{TROM SEL}}$. Any one of a plurality of commercially available read-only memories may be used for the ROM 50. In the presently preferred embodiment, commercially available Part No. SY2333 is used.

Referring now to this logic circuit (lower left-hand corner of FIG. 2), the NAND gate 81 receives the address signal A_8 and also the alternate stack signal identified as $\overline{\text{ALT STK}}$. The output of this gate provides one input to the AND gate 87. The A_8 signal is also coupled through the inverter 82 to one input terminal of the NAND gates 85 and 86. The address signals A_9 and A_{10} are coupled to the input terminals of the NOR gate 83. The output of this gate is coupled to one input terminal of the NAND gates 85 and 86 and the AND gate 87. The address signals A_{11} - A_{15} are coupled to the input terminals of the NOR gate 84. The signal A_{11} is also coupled to an input terminal of the NAND gate 85.

The outputs of the AND gates 87 and 88 (through NOR gate 89), controls the multiplexers 47a and 47b. When the output of gate 89 is low the Z-bus is selected, otherwise the address signals from the CPU 65 are selected.

The logic circuit above-described, along with the Z-bus and Z-register provide enhanced performance for the computer. First, this circuit permits the zero page or base page data to be stored throughout the RAM 60 rather than just on zero page. Secondly, this circuit enables addressing of alternate stack locations (other than page one). Lastly, this circuit through the Z-register provides a RAM pointer for direct memory access (DMA).

Assume for purposes of discussion that the CPU 65 is addressing the zero page of memory. That is, the higher order address bits A_8 - A_{15} are all zeros. The zeros for A_9 - A_{15} are detected by the gates 83 and 84. If all the inputs to these gates are zeros, the outputs of these gates are high which condition is communicated to the gate 87. A_8 which is also low, insures that the output of gate 81 will be high. Thus, all the inputs to gate 87 are high, causing the signal at the output of the gate 89 to drop. When this occurs, the Z-bus is selected. Instead of all the binary zeros from the CPU being coupled to the main memory (RAM 60), the contents of the Z-register form part of the address for the memory. Therefore, even though the CPU 65 has selected the zero page, nonetheless data may be written into or from any location of RAM 60 (including the zero page). This enhances the performance of the CPU, since for example, the time consumed in shifting data to and from a single zero page is minimized.

Normally, the CPU 65 selects page one for stack locations. This occurs when A_8 is high and A_9 - A_{15} are low. Assume first that the alternate stack locations have not been selected. Both inputs to gate 81 are high and its output is low. The low input to the gate 87 prevents the selection of the Z-bus. Thus, for these conditions the address signals A_0 - A_7 select stack locations on page one.

6

Next assume that page one has been selected by the CPU and that the $\overline{\text{ALT STK}}$ signal is low, indicating the alternate stack locations are to be selected. (A flag is set by the CPU to change the $\overline{\text{ALT STK}}$ signal). Since the $\overline{\text{ALT STK}}$ signal is low and A_8 is high, a high output occurs from the gate 81. All the inputs to gates 83 and 84 are low, therefore, high outputs occur from both these gates. The conditions of gate 87 are met, causing a high output from this gate and lowering the output from the gate 89. The Z-bus is thus selected by the multiplexers 47a and 47b. This allows the contents of the Z-register to be used as alternate locations. Non-zero page locations are assured by inverting A_8 . The exclusive OR gate 90 acts as a selective inverter. If A_8 is high and Z_0 is low, then A_8 at the output of the multiplexer 47a will be low. Note that during zero page selection when A_8 is low, the Z_0 signal is directly communicated through gate 90 to the output of multiplexer 47a.

Thus, the logic circuits along with the $\overline{\text{ALT STK}}$ signal allows alternate stack locations to be selected through the Z-bus. This further enhances the performance of the CPU which would otherwise be limited to page one for stack locations.

The logic circuit of FIG. 2 is also used along with the Z-register to provide a pointer during direct memory access (DMA). Assume that direct access to the computer's memory is required by a peripheral apparatus. To initiate the DMA mode the CPU provides an address between F800 and F8FF. Through a logic circuit not illustrated in FIGS. 2 and 3, the ROM SEL signal is brought low for addresses between F000 and FFFF. This signal is communicated to gate 93 and causes the output of gate 92 to rise ($\overline{\text{DMA1}}$ is high at this time). This rise in potential is communicated to one input of the gate 85. Additionally, gate 85 senses that the address bits A_8 , A_9 and A_{10} are low. This information is coupled to gate 85 through the inverter 82 and the NOR gate 83 as high signals. Also the fact that A_{11} is high is directly communicated to gate 85. Thus with the address between F800 and F8FF the $\overline{\text{DMA OK}}$ signal drops in potential. This is sensed by the peripheral apparatus which in turn causes the $\overline{\text{DMA 1}}$ signal to drop and provides a ready signal to the CPU 65. With the completion of this handshake, data may begin to be transferred to the RAM.

The $\overline{\text{DMA 1}}$ signal through gate 93 and inverter 93 forces the $\overline{\text{TROM SEL}}$ signal low. This signal in addition to being communicated to the ROM 50, is coupled to the buffer 99 through gate 100, disabling this buffer (during the reading of ROM 50). Also, the ready signal causes the CPU to come to a hard stop. Importantly, the $\overline{\text{DMA 1}}$ signal, after passing through the inverter 94 and the gates 88 and 89, assures the selection of the Z-register. The contents of the Z-register are fixed and provide a pointer to a page in the RAM.

Under the above conditions, the CPU increments the lower 8-bits of the address signal. The ROM 50 furnishes the instructions for incrementing the address, specifically SBC #1 and BEQ. The peripheral apparatus provides the data or receives the data in synchronization with the CPU operation. The peripheral also furnishes a read/write signal to indicate which operation is to occur. Data is then written into RAM via bus N2 and bus 42, or read from RAM via the A and B buses and bus N2.

Importantly, with the above DMA arrangement, addresses from the peripheral apparatus are not neces-

4,533,909

7

sary and the Z-register is used to provide a pointer to a page in RAM 60.

MEMORY SUBSYSTEM

The memory subsystem shown in FIG. 1 as the address control means 59 and RAM 60 is illustrated in detail in FIGS. 4, 5 and 6 as mentioned. In FIGS. 4 and 5, the memory control means is shown, while in FIG. 6 the memory devices and their organization are illustrated. The address control means of FIGS. 4 and 5 receives the address signals from the CPU 65 (A_0 - A_{15}), the count in the vertical and horizontal counters (counter 58 of FIG. 1) which are used during display modes, control signals from the CPU and other signals. In general, this control means develops the address signals which are coupled to the RAM of FIG. 6 including the column address and row address signals, commonly referred to as \overline{CAS} and \overline{RAS} . Other related functions are also shown in FIGS. 4 and 5, such as the circuitry which provides display scrolling, indirect RAM addressing and memory mapping.

The CPU 65 of FIG. 3 provides a 16-bit address for addressing the memory. Under ordinary circumstances this address limits the memory capacity to 64 K bytes. This size memory is insufficient in many applications, as for example, to effectively use the Pascal program language. As will be described in greater detail, the address control means of FIGS. 4 and 5 enable the use of a memory having a 96K byte or 128K byte capacity. One well-known technique which is used with the present invention for increasing this capacity is bank switching; this switching occurs under the control of the CPU. In addition, the address control means uses a unique indirect addressing mode which provides the benefits of bank switching, however, this mode does not require CPU control. This greatly enhances CPU operation with the larger memory (as will be described) when compared to the CPU controlled bank switching.

Referring first to FIG. 6, the RAM configuration is illustrated for a capacity of 96K bytes. The memory is organized into six rows, each of which includes eight 16K memory devices such as rows 111 and 112. In the presently preferred embodiment, Part No. 4116 MOS dynamic RAMs are used. (The pin designations and signal designations refer to this memory device.) Obviously, other memory devices may be employed.

Input data to these memory devices 106 is provided from the bus 42. Each line in the bus 42 is connected to the data input terminal of one device 106 in each row. The interconnection of this bus with each of the memory devices is not shown in FIG. 6 in order not to over-complicate this drawing. By way of example, however, line 107 connects the data bit D7 to the data input terminal of one of the memory devices in each of the six rows.

Three rows of devices 106 have their output terminals coupled to the A bus, and three rows are similarly coupled to the B bus. By way of example, line 108 connects three output terminals of devices 106 to the DB7 line of the B bus while line 109 connects three output terminals of the devices 106 to the DA7 line of the A bus.

The described memory devices 106 are each organized as a 16KX1 memory. Thus, each device receives a 14-bit address which is time multiplexed into two, 7-bit addresses. This multiplexing occurs under the control of the \overline{CAS} and \overline{RAS} signals as is well-known. The lines coupling the address signals to each of the

8

devices in FIG. 6 are not illustrated. However, in the lower right-hand corner of FIG. 6, the various signals applied to each device (including the address signals), along with the corresponding pin numbers are shown. Other circuitry not illustrated is the refresh control circuitry which operates in a well-known manner in conjunction with the \overline{CAS} , \overline{RAS} and address signals to refresh the dynamic devices.

Each row of memory devices 106 receives a unique combination of \overline{CAS} and \overline{RAS} signals. For example, row 111 receives \overline{CAS} 5, 7 and \overline{RAS} 4, 5; similarly, row 112 receives \overline{CAS} 0 and \overline{RAS} 0, 3. The generation of these \overline{CAS} and \overline{RAS} signals is described in conjunction with FIG. 5. These signals (along with the 14-bit address signals) permit the selection of a single 8-bit location in the 96K byte memory (for writing) and also the selection (for reading) of 16-bit locations.

The memory of FIG. 6 may be expanded to a 128K byte memory by using 32K memory devices, such as Part No. 4132. In this case, four rows of eight, 32K memory devices are used with each row receiving two \overline{CAS} and \overline{RAS} signals.

Before reviewing FIG. 4, a general understanding of the organization of the display is helpful. The display, during certain modes, is organized into 80 horizontal segments and 24 vertical segments for a total of 1920 blocks. 11-bits of the counter 58 of FIG. 1 are used as part of the address signals for the memory to access data for displaying during these modes. These counter signals are shown in FIG. 4 as H_0 - H_5 and V_0 - V_4 . During other display modes each horizontal segment is further divided into 8 segments (e.g. for displaying 80 alpha numeric characters per line). This requires 3 additional vertical timing signals shown as V_A , V_B and V_C in FIGS. 4 and 7.

Often in the prior art, two separate counters are used to supply the timing/address signals for accessing a memory when the data in the memory is displayed. The count in one counter represents the horizontal lines of the screen (vertical count) and the other the position along each line, (horizontal or dot count). In many prior art displays the most significant bit of the dot counter is used to increment the line counter. Data in memory intended for display is mapped with a one-to-one correlation to the counts in these counters. In another prior art system (implemented in the Apple-II computer sold by Apple Computer, Inc.) this one-to-one correlation is not used. Rather, to conserve on circuitry, a single counter is employed and a more dispersed mapping is used in the memory. (Note that where a maximum horizontal count of 80 is used, this number cannot be represented by all ones in a digital counter and thus the vertical counter cannot easily be incremented by the most significant bit in the horizontal counter.) Since this more dispersed mapping technique is part of the prior art and not critical to an understanding of the present invention, it shall not be described in detail. However, the manner in which it is implemented shall be discussed in conjunction with the adder 114 of FIG. 4. For purposes of discussion, the signals from the counter 58 of FIG. 1 are designated as either vertical (V) or horizontal (H).

Referring now to FIG. 4, the selection of either the counter signals on the address signals from the CPU is made by the multiplexers 116, 117, 118 and 119. Each of these commercially available multiplexers (Part No. 153) couples one of four input lines to an output line. There are eight inputs to multiplexers 116, 117 and 118

and the outputs of these multiplexers provide the address signals for the memories (AR0 through AR5). The multiplexer 119 has four inputs on its pins 3, 4, 5, 6 and provides a single output on pin 7, the AR6 address signal. (The signals supplied to pins 11, 12 and 13 of multiplexer 119 are for clamping purposes only.)

The \overline{AX} signal is applied to the pin 14 of each of the multiplexers. The signal on this line and the signal applied to pin 2, determines which of the four inputs is coupled to each of the outputs of the multiplexers. The \overline{AX} signal is a RAM timing signal for clocking the first 7 bits and second 7 bits of the multiplexed 14-bit address applied to each of the memory devices 106. The other control signal to the multiplexers is developed through the AND gate 123. The inputs to this gate are the display signal (DSPLY) which indicates that the computer is in a display mode and a clocking signal, specifically a 1 MHz timing signal (CIM). The output of the AND gate 123 determines whether the address signals from the CPU or the signals associated with the counter 58 of FIG. 1 are selected.

Assume for purposes of discussion that the display has not been selected, and thus, the output of gate 123 is low. The \overline{AX} signal then selects for pin 7 of multiplexer 116 first the address signal A_0 and then A_6 . Likewise, each of the multiplexers selects an address signal (except for those associated with exclusive OR gates 124 and 125 which shall be discussed). If the display signal is high and an output is present from the gate 123, then, by way of example, the \overline{AX} signal first causes the H_1 signal and then the V_1 signal to be connected to the AR1 address line. Similarly, signals corresponding to the vertical and horizontal count are coupled to the other address lines during display modes.

The adder 114 is an ordinary digital adder for adding two 4-bit digital nibbles and for providing a digital sum signal. A commercially available adder (Part No. 283) is employed. The carry-in terminal (pin 7) is grounded and no carry-outs occur since one of the inputs (pin 12) is grounded. The adder sums the digital signal corresponding to H_3 , H_4 and H_5 with the digital signal corresponding to V_3 , V_4 , V_3 , V_4 . The resultant sum signal is coupled to the multiplexers 116, 117 and 118 as illustrated. The summing of these horizontal and vertical counter signals is used to provide the more dispersed mapping as previously discussed.

The adder 121 is identical to adder 114 and is coupled to sum the three least significant vertical counter bits from the counter 58 (FIG. 2) with the signals VA1, VB1 and VC1. The sum is selected by the multiplexer 120 during the high resolution display modes and also during scrolling as will be described. These sum signals are coupled to the multiplexers 117, 118 and 119. During the low resolution display modes, the multiplexer 120 couples ground signals or the page 2 signal ($\overline{PG2}$) to the multiplexers 117, 118 and 119. (The $\overline{PG2}$ signal is used for special mapping purposes, not pertinent to the present invention.) During the high resolution modes when the display is not being scrolled, the VA1, VB2 and VB3 signals are at ground potential and thus no summing occurs within adder 121 and the VA, VB and VC signals are coupled directly to the multiplexers 117, 118 and 119.

The address signals A_{10} , A_{11} , and A_{13} from the CPU are coupled to the multiplexers 117, 118 and 119, respectively, through exclusive OR gates 124, 125, and 126, respectively. The other input terminals to gates 124 and 125 receive the C_3 signal, while the other input

terminal of the gate 126 receives the C_1 signal. (The development of the C_1 and C_3 signals is illustrated in FIG. 5.) The gates 124, 125 and 126 provide mapping compensation within the memory. As the computer and memory are presently implemented, the sequence in which the various portions of the display are generated is not the same as the sequence in which the data is removed from memory for display. These gates provide compensating addresses and, in effect, cause a remapping so that the proper sequence is maintained when data is read from the memory for the display. These gates are shown to provide a complete disclosure of the presently preferred embodiment, however, they are not critical to the present invention.

In operation, the circuitry of FIG. 4, as mentioned, selects the address signals which are applied to each of the memory devices, either from the CPU or counter if the display mode is selected. It should be noted that not all of the address bits from the CPU are coupled to the multiplexers 116 through 119. Some of these address bits, as will be described in conjunction with FIG. 5, are used to develop the various \overline{CAS} and \overline{RAS} signals and thus select different rows within the memory of FIG. 6.

The scrolling operation which is used is somewhat unusual in that each line of the display is separately moved up (line-by-line) with one line of data in memory being moved for each frame. This technique provides a uniform, esthetically pleasing, scroll. Scrolling the screen one line per frame can be achieved by moving all the data in the memory into a new position for each frame. This would be very time consuming and impractical. With the described technique, only one-eighth of the data in the memory is moved for each new frame.

Referring to the adder 121, as mentioned, the signals V_A , V_B and V_C are the three least significant vertical counter bits from the counter 58. These bits or counts, by way of example, represent the 8 horizontal lines of each character. In adder 12, a 3-bit digital signal, VA1, VB1 and VC1, is added to the count from counter 58. This 3-bit signal is constant during each frame, however, it is incremented for each new frame.

During a first frame, 000 is added to the vertical count. During a second frame, 001 is added, and during a third frame 010 is added, and so on. By adding this digital signal to the count from counter 58, the addresses to the memory are changed in the vertical sense. During the first frame when 000 is added, the display remains unaffected. During the next frame, when 001 is added to the vertical count, instead of first displaying the first line of a character, the second line of each character is displayed at the top of each character space and each subsequent line of the character is likewise moved up one line. If data in memory is not moved, the first line of the character would appear at the bottom of each character. Note when 001 is added to 111 from the counter, 000 results. Thus, the first line of characters would be addressed when the beam is scanning the eighth line of characters. To prevent this, the data corresponding to the first line of each character is moved in memory for this frame. The first line of one character is moved up and becomes the bottom line of the character directly above it. When 010 is added, the process is again repeated. For example, the third line of each character is first displayed in each character space and the second line of each character is moved up to become the bottom line of the character directly above it. This process is repeated to scroll the data. The movement of

11

data in memory is controlled by the CPU in a well-known manner.

Thus, through use of adder 121, an even, continuous scroll is obtained without moving all the data in memory for each frame. Rather, only 1/8th of the data is moved for each frame.

Referring now to FIG. 5, the circuitry used to extend the addressing from the CPU is illustrated. In general, the CAS signals are generated by the ROMs 127 and 128. The RAS signals are generated by the ROM 132. The multiplexer 130 allows the selection of either the bank switching signals, or the unique indirect addressing mode when "bank switching" occurs without direct commands from the CPU.

The CAS ROM 127 receives as an address the following signals: PRAS_{0,3}, φ₃, PRAS_{1,2}, AY, DHIREs, R/W, A₁₁, A₁₃, A₁₄, and A₁₅. As the PRAS_{0,3} and PRAS_{1,2} represent the RAS signals being used. These signals are high when the respective RAS signal is active. As previously mentioned, the AY signal is high for display modes and the DHIREs signal is high for high resolution display modes. The CAS ROM 128 receives as address signals the ABK₁, ABK₂, and ABK₃ signals and also DHIREs, AY, IND, A₁₁, A₁₃, A₁₄, and A₁₅.

The ROMs 127 and 128 are programmed to implement the following equations.

$$\overline{PCAS0} = (PRAS0,3 \cdot \overline{DHIREs} \cdot \overline{AY} + AY \cdot (\overline{A15} \cdot \overline{A14} \cdot \overline{A13} \cdot \overline{A11} \cdot R/WN + \overline{A15} \cdot \overline{A14} \cdot \overline{A13} \cdot R/WN + A15 \cdot \overline{A14} \cdot A13 + A15 \cdot A14 \cdot A13 \cdot \overline{A11})) \quad (1)$$

$$\overline{PCAS2} = (DHIREs \cdot \overline{AY} + AY \cdot (\overline{ABK1} \cdot \overline{ABK2} \cdot \overline{ABK3} \cdot \overline{IND} + ABK1 \cdot ABK2 \cdot ABK3) \cdot (\overline{A15} \cdot A14) + AY \cdot IND \cdot \overline{ABK1} \cdot \overline{ABK2} \cdot \overline{ABK3} \cdot \overline{A15} \cdot (\overline{A14} \cdot A13 + A14 \cdot \overline{A13})) \quad (2)$$

$$PCAS3 = (PRAS0,3 \cdot \overline{DHIREs} \cdot \overline{AY} + AY \cdot (\overline{A15} \cdot \overline{A14} \cdot \overline{A13} \cdot A11 + A15 \cdot A14 \cdot \overline{A13} \cdot \overline{A11} + A15 \cdot A14 \cdot \overline{A13})) \quad (3)$$

$$\overline{PCAS4,6} = (AY \cdot \overline{IND} \cdot \overline{ABK3} \cdot \overline{A15} \cdot (ABK1 \cdot \overline{ABK2} + ABK1) \cdot ABK2) \cdot (\overline{A14} \cdot A13 + A14 \cdot \overline{A13}) + AY \cdot IND \cdot \overline{ABK3} \cdot (\overline{ABK2} \cdot \overline{ABK1} \cdot A15 + \overline{ABK2} \cdot ABK1 + ABK2 \cdot \overline{ABK1} \cdot \overline{A15}) \cdot A14 + AY \cdot \overline{IND} \cdot ABK1 \cdot ABK2 \cdot \overline{ABK3} \cdot (\overline{A15} \cdot \overline{A14} \cdot A13 + A15 \cdot \overline{A14} \cdot \overline{A13}) + AY \cdot IND \cdot \overline{ABK3} \cdot ABK2 \cdot \overline{A15} \cdot ABK1 + A15 \cdot \overline{ABK1} \cdot ABK1) \cdot (\overline{A14} \cdot \overline{A13} + A14 \cdot \overline{A13})) \quad (4)$$

$$\overline{PCAS5,7} = (AY \cdot \overline{IND} \cdot \overline{ABK3} \cdot (ABK1 \cdot \overline{ABK2} + ABK1) \cdot ABK2) \cdot (\overline{A14} \cdot A13 + A14 \cdot \overline{A13}) + AY \cdot IND \cdot \overline{ABK3} \cdot (\overline{ABK2} \cdot \overline{ABK1} \cdot A15 + \overline{ABK2} \cdot ABK1 + ABK2 \cdot \overline{ABK1} \cdot \overline{A15}) \cdot A14 + AY \cdot \overline{IND} \cdot ABK1 \cdot ABK2 \cdot \overline{ABK3} \cdot (\overline{A15} \cdot \overline{A14} \cdot A13 + A15 \cdot \overline{A14} \cdot \overline{A13}) + AY \cdot IND \cdot \overline{ABK3} \cdot ABK2 \cdot \overline{A15} \cdot ABK1 + A15 \cdot \overline{ABK1} \cdot ABK1) \cdot (\overline{A14} \cdot \overline{A13} + A14 \cdot \overline{A13})) \quad (5)$$

12

$$\overline{PCAS5,7} = (AY \cdot \overline{IND} \cdot \overline{ABK3} \cdot (ABK1 \cdot \overline{ABK2} + ABK1) \cdot ABK2) \cdot (\overline{A14} \cdot A13 + A14 \cdot \overline{A13}) + AY \cdot IND \cdot \overline{ABK3} \cdot (\overline{ABK2} \cdot \overline{ABK1} \cdot A15 + \overline{ABK2} \cdot ABK1 + ABK2 \cdot \overline{ABK1} \cdot \overline{A15}) \cdot A14 + AY \cdot \overline{IND} \cdot ABK1 \cdot ABK2 \cdot \overline{ABK3} \cdot (\overline{A15} \cdot A14) + AY \cdot IND \cdot \overline{ABK3} \cdot ABK2 \cdot (\overline{A15} \cdot ABK1 + A15 \cdot \overline{ABK1}) \cdot (\overline{A14} \cdot A13 + A14 \cdot \overline{A13})) \quad (5)$$

In effect these ROMs are programmed to allow selection of predetermined rows in the memory, based on the address signals A₁₀, A₁₃, A₁₄ and A₁₅ (ignoring for a moment the contribution of the RAS signals and the other signals appearing in the equations).

The outputs of the CAS ROMs 127 and 128 are coupled to the register 131. Register 131 is a commercially available register which permits the enabling of output signals (Part No. 374). During accessing of the memory the various CAS signals (CAS₀ through CAS₇) are coupled to the memory of FIG. 6 to permit selection of the appropriate memory devices. The signal USELB from CAS ROM 127 through register 131 selects either the A bus or B bus. This signal is coupled to the multiplexers 43a and 43b of FIG. 3.

During normal operation, the multiplexer 130 selects the bank switching signals BCKSW₁ through BCKSW₄. These four signals (or alternatively four signals from the A bus) provide four of the inputs (address signals) to the ROM 132. The other inputs to this ROM are the DHIREs, Z PAGE, PA₈, PA₁₅, RFSH (refresh), and AY signals. These address signals select the RAS_{0,3}; RAS_{1,2}; RAS_{4,5} and RAS_{6,7} signals. The ROM 132 is programmed to implement the following four equations.

$$PRAS0,3 = \overline{AY} \cdot (\overline{DHIREs} + RFSH) + (ABK4 \cdot (Z Page \cdot \overline{PA8}) + ABK1 \cdot ABK2 \cdot ABK3) \cdot AY \quad (6)$$

$$PRAS1,2 = \overline{AY} \cdot (DHIREs + RFSH) + AY \cdot (\overline{ABK1} \cdot \overline{ABK2} \cdot \overline{ABK3} \cdot (ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot \overline{PA15}) + ABK1 \cdot ABK2 \cdot ABK3) + AY \cdot \overline{ABK3} \cdot (\overline{ABK1} \cdot ABK2 \cdot ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot PA15 + ABK1 \cdot ABK2 \cdot (ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot \overline{PA15})) \quad (7)$$

$$PRAS4,5 = RFSH \cdot \overline{AY} + AY \cdot \overline{ABK2} \cdot \overline{ABK3} \cdot (\overline{ABK1} \cdot ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot \overline{PA15}) \quad (8)$$

13

-continued

$$PA15 + ABK1 \cdot (ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot \overline{PA15})$$

$$PRAS6, 7 = RFSH \cdot \overline{AY} + AY \cdot \overline{ABK3} \cdot (ABK1 \cdot$$

$$\overline{ABK2} \cdot ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot PA15 + \overline{ABK1} \cdot$$

$$ABK2 \cdot (ABK4 \cdot (ZPAGE \cdot \overline{PA8}) \cdot \overline{PA15})$$

Thus, the bank switching signals (along with the other input signals to ROM 132) select predetermined rows in memory in conjunction with the CAS signals.

The output signals of the ROM 132 are coupled through the NAND gates 142, 143, 144 and 145 to the memory. The other input terminals of these gates receive the RAS timing signal. In this manner, the output signals of the ROM 132 are clocked through the gates 142 through 145 to provide the RAS signals shown in FIGS. 5 and 6.

An important feature to the presently described computer is provided by the circuitry shown within the dotted line 146. The AND gate 148 receives, at its input terminals, the DA7, A₁₂, and C₃ signals. The NOR gate 149 receives the zero page and A₁₅ signal. The output of gate 149 provides one input to the gate 148 and also one input to the AND gate 150. The output of gate 148 provides another input signal to gate 150 and this signal (line 153) is one of the two control signals coupled to the multiplexer 130. The AND gates 150 and 151 also receive a SYNC signal and the φ₀ signal. The output of the gates 150 and 151 are coupled to a NOR gate 152 with the output of the gate 152 (line 154) coupled to the other control terminal of the multiplexer 130.

The gates 150, 151 and 152 effectively form a clock for multiplexer/register 130 (multiplexer 130 is a commercial part, Part No. 399, which effectively is a register/multiplexer). This selects the lower four input lines to the multiplexer 130. However, because of the synchronization signal applied to gate 151, the multiplexer 130 selects the bank switching signals each time an OP code is fetched by the CPU.

To understand the operation of the circuit shown within the dotted line 146 it should be recalled that the memory of FIG. 6 provides a 16-bit output. As mentioned, during certain display modes, 16-bits/msec. are needed for display purposes. In nondisplay modes, only 8-bits are required, particularly for interaction with the CPU. When the memory is addressed by the CPU during the indirect addressing modes the data on the A bus is not ordinarily used. However, with the circuitry shown within the dotted line 146, this otherwise "unused" data is put to use to provide the equivalent of the bank switching signals through multiplexer 130.

Whenever the CPU selects a predetermined range of addresses, the multiplexer 130 selects the equivalent of the bank switching signals from the A bus provided DA7 is high. (This occurs when addressing as zero page the address space -1800 through 1FFF.) Once the signal on line 153 is high it is latched through gates 150, 151 and 152 causing the multiplexer 130 to select the four bits from the A bus (assuming the timing signals are high). Even if the next reference from the CPU is not to this special address range, the multiplexer 130 nonetheless remains latched with the four bits from the data bus. Once the SYN pulse drops, however, which is an indication that an OP code is being fetched, the signal on

14

line 154 rises in potential, causing the multiplexer to switch back to the bank switching signals.

Effectively, what occurs is that when the CPU selects this special address range, (and provided DA7 is high) the bits DA0 through DA3 which are stored in memory, cause a remapping, that is, the address from the CPU accesses a different part of the memory. With the fetching of each OP code, the mapping automatically returns to the bank switching signals. Importantly, the remapping, which occurs is controlled by the bits stored in the RAM (DA₀ through DA₃). Thus, with the remapping information stored in RAM, toggling can occur between different portions of the memory without requiring bank switching signals, or the like from the CPU. This enhances the CPU's performance since CPU time is not used for remapping. Additionally, it provides an easy tool for programming.

For some program languages it is desirable to separate data and the program into separate portions of the memory. For example, the 128K memory can be divided into two 64K memories, one for program and one for data. Switching can occur between these memory portions without the generation of bank switching signals by the CPU with the above described circuit. This arrangement is particularly useful when using the Pascal program language.

DISPLAY SUBSYSTEM

The display subsystem 48 of FIG. 1 receives data from the A bus and B bus and converts the data into video signals which may be used for displaying alphanumeric characters or other images on a standard raster scanned cathode ray tube display. The display subsystem 48 specifically generates on line 197, a standard NTSC color video signal and a video black and white video signal on line 198 (FIG. 8). This display subsystem, in addition to other inputs, receives a synchronization signal, and several clocking signals. For sake of simplicity, the standard color reference signal of 3.579545 MHz is shown as C3.5M. Twice this frequency and four times this frequency are shown as C7M and C14M, respectively.

Before describing the details of the display subsystem 48, a discussion of a prior art display system will be helpful in understanding the present display subsystem. In U.S. Pat. No. 4,136,359, a video display system is described which is implemented in a commercially available computer, Apple-II, sold by Apple Computer, Inc., of Cupertino, Calif. In this system, 4-bit digital words are shifted in parallel into a shift register. These words are then circulated in the shift register at 14 MHz to define a waveform having components at 3.5 MHz. Referring to FIG. 9, line 206, assume that the digital word 0001 is placed in the shift register and circulated at a rate of 14 MHz. The resultant signal which has a component of 3.5 MHz is shown on line 206. The phase relationship of this component to the 3.5 MHz reference signal determines the color of the resultant video signal. This relationship is changed by changing the 4-bit word placed in the shift register. As explained in the above-referenced patent, if the signal 1000 is placed in the register and circulated, the resultant phase relationship of the 3.5 MHz component results in the color brown, this signal is shown on line 208. With this prior art technique, the luminance was determined by the DC component of the signals such as shown on lines 206 and 208.

4,533,909

15

The display subsystem 48 of FIG. 1 also uses 4-bit words to generate the various color signals in a manner somewhat similar to the above-described system. Referring to FIG. 8 4-bit words representative of colors (16 possible colors) are coupled to the bus 180 (The generation of these words shall be described in detail in conjunction with FIG. 7.) Instead of using a shift register which circulates the 4-bit word, the same result is achieved by using a multiplexer 205 which sequentially selects each of the lines of the bus 180. The signals on bus 180 also provide a luminance signal and a black and white video signal with a gray scale.

The 4 lines of the bus 180 are coupled to multiplexer 205; this multiplexer also receives the C7M and the C3M/ timing signals (again, Commercial Part No. 135 is used with the pin connections shown in FIG. 8). These two timing signals cause each of the four lines to be sequentially selected and coupled to line 191. (Note that the order in which each of the lines of the bus 180 is selected does not change.)

In effect, the multiplexer operates to serialize the parallel signal from bus 180. Assume for sake of explanation that the digital signals on bus 180 are 1000 as indicated in FIG. 8. The signal on line 191 will then be 10001000 The output of the multiplexer 205 coupled to the input of the inverter 204 also receives in a sequential order, the signals from bus 180, however, in a different order. For the example shown, the input to inverter 204 is 00100010 After inversion, this results in the signal 11011101 . . . on line 192. Effectively, the signals on lines 191 and 192 are added by resistors 199 and 200. The resultant waveform is an AC signal (no DC component) shown in FIG. 9 on line 209. Thus, with the described circuit, a chroma signal is generated, having a predetermined phase relationship to the 5.5 MHz color reference signal. This phase relationship which is varied by changing the signals on bus 180 determines the color of the video signal on line 197.

In the prior art display discussed above, the DC component of the color signal determines the luminance. In the present invention, the signals on bus 180 are coupled to the base of transistor 195, consists of an AC signal from resistors 199 and 200, and the luminance level also determined by the signals on bus 180. These inputs to transistor 195, along with the C3.5M signal, generate a NTSC color signal on line 197 of improved quality when compared to the discussed prior art system.

In some cases, the signals on bus 180 are all binary ones or all binary zeros. When this occurs, there is no AC component from resistors 199 and 200 (no color signal) and the resultant signal on line 197 is either "black" or "white".

The lines of bus 180 are also coupled through resistors to the base of a transistor 196. Each of these resistors have a different value to provide a "weighting" to the binary signal. This weighting is used for non-color displays to provide "gray" shades as opposed to having a display with only black and white. The binary signals on bus 180 drive the transistor 196 to provide a video signal on line 198. RGB is generated with weighted sums of these same five signals.

Referring now to FIG. 7, data from memory is coupled from the A bus and B bus to registers 159 and 158, respectively. These registers are clocked by the 1 MHz clocking signal and its complement, thus permitting the sequential transfer of 8-bit words every 0.5 msec. As will be described, in some display modes the data is

16

transferred at the 2 MHz rate, and in other display modes, at a 1 MHz rate.

The registers 158 and 159 are coupled to an 8 line display bus 160. This display bus transfers data to registers 164 and 173, and also addresses to a memory 162. The registers 164 and 173 and memory 162 are enabled during specific display modes as will be apparent.

The character memory 162, in the presently preferred embodiment, is a random-access memory which stores patterns representative of alpha-numeric characters. Each time the computer is powered up, the character information is transferred from the ROM 50 into the character memory 162 during an initialization period. During character display modes, the signals from the display bus 160 are addresses, identifying particular alpha-numeric characters stored within the character memory 160. The vertical counter signals V_A, V_B, and V_C (previously discussed in conjunction with adder 121 of FIG. 4) identify the particular line in each character which is to be displayed. Thus, the generation of the digital signals representative of each of the characters occurs in an ordinary manner. The 7-bit signal representative of each line of each character (memory output) is coupled to the shift register 167. Through timing signals not shown, either the register 164 or the character memory 162 is selected to allow the shift register 167 to receive either data directly from the A bus or B bus, or alpha-numeric character information from the memory 162.

The 7-bits of information from either memory 162 or register 164 are serialized by the shift register 167 either at a 7 MHz rate or 14 MHz rate, depending upon the display mode. The serialized data is coupled by line 185 to the multiplexer 169, pins 1 and 4. The inverse of this data is also coupled to multiplexer 169, pin 3. Line 185 is also coupled as one input to the multiplexer 166 and to the register 170 (input 1).

The output 1 of register 170 (line 186) is coupled to the multiplexer 169, pin 1; to register 170 (input 2); and to multiplexer 166. Output 2 of register 170 (line 187) is coupled to input 3 of register 170 and also to multiplexer 166. Output 3 of register 170 (line 187) provides a third input to the multiplexer 166. Input 4 of the register 170 receives the output of the multiplexer 169 (line 189). Output 4 of register 170 (line 190) provides one control signal for the multiplexer 171.

The multiplexer 171 selects either the four lines of bus 183 or the four lines of bus 184. The output of multiplexer 171, bus 180, provides the 4-bit signal discussed in conjunction with FIG. 8. During one of the high resolution display modes (AHIRES), the multiplexer 171 is controlled by a timing signal from the output of the gate 178.

The multiplexer 166 selects either the lines of bus 181 or bus 182. The output of this multiplexer provides the signals for the bus 184. In all but the AHIRES display mode, multiplexer 166 selects bus 181. Thus, typically, the multiplexer 171 receives the signals from bus 174.

For purposes of description above, and also for purposes of explaining for some of the display modes below a simplifying assumption has been made. The signals coupled to the bus 180 by multiplexer 171, for most modes, are controlled by the serialized signal on line 190. This serialized signal is in synchronization with the C7M or C14M clocking signals. The multiplexer 205 of FIG. 8, which as described above, does the "spinning" for the parallel digital signal on bus 180, operates in synchronization with the multiplexer 171. In the descrip-

4,533,909

17

tion above, and except when otherwise noted below, it is assumed that, by way of example, if the multiplexer 171 is coupling all binary ones and zeros onto bus 180, the signal on line 191 will be either ones or zeros. Also for this condition the signal on line 192 will be all binary zeros or ones, and thus, no AC signal is generated at the base of transistor 195. However, as actually implemented, there is a "phase" difference between the clocking of the multiplexer 171 when compared to the sampling of the signals from bus 180 by the multiplexer 205. This results in a first constant AC signal on the gate of transistor 195 even when it appears that all binary ones are on bus 180, and a second constant AC signal when all binary zeros are on the bus 180. Thus, in this specification, when it states that "black" or "white" signals are being generated, instead, as currently implemented, two constant colors are generated on a color display. Where a true black and white is desired, color suppression is introduced such as through the color burst signal.

The circuit of FIG. 7, along with the circuit of FIG. 8, provides the capability for several distinct display modes. The first of these modes provides a display consisting of 40 characters (or spaces) per horizontal line. This requires a data rate of 8-bits/MHz or half the data rate the memory is capable of delivering. In this mode, data is loaded from the A bus during every other 0.5 μ sec period. (B bus is not used during this mode.) This data addresses the character memory 162, and along with the signals V_A , V_B and V_C , provides the appropriate character line (7-bits) to the shift register 167. During this mode, registers 164 and 173 are disabled. The shift register 167 for this mode shifts the data at a data rate of 7 MHz (note $\overline{CH80}$ is high, allowing the 7 MHz signal from gate 175 to control the shift register 167). Each 7-bit signal is shifted serially onto line 185 and then to line 189 since multiplexer 169 selects pin 4. The data is shifted through the register 170 onto line 190. The serial binary signal on line 190 causes the selection of buses 183 or 184

The four lines of bus 183 during this mode are coupled to +V (register 173 is disabled); therefore the selection of bus 184 provides four binary ones. The selection of bus 184 provides four binary zeros through bus 181. Thus, the serial binary signal on line 190 provides either all binary ones or all binary zeros to bus 180. As discussed the circuit of FIG. 8 will provide a black and white display with 40 characters per line.

If the inverse and flashing timing means 172 is selected, each time the shift register 167 is loaded, multiplexer 169 shifts between pins 3 and 4. This causes the characters to change from white characters on a black background to black characters on a white background, and so on.

During the 80 character per line display mode, the registers 158 and 159 are each loaded during sequential 0.5 μ sec periods (this utilizes the 2 MHz cycle rate previously discussed). The shift register 167 shifts the character data from memory 162 at a 14 MHz rate. The serialized data at the 14 MHz rate is shifted through the register 170 and again controls the multiplexer 171 as previously described. (Note that register 170 is always clocked at the 14 MHz rate.) Flashing again can be obtained as previously discussed.

In another alpha-numeric character display mode, the background of each character may be in one color and the character itself (foreground) in another color. This mode provides 40 characters per line. The character

18

identification (address for RAM 162), is furnished on the A bus to register 159 at a frequency of 1 MHz. The color information (background color and foreground color) is furnished on the B bus as two 4-bit words to register 158. In the manner previously described, the address from register 159 selects the appropriate character from memory 162 and provides this information to shift register 167. The color information from the B bus is transferred to register 173. For purposes of explanation, assume that the 4-bits identifying the color red for the background are on bus 184 (from register 173 and multiplexer 166) and that 4-bits representing the color blue for the foreground are on bus 183. (Note that when register 173 is enabled, the signals from the register override the binary ones and zeros which otherwise appear on the lines of bus 174.) The serial binary signal representative of the character itself on line 190, selects either the color blue from bus 183 for the character itself or the color red from bus 184 for the background. The digital signals representative of these colors are transferred to bus 180 and provide the color data to the circuit of FIG. 8. For black and white displays, a "gray" scale is provided through the weighting circuit associated with transistor 196 of FIG. 8. Again, the multiplexer 169 may, through the timing means 172, alternate between the signal of line 185 and its inverse, which will have the effect of interchanging the foreground and background colors.

During the high resolution graphics modes, the character memory 162 is not used, but rather, data from the memory directly provides pattern information for display. This requires more mapping of data from within the main memory since new data is required for each line of the display. (Note that when characters are displayed, the character memory 162 provides the different signals required for the 8 lines of each character row. During these high resolution modes, the register 164 is enabled and the character memory 162 is disabled. Thus, the data from the A bus and B bus is shifted into the shift register 167. In these modes, the "HRES" signal to multiplexer 169 causes this multiplexer to select between pins 1 and 2. Pin 2 provides the signal directly from the shift register 167 while the signal on pin 1 is effectively the signal on line 185 delayed by one period of the C14M signal. This delay occurs through the register 170 from input 2 to output 2 since register 170 is clocked at C14M.

During a first graphics mode, data from the display bus 160 is loaded into shift register 167 at the rate of 7-bits/MHz. The data is serialized on line 185 and in the manner previously described for displaying characters, controls the selection of all binary ones and all binary zeros through the multiplexer 171. Note, as mentioned before, in the presently preferred embodiment, unless color suppression is used, this will not result in a black and white display, but rather a two-color display. If a high bit is present on line 140 of the display bus, the inverse and flashing timing means 172 causes the multiplexer 169 to alternate between pins 1 and 2. This switching occurs at a 1 MHz rate and provides a phase shift for every other 7-bits of data coupled to the multiplexer 171 on line 190. This results in an additional color being generated on the display for every other 7-bits of data.

For the above-described graphics modes when shift register 161 is shifting at a 7 MHz rate, 8-bits may be coupled to the bus 160 during each period. Specifically, as in the case of the differing background and fore-

4,533,909

19

ground colors for the 40 character per line display mode, two 4-bit color words are shifted into register 173 at a rate of 1 MHz. Then, the multiplexer 171 selects between two predetermined colors on buses 183 and 184. Note these colors can be changed at a 1 MHz rate. 5

In an additional color mode identified as "AHIRES", multiplexer 171 operates under the control of gates 176, 177 and 178. In effect, multiplexer 171 selects bus 184 and latches the signals on this bus every four cycles of the C14M clock. Data is shifted into the shift register 167 from the A bus and B bus every 0.5µ sec the register 167 operates under the control of the C14M signal. Each data bit on line 185 is shifted first to line 186, then to line 187 and finally to line 188. These lines are coupled to the multiplexer 171 through multiplexer 166 which selects bus 182 since AHIRES is high. In effect, what occurs is that 4-bit color words are serialized onto line 185 and then brought back into parallel on bus 182. Since multiplexer 171 latches the signals on bus 184 every four cycles of the C14M signal, a new color word is generated at a 3.5 MHz rate on the bus 180. The resultant display is 140 by 192 colored blocks wherein each block can be any one of 16 colors. 15

In the last display mode, typically used with color suppression, data is shifted into the shift register 167 from the display bus at the rate of 14-bits/MHz. The data is serialized onto line 185 and controls the selection of either all binary ones or all zeros through multiplexer 171. This provides the highest resolution graphics display for the system. 20

Thus, a microcomputer with video display capability has been described. The computer is fabricated from commercially available parts and provides high utilization of these parts. Numerous existing programs including many of those which operate on the Apple-II computer, may be employed in the above-described computer. 25

I claim:

1. In a digitally controlled, raster scanned, video display for use with a microcomputer, or the like, which display provides color images in response to chroma signals having predetermined phase relationships to a reference signal of frequency (f), a circuit for providing a digitally controlled chroma signal comprising: 30

digital word generation means for generating predetermined digital signals;

serializing means coupled to said generation means for repeating said word in a serial form at a prede-

50

55

60

65

20

terminated frequency so as to provide frequency components at said frequency f;

converting means, coupled to said serializing means for converting outputs from said serializing means to an AC signal;

whereby a video chroma signal is generated.

2. The circuit defined by claim 1 including additional circuit means coupled to said digital word generation means for providing a DC luminance signal.

3. The circuit defined by claim 1 wherein said digital words are coupled to a resistive weighting network for providing a gray scale video signal.

4. The circuit defined by claim 1 wherein said digital words are 4-bit words and wherein said predetermined frequency is equal to 4f.

5. The circuit defined by claim 4 wherein said serializing means comprises a multiplexer which is controlled in synchronization with said frequency f.

6. The circuit defined by claim 5 wherein said converting means includes an inverter coupled to an output of said multiplexer.

7. The circuit defined by claim 6 including additional circuit means coupled to said digital word generation means for providing a DC luminance signal.

8. The circuit defined by claim 1 wherein said digital word generation means comprises:

- a source of digital data for controlling said display;
- a first register coupled to receive data from said source of data;

a multiplexer for selecting between two buses, the output of said multiplexer coupled to said serializing means, said buses coupled to said first register, a shift register coupled to receive data from said source of data, said shift register providing a serialized digital signal for controlling said multiplexer. 35

9. The circuit defined by claim 8 including a character memory for storing data representative of alpha numeric characters, said memory coupled to receive address from said source of data, the output of said memory coupled to said shift register.

10. The circuit defined by claim 9 wherein when said first register is disabled, one of said two buses is clamped to provide all binary ones, and the other of said buses provides all binary zeros.

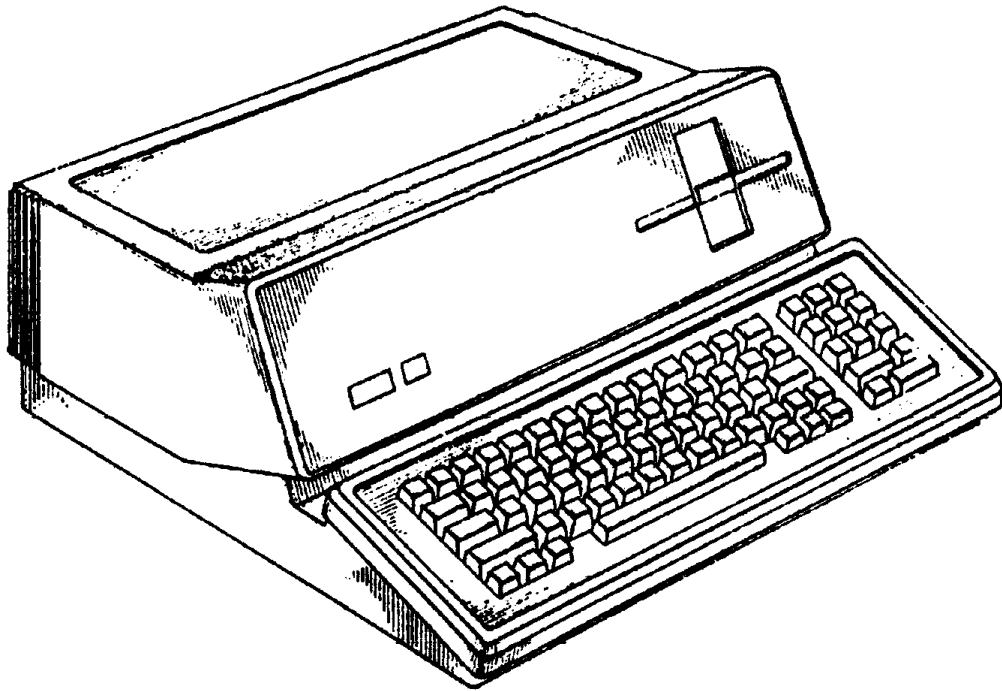
11. The circuit defined by claim 10 wherein said shift register is controlled by a plurality of clocking signals, all of which are synchronized with said frequency f.

* * * * *

FINIS



Apple /// Computer Information



APPLE /// ROM INFO

ADDED BY DAVID T CRAIG • 2006

*Apple /// ROM Information***APPLE /// ROM INFORMATION***ROM
REVISION
#0*

by
David Craig
736 Edgewater, Wichita, Kansas 67230

1986

This document describes the Apple /// microcomputer ROM organization. The ROM listing used was from Apple Computer's patent (# 4,383,296) of May 10, 1983 as assigned to Wendell B. Sander. The ROM listing appears to be from December 20, 1979.

The ROM occupies 4K bytes of memory in the address range \$F000—\$FFFF. This ROM is used by the Apple /// at system power-up to test various hardware components, initialize the character generator bitmap, and boot SOS (Sophisticated Operating System) from the Apple ///'s internal floppy diskette drive.

The ROM is organized as follows (routine names in lowercase were created by me since the source code did not contain a name at the particular location):

Addresses	Name	Description
F000-F124	REGWTS	Read/Write a disk track and sector
F125-F12A	SETTRK	Set slot dependent track location
F12B-F13D	CHKDRV	Check if disk motor is stopped
F13E-F147	DRVINDX	Get index to drive number
F148-F1B9	READ16	Read disk sector
F1BA-F1BC	GOSERV	Interrupt service vector
F1BD-F218	ROADR16	Read disk sector address field
F219-F2B2	WRITE16	Write disk sector
F2B3-F2BB	SERVICE	Interrupt servicer
F2BC-F2C5	WNIBL9	Write 7-bit nibbles to disk
F2C6-F310	PRENIB16	Pre-nibblize disk sector data
F311-F354	POSTNIB16	Post-nibblize disk sector data
F355-F395	NIBL	6-bit to 7-bit nibble conversion table
F396-F3FF	DNIBL	7-bit to 6-bit denibbleize conversion table
F400-F455	SEEK	Disk track seeker
F456-F466	MSWAIT	100 microsecond delayer
F467-F46F	ONTABLE	Disk phase ON time table (in 100 microsecs)
F470-F478	OFFTABLE	Disk phase OFF time table (in 100 microsecs)
F479-F49F	BLOCKIO	Read/write a disk block (2 sectors)

Apple /// ROM Information

F4A0-F4A7	SECTABL	Block to sector conversion table
F4A8-F4C4	ANALOG	Joystick read routine
F4C5-F4CC	RAMTBL	RAM test bytes
F4CD-F4ED	CHPG	Hardware component phrases (eg "RAM", "ROM",...)
F4EE-F523	DIAGN	ROM system power-up entry (calls RECON [F689])
F524-F531	NXBYT	Test RAM page 0 (Zero Page)
F532-F545	CNTWR	Test RAM page 1 (Stack Page)
F546-F574	memsize	Size the RAM
F575-F589	ERRLP	Display screen error line ("DIAGNOSTICS")
F58A-F5E6	zpgstktst	Test RAM zero page & stack page
F5E7-F60C	ROMTST	Test ROM hardware
F60D-F63D	VIATST	Test VIA hardware
F63E-F652	ACIA	Test ACIA hardware
F653-F67A	ATD	Test A/D hardware
F67B-F688	KEYPLUG	Test keyboard plugin
F689-F6C1	RECON	Reconfigure system (tests for Apple-1 key)
F6C2-F6E5	SEX	System exerciser
F5E6-F737	USRENTY	Main RAM tester
F738-F747	STRWT	Error message string writer
F748-F77A	RAM	Determine size of RAM
F77B-F783	MESSERR	Display error message
F784-F7A0	RAMSET	Setup RAM
F7A1-F7C8	PTRINC	Increment extended addressing pointer
F7C9-F7F6	RAMERR	RAM error handler
F7F7-F7FF	RAMWT	RAM write
F800-F900	RET1	Nested RTS 'table' routine
F901-F92B	ENTRY	SARA Monitor entry point
F92C-F95D	GETNUM	Get number from user
F92E-F96B	TOSUB	Execute Monitor command
F96C-F97B	CMDTAB	Monitor command code table
F97C-F98B	CMDVEC	Monitor command vector table (byte-long entries)
F98C-F9AB	NXTA4	Increment 2 byte pointer
F9AC-F9C1	PRBYTE	Output a byte to screen
F9C2-F9C8	PRBYCOL	Output a byte followed by a colon
F9C9-F9D3	TST8OWID	Test for 80-column screen width
F9D4-F9DE	A1PC	Test for new P.C.
F9DF-FA06	ASCII1	Store user ASCII string into memory
FA07-FA25	ASCII	Fetch ASCII character from keyboard
FA26-FA2B	CRMON	Dump line of hexadecimal bytes due to user CR
FA2C-FA3A	MOVE	Move bytes around in memory
FA3B-FA51	VRFY	Verify memory byte range
FA52-FA77	MISMATCH	Output verify mismatch data line
FA78-FA7A	USER	User control vector
FA7B-FA82	JUMP	Transfer control to user routine
FAB3-FA90	RWERROR	Output error number
FA91-FA99	DEST	Copy source pointer to destination pointer
FA9A-FAB7	SEP	Test for separator character in input line
FAB8-FABF	SETMODE	Setup user mode
FAC0-FAE8	READ	Handle Monitor READ disk block command
FAE9-FB20	DUMPB	Output line of memory bytes
FB21-FB48	DUMPASC	Output line of memory bytes as ASCII
FB49-FB4E	COL80	Setup 80-column display mode
FB4F-FB92	COL40	Setup 40-column display mode

ENTRY POINT

Apple /// ROM Information

FB93-FBA3	CONTROL	Handle user control character input
FBA4-FBB6	CURUP	Handle cursor up motion
FBB7-FBC8	CURIGHT	Handle cursor right motion
FBC9-FBD4	DURDOWN	Handle cursor down motion
FBD5-FBD8	LSTBACK	Handle backspace motion
FBD9-FBF1	CURLEFT	Handle cursor left motion
FBF2-FC04	COUT2	Output character to screen
FC05-FC24	BASCALC1	Compute character base address for screen output
FC25-FC32	COUT	Output character to current output device
FC33-FC35	COUT1	Character output vector
FC36-FC51	TSTBELL	Handle BELL character output (beep speaker)
FC52-FC5A	LNFD	Handle LINE FEED character output
FC5B-FC9C	SCROLL	Scroll screen lines
FC9D-FCAC	DISPLAY	Display character on 40-column screen
FCAD-FCBA	DSPL80	Display character on 80-column screen
FCBB-FCD4	NOTCR	Handle non-control character output
FCD5-FD0B	GETLNZ	Read user ASCII line from keyboard
FDOC-FD0E	RDKEY	Read keyboard key input vector
FD0F-FD47	KEYIN	Read raw keyboard key
FD48-FD5F	ESC3	Handle ESC character cursor motion
FD60-FD76	RDCHAR	Read keyboard character
FD77-FD7E	GOESC	ESC key cursor motion handler
FD7F-FD87	ESCVECT	ESC key editing command key code table
FD88-FD97	PICK	Read character from current cursor location
FD98-FDC5	CLDSTART	Cold boot system (initialize ROM globals)
FDC6-FEAD	GENENTR	Load character generator RAM with bitmap
FEAE-FEC4	VRETRCE	Wait/poll for CRT vertical retrace
FEC5-FFB3	CHRSET	Character generator character bitmap table
FFB4-FFB7	HOOKS	Output/Input vectors
FFB8-FFB8	VBOUNDS	Screen dimension bounds (0, 80, 0, 24)
FFBC-FFBF	NMIIRQ	NMI request vector (JMP RECON [F689] RTI)
FFC0-FFEF	applecwrite	Apple Computer, Inc. 1980 copyright phrase
FFF0-FFF9	ESCTABL	ESC character table
FFFA-FFFB	NMI	NMI vector [FFCA]
FFFC-FFFD	RESET	RESET vector [F4EE] (Power-up Diagnostics)
FFFE-FFFF	IRG	IRG vector [FFCD]

---- *The End* ----

Assembler: Apple /// Pascal TLA 6502 Assembler
(converted to TLA format most likely by Scott Stinson)

41 pages

Source Code Listing

for

Apple ///

Sara ROM

\$F000 - \$FFFF | 4KB

REVISION 1

(see A/// patent for Rev 0 ROM)

David T. Craig
736 Edgewater
Wichita, Kansas 67230

Copyrighted Jan. 1980

Source Code Listing

for

Apple ///

ROM - Disk I/O

David T. Craig
736 Edgewater
Wichita, Kansas 67230

```

0000| ;*****
0000| ; APPLE /// ROM - DISK I/O ROUTINES
0000| ; COPYRIGHT 1979 BY APPLE COMPUTER, INC.
0000| ;*****
0000|
0000| .ABSOLUTE
0000| .PROC DISKIO
0000| .ORG 0F000
0000|
F000| ;*****
F000| ; CRITICAL TIMING *
F000| ; REQUIRES PAGE BOUND *
F000| ; CONSIDERATIONS FOR *
F000| ; CODE AND DATA *
F000| ; -----CODE----- *
F000| ; VIRTUALLY THE ENTIRE *
F000| ; 'WRITE' ROUTINE *
F000| ; MUST NOT CROSS *
F000| ; PAGE BOUNDARIES *
F000| ; CRITICAL BRANCHES IN *
F000| ; THE 'WRITE', 'READ', *
F000| ; AND 'READ ADR' SUBRS *
F000| ; WHICH MUST NOT CROSS *
F000| ; PAGE BOUNDARIES ARE *
F000| ; NOTED IN COMMENTS *
F000| ;*****
F000| ;
F000| ; EQUATES *
F000| ;
F000| 0200 NBUF1 .EQU 0200
F000| 0302 NBUF2 .EQU 0302 ; (ZERO PAGE AT $300)
F000| ;
F000| 0080 HRDERRS .EQU 80
F000| 00E0 DVMOT .EQU 0E0
F000| ;
F000| 0081 IBSLOT .EQU 81
F000| 0082 IBDRVN .EQU IBSLOT+1
F000| 0083 IBTRK .EQU IBSLOT+2
F000| 0084 IBSECT .EQU IBSLOT+3
F000| 0085 IBBUFP .EQU IBSLOT+4 ; & 5
F000| 0087 IBCMD .EQU IBSLOT+6
F000| 0088 IBSTAT .EQU IBSLOT+7
F000| 0089 IBSMOD .EQU IBSLOT+8
F000| 0089 CSUM .EQU IBSMOD ; USED ALSO FOR ADDRESS HEADER CKSUM
F000| 008A IOBPDN .EQU IBSLOT+9
F000| 008B IMASK .EQU IBSLOT+0A
F000| 008C CURTRK .EQU IBSLOT+0B
F000| 0085 DRVOTRK .EQU CURTRK-7
F000| ; SLOT 4, DRIVE 1
F000| ; SLOT 4, DRIVE 2
F000| ; SLOT 5, DRIVE 1
F000| ; SLOT 5, DRIVE 2
F000| ; SLOT 6, DRIVE 1
F000| ; SLOT 6, DRIVE 2
F000| 0093 RETRYCNT .EQU IBSLOT+12
F000| 0094 SEEKCNT .EQU IBSLOT+13
F000| 009B BUF .EQU IBSLOT+1A
F000| 009F ENVTEMP .EQU IBSLOT+1E
F000| ; IBSLOT+$1F NOT USED
F000| ;*****
F000| ;
F000| ; ----READADR---- *
F000| ;*****
F000| ;
F000| 0095 COUNT .EQU IBSLOT+14 ; 'MUST FIND' COUNT.
F000| 0095 LAST .EQU IBSLOT+14 ; 'ODD BIT' NIBLS.
F000| 0096 CKSUM .EQU IBSLOT+15 ; CHECKSUM BYTE.
F000| 0097 CSSTV .EQU IBSLOT+16 ; FOUR BYTES
F000| ; CHECKSUM, SECTOR, TRACK, AND VOLUME.
F000| ;*****
F000| ;
F000| ; ----WRITE---- *
F000| ;*****
F000| ;
F000| ; USES ALL NBUFS *
F000| ; AND 32-BYTE *
F000| ; DATA TABLE 'NIBL' *
F000| ;*****
F000| ;
F000| ;*****
F000| ;
F000| ; ----READ---- *
F000| ;*****
F000| ;
F000| ; USES ALL NBUFS *
F000| ; USES LAST 54 BYTES *
F000| ; OF A CODE PAGE FOR *

```



```

F000|          ;   SIGNIFICANT BYTES   *
F000|          ;   OF DNIBL TABLE.  *
F000|          ;   *
F000|          ; *****
F000|          ;
F000|          ; *****
F000|          ;
F000|          ;   ----SEEK----   *
F000|          ;   *
F000|          ; *****
F000|          ;
F000| 0095   TRKCNT   .EQU   COUNT           ; HALFTRACKS MOVED COUNT.
F000| 009D   PRIOR   .EQU   IBSLOT+1C
F000| 009E   TRKN    .EQU   IBSLOT+1D
F000|          ;
F000|          ; *****
F000|          ;
F000|          ;   ----MSWAIT----   *
F000|          ;   *
F000|          ; *****
F000|          ;
F000| 0099   MONTIMEL .EQU   CSSTV+2         ; MOTOR-ON TIME
F000| 009A   MONTIMEH .EQU   MONTIMEL+1     ; COUNTERS.
F000|          ;
F000|          ; *****
F000|          ;
F000|          ;   DEVICE ADDRESS   *
F000|          ;   ASSIGNMENTS   *
F000|          ; *****
F000|          ;
F000| C080   PHASEOFF .EQU   0C080          ; STEPPER PHASE OFF.
F000| C081   PHASEON  .EQU   0C081          ; STEPPER PHASE ON.
F000| C08C   Q6L     .EQU   0C08C          ; Q7L,Q6L=READ
F000| C08D   Q6H     .EQU   0C08D          ; Q7L,Q6H=SENSE WPROT
F000| C08E   Q7L     .EQU   0C08E          ; Q7H,Q6L=WRITE
F000| C08F   Q7H     .EQU   0C08F          ; Q7H,Q6H=WRITE STORE
F000| FFEF   INTERRUPT .EQU   0FEF
F000| FDFD   ENVIRON  .EQU   0FDF
F000| 0080   ONEMEG  .EQU   80
F000| 007F   TWOMEG  .EQU   7F
F000|          ;
F000|          ; *****
F000|          ;
F000|          ; EQUATES FOR RWTS AND BLOCK
F000|          ; *****
F000|          ;
F000| C088   MOTOROFF .EQU   0C088
F000| C089   MOTORON  .EQU   0C089
F000| C08A   DRVOEN   .EQU   0C08A
F000| C08B   DRV1EN   .EQU   0C08B
F000| C081   PHASON   .EQU   0C081
F000| C080   PHSOFF   .EQU   0C080
F000| 0097   TEMP     .EQU   CSSTV          ; PUT ADDRESS INFO HERE
F000| 0097   CSUM1    .EQU   TEMP
F000| 0098   SECT     .EQU   CSUM1+1
F000| 0099   TRACK   .EQU   SECT+1
F000| 0099   TRKN1   .EQU   TRACK
F000| 009A   VOLUME  .EQU   TRACK+1
F000| 0083   IBRERR  .EQU   HRDERRS+3
F000| 0082   IBDERR  .EQU   HRDERRS+2
F000| 0081   IBWPER  .EQU   HRDERRS+1
F000| 0080   IBNODRV .EQU   HRDERRS
F000|          ;
F000|          ; *****
F000|          ;
F000|          ;   READ WRITE A   *
F000|          ;   TRACK AND SECTOR *
F000|          ; *****
F000|          ;
F000| A0 01   REGRWTS LDY   #01           ; RETRY COUNT
F002| A6 81   LDY   IBSLOT           ; GET SLOT # FOR THIS OPERATION
F004| 84 94   STY   SEEKCNT         ; ONLY ONE RECALIBRATE PER CALL
F006| A9 05   LDA   #005
F008| 85 8F   STA   08F
F00A| 08     PHP
F00B| 68     PLA
F00C| 6A     ROR   A
F00D| 6A     ROR   A           ; GET INTERRUPT FLAG INTO BIT 7
F00E| 6A     ROR   A
F00F| 6A     ROR   A
F010| 85 8B   STA   IMASK
F012| AD DFFF LDA   ENVIRON         ; PRESERVE ENVIRONMENT
F015| 85 9F   STA   ENVTEMP
F017| 20 2BF1 JSR   CHKDRV          ; SET ZERO FLAG IF MOTOR STOPPED
F01A| 08     PHP
F01B| A5 85   LDA   IBBUFF          ; MOVE OUT POINTER TO BUFFER INTO ZPAGE
F01D| 85 9B   STA   BUF
    
```

```

F01F| A5 86          LDA    IBBUFF+1
F021| 85 9C          STA    BUF+1
F023| A9 E0          LDA    #DVMOT
F025| 85 9A          STA    MONTIMEH
F027| A5 82          LDA    IBDRVN      ; DETERMINE DRIVE ONE OR TWO
F029| C5 8A          CMP    IOBPDN      ; SAME DRIVE USED BEFORE
F02B| 85 8A          STA    IOBPDN      ; SAVE IT FOR NEXT TIME
F02D| 08            PHP    ; KEEP RESULTS OF COMPARE
F02E| 6A            ROR    A           ; GET DRIVE NUMBER INTO CARRY
F02F| BD 89C0        LDA    MOTORON,X   ; TURN ON THE DRIVE
F032| 9001          BCC    DRIVSEL     ; BRANCH IF DRIVE 1 SELECTED
F034| E8            INX    ; SELECT DRIVE 2
F035| BD 8AC0        LDA    DRVOEN,X
F038| 20 4CF3        DRIVSEL JSR    SETIMEG    ; INSURE ONE MEGAHERTZ OPERATION
F03B| 28            PLP    ; WAS IT SAME DRIVE?
F03C| F00A          BEQ    OK
F03E| 28            PLP    ; MUST INDICATE DRIVE OFF BY SETTING ZERO FLAG
F03F| A0 07          LDY    #07         ; DELAY 150 MS BEFORE STEPPING
F041| 20 56F4        DRVWAIT JSR    MSWAIT     ; (ON RETURN A=0)
F044| 88            DEY
F045| D0FA          BNE    DRVWAIT
F047| 08            PHP    ; NOW ZERO FLAG SET
F048| A5 83          OK      LDA    IBTRK      ; GET DESTINATION TRACK
F04A| A6 81          LDA    IBSLOT     ; RESTORE PROPER X (SLOT*16)
F04C| 20 04F1        JSR    MYSEEK     ; AND GO TO IT
F04F| 28            ; NOW AT THE DESIRED TRACK WAS THE MOTOR ON TO START WITH?
F050| D017          PLP    ; WAS MOTOR ON?
F052|              BNE    TRYTRK   ; IF SO, DON'T DELAY, GET IT TODAY!
F052|              ;
F052|              ; MOTOR WAS OFF, WAIT FOR IT TO SPEED UP
F052|              ;
F052| A0 12          MOTOF  LDY    #12    ; WAIT EXACTLY 100 US FOR EACH COUNT
F054| 88            CONWAIT DEY    ; IN MONTIME
F055| D0FD          BNE    CONWAIT
F057| E6 99          INC    MONTIMEH   ; COUNT UP TO 0000
F059| D0F7          BNE    MOTOF
F05B| E6 9A          INC    MONTIMEH
F05D| 30F3          BMI    MOTOF
F05F|              ;
F05F|              ; *****
F05F|              ; MOTOR SHOULD BE UP TO SPEED
F05F|              ; IF IT STILL LOOKS STOPPED THEN
F05F|              ; THE DRIVE IS NOT PRESENT.
F05F|              ; *****
F05F|              ;
F05F| 20 2BF1        JSR    CHKDRV     ; IS DRIVE PRESENT?
F062| D005          BNE    TRYTRK   ; YES, CONTINUE
F064| A9 80          NODRIVER LDA #IBNODRV ; NO, GET TELL EM NO DRIVE
F066| 4C EAF0        JMP    HNDLERR
F069|              ;
F069|              ; NOW CHECK IF IT IS NOT THE FORMAT DISK COMMAND,
F069|              ; LOCATE THE CORRECT SECTOR FOR THIS OPERATION
F069|              ;
F069| A5 87          TRYTRK LDA    IBCMD   ; GET COMMAND CODE #
F06B| F076          BEQ    ALLDONE   ; IF NULL COMMAND, GO HOME TO BED
F06D| C9 03          CMP    #03      ; COMMAND IN RANGE?
F06F| B072          BCS    ALLDONE   ; NO, DO NOTHING!
F071| 6A            ROR    A           ; SET CARRY=1 FOR READ, 0 FOR WRITE
F072| B00B          BCS    TRYTRK2  ; MUST PRENIBBLIZE FOR WRITE
F074| AD DFFF        AD     DFFF
F077| 29 7F          AND    #TWOMEG   ; SHIFT TO HIGH SPEED!
F079| 8D DFFF        STA    ENVIRON
F07C| 20 C4F2        JSR    PRENIB16
F07F| A0 7F          TRYTRK2 LDY    #7F    ; ONLY 127 RETRIES OF ANY KIND
F081| 84 93          STY    RETRYCNT
F083| A6 81          TRYADR  LDX    IBSLOT ; GET SLOT NUM INTO X-REG
F085| 20 B9F1        JSR    RDADR16   ; READ NEXT ADDRESS FIELD
F088| 9022          BCC    RDRIGHT  ; IF READ IS RIGHT, HURRAH!
F08A| 20 AAF1        TRYADR2 JSR    CHKINT   ; BRANCH TO CHECK FOR INTERRUPTS
F08D| C6 93          DEC    RETRYCNT  ; ANOTHER MISTAKE!!
F08F| 10F2          BPL    TRYADR   ; WELL, LET IT GO THIS TIME
F091| C6 94          DEC    SEKCNTR  ; ONLY RECALIBRATE ONCE!
F093| D053          BNE    DRVERR   ; TRIED TO RECALIBRATE A SECOND TIME, ERROR!
F095| A5 8F          LDA    08F      ; ANOTHER MISTAKE!!
F097| 30EA          BMI    TRYADR   ; WELL, LET IT GO THIS TIME
F099| A5 8C          LDA    CURTRK
F09B| 48            PHA    ; SAVE TRACK WE REALLY WANT
F09C| A9 60          LDA    #60      ; RECALIBRATE ALL OVER AGAIN! ERROR!
F09E| 20 25F1        JSR    SETTRK   ; PRETEND TO BE ON TRACK 80
F0A1| A9 00          LDA    #00
F0A3| 20 04F1        JSR    MYSEEK   ; MOVE TO TRACK 00
F0A6| 68            GOCAL1 PLA
F0A7| 20 04F1        GOCAL  JSR    MYSEEK ; GO TO CORRECT TRACK THIS TIME!
F0AA| 90D7          BCC    TRYADR   ; LOOP BACK, TRY AGAIN ON THIS TRACK
F0AC|              ;
F0AC|              ; HAVE NOW READ AN ADDRESS FIELD CORRECTLY.
F0AC|              ; MAKE SURE THIS IS THE TRACK, SECTOR, AND VOLUME DESIRED.
F0AC|              ;
F0AC| A4 99          RDRIGHT LDY    TRACK ; ON THE RIGHT TRACK?
    
```

CMD
1 -> Read
2 -> Write

```

F0AE| C4 8C          CPY      CURTRK
F0B0| F00E          BEQ      RTRRK      ; IF SO, GOOD
F0B2|                ;
F0B2|                ; RECALIBRATING FROM THIS TRACK
F0B2|                ;
F0B2| A5 8C          LDA      CURTRK      ; PRESERVE DESTINATION TRACK
F0B4| 48             PHA
F0B5| 98             TYA
F0B6| 0A            ASL      A
F0B7| 20 25F1       JSR      SETTRK
F0BA| 68             PLA
F0BB| 20 04F1       JSR      MYSEEK
F0BE| 90CA          BCC      TRYADR2
F0C0| A5 9A          RTRRK    LDA      VOLUME      ; GET ACTUAL VOLUME HERE
F0C2| 85 89          STA      IBSMOD      ; TELL OPSYS WHAT VOLUME WAS THERE
F0C4| A5 98          CORRECTVOL LDA    SECT        ; CHECK IF THIS IS THE RIGHT SECTOR
F0C6| C5 84          CMP      IBSECT
F0C8| D0C0          BNE      TRYADR2      ; NO, TRY ANOTHER SECTOR
F0CA| A5 87          LDA      IBCMD
F0CC| 4A            LSR      A
F0CD| 902A          BCC      WRIT        ; CARRY WAS SET FOR READ OPERATION,
F0CF| 20 40F1       JSR      READ16       ; CLEARED FOR WRITE
F0D2| B0B6          BCS      TRYADR2      ; CARRY SET UPON RETURN IF BAD READ
F0D4| AD DFFF       LDA      ENVIRON
F0D7| 29 7F         AND      #TWOMEG
F0D9| 8D DFFF       STA      ENVIRON      ; SET TWO MEGAHERTZ
F0DC| 20 0FF3       JSR      POSTNIB16    ; DO PARTIAL POSTNIBBLE CONVERSION
F0DF| A6 81          LDX      IBSLOT      ; RESTORE SLOTNUM INTO X
F0E1| B0A7          BCS      TRYADR2      ; CHECKSUM ERROR
F0E3| 18            ALLDONE  CLC
F0E4| A9 00          LDA      #00         ; NO ERROR
F0E6| 9003          BCC      ALDONE1     ; SKIP OVER NEXT BYTE WITH BIT OPCODE
F0E8| A9 82          DRVERR   LDA      #IBDERR    ; BAD DRIVE
F0EA| 38            HNDLERR SEC          ; INDICATE AN ERROR
F0EB| 85 88          ALDONE1  STA      IBSTAT      ; GIVE HIM ERROR
F0ED| BD 88C0       LDA      MOTOROFF,X  ; TURN IT OFF
F0F0| 20 AAF1       JSR      CHKINT      ; BRANCH TO CHECK FOR INTERRUPTS
F0F3| A5 9F          LDA      ENVTEMP     ; RESTORE ORIGINAL ENVIRONMENT
F0F5| 8D DFFF       STA      ENVIRON
F0F8| 60            RTS
F0F9|                ;
F0F9| 20 16F2       WRIT     JSR      WRITE16    ; WRITE NYBBLES NOW
F0FC| 90E5          BCC      ALLDONE     ; IF NO ERRORS
F0FE| A9 81          LDA      #IBWPER     ; DISK IS WRITE PROTECTED!!
F100| 50E8          BVC      HNDLERR     ; TAKEN IF TRUELY WRITE PROTECT ERROR
F102| D086          BNE      TRYADR2     ; OTHERWISE ASSUME AN INTERRUPT MESSED THINGS UP
F104|                ;
F104|                ; THIS IS THE 'SEEK' ROUTINE
F104|                ; SEEKS TRACK 'N' IN SLOT #X/$10
F104|                ; IF DRIVENO IS NEGATIVE, ON DRIVE 0
F104|                ; IF DRIVENO IS POSITIVE, ON DRIVE 1
F104|                ;
F104| 0A            MYSEEK   ASL      A          ; ASSUME TWO PHASE STEPPER.
F105| 85 99          SEEK1    STA      TRKN1     ; SAVE DESTINATION TRACK(*2)
F107| 20 18F1       JSR      ALLOFF      ; TURN ALL PHASES OFF TO BE SURE.
F10A| 20 3EF1       JSR      DRVINDX     ; GET INDEX TO PREVIOUS TRACK FOR CURRENT DRIVE
F10D| B5 85          LDA      DRVOTRK,X
F10F| 85 8C          STA      CURTRK      ; THIS IS WHERE I AM
F111| A5 99          LDA      TRKN1      ; AND WHERE I'M GOING TO
F113| 95 85          STA      DRVOTRK,X
F115| 20 00F4       GOSEEK   JSR      SEEK        ; GO THERE!
F118| A0 03          ALLOFF   LDY      #03    ; TURN OFF ALL PHASES BEFORE RETURNING
F11A| 98            NXOFF    TYA
F11B| 20 4AF4       JSR      CLRPHASE    ; (SEND PHASE IN ACC.)
F11E| 88            DEY
F11F| 10F9          BPL      NXOFF
F121| 46 8C          LSR      CURTRK      ; DIVIDE BACK NOW
F123| 18            CLC
F124| 60            RTS
F125|                ;
F125|                ; THIS SUBROUTINE SETS THE SLOT DEPENDENT TRACK
F125|                ; LOCATION
F125|                ;
F125| 20 3EF1       SETTRK   JSR      DRVINDX     ; GET INDEX TO DRIVE NUMBER
F128| 95 85          STA      DRVOTRK,X
F12A| 60            RTS
F12B|                ;
F12B|                ; *****
F12B|                ;
F12B|                ; SUBR TO TELL IF MOTOR IS STOPPED
F12B|                ;
F12B|                ; IF MOTOR IS STOPPED, CONTROLLER'S
F12B|                ; SHIFT REG WILL NOT BE CHANGING.
F12B|                ;
F12B|                ; RETURN Y=0 AND ZERO FLAG SET IF IT IS STOPPED.
F12B|                ;
F12B|                ; *****
F12B|                ;
F12B| A0 00          CHKDRV   LDY      #00     ; INIT LOOP COUNTER
F12D| BD 8CC0       CHKDRV1  LDA      Q6L,X    ; READ THE SHIFT REG
    
```

```

F130| 20 3DF1          JSR    CKDRTS    ; DELAY
F133| 48              PHA
F134| 68              PLA
F135| DD 8CC0          CMP     Q6L,X    ; HAS SHIFT REG CHANGED?
F138| D003             BNE    CKDRTS    ; YES, MOTOR IS MOVING
F13A| 88              DEY
F13B| D0F0             BNE    CHKDRV1   ; NO, DEC RETRY COUNTER
F13D| 60              RTS          ; AND TRY 256 TIMES
F13E|                 ; THEN RETURN
F13E| 48              DRVINDX  PHA          ; PRESERVE ACC.
F13F| 8A              TXA          ; GET SLOT(*$10)/8
F140| 4A              LSR     A
F141| 4A              LSR     A
F142| 4A              LSR     A
F143| 05 82           ORA     IBDRVN   ; FOR DRIVE 0 OR 1
F145| AA              TAX          ; INTO X FOR INDEX TO TABLE
F146| 68              PLA          ; RESTORE ACC.
F147| 60              RTS
F148|                 ;
F148|                 ;*****
F148|                 ; NOTE: FORMATTING ROUTINES
F148|                 ;     NOTE INCLUDED FOR SOS
F148|                 ;*****
F148|                 ;*****
F148|                 ; READ SUBROUTINE
F148|                 ; (16-SECTOR FORMAT)
F148|                 ;*****
F148|                 ; READS ENCODED BYTES
F148|                 ; INTO NBUF1 AND NBUF2
F148|                 ;
F148|                 ; FIRST READS NBUF2
F148|                 ;     HIGH TO LOW,
F148|                 ; THEN READS NBUF1
F148|                 ;     LOW TO HIGH.
F148|                 ;
F148|                 ; ---- ON ENTRY ----
F148|                 ;
F148|                 ; X-REG: SLOTNUM
F148|                 ;     TIMES $10.
F148|                 ;
F148|                 ; READ MODE (Q6L, Q7L)
F148|                 ;
F148|                 ; ---- ON EXIT ----
F148|                 ;
F148|                 ; CARRY SET IF ERROR
F148|                 ;
F148|                 ; IF NO ERROR:
F148|                 ; A-REG HOLDS $AA.
F148|                 ; X-REG UNCHANGED.
F148|                 ; Y-REG HOLDS $00.
F148|                 ; CARRY CLEAR.
F148|                 ; ---- CAUTION ----
F148|                 ;
F148|                 ; OBSERVE
F148|                 ; 'NO PAGE CROSS'
F148|                 ; WARNINGS ON
F148|                 ; SOME BRANCHES!!
F148|                 ;
F148|                 ; ---- ASSUMES ----
F148|                 ;
F148|                 ; 1 USEC CYCLE TIME
F148|                 ;
F148|                 ;*****
F148| A0 20           READ16  LDY     #20    ; 'MUST FIND' COUNT.
F14A| 88           RSYNC   DEY          ; IF CAN'T FIND MARKS.
F14B| F06A          BEQ     RDERR     ; THEN EXIT WITH CARRY SET
F14D| BD 8CC0          RD1     LDA     Q6L,X    ; READ NIBL.
F150| 10FB          BPL     RD1       ; *** NO PAGE CROSS! ***
F152| 49 D5          RSYNC1  EOR     #0D5   ; DATA MARK1?
F154| D0F4          BNE     RSYNC     ; LOOP IF NOT.
F156| EA           NOP          ; DELAY BETWEEN NIBLS.
F157| BD 8CC0          RD2     LDA     Q6L,X    ;
F15A| 10FB          BPL     RD2       ; *** NO PAGE CROSS! ***
F15C| C9 AA          CMP     #0AA   ; DATA MARK 2?
F15E| D0F2          BNE     RSYNC1   ; (IF NOT, IS IT DM1?)
F160| A0 55          LDY     #055   ; INIT NBUF2 INDEX.
F162|                 ; ( ADDED NIBL DELAY)
F162| EA           NOP          ; DELAY BETWEEN NIBLS.
F163| BD 8CC0          RD3     LDA     Q6L,X    ;
F166| 10FB          BPL     RD3       ; *** NO PAGE CROSS! ***
F168| C9 AD          CMP     #0AD   ; DATA MARK 3?
F16A| D0E6          BNE     RSYNC1   ; (IF NOT, IS IT DM1?)
F16C|                 ; (CARRY SET IF DM3!)
    
```

← Seems like "Note"
should be "Not"

10/31/89 9:56

HD:Apple ///:ROM - Disk I/O

Page 6

```

F16C| EA                NOP                ; DELAY BETWEEN NIBLS.
F16D| EA                NOP                ; DELAY BETWEEN NIBLS.
F16E| BD 8CC0          RD4          LDA      Q6L,X
F171| 10FB              BPL      RD4          ; *** NO PAGE CROSS! ***
F173| 99 0203          STA      NBUF2,Y      ; STORE BYTES DIRECTLY
F176| AD EFFF          LDA      INTERRUPT    ; POLL INTERRUPT LINE
F179| 05 8B             ORA      IMASK        ; (THIS MAY BE USED TO INVALIDATE POLL)
F17B| 1037              BPL      GOSERV
F17D| 88                DEY
F17E| 10EE              BPL      RD4          ; INDEX TO NEXT
F180| C8                RD5          INY
F181| BD 8CC0          RD5A         LDA      Q6L,X      ; (FIRST TIME Y=0)
F184| 10FB              BPL      RD5A         ; GET ENCODED BYTES OF NBUF1
F186| 99 0002          STA      NBUF1,Y
F189| AD EFFF          LDA      INTERRUPT    ; POLL INTERRUPT LINE
F18C| 05 8B             ORA      IMASK        ; (THIS MAY BE USED TO INVALIDATE POLL)
F18E| 1024              BPL      GOSERV
F190| C0 E4             CPY      #0E4        ; WITHIN 1 MS OF COMPLETION?
F192| D0EC              BNE      RD5
F194| C8                INY
F195| BD 8CC0          RD6          LDA      Q6L,X      ; NO POLL FROM NOW ON
F198| 10FB              BPL      RD6
F19A| 99 0002          STA      NBUF1,Y
F19D| C8                INY          ; FINISH OUT NBUF1 PAGE
F19E| D0F5              BNE      RD6
F1A0| BD 8CC0          RDCKSUM      LDA      Q6L,X      ; GET CHECKSUM BYTE.
F1A3| 10FB              BPL      RDCKSUM
F1A5| 85 96             STA      CKSUM
F1A7| 20 01F2          JSR      RDA6        ; CHECK BIT SLIP MARKS
F1AA|                   ;
F1AA|                   ; CHECK FOR INTERRUPTS
F1AA|                   ;
F1AA| 24 8B             CHKINT      BIT      IMASK    ; SHOULD INTERRUPTS BE ALLOWED?
F1AC| 1004              BPL      $010        ; YES, ALLOW THEM.
F1AE| 24 8F             BIT      $08F
F1B0| 1001              BPL      $020
F1B2| 58                $010        CLI
F1B3| 60                $020        RTS
F1B4|                   ;
F1B4| 20 AAF2          GOSERV      JSR      SERVICE  ; GO TO SERVICE INTERRUPT
F1B7| 38                RDERR      SEC
F1B8| 60                RTS
F1B9|                   ;
F1B9|                   ;*****
F1B9|                   ;
F1B9|                   ; READ ADDRESS FIELD *
F1B9|                   ; SUBROUTINE *
F1B9|                   ; (16-SECTOR FORMAT) *
F1B9|                   ; *
F1B9|                   ;*****
F1B9|                   ;
F1B9|                   ; READS VOLUME, TRACK *
F1B9|                   ; AND SECTOR *
F1B9|                   ; *
F1B9|                   ; ---- ON ENTRY ---- *
F1B9|                   ; *
F1B9|                   ; XREG: SLOTNUM TIMES $10 *
F1B9|                   ; *
F1B9|                   ; READ MODE (Q6L, Q7L) *
F1B9|                   ; *
F1B9|                   ; ---- ON EXIT ---- *
F1B9|                   ; *
F1B9|                   ; CARRY SET IF ERROR *
F1B9|                   ; *
F1B9|                   ; IF NO ERROR: *
F1B9|                   ; A-REG HOLDS $AA. *
F1B9|                   ; Y-REG HOLDS $00. *
F1B9|                   ; X-REG UNCHANGED. *
F1B9|                   ; CARRY CLEAR. *
F1B9|                   ; *
F1B9|                   ; CSSTV HOLDS CHKSUM, *
F1B9|                   ; SECTOR, TRACK, AND *
F1B9|                   ; VOLUME READ. *
F1B9|                   ; *
F1B9|                   ; USES TEMPS COUNT, *
F1B9|                   ; LAST, CSUM, AND *
F1B9|                   ; 4 BYTES AT CSSTV. *
F1B9|                   ; *
F1B9|                   ; ---- EXPECTS ---- *
F1B9|                   ; *
F1B9|                   ; ORIGINAL 10-SECTOR *
F1B9|                   ; NORMAL DENSITY NIBLS *
F1B9|                   ; (4-BIT), ODD BITS, *
F1B9|                   ; THEN EVEN *
F1B9|                   ; *
F1B9|                   ; ---- CAUTION ---- *
F1B9|                   ; *
F1B9|                   ; OBSERVE *
F1B9|                   ; 'NO PAGE CROSS' *
F1B9|                   ; WARNINGS ON *

```

```

F1B9|          ;      SOME BRANCHES!!      *
F1B9|          ;                          *
F1B9|          ;      ---- ASSUMES ----      *
F1B9|          ;                          *
F1B9|          ;      1 USEC CYCLE TIME      *
F1B9|          ;                          *
F1B9|          ;*****
F1B9|
F1B9| A0 FC      RDRADR16  LDY      #0FC
F1BB| 84 95      STY      COUNT      ; 'MUST FIND' COUNT.
F1BD| C8        RDASYN   INY
F1BE| D004      BNE      RDA1      ; LOW ORDER OF COUNT
F1C0| E6 95      INC      COUNT      ; (2K NIBLS TO FIND
F1C2| F0F3      BEQ      RDERR     ; ADR MARK, ELSE ERR)
F1C4| BD 8CC0   RDA1     LDA      Q6L,X ; READ NIBL.
F1C7| 10FB      BPL      RDA1      ; *** NO PAGE CROSS! ***
F1C9| C9 D5      RDASN1  CMP      #0D5   ; ADR MARK 1?
F1CB| D0F0      BNE      RDASYN    ; (LOOP IF NOT)
F1CD| EA        NOP
F1CE| BD 8CC0   RDA2     LDA      Q6L,X ; ADDED NIBL DELAY
F1D1| 10FB      BPL      RDA2      ; *** NO PAGE CROSS! ***
F1D3| C9 AA      CMP      #0AA   ; ADR MARK 2?
F1D5| D0F2      BNE      RDASN1    ; (IF NOT, IS IT AM1?)
F1D7| A0 03      LDY      #03      ; INDEX FOR 4-BYTE READ
F1D9|          ;      (ADDED NIBL DELAY)
F1D9| BD 8CC0   RDA3     LDA      Q6L,X
F1DC| 10FB      BPL      RDA3      ; *** NO PAGE CROSS! ***
F1DE| C9 96      CMP      #96   ; ADR MARK 3?
F1E0| D0E7      BNE      RDASN1    ; (IF NOT IS IT AM1?)
F1E2|          ;      (LEAVES CARRY SET!)
F1E2| 78        SEI
F1E3| A9 00      LDA      #00      ; DISABLE INTERRUPT SYSTEM
F1E5| 85 89      STA      CSUM      ; INIT CHECKSUM
F1E7| BD 8CC0   RDA4     LDA      Q6L,X ; READ 'ODD BIT' NIBBL
F1EA| 10FB      BPL      RDA4      ; *** NO PAGE CROSS! ***
F1EC| 2A        ROL      A          ; ALIGN ODD BITS, 1' INTO LSB
F1ED| 85 95      STA      LAST     ; (SAVE THEM)
F1EF| BD 8CC0   RDA5     LDA      Q6L,X ; READ 'EVEN BIT' NIBL
F1F2| 10FB      BPL      RDA5      ; *** NO PAGE CROSS ***
F1F4| 25 95      AND      LAST     ; MERGE ODD AND EVEN BITS
F1F6| 99 97 00   STA      CSSTV,Y  ; STORE DATA BYTE
F1F9| 45 89      EOR      CSUM
F1FB| 88        DEY
F1FC| 10E7     BPL      RDAFLD   ; LOOP ON 4 DATA BYTES.
F1FE| A8        TAY
F1FF| D0B6     BNE      RDERR   ; IF FINAL CHECKSUM
F201| BD 8CC0   RDA6     LDA      Q6L,X ; NONZERO, THEN ERROR
F204| 10FB      BPL      RDA6      ; FIRST BIT SLIP NIBBL
F206| C9 DE      CMP      #0DE   ; *** NO PAGE CROSS! ***
F208| D0AD     BNE      RDERR   ; ERROR IF NONMATCH
F20A| EA        NOP
F20B| BD 8CC0   RDA7     LDA      Q6L,X ; DELAY
F20E| 10FB      BPL      RDA7      ; SECOND BIT-SLIP NIBL
F210| C9 AA      CMP      #0AA   ; *** NO PAGE CROSS! ***
F212| D0A3     BNE      RDERR   ; ERROR IF NOMATCH
F214| 18        RDEXIT  CLC
F215| 60        WEXIT   RTS
F216|          ;
F216|          ;*****
F216|          ;
F216|          ;      WRITE SUBR      *
F216|          ;      (16-SECTOR FORMAT) *
F216|          ;                          *
F216|          ;*****
F216|          ;
F216|          ;      WRITES DATA FROM *
F216|          ;      NBUF1 AND NBUF2 *
F216|          ;                          *
F216|          ;      FIRST NBUF2, *
F216|          ;      HIGH TO LOW. *
F216|          ;      THEN NBUF1, *
F216|          ;      LOW TO HIGH *
F216|          ;                          *
F216|          ;      ---- ON ENTRY ---- *
F216|          ;                          *
F216|          ;      X-REG SLOTNUM *
F216|          ;      TIMES $10 *
F216|          ;                          *
F216|          ;      ---- ON EXIT ---- *
F216|          ;                          *
F216|          ;      CARRY SET IF ERROR. *
F216|          ;      (W PROT VIOLATION) *
F216|          ;                          *
F216|          ;      IF NO ERROR: *
F216|          ;                          *
F216|          ;      A-REG UNCERTAIN. *
F216|          ;      X-REG UNCHANGED. *
F216|          ;      Y-REG HOLDS $00. *
F216|          ;      CARRY CLEAR. *
    
```

```

F216|      ;      *
F216|      ; ---- ASSUMES ---- *
F216|      ;      *
F216|      ; 1 USEC CYCLE TIME *
F216|      ;      *
F216|      ;*****
F216|      ;
F216| 38      WRITE16  SEC      ; ANTICIPATE WPROT ERR.
F217| B8      CLV          ; TO INDICATE WRITE PROTECT ERROR INSTEAD OF
F218|          ;          ; INTERRUPT
F218| BD 8DC0  LDA      Q6H,X
F21B| BD 8EC0  LDA      Q7L,X      ; SENSE WPROT FLAG.
F21E| 30F5    BMI      WEXIT      ; BRANCH IF WRITE PROTECTED
F220| A9 FF    WRIT1   LDA      #0FF      ; SYNC DATA.
F222| 9D 8FC0  STA      Q7H,X      ; (5) GOTO WRITE MODE
F225| 1D 8CC0  ORA      Q6L,X      ; (4)
F228| A0 04    LDY      #04        ; (2) FOR FIVE NIBLS.
F22A| EA      NOP          ; (2)
F22B| 48      PHA          ; (4)
F22C| 68      PLA          ; (3)
F22D| 48      WSYNC   PHA          ; (4) EXACT TIMING
F22E| 68      PLA          ; (3)
F22F| 20 BBF2 JSR      WNIBL7      ; (13,9,6) WRITE SYNC
F232| 88      DEY          ; (2)
F233| D0F8    BNE      WSYNC      ; (2*) MUST NOT CROSS PAGE!
F235| A9 D5    LDA      #0D5      ; (2) 1ST DATA MARK
F237| 20 BAF2 JSR      WNIBL9      ; (15,9,6)
F23A| A9 AA    LDA      #0AA      ; (2) 2ND DATA MARK
F23C| 20 BAF2 JSR      WNIBL9      ; (15,9,6)
F23F| A9 AD    LDA      #0AD      ; (2) 3RD DATA MARK
F241| 20 BAF2 JSR      WNIBL9      ; (15,9,6)
F244| A0 55    LDY      #55        ; (2) NBUF2 INDEX
F246| EA      NOP          ; (2) FOR TIMING
F247| EA      NOP          ; (2)
F248| EA      NOP          ; (2)
F249| D008    BNE      VRYFRST    ; (3) BRANCH ALWAYS
F24B| AD EFFF  WINTRPT LDA      INTERUPT    ; (4) POLL INTERRUPT LINE
F24E| 05 8B    ORA      IMASK      ; (3)
F250| 38      SEC          ; (2)
F251| 1057    BPL      SERVICE    ; (2) BRANCH IF INTERRUPT HAS OCCURED
F253| 3000    BMI      WRTFRST    ; (3) FOR TIMING.
F255| B9 0203 WRTFRST LDA      NBUF2,Y      ; (4)
F258| 9D 8DC0  STA      Q6H,X      ; (5) STORE ENCODED BYTE
F25B| BD 8CC0  LDA      Q6L,X      ; (4) TIME MUST = 32 US PER BYTE!
F25E| 88      DEY          ; (2)
F25F| 10EA    BPL      WINTRPT    ; (3) (2 IF BRANCH NOT TAKEN)
F261| 98      TYA          ; (2) INSURE NO INTERRUPT THIS BYTE
F262| 3003    BMI      WMIDLE      ; (3) BRANCH ALWAYS.
F264| AD EFFF  WNTRPT1 LDA      INTERUPT    ; (4) POLL INTERRUPT LINE
F267| 05 8B    WMIDLE  ORA      IMASK      ; (3)
F269| 38      SEC          ; (2)
F26A| 3002    BMI      WDATA2      ; (3) BRANCH IF NO INTERRUPT
F26C| 103C    BPL      SERVICE    ; GO SERVICE INTERRUPT.
F26E| C8      INY          ; (2)
F26F| B9 0002 WDATA2  LDA      NBUF1,Y      ; (4)
F272| 9D 8DC0  STA      Q6H,X      ; (5) STORE ENCODED BYTE
F275| BD 8CC0  LDA      Q6L,X      ; (4)
F278| C0 E4    CPY      #0E4      ; (2) WITHIN 1 MS OF COMPLETION?
F27A| D0E8    BNE      WNTRPT1    ; (3) (2) NO KEEP WRITING AND POLLING.
F27C| EA      NOP          ; (2)
F27D| C8      INY          ; (2)
F27E| EA      WDATA3  NOP          ; (2)
F27F| EA      NOP          ; (2)
F280| 48      PHA          ; (4)
F281| 68      PLA          ; (3)
F282| B9 0002 WDATA3  LDA      NBUF1,Y      ; (4) WRITE LAST OF ENCODED BYTES
F285| 9D 8DC0  STA      Q6H,X      ; (5) WITHOUT POLLING INTERRUPTS.
F288| BD 8CC0  LDA      Q6L,X      ; (4)
F28B| A5 96    LDA      CKSUM      ; (3) NORMALLY FOR TIMING
F28D| C8      INY          ; (2)
F28E| D0EE    BNE      WDATA3      ; (3) (2)
F290| F000    BEQ      WRCKSUM      ; (3) BRANCH ALWAYS
F292| 20 BBF2 JSR      WNIBL7      ; (13,9,6) GO WRITE CHECK SUM!!
F295| 48      PHA          ; (3)
F296| 68      PLA          ; (4)
F297| B9 C0F3 WRBITSLMK LDA      BITSPLPMK,Y ; (4) LOAD BIT SLIP MARK
F29A| 20 BDF2 JSR      WNIBL      ; (6,9,6)
F29D| C8      INY          ; (2)
F29E| C0 04    CPY      #04        ; (2)
F2A0| D0F5    BNE      WRBITSLMK ; (2) (3)
F2A2| 18      CLC          ; (2)
F2A3| BD 8EC0  NOWRITE LDA      Q7L,X      ; OUT OF WRITE MODE.
F2A6| BD 8CC0  LDA      Q6L,X      ; TO READ MODE.
F2A9| 60      RTS          ; RETURN FROM WRITE.
F2AA|      ;
F2AA| 2C 54F3 SERVICE BIT  SEV          ; SET VFLAG TO INDICATE INTERRUPT
F2AD| 20 A3F2 JSR      NOWRITE      ; TAKE IT OUT OF WRITE MODE!
F2B0| A5 8F    LDA      08F
F2B2| 1002    BPL      $010
F2B4| 85 8B    STA      IMASK
    
```

```

F2B6| C6 8F          $010      DEC      08F
F2B8| 58
F2B9| 60
F2BA|
F2BA| ;
F2BA| ;*****
F2BA| ;
F2BA| ;       7-BIT NIBL WRITE SUBRS *
F2BA| ;
F2BA| ;       A-REG OR'D PRIOR EXIT *
F2BA| ;       CARRY CLEARED
F2BA| ;
F2BA| ;*****
F2BA| ;
F2BA| 18           WNIBL9    CLC          ; (2) 9 CYCLES, THEN WRITE
F2BB| 48           WNIBL7    PHA          ; (3) 7 CYCLES, THEN WRITE
F2BC| 68
F2BD| 9D 8DC0      WNIBL     STA      Q6H,X   ; (4)
F2C0| 1D 8CC0      ORA      Q6L,X   ; (5) NIBL WRITE SUB
F2C3| 60           RTS
F2C4|
F2C4| ;
F2C4| ;*****
F2C4| ;
F2C4| ;       PRENIBILIZE SUBR *
F2C4| ;       (16-SECTOR FORMAT) *
F2C4| ;
F2C4| ;*****
F2C4| ;
F2C4| ;       CONVERTS 256 BYTES OF *
F2C4| ;       USER DATA IN (BUF) INTO *
F2C4| ;       ENCODED BYTES TO BE *
F2C4| ;       WRITTEN DIRECTLY TO DISK *
F2C4| ;       ENCODED CHECK SUM IN *
F2C4| ;       ZERO PAGE 'CKSUM' *
F2C4| ;
F2C4| ;       ---- ON ENTRY ---- *
F2C4| ;
F2C4| ;       BUF IS 2-BYTE POINTER *
F2C4| ;       TO 256 BYTES OF USER *
F2C4| ;       DATA. *
F2C4| ;
F2C4| ;       A-REG CHECK SUM. *
F2C4| ;       X-REG UNCERTAIN *
F2C4| ;       Y-REG HOLDS 0. *
F2C4| ;       CARRY SET. *
F2C4| ;
F2C4| ;*****
F2C4| ;
F2C4| A2 02         PRENIB16  LDX      #02          ; START NBUF2 INDEX.
F2C6| A0 00         LDY      #00          ; START USER BUF INDEX.
F2C8| 88           DEY
F2C9| B1 9B         LDA      (BUF),Y      ; NEXT USER BYTE
F2CB| 4A           LSR      A
F2CC| 3E 0103      ROL      NBUF2-1,X
F2CF| 4A           LSR      A
F2D0| 3E 0103      ROL      NBUF2-1,X
F2D3| 99 0102      STA      NBUF1+1,Y
F2D6| E8           INX
F2D7| E0 56         CPX      #56
F2D9| 90ED        BCC      PRENIB1      ; BR IF NO WRAPAROUND
F2DB| A2 00         LDX      #00          ; RESET NBUF2 INDEX
F2DD| 98           TYA
F2DE| D0E8        BNE      PRENIB1      ; (DONE IF ZERO)
F2E0| A0 56         LDY      #56          ; (ACC=0 FOR CHECK SUM)
F2E2| 59 0003      EOR      NBUF2-2,Y
F2E5| 29 3F         PRENIB2  AND      #03F
F2E7| AA           TAX
F2E8| BD 55F3      LDA      NIBL,X
F2EB| 99 0103      STA      NBUF2-1,Y
F2EE| B9 0003      LDA      NBUF2-2,Y
F2F1| 88           DEY
F2F2| D0EE        BNE      PRENIB3      ; LOOP UNTIL ALL OF NBUF2 IS CONVERTED.
F2F4| 29 3F         AND      #3F
F2F6| 59 0102      PRENIB4  EOR      NBUF1+1,Y
F2F9| AA           TAX
F2FA| BD 55F3      LDA      NIBL,X
F2FD| 99 0002      STA      NBUF1,Y
F300| B9 0102      LDA      NBUF1+1,Y
F303| C8           INY
F304| D0F0        BNE      PRENIB4
F306| AA           TAX
F307| BD 55F3      LDA      NIBL,X
F30A| 85 96         STA      CKSUM
F30C| 4C 4CF3      JMP      SETIMEG
F30F|
F30F| ;
F30F| ;*****
F30F| ;
F30F| ;       POSTNIBLIZE SUBR *
F30F| ;       16-SECTOR FORMAT *
F30F| ;
F30F| ;*****

```



```

F30F|          ;
F30F| 38          POSTNIB16 SEC
F310| A0 55      LDY          #55          ; FIRST CONVERT TO 6 BIT NIBBLES
F312| A9 00      LDA          #00          ; INIT CHECK SUM
F314| BE 0203    PNIBL1     LDX          NBUF2,Y  ; GET ENCODED BYTE
F317| 5D 00F3    EOR          DNIBL,X
F31A| 3030      BMI          SET1MEG      ; SET 1 MHZ
F31C| 99 0203    STA          NBUF2,Y  ; REPLACE WITH 6 BIT EQUIV.
F31F| 88          DEY
F320| 10A6      BPL          PRENIB1     ; LOOP UNTIL DONE WITH NIBBLE BUFFER 2
F322| C8          INY          ; NOW Y=0
F323| BE 0002    PNIBL2     LDX          NBUF1,Y  ; DO THE SAME WITH
F326| 5D 00F3    EOR          DNIBL,X
F329| 99 0002    STA          NBUF1,Y  ; NIBBLE BUFFER 1
F32C| C8          INY          ; DO ALL 256 BYTES
F32D| D0F4      BNE          PNIBL2
F32F| A6 96      LDX          CKSUM      ; MAKE SURE CHECK SUM MATCHES
F331| 5D 00F3    EOR          DNIBL,X  ; BETTER BE ZERO
F334| D016      BNE          POSTERR     ; BRANCH IF IT IS
F336| A2 56      POST1      LDX          #56          ; INIT NBUF2 INDEX
F338| CA          POST2      DEX          ; NBUF IDX $55 TO $00
F339| 30FB      BMI          POST1     ; WRAPAROUND IF NEG
F33B| B9 0002    LDA          NBUF1,Y
F33E| 5E 0203    LSR          NBUF2,X  ; SHIFT 2 BITS FROM
F341| 2A          ROL          A          ; CURRENT NBUF2 NIBL
F342| 5E 0203    LSR          NBUF2,X  ; CURRENT NBUF1
F345| 2A          ROL          A          ; NIBL.
F346| 91 9B      STA          (BUF),Y  ; BYTE OF USER DATA
F348| C8          INY          ; NEXT USER BYTE
F349| D0ED      BNE          POST2
F34B| 18          CLC          ; GOOD DATA
F34C| F34C      POSTERR     .EQU          *
F34C| AD DFFF    SET1MEG     LDA          ENVIRON
F34F| 09 80      ORA          #ONEMEG     ; SET TO ONE MEGAHERTZ CLOCK RATE
F351| 8D DFFF    STA          ENVIRON
F354| 60          SEV          RTS          ; (SEV USED TO SET VFLAG)
F355|          ;
F355|          ;*****
F355|          ;
F355|          ;          6-BIT TO 7-BIT          *
F355|          ;          NIBL CONVERSION TABLE *
F355|          ;          *
F355|          ;*****
F355|          ;
F355|          ;          CODES WITH MORE THAN          *
F355|          ;          ONE PAIR OF ADJACENT          *
F355|          ;          ZEROES OR WITH NO          *
F355|          ;          ADJACENT ONES (EXCEPT          *
F355|          ;          B7) ARE EXCLUDED.          *
F355|          ;          *
F355|          ;*****
F355|          ;
F355| 96 97 9A 9B 9D 9E 9F NIBL .BYTE 96,97,9A,9B,9D,9E,9F,0A6,0A7,0AB,0AC,0AD,0AE,0AF,0B2,0B3,0B4,0B5
F35C| A6 A7 AB AC AD AE AF
F363| B2 B3 B4 B5
F367| B6 B7 B9 BA BB BC BD .BYTE 0B6,0B7,0B9,0BA,0BB,0BC,0BD,0BE,0BF,0CB,0CD,0CE,0CF,0D3,0D6,0D7
F36E| BE BF CB CD CE CF D3
F375| D6 D7
F377| D9 DA DB DC DD DE DF .BYTE 0D9,0DA,0DB,0DC,0DD,0DE,0DF,0E5,0E6,0E7,0E9,0EA,0EB,0EC,0ED,0EE
F37E| E5 E6 E7 E9 EA EB EC
F385| ED EE
F387| EF F2 F3 F4 F5 F6 F7 .BYTE 0EF,0F2,0F3,0F4,0F5,0F6,0F7,0F9,0FA,0FB,0FC,0FD,0FE,0FF
F38E| F9 FA FB FC FD FE FF
F395|          ;
F395|          ;*****
F395|          ;
F395|          ;          7-BIT TO 6-BIT          *
F395|          ;          'DENIBLIZE' TABL          *
F395|          ;          (16-SECTOR FORMAT)          *
F395|          ;          *
F395|          ;          VALID CODES          *
F395|          ;          $96 TO $FF ONLY.          *
F395|          ;          *
F395|          ;          *
F395|          ;          CODES WITH MORE THAN          *
F395|          ;          ONE PAIR OF ADJACENT          *
F395|          ;          ZEROES OR WITH NO          *
F395|          ;          ADJACENT ONES (EXCEPT          *
F395|          ;          BIT 7) ARE EXCLUDED          *
F395|          ;          *
F395|          ;*****
F395|          ;
F395| F300      DNIBL      .EQU          REGRWTS+300
F395| 01 00 01      .BYTE          01,00,01
F398| 98 99 02 03 9C 04 05 .BYTE 98,99,02,03,9C,04,05,06,0A0,0A1,0A2,0A3,0A4,0A5,07,08,0A8
F39F| 06 A0 A1 A2 A3 A4 A5
F3A6| 07 08 A8
F3A9| A9 AA 09 0A 0B 0C 0D .BYTE 0A9,0AA,09,0A,0B,0C,0D,0B0,0B1,0E,0F,10,11,12,13,0B8,14,15
F3B0| B0 B1 0E 0F 10 11 12
F3B7| 13 B8 14 15
F3BB| 16 17 18 19 1A .BYTE 16,17,18,19,1A
    
```

```

F3C0| DE AA EB FF C4 C5 C6 BITSLLIPMK .BYTE 0DE,0AA,0EB,0FF,0C4,0C5,0C6,0C7,0C8,0C9,0CA,1B,0CC,1C,1D,1E
F3C7| C7 C8 C9 CA 1B CC 1C
F3CE| 1D 1E
F3D0| D0 D1 D2 1F D4 D5 20 .BYTE 0D0,0D1,0D2,1F,0D4,0D5,20,21,0D8,22,23,24,25,26,27,28,0E0,0E1
F3D7| 21 D8 22 23 24 25 26
F3DE| 27 28 E0 E1
F3E2| E2 E3 E4 29 2A 2B E8 .BYTE 0E2,0E3,0E4,29,2A,2B,0E8,2C,2D,2E,2F,30,31,32,0F0,0F1,33,34
F3E9| 2C 2D 2E 2F 30 31 32
F3F0| F0 F1 33 34
F3F4| 35 36 37 38 F8 39 3A .BYTE 35,36,37,38,0F8,39,3A,3B,3C,3D,3E,3F
F3FB| 3B 3C 3D 3E 3F

```

```

F400|
F400| ;*****
F400| ;
F400| ; FAST SEEK SUBROUTINE
F400| ;
F400| ;*****
F400| ;
F400| ; ---- ON ENTRY ----
F400| ;
F400| ; X-REG HOLDS SLOTNUM
F400| ; TIMES $10
F400| ;
F400| ; A-REG HOLDS DESIRED
F400| ; HALFTRACK.
F400| ;
F400| ; CURTRK HOLDS DESIRED
F400| ; HALFTRACK.
F400| ;
F400| ; ---- ON EXIT ----
F400| ;
F400| ; A-REG UNCERTAIN.
F400| ; Y-REG UNCERTAIN.
F400| ; X-REG UNDISTURBED.
F400| ;
F400| ; CURTRK AND TRKN HOLD
F400| ; FINAL HALFTRACK.
F400| ;
F400| ; PRIOR HOLDS PRIOR
F400| ; HALFTRACK IF SEEK
F400| ; WAS REQUIRED.
F400| ;
F400| ; MONTIMEL AND MONTIMEH
F400| ; ARE INCREMENTED BY
F400| ; THE NUMBER OF
F400| ; 100 USEC QUANTUMS
F400| ; REQUIRED BY SEEK
F400| ; FOR MOTOR ON TIME
F400| ; OVERLAP.
F400| ;
F400| ; --- VARIABLES USED ---
F400| ;
F400| ; CURTRK, TRKN, COUNT,
F400| ; PRIOR, SLOTTIME
F400| ; MONTIMEL, MONTIMEH
F400| ;
F400| ;*****
F400| ;
F400| 85 9E SEEK STA TRKN ; SAVE TARGET TRACK
F400| C5 8C CMP CURTRK ; ON DESIRED TRACK?
F400| F042 BEQ SETPHASE ; YES, ENERGIZE PHASE AND RETURN
F400| A9 00 LDA #00
F400| 85 95 STA TRKCNT ; HALFTRACK COUNT.
F400| A5 8C LDA CURTRK ; SAVE CURTRK FOR
F400| 85 9D STA PRIOR ; DELAYED TURN OFF.
F400| 38 SEC
F400| E5 9E SBC TRKN ; DELTA-TRACKS.
F400| F031 BEQ SEEKEND ; BR IF CURTRK=DESTINATION
F400| B006 BCS OUT ; (MOVE OUT, NOT IN)
F400| 49 FF EOR #0FF ; CALC TRKS TO GO.
F400| E6 8C INC CURTRK ; DECR CURRENT TRACK (OUT)
F400| 9004 BCC MINTST ; (ALWAYS TAKEN).
F400| 69 FE OUT ADC #0FE ; CALC TRACKS TO GO.
F400| C6 8C DEC CURTRK ; DECR CURRENT TRACK (OUT)
F400| C5 95 MINTST CMP TRKCNT
F400| 9002 BCC MAXTST ; AND 'TRKS MOVED'
F400| A5 95 LDA TRKCNT
F400| C9 09 MAXTST CMP #09
F400| B002 BCS STEP2 ; IF TRKCNT>$08 LEAVE Y ALONE (Y=$08)
F400| A8 STEP TAY ; ELSE SET ACCELERATION INDEX IN Y
F400| 38 SEC
F400| 20 48F4 STEP2 JSR SETPHASE
F400| B9 67F4 LDA ONTABLE,Y ; FOR 'ONTIME'
F400| 20 56F4 JSR MSWAIT ; (100 USEC INTERVALS)
F400| A5 9D LDA PRIOR
F400| 18 CLC ; FOR PHASE OFF
F400| 20 4AF4 JSR CLRPHASE ; TURN OFF PRIOR PHASE
F400| B9 70F4 LDA OFFTABLE,Y ; THEN WAIT 'OFFTIME'
F400| 20 56F4 JSR MSWAIT ; (100 USEC INTERVALS)
F400| E6 95 INC TRKCNT ; 'TRACKS MOVED' COUNT.

```

10/31/89 9:56

HD:Apple ///:ROM - Disk I/O

Page 12

```

F442| D0C6          BNE     SEEK2      ; (ALWAYS TAKEN)
F444| 20 56F4      SEEKEND JSR     MSWAIT     ; SETTLE 25 MSEC
F447| 18             CLC          ; SET FOR PHASE OFF
F448| A5 8C          SETPHASE LDA     CURTRK     ; GET CURRENT TRACK
F44A| 29 03          CLRPHASE AND     #03         ; MASK FOR 1 AND 4 PHASES
F44C| 2A             ROL         A             ; DOUBLE FOR PHASE ON/OFF INDEX
F44D| 05 81          ORA         IBSLOT
F44F| AA             TAX
F450| BD 80C0        LDA     PHASEOFF,X      ; TURN ON/OFF ONE PHASE
F453| A6 81          LDX     IBSLOT         ; RESTORE X-REG
F455| 60             SEEKRTS RTS          ; AND RETURN
F456|
F456| ;
F456| ;*****
F456| ;
F456| ;      MSWAIT SUBROUTINE
F456| ;
F456| ;*****
F456| ;
F456| ;      DELAYS A SPECIFIED
F456| ;      NUMBER OF 100 USEC
F456| ;      INTERVALS FOR MOTOR
F456| ;      ON TIMING
F456| ;
F456| ;      ---- ON EXIT ----
F456| ;
F456| ;      A-REG HOLDS $00
F456| ;      X-REG HOLDS $00
F456| ;      Y-REG UNCHANGED
F456| ;      CARRY SET
F456| ;
F456| ;      MONTIMEL, MONTIMEH
F456| ;      ARE INCREMENTED ONCE
F456| ;      PER 100 USEC INTERVAL
F456| ;      FOR MOTOR ON TIMING
F456| ;
F456| ;      ---- ASSUMES ----
F456| ;
F456| ;      1 USEC CYCLE TIME
F456| ;
F456| ;*****
F456| ;
F456| A2 11          MSWAIT LDX     #11
F458| CA            MSW1  DEX
F459| D0FD          MSW1  BNE     MSW1      ; DELAY 86 USEC
F45B| E6 99          INC     MONTIMEL
F45D| D002          BNE     MSW2
F45F| E6 9A          INC     MONTIMEH      ; DOUBLE BYTE INCREMENT
F461| 38            MSW2  SEC
F462| E9 01          SBC     #01           ; DONE IN INTERVALS
F464| D0F0          BNE     MSWAIT       ; (A-REG COUNTS)
F466| 60            RTS
F467|
F467| ;
F467| ;*****
F467| ;
F467| ;      PHASE ON-, OFF-TIME
F467| ;      TABLES IN 100-USEC
F467| ;      INTERVALS. (SEEK)
F467| ;
F467| ;*****
F467| ;
F467| 01 30 28 24 20 1E 1D ONTABLE .BYTE 01,30,28,24,20,1E,1D,1C,1C
F46E| 1C 1C
F470| 70 2C 26 22 1F 1E 1D OFFTABLE .BYTE 70,2C,26,22,1F,1E,1D,1C,1C
F477| 1C 1C
F479|
F479| 86 83          BLOCKIO STX     IBTRK
F47B| A0 05          LDY     #05
F47D| 48            PHA
F47E| 0A            TRKSEC ASL     A
F47F| 26 83          ROL     IBTRK
F481| 88            DEY
F482| D0FA          BNE     TRKSEC
F484| 68            PLA
F485| 29 07          AND     #07
F487| A8            TAY
F488| B9 A0F4        LDA     SECTABL,Y
F48B| 85 84          STA     IBSECT
F48D| 20 00F0        JSR     REGRWTS
F490| B00B          BCS     QUIT
F492| E6 86          INC     IBBUFP+1
F494| E6 84          INC     IBSECT
F496| E6 84          INC     IBSECT
F498| 20 00F0        JSR     REGRWTS
F49B| C6 86          DEC     IBBUFP+1
F49D| A5 88          QUIT  LDA     IBSTAT
F49F| 60            RTS
F4A0|
F4A0| 00 04 08 0C 01 05 09 SECTABL .BYTE 00,04,08,0C,01,05,09,0D
F4A7| 0D
F4A8| ;*****

```

```

F4A8| ; ; *
F4A8| ; JOYSTICK READ ROUTINE *
F4A8| ; *
F4A8| ;*****
F4A8| ; ENTRY ACC= COUNT DOWN HIGH *
F4A8| ; X&Y= DON'T CARE *
F4A8| ; *
F4A8| ; EXIT ACC= TIMER HIGH BYTE *
F4A8| ; Y= TIMER LOW BYTE *
F4A8| ; CARRY CLEAR *
F4A8| ; *
F4A8| ; IF CARRY SET, ROUTINE *
F4A8| ; WAS INTERRUPTED & *
F4A8| ; ACC & Y ARE INVALID *
F4A8| ;*****
F4A8| ;
F4A8| FFD9 TIMLATCH .EQU 0FFD9
F4A8| FFD8 TIMER1L .EQU 0FFD8
F4A8| FFD9 TIMER1H .EQU 0FFD9
F4A8| C066 JOYRDY .EQU 0C066
F4A8| ;
F4A8| F4A8 ANALOG .EQU * ; CARRY SHOULD BE SET!
F4A8| 8D D9FF STA TIMLATCH ; START THE TIMER!
F4AB| AD EFFF ANLOG1 LDA INTERRUPT
F4AE| 2D 66C0 AND JOYRDY ; WAIT FOR ONE OR THE OTHER TO GO LOW
F4B1| 30F8 BMI ANLOG1
F4B3| AD 66C0 LDA JOYRDY ; WAS IT REALLY THE JOYSTICK?
F4B6| 300C BMI GOODTIME ; NOPE, WHAT TIME IS IT?
F4B8| 18 CLC ; TIME'S A SLIP SLIDIN AWAY
F4B9| AD D9FF LDA TIMER1H ; NOW, WHAT TIME IS IT?
F4BC| AC D8FF LDY TIMER1L
F4BF| 1003 BPL GOODTIME ; TIME WAS VALID!
F4C1| AD D9FF LDA TIMER1H ; HI BYTE CHANGED
F4C4| 60 GOODTIME RTS
F4C5| ;
F4C5| .END
    
```

 SYMBOL TABLE DUMP

AB - Absolute LB - Label UD - Undefined MC - Macro
 RF - Ref DF - Def PR - Proc FC - Func
 PB - Public PV - Private CS - Consts

ALDONE1	LB F0EB	ALLDONE	LB F0E3	ALLOFF	LB F118	ANALOG	LB F4A8	ANLOG1	LB F4AB
BITSLIP	LB F3C0	BLOCKIO	LB F479	BUF	AB 009B	CHKDRV	LB F12B	CHKDRV1	LB F12D
CHKINT	LB F1AA	CKDRTS	LB F13D	CKSUM	AB 0096	CLRPHASE	LB F44A	CONWAIT	LB F054
CORRECTV	LB F0C4	COUNT	AB 0095	CSSTV	AB 0097	CSUM	AB 0089	CSUM1	AB 0097
CURTRK	AB 008C	DISKIO	PR ----	DNIBL	LB F300	DRIVSEL	LB F035	DRV1EN	AB C08B
DRVERR	LB F0E8	DRVINDX	LB F13E	DRVON	AB C08A	DRVOTRK	AB 0085	DRVWAIT	LB F041
DMOT	AB 00E0	ENVIRON	AB FFD9	ENVTEMP	AB 009F	GOCAL	LB F0A7	GOCAL1	LB F0A6
GOODTIME	LB F4C4	GOSEK	LB F115	GOSERV	LB F1B4	HNDLERR	LB F0EA	HRDERRS	AB 0080
IBBUFP	AB 0085	IBCMD	AB 0087	IBDERR	AB 0082	IBDRVN	AB 0082	IBNODRV	AB 0080
IBRERR	AB 0083	IBSECT	AB 0084	IBSLOT	AB 0081	IBSMOD	AB 0089	IBSTAT	AB 0088
IBTRK	AB 0083	IBWPER	AB 0081	IMASK	AB 008B	INTERUPT	AB FFEF	IOBPDN	AB 008A
JOYRDY	AB C066	LAST	AB 0095	MAKTST	LB F425	MINTST	LB F41F	MONTIMEH	AB 009A
MONTIMEL	AB 0099	MOTOF	LB F052	MOTOROFF	AB C088	MOTORON	AB C089	MSW1	LB F458
MSW2	LB F461	MSWAIT	LB F456	MYSEEK	LB F104	NBUF1	AB 0200	NBUF2	AB 0302
NIBL	LB F355	NODRIVER	LB F064	NOWRITE	LB F2A3	NXOFF	LB F11A	OFFTABLE	LB F470
OK	LB F048	ONEMEG	AB 0080	ONTABLE	LB F467	OUT	LB F41B	PHASEOFF	AB C080
PHASEON	AB C081	PHASON	AB C081	PHSOFF	AB C080	PNIBL1	LB F314	PNIBL2	LB F323
POST1	LB F336	POST2	LB F338	POSTERR	LB F34C	POSTNIB1	LB F30F	PRENIB1	LB F2C8
PRENIB16	LB F2C4	PRENIB2	LB F2E5	PRENIB3	LB F2E2	PRENIB4	LB F2F6	PRIOR	AB 009D
Q6H	AB C08D	Q6L	AB C08C	Q7H	AB C08F	Q7L	AB C08E	QUIT	LB F49D
RD1	LB F14D	RD2	LB F157	RD3	LB F163	RD4	LB F16E	RD5	LB F180
RD5A	LB F181	RD6	LB F195	RDA1	LB F1C4	RDA2	LB F1CE	RDA3	LB F1D9
RDA4	LB F1E7	RDA5	LB F1E9	RDA6	LB F201	RDA7	LB F20B	RDADR16	LB F1B9
RDAFLD	LB F1E5	RDASN1	LB F1C9	RDASYN	LB F1BD	RDCKSUM	LB F1A0	RDERR	LB F1B7
RDEXIT	LB F214	RDRIGHT	LB F0AC	READ16	LB F148	REGRWTS	LB F000	RETRYCNT	AB 0093
RSYNC	LB F14A	RSYNCL	LB F152	RTRK	LB F0C0	SECT	AB 0098	SECTABL	LB F4A0
SEEK	LB F400	SEEK1	LB F105	SEEK2	LB F40A	SEEKCNT	AB 0094	SEEKEND	LB F444
SEEKRTS	LB F455	SERVICE	LB F2AA	SETIMEG	LB F34C	SETPHASE	LB F448	SETTRK	LB F125
SEV	LB F354	STEP	LB F429	STEP2	LB F42B	TEMP	AB 0097	TIMER1H	AB FFD9
TIMER1L	AB FFD8	TIMLATCH	AB FFD9	TRACK	AB 0099	TRKCNT	AB 0095	TRKN	AB 009E
TRKN1	AB 0099	TRKSEC	LB F47E	TRYADR	LB F083	TRYADR2	LB F08A	TRYTRK	LB F069
TRYTRK2	LB F07F	TWOMEG	AB 007F	VOLUME	AB 009A	VRYFRST	LB F253	WDATA2	LB F26E
WDATA3	LB F27E	WEXIT	LB F215	WINTREP	LB F24B	WMIDLE	LB F267	WNIBL	LB F2BD
WNIBL7	LB F2BB	WNIBL9	LB F2BA	WNTREP1	LB F264	WRBITSLM	LB F297	WRCKSUM	LB F292
WRIT	LB F0F9	WRIT1	LB F220	WRITE16	LB F216	WRTRFRST	LB F255	WSYNC	LB F22D

Assembly complete: 1076 lines
 0 Errors flagged on this Assembly

 6502 OPCODE STATIC FREQUENCIES

ADC : 1 m
 AND : 8 | *****

```

ASL : 3 | **
BCC : 10 | *****
BCS : 7 | *****
BEQ : 8 | *****
BIT : 3 | **
BMI : 10 | *****
BNE : 38 | *****
BPL : 28 | *****
BVC : 1 m
CLC : 9 | *****
CLI : 2 | *
CLV : 1 m
CMP : 14 | *****
CPX : 1 m
CPY : 4 | ***
DEC : 5 | ****
DEX : 2 | *
DEY : 13 | *****
EOR : 8 | *****
INC : 10 | *****
INX : 2 | *
INY : 12 | *****
JMP : 2 | *
JSR : 39 | *****
LDA : 86 M *****
LDX : 12 | *****
LDY : 18 | *****
LSR : 9 | *****
NOP : 13 | *****
ORA : 9 | *****
PHA : 10 | *****
PHP : 4 | ***
PLA : 11 | *****
PLP : 3 | **
ROL : 7 | *****
ROR : 6 | *****
RTS : 16 | *****
SBC : 2 | *
SEC : 9 | *****
SEI : 1 m
STA : 42 | *****
STX : 1 m
STY : 3 | **
TAX : 5 | ****
TAY : 3 | **
TXA : 1 m
TYA : 4 | ***

```

```

Minimum frequency = 1
Maximum frequency = 86
Average frequency = 10

```

Unused opcodes:

BRK BVS CLD RTI SED TSX TXS

Program opcode usage: 87 %

(1.00) That's all, Folks ...

Source Code Listing
for

Apple ///

**ROM
Diagnostics**

David T. Craig
736 Edgewater
Wichita, Kansas 67230

```

0000| :*****
0000| : APPLE /// ROM - DIAGNOSTIC ROUTINES
0000| : COPYRIGHT 1979 BY APPLE COMPUTER, INC.
0000| :*****
0000|
0000| .ABSOLUTE
0000| .PROC SARATESTS
0000|
0000| ;*****
0000| ;
0000| ; SARA DIAGNOSTIC TEST ROUTINES
0000| ;
0000| ; DECEMBER 18, 1979
0000| ; BY
0000| ; W. BROEDNER & R. LASHLEY
0000| ;
0000| ; COPYRIGHT 1979 BY APPLE COMPUTER, INC.
0000| ;
0000| ;*****
0000| 0001 ROM .EQU 01
0000| 0000 ZRPG .EQU 00
0000| 0000 ZRPG1 .EQU 10
0000| 0018 PTRLO .EQU ZRPG1+08
0000| 0019 PTRHI .EQU ZRPG1+09
0000| 001A BNK .EQU ZRPG1+0A
0000| 0087 IBCMD .EQU 87
0000| 0085 IBBUFP .EQU 85
0000| 0091 PREVTRK .EQU 91
0000| F479 BLOCKIO .EQU 0F479
0000| 005D CV .EQU 5D
0000| 00FF STK0 .EQU 0FF
0000| 1419 IBNK .EQU 1400+PTRHI
0000| 1810 PHPR .EQU 1800+ZRPG1
0000| C000 KYBD .EQU 0C000
0000| C008 KEYBD .EQU 0C008
0000| C010 KBDSTRB .EQU 0C010
0000| C058 PDLEN .EQU 0C058
0000| C047 ADRS .EQU 0C047
0000| C050 GRMD .EQU 0C050
0000| C051 TXTMD .EQU 0C051
0000| C066 ADTO .EQU 0C066
0000| C0D0 DISKOFF .EQU 0C0D0
0000| C0F1 ACIAST .EQU 0C0F1
0000| C0F2 ACIACM .EQU 0C0F2
0000| C0F3 ACIACN .EQU 0C0F3
0000| C100 SLT1 .EQU 0C100
0000| C200 SLT2 .EQU 0C200
0000| C300 SLT3 .EQU 0C300
0000| C400 SLT4 .EQU 0C400
0000| CFFF EXPROM .EQU 0CFFF
0000| FFD0 ZPREG .EQU 0FFD0
0000| FFD1 SYSD1 .EQU 0FFD1
0000| FFD2 SYSD2 .EQU 0FFD2
0000| FFD3 SYSD3 .EQU 0FFD3
0000| FFE0 SYSE0 .EQU 0FFE0
0000| FFE1 BNKSW .EQU 0FFE1
0000| FFE2 SYSE2 .EQU 0FFE2
0000| FFE3 SYSE3 .EQU 0FFE3
0000| FC25 COUT .EQU 0FC25
0000| FD07 CROUT1 .EQU 0FD07
0000| FD0F KEYIN .EQU 0FD0F
0000| FBC7 SETCVH .EQU 0FBC7
0000| FD98 CLDSTRT .EQU 0FD98
0000| FD9D SETUP .EQU 0FD9D
0000| F901 MONITOR .EQU 0F901
0000| ;
0000| .ORG 0F4C5
F4C5| 00 B1 B2 BA B9 10 00 RAMTBL .BYTE 00,0B1,0B2,0BA,0B9,10,00,13
F4CC| 13
F4CD| F4CD CHPG .EQU *
F4CD| 52 41 .ASCII "RA"
F4CF| CD .BYTE 0CD ; M
F4D0| 52 4F .ASCII "RO"
F4D2| CD .BYTE 0CD ; M
F4D3| 56 49 .ASCII "VI"
F4D5| C1 .BYTE 0C1 ; A
F4D6| 41 43 49 .ASCII "ACI"
F4D9| C1 .BYTE 0C1 ; A
F4DA| 41 2F .ASCII "A/"
F4DC| C4 .BYTE 0C4 ; D
F4DD| 44 49 41 47 4E 4F 53 .ASCII "DIAGNOSTI"
F4E4| 54 49
F4E6| C3 .BYTE 0C3 ; C
F4E7| 5A .ASCII "Z"
F4E8| D0 .BYTE 0D0 ; P
F4E9| 52 45 54 52 .ASCII "RETR"
F4ED| D9 .BYTE 0D9 ; Y
F4EE| ;
F4EE| ; SETUP SYSTEM
    
```

W = Walt

Broedner
later designed
the hardware
for the
Apple IIe
computer
which was
released in
January 1983

```

F4EE|          ;
F4EE|          ;
F4EE| A9 53      LDA      #52+ROM      ; TURN OFF SCREEN, SET 2MHZ SPEED
F4F0| 8D DFFF    STA      SYSD1      ; AND RUN OFF ROM
F4F3| A2 00      LDX      #00          ; SET BANK SWITCH TO ZERO
F4F5| 8E E0FF    STX      SYSE0
F4F8| 8E EFFF    STX      BNKSW
F4FB| 8E D0FF    STX      ZPREG      ; AND SET ZERO PAGE SAME
F4FE| CA         DEX
F4FF| 8E D2FF    STX      SYSD2      ; PROGRAM DDR'S
F502| 8E D3FF    STX      SYSD3
F505| 9A         TXS
F506| E8         INX
F507| A9 0F      LDA      #0F
F509| 8D E3FF    STA      SYSE3
F50C| A9 3F      LDA      #3F
F50E| 8D E2FF    STA      SYSE2
F511| A0 0E      LDY      #0E
F513| B9 D0C0    DISK1  LDA      DISKOFF,Y
F516| 88         DEY
F517| 88         DEY
F518| 10F9       BPL      DISK1
F51A| AD 08C0    LDA      KEYBD
F51D| 29 04      AND      #04
F51F| D003       BNE      NXBYT
F521| 4C 86F6    JMP      RECON
F524|          ;
F524|          ; VERIFY ZERO PAGE
F524|          ;
F524| A9 01      NXBYT  LDA      #01          ; ROTATE A 1 THROUGH
F526| 95 00      NXBIT  STA      ZRPG,X      ; EACH BIT IN THE 0 PG
F528| D5 00      CMP      ZRPG,X      ; TO COMPLETELY TEST
F52A| D0FE       NOGOOD BNE      NOGOOD      ; THE PAGE. HANG IF NOGOOD.
F52C| 0A         ASL      A           ; TRY NEXT BIT OF BYTE
F52D| D0F7       BNE      NXBIT      ; UNTIL BYTE IS ZERO.
F52F| E8         INX          ; CONTINUE UNTIL PAGE
F530| D0F2       BNE      NXBYT      ; IS DONE.
F532| 8A         CNTWR  TXA          ; PUSH A DIFFERENT
F533| 48         PHA          ; BYTE ONTO THE
F534| E8         INX          ; STACK UNTIL ALL
F535| D0FB       BNE      CNTWR      ; STCK BYTES ARE FULL.
F537| CA         DEX          ; THEN PULL THEM
F538| 86 18     STX      PTRLO      ; OFF AND COMPARE TO
F53A| 68         PULBT  PLA          ; THE COUNTER GOING
F53B| C5 18     CMP      PTRLO      ; BACKWARDS. HANG IF
F53D| D0EB       BNE      NOGOOD      ; THEY DON'T AGREE.
F53F| C6 18     DEC      PTRLO      ; GET NEXT COUNTER BYTE
F541| D0F7       BNE      PULBT      ; CONTINUE UNTIL STACK
F543| 68         PLA          ; IS DONE. TEST LAST BYTE
F544| D0E4       BNE      NOGOOD      ; AGAINST ZERO.
F546|          ;
F546|          ; SIZE IN MEMORY
F546|          ;
F546| A2 08      NOMEM  LDX      #08          ; ZERO THE BYTES USED TO DISPLAY
F548| 95 10     STA      ZRPG1,X      ; THE BAD RAM LOCATIONS
F54A| CA         DEX          ; EACH BYTE= A CAS LINE
F54B| 10FB      BPL      NOMEM      ; ON THE SARA BOARD.
F54D| A2 02     LDX      #02          ; STARTING AT PAGE 2
F54F| 86 19     STX      PTRHI      ; TEST THE LAST BYTE
F551| A9 00     LDA      #00          ; IN EACH MEM PAGE TO
F553| A0 FF     LDY      #0FF        ; SEE IF THE CHIPS ARE
F555| 91 18     STA      (PTRLO),Y ; THERE.. (AVOID 0 & STK PAGES)
F557| D1 18     CMP      (PTRLO),Y ; CAN THE BYTE BE 0'D?
F559| F007     BEQ      NMEM2
F55B| 20 48F7   JSR      RAM          ; NO, FIND WHICH CAS IT IS.
F55E| 94 10     STY      ZRPG1,X      ; SET CORRES. BYTE TO $FF
F560| A6 19     LDX      PTRHI      ; RESTORE X REGISTER
F562| E8         INX          ; AND INCREMENT TO NEXT
F563| E0 C0     CPX      #0C0        ; PAGE UNTIL I/O IS REACHED.
F565| D0E8       BNE      NMEM1
F567| A2 20     LDX      #20          ; THEN RESET TO PAGE 20
F569| EE EFFF    INC      BNKSW      ; AND GOTO NEXT BANK TO
F56C| AD EFFF    LDA      BNKSW      ; CONTINUE. (MASK INPUTS
F56F| 29 0F     AND      #0F        ; FROM BANKSWITCH TO SEE
F571| C9 03     CMP      #03        ; WHAT SWITCH IS SET TO)
F573| D0DA       BNE      NMEM1      ; CONTINUE UNTIL BANK '3'
F575|          ;
F575|          ; SETUP SCREEN
F575|          ;
F575| 20 9DFD    ERRLP  JSR      SETUP      ; CALL SCRNM SETUP ROUTINE
F578| A2 00     LDX      #00          ; SETUP I/O AGAIN
F57A| 8E E0FF    STX      SYSE0      ; FOR VIA TEST
F57D| CA         DEX          ; PROGRAM DATA DIR
F57E| 8E D2FF    STX      SYSD2      ; REGISTERS
F581| 8E D3FF    STX      SYSD3
F584| A9 3F      LDA      #3F
F586| 8D E2FF    STA      SYSE2
F589| A9 0F      LDA      #0F
F58B| 8D E3FF    STA      SYSE3
F58E| A2 10     LDX      #10          ; HEADING OF 'DIAGNOSTICS' WITH
    
```



```

F590| 20 38F7      JSR    STRWT      ; THIS SUBROUTINE
F593| A2 00        ERRLP1 LDY    #00      ; PRINT 'RAM'
F595| 86 5D        STX    CV        ; SET CURSOR TO 2ND LINE
F597| A9 04        LDA    #04      ; SPACE CURSOR OUT 3
F599| 20 C7FB      JSR    SETCVH    ; (X STILL=0 ON RETURN)
F59C| 20 38F7      JSR    STRWT      ; THE SAME SUBROUTINE
F59F| A2 07        LDX    #07      ; FOR BYTES 7 - 0 IN
F5A1| F5A1        RAMWT1 .EQU    *
F5A1| B5 10        LDA    ZRPG1,X   ; OUT EACH BIT AS A
F5A3| A0 08        LDY    #08      ; ' ' OR '1' FOR INDICATE BAD OR MISSING RAM
F5A5| 0A          RAMWT2 ASL    A        ; CHIPS SUBROUTINE 'RAM' RAM
F5A6| 48          PHA          ; SETS UP THESE BYTES
F5A7| A9 AE        LDA    #0AE     ; LOAD A '.' TO ACC.
F5A9| 9002        BCC    RAMWT4
F5AB| A9 31        LDA    #31     ; LOAD A '1' TO ACC.
F5AD| 20 25FC      RAMWT4 JSR    COUT     ; AND PRINT IT
F5B0| 68          PLA          ; RESTORE BYTE
F5B1| 88          DEY          ; AND ROTATE ALL 8
F5B2| D0F1        BNE    RAMWT2   ; TIMES
F5B4| 20 07FD      JSR    CROUT1   ; CLEAR TO END OF LINE.
F5B7| CA          DEX          ;
F5B8| 10E7        BPL    RAMWT1
F5BA|           ;
F5BA|           ; ZPG & STK TEST
F5BA|           ;
F5BA| 9A          TXS          ;
F5BB| 8C EFFF      STY    BNKSW
F5BE| 98          ZP1    TYA          ;
F5BF| 8D D0FF      STA    ZPREG
F5C2| 85 FF      STA    STK0
F5C4| C8          INY          ;
F5C5| 98          TYA          ;
F5C6| 48          PHA          ;
F5C7| 68          PLA          ;
F5C8| C8          INY          ;
F5C9| C0 20      CPY    #20
F5CB| D0F1        BNE    ZP1
F5CD| A0 00      LDY    #00
F5CF| 8C D0FF      STY    ZPREG
F5D2| 86 18      STX    PTRLO
F5D4| E8          ZP2    INX          ;
F5D5| 86 19      STX    PTRHI
F5D7| 8A          TXA          ;
F5D8| D1 18      CMP    (PTRLO),Y
F5DA| D006        BNE    ZP3
F5DC| E0 1F      CPX    #1F
F5DE| D0F4        BNE    ZP2
F5E0| F005        BEQ    ROMTST
F5E2| F5E2        ZP3    .EQU    *      ; CHIP IS THERE, BAD ZERO AND STACK
F5E2| A2 1A      LDX    #1A     ; SO PRINT 'ZP' MESSAGE
F5E4| 20 7BF7      JSR    MESSERR  ; & SET FLAG (2MHZ MODE)
F5E7|           ;
F5E7|           ; ROM TEST ROUTINE
F5E7|           ;
F5E7| A9 00      ROMTST LDA    #00      ; SET POINTERS TO
F5E9| A8          TAY          ; $F000
F5EA| A2 F0      LDX    #0F0
F5EC| 85 18      STA    PTRLO
F5EE| 86 19      STX    PTRHI    ; SET X TO $FF
F5F0| A2 FF      LDX    #0FF    ; FOR WINDOWING I/O
F5F2| 51 18      ROMTST1 EOR    (PTRLO),Y ; COMPUTE CHKSUM ON
F5F4| E4 19      CPX    PTRHI    ; EACH ROM BYTE,
F5F6| D006        BNE    ROMTST2 ; WINDOW OUT
F5F8| C0 BF      CPY    #0BF    ; RANGES FFC0-FFEF
F5FA| D002        BNE    ROMTST2
F5FC| A0 EF      LDY    #0EF
F5FE| C8          ROMTST2 INY          ;
F5FF| D0F1        BNE    ROMTST1
F601| E6 19      INC    PTRHI
F603| D0ED        BNE    ROMTST1
F605| A8          TAY          ; TEST ACC. FOR 0
F606| F005        BEQ    VIATST   ; YES, NEXT TEST
F608| A2 03      LDX    #03     ; PRINT 'ROM' AND
F60A| 20 7BF7      JSR    MESSERR  ; SET ERROR
F60D|           ;
F60D|           ; VIA TEST ROUTINE
F60D|           ;
F60D| 18          VIATST CLC          ; SET UP FOR ADDING BYTES
F60E| D8          CLD          ;
F60F| AD E0FF     LDA    SYSE0    ; MASK OFF INPUT BITS
F612| 29 3F      AND    #3F     ; AND STORE BYTE IN
F614| 85 18      STA    PTRLO    ; TEMPOR. LOCATION
F616| AD EFFF     LDA    BNKSW    ; MASK OFF INPUT BITS
F619| 29 4F      AND    #4F     ; AND ADD TO STORED
F61B| 65 18      ADC    PTRLO    ; BYTE IN TEMP. LOC.
F61D| 6D D0FF     ADC    ZPREG    ; ADD REMAINING
F620| 85 18      STA    PTRLO    ; REGISTERS OF THE
F622| AD DFFF     LDA    SYSD1    ; VIA'S
F625| 29 5F      AND    #5F     ; (MASK THIS ONE)
F627| 65 18      ADC    PTRLO    ; AND TEST
    
```

10/31/89 9:47

HD:Apple ///:ROM - Sara Tests

Page 4

```

F629| 6D D2FF          ADC    SYSD2    ; TO SEE
F62C| 6D D3FF          ADC    SYSD3    ; IF THEY AGREE
F62F| 6D E2FF          ADC    SYSE2    ; WITH THE RESET
F632| 6D E3FF          ADC    SYSE3    ; CONDITION.
F635| C9 E1             CMP    #0E0+ROM ; =E1?
F637| F005             BEQ    ACIA     ; YES, NEXT TEST
F639| A2 06            LDX    #06      ; NO, PRINT 'VIA' MESS
F63B| 20 7BF7          JSR    MESSERR  ; AND SET ERROR FLAG
F63E|                  ;
F63E|                  ; ACIA TEST
F63E|                  ;
F63E| 18              ACIA     CLC      ; SET UP FOR ADDITION
F63F| A9 9F            LDA     LDA     #9F    ; MASK INPUT BITS
F641| 2D F1C0          AND    ACIAST   ; FROM STATUS REG
F644| 6D F2C0          ADC    ACIACM   ; AND ADD DEFAULT STATES
F647| 6D F3C0          ADC    ACIACN   ; OIF CONTROL AND COMMAND
F64A| C9 10            CMP    #10     ; REGS. =10?
F64C| F005             BEQ    ATD      ; YES, NEXT TEST
F64E| A2 09            LDX    #09     ; NO, 'ACIA' MESSAGE AND
F650| 20 7BF7          JSR    MESSERR  ; THEN SET ERROR FLAG
F653|                  ;
F653|                  ; A/D TEST ROUTINE
F653|                  ;
F653| A9 C0            ATD     LDA     #0C0   ;
F655| 8D DCFE          STA     STA     0FFDC  ;
F658| AD 5AC0          LDA     LDA     PDLEN+2 ;
F65B| AD 5EC0          LDA     LDA     PDLEN+6 ;
F65E| AD 5CC0          LDA     LDA     PDLEN+4 ;
F661| A0 20            LDY    #20     ;
F663| 88              ADCTST1 DEY     ; WAIT FOR 40 USEC
F664| D0FD            BNE    ADCTST1 ;
F666| AD 5DC0          LDA     LDA     PDLEN+5 ; SET A/D RAMP
F669| C8              ADCTST3 INY     ; COUNT FOR CONVERSION
F66A| F00A            BEQ    ADCERR   ;
F66C| AD 66C0          LDA     LDA     ADTO    ; IF BIT 7=1?
F66F| 30F8            BMI    ADCTST3 ; YES, CONTINUE
F671| 98              TYA     ;
F672| 29 E0            AND    #0E0    ; NO, MOVE COUNT TO ACC
F674| F005             BEQ    KEYPLUG  ; ACC<32
F676| F676            ADCERR .EQU    *    ; NO,
F676| A2 0D            LDX    #0D     ; PRINT 'A/D' MESS
F678| 20 7BF7          JSR    MESSERR  ; AND SET ERROR FLAG
F67B|                  ;
F67B|                  ; KEYBOARD PLUGIN TEST
F67B|                  ;
F67B| AD 08C0          KEYPLUG LDA    KEYBD  ; IS KYBD PLUGGED IN?
F67E| 0A              ASL    A        ; (IS LIGHT CURRENT
F67F| 1041             BPL    SEX      ; PRESENT?) NO, BRANCH
F681| AD DFFF          LDA     LDA     SYSD1  ; IS ERROR FLAG SET?
F684| 303C            BMI    SEX      ; ERROR HANG
F686|                  ;
F686|                  ; RECONFIGURE THE SYSTEM
F686|                  ;
F686| A9 77            RECON   LDA     #77    ; TURN ON SCREEN
F688| 8D DFFF          STA     STA     SYSD1  ;
F68B| 20 98FD          JSR    JSR     CLDSTRT ; INITIALIZE MONITOR AND DEFAULT CHARACTER SET
F68E| 2C 10C0          BIT    BIT     KBDSTRB ; CLEAR KEYBOARD
F691| AD FFCF          LDA     LDA     EXPROM  ; DISABLE ALL SLOTS
F694| AD 20C0          LDA     LDA     0C020  ;
F697| A9 10            LDA     LDA     #10    ; TEST FOR "APPLE 1"
F699| 2D 08C0          AND    AND     KEYBD   ;
F69C| D003            BNE    BNE     BOOT    ; NO, DO REGULAR BOOT
F69E| 20 01F9          JSR    JSR     MONITOR  ; AND NEVER COME BACK
F6A1| A2 01            BOOT   LDX    #01     ; READ BLOCK 0
F6A3| 86 87            STX    STX     IBCMD   ;
F6A5| CA              DEX     ;
F6A6| 86 85            STX    STX     IBBUFP  ; INTO RAM AT $A000
F6A8| A9 A0            LDA     LDA     #0A0   ;
F6AA| 85 86            STA     STA     IBBUFP+1 ;
F6AC| 4A              LSR    A        ; FOR TRACK 80
F6AD| 85 91            STA     STA     PREVTRK ; MAKE IT RECALIBRATE TOO!
F6AF| 8A              TXA     ;
F6B0| 20 79F4          JSR    JSR     BLOCKIO ;
F6B3| 900A            BCC    BCC     GOBOOT  ; IF WE'VE SUCCEEDED. DO IT UP
F6B5| A2 1C            LDX    LDX     #1C    ;
F6B7| 20 38F7          JSR    JSR     STRWT   ; 'RETRY'
F6BA| 20 0FFD          JSR    JSR     KEYIN   ;
F6BD| B0E2            BCS    BCS     BOOT    ;
F6BF| 4C 00A0          GOBOOT JMP    0A000   ; GO TO IT FOOL...
F6C2|                  ;
F6C2|                  ; SYSTEM EXERCISER
F6C2|                  ;
F6C2| A0 7F            SEX    LDY    #7F    ; TRY FROM
F6C4| 98              SEX1   TYA     ; $7F TO 0
F6C5| 29 FE            AND    AND     #0FE   ; ADD.=
F6C7| 49 4E            EOR    EOR     #4E   ; $4E OR $4F
F6C9| F003            BEQ    BEQ     SEX2   ; YES, SKP
F6CB| B9 00C0          LDA     LDA     KYBD,Y ; NO, CONT
F6CE| 88              SEX2   DEY     ; NEXT ADD
F6CF| D0F3            BNE    BNE     SEX1

```

```

F6D1| AD 51C0          LDA    TXTMD      ; SET TXT
F6D4| B9 00C1          SEX3   LDA    SLT1,Y   ; EXERCISE
F6D7| B9 00C2          LDA    SLT2,Y   ; ALL
F6DA| B9 00C3          LDA    SLT3,Y   ; SLOTS
F6DD| B9 00C4          LDA    SLT4,Y
F6E0| AD FFCF          LDA    EXPROM    ; DISABLE EXPANSION ROM AREA
F6E3| C8                INY
F6E4| D0EE          BNE    SEX3
F6E6|                ;
F6E6|                ; RAM TEST ROUTINE
F6E6|                ;
F6E6| A9 73            USRETRY LDA    #72+ROM
F6E8| 8D DFFF          STA    SYSD1
F6EB| A9 18            LDA    #18
F6ED| 8D D0FF          STA    ZPREG
F6F0| A9 00            LDA    #00
F6F2| A2 07            LDY    #07
F6F4| 95 10          RAMTST0 STA    ZRPG1,X
F6F6| CA                DEX
F6F7| 10FB          BPL    RAMTST0
F6F9| 20 84F7        JSR    RAMSET
F6FC| 08                PHP
F6FD| 20 F6F7        RAMTST1 JSR    RAMWT
F700| 20 F6F7        JSR    RAMWT
F703| 28                PLP
F704| 6A            ROR    A
F705| 08                PHP
F706| 20 A1F7        JSR    PTRINC
F709| D0F2          BNE    RAMTST1
F70B| 20 84F7        JSR    RAMSET
F70E| 08                PHP
F70F| 20 FAF7        RAMTST4 JSR    RAMRD
F712| 48                PHA
F713| A9 00            LDA    #00
F715| 91 18          STA    (PTRLO),Y
F717| 68                PLA
F718| 28                PLP
F719| 6A            ROR    A
F71A| 08                PHP
F71B| 20 A1F7        JSR    PTRINC
F71E| D0EF          BNE    RAMTST4
F720|                ;
F720|                ; RETURN TO START
F720|                ;
F720| A9 00            LDA    #00
F722| 8D EFFF          STA    BNKSW
F725| 8D D0FF          STA    ZPREG
F728| A2 07            LDY    #07
F72A| BD 1018        RAMTST6 LDA    PHPR,X
F72D| 95 10          STA    ZRPG1,X
F72F| CA                DEX
F730| 10F8          BPL    RAMTST6
F732| 20 7EF7        JSR    ERROR
F735| 4C 75F5        JMP    ERRLP
F738|                ;
F738|                ; *****
F738|                ; SARA TEST SUBROUTINES
F738|                ; *****
F738|                ;
F738| BD CDF4          STRWT  LDA    CHPG,X
F73B| 48                PHA
F73C| 09 80          ORA    #80      ; NORMAL VIDEO
F73E| 20 25FC        JSR    COUT     ; & PRINT
F741| E8                INX           ; NXT
F742| 68                PLA           ; CHR
F743| 10F3          BPL    STRWT
F745| 4C 07FD        JMP    CROUT1  ; CLR TO END OF LINE
F748|                ;
F748|                ; SUBROUTINE RAM
F748|                ;
F748| 48                RAM    PHA      ; SV ACC
F749| 8A                TXA      ; CONVRT
F74A| 4A                LSR     A      ; ADD TO
F74B| 4A                LSR     A      ; USE FOR
F74C| 4A                LSR     A      ; 8 ENTRY
F74D| 4A                LSR     A
F74E| 08                PHP
F74F| 4A                LSR     A
F750| 28                PLP
F751| AA                TAX
F752| BD C5F4        LDA    RAMTBL,X ; LOOKUP
F755| 1014          BPL    RAM0     ; IF VAL
F757| 48                PHA      ; <0, GET
F758| AD EFFF        LDA    BNKSW   ; WHICH
F75B| 29 0F        AND    #0F
F75D| AA                TAX
F75E| 68                PLA
F75F| E0 00          CPX    #00
F761| F013          BEQ    RAM1     ; BANK?
F763| 4A                LSR     A      ; SET
    
```

10/31/89 9:47

HD:Apple ///:ROM - Sara Tests

Page 6

```

F764| 4A          LSR    A          ; PROPER
F765| 4A          LSR    A          ; RAM
F766| CA          DEX    ; VAL
F767| D00D        BNE    RAM1
F769| 29 05       AND    #05       ; CONVERT
F76B| D009        BNE    RAM1       ; TO VAL
F76D| 8A          TXA
F76E| F002        BEQ    RAM00
F770| A9 03       LDA    #03
F772| 9002        BCC    RAM1
F774| 49 03       EOR    #03
F776| 29 07       AND    #07       ; BANKSW
F778| AA          TAX
F779| 68          PLA
F77A| 60          RTS
F77B|             ;
F77B|             ; SUBROUTINE ERROR
F77B|             ;
F77B| 20 38F7     MESSERR JSR    STRWT    ; PRINT MESSAGE FIRST
F77E| A9 F3       ERROR   LDA    #0F2+ROM    ; SET 1
F780| 8D DFFF     STA    SYSD1      ; MHZ MO
F783| 60          RTS
F784|             ;
F784|             ; SUBROUTINE RAMSET
F784|             ;
F784| A2 01       RAMSET  LDX    #01
F786| 86 1A       STX    BNK
F788| A0 00       LDY    #00
F78A| A9 AA       LDA    #0AA
F78C| 38          SEC
F78D| 48          RAMSET1 PHA
F78E| 08          PHP
F78F| A5 1A       LDA    BNK
F791| 09 80       ORA    #80
F793| 8D 1914     STA    IBNK
F796| A9 02       LDA    #02
F798| 85 19       STA    PTRHI
F79A| A2 00       LDX    #00
F79C| 86 18       STX    PTRLO
F79E| 28          PLS
F79F| 68          PLA
F7A0| 60          RTS
F7A1|             ;
F7A1|             ; SUBROUTINE PTRINC
F7A1|             ;
F7A1| 48          PTRINC  PHA
F7A2| E6 18       INC    PTRLO
F7A4| D01D        BNE    RETS
F7A6| A5 1A       LDA    BNK
F7A8| 100E        BPL    PINC1
F7AA| A5 19       LDA    PTRHI
F7AC| C9 13       CMP    #13
F7AE| F006        BEQ    PINC2
F7B0| C9 17       CMP    #17
F7B2| D004        BNE    PINC1
F7B4| E6 19       INC    PTRHI
F7B6| E6 19       PINC2  INC    PTRHI
F7B8| E6 19       PINC1  INC    PTRHI
F7BA| D007        BNE    RETS
F7BC| C6 1A       DEC    BNK
F7BE| C6 1A       DEC    BNK
F7C0| 20 8DF7     JSR    RAMSET1
F7C3| 68          RETS   PLA
F7C4| A6 1A       LDX    BNK
F7C6| E0 FD       CPX    #0FD
F7C8| 60          RTS
F7C9|             ;
F7C9|             ; SUBROUTINE RAMERR
F7C9|             ;
F7C9| 48          RAMERR  PHA
F7CA| A6 19       LDX    PTRHI
F7CC| A4 1A       LDY    BNK
F7CE| 3019        BMI    RAMERR4
F7D0| 8A          TXA
F7D1| 301D        BMI    RAMERR5
F7D3| 18          CLC
F7D4| 69 20       ADC    #20
F7D6| 8C EFFF     RAMERR2 STY    BNKSW
F7D9| AA          TAX
F7DA| 20 48F7     RAMERR3 JSR    RAM
F7DD| 68          PLA
F7DE| 48          PHA
F7DF| A0 00       LDY    #00
F7E1| 51 18       EOR    (PTRLO), Y
F7E3| 15 10       ORA    ZRPG1, X
F7E5| 95 10       STA    ZRPG1, X
F7E7| 68          PLA
F7E8| 60          RTS
F7E9| A9 00       RAMERR4 LDA    #00
F7EB| 8D EFFF     STA    BNKSW
    
```

10/31/89 9:47

HD:Apple ///:ROM - Sara Tests

Page 7

```

F7EE| F0EA          BEQ      RAMERR3
F7F0| 38           RAMERR5  SEC
F7F1| E9 60        SBC      #60
F7F3| C8           INY
F7F4| D0E0        BNE      RAMERR2
F7F6|              ;
F7F6|              ; SUBROUTINE RAMWT
F7F6|              ;
F7F6| 49 FF        RAMWT    EOR      #0FF
F7F8| 91 18        STA      (PTRLO),Y
F7FA| D1 18        RAMRD    CMP      (PTRLO),Y
F7FC| D0CB        BNE      RAMERR
F7FE| 60         RET1    RTS
F7FF|             .END
F7FF|
    
```

 SYMBOL TABLE DUMP

AB - Absolute LB - Label UD - Undefined MC - Macro
 RF - Ref DF - Def PR - Proc FC - Func
 PB - Public PV - Private CS - Consts

```

ACIA    LB F63E | ACIACM    AB C0F2 | ACIACN    AB C0F3 | ACIAST    AB C0F1 | ADCERR    LB F676 |
ADCTST1 LB F663 | ADCTST3    LB F669 | ADRS      AB C047 | ADTO      AB C066 | ATD        LB F653 |
BLOCKIO AB F479 | BNK        AB 001A | BNKSW     AB FFEF | BOOT      LB F6A1 | CHPG      LB F4CD |
CLDSTRT AB FD98 | CNTWR     LB F532 | COUT      AB FC25 | CROUT1    AB FD07 | CV        AB 005D |
DISK1    LB F513 | DISKOFF    AB C0D0 | ERRLP     LB F575 | ERRLP1    LB F593 | ERROR     LB F77E |
EXPR0M    AB CFFF | GOBOOT    LB F6BF | GRMD      AB C050 | IBBUFP    AB 0085 | IBCMD     AB 0087 |
IBNK     AB 1419 | KBDSTRB    AB C010 | KEYBD     AB C008 | KEYIN     AB FD0F | KEYPLUG   LB F67B |
KYBD     AB C000 | MESSERR    LB F77B | MONITOR    AB F901 | NMEM1     LB F54F | NMEM2     LB F562 |
NOGOOD    LB F52A | NOMEM     LB F548 | NXBIT     LB F526 | NXBYT     LB F524 | PDLEN     AB C058 |
PHPR     AB 1810 | PINC1     LB F7B8 | PINC2     LB F7B6 | PREVTRK   AB 0091 | PTRHI     AB 0019 |
PTRINC    LB F7A1 | PTRLO     AB 0018 | PULBT     LB F53A | RAM        LB F748 | RAM0      LB F76B |
RAM00     LB F772 | RAM1      LB F776 | RAMERR    LB F7C9 | RAMERR2    LB F7D6 | RAMERR3    LB F7DA |
RAMERR4    LB F7E9 | RAMERR5    LB F7F0 | RAMRD     LB F7FA | RAMSET    LB F784 | RAMSET1    LB F78D |
RAMTBL    LB F4C5 | RAMTST0    LB F6F4 | RAMTST1    LB F6FD | RAMTST4    LB F70F | RAMTST6    LB F72A |
RAMWT     LB F7F6 | RAMWT1    LB F5A1 | RAMWT2    LB F5A5 | RAMWT4    LB F5AD | RECON     LB F686 |
RET1     LB F7FE | RETS      LB F7C3 | ROM        AB 0001 | ROMTST    LB F5E7 | ROMTST1    LB F5F2 |
ROMTST2    LB F5FE | SARATEST   PR ---- | SETCVH    AB FBC7 | SETUP     AB FD9D | SEX        LB F6C2 |
SEX1     LB F6C4 | SEX2      LB F6CE | SEX3      LB F6D4 | SLT1      AB C100 | SLT2      AB C200 |
SLT3     AB C300 | SLT4      AB C400 | STK0      AB 00FF | STRWT     LB F738 | SYS1      AB FFD0 |
SYSD2    AB FFD0 | SYSD3     AB FFD3 | SYSE0     AB FFE0 | SYSE2     AB FFE2 | SYSE3     AB FFE3 |
TXTMD    AB C051 | USRENTY    LB F6E6 | VIATST    LB F60D | ZP1        LB F5BE | ZP2        LB F5D4 |
ZP3      LB F5E2 | ZPREG     AB FFD0 | ZRPG      AB 0000 | ZRPG1     AB 0010 |
    
```

Assembly complete: 545 lines
 0 Errors flagged on this Assembly

 6502 OPCODE STATIC FREQUENCIES

```

ADC : 10 | *****
AND : 12 | *****
ASL : 3  | ***
BCC : 3  | ***
BCS : 1  | m *
BEQ : 12 | *****
BIT : 1  | m *
BMI : 4  | ****
BNE : 31 | *****
BPL : 9  | *****
CLC : 3  | ***
CLD : 1  | m *
CMP : 10 | *****
CPX : 5  | *****
CPY : 2  | **
DEC : 3  | ***
DEX : 9  | *****
DEY : 5  | *****
EOR : 5  | *****
INC : 6  | *****
INX : 6  | *****
INY : 6  | *****
JMP : 4  | ****
JSR : 29 | *****
LDA : 56 | M *****
LDX : 24 | *****
LDY : 10 | *****
LSR : 9  | *****
ORA : 3  | ***
PHA : 11 | *****
PHP : 6  | *****
PLA : 12 | *****
PLP : 4  | ****
ROR : 2  | **
RTS : 6  | *****
SBC : 1  | m *
    
```

10/31/89 9:47

HD:Apple ///:ROM - Sara Tests

Page 8

```
SEC : 2 | **
STA : 30 | *****
STX : 18 | *****
STY : 4 | ****
TAX : 4 | ****
TAY : 2 | **
TXA : 6 | *****
TXS : 2 | **
TYA : 4 | ****
```

```
Minimum frequency = 1
Maximum frequency = 56
```

```
Average frequency = 8
```

Unused opcodes:

```
BRK BVC BVS CLI CLV NOP ROL RTI SED SEI TSX
```

```
Program opcode usage: 80 %
```

(1.00) That's all, Folks ...

Source Code Listing
for
Apple ///

ROM - Monitor

David T. Craig
736 Edgewater
Wichita, Kansas 67230

```

0000| ;*****
0000| ; APPLE /// ROM - MONITOR
0000| ; COPYRIGHT 1979 BY APPLE COMPUTER, INC.
0000| ;*****
0000|
0000| .ABSOLUTE
0000| .PROC MONITOR
0000| .ORG 0F7FE
F7FE| ;
F7FE| ;
F7FE| 60 RET1 RTS
F7FF| 3F .BYTE 03F
F800| E9 01 SBC #01
F802| F0FA BEQ RET1
F804| E9 01 SBC #01
F806| F0F6 BEQ RET1
F808| E9 01 SBC #01
F80A| F0F2 BEQ RET1
F80C| E9 01 SBC #01
F80E| F0EE BEQ RET1
F810| E9 01 SBC #01
F812| F0EA BEQ RET1
F814| E9 01 SBC #01
F816| F0E6 BEQ RET1
F818| E9 01 SBC #01
F81A| F0E2 BEQ RET1
F81C| E9 01 SBC #01
F81E| F0DE BEQ RET1
F820| E9 01 SBC #01
F822| F0DA BEQ RET1
F824| E9 01 SBC #01
F826| F0D6 BEQ RET1
F828| E9 01 SBC #01
F82A| F0D2 BEQ RET1
F82C| E9 01 SBC #01
F82E| F0CE BEQ RET1
F830| E9 01 SBC #01
F832| F0CA BEQ RET1
F834| E9 01 SBC #01
F836| F0C6 BEQ RET1
F838| E9 01 SBC #01
F83A| F0C2 BEQ RET1
F83C| E9 01 SBC #01
F83E| F0BE BEQ RET1
F840| E9 01 SBC #01
F842| F0BA BEQ RET1
F844| E9 01 SBC #01
F846| F0B6 BEQ RET1
F848| E9 01 SBC #01
F84A| F0B2 BEQ RET1
F84C| E9 01 SBC #01
F84E| F0AE BEQ RET1
F850| E9 01 SBC #01
F852| F0AA BEQ RET1
F854| E9 01 SBC #01
F856| F0A6 BEQ RET1
F858| E9 01 SBC #01
F85A| F0A2 BEQ RET1
F85C| E9 01 SBC #01
F85E| F09E BEQ RET1
F860| E9 01 SBC #01
F862| F09A BEQ RET1
F864| E9 01 SBC #01
F866| F096 BEQ RET1
F868| E9 01 SBC #01
F86A| F092 BEQ RET1
F86C| E9 01 SBC #01
F86E| F08E BEQ RET1
F870| E9 01 SBC #01
F872| F08A BEQ RET1
F874| E9 01 SBC #01
F876| F086 BEQ RET1
F878| E9 01 SBC #01
F87A| F082 BEQ RET1
F87C| E9 01 SBC #01
F87E| F002 BEQ RET3
F880| E9 01 SBC #01
F882| F07C BEQ RET2
F884| E9 01 SBC #01
F886| F078 BEQ RET2
F888| E9 01 SBC #01
F88A| F074 BEQ RET2
F88C| E9 01 SBC #01
F88E| F070 BEQ RET2
F890| E9 01 SBC #01
F892| F06C BEQ RET2
F894| E9 01 SBC #01
F896| F068 BEQ RET2
F898| E9 01 SBC #01
F89A| F064 BEQ RET2
    
```


10/31/89 10:04

HD:Apple ///:ROM - Monitor

Page 2

F89C	E9 01	SBC	#01
F89E	F060	BEQ	RET2
F8A0	E9 01	SBC	#01
F8A2	F05C	BEQ	RET2
F8A4	E9 01	SBC	#01
F8A6	F058	BEQ	RET2
F8A8	E9 01	SBC	#01
F8AA	F054	BEQ	RET2
F8AC	E9 01	SBC	#01
F8AE	F050	BEQ	RET2
F8B0	E9 01	SBC	#01
F8B2	F04C	BEQ	RET2
F8B4	E9 01	SBC	#01
F8B6	F048	BEQ	RET2
F8B8	E9 01	SBC	#01
F8BA	F044	BEQ	RET2
F8BC	E9 01	SBC	#01
F8BE	F040	BEQ	RET2
F8C0	E9 01	SBC	#01
F8C2	F03C	BEQ	RET2
F8C4	E9 01	SBC	#01
F8C6	F038	BEQ	RET2
F8C8	E9 01	SBC	#01
F8CA	F034	BEQ	RET2
F8CC	E9 01	SBC	#01
F8CE	F030	BEQ	RET2
F8D0	E9 01	SBC	#01
F8D2	F02C	BEQ	RET2
F8D4	E9 01	SBC	#01
F8D6	F028	BEQ	RET2
F8D8	E9 01	SBC	#01
F8DA	F024	BEQ	RET2
F8DC	E9 01	SBC	#01
F8DE	F020	BEQ	RET2
F8E0	E9 01	SBC	#01
F8E2	F01C	BEQ	RET2
F8E4	E9 01	SBC	#01
F8E6	F018	BEQ	RET2
F8E8	E9 01	SBC	#01
F8EA	F014	BEQ	RET2
F8EC	E9 01	SBC	#01
F8EE	F010	BEQ	RET2
F8F0	E9 01	SBC	#01
F8F2	F00C	BEQ	RET2
F8F4	E9 01	SBC	#01
F8F6	F008	BEQ	RET2
F8F8	E9 01	SBC	#01
F8FA	F004	BEQ	RET2
F8FC	E9 01	SBC	#01
F8FE	F000	BEQ	RET2
F900	60	RET2	RTS
F901			
F901			
F901	0058	SCRNLOC	.EQU 58
F901			
F901	0058	LMARGIN	.EQU SCRNLOC
F901	0059	RMARGIN	.EQU SCRNLOC+1
F901	005A	WINTOP	.EQU SCRNLOC+2
F901	005B	WINBTM	.EQU SCRNLOC+3
F901	005C	CH	.EQU SCRNLOC+4
F901	005D	CV	.EQU SCRNLOC+5
F901	005E	BAS4L	.EQU SCRNLOC+6
F901	005F	BAS4H	.EQU SCRNLOC+7
F901	0060	BAS8L	.EQU SCRNLOC+8
F901	0061	BAS8H	.EQU SCRNLOC+9
F901	0058	TBAS4L	.EQU SCRNLOC+A
F901	0063	TBAS4H	.EQU SCRNLOC+0B
F901	0064	TBAS8L	.EQU SCRNLOC+0C
F901	0065	TBAS8H	.EQU SCRNLOC+0D
F901	0066	FORGND	.EQU SCRNLOC+0E
F901	0067	BKGND	.EQU SCRNLOC+0F
F901	0068	MODES	.EQU SCRNLOC+10
F901	0069	CURSOR	.EQU SCRNLOC+11
F901	006A	STACK	.EQU SCRNLOC+12
F901	006B	PROMPT	.EQU SCRNLOC+13
F901	006C	TEMPX	.EQU SCRNLOC+14
F901	006D	TEMPY	.EQU SCRNLOC+15
F901	006E	CSWL	.EQU SCRNLOC+16
F901	006F	CSWH	.EQU SCRNLOC+17
F901	0070	KSWL	.EQU SCRNLOC+18
F901	0071	KSWH	.EQU SCRNLOC+19
F901	0072	PCL	.EQU SCRNLOC+1A
F901	0073	PCH	.EQU SCRNLOC+1B
F901	0074	A1L	.EQU SCRNLOC+1C
F901	0075	A1H	.EQU A1L+1
F901	0076	A2L	.EQU A1L+2
F901	0077	A2H	.EQU A1L+3
F901	0078	A3L	.EQU A1L+4
F901	0079	A3H	.EQU A1L+5
F901	007A	A4L	.EQU A1L+6

```

F901| 007B      A4H      .EQU  ALL+7
F901| 007C      STATE    .EQU  ALL+8
F901| 007D      YSAV     .EQU  ALL+9
F901| 007E      INBUF    .EQU  ALL+0A
F901| 0080      TEMP     .EQU  ALL+0C
F901| 0069      MASK     .EQU  CURSOR
F901|           ;
F901| C000      KBD      .EQU  0C000
F901| C010      KBDSTRB .EQU  0C010
F901|           ;
F901| 0358      USERADR  .EQU  358
F901| F479      BLOCKIO  .EQU  0F479
F901| F686      RECON   .EQU  0F686      ; AS OF 12/20/1979
F901| F4EE      DIAGN   .EQU  0F4EE
F901| 0050      INBUFLN .EQU  50      ; ONLY 80 BYTES ($3A0-$3EF)
F901| 0081      IBSLOT   .EQU  81
F901| 0082      IBDRVN  .EQU  IBSLOT+1
F901| 0085      IBBUFP  .EQU  IBSLOT+4
F901| 0087      IBCMD   .EQU  IBSLOT+6
F901|           ;
F901| F901      ENTRY   .EQU  *
F901| BA        TSX
F902| 86 6A     STX      STACK
F904| D8        MON    CLD      ; MUST BE HEX MODE
F905| 20 4EFC   JSR      BELL
F908| A6 6A     MONZ   LDX      STACK ; RESTORE STACK TO ORIGINAL LOCATION
F90A| 9A        TXS
F90B| A9 DF     LDA      #0DF   ; PROMPT (APPLE) FOR SARA MONITOR
F90D| 85 6B     STA      PROMPT
F90F| 20 D5FC   JSR      GETLNZ  ; GET A LINE OF INPUT
F912| 20 67F9   SCAN   JSR      ZSTATE ; SET REGULAR SCAN
F915| 20 2CF9   NXTINP JSR      GETNUM  ; ATTEMPT TO READ HEX BYTE
F918| 84 7D     STY      YSAV   ; STORE CURRENT INPUT POINTER
F91A| A0 12     LDY      #12   ; 18 COMMANDS
F91C| 88        CMDSRCH DEY
F91D| 30E5     BMI      MON   ; GIVE UP IF UNRECOGNIZABLE
F91F| D9 6CF9   CMP      CMDTAB,Y ; FOUND?
F922| D0F8     BNE      CMDSRCH ; NO KEEP LOOKING
F924| 20 5EF9   JSR      TOSUB  ; PERFORM FUNCTION
F927| A4 7D     LDY      YSAV   ; GET NEXT POINTER
F929| 4C 15F9   JMP      NXTINP ; DO NEXT COMMAND
F92C|           ;
F92C| A2 00     GETNUM  LDX      #00   ; CLEAR A2
F92E| 86 76     STX      A2L
F930| 86 77     STX      A2H
F932| B1 7E     NXTCHR  LDA      (INBUF),Y
F934| C8        INY
F935| 49 B0     EOR      #0B0
F937| C9 0A     CMP      #0A   ; TEST FOR DIGIT
F939| 9006     BCC      DIGIT  ; SAVE IT IF 1-9
F93B| 69 88     ADC      #88   ; TEST FOR HEX A-F
F93D| C9 FA     CMP      #0FA
F93F| 902A     BCC      DIGRET
F941| A2 03     DIGIT  LDX      #03
F943| 0A       ASL      A
F944| 0A       ASL      A
F945| 0A       ASL      A
F946| 0A       ASL      A
F947| 0A       NXTBIT  ASL      A ; SHIFT HEX DIGITS INTO A2
F948| 26 76     ROL      A2L
F94A| 26 77     ROL      A2H
F94C| CA       DEX
F94D| 10F8     BPL      NXTBIT ; SHIFTED ALL YET?
F94F| A5 7C     NXTBAS  LDA      STATE
F951| D006     BNE      NXTBS2 ; IF ZERO THEN COPY TO A1,3
F953| B5 77     LDA      A2H,X
F955| 95 75     STA      A1H,X
F957| 95 79     STA      A3H,X
F959| E8        NXTBS2  INX
F95A| F0F3     BEQ      NXTBAS
F95C| D0D4     BNE      NXTCHR
F95E|           ; SWITCH ROUTINE FOR CHARACTER
F95E|           ;
F95E| A9 FA     TOSUB   LDA      #0FA   ; PUSH ADDRESS OR FUNCTION
F960| 48        PHA
F961| B9 7DF9   LDA      CMDVEC,Y ; AND RETURN IT
F964| 48        PHA
F965| A5 7C     LDA      STATE ; PASS MODE VIA ACC.
F967| A0 00     ZSTATE  LDY      #00
F969| 84 7C     STY      STATE ; RESET STATE OF SCAN
F96B| 60       DIGRET  RTS
F96C| F96C     CMDTAB  .EQU  *
F96C| 00       .BYTE  00   ; G =GP (CALL) SUBROUTINE
F96D| 03       .BYTE  03   ; J =JUMP (CONT) PROGRAM
F96E| 06       .BYTE  06   ; M =MOVE MEMORY
F96F| EB       .BYTE  EB   ; R =READ DISK BLOCK
F970| EC       .BYTE  EC   ; S =MEMORY SEARCH
F971| EE       .BYTE  EE   ; U =USER FUNCTION
F972| EF       .BYTE  EF   ; V =VERIFY MEMORY BLOCKS
    
```

```

F973| F0          .BYTE 0F0      ; W  =WRITE DISK BLOCK
F974| F1          .BYTE 0F1      ; X  =REPEAT COMMAND LINE
F975| 99          .BYTE 99       ; SP =SPACE (BYTE SEPARATOR)
F976| 9B          .BYTE 9B       ; "  =ASCII (HI BIT ON)
F977| A0          .BYTE 0A0      ; '  =ASCII (HI BIT OFF)
F978| 93          .BYTE 93       ; :  =SET STORE MODE
F979| A7          .BYTE 0A7      ; .  =RANGE SEPARATOR
F97A| A8          .BYTE 0A8      ; /  =COMMAND SEPARATOR
F97B| 95          .BYTE 95       ; <  =DEST/SOURCE SEPARATOR
F97C| C6          .BYTE 0C6      ; CR =CARRIAGE RETURN
F97D|             ;
F97D| F97D        CMDVEC    .EQU *
F97D| 90          .BYTE 90       ; GO-1
F97E| 8E          .BYTE 8E       ; JUMP-1
F97F| 3F          .BYTE 3F       ; MOVE-1
F980| D3          .BYTE 0D3      ; READ-1
F981| 08          .BYTE 08       ; SEARCH-1
F982| 8B          .BYTE 8B       ; USER-1
F983| 4E          .BYTE 4E       ; VRFY-1
F984| D6          .BYTE 0D6      ; WRTE-1
F985| 2C          .BYTE 2C       ; REPEAT-1
F986| B7          .BYTE 0B7      ; SPCE-1
F987| 1A          .BYTE 1A       ; ASCII-1
F988| 1C          .BYTE 1C       ; ASCII0-1
F989| CB          .BYTE 0CB      ; SETMODE-1
F98A| CB          .BYTE 0CB      ; SETMODE-1
F98B| AD          .BYTE 0AD      ; SEP-1
F98C| A4          .BYTE 0A4      ; DEST-1
F98D| 39          .BYTE 39       ; CRMON-1
F98E|             ;
F98E|             ;
F98E| E6 7A        NXTA4    INC  A4L      ; BUMP 16 BIT POINTERS
F990| D002        BNE  NXTA1
F992| E6 7B        INC  A4H
F994| E6 74        NXTA1    INC  A1L      ; BUMP A1
F996| D005        BNE  TSTA1
F998| E6 75        INC  A1H
F99A| 38          SEC
F99B| F010        BEQ  RETA1    ; IN CASE OF ROLL OVER
F99D| A5 74        TSTA1    LDA  A1L
F99F| 38          SEC
F9A0| E5 76        SBC  A2L
F9A2| 85 80        STA  TEMP
F9A4| A5 75        LDA  A1H
F9A6| E5 77        SBC  A2H
F9A8| 05 80        ORA  TEMP
F9AA| D001        BNE  RETA1    ; IF A1 LESS THAN OR EQUAL TO A2
F9AC| 18          CLC
F9AD| 60          RETA1    RTS
F9AE|             ;
F9AE|             ;
F9AE| 48          PRBYTE    PHA      ; SAVE LOW NIBBLE
F9AF| 4A          LSR  A
F9B0| 4A          LSR  A      ; SHIFT HI NIBBLE TO PRINT.
F9B1| 4A          LSR  A
F9B2| 4A          LSR  A
F9B3| 20 B9F9     JSR  PRHEXZ
F9B6| 68          PLA
F9B7| 29 0F        PRHEX    AND  #0F     ; STRIP HI NIBBLE
F9B9| 09 B0        PRHEXZ   ORA  #0B0    ; MAKE IT NUMERIC
F9BB| C9 BA        CMP  #0BA    ; IS IT '>'9'
F9BD| 9002        BCC  PRHEX2
F9BF| 69 06        ADC  #06     ; MAKE IT 'A'-'F'
F9C1| 4C 39FC     PRHEX2   JMP  COUT
F9C4|             ;
F9C4| 20 AEF9     PRBYCOL  JSR  PRBYTE
F9C7|             ;
F9C7| A9 BA        PRCOLON  LDA  #0BA    ; PRINT A COLON
F9C9| D0F6        BNE  PRHEX2   ; BRANCH ALWAYS
F9CB|             ;
F9CB| A9 07        TST80WID  LDA  #07     ; ANTICIPATE
F9CD| 24 68        BIT  MODES   ; TEST FOR 80
F9CF| 5002        BVC  SVMASK
F9D1| A9 0F        LDA  #0F
F9D3| 85 69        SVMASK   STA  MASK
F9D5| 60          RTS
F9D6|             ;
F9D6| 8A          A1PC     TXA      ; TEST FOR NEW PC
F9D7| F007        BEQ  OLDPC
F9D9| B5 74        A1PC1    LDA  A1L,X
F9DB| 95 72        STA  PCL,X
F9DD| CA          DEX
F9DE| 10F9       BPL  A1PC1
F9E0| 60          OLDPC    RTS
F9E1|             ;
F9E1| 85 69        ASCII1   STA  MASK    ; SAVE HI BIT STATUS
F9E3| A4 7D        ASCII2   LDY  YSAV   ; MOVE ASCII TO MEMORY
F9E5| B1 7E        LDA  (INBUF),Y
F9E7| E6 7D        INC  YSAV   ; BUMP FOR NEXT THING.
F9E9| A0 00        LDY  #00
    
```

F9EB	C9 A2		CMP	#0A2		; ASCII " ?
F9ED	D005		BNE	ASCII3		; NOPE, CONTINUE.
F9EF	A5 69		LDA	MASK		
F9F1	1032		BPL	BITON		; HE'S CHANGED MODES.
F9F3	60		RTS			
F9F4	C9 A7	ASCII3	CMP	#0A7		; ASCII ' ?
F9F6	D005		BNE	CRCHK		; NO, TEST FOR EOL.
F9F8	A5 69		LDA	MASK		
F9FA	302D		BMI	BITOFF		; CHANGE MODES.
F9FC	60		RTS			
F9FD						
F9FD	C9 8D	CRCHK	CMP	#8D		; END OF LINE?
F9FF	F007		BEQ	ASCDONE		; YES, FINISHED
FA01	25 69		AND	MASK		
FA03	20 C3FA		JSR	STOR1		; GO STORE IT!
FA06	D0DB		BNE	ASCII2		; DO NEXT.
FA08	60	ASCDONE	RTS			
FA09						
FA09						
FA09	B1 74	SEARCH	LDA	(A1L),Y		; LOAD SEARCH BYTE
FA0B	C5 7A		CMP	A4L		
FA0D	D006		BNE	SRCH1		
FA0F	20 75FA		JSR	PRINTA1		; DUMP MEMORY
FA12	20 EFFC		JSR	CROUT		
FA15	20 94F9	SRCH1	JSR	NXTA1		; INCREMENT POINTER
FA18	90EF		BCC	SEARCH		; CONTINUE SEARCH
FA1A	60		RTS			; RETURN
FA1B						
FA1B						
FA1B	38	ASCII	SEC			; INDICATE HI ON.
FA1C	90		.BYTE	90		; (BCC - NEVER TAKEN)
FA1D	18	ASCII0	CLC			; INDICATE HI OFF
FA1E	AA	CKMDE	TAX			; SAVE STATE
FA1F	86 7C		STX	STATE		; RETAIN STATE
FA21	49 BA		EOR	#0BA		; ARE WE IN STORE MODE?
FA23	D07D		BNE	ERROR		
FA25	A9 FF	BITON	LDA	#0FF		; SET HI BIT UNMASKED
FA27	B0B8		BCS	ASCII1		
FA29	A9 7F	BITOFF	LDA	#7F		; MASK HI BIT
FA2B	10B4		BPL	ASCII1		; ALWAYS BRANCHES
FA2D	2C 00C0	REPEAT	BIT	KBD		; REPEAT UNTIL KEYPRESS
FA30	1003		BPL	REPEAT1		
FA32	4C 0FFD		JMP	KEYIN		
FA35	68	REPEAT1	PLA			; CLEAN UP STACK
FA36	68	LFA36	PLA			
FA37	4C 12F9		JMP	SCAN		
FA3A						
FA3A						
FA3A	20 B4FA	CRMON	JSR	BL1		
FA3D	4C 08F9		JMP	MONZ		
FA40						
FA40						
FA40	20 9DF9	MOVE	JSR	TSTA1		; TEST VALID RANGE
FA43	B05D		BCS	ERROR		
FA45	B1 74	MOVNXT	LDA	(A1L),Y		; COMPARE BYTE FOR BYTE
FA47	91 7A		STA	(A4L),Y		
FA49	20 8EF9		JSR	NXTA4		; BUMP BOTH A1 AND A4
FA4C	90F7		BCC	MOVNXT		
FA4E	60		RTS			; ALL DONE WITH MOVE
FA4F						
FA4F						
FA4F	20 9DF9	VRFY	JSR	TSTA1		; TEST VALID RANGE
FA52	B04E		BCS	ERROR		
FA54	B1 74	VRFY1	LDA	(A1L),Y		; COMPARE BYTE FOR BYTE
FA56	D1 7A		CMP	(A4L),Y		; MATCH?
FA58	F006		BEQ	VRFY2		; YES, DO NEXT.
FA5A	20 66FA		JSR	MISMATCH		; PRINT BOTH BYTES
FA5D	20 EFFC		JSR	CROUT		; GOTO NEWLINE
FA60	20 8EF9	VRFY2	JSR	NXTA4		; BUMP BOTH A1 AND A4
FA63	90EF		BCC	VRFY1		
FA65	60		RTS			; VERIFY DONE.
FA66						
FA66	A5 7B	MISMATCH	LDA	A4H		; PRINT ADDRESS OF A4
FA68	20 AEF9		JSR	PRBYTE		
FA6B	A5 7A		LDA	A4L		
FA6D	20 C4F9		JSR	PRBYCOL		; OUTPUT A COLON FOR SEPARATOR
FA70	B1 7A		LDA	(A4L),Y		; AND THE DATA
FA72	20 84FA		JSR	PRBYTSP		; PRINT THE BYTE AND A SPACE
FA75	20 87FA	PRINTA1	JSR	PRSPC		; LEAD WITH A SPACE
FA78	A5 75		LDA	A1H		; OUTPUT ADDRESS A1
FA7A	20 AEF9		JSR	PRBYTE		
FA7D	A5 74		LDA	A1L		
FA7F	20 C4F9		JSR	PRBYCOL		; SEPARATE WITH A COLON
FA82	B1 74	PRA1BYTE	LDA	(A1L),Y		; PRINT BYTE POINTED TO BY A1
FA84	20 AEF9	PRBYTSP	JSR	PRBYTE		
FA87	A9 A0	PRSPC	LDA	#0A0		; PRINT A SPACE
FA89	4C 39FC		JMP	COUT		; END VIA OUTPUT ROUTINE.
FA8C						
FA8C	4C 5803	USER	JMP	USERADR		
FA8F						

FA8F	68	JUMP	PLA		
FA90	68		PLA		; LEAVE STACK WITH NOTHIN' ON IT.
FA91	20 D6F9	GO	JSR	A1PC	; STUFF PROGRAM COUNTER
FA94	6C 7200		JMP	@PCL	; JUMP TO USER PROG.
FA97					
FA97	FA97	RWERROR	.EQU	*	; PRINT ERROR NUMBER
FA97	20 AEF9		JSR	PRBYTE	; PRINT THE OFFENDER
FA9A	A9 A1		LDA	#0A1	; FOLLOWED BY A "!"
FA9C	20 39FC		JSR	COUT	
FA9F	20 07FD	ERROR2	JSR	NOSTOP	; OUTPUT A CARRIAGE RETURN (NO STOPLST)
FAA2	4C 04F9	ERROR	JMP	MON	
FAA5					
FAA5	A5 76		LDA	A2L	; COPY A2 TO A4 FOR DESTINATION OP
FAA7	85 7A	DEST	STA	A4L	
FAA9	A5 77		LDA	A2H	
FAAB	85 7B		STA	A4H	
FAAD	60		RTS		
FAAE					
FAAE	20 B8FA	SEP	JSR	SPCE	; SEPARATOR TEST STORE MODE OR DUMP.
FAB1	98		TYA		; ZERO MODE.
FAB2	F01D		BEQ	SETMDZ	; BRANCH ALWAYS
FAB4					
FAB4	C6 7D	BL1	DEC	YSAV	; TEST FOR NO LINE
FAB6	F045		BEQ	DUMP8	; IF NO LINE, GIVEM A ROW OF BYTES
FAB8	CA	SPCE	DEX		; TEST IF AFTER ANOTHER SPACE
FAB9	D016		BNE	SETMDZ	
FABB	C9 BA		CMP	#0BA	; STORE MODE?
FABD	D04B		BNE	TSTDUMP	
FABF	85 7C	STOR	STA	STATE	; KEEP IT IN STORE STATE
FAC1	A5 76		LDA	A2L	; GET BYTE TO BE STORED
FAC3	91 78	STOR1	STA	(A3L), Y	; PUT IT IN MEMORY.
FAC5	E6 78		INC	A3L	; BUMP POINTER
FAC7	D002		BNE	DUMMY	
FAC9	E6 79		INC	A3H	
FACB	60	DUMMY	RTS		; ALSO USED FOR '/' TO CLEAR MODE
FACC					
FACC	A4 7D	SETMODE	LDY	YSAV	; USE INPUT CHARACTER
FACE	88		DEY		
FACF	B1 7E		LDA	(INBUF), Y	; TO SET MODE
FAD1	85 7C	SETMDZ	STA	STATE	
FAD3	60		RTS		
FAD4					
FAD4	A9 01	READ	LDA	#01	; GET DISK COMMAND TO READ
FAD6	2C		.BYTE	2C	; DUMMY BIT TO SKIP 2 BYTES
FAD7	A9 02	WRTE	LDA	#02	; SET DISK COMMAND TO WRITE
FAD9	85 87	SAVCMD	STA	IBCMD	
FADB	A5 74	RWLOOP	LDA	A1L	
FADD	85 85		STA	IBBUFP	; COMMAND FORMAT IS
FADF	A5 75		LDA	A1H	; BLOCKNUMBER <ADDRESS END ADDRESS
FAE1	85 86		STA	IBBUFP+1	
FAE3	A6 7B		LDX	A4H	; SEND BLOCK NUMBER VIA X & A
FAE5	A5 7A		LDA	A4L	
FAE7	78		SEI		; NO INTERRUPTS WHILE IN MONITOR
FAE8	20 79F4		JSR	BLOCKIO	; DO DISKO FEVER
FAEB	B0AA		BCS	RWERROR	; GIVE UP IF ERROR ENCOUNTERED
FAED	E6 7A		INC	A4L	; BUMP BLOCK NUMBER
FAEF	D002		BNE	NOVER	
FAF1	E6 7B		INC	A4H	
FAF3	E6 75	NOVER	INC	A1H	; BUMP RAM ADDRESS BY 512 BYTES
FAF5	E6 75		INC	A1H	
FAF7	20 9DF9		JSR	TSTA1	; TEST FOR FINISHED
FAFA	90DF		BCC	RWLOOP	; NOT DONE, DO NEXT BLOCK
FAFC	60		RTS		
FAFD					
FAFD	A5 75	DUMP8	LDA	A1H	
FAFF	85 77		STA	A2H	
FB01	20 CBF9		JSR	TST80WID	; GET WIDTH MASK INTO ACC
FB04	05 74		ORA	A1L	
FB06	85 76		STA	A2L	
FB08	D006		BNE	DUMP0	; BRANCH ALWAYS
FB0A					
FB0A	4A	TSTDUMP	LSR	A	; DUMP?
FB0B	B095	ERROR1	BCS	ERROR	
FB0D	20 CBF9	DUMP	JSR	TST80WID	; SET FOR EITHER 80 OR 40 COLUMNS
FB10	A5 74	DUMP0	LDA	A1L	
FB12	85 7A		STA	A4L	
FB14	A5 75		LDA	A1H	
FB16	85 7B		STA	A4H	
FB18	20 9DF9		JSR	TSTA1	; TEST FOR VALID RANGE
FB1B	B0EE		BCS	ERROR1	
FB1D	20 75FA	DUMP1	JSR	PRINTA1	; PRINT ADDRESS AND FIRST BYTE
FB20	20 94F9	DUMP2	JSR	NXTA1	
FB23	B010		BCS	DUMPASC	; END WITH ASCII
FB25	A5 74		LDA	A1L	; TEST END OF LINE
FB27	25 69		AND	MASK	; FOR 40/80 COLUMN
FB29	D005		BNE	DUMP3	
FB2B	20 35FB		JSR	DUMPASC	
FB2E	D0ED		BNE	DUMP1	; BRANCH ALWAYS
FB30	20 82FA	DUMP3	JSR	PRA1BYTE	; GO PRINT NEXT BYTE AND A SPACE
FB33	D0EB		BNE	DUMP2	; ALWAYS (ACC JUST PULLED AS \$A0)

FB35					
FB35	A5 7A	DUMPASC	LDA	A4L	; RESET TO BEGINNING OF LINE
FB37	85 74		STA	ALL	
FB39	A5 7B		LDA	A4H	
FB3B	85 75		STA	A1H	
FB3D	20 87FA		JSR	PRSPC	; PRINT AN EXTRA SPACE
FB40	A0 00	ASC1	LDY	#00	; TO INDEX MEMORY INDIRECT
FB42	B1 74		LDA	(ALL),Y	
FB44	09 80		ORA	#80	; SET NORMAL VIDEO
FB46	C9 A0		CMP	#0A0	; TEST FOR CONTROL CHARACTERS
FB48	B002		BCS	ASC2	; OK TO PRINT NON CONTROLS
FB4A	A9 AE		LDA	#0AE	; OTHERWISE PRINT A SPACE
FB4C	20 39FC	ASC2	JSR	COUT	; PUT IT OUT
FB4F	20 8EF9		JSR	NXTA4	; BUMP BOTH A1 AND A4
FB52	B006		BCS	ASC3	; FINISHED
FB54	A5 74		LDA	ALL	; TEST END OF LINE
FB56	25 69		AND	MASK	
FB58	D0E6		BNE	ASC1	; NOT DONE, PRINT NEXT
FB5A	4C EFFC	ASC3	JMP	CROUT	
FB5D					
FB5D					
FB5D	38	COL80	SEC		; INDICATE 80 COLUMNS
FB5E	AD 53C0		LDA	0C053	; GOTO 80 COLUMN MODE
FB61	B004		BCS	SET80	; BRANCH ALWAYS
FB63					
FB63	18	COL40	CLC		; INDICATE 40 COLUMNS DESIRED
FB64	AD 52C0		LDA	0C052	; GOTO 40 COLUMN MODE
FB67	A5 68	SET80	LDA	MODES	
FB69	09 40		ORA	#40	; ASSUME 80
FB6B	B002		BCS	SET80A	; AND BRANCH IF IT IS
FB6D	29 BF		AND	#0BF	; BUT FIX FOR 40 IF NOT
FB6F	85 68	SET80A	STA	MODES	
FB71	09 7F		ORA	#7F	; ISOLATE BIT 7
FB73	29 A0		AND	#0A0	; (BIT 7 SETS NORMAL/INVERSE)
FB75	85 66		STA	FORGND	
FB77	B002		BCS	SET80B	; AGAIN ASSUMES 80 COLUMNS
FB79	A9 F0		LDA	#0F0	; IF NOT, SET FOR/BACKGROUND COLOR
FB7B	85 67	SET80B	STA	BKGND	
FB7D					
FB7D	A5 58	CLSCRN	LDA	LMARGIN	; SET CURSOR TO TOP LEFT OF WINDOW
FB7F	85 5C		STA	CH	
FB81	A5 5A		LDA	WINTOP	
FB83	85 5D		STA	CV	; NOW DROP INTO CLEAR END OF PAGE
FB85					
FB85	A5 5C	CLEOP	LDA	CH	; SAVE CURRENT CURSOR POSITION
FB87	48		PHA		
FB88	A5 5D		LDA	CV	
FB8A	48		PHA		
FB8B	20 C5FB		JSR	SETCV	
FB8E	20 A2FB	CLEOP1	JSR	CLEOL	; CLEAR TO END OF FIRST LINE
FB91	A5 58		LDA	LMARGIN	
FB93	85 5C		STA	CH	
FB95	20 DDFB		JSR	CURDOWN	; GOTO NEXT LINE
FB98	90F4		BCC	CLEOP1	
FB9A	68		PLA		
FB9B	A8		TAY		
FB9C	68		PLA		; RESTORE CURSOR POSITION
FB9D	85 5C		STA	CH	
FB9F	98		TYA		; GET OLD CV IN ACC AGAIN
FBA0	B023		BCS	SETCV	; BRANCH ALWAYS
FBA2					
FBA2	A5 5C	CLEOL	LDA	CH	; CLEAR TO END OF LINE FIRST
FBA4	4C 89FC		JMP	CLEOL1	
FBA7					
FBA7	C9 80	CONTROL	CMP	#80	
FBA9	9065		BCC	DISPLAYX	; IF INVERSE
FBAB	C9 8D	TSTCR	CMP	#8D	; IF CARRIAGE RETURN THEN NEW LINE
FBAD	D03A		BNE	TSTBACK	
FBAF	20 A2FB	CARRAGE	JSR	CLEOL	; FIRST CLEAR TO THE END OF THIS LINE
FBB2	20 D7FB		JSR	SETCHZ	; RESET CURSOR AND GOTO NEXT LINE (CARRY IS SET)
FBB5	4C 16FC		JMP	NXTLIN	; THEN GOTO THE NEXT LINE.
FBB8					
FBB8					
FBB8	A5 5D	CURUP	LDA	CV	; TEST FOR TOP OF SCREEN
FBBA	C6 5D		DEC	CV	; ANTICIPATE 'NOT' TOP
FBBC	C5 5A		CMP	WINTOP	
FBBE	D002		BNE	CURUP1	; IT'S NOT TOP, CONTINUE
FBC0	A5 5B		LDA	WINBTM	; WRAP AROUND TO BOTTOM
FBC2	38	CURUP1	SEC		; DECREMENT BY ONE
FBC3	E9 01		SBC	#01	
FBC5	85 5D	SETCV	STA	CV	; SAVE NEW VERTICAL LINE
FBC7	FBC7	BASCALC	.EQU	*	
FBC7	FBC7	CURDN1	.EQU	*	
FBC7	A5 5D		LDA	CV	; GET VALUES FOR FIRST PAGE (\$400)
FBC9	104E		BPL	BASCALC1	; ALWAYS
FBCB					
FBCB	24 68	CURIGHT	BIT	MODES	; TEST FOR 80 OR 40
FBCD	7002		BVS	RIGHT1	
FBCF	E6 5C		INC	CH	
FBD1	E6 5C	RIGHT1	INC	CH	; BUMP CURSOR HORIZONTAL

10/31/89 10:04

HD:Apple ///:ROM - Monitor

Page 8

```

FBD3| A5 5C          LDA      CH          ; TEST FOR NEW LINE
FBD5| C5 59          CMP      RMARGIN
FBD7| A5 58          SETCHZ  LDA      LMARGIN      ; JUST IN CASE WE HAVE.
FBD9| 905D          BCC     CTRLRET
FBD8| 85 5C          SETCVH  STA      CH          ; CURSOR AT START OF NEXT LINE
FBDD|              ; DROP INTO CURDOWN FOR WRAP AROUND
FBDD|              ;
FBDD| E6 5D          CURDOWN INC     CV          ; MOVE CURSOR DOWN ONE LINE
FBDF| A5 5D          LDA      CV          ; ANTICIPATE NOT BOTTOM
FBE1| C5 5B          CMP     WINBTM      ; TEST FOR BOTTOM
FBE3| 90E2          BCC     CURDN1
FBE5| A5 5A          LDA     WINTOP
FBE7| B0DC          BCS     SETCV      ; BRANCH ALWAYS
FBE9|              ;
FBE9| C9 88          TSTBACK CMP    #88         ; BACKSPACE?
FBEB| D05D          BNE     TSTBELL
FBED| 24 68          CURLEFT BIT    MODES      ; TEST FOR FOURTY OR EIGHTY MODE
FBEF| 7002          BVS     LEFT80
FBF1| C6 5C          BFBF1  DEC     CH
FBF3| C6 5C          BFBF3  DEC     CH
FBF5| 3006          BFBF5  BMI    LEFTUP
FBF7| A5 5C          BFBF7  LDA     CH          ; TEST FOR WRAP AROUND
FBF9| C5 58          BFBF9  CMP    LMARGIN
FBFB| 103B          BFBFB  BPL    CTRLRET
FBFD| 20 B8FB        BFBFD  JSR    CURUP
FC00| A5 59          FC000  LDA     RMARGIN
FC02| 85 5C          FC002  STA     CH          ; SAVE NEW CURSOR POSITION
FC04| D0E7          FC004  BNE     CURLEFT      ; BRANCH ALWAYS
FC06|              ;
FC06| C9 A0          FC006  COUT2  CMP    #0A0        ; IS IT CONTROL CHARACTER
FC08| 909D          FC008  BCC     CONTROL
FC0A| 24 68          FC00A  BIT    MODES      ; TEST FOR INVERSE
FC0C| 3002          FC00C  BMI    DISPLAYX  ; NO PUT IT OUT
FC0E| 29 7F          FC00E  AND    #7F         ; STRIP HI BIT
FC10| 20 9DFC        FC100  DISPLAYX JSR   DISPLAY
FC13|              ;
FC13| 20 CFBF        FC103  INCHORZ JSR   CURRIGHT      ; MOVE CURSOR RIGHT
FC16| B043          FC106  NXTLIN  BCS    SCROLL      ; IT'S BOTTOM, RESET CH=0 AND SCROLL
FC18| 60           FC108  RTS     ; RESET CH ONLY
FC19| 08           FC109  BASCALC1 PHP   ; CALC BASE ADR IN BAS4L,H
FC1A| 48           FC10A  PHA
FC1B| 4A           FC10B  LSR    A          ; FOR GIVEN LINE NO.
FC1C| 29 03        FC10C  AND    #03         ; 0<=LINE NO.<$17
FC1E| 09 04        FC10E  ORA    #04         ; ARG=000ABCDE, GENERATE
FC20| 85 5F        FC110  STA    BAS4H      ; BAS4H=000001CD
FC22| 49 0C        FC112  EOR    #0C
FC24| 85 61        FC114  STA    BAS8H
FC26| 68           FC116  PLA          ; AND
FC27| 29 18        FC118  AND    #18         ; BAS4L=EABAB000
FC29| 9002        FC11A  BCC    BSCLC2
FC2B| 69 7F        FC11C  ADC    #7F
FC2D| 85 5E        FC11E  BSCLC2 STA  BAS4L
FC2F| 0A           FC120  ASL    A
FC30| 0A           FC122  ASL    A
FC31| 05 5E        FC124  ORA    BAS4L
FC33| 85 5E        FC126  STA    BAS4L
FC35| 85 60        FC128  STA    BAS8L      ; SAME FOR PAGE 2
FC37| 28           FC130  PLP
FC38| 60           FC132  CTRLRET RTS
FC39|              ;
FC39| 48           FC134  COUT   PHA          ; SAVE CHARACTER
FC3A| 84 6D        FC136  STY
FC3C| 86 6C        FC138  STX
FC3E| 20 47FC      FC13A  JSR    COUT1
FC41| A4 6D        FC13C  LDY
FC43| A6 6C        FC13E  LDY
FC45| 68           FC140  PLA
FC46| 60           FC142  RTS
FC47| 6C 6E00     FC144  COUT1  JMP    @CSWL      ; NORMALLY COUT1
FC4A|              ;
FC4A| C9 87          FC146  TSTBELL CMP   #87         ; BELL?
FC4C| D004          FC148  BNE     LNFD        ; NO TEST FOR FORM FEED
FC4E| AE 40C0      FC14A  BELL   LDX    #0C040   ; SOUND BELL
FC51| 60           FC14C  RTS
FC52| C9 8A          FC14E  LNFD   CMP    #8A         ; LINE FEED?
FC54| D0E2          FC150  BNE     CTRLRET
FC56| 20 DDFB      FC152  JSR    CURDOWN    ; MOVE CURSOR DOWN A LINE
FC59| 90DD          FC154  BCC    CTRLRET    ; BRANCH IF NO SCROLL NECESSARY.
FC5B|              ;
FC5B| A5 5A          FC156  SCROLL  LDA    WINTOP      ; START WITH TOP LINE
FC5D| 48           FC158  PHA          ; SAVE IT FOR NOW
FC5E| 20 C5FB      FC15A  JSR    SETCV      ; GET BASCALC FOR THIS LINE
FC61| A2 03          FC15C  SCRL1  LDX    #03         ; MOVE CURRENT BASCALC AS DESTINATION
FC63| B5 5E          FC15E  SCRL2  LDA    BAS4L,X
FC65| 95 58        FC160  STA    TBAS4L,X   ; (TEMPORARY BASE ADDR.)
FC67| CA           FC162  DEX
FC68| 10F9         FC164  BPL    SCRL2
FC6A| 68           FC166  PLA          ; GET DESTINATION LINE
FC6B| 18           FC168  CLC
    
```

FC6C	69 01		ADC	#01		; CALCULATE SOURCE LINE.
FC6E	C5 5B		CMP	WINBTM		; IS IT THE LAST LINE?
FC70	B015		BCS	LASTLN		; YES, CLEAR IT
FC72	48		PHA			; SAVE AS NEXT DESTINATION LINE
FC73	20 C5FB		JSR	SETCV		; GET BASE ADDR FOR SOURCE LINE
FC76	A5 59		LDA	RMARGIN		; MOVE SOURCE TO DESTINATION
FC78	4A		LSR	A		; DIVIDE BY 2
FC79	A8		TAY			
FC7A	88		DEY			; DONE YET
FC7B	30E4	SCRL3	BMI	SCRL1		; YES, DO NEXT LINE
FC7D	B1 5E		LDA	(BAS4L), Y		
FC7F	91 58		STA	(TBAS4L), Y		
FC81	B1 60		LDA	(BAS8L), Y		
FC83	91 64		STA	(TBAS8L), Y		
FC85	90F3		BCC	SCRL3		; BRANCH ALWAYS
FC87	A5 58	LASTLN	LDA	LMARGIN		; BLANK FILL THE LAST LINE
FC89	4A	CLEOL1	LSR	A		; DIVIDE BY 2
FC8A	A8		TAY			
FC8B	B004		BCS	CLEOL2		
FC8D	A5 66		LDA	FORGND		; (NORMALLY A SPACE)
FC8F	91 5E		STA	(BAS4L), Y		
FC91	A5 67	CLEOL2	LDA	BKGND		; (IF 80 COLUMNS, ALSO A SPACE)
FC93	91 60		STA	(BAS8L), Y		
FC95	C8		INY			
FC96	98		TYA			; TEST FOR END OF LINE
FC97	0A		ASL	A		; MULT BY 2 AGAIN
FC98	C5 59		CMP	RMARGIN		
FC9A	90ED		BCC	CLEOL1		; CONTINUE IF MORE TO DO.
FC9C	60		RTS			; ALL DONE.
FC9D						
FC9D	24 68	DISPLAY	BIT	MODES		; TEST FOR 40 OR 80
FC9F	700C		BVS	DSPL80		; STORE THE SINGLE CHARACTERS AND RETURN
FCA1	46 5C		LSR	CH		; INSURE PROPER 40 COLUMN DISPLAY
FCA3	06 5C		ASL	CH		; BY DROPPING BIT 0
FCA5	20 ADFC		JSR	DSPL80		; DISPLAY IN \$400 PAGE.
FCA8	A5 67		LDA	BKGND		; ALSO SET BACKGROUND COLOR
FCAA	91 60	DSPBKGND	STA	(BAS8L), Y		
FCAC	60		RTS			
FCAD						
FCAD	48	DSPL80	PHA			; PRESERVE CHARACTER
FCAE	A5 5C		LDA			; DETERMINE WHICH PAGE
FCB0	4A		LSR	A		
FCB1	A8		TAY			
FCB2	68		PLA			
FCB3	B0F5		BCS	DSPBKGND		; BRANCH IF \$900 PAGE
FCB5	91 5E		STA	(BAS4L), Y		
FCB7	60		RTS			
FCB8						
FCB8	B1 7E	NOTCR	LDA	(INBUF), Y		; ECHO CHARACTER
FCBA	20 39FC		JSR	COUT		
FCBD	C9 88		CMP	#88		; BACKSPACE
FCBF	F01D		BEQ	BKSPCE		
FCC1	C9 98		CMP	#98		; CANCEL?
FCC3	F008		BEQ	CANCEL		
FCC5	E6 80		INC	TEMP		
FCC7	A5 80		LDA	TEMP		
FCC9	C9 50		CMP	#INBUFLEN		
FCCB	D017		BNE	NXTCHAR		; NO WRAP AROUND ALLOWED.
FCCD	A9 DC	CANCEL	LDA	#0DC		; OUTPUT BACKSLASH
FCCF	20 39FC		JSR	COUT		
FCD2	20 EFFC		JSR	CROUT		
FCD5	FCD5	GETLNZ	.EQU	*		
FCD5	A5 6B	GETLN	LDA	PROMPT		
FCD7	20 39FC		JSR	COUT		
FCDA	A0 01		LDY	#01		
FCDC	84 80		STY	TEMP		; START AT BEGINNING OF INBUF
FCDE	A4 80	BKSPCE	LDY	TEMP		
FCE0	F0F3		BEQ	GETLN		
FCE2	C6 80		DEC	TEMP		; BACK UP INPUT BUFFER
FCE4	20 60FD	NXTCHAR	JSR	RDCHAR		; GET INPUT
FCE7	A4 80		LDY	TEMP		
FCE9	91 7E		STA	(INBUF), Y		
FCEB	C9 8D		CMP	#8D		
FCEd	D0C9		BNE	NOTCR		
FCEf	FCEf	CROUT	.EQU	*		
FCEf	2C 00C0		BIT	KBD		; TEST FOR START/STOP
FCF2	1013		BPL	NOSTOP		
FCF4	20 2EFD		JSR	KEYIN3		; READ KBD
FCF7	C9 A0		CMP	#0A0		; IS IT A SPACE?
FCF9	F007		BEQ	STOPLST		; YES, PAUSE TIL NEXT KEYPRESS.
FCFB	C9 89		CMP	#89		; QUIT THIS OPERATION
FCFD	D008		BNE	NOSTOP		; NO, IGNORE THIS KEY.
FCFF	4C 9FFA		JMP	ERROR2		; YES, RESTART
FD02	AD 00C0	STOPLST	LDA	KBD		
FD05	10FB		BPL	STOPLST		
FD07	A9 8D	NOSTOP	LDA	#8D		
FD09	4C 39FC		JMP	COUT		
FD0C						
FD0C	6C 7000	RDKEY	JMP	@KSWL		
FD0F						

10/31/89 10:04

HD:Apple ///:ROM - Monitor

Page 10

```

FD0F| A9 7F          KEYIN   LDA    #7F          ; MAKE SURE FIRST IS CURSOR
FD11| 85 63          STA    TBAS4H
FD13| 20 88FD        JSR    PICK        ; GO READ SCREEN
FD16| 48             KEYIN1  PHA    ; SAVE CHR AT CURSOR POSITION
FD17| 20 35FD        JSR    KEYWAIT     ; TEST FOR KEYPRESS
FD1A| B008          BCS    KEYIN2     ; GO GET IT
FD1C| A5 69          LDA    CURSOR     ; GIVE THEM AN UNDERScore FOR A TIME
FD1E| 20 9DFC        JSR    DISPLAY
FD21| 20 35FD        JSR    KEYWAIT     ; GO SEE IF KEYPRESSED
FD24| 68             KEYIN2  PLA    ;
FD25| 08            PHP    ; SAVE KEYPRESS STATUS
FD26| 48            PHA
FD27| 20 9DFC        JSR    DISPLAY
FD2A| 68            PLA
FD2B| 28            PLP
FD2C| 90E8          BCC    KEYIN1
FD2E| AD 00C0       KEYIN3  LDA    KBD
FD31| 2C 10C0       KEYIN4  BIT    KBDSTRB   ; CLEAR KEYBOARD STROBE
FD34| 60            RTS
FD35| E6 58          KEYWAIT INC    TBAS4L   ; JUST KEEP COUNTING
FD37| D009          BNE    KWAIT2
FD39| E6 63          INC    TBAS4H
FD3B| A9 7F          LDA    #7F        ; TEST FOR DONE
FD3D| 18            CLC
FD3E| 25 63          AND    TBAS4H
FD40| F005          BEQ    KEYRET     ; RETURN IF TIMED OUT
FD42| 0E 00C0       KWAIT2  ASL    KBD
FD45| 90EE          BCC    KEYWAIT
FD47| 60            KEYRET  RTS
FD48| ;
FD48| ;
FD48| FD48          ESC3    .EQU    *
FD48| 20 77FD        JSR    GOESC
FD4B| A5 68          ESCAPE  LDA    MODES     ; SET TO + SIGN FOR CURSOR MOVES
FD4D| 29 80          AND    #80
FD4F| 49 AB          EOR    #0AB
FD51| 85 69          STA    CURSOR
FD53| 20 0CFD        ESC1    JSR    RDKEY     ; READ NEXT CHARACTER
FD56| A0 08          LDY    #08        ; TEST FOR ESCAPE COMMAND
FD58| D9 F0FF        ESC2    CMP    ESCTABL,Y
FD5B| F0EB          BEQ    ESC3
FD5D| 88            DEY
FD5E| 10F8          BPL    ESC2        ; LOOP TIL FOUND OR DONE
FD60| ;
FD60| A9 80          RDCHAR  LDA    #80        ; GO READ A CHARACTER
FD62| 25 68          AND    MODES
FD64| 85 69          STA    CURSOR     ; SAVE STANDARD CURSOR
FD66| 20 0CFD        JSR    RDKEY
FD69| C9 9B          CMP    #9B        ; ESCAPE CHARACTER?
FD6B| F0DE          BEQ    ESCAPE
FD6D| C9 95          CMP    #95        ; FORWARD COPY?
FD6F| D0D6          BNE    KEYRET
FD71| 20 88FD        JSR    PICK        ; GET CHARACTER FROM SCREEN
FD74| 09 80          ORA    #80        ; SET TO NORMAL ASCII
FD76| 60            RTS
FD77| ;
FD77| A9 FB          GOESC   LDA    #0FB
FD79| 48            PHA
FD7A| B9 7FFD        LDA    ESCVECT,Y
FD7D| 48            PHA
FD7E| 60            RTS
FD7F| A1            ESCVECT .BYTE    0A1   ; CLEOL-1
FD80| 84            .BYTE    84       ; CLEOP-1
FD81| 7C            .BYTE    7C       ; CLSCRN-1
FD82| 62            .BYTE    62       ; COL40-1
FD83| 5C            .BYTE    5C       ; COL80-1
FD84| EC            .BYTE    0EC      ; CURLEFT-1
FD85| CA            .BYTE    0CA      ; CURRIGHT-1
FD86| DC            .BYTE    0DC      ; CURDOWN-1
FD87| B7            .BYTE    0B7      ; CURUP-1
FD88| ;
FD88| A5 5C          PICK    LDA    CH        ; GET A CHARACTER AT CURRENT CURSOR POSITION
FD8A| 4A            LSR    A           ; DETERMINE WHICH PAGE.
FD8B| A8            TAY
FD8C| 24 68          BIT    MODES     ; AND IF 80 COLUMN MODE
FD8E| 5005          BVC    PICK40    ; FORGET CARRY IF 40 COLUMNS
FD90| 9003          BCC    PICK40    ; GET CHARACTER FROM $400
FD92| B1 60          LDA    (BAS8L),Y
FD94| 60            RTS
FD95| B1 5E          PICK40  LDA    (BAS4L),Y
FD97| 60            RTS
FD98| ;
FD98| FD98          CLDSTRT .EQU    *
FD98| A9 03          LDA    #03
FD9A| 8D D0FF        STA    0FFD0     ; ZERO PAGE IS ON 3!
FD9D| FD9D          SETUP  .EQU    *
FD9D| D8            CLD    ; OF COURSE!
FD9E| A2 03          LDX    #03
FDA0| 86 7F          STX    INBUF+1
FDA2| BD BCFB        SETUP1  LDA    NMIRQ,X
    
```

10/31/89 10:04

HD:Apple ///:ROM - Monitor

Page 11

```

FDA5| 9D CAFF          STA    0FFCA,X
FDA8| BD B4FF          LDA    HOOKS,X
FDAB| 95 6E            STA    CSWL,X
FDAD| BD B8FF          LDA    VBOUNDS,X
FDB0| 95 58            STA    LMARGIN,X
FDB2| CA                DEX
FDB3| 10ED            BPL    SETUP1
FDB5| 85 82            STA    IBDRVN
FDB7| A9 A0            LDA    #0A0      ; INPUT BUFFER AT $3A0
FDB9| 85 7E            STA    INBUF
FDBB| A9 60            LDA    #60
FDBD| 85 81            STA    IBSLOT
FDBF| A9 FF            LDA    #0FF
FDC1| 85 68            STA    MODES
FDC3| 20 63FB        JSR    COL40     ; SET 40 COLUMNS, CLEAR SCREEN
FDC6|                  ;
FDC6| 00A0            ADR    .EQU    0A0
FDC6| 00A0            CPORTL .EQU    ADR
FDC6| 00A1            CPORTH .EQU    ADR+1
FDC6| 00A2            CTEMP  .EQU    ADR+2
FDC6| 00A3            CTEMP1 .EQU    ADR+3
FDC6| 00A4            YTEMP  .EQU    ADR+4
FDC6| 00C0            ROWTEMP .EQU    ADR+20
FDC6| C0DB            CWRTON  .EQU    0C0DB
FDC6| C0DA            CWRTOFF .EQU    0C0DA
FDC6| FFEC            CB2CTRL .EQU    0FFEC
FDC6| FFED            CB2INT  .EQU    0FFED
FDC6|                  ;
FDC6|                  ;
FDC6| A9 78            GENENTR LDA    #78      ; INIT SCREEN INDX LOCATIONS
FDC8| 85 A0            STA    CPORTL
FDCA| A9 08            LDA    #08
FDCC| 85 A1            STA    CPORTH
FDCE| A9 F0            LDA    #0F0     ; SET UP INDEX TO CHRSET
FDD0| 85 A4            STA    YTEMP
FDD2| A9 00            LDA    #00
FDD4| AA              TAX
FDD5| 95 C0            ZIPTEMPS STA  ROWTEMP,X
FDD7| E8              INX
FDD8| E0 20            CPX    #20
FDDA| D0F9            BNE    ZIPTEMPS
FDDC| A9 05            LDA    #05     ; FAKE THE FIRST BIT PATTERN
FDEE| 18              CLC          ; (PHANTOM 9TH BIT SHIFTED AS BIT 0)
FDDF| 08              PHP
FDE0| 48              PHA
FDE1| 86 A2            GENASC  STX    CTEMP   ; GENERATE THE ASCII
FDE3| A0 07            GASC11  LDY    #07     ; CODES FOR THE FIRST PASS
FDE5| A6 A2            GASC12  LDX    CTEMP
FDE7| 8A              GASC13  TXA
FDE8| 91 A0            STA    (CPORTL),Y ; $XXF=CHR 0 / 4
FDEA| E8              INX       ; $XXE=CHR 1 / 5
FDEB| 88              DEY       ; $XXD=CHR 2 / 6
FDEC| 3006           BMI      GASC14   ; $XXC=CHR 3 / 7
FDEE| C0 03           CPY      #03     ; $XXB=CHR 0 / 4
FDF0| D0F5           BNE      GASC13   ; $XXA=CHR 1 / 5
FDF2| F0F1           BEQ      GASC12   ; $XX9=CHR 2 / 6
FDF4| 20 99FE        GASC14  JSR    NXTPORT  ; $XX8=CHR 3 / 7
FDF7| B008           BCS     CBYTES   ; GO DECODE CHARACTER TABLE
FDF9| C9 0A           CMP     #0A     ; SECOND SET OF 4?
FDFB| D0E6           BNE     GASC11
FDFD| A2 24           LDX    #24
FDFE| D0E0           BNE     GENASC   ; BRANCH ALWAYS
FE01| 68              CBYTES  PLA       ; RESTORE BIT PATTERN
FE02| 28              PLP
FE03| A2 17           LDX    #17     ; (4 CHARACTERS OF 6 ROWS)
FE05| A0 05           LDY    #05     ; (FIVE COLUMNS)
FE07| 36 C4           CSHFT  ROL     ROWTEMP+4,X ; BREAK BYTE INTO
FE09| 0A              ASL     A       ; 5 BIT GROUPS
FE0A| D00E           BNE     SHFTCNT  ; BRANCH IF MORE BITS IN THIS BYTE
FE0C| 84 A2           STY    CTEMP
FE0E| C6 A4           DEC     YTEMP   ; (NOTE. CARRY IS SET)
FE10| F016           BEQ     DONE    ; BRANCH IF ALL DONE
FE12| A4 A4           LDY    YTEMP   ; GET CHARACTER TABLE INDEX
FE14| B9 C4FE        LDA    CHRSET-1,Y
FE17| 2A              ROL     A       ; (CARRY KEEPS BYTE NON-ZERO UNTIL ALL 8 ARE
FE18|                  ; ARE SHIFTED)
FE18| A4 A2           LDY    CTEMP   ; RESTORE COLUMN COUNT
FE1A| 88              SHFTCNT DEY     ; GOT ALL FIVE BITS?
FE1B| D0EA           BNE     CSHFT   ; NO, DO NEXT
FE1D| CA              DEX
FE1E| 10E5           BPL    CCOLMS  ; ALL ROWS DONE
FE20| 08              PHP       ; NO, DO NEXT
FE21| 48              PHA     ; SAVE REMAINING BIT PATTERN AND CARRY
FE22| 20 28FE        JSR    STORCHRS ; MOVE EM TO NON DISPLAYED VIDEO AREA
FE25| 4C 01FE        JMP    CBYTES
FE28|                  ;
FE28| FE28            DONE  .EQU    *
FE28|                  ;
FE28| A2 1F            STORCHRS LDX  #1F   ; MOVE CHARACTER PATTERNS TO VIDEO AREA
FE2A| A0 00            STORSET LDY  #00
    
```

FE2C	B5 C0	STOROW	LDA	ROWTEMP, X	
FE2E	0A		ASL	A	; SHIFT TO CENTER
FE2F	29 3E		AND	#3E	; STRIP EXTRA GARBAGE
FE31	91 A0		STA	(CPORTL), Y	
FE33	CA		DEX		
FE34	C8		INY		
FE35	C0 08		CPY	#08	; THIS GROUP DONE
FE37	D0F3		BNE	STOROW	; NO, NEXT ROW
FE39	20 99FE		JSR	NXTPORT	
FE3C	C9 08		CMP	#08	
FE3E	F004		BEQ	GENDONE	; ALL ROWS STORED?
FE40	8A		TXA		
FE41	10E7		BPL	STORSET	
FE43	60		RTS		; PARTIAL SET (\$478-\$5FF)
FE44					
FE44	A9 01	GENDONE	LDA	#01	; SET NORMAL MODE
FE46	85 A2		STA	CTEMP	
FE48	A9 60	GEN1	LDA	#60	; PREPARE TO SEND BYTES TO CHARACTER
FE4A	2C DBC0		BIT	CWRTON	; GENERATOR RAM
FE4D	20 AEF6		JSR	VRETRCE	; WAIT FOR NEXT VERTICAL RETRACE
FE50	A9 20		LDA	#20	; WAIT AGAIN
FE52	20 AEF6		JSR	VRETRCE	
FE55	2C DAC0		BIT	CWRTOFF	; CHARACTERS ARE NOW LOADED
FE58	20 88FE		JSR	ALTCHR	; REPEAT THIS SET FOR OTHER 64 CHARACTERS
FE5B	C6 A2		DEC	CTEMP	; HAVE WE DONE ALTERNATES YET?
FE5D	1016		BPL	GEN2	; NO, DO IT!
FE5F	A9 08		LDA	#08	; BUMP ASCII VALUES FOR NEXT SET
FE61	85 A1		STA	CPORTH	
FE63	A0 07	NXTASCI	LDY	#07	; THE USUAL COUNTDOWN
FE65	B1 A0	NXTASC2	LDA	(CPORTL), Y	
FE67	18		CLC		
FE68	69 08		ADC	#08	
FE6A	91 A0		STA	(CPORTL), Y	
FE6C	88		DEY		
FE6D	10F6		BPL	NXTASC2	
FE6F	20 99FE		JSR	NXTPORT	
FE72	90EF		BCC	NXTASCI	
FE74	60		RTS		
FE75	A0 03	GEN2	LDY	#03	; SETUP ALTERNATE WITH UNDERLINES
FE77	A9 7F		LDA	#7F	
FE79	99 FC05	UNDER	STA	05FC, Y	
FE7C	99 FC07		STA	07FC, Y	
FE7F	88		DEY		
FE80	10F7		BPL	UNDER	
FE82	A9 08		LDA	#08	
FE84	85 A1		STA	CPORTH	
FE86	D0C0		BNE	GEN1	
FE88					
FE88	A0 07	ALTCHR	LDY	#07	; ADJUST ASCII FOR ALTERNATE SET
FE8A	B1 A0	ALTC1	LDA	(CPORTL), Y	
FE8C	49 20		EOR	#20	; \$20--> \$40-->\$60
FE8E	91 A0		STA	(CPORTL), Y	
FE90	88		DEY		
FE91	10F7		BPL	ALTC1	; ADJUST THEM ALL
FE93	20 99FE		JSR	NXTPORT	
FE96	90F0		BCC	ALTCHR	
FE98	60		RTS		
FE99					
FE99	A5 A0	NXTPORT	LDA	CPORTL	; CONVERT \$78->\$F8 OR \$F8-\$78
FE9B	49 80		EOR	#80	
FE9D	85 A0		STA	CPORTL	
FE9F	3002		BMI	NOHIGH	
FEA1	E6 A1		INC	CPORTH	
FEA3	A5 A1	NOHIGH	LDA	CPORTH	
FEA5	C9 0C		CMP	#0C	
FEA7	D004		BNE	PORTDN	
FEA9	A9 04		LDA	#04	
FEAB	85 A1		STA	CPORTH	
FEAD	60	PORTDN	RTS		
FEAE					
FEAE	85 A3	VRETRCE	STA	CTEMP1	; SAVE BITS TO BE STORED
FEB0	AD ECFE		LDA	CB2CTRL	; CONTROL PORT FOR 'CB2'
FEB3	29 3F		AND	#3F	; RESET HI BITS TO 0
FEB5	05 A3		ORA	CTEMP1	
FEB7	8D ECFE		STA	CB2CTRL	
FEBA	A9 08		LDA	#08	; TEST VERTICAL RETRACE
FEBC	8D EDFE		STA	CB2INT	
FEBF	2C EDFE	VWAIT	BIT	CB2INT	; WAIT FOR RETRACE
FEC2	F0FB		BEQ	VWAIT	
FEC4	60		RTS		
FEC5					
FEC5	FEC5	CHRSET	.EQU	*	
FEC5					
FEC5	F0 01 82 18 40 84 81		.BYTE	0F0, 01, 82, 18, 40, 84, 81, 2F, 58, 44, 81, 29, 02, 1E, 01, 91, 7C, 1F, 49, 30	
FEC6	2F 58 44 81 29 02 1E				
FED3	01 91 7C 1F 49 30				
FED9	8A 08 43 14 31 2A 22		.BYTE	8A, 08, 43, 14, 31, 2A, 22, 13, 0E3, 0F7, 0C4, 91, 48, 0A2, 0DA, 24, 0C6, 4A	
FEE0	13 E3 F7 C4 91 48 A2				
FEE7	DA 24 C6 4A				
FEEB	62 8C 24 C6 F8 63 8C		.BYTE	62, 8C, 24, 0C6, 0F8, 63, 8C, 0C1, 46, 17, 52, 8A, 0AF, 16, 14, 0E3, 33, 31	

```

FEF2| C1 46 17 52 8A AF 16
FEF9| 14 E3 33 31
FEFD| C6 F8 DC 73 3F 46 17
FF04| 62 8C 21 E6 18 6A 8D      .BYTE   0C6,0F8,0DC,73,3F,46,17,62,8C,21,0E6,18,6A,8D,61,0CF,18,62
FF0B| 61 CF 18 62
FF0F| 74 D1 B9 18 49 4C 91      .BYTE   74,0D1,0B9,18,49,4C,91,0C0,0F3,09,2C,91,0C0,14,1D,8C,0EF,07
FF16| C0 F3 09 2C 91 C0 14
FF1D| 1D 8C EF 07
FF21| 17 43 88 31 84 1E DF      .BYTE   17,43,88,31,84,1E,0DF,0B,31,84,0F8,0FE,77,3E,3E,17,62,8C,0FD
FF28| 0B 31 84 F8 FE 77 3E
FF2F| 3E 17 62 8C FD
FF34| C7 50 E3 0B 51 C5 E8      .BYTE   0C7,50,0E3,0B,51,0C5,0E8,0C8,73,18,0C,42,3E,01,02,20,42,3E
FF3B| C8 73 18 0C 42 3E 01
FF42| 02 20 42 3E
FF46| 41 18 8C 08 00 70 EE      .BYTE   41,18,8C,08,00,70,0EE,00,11,11,21,11,02,0E0,3C,21,31,02,0E0
FF4D| 00 11 11 21 11 02 E0
FF54| 3C 21 31 02 E0
FF59| 1C 00 C8 B9 80 62 14      .BYTE   1C,00,0C8,0B9,80,62,14,1F,46,0A2,0DE,43,2C,04,88,0BE,0FF,0CE
FF60| 1F 46 A2 DE 43 2C 04
FF67| 88 BE FF CE
FF6B| 7D 37 49 88 95 18 98      .BYTE   7D,37,49,88,95,18,98,09,62,0D1,44,0E8,88,0FB,02,90,40,00,10
FF72| 09 62 D1 44 E8 88 FB
FF79| 02 90 40 00 10
FF7E| E0 03 02 00 40 00 00      .BYTE   0E0,03,02,00,40,00,00,08,00,00,28,10,42,44,25,82,0B8,2F,48
FF85| 08 00 00 28 10 42 44
FF8C| 25 82 B8 2F 48
FF91| 25 44 10 82 02 00 2F      .BYTE   25,44,10,82,02,00,2F,5A,40,45,02,8E,64,50,90,01,3E,26,42,80
FF98| 5A 40 45 02 8E 64 50
FF9F| 90 01 3E 26 42 80
FFA5| 21 80 00 05 00 F8 80      .BYTE   21,80,00,05,00,0F8,80,00,05,08,0F8,80,28,05,88
FFAC| 00 05 08 F8 80 28 05
FFB3| 88
FFB4|                                ;
FFB4| FFB4                                HOOKS      .EQU    *
FFB4| 06FC                                .WORD   COUT2
FFB6| 0FFD                                .WORD   KEYIN
FFB8| FFB8                                VBOUNDS   .EQU    *
FFB8| 00 50 00 18                        .BYTE   00,50,00,18
FFBC|                                ;
FFBC| 4C 86F6                            NMIRQ     JMP     RECON      ; IN DIAGNOSTICS
FFBF| 40                                    RTI
FFC0|                                ;
FFC0| 43 4F 50 59 52 49 47      .ASCII   "COPYRIGHT JANUARY, 1980 APPLE COMPUTER INC..JRH"
FFC7| 48 54 20 4A 41 4E 55
FFCE| 41 52 59 2C 20 31 39
FFD5| 38 30 20 20 41 50 50
FFDC| 4C 45 20 43 4F 4D 50
FFE3| 55 54 45 52 20 49 4E
FFEA| 43 2E 2E 4A 52 48
FFF0|                                ;
FFF0| CC D0 D3 B4 B8 88 95      ESCTABL   .BYTE   0CC,0D0,0D3,0B4,0B8,88,95,8A,8B,00
FFF7| 8A 8B 00
FFFA|                                ;
FFFA| CAFF                                NMI       .WORD   0FFCA
FFFC| EEF4                                RESET     .WORD   DIAGN      ; NOTHING
FFFE| CDFF                                IRQ       .WORD   0FFCD
0000|
0000|                                .END
    
```

↑
J.R. Huston
(also worked
on SOS)

J=James
R=Richard
aka
Dick
Huston

SYMBOL TABLE DUMP

AB - Absolute	LB - Label	UD - Undefined	MC - Macro
RF - Ref	DF - Def	PR - Proc	FC - Func
PB - Public	PV - Private	CS - Consts	
A1H	AB 0075	ALL	AB 0074
A2L	AB 0076	A3H	AB 0079
ADR	AB 00A0	ALTC1	LB FE8A
ASC3	LB FB5A	ASCDONE	LB FA08
ASCII12	LB F9E3	ASCII3	LB F9F4
BAS8L	AB 0060	BASCALC	LB FBC7
BITON	LB FA25	BKGNL	AB 0067
BSCLC2	LB FC2D	CANCEL	LB FCCD
CBYTES	LB FE01	CCOLMS	LB FE05
CLDSTRT	LB FD98	CLEOL	LB FBA2
CLEOP1	LB FB8E	CLSCRN	LB FB7D
COL40	LB FB63	COL80	LB FB5D
COUT2	LB FC06	CPORTR	AB 00A1
CROUT	LB FCEF	CSHFT	LB FE07
CTEMP1	AB 00A3	CTRLRET	LB FC38
CURLEFT	LB FBED	CURSOR	AB 0069
CWRTOFF	AB C0DA	CWRTRN	AB C0DB
DIGRET	LB F96B	DISPLAY	LB FC9D
DSPL80	LB FCAD	DUMMY	LB FACB
DUMP2	LB FB20	DUMP3	LB FB30
ERROR	LB FAA2	ERROR1	LB FB0B
ESC3	LB FD48	ESCAPE	LB FD4B
GASCI1	LB FDE3	GASCI2	LB FDE5
		GASCI3	LB FDE7
		GASCI4	LB FDF4
		A1PC	LB F9D6
		A3L	AB 0078
		ALTCHR	LB FE88
		ASC1	LB FB40
		ASCII0	LB FA1D
		BAS4H	AB 005F
		BASCALC1	LB FC19
		BKSPCE	LB FCDE
		CARRAGE	LB FBAF
		CH	AB 005C
		CLEOL1	LB FC89
		CMDSRCH	LB F91C
		CONTROL	LB FBA7
		CPORTR	AB 00A0
		CSWH	AB 006F
		CURDN1	LB FBC7
		CURUP	LB FBB8
		DEST	LB FAA5
		DISPLAYX	LB FC10
		DUMP	LB FB0D
		DUMP8	LB FAFD
		ERROR2	LB FA9F
		ESCTABL	LB FFF0
		GASCI1	LB FDE3
		A2H	AB 0077
		A4H	AB 007B
		ASC2	LB FB4C
		ASCII1	LB F9E1
		BAS8H	AB 0061
		BITOFF	LB FA29
		BLOCKIO	AB F479
		CB2INT	AB FFE2
		CKMDE	LB FA1E
		CLEOP	LB FB85
		CMDVEC	LB F97D
		COUT1	LB FC47
		CRMON	LB FA3A
		CTEMP	AB 00A2
		CURIGHT	LB FBCB
		CV	AB 005D
		DIGIT	LB F941
		DSPBKEND	LB FCAA
		DUMP1	LB FB1D
		ENTRY	LB F901
		ESC2	LB FD58
		FORGND	AB 0066
		GEN1	LB FE48

GEN2	LB FE75	GENASC	LB FDE1	GENDONE	LB FE44	GENENTR	LB FDC6	GETLN	LB FCD5
GETLNZ	LB FCD5	GETNUM	LB F92C	GO	LB FA91	GOESC	LB FD77	HOOKS	LB FFB4
IBBUF	AB 0085	IBCMD	AB 0087	IBDRVN	AB 0082	IBSLOT	AB 0081	INBUF	AB 007E
INBUFLN	AB 0050	INCHORZ	LB FC13	IRQ	LB FFFE	JUMP	LB FA8F	KBD	AB C000
KBDSTRB	AB C010	KEYIN	LB FD0F	KEYIN1	LB FD16	KEYIN2	LB FD24	KEYIN3	LB FD2E
KEYIN4	LB FD31	KEYRET	LB FD47	KEYWAIT	LB FD35	KSWH	AB 0071	KSWL	AB 0070
KWAIT2	LB FD42	LASTLN	LB FC87	LEFT80	LB FBF3	LEFTUP	LB FBF0	LFA36	LB FA36
LMARGIN	AB 0058	LNFD	LB FC52	MASK	AB 0069	MISMATCH	LB FA66	MODES	AB 0068
MON	LB F904	MONITOR	PR ----	MONZ	LB F908	MOVE	LB FA40	MOVNXT	LB FA45
NMI	LB FFFA	NMIRQ	LB FFBC	NOHIGH	LB FEA3	NOSTOP	LB FD07	NOTCR	LB FCB8
NOVER	LB FAF3	NXTA1	LB F994	NXTA4	LB F98E	NXTASC2	LB FE65	NXTASCI	LB FE63
NXTBAS	LB F94F	NXTBIT	LB F947	NXTBS2	LB F959	NXTCHAR	LB FCE4	NXTCHR	LB F932
NXTINP	LB F915	NXTLIN	LB FC16	NXTPORT	LB FE99	OLDPC	LB F9E0	PCH	AB 0073
PCL	AB 0072	PICK	LB FD88	PICK40	LB FD95	PORTDN	LB FEAD	PRA1BYTE	LB FA82
PRBYCOL	LB F9C4	PRBYTE	LB F9AE	PRBYTSP	LB FA84	PRCOLON	LB F9C7	PRHEX	LB F9B7
PRHEX2	LB F9C1	PRHEXZ	LB F9B9	PRINTA1	LB FA75	PROMPT	AB 006B	PRSPC	LB FA87
RCHAR	LB FD60	RKEY	LB FD0C	READ	LB FAD4	RECON	AB F686	REPEAT	LB FA2D
REPEAT1	LB FA35	RESET	LB FFFC	RET1	LB F7FE	RET2	LB F900	RET3	LB F882
RETA1	LB F9AD	RIGHT1	LB FBD1	RMARGIN	AB 0059	ROWTEMP	AB 00C0	RWERROR	LB FA97
RWLOOP	LB FADB	SAVCMD	LB FAD9	SCAN	LB F912	SCRL1	LB FC61	SCRL2	LB FC63
SCRL3	LB FC7A	SCRNLOC	AB 0058	SCROLL	LB FC5B	SEARCH	LB FA09	SEP	LB FAAE
SET80	LB FB67	SET80A	LB FB6F	SET80B	LB FB7B	SETCHZ	LB FBD7	SETCV	LB FBC5
SETCVH	LB FBDB	SETMDZ	LB FAD1	SETMODE	LB FACC	SETUP	LB FD9D	SETUP1	LB FDA2
SHFTCNT	LB FE1A	SPCE	LB FAB8	SRCH1	LB FA15	STACK	AB 006A	STATE	AB 007C
STOPLST	LB FD02	STOR	LB FABF	STOR1	LB FAC3	STORCHRS	LB FE28	STOROW	LB FE2C
STORSET	LB FE2A	SVMASK	LB F9D3	TBAS4H	AB 0063	TBAS4L	AB 0058	TBAS8H	AB 0065
TBAS8L	AB 0064	TEMP	AB 0080	TEMPX	AB 006C	TEMPY	AB 006D	TOSUB	LB F95E
TST80WID	LB F9CB	TSTA1	LB F99D	TSTBACK	LB FBE9	TSTBELL	LB FC4A	TSTCR	LB FBAB
TSTDUMP	LB FB0A	UNDER	LB FE79	USER	LB FA8C	USERADR	AB 0358	VBOUNDS	LB FFB8
VRETRCE	LB FEAE	VERFY	LB FA4F	VERFY1	LB FA54	VERFY2	LB FA60	VWAIT	LB FEBF
WINBTM	AB 005B	WINTOP	AB 005A	WRTE	LB FAD7	YSAV	AB 007D	YTEMP	AB 00A4
ZIPTEMPS	LB FDD5	ZSTATE	LB F967						

Assembly complete: 1129 lines
 0 Errors flagged on this Assembly

6502 OPCODE STATIC FREQUENCIES

ADC	: 5	***
AND	: 14	*****
ASL	: 12	*****
BCC	: 21	*****
BCS	: 20	*****
BEQ	: 82	*****
BIT	: 12	*****
BMI	: 7	****
BNE	: 41	*****
BPL	: 18	*****
BVC	: 2	*
BVS	: 3	*
CLC	: 7	****
CLD	: 2	*
CMP	: 35	*****
CPX	: 1	m
CPY	: 2	*
DEC	: 7	****
DEX	: 7	****
DEY	: 9	****
EOR	: 6	***
INC	: 18	*****
INX	: 3	*
INY	: 3	*
JMP	: 18	*****
JSR	: 79	*****
LDA	: 117	M *****
LDX	: 12	*****
LDY	: 20	*****
LSR	: 11	*****
ORA	: 10	*****
PHA	: 16	*****
PHP	: 4	**
PLA	: 14	*****
PLP	: 3	*
ROL	: 4	**
RTI	: 1	m
RTS	: 34	*****
SBC	: 67	*****
SEC	: 5	***
SEI	: 1	m
STA	: 72	*****
STX	: 7	****
STY	: 5	***
TAX	: 2	*
TAY	: 5	***
TSX	: 1	m
TXA	: 2	*
TXS	: 1	m
TYA	: 3	*

10/31/89 10:04

HD:Apple ///:ROM - Monitor

Page 15

Minimum frequency = 1
Maximum frequency = 117

Average frequency = 17

Unused opcodes:

BRK CLI CLV NOP ROR SED

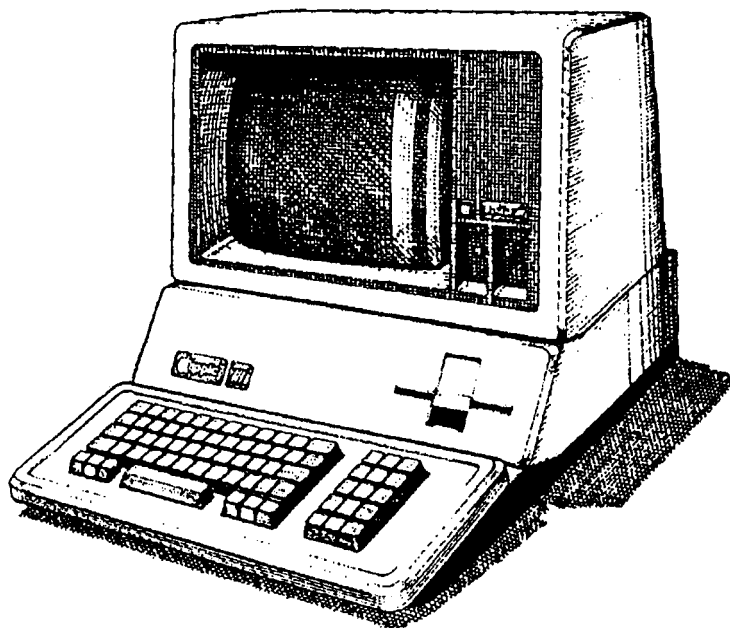
Program opcode usage: 89 %

(1.00) That's all, Folks ...

≡FINIS≡



Apple III Computer Information



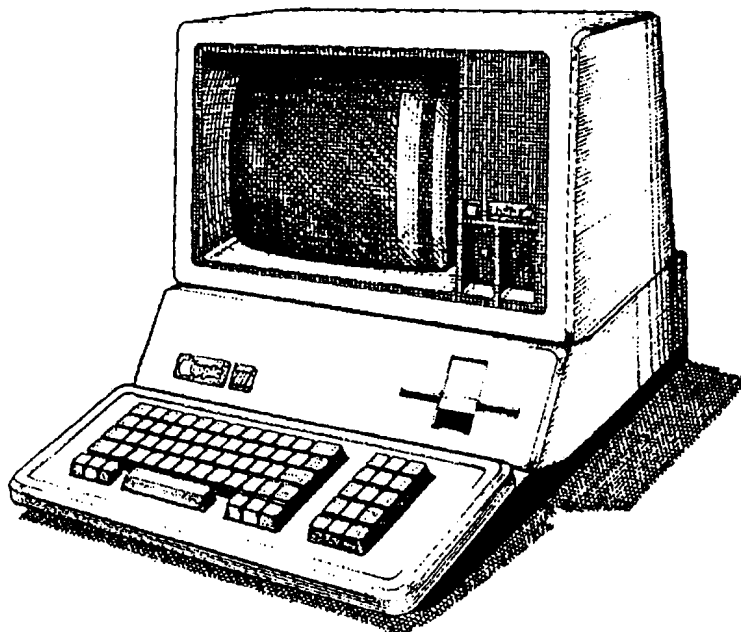
Inside the Apple III ROM

Document Table of Contents

- Revision 2 • 04 Dec 1997
- Revision 1 • 30 Nov 1997



Apple III Computer Information



Inside the Apple III ROM

Revision 2 • 04 Dec 1997

Inside the Apple /// Computer ROM

David T. Craig • 04 December 1997
71533.606@compuserve.com

TABLE OF CONTENTS

- 1 INTRODUCTION
- 2 ROM SECTIONS
- 3 IMPORTANT ROM ROUTINES
- 4 ROM TABLES
- 5 ROM USAGE BY SOS
- 6 MONITOR COMMANDS
- 7 A FEW COMMENTS
- 8 REFERENCES
- 9 DOCUMENT MODIFICATION HISTORY

1 INTRODUCTION

This document provides a general overview of the contents of the Apple /// computer ROM revision 1. This information should be used in conjunction with a copy of the ROM source code listing. The audience of this document is anyone with an interest in the technology of the Apple /// computer's hardware and software.

NOTE

There were two revisions of the Apple /// ROM, revision 0 and revision 1. Revision 0 ROMs had at address F1B9 the value 60. Revision 1 ROMs had at address F1B9 the value A0.

This ROM contains 4 KB of 6502 programming and several data tables. The ROM occupies memory addresses F000–FFFF. The basic purpose of the ROM is to test the Apple /// computer hardware and boot an operating system from the ///'s built-in floppy disk drive. The ROM also contains a simple Monitor program whose purpose is to allow the user to interact with the /// at the hexadecimal level.

Apple planned from an architectural perspective to support two 4K ROMs. But only one ROM was ever created. The Environment Register let you control which ROM was active. Both ROMs shared the same address space so you could only have one ROM active at a time. This feature would have doubled the ROM's effective size providing Apple with more room for ROM-based features that higher-level /// software (e.g. SOS) could have used.

When the Apple /// computer is turned on the ROM's flow of execution is as follows:

- 1) The ROM starts execution at the address contained in FFFC-FFFD (RESET) which is address F4EE (DIAGN).
- 2) Diagnostics (DIAGN/F4EE) starts. The diagnostic first initializes some memory for the ROM's use. If the Open Apple and the Control keys are held down then enter the ROM Monitor. Otherwise run several diagnostic checks of the /// hardware (tests zero page, sizes memory, initializes screen text buffer, tests stack memory, tests ROM checksum, tests VIA chip, tests ACIA chip, tests A/D circuitry, tests keyboard connection). Any diagnostic failures display an error message and the user has to reset the computer.
- 3) Read block 0 (512 bytes) to address A000 from the floppy disk in the built-in disk drive (BOOT/F6A1). If no disk is found or block 0 cannot be read then display "RETRY" and wait for the user to reset the computer. If the block is successfully read then execute the block contents (this is called the SOS Bootstrap Loader: see section ROM USAGE BY SOS).

2 ROM SECTIONS

Section	Address	Purpose
Disk I/O	F000-F4C4	Read and write floppy disk blocks (512 bytes each)
Diagnostics	F4C5-F7FE	Diagnose the /// hardware
Monitor	F7FF-FFFF	Interacts with user so user can do simple things

3 IMPORTANT ROM ROUTINES

BLOCKIO / F479	Reads or write a disk block (512 bytes), calls routine REGRWTS (F000) which reads a sector (256 bytes) from the disk.
BOOT / F6A1	Read floppy disk block *0 into address A000, execute the block.
ENTRY / F901	Monitor entry point.
DIAGN / F4EE	Diagnostic entry point.
USREENTRY/F6E6	Tests RAM and displays a table showing chip failures (users may execute this routine from the Monitor). This test is aimed at Apple ///s with 128K of RAM that exists on the older 12-Volt RAM boards. Though this routine will work with the newer 5-Volt RAM boards (256K) this test shows wrong information when RAM errors occur since the two RAM boards contain a different number of RAM chips. You can identify the different RAM boards as follows: The 5V boards have a large gray ceramic resistor near the edge and the 12V boards have a small blue tubular capacitor. To test the ///'s RAM you really should use Apple's /// Diagnostics Disk which lets you specify which RAM board you have.

4 ROM TABLES

Here's a list of the important data tables in the ROM. This list does not include disk I/O tables.

Table Name / Address	Contents
CHRSET / FEC5-FFB3	Default character set (overridden when SOS loads the character set from SOS.DRIVER)
Copyright / FFC0-FFEF	Copyright message (contains the initials "JRH" for J. R. "Dick" Huston who was a key player behind the /// and SOS)
NMI / FFFA-FFFF	Jump address for the Non-Maskable Interrupt signal
RESET / FFFC-FFFF	Jump address when the /// is powered on
IRQ / FFFE-FFFF	Jump address for the Interrupt Request signal

5 ROM USAGE BY SOS

The Apple /// operating system (SOS = Sophisticated Operating System or Sara's OS) uses several ROM routines. These routines seem to all be related to disk block I/O. The following discussion is based on SOS version 1.3.

When the ROM loads block 0 from a SOS disk the ROM is loading the SOS Bootstrap Loader program. This program, which is at most 512 bytes in length, uses the ROM routine REGRWTS (F000) to read the SOS Loader into memory. This program does not test the ROM revision. It is interesting to note that ROM routine BLOCKIO is not used, instead a lower-level routine (REGRWTS) is used.

The SOS Loader determines if the ROM is revision 1 by comparing address F1B9's contents against A0 (reference: SOS source file SOSLDR.D.SRC). If this comparison fails then SOS displays on the screen the error "ROM ERROR: PLEASE NOTIFY YOUR DEALER." If the ROM revision is correct then the SOS loader uses the ROM's disk I/O routines to read more of SOS into memory.

The disk /// driver that is built into SOS also uses the ROM to perform disk block I/O (reference: DISK3.SRC). It is interesting to note that when the disk driver is initialized the driver checks if the ROM revision is 0 or 1. A revision of 0 is detected if address F1B9 contains 60. If neither revision is found then the disk driver returns an error to SOS (I don't think this will ever happen since the SOS loader has already determined that the ROM is revision 1). For a valid ROM revision the disk driver sets up several jump vectors which point to the appropriate addresses in the ROM for the various ROM routines needed by the disk driver. Therefore, the disk driver seems compatible with either ROM revision whereas the SOS loader likes only revision 1.

The .CONSOLE driver source listing appears to not use any ROM routines even though the ROM contains 40 and 80 column text routines and keyboard input routines. I assume the console driver was much more sophisticated than the ROM's text features and so using the ROM routines would not have worked well for this driver. I also assume that if the console driver used the ROM that when ROM

revision 1 was built the console driver would have had to be changed and Apple (smartly) did not want to do this.

6 MONITOR COMMANDS

Holding down the Open Apple and Control keys when the /// starts or when you press the Reset key activates the /// ROM Monitor. The screen will display in the upper left corner a small right-facing arrow with a blinking underscore character as the cursor. The Monitor's commands are based on the Apple]['s Monitor commands but some commands have changed slightly and others are new for the (newer) ///.

The Monitor supports the following commands:

addr1.addr2	Dump memory data to screen from address 1 to address 2 and display ASCII character at the right of the screen.
CARRIAGE RETURN	Dump next line of addresses to the screen.
SPACE	Pause current memory dump. Press again to continue.
addr:byte_list	Store starting at the address the list of bytes.
addr:'text'	Store text starting at address with high bit clear.
addr:"text"	Store text starting at address with high bit set.
addr3<addr1.addr2M	Move data in addresses 1-2 to address 3.
addr3<addr1.addr2V	Verify data in addresses 1-2 is the same as data starting at address 3.
byte<addr1.addr2S	Search memory in address range 1-2 for the byte.
block<addr1.addr2W	Write address range to disk starting at the disk block.
block<addr1.addr2R	Read disk starting at block to the address range.
addrG	Call subroutine at the address.
addrJ	Jump to the address.
U	Call user routine starting at address \$03F8.
X	Repeat last command line until you press the SPACE BAR.
ESC-8	Display 80 columns of text.
ESC-4	Display 40 columns of text.
/	Seperate multiple commands on the same line.
CTRL-I	Interrupt current operation, return to Monitor command line.

Note: See Wells' *Apple /// Entry Points* article for a great overview of the ROM Monitor, its commands (with some syntax errors), and the memory locations that need setting up for the key ROM routines to work. Apple's */// Service Reference Manual* (p. 13.57) has a list of Monitor commands. Anderson's *The Apple Nobody Knows* also has good Monitor command info.

To obtain a binary dump of the /// ROM you can do the following:

1. Initialize a disk on either the /// or an Apple][computer.
2. Insert the new disk in the ///.
3. Start the /// and hold down the Open Apple and Control keys.
4. You should be in the /// Monitor.
5. Type 0<F000.FFFW to write the ROM to disk blocks 0 to 7
6. Use a disk block reader on the /// or the][to read the ROM blocks and save them to a real file.

This disk writing is needed since the ROM does not provide a command for redirecting screen output to the ///'s serial port. But, I've read that you can output the ROM contents to the ///'s serial port but this involves using the Monitor to write a small program. If anyone has such a program please send a copy my way.

7 A FEW COMMENTS

I find it interesting, at least from a software engineering perspective, to note that in my opinion the /// ROM is missing several key features which I thought any system ROM would need. The ROM is missing two features which I think would have been useful to Apple and outside /// programmers:

- 1) The ROM does not have an explicit version number which exists at a specific ROM address. This version number could be used to validate the ROM in case there were several different ROMs (as there were). Apple uses a pseudo ROM version number (called the revision number) during the loading of SOS but this is somewhat lame in my opinion.
- 2) The ROM does not have a dispatch routine for use by the OS or applications that want to use ROM routines. This dispatch routine would reside at a specific address (e.g., F000) and it would take as input a command number and a set of parameters. These parameters could be passed via registers or on the stack. This routine would allow Apple to change the ROM and ROM "users" would not need to change their programming as long as they used the selector routine. The Apple][ROM did not have such a routine which caused Apple many headaches when it wanted to change the Apple][ROM and had to keep lots of routines in their same place.
- 3) The ROM source code is rather sparse concerning comments. It would be nice if the ROM contained detailed information about what each routine did and how to call the routines. Obviously, Apple did not expect anyone but Apple's own programmers to ever see the ROM source or use the ROM routines. (I've seen the Lisa computer's ROM listing which is much better documented than the ///'s and both are comparable in terms of age).

8 REFERENCES

Apple /// ROM Listing - Revision 0

This can be found in the Apple /// patent (#4,383,296) dated May 1983. Note that in places this ROM listing is not always readable.

Apple /// ROM Listing - Revision 1

I have a very readable listing of the revision 1 ROM that was printed on a laser printer.

Apple /// Service Reference Manual (Level 2)

This almost 500 page document by Apple has everything you would want to know about the ///'s hardware, low-level software, and how to service a broken ///. Includes descriptions of the System Monitor (a.k.a. Development Monitor) [page 17.3] and the built-in RAM test routine [page 13.51].

Apple /// SOS Bootstrap Loader Listing

Shows how 512 bytes of code are used to load SOS from disk into the ///'s memory.

The following articles provide good ROM information:

Apple /// Entry Points, Andy Wells, Call-APPLE, October 1981

Apple /// Dabbling, Rick Smith, Apple Orchard, Summer 1981

/// Bits: John Jeppson's Guided Tour of Highway ///, John Jeppson, Softalk, May 1983

The Apple Nobody Knows, Alan Anderson, Apple Orchard, Fall 1981

Unlocking the Apple /// - Part 3, Alan Anderson, Apple Orchard, September 1982

Apple ///: 12-Volt 128K Internal Diagnostics, Apple Technical Information Library

9 DOCUMENT MODIFICATION HISTORY

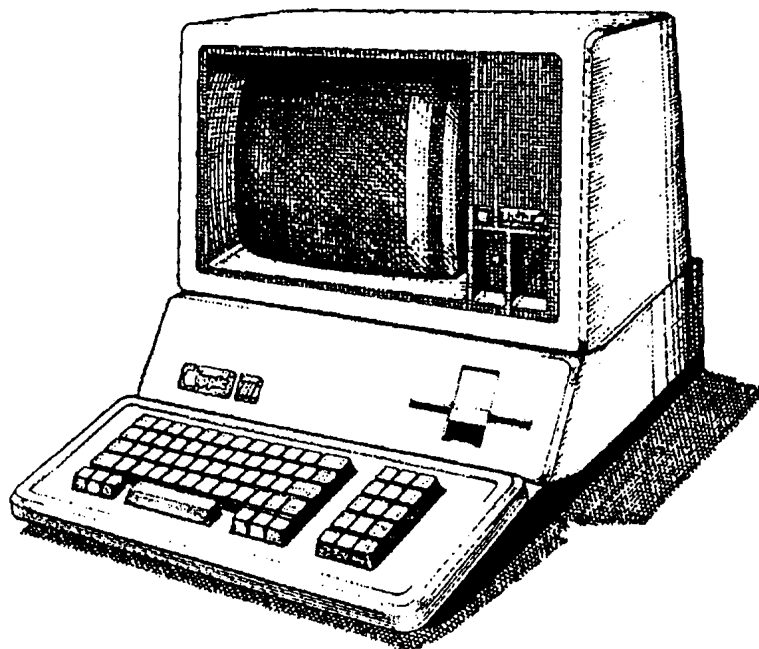
30 Nov 1997 Created this document.

04 Dec 1997 Corrected a few problems, extended the Reference section to include more /// articles pertaining to the /// ROM, added this section, added section MONITOR COMMANDS.

###



Apple III Computer Information



Inside the Apple III ROM

Revision 1 • 30 Nov 1997

Inside the Apple /// Computer ROM

David T. Craig • 30 November 1997
71533.606@compuserve.com

TABLE OF CONTENTS

- 1 INTRODUCTION
- 2 ROM SECTIONS
- 3 IMPORTANT ROM ROUTINES
- 4 ROM TABLES
- 5 ROM USAGE BY SOS
- 6 A FEW COMMENTS
- 7 REFERENCES

1 INTRODUCTION

This document provides a general overview of the contents of the Apple /// computer ROM revision 1. This information should be used in conjunction with a copy of the ROM source code listing. The audience of this document is anyone with an interest in the technology of the Apple /// computer's hardware and software.

NOTE

There were two revisions of the Apple /// ROM, revision 0 and revision 1. Revision 0 ROMs had at address F1B9 the value 60. Revision 1 ROMs had at address F1B9 the value A0.

This ROM contains 4 KB of 6502 programming and several data tables. The ROM occupies memory addresses F000–FFFF. The basic purpose of the ROM is to test the Apple /// computer hardware and boot an operating system from the ///'s built-in floppy disk drive. The ROM also contains a simple Monitor program whose purpose is to allow the user to interact with the /// at the hexadecimal level.

When the Apple /// computer is turned on the ROM's flow of execution is as follows:

- 1) The ROM starts execution at the address contained in FFFC–FFFD (RESET) which is address F4EE (DIAGN).
- 2) Diagnostics (DIAGN/F4EE) starts. The diagnostic first initializes some memory for the ROM's use. If the Open Apple key is held down then enter the ROM Monitor. Otherwise run several diagnostic checks of the /// hardware (tests zero page, sizes memory, initializes screen text buffer,

tests stack memory, tests ROM checksum, tests VIA chip, tests ACIA chip, tests A/D circuitry, tests keyboard connection). Any diagnostic failures display an error message and the user has to reset the computer.

- 3) Read block 0 (512 bytes) to address A000 from the floppy disk in the built-in disk drive (BOOT/F6A1). If no disk is found or block 0 cannot be read then display "RETRY" and wait for the user to reset the computer. If the block is successfully read then execute the block contents (this is called the SOS Bootstrap Loader: see section ROM USAGE BY SOS).

2 ROM SECTIONS

Section	Address	Purpose
Disk I/O	F000-F4C4	Read and write floppy disk blocks (512 bytes each)
Diagnostics	F4C5-F7FE	Diagnose the /// hardware
Monitor	F7FF-FFFF	Interacts with user so user can do simple things

3 IMPORTANT ROM ROUTINES

BLOCKIO / F479	Reads or write a disk block (512 bytes), calls routine REGRWTS (F000) which reads a sector (256 bytes) from the disk
BOOT / F6A1	Read floppy disk block #0 into address A000, execute the block
ENTRY / F901	Monitor entry point
DIAGN / F4EE	Diagnostic entry point
USRENTY/F6E6	Tests RAM and displays a table showing chip failures (users may execute this routine from the Monitor)

4 ROM TABLES

Here's a list of the important data tables in the ROM. This list does not include disk I/O tables.

Table Name / Address	Contents
CHRSET / FEC5-FFB3	Default character set (overridden when SOS loads the character set from SOS.DRIVER)
Copyright / FFC0-FFEF	Copyright message (contains the initials "JRH" for J. R. Huston who was a key player behind the /// and SOS)
NMI / FFFA-FFFB	Jump address for the Non-Maskable Interrupt signal
RESET / FFFC-FFFD	Jump address when the /// is powered on

IRQ / FFFE-FFFF Jump address for the Interrupt Request signal

5 ROM USAGE BY SOS

The Apple /// operating system (SOS) uses several ROM routines. These routines seem to all be related to disk block I/O. The following discussion is based on SOS version 1.3.

When the ROM loads block 0 from a SOS disk the ROM is loading the SOS Bootstrap Loader program. This program, which is at most 512 bytes in length, uses the ROM routine REGRWTS (F000) to read the SOS Loader into memory. This program does not test the ROM revision. It is interesting to note that ROM routine BLOCKIO is not used, instead a lower-level routine (REGRWTS) is used.

The SOS Loader determines if the ROM is revision 1 by comparing address F1B9's contents against A0 (reference: SOS source file SOSLDR.D.SRC). If this comparison fails then SOS displays on the screen the error "ROM ERROR: PLEASE NOTIFY YOUR DEALER." If the ROM revision is correct then the SOS loader uses the ROM's disk I/O routines to read more of SOS into memory.

The disk /// driver that is built into SOS also uses the ROM to perform disk block I/O (reference: DISK3.SRC). It is interesting to note that when the disk driver is initialized the driver checks if the ROM revision is 0 or 1. A revision of 0 is detected if address F1B9 contains 60. If neither revision is found then the disk driver returns an error to SOS (I don't think this will ever happen since the SOS loader has already determined that the ROM is revision 1). For a valid ROM revision the disk driver sets up several jump vectors which point to the appropriate addresses in the ROM for the various ROM routines needed by the disk driver. Therefore, the disk driver seems compatible with either ROM revision whereas the SOS loader likes only revision 1.

6 A FEW COMMENTS

I find it interesting, at least from a software engineering perspective, that the ROM is missing some key features which I thought any system ROM would need. The ROM is missing two features which I think would have been useful to Apple and outside /// programmers:

- 1) The ROM does not have an explicit version number which exists at a specific ROM address. This version number could be used to validate the ROM in case there were several different ROMs (as there were). Apple uses a pseudo ROM version number (called the revision number) during the loading of SOS but this is somewhat lame in my opinion.
- 2) The ROM does not have a selector routine for use by the OS or applications that want to use ROM routines. This selector would reside at a specific address (e.g., F000) and it would take as input a command number and a set of parameters. These parameters could be passed via registers or on the stack. This routine would allow Apple to change the ROM and ROM

“users” would not need to change their programming as long as they used the selector routine. The Apple][ROM did not have such a routine which caused Apple many headaches when it wanted to change the Apple][ROM and had to keep lots of routines in their same place.

7 REFERENCES

Apple /// ROM Listing

I have a very nice listing of revision 1 ROM. A listing (that is somewhat readable) for the earlier revision 0 ROM may be found in the Apple /// patent.

Apple /// Service Reference Manual (Level 2)

This almost 500 page book by Apple has everything you would want to know about the ///'s hardware, low-level software, and how to service a broken ///. Includes descriptions of the System Monitor (a.k.a. Development Monitor) [page 17.3] and the built-in RAM test routine [page 13.51].

Apple /// SOS Bootstrap Loader Listing

Shows how 512 bytes of code is used to load SOS from disk into the ///'s memory.



Apple /// Computer Technical Information

SOME COMMENTS ABOUT THE APPLE /// COMPUTER BOOT ROM

David T Craig -- 27 February 2004

BACKGROUND

The Apple /// computer was introduced by Apple Computer in 1980 and was discontinued in 1985.

This computer was a microcomputer with originally 128 KB of RAM memory expandable to 256 KB of RAM. It featured a 4 KB ROM (addressed from \$F000 to \$FFFF hexadecimal) which housed the initial programming that executed when the user turned on the computer. This ROM contained programming for the following functions:

- + diagnose hardware circuitry and memory
- + load and run a disk operating system (i.e. "boot")
- + provide an interface to a simple monitor program

The author wrote these comments after looking at the Apple /// ROM listing as found in Apple Computer's patent number 4,383,296 dated 10 May 1983. This analysis occurred during a scanning of the Apple /// patent.

ROM COMMENTS

The Apple /// patent's ROM program listing is terrible in terms of printed quality. Many parts are very faint and impossible to read. I assume this was done on purpose by Apple's legal department so that Apple's competitors would not be able to duplicate this ROM programming easily.

Some Comments about the Apple /// Computer Boot ROM
David T Craig -- 27 February 2004 -- 1 of 3

The ROM programming does not seem to have been built for expansion. By this I mean the programming seems to have been written to just make it work and no long term thought was given to the ROM programming's organization.

There were two versions of the ROM. The Apple /// operating system (OS) programming needed to differentiate between the ROM versions since the ROM contained several routines which the OS used. This version determination was not done in a logical way. A memory location was chosen at random (at least it seems this way to me) to serve as the ROM's "version number". The OS had to test this "version number" when it needed to use specific ROM services.

The ROM version also determined the location of several ROM routines which the Apple /// OS used.

The ROM's organization could have been improved greatly in my opinion if it was organized differently. At the beginning of the ROM address space (\$F000) include a short header containing the following:

- \$F000 - ROM version number
- \$F001 - ROM size (K bytes)
- \$F002 - ROM checksum (2 bytes)
- \$F003 - ROM routine dispatch jump vector (3 bytes)
- \$F006 - ROM copyright notice (e.g. "(c) Apple Computer 1980")

The remainder of the ROM would have contained whatever programming and table data was needed.

The routine dispatch jump vector would be a standard jump instruction to a routine in the ROM whose purpose would be to let outside programs such as the operating system, device drivers, or even application programs access ROM routines in a ROM version independent manner. The dispatch routine would take as input a command number (in say the CPU's A register) and return result information in the CPU's X and Y registers. The A register on return would contain an error result with 0 meaning no error. Or, some fixed memory area could be use to handle ROM routine parameters. This dispatch mechanism could be seen as a BIOS (basic input output system).

Possible dispatch routines could be:

- + Restart or Cold start or Warm start the computer
- + Read a block from a disk drive
- + Write a block to a disk drive
- + Return size in blocks of a disk drive
- + Checksum the ROM for diagnostic purposes
- + Test computer's RAM memory for diagnostic purposes
- + Enter the Apple /// Monitor program

This dispatch mechanism would have simplified the Apple /// OS use of the ROM services since the ROM would always be accessed from just one address (\$F003). If the OS requested a ROM service which was unavailable (e.g. an old ROM was installed) then the ROM would tell the OS that the service did not exist via a dispatch error result.

CONCLUSION

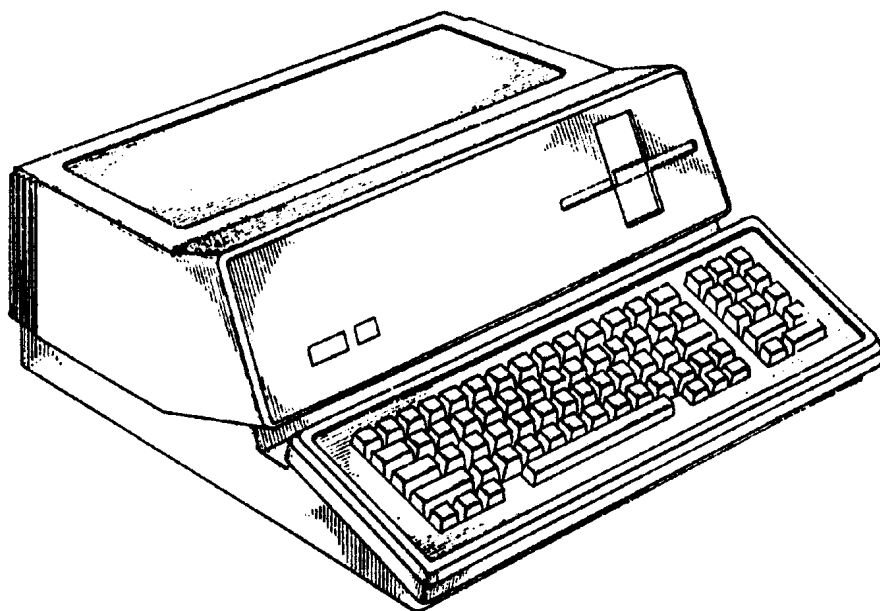
Hopefully this little commentary provides some useful information to its reader. If you are interested in the Apple /// computer you should see its patents (one is for the Apple ///, the other is for the Apple /// Plus). The first patent contains the full ROM listing, but the author has a real digital copy which is much more readable.

Enjoy.

###

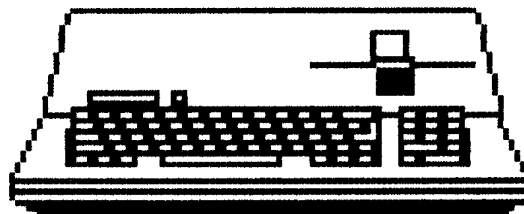


Apple /// Computer Information



APPLE /// SOS BOOTSTRAP LOADER SOURCE

ADDED BY DAVID T CRAIG • 2006



Apple ///
Apple ///+

Apple /// SOS Technical Information

SOS 1.3

Floppy Bootstrap Loader

Source Code Listing

This listing shows the code which is found at the beginning of a SOS boot disk. When the Apple /// computer starts the computer's ROM loads this code from the floppy disk and executes the code. This code loads the Apple ///'s operating system, SOS.

Source Code Listing
for

Apple ///

**SOS Floppy
Bootstrap Loader**

David T. Craig
736 Edgewater
Wichita, Kansas 67230

10/31/89 9:45

HD:Apple ///:SOS Floppy Bootstrap Loader

Page 1

```

0000| ;*****
0000| ; APPLE /// BOOTSTRAP LOADER FOR FLOPPY DISK
0000| ; - Disassembled 10-March-1988 by Scott Stinson
0000| ;*****
0000|
0000| .ABSOLUTE
0000| .PROC BOOTSTRAPLOADER
0000| .ORG 0A000
A000|
A000| ;-----
A000| ; EQUATES
A000| ;-----
A000|
A000| ; ZERO PAGE LOCATIONS
A000| ;-----
A000|
A000| 0082 IBDRVN .EQU 82 ; DRIVE NUMBER
A000| 0083 IBTRK .EQU 83 ; TRACK NUMBER
A000| 0084 IBSECT .EQU 84 ; SECTOR NUMBER
A000| 0085 IBBUFP .EQU 85 ; BUFFER POINTER
A000| 0087 IBCMD .EQU 87 ; COMMAND NUMBER
A000| 00E3 IBBUPTMP .EQU 0E3 ; BUFFER POINTER TEMPORARY
A000| 00E5 FILECNT .EQU 0E5 ; FILE COUNT
A000| 00E7 INDXBLCNT .EQU 0E7 ; INDEX BLOCK COUNT
A000| 00E8 SOSJMPADR .EQU 0E8 ; SOS JUMP ADDRESS
A000|
A000| ;-----
A000| ; HARDWARE I/O ADDRESSES
A000| ;-----
A000|
A000| 0628 SCREENLOC .EQU 0628 ; SCREEN LOCATION
A000| C010 KBDSTROBE .EQU 0C010 ; KEYBOARD STROBE
A000| C040 IOBEEP .EQU 0C040 ; I/O BEEP
A000|
A000| ;-----
A000| ; GENERAL EQUATES
A000| ;-----
A000|
A000| 0040 RETINT .EQU 40 ; RETURN FROM INTERRUPT
A000| 0C00 IDXBLK1 .EQU 0C00 ; INDEX BLOCK 1
A000| 0D00 IDXBLK2 .EQU 0D00 ; INDEX BLOCK 2
A000| 1E00 LOADADR .EQU 1E00 ; LOADING ADDRESS
A000| 1E08 OFFSET .EQU 1E08 ; OFFSET
A000| 2000 FIRSTPAGE .EQU 2000 ; FIRST PAGE
A000| A200 MAINBUFP .EQU 0A200 ; MAIN BUFFER
A000| F000 REGRWTS .EQU 0F000 ; READ/WRITE SECTOR ROUTINE
A000| F4A0 SECTABL .EQU 0F4A0 ; SECTOR TABLE
A000| FFCA NMIVECTOR .EQU 0FFCA ; NON-MASKABLE INTERRUPT VECTOR
A000| FDFD EREG .EQU 0FFFD ; ENVIRONMENT REGISTER
A000| FFEF BREG .EQU 0FFEF ; BANK REGISTER
A000|
A000| ;-----
A000| ; ENTRY POINT
A000| ;-----
A000|
A000| ENTRY SEI ; SET INTERRUPT DISABLE
A001| D8 CLD ; CLEAR DECIMAL FLAG
A002| A9 77 LDA #77 ; LOAD ACCUMULATOR WITH $77
A004| 8D DFFF STA EREG ; STORE IN ENVIRONMENT REGISTER
A007| ; SET 2 MHZ, I/O SPACE ENABLED, SCREEN ENABLED,
A007| ; RESET ENABLED, WRITE PROTECT NOT ENABLED,
A007| ; PRIMARY STACK, AND ROM SELECTED
A007| A2 FF LDX #0FF ; LOAD ACCUMULATOR WITH $FF
A009| 9A TXS ; TRANSFER X-REGISTER TO STACK POINTER
A00A| 2C 10C0 BIT KBDSTROBE ; CLEAR KEYBOARD
A00D| A9 40 LDA #RETINT ; LOAD ACCUMULATOR WITH RETURN FROM INTERRUPT
A00F| 8D CAFF STA NMIVECTOR ; STORE IN NON-MASKABLE INTERRUPT VECTOR
A012| A9 07 LDA #07 ; LOAD ACCUMULATOR WITH $07
A014| 8D EFFF STA BREG ; STORE IN BANK REGISTER
A017| A9 00 LDA #00 ; LOAD ACCUMULATOR WITH $00
A019| CE EFFF $010 DEC BREG ; DECREMENT BANK REGISTER
A01C| 8D 0020 STA FIRSTPAGE ; STORE IN FIRST PAGE OF BANK
A01F| AE 0020 LDX FIRSTPAGE ; LOAD X-REGISTER WITH FIRST PAGE BYTE
A022| D0F5 BNE $010 ; BRANCH IF BYTE IS NOT EQUAL TO $00
A024|
A024| ;-----
A024| ; This section reads in the SOS directory.
A024| ;-----
A024|
A024| A9 00 READSOSDIR LDA #00 ; LOAD ACCUMULATOR WITH $00-BLOCK HIGH BYTE
A026| 85 85 STA IBBUFP ; STORE IN BUFFER POINTER LOW BYTE
A028| A2 A2 LDX #0A2 ; LOAD X-REGISTER WITH $A2
A02A| 86 86 STX IBBUFP+1 ; STORE IN BUFFER POINTER HIGH BYTE
A02C| A2 02 LDX #02 ; LOAD X-REGISTER WITH $02-BLOCK LOW BYTE
A02E| A4 85 RDSOSDIRLP LDY IBBUFP ; LOAD Y-REGISTER WITH BUFFER POINTER LOW BYTE
A030| 84 E3 STY IBBUPTMP ; STORE IN BUFFER POINTER TEMPORARY LOW BYTE
A032| A4 86 LDY IBBUFP+1 ; LOAD Y-REGISTER WITH BUFFER POINTER HIGH BYTE
A034| 84 E4 STY IBBUPTMP+1 ; STORE IN BUFFER POINTER TEMPORARY HIGH BYTE
A036| 20 1DA1 JSR READBLK ; JUMP TO READ A BLOCK FROM FLOPPY DISK DRIVE
    
```

DAVID J. CRAIG



ROM



```

A039| A0 02          LDY    #02          ; LOAD Y-REGISTER WITH $02
A03B| B1 E3          LDA    @IBBUPTMP,Y ; LOAD ACCUMULATOR WITH NEXT BLOCK TO READ LOW
A03D|                ; BYTE
A03D| AA            TAX          ; TRANSFER ACCUMULATOR TO X-REGISTER
A03E| C8            INY          ; INCREMENT Y-REGISTER
A03F| B1 E3          LDA    @IBBUPTMP,Y ; LOAD ACCUMULATOR WITH NEXT BLOCK TO READ HIGH
A041|                ; BYTE
A041| D0EB          BNE    RDSOSDIRLP ; BRANCH IF NEXT BLOCK TO READ HIGH BYTE IS NOT
A043|                ; EQUAL TO ZERO
A043| E0 00          CPX    #00          ; CHECK TO SEE IF NEXT BLOCK TO READ LOW BYTE IS
A045|                ; ZERO
A045| D0E7          BNE    RDSOSDIRLP ; BRANCH IF NEXT BLOCK TO READ LOW BYTE IS NOT
A047|                ; EQUAL TO ZERO
A047|
A047|                ;-----
A047|                ; This section searches the SOS directory for the SOS.KERNEL file.
A047|                ;-----
A047| AD 25A2        SRCHSOSKER LDA    MAINBUFF+25 ; LOAD ACCUMULATOR WITH FILE COUNT LOW BYTE
A04A| 85 E5          STA    FILECNT      ; STORE IN FILE COUNT LOW BYTE
A04C| AD 26A2        LDA    MAINBUFF+26 ; LOAD ACCUMULATOR WITH FILE COUNT HIGH BYTE
A04F| 85 E6          STA    FILECNT+1    ; STORE IN FILE COUNT HIGH BYTE
A051| 05 E5          ORA    FILECNT      ; OR ACCUMULATOR WITH FILE COUNT LOW BYTE
A053| D003          BNE    $010      ; BRANCH IF FILE COUNT IS NOT EQUAL TO ZERO
A055| 4C 56A1        JMP    WRNTFNDERR   ; JUMP TO WRITE NOT FOUND ERROR MESSAGE TO
A058|                ; SCREEN
A058| A5 E5          $010    LDA    FILECNT      ; LOAD ACCUMULATOR WITH FILE COUNT LOW BYTE
A05A| D002          BNE    $020      ; BRANCH IF NOT EQUAL TO $00
A05C| C6 E6          DEC    FILECNT+1    ; DECREMENT FILE COUNT HIGH BYTE
A05E| C6 E5          $020    DEC    FILECNT      ; DECREMENT FILE COUNT LOW BYTE
A060| A9 2B          LDA    #2B          ; LOAD ACCUMULATOR WITH $28
A062| 85 85          STA    IBBUFF      ; STORE IN BUFFER POINTER LOW BYTE
A064| A9 A2          LDA    #0A2         ; LOAD ACCUMULATOR WITH $A2
A066| 85 86          STA    IBBUFF+1    ; STORE IN BUFFER POINTER HIGH BYTE
A068| AE 24A2        LDX    MAINBUFF+24 ; LOAD X-REGISTER WITH ENTRIES PER BLOCK
A06B| CA            DEX          ; DECREMENT X-REGISTER
A06C| A0 00          SRCHLP   LDY    #00          ; LOAD Y-REGISTER WITH $00
A06E| B1 85          LDA    @IBBUPTMP,Y ; LOAD ACCUMULATOR WITH STORAGE TYPE AND NAME
A070|                ; LENGTH BYTE
A070| F01A          BEQ    $020      ; BRANCH IF EQUAL TO ZERO
A072| 29 0F          AND    #0F          ; MASK OFF BITS 4,5,6,7
A074| CD 92A1        CMP    FLNMLELEN   ; COMPARE WITH FILE NAME LENGTH
A077| D013          BNE    $020      ; BRANCH IF NOT EQUAL TO ZERO
A079| A8            TAY          ; TRANSFER NAME LENGTH TO Y-REGISTER
A07A| B1 85          $010    LDA    @IBBUPTMP,Y ; LOAD ACCUMULATOR WITH FILE NAME BYTE
A07C| D9 92A1        CMP    FINME-1,Y   ; COMPARE WITH FILE NAME BYTE
A07F| D00B          BNE    $020      ; BRANCH IF NOT EQUAL
A081| 88            DEY          ; DECREMENT NAME LENGTH
A082| D0F6          BNE    $010      ; BRANCH IF NAME LENGTH NOT EQUAL TO ZERO
A084| B1 85          LDA    @IBBUPTMP,Y ; LOAD ACCUMULATOR WITH STORAGE TYPE AND NAME
A086|                ; LENGTH BYTE
A086| 29 F0          AND    #0F0        ; MASK OFF BITS 0,1,2,3
A088| C9 20          CMP    #20         ; COMPARE WITH $20 FOR SAPLING FILE
A08A| F032          BEQ    READIDXBLK  ; BRANCH IF EQUAL TO READ INDEX BLOCK
A08C| 08            $020    PHP          ; PUSH PROCESSOR STATUS ON STACK
A08D| CA            DEX          ; DECREMENT ENTRIES PER BLOCK
A08E| F010          BEQ    $030      ; BRANCH IF ENTRIES PER BLOCK IS EQUAL TO ZERO
A090| 18            CLC          ; CLEAR CARRY
A091| A5 85          LDA    IBBUFF      ; LOAD ACCUMULATOR WITH BUFFER POINTER LOW BYTE
A093| 6D 23A2        ADC    MAINBUFF+23 ; ADD ENTRY LENGTH LOW BYTE
A096| 85 85          STA    IBBUFF      ; STORE IN BUFFER POINTER LOW BYTE
A098| A5 86          LDA    IBBUFF+1    ; LOAD ACCUMULATOR WITH BUFFER POINTER HIGH BYTE
A09A| 69 00          ADC    #00         ; ADD $00
A09C| 85 86          STA    IBBUFF+1    ; STORE IN BUFFER POINTER HIGH BYTE
A09E| D009          BNE    $040      ; BRANCH ALWAYS
A0A0| A9 04          $030    LDA    #04          ; LOAD ACCUMULATOR WITH $04
A0A2| 85 85          STA    IBBUFF      ; STORE IN BUFFER POINTER LOW BYTE
A0A4| E6 86          LDA    IBBUFF+1    ; LOAD ACCUMULATOR WITH BUFFER POINTER HIGH BYTE
A0A6| AE 24A2        LDX    MAINBUFF+24 ; LOAD X-REGISTER WITH ENTRIES PER BLOCK
A0A9| 28            $040    PLP          ; PULL PROCESSOR STATUS FROM STACK
A0AA| F0C0          BEQ    SRCHLP     ; BRANCH IF NOT EQUAL TO ZERO
A0AC| 38            SEC          ; SET CARRY
A0AD| A5 E5          LDA    FILECNT      ; LOAD ACCUMULATOR WITH FILE COUNT LOW BYTE
A0AF| E9 01          SBC    #01         ; SUBTRACT $01
A0B1| 85 E5          STA    FILECNT      ; STORE IN FILE COUNT LOW BYTE
A0B3| A5 E6          LDA    FILECNT+1    ; LOAD ACCUMULATOR WITH FILE COUNT HIGH BYTE
A0B5| E9 00          SBC    #00         ; SUBTRACT $00
A0B7| 85 E6          STA    FILECNT+1    ; STORE IN FILE COUNT HIGH BYTE
A0B9| B0B1          BCS    SRCHLP     ; BRANCH IF MORE FILE ENTRIES
A0BB| 4C 56A1        JMP    WRNTFNDERR   ; JUMP TO WRITE NOT FOUND ERROR MESSAGE TO
A0BE|                ; SCREEN
A0BE|
A0BE|                ;-----
A0BE|                ; This section reads in the index block of the SOS.KERNEL file.
A0BE|                ;-----
A0BE|
A0BE| A0 11          READIDXBLK LDY    #11          ; LOAD Y-REGISTER WITH $11
A0C0| B1 85          LDA    @IBBUPTMP,Y ; LOAD KEY POINTER LOW BYTE
A0C2| AA            TAX          ; TRANSFER ACCUMULATOR TO X-REGISTER-BLOCK LOW
A0C3|                ; BYTE

```

10/31/89 9:45

HD:Apple ///:SOS Floppy Bootstrap Loader

Page 3

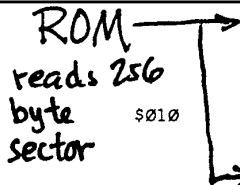
```

A0C3| C8                INY                ; INCREMENT Y-REGISTER
A0C4| B1 85            LDA                @IBBUF, Y ; LOAD KEY POINTER HIGH BYTE
A0C6| A0 00            LDY                #00                ; LOAD Y-REGISTER WITH $00
A0C8| 84 85            STY                IBBUF                ; STORE IN BUFFER POINTER LOW BYTE
A0CA| A0 0C            LDY                #0C                ; LOAD Y-REGISTER WITH $0C
A0CC| 84 86            STY                IBBUF+1            ; STORE IN BUFFER POINTER HIGH BYTE
A0CE| 20 1DA1           JSR                READBLK            ; JUMP TO READ A BLOCK FROM FLOPPY DISK DRIVE
A0D1|
A0D1| ;-----
A0D1| ; This section reads in the first block of the SOS.KERNEL file.
A0D1| ;-----
A0D1|
A0D1| AE 000C           RDISOSKER LDX        IDXBLK1            ; LOAD X-REGISTER WITH INDEX BLOCK LOW BYTE
A0D4| AD 000D           LDA        IDXBLK2            ; LOAD ACCUMULATOR WITH INDEX BLOCK HIGH BYTE
A0D7| A0 00            LDY                #00                ; LOAD Y-REGISTER WITH $00
A0D9| 84 85            STY                IBBUF                ; STORE IN BUFFER POINTER LOW BYTE
A0DB| A0 1E            LDY                #1E                ; LOAD Y-REGISTER WITH $1E
A0DD| 84 86            STY                IBBUF+1            ; STORE IN BUFFER POINTER HIGH BYTE
A0DF| 20 1DA1           JSR                READBLK            ; JUMP TO READ A BLOCK FROM FLOPPY DISK DRIVE
A0E2|
A0E2| ;-----
A0E2| ; This section does a verification of the SOS.KERNEL file to make
A0E2| ; sure it is the proper SOS.KERNEL file. It checks for "SOS KRNL" in
A0E2| ; the first 8 bytes of the file.
A0E2| ;-----
A0E2|
A0E2| A0 08            FLVRFY      LDY        #08                ; LOAD Y-REGISTER WITH $08
A0E4| B9 FF1D           FLVRFYLP   LDA        LOADADR-1, Y ; LOAD ACCUMULATOR WITH BYTE FROM SOS.KERNEL
A0E7| D9 9CA1           CMP        FLVERIFY-1, Y ; COMPARE WITH VERIFICATION BYTE
A0EA| F003            BEQ        #010                ; BRANCH IF EQUAL
A0EC| 4C 6AA1           JMP        WRINKERERR          ; JUMP TO WRITE INVALID KERNEL ERROR MESSAGE TO
A0EF|                                     ; SCREEN
A0EF| 88                $010      DEY                ; DECREMENT Y-REGISTER
A0F0| D0F2            BNE        FLVRFYLP          ; BRANCH IF NOT EQUAL TO ZERO TO CHECK REST OF 8
A0F2|                                     ; SOS.KERNEL BYTES
A0F2| ;-----
A0F2| ; This section reads in the SOS.KERNEL file.
A0F2| ;-----
A0F2|
A0F2| A9 01            RDSOSKER   LDA        #01                ; LOAD ACCUMULATOR WITH $01
A0F4| 85 E7            STA        INDXBLKCNT        ; STORE IN INDEX BLOCK COUNT
A0F6| A4 E7            RDSOSKELP LDY        INDXBLKCNT        ; LOAD Y-REGISTER WITH INDEX BLOCK COUNT
A0F8| BE 000C           LDX        IDXBLK1, Y        ; LOAD X-REGISTER WITH BLOCK LOW BYTE
A0FB| B9 000D           LDA        IDXBLK2, Y        ; LOAD ACCUMULATOR WITH BLOCK HIGH BYTE
A0FE| D004            BNE        #010                ; BRANCH IF BLOCK HIGH BYTE IS NOT EQUAL TO ZERO
A100| E0 00            CPX        #010                ; CHECK TO SEE IF BLOCK LOW BYTE IS NOT EQUAL TO
A102|                                     ; ZERO
A102| F007            BEQ        JUMPSOSKER        ; BRANCH IF BLOCK LOW BYTE IS NOT EQUAL TO ZERO
A104| 20 1DA1           JSR        READBLK            ; JUMP TO READ A BLOCK FROM FLOPPY DISK DRIVE
A107| E6 E7            INC        INDXBLKCNT        ; INCREMENT INDEX BLOCK COUNT
A109| D0EB            BNE        RDSOSKELP        ; BRANCH IF INDEX BLOCK COUNT IS NOT EQUAL TO
A10B|                                     ; ZERO TO READ MORE OF THE SOS.KERNEL
A10B| ;-----
A10B| ; This section jumps to the SOS.KERNEL loader.
A10B| ;-----
A10B|
A10B| 18                JUMPSOSKER CLC                ; CLEAR CARRY
A10C| A9 0E            LDA        #0E                ; LOAD ACCUMULATOR WITH $0E
A10E| 6D 081E           ADC        OFFSET            ; ADD OFFSET LOW BYTE
A111| 85 E8            STA        SOSJMPADR         ; STORE IN SOS JUMP ADDRESS LOW BYTE
A113| A9 1E            LDA        #1E                ; LOAD ACCUMULATOR WITH $1E
A115| 6D 091E           ADC        OFFSET+1          ; ADD OFFSET HIGH BYTE
A118| 85 E9            STA        SOSJMPADR+1       ; STORE IN SOS JUMP ADDRESS HIGH BYTE
A11A| 6C E800           JMP        @SOSJMPADR        ; JUMP TO SOS.KERNEL LOADER
A11D|
A11D| ;-----
A11D| ; This section reads a block of data from the floppy disk drive.
A11D| ; On entry the x-register contains the block low byte and the
A11D| ; accumulator contains the block high byte.
A11D| ;-----
A11D|
A11D| 86 83            READBLK   STX        IBTRK            ; STORE BLOCK LOW BYTE IN TRACK NUMBER
A11F| 4A                LSR        A                ; DIVIDE BLOCK BY 8 TO GET TRACK NUMBER
A120| 66 83            ROR        IBTRK
A122| 4A                LSR        A
A123| 66 83            ROR        IBTRK
A125| 4A                LSR        A
A126| 66 83            ROR        IBTRK
A128| 8A                TXA                ; TRANSFER X-REGISTER WHICH CONTAINS THE BLOCK
A129|                                     ; LOW BYTE TO ACCUMULATOR
A129| 29 07            AND        #07                ; MASK OFF BITS 3,4,5,6,7
A12B| AA                TAX                ; TRANSFER ACCUMULATOR TO X-REGISTER
A12C| BD A0F4           LDA        SECTABL, X        ; LOAD ACCUMULATOR WITH PROPER SECTOR TO READ
A12F| 85 84            STA        IBSECT            ; STORE IN SECTOR NUMBER
A131| A9 01            LDA        #01                ; LOAD ACCUMULATOR WITH $01
A133| 85 87            STA        IBCMD             ; STORE IN COMMAND NUMBER
A135| A9 00            LDA        #00                ; LOAD ACCUMULATOR WITH $00
A137| 85 82            STA        IBDRVN           ; STORE IN DRIVE NUMBER

```

```

A139| 20 00F0      JSR   REGRWTS    ; JUMP TO READ A SECTOR FROM FLOPPY DISK
A13C| 9005        BCC   $010      ; BRANCH IF NO DISK ERRORS OCCURED
A13E| A2 FF      LDX   #0FF      ; LOAD ACCUMULATOR WITH $FF
A140| 9A          TXS   ; TRANSFER X-REGISTER TO STACK POINTER
A141| B03B        BCS   WRDISKERR  ; BRANCH TO WRITE DISK ERROR MESSAGE TO SCREEN
A143| E6 86      INC   IBBUFP+1  ; INCREMENT BUFFER POINTER HIGH BYTE
A145| E6 84      INC   IBSECT    ; INCREMENT SECTOR NUMBER
A147| E6 84      INC   IBSECT    ; INCREMENT SECTOR NUMBER
A149| 20 00F0      JSR   REGRWTS    ; JUMP TO READ A SECTOR FROM FLOPPY DISK
A14C| 9005        BCC   $020      ; BRANCH IF NO DISK ERRORS OCCURED
A14E| A2 FF      LDX   #0FF      ; LOAD ACCUMULATOR WITH $FF
A150| 9A          TXS   ; TRANSFER X-REGISTER TO STACK POINTER
A151| B02B        BCS   WRDISKERR  ; BRANCH TO WRITE DISK ERROR MESSAGE TO SCREEN
A153| E6 86      INC   IBBUFP+1  ; INCREMENT BUFFER POINTER HIGH BYTE
A155| 60          RTS   ; RETURN TO CALLER
A156|
A156| ;-----
A156| ; This section writes the not found error message to the screen.
A156| ;-----
A156|
A156| A2 1B      WRNTFNDERR LDX   #1B      ; LOAD X-REGISTER WITH $1B
A158| A0 21      LDY   #21      ; LOAD Y-REGISTER WITH $21
A15A| BD A4A1    LDA   $010      NTFNDERR-1,X ; LOAD ACCUMULATOR WITH NOT FOUND ERROR MESSAGE
A15D|           ; BYTE
A15D| 99 2806    STA   SCREENLOC,Y ; WRITE IT TO THE SCREEN
A160| 88        DEY   ; DECREMENT Y-REGISTER
A161| CA        DEX   ; DECREMENT X-REGISTER
A162| D0F6      BNE   $010      ; BRANCH IF MORE CHARACTERS TO WRITE ON SCREEN
A164| AD 40C0    LDA   IOBEEP   ; BEEP SPEEKER
A167| 4C 67A1    JMP   $020      ; HANG FOREVER !!
A16A|
A16A| ;-----
A16A| ; This section writes the invalid kernel error message to the screen.
A16A| ;-----
A16A|
A16A| A2 13      WRINKERERR LDX   #13      ; LOAD X-REGISTER WITH $13
A16C| A0 1D      LDY   #1D      ; LOAD Y-REGISTER WITH $1D
A16E| BD BFA1    LDA   $010      INVKEERR-1,X ; LOAD ACCUMULATOR WITH INVALID KERNEL ERROR
A171|           ; MESSAGE BYTE
A171| 99 2806    STA   SCREENLOC,Y ; WRITE IT TO THE SCREEN
A174| 88        DEY   ; DECREMENT Y-REGISTER
A175| CA        DEX   ; DECREMENT X-REGISTER
A176| D0F6      BNE   $010      ; BRANCH IF MORE CHARACTERS TO WRITE ON SCREEN
A178| AD 40C0    LDA   IOBEEP   ; BEEP SPEEKER
A17B| 4C 7BA1    JMP   $020      ; HANG FOREVER !!
A17E|
A17E| ;-----
A17E| ; This section writes the disk error message to the screen.
A17E| ;-----
A17E|
A17E| A2 0A      WRDISKERR LDX   #0A      ; LOAD X-REGISTER WITH $0A
A180| A0 18      LDY   #18      ; LOAD Y-REGISTER WITH $18
A182| BD D2A1    LDA   $010      DISKERR-1,X ; LOAD ACCUMULATOR WITH DISK ERROR MESSAGE BYTE
A185| 99 2806    STA   SCREENLOC,Y ; WRITE IT TO THE SCREEN
A188| 88        DEY   ; DECREMENT Y-REGISTER
A189| CA        DEX   ; DECREMENT X-REGISTER
A18A| D0F6      BNE   $010      ; BRANCH IF MORE CHARACTERS TO WRITE ON SCREEN
A18C| AD 40C0    LDA   IOBEEP   ; BEEP SPEEKER
A18F| 4C 8FA1    JMP   $020      ; HANG FOREVER !!
A192|
A192| ;-----
A192| ; STORAGE FOR THE ERROR MESSAGE AND FILE VERIFICATION ROUTINES
A192| ;-----
A192|
A192| 0A        FLNMELEN .BYTE 0A
A193| 53 4F 53 2E 4B 45 52 FLNME .ASCII "SOS.KERNEL"
A19A| 4E 45 4C
A19D| 53 4F 53 20 4B 52 4E FLVERIFY .ASCII "SOS KRNL"
A1A4| 4C
A1A5| 46 49 4C 45 20 27 53 NTFNDERR .ASCII "FILE 'SOS.KERNEL' NOT FOUND"
A1AC| 4F 53 2E 4B 45 52 4E
A1B3| 45 4C 27 20 4E 4F 54
A1BA| 20 46 4F 55 4E 44
A1C0| 49 4E 56 41 4C 49 44 INVKEERR .ASCII "INVALID KERNEL FILE"
A1C7| 20 4B 45 52 4E 45 4C
A1CE| 20 46 49 4C 45
A1D3| 44 49 53 4B 20 45 52 DISKERR .ASCII "DISK ERROR"
A1DA| 52 4F 52
A1DD|
A1DD| .END
    
```



SYMBOL TABLE DUMP

AB - Absolute LB - Label UD - Undefined MC - Macro
 RF - Ref DF - Def PR - Proc FC - Func
 PB - Public PV - Private CS - Consts

BOOTSTRA PR ---- | BREG AB FFEF | DISKERR LB A1D3 | ENTRY LB A000 | EREG AB FFDF |

10/31/89 9:45

HD:Apple ///:SOS Floppy Bootstrap Loader

Page 5

```

FILECNT AB 00E5 | FIRSTPAG AB 2000 | FLNME LB A193 | FLNMELEN LB A192 | FLVERIFY LB A19D |
FLVRFY LB A0E2 | FLVRFYLP LB A0E4 | IBBUFP AB 0085 | IBBUPTM AB 00E3 | IBCMD AB 0087 |
IBDRVN AB 0082 | IBSECT AB 0084 | IBTRK AB 0083 | IOBEEP AB C040 | JUMPSOSK LB A10B | KBDSTROB AB C010 |
INDXBLKC AB 00E7 | INVKEERR LB A1C0 | IOBEEP AB C040 | JUMPSOSK LB A10B | KBDSTROB AB C010 |
LOADADR AB 1E00 | MAINBUFF AB A200 | NMIVECTO AB FFCA | NTFNDERR LB A1A5 | OFFSET AB 1E08 |
RD1SOSKE LB A0D1 | RDSOSDIR LB A02E | RDSOSKEL LB A0F6 | RDSOSKER LB A0F2 | READBLK LB A11D |
READIDXB LB A0BE | READSOSD LB A024 | REGRWTS AB F000 | RETINT AB 0040 | SCREENLO AB 0628 |
SECTABL AB F4A0 | SOSJMPAD AB 00E8 | SRCHLP LB A06C | SRCHSOSK LB A047 | WRDISKER LB A17E |
WRINKERE LB A16A | WRNTPNDE LB A156 |
    
```

Assembly complete: 363 lines
 0 Errors flagged on this Assembly

 6502 OPCODE STATIC FREQUENCIES

```

ADC : 4 | ****
AND : 3 | ***
BCC : 2 | **
BCS : 3 | ***
BEQ : 6 | *****
BIT : 1 m *
BNE : 15 | *****
CLC : 2 | **
CLD : 1 m *
CMP : 4 | ****
CPX : 2 | **
DEC : 3 | ***
DEX : 5 | *****
DEY : 5 | *****
INC : 6 | *****
INY : 2 | **
JMP : 7 | *****
JSR : 6 | *****
LDA : 37 M *****
LDX : 12 | *****
LDY : 14 | *****
LSR : 3 | ***
ORA : 1 m *
PHP : 1 m *
PLP : 1 m *
ROR : 3 | ***
RTS : 1 m *
SBC : 2 | **
SEC : 1 m *
SEI : 1 m *
STA : 23 | *****
STX : 2 | **
STY : 6 | *****
TAX : 3 | ***
TAY : 1 m *
TXA : 1 m *
TXS : 3 | ***
    
```

Minimum frequency = 1
 Maximum frequency = 37

Average frequency = 5

Unused opcodes:

ASL BMI BPL BRK BVC BVS CLI CLV CPY EOR INX NOP PHA PLA ROL RTI
 SED TSX TYA

Program opcode usage: 66 %

 (1.00) That's all, Folks ...

*seems like an
early version*

• Apple /// Computer Information

APPLE /// SOS BOOTSTRAP LOADER HEXADECIMAL DUMP

Source

DISK1.dofile as found with Chris Smolinski's Macintosh SARA emulator application

Printed by David T. Craig • December 1997

This hex dump, which was produced by the Apple Macintosh MPW DumpFile tool, lists the Apple /// SOS Bootstrap Loader. This 512 byte loader exists at block 0 of SOS disks and is loaded by the Apple /// ROM into memory addresses \$A000-\$A1FF. This code's purpose is to begin the loading of SOS from the floppy disk into the ///'s memory.

```

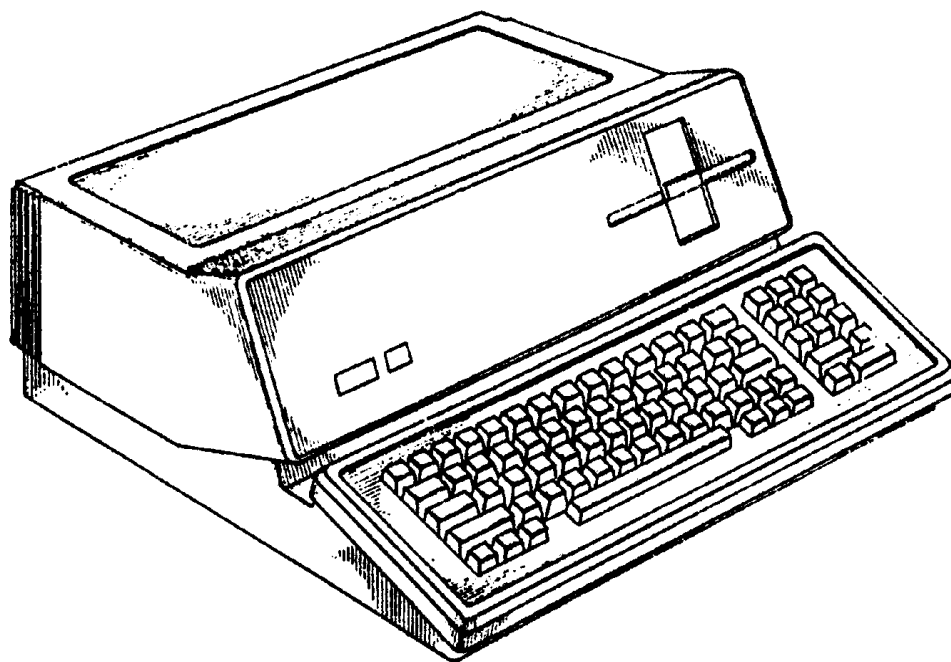
0: 4C 6E A0 53 4F 53 20 42 4F 4F 54 20 20 31 2E 31 Ln+SOS.BOOT..1.1
10: 20 0A 53 4F 53 2E 4B 45 52 4E 45 4C 20 20 20 20 ..SOS.KERNEL....
20: 20 53 4F 53 20 4B 52 4E 4C 49 2F 4F 20 45 52 52 .SOS.KRNLi/O.ERR
30: 4F 52 08 00 46 49 4C 45 20 27 53 4F 53 2E 4B 45 OR..FILE.'SOS.KE
40: 52 4E 45 4C 27 20 4E 4F 54 20 46 4F 55 4E 44 25 RNEL'.NOT.FOUND%
50: 00 49 4E 56 41 4C 49 44 20 4B 45 52 4E 45 4C 20 .INVALID.KERNEL.
60: 46 49 4C 45 3A 00 00 0C 00 1E 0E 1E 04 A4 78 D8 FILE:.....$xÿ
70: A9 77 8D DF FF A2 FB 9A 2C 10 C0 A9 40 8D CA FF @wçfl~ç'ö,.,¿@æç ~
80: A9 07 8D EF FF A2 00 CE EF FF 8E 00 20 AD 00 20 @.çÖ~ç.ÆÖ~é.≠..
90: D0 F5 A9 01 85 E0 A9 00 85 E1 A9 00 85 A9 A2 -i@.Ö†@.Ö.©.ÖÖ©ç
A0: 85 86 20 BE A1 E6 E0 A9 00 85 E6 E6 86 E6 86 E6 ÖÜ.æ°Ê†@.ÖÊËÜÊÜÊ
B0: E6 20 BE A1 A0 02 B1 85 85 E0 C8 B1 85 85 E1 D0 Ê.æ°†.±ÖÖ†»±ÖÖ.-
C0: EA A5 E0 D0 E6 AD 6C A0 85 E2 AD 6D A0 85 E3 18 Í.†-Ê≠†tÖ,≠m†Ö,,.
D0: A5 E3 69 02 85 E5 38 A5 E2 ED 23 A4 85 E4 A5 E5 .„i.ÖÂ8.,Ì#SÖ%•Â
E0: E9 00 85 E5 A0 00 B1 E2 29 0F CD 11 A0 D0 21 A8 È.ÖÂ†.±,).Ö.†-!@
F0: B1 E2 D9 11 A0 D0 19 88 D0 F6 A0 00 B1 E2 29 F0 ±,ÿ.†-.à-^†.±,)
100: 53 4F 53 20 4B 52 4E 4C 62 00 01 00 0E 2E 44 31 SOS.KRNLb....D1
110: 2F 53 4F 53 2E 49 4E 54 45 52 50 AA A5 A0 F9 A0 /SOS.INTERP™.†~†
120: A0 A5 A0 A0 A5 A0 A0 C5 A0 A0 98 A0 F0 A1 A0 CC †.††.††~††ò†•††Ä
130: A0 A0 C5 A0 A0 A0 A0 EE A0 A0 C4 0E 2E 44 31 ††~††††††††††.D1
140: 2F 53 4F 53 2E 44 52 49 56 45 52 FF 9A A0 FF 9A /SOS.DRIVER~ö†~ö
150: A0 A0 A0 A0 D0 A0 A0 C1 A0 A0 8A A0 A0 F9 A0 C1 ††††-††;††ä††††;
160: E9 A0 9E A1 A0 F5 A0 A0 A5 A0 A0 88 00 00 88 0C È†û°††††.††à..à.
170: A9 00 AA 9D 00 1A 9D 00 16 9D 00 1B 9D 00 18 9D ©.™ù..ù..ù..ù.ù
180: 00 14 9D 00 01 CA D0 EB A9 30 8D DF FF A2 FB 9A ..ù..-Î©0çfl~ç'ö
190: A9 1A 8D D0 FF 20 D4 1F AD DF FF 29 10 09 28 8D ©.ç-~.~.≠fl~)..(ç
1A0: DF FF A2 FF 9A A9 1A 8D D0 FF AD 01 19 8D EF FF fl~ç~ö©.ç-~.~.çÖ~
1B0: 6C 02 00 AA AD EF FF 48 8E EF FF A5 27 05 26 F0 1..™≠Ö~HéÖ~.~.&
1C0: 33 A5 26 D0 02 C6 27 C6 26 18 A5 23 65 27 85 23 3•&-..Δ'Δ&..#e'Ö#
1D0: A5 25 65 27 85 25 E6 27 A4 26 F0 07 B1 22 91 24 •%e'Ö%Ê'§&•.±"è$
1E0: 88 D0 F9 B1 22 91 24 88 C6 23 C6 25 C6 27 D0 EC à-~±"è$àΔ#Δ%Δ'-Ï
1F0: E6 23 E6 25 68 8D EF FF 60 18 A5 24 65 10 85 10 Ê#Ê%hçÖ~`.~$e.Ö.

```

###



Apple /// Computer Information

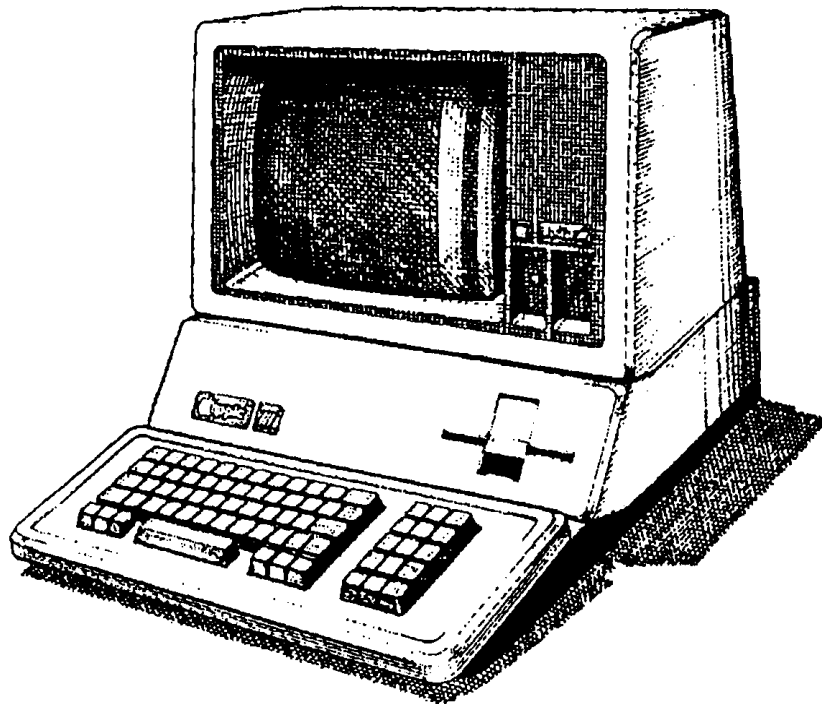


APPLE /// EMULATOR IDEAS

ADDED BY DAVID T CRAIG • 2006



Apple III Computer Information



Apple III Emulator Ideas

Version 4 • 12 Dec 1997



SOME IDEAS ABOUT AN APPLE /// COMPUTER EMULATOR

David T. Craig -- 12 December 1997 -- Version 4

941 Calle Mejia #1006, Santa Fe, NM 87501 USA
 e-mail: 71533.606@compuserve.com

TABLE OF CONTENTS

1.0 PURPOSE
 2.0 EMULATOR GOALS
 3.0 EMULATOR USER INTERFACE
 4.0 DISK IMAGES
 5.0 6502 CPU EMULATION
 6.0 ROM EMULATION
 7.0 MEMORY-MAPPED I/O EMULATION
 8.0 MEMORY BANK SWITCHING EMULATION
 9.0 SOS SYSTEM CALL EMULATION
 10.0 DEVICE DRIVER EMULATION
 11.0 KEYBOARD SUPPORT
 12.0 MONITOR SUPPORT
 13.0 APPLE][EMULATION DISK SUPPORT
 14.0 WHAT LANGUAGE SHOULD THE /// EMULATOR BE WRITTEN IN?
 15.0 WHAT TARGET MACHINES SHOULD BE SUPPORTED?
 16.0 EMULATOR DEBUGGING FACILITIES
 17.0 EMULATOR MEMORY STRUCTURE
 18.0 WHAT'S NEXT?
 19.0 REFERENCES

*disk:
 new: READFILES
 read all
 files from
 disk to
 host computer
 disk*

*Corrections — by page #
 1 - Mod. History = add 2 spaces
 9 - "C" char set bank → "K"
 10 - _ : add extra space between
 all build command
 line words.*

MODIFICATION HISTORY

28 Nov 1997 -- Version 1
 Created by David T. Craig.

04 Dec 1997 -- Version 2

New sections: MONITOR SUPPORT, EMULATOR DEBUGGING FACILITIES.
 Updated sections: DISK IMAGES, MEMORY BANK SWITCHING EMULATOR, SOS SYSTEM CALL EMULATION, REFERENCES.
 Added several good comments by Chris Smolinski (he's writing a /// emulator called SARA).

*14 - DISK BUFFER
 15 - SS*

*16 - BPE more general
 16-17 del some BP words*

21 - BPNIE FONT FONTRON

*13 - HAPP and name
 10 - change RD cmd to show
 (15) P and E bit names (upper case = 1, lower = 0)
 12 - ZPAGE - show offset byte line,
 same for adr1, adr2
 13 - S cmd*

09 Dec 1997 -- Version 3

DISK IMAGES: Updated info about DTCMake3///DiskImage Mac application, made disk image file an all-text file.
 SOS SYSTEM CALL EMULATION: typo Silentypr --> Silentye.
 WHAT TARGET MACHINES SHOULD BE SUPPORTED: More pre-68040 Mac comments.
 EMULATOR DEBUGGING FACILITIES: typo affects --> affect, added info about enabling/disabling SOS BRK disassembly, same for ProDOS, added list of emulator debugging commands.
 EMULATOR MEMORY STRUCTURE: New section.

12 Dec 1997 -- Version 4

EMULATOR DEBUGGING FACILITIES: Added examples to every debugging command. Added commands SNAPSHOTW, SNAPSHOTR, ZPAGE, SPAGE, EPAGE, DRIVERS, macro commands.

Some Ideas about an Apple /// Computer Emulator -- Version 4
 David T Craig -- 12 Dec 1997 -- 1 / 23



1.0 PURPOSE

This document describes some ideas about implementing a software emulator for the Apple /// computer. These ideas are based on my experiences with the Apple /// computer and its software programming. No specific target machine is mentioned in this document since these ideas should be non-target machine specific. These ideas are submitted to stimulate thought about such an emulator and hopefully inspire someone to produce a working Apple /// emulator.

The technical details behind the Apple /// computer, its operating system (SOS), and /// programs (e.g. AppleWriter ///) are based on my extensive collection of /// technical manuals, specification sheets, and many /// technical articles (Dr. John Jeppson's articles are very exhaustive and full of lots of neat /// techoid stuff). I have around 15 Apple manuals, the majority of which were published by Apple, which include user manuals and the technical programming manuals.

For those people seriously interested in implementing an Apple /// emulator program I highly recommend that they have at least the Apple /// Service Reference Manual. This manual, which is almost 500 pages long, is the definitive reference for how the Apple /// computer works. Most of its contents describe theory of operation even though its title suggests service-type information only. The important features of this manual for a /// emulator writer are the /// memory map and the /// memory mapped I/O locations.

I also own an Apple /// computer which still today works very well. I programmed the /// many moons ago and have worked professionally as an Apple Macintosh computer programmer since 1984.

Note: All comments are welcome. If you have anything to add or correct please let me know and I will update the master copy of this document.

2.0 EMULATOR GOALS

The /// emulator should provide a complete emulation environment for the faithful execution of Apple /// and /// Plus programs. As far as the emulator user is concerned when they run the emulator program their computer should work just like an Apple /// computer and all /// visual fidelity should be maintained. Emulation of the Apple /// Plus computer may also be supported (this means the /// Plus' interlaced screen). If the /// Plus is supported by the emulator you may want to let the user specify if they want to run a /// or a /// Plus.


I think it would be beyond neat if the emulator could run Apple's running horses demo and the other /// demos.

The /// emulator should support an Apple /// computer with at least 256K of memory and four floppy 140K disks (.D1, .D2, .D3, .D4). Support for 512K of memory may also exist since the ///'s operating system (SOS) supports up to 512K of memory. Memory size, if variable, should always be a multiple of 32K. I believe the lowest memory size supported by the /// (ROM?) is 96K. Support for a ProFile disk may also exist (for this disk there would need to be a disk image with a size of 5M). The first floppy disk (.D1) would correspond to the floppy disk drive that is built into the Apple ///. The other disks correspond to external disks and should exist as image files with specific file names (e.g. "Apple 3 D1", "Apple 3 D2", etc). The ProFile disk image file should also have a specific file name (e.g. "Apple 3 ProFile").

Image file names should have an extension (e.g. ".D3I") since this is needed by PCs.

3.0 EMULATOR USER INTERFACE

When the user runs the Apple /// emulator program the user should see on their computer screen a screen (or a window representing the screen on GUI systems) corresponding to the ///'s screen which the user would see if they were in front of a real Apple /// computer. All /// text and graphic modes should be supported by the

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 2 / 23



/// emulator (this includes the special modes supported by the /// Plus and its interlaced screen architecture).

I recommend that the emulator also support a screen dump facility that writes the current /// screen to either a text file (for text modes) or to a graphic file (for graphic modes) or always just creates a graphic file. The screen dump graphic file should be a standard graphic file for whatever target machine your support (e.g. on the IBM PC running Windows produce .BMP files, on the Apple Macintosh produce PICT files). Since the /// supports custom character sets dumping the screen to a PICT file (or to the target computer's clipboard) may be the best solution.

The emulator screen if implemented in a GUI window may also display a status area at the bottom of the window. This status area would display at least two lines of text and would keep the user informed of what the emulator was doing internally.

4.0 DISK IMAGES

The /// emulator should read disk image files which correspond directly to real /// 140K disks. When the /// emulator starts it should look in its folder and if there exists a /// disk image file the emulator should boot this image. If there are multiple disk image files then the emulator may want to display a list of these images and have the user select an image to boot.

The disk images should be exact copies of real /// disks. To make copies of these disks there should exist an utility program that runs on the /// computer and which outputs disk block data to the /// serial port (I plan to make this utility and call it DTCDumpIt). This utility's output should be a hex/ascii dump that specifies block numbers and has a checksum for each line of data. This utility should ask the user if it should dump a file or a disk.

On the target machine there should exist a similar utility that inputs the disk block data and creates a disk image file. I recommend that the transmitted disk block data consist of a hex dump with block number and checksum information in a human readable fashion. The receiving program (on the target computer) would read this human readable information, verify that the data was sent correctly, and produce binary disk image file images (I plan to create this utility for the Apple Macintosh and call it DTCMake///DiskImage).

There should also exist a disk image file for the ///'s Boot ROM (recommended file name: "Apple 3 Boot ROM"). This image should contain the 4K ROM image. This ROM should be the Revision 1 ROM (not Revision 0) since this was the last ROM produced and SOS 1.3 (the last SOS) requires this ROM.


Users should also be able to format a disk image by specifying the disk drive device name (e.g. .D2). Users should then be able to name the disk image so that they can use it later. Users should be able to assign specific disk images to specific disk drives.

I recommend that all disk image files have a very specific internal format. This format should support the verification of disk image files so that if a disk image file becomes corrupted in some fashion the /// emulator can detect this corruption, not use the image, and alert the user.

Note: Support for existing Apple][disk image files may be feasible but I recommend against this since the format of these images could change.

The proposed image format:

The disk image file contains two parts, a header part and a data part. The header part appears first followed by the data part. The header part contains identification and verification information. The data part contains the actual disk blocks for the /// disk. This file contains only text, no binary data appears here in any fashion. The only non-text information that can appear in these files is the Carriage Return (CR) and the Line Feed (LF) characters. The emulator should ignore

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 3 / 23



LFs if appropriate. All information appears in lines with a maximum length of 255 characters. Character case is immaterial. Blank lines are ignored. The reason for this format is so these image files can be transferred over the internet without the need for any binary-to-text conversion. Also, text-only files can easily be viewed by people using a word processor.

The header part contains:

Line	Comments
Signature	"APPLE /// DISK IMAGE"
Version	"VERSION" version number (e.g. "1")
Image Name	"IMAGE NAME" name of image, anything the user wants, most likely the name of the interpreter on the disk, e.g. "Apple Writer ///"
Creation Date	"CREATED" date image file created, "YYYY-MM-DD"
Created by Name	"CREATED BY" name of person or company who created this image
Comment	"COMMENT" comment for anything user wants
Data Size	"DATA SIZE" size of data part (decimal, e.g. "143360")
Data Checksum	"DATA CHECKSUM" hexadecimal checksum (e.g. "FA7C3188")
Reserved 1	"RESERVED"
Reserved 2	"RESERVED"
Reserved 3	"RESERVED"
Reserved 4	"RESERVED"
Tech Comment	"TECH COMMENT" name of program that this is for
Header Checksum	"HEADER CHECKSUM" hexadecimal checksum (e.g. "B97C31D5")

Notes:

The checksum should be calculated as the exclusive-OR of each byte followed by a left rotation of 1 bit. Checksum starts with zero. Checksums should always be 4 bytes in size and be stored in the header as an 8 character string.

The Tech Comment's purpose is to allow people who obtain an image file to be able to contact someone about the file's purpose.

The data part contains lines representing 16 bytes from the original disk. Each line has a specific format which begins with the starting disk address for the line, 16 bytes, the ASCII equivalent of the 16 bytes, and a checksum for the bytes of the line with the format:

```
[00000000] 0123 4567 89ab cdef 0123 4567 89ab cdef [1234567890123456] 12345678
```

The last line of the file must be the word "FINIS".

Sample disk image file:

```
APPLE /// DISK IMAGE
VERSION 1
IMAGE NAME Apple Writer ///
CREATED 1997-10-11
CREATED BY David T. Craig
COMMENT Thanks to Paul Lutus
DATA SIZE 16
DATA CHECKSUM FA7C3188
RESERVED
RESERVED
RESERVED
RESERVED
TECH COMMENT For David Craig's /// Emulator - 71533.606@compuserve.com
HEADER CHECKSUM B97C31D5

[00000000] 0123 4567 89ab cdef 0123 4567 89ab cdef [Apple.///.Emul..] FA7C3188
```

Some Ideas about an Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 4 / 23



FINIS

5.0 6502 CPU EMULATION

The heart of the /// emulator should be the emulation of the 6502 CPU. The heart may be referred to as the "6502 engine." The emulator should support all of the 6502 instructions, the 6502 registers, and the special Apple /// registers (e.g. the bank switch register, the environment register, and the zero-page register). Special register descriptions and usage can be found in the Apple /// SOS Reference Manual.

The 6502 engine must be smart about accessing memory and use the bank switch and environment registers correctly.

If this level of the /// emulation is complete and robust the rest of the /// emulator should work much more easily.

Support for special /// features may also exist at this level of the /// emulator. For example, the /// emulator may not want to emulate all of the ///'s memory-mapped I/O features, but instead intercept access to special areas or routines and call the target machine's operating system to handle these features. See sections ROM EMULATION and MEMORY-MAPPED I/O EMULATION for more details.

6.0 ROM EMULATION

The /// emulator should also support as much as possible the ///'s Boot ROM. This means the Boot ROM's routines should work for the most part as-is.

Note: I have a listing of the Boot ROM which could be useful for this emulation discussion.

For the Boot ROM's floppy disk I/O support I recommend that all the gory details here not be supported directly at the memory-mapped I/O level but instead the /// emulator should emulate this I/O. Specifically, the /// emulator should intercept any access to the Boot ROM routines which read or write disk blocks and use the appropriate target machine operating system routines to accomplish this feature.

The /// emulator should also initialize the ROM's character set which the ROM normally loads into a special RAM chip that is not accessible to the ///'s 6502 processor. See section MEMORY BANK SWITCHING EMULATION for more details.

7.0 MEMORY-MAPPED I/O EMULATION

All memory-mapped I/O locations that in some way deal with the physical world need to be handled by the /// emulator. These areas include such addresses as the speaker addresses. The Apple /// Service Reference Manual provides detailed information about these addresses.


All accesses to memory by the /// emulator must respect the bank switch and environment register settings so that the emulator does not try to access a memory-mapped address when that address is not mapped into the 6502 address space.

Programs which access low-level I/O locations such as the disk I/O addresses should not be supported. I assume most /// programs will access hardware components using SOS or device drivers.

Note: Chris Smolinski says that emulating the low-level stuff on a Power PC-based Macintosh is not very difficult and works rather fast (he's implemented in his SARA emulator the ///'s floppy disk I/O).

8.0 MEMORY BANK SWITCHING EMULATION

The /// emulator must also fully support the ///'s bank switched and enhanced indirect addressing memory architecture. Detailed descriptions and usage of /// memory handling can be found in the Apple /// SOS Reference Manual.

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 5 / 23



The /// emulator should also support the ///'s character set RAM chip. This holds the bitmap descriptions of each of the 128 characters in the /// character. This RAM area, which is not accessible to the ///'s 6502 CPU, holds 1024 bytes. See the Apple /// Standard Device Drivers Manual (Console Character Sets section) for more information.

Note: I believe the storage of the Boot ROM character set is different than the storage of the character set in the SOS.DRIVER file. I believe the ROM character set has bits that are reversed compared to the SOS.DRIVER character set.

The storage of text and graphics in memory should be supported also. This should happen automatically when a /// program writes to the text/graphic memory buffers. The emulator needs to detect such writes and update its screen as appropriate.

9.0 SOS SYSTEM CALL EMULATION

The majority of system calls to SOS and its drivers should most likely not be intercepted by the /// emulator. But certain calls may need to be intercepted unless a lower level of the /// emulator intercepts these feature already. System calls to SOS or drivers that may need intercepting by the /// emulator could be:

- o Disk I/O (.D[1-4] and .PROFILE drivers)
- o Keyboard I/O (.CONSOLE driver)
- o Screen I/O (.CONSOLE and .GRAPHIC drivers)
- o Sound generation (.AUDIO driver)
- o Serial port I/O (.RS232 driver)
- o Silentye Printer (.SILENTYPE) [I'm not sure about support for this]
- o Clock I/O (Y2K dates may be a problem)

I recommend that the /// emulator intercept all activity dealing with the above and have the target machine perform the equivalent features. For example, to read or write a disk block the /// emulator should have a routine that accesses the appropriate location in the disk image file.

The /// emulator may also provide the user with some type of setup options so that the user can specify specific properties of some of the above drivers. For example, if the target machine supports several output ports the emulator may let the user specify which port to use (e.g. for the .PRINTER driver the user could assign it to a specific serial or parallel port on the target machine).

Note: The ///'s clock does not support the year 2000 or greater. I think the emulator should support Y2K dates but I'm not sure if SOS's file system date stamps will support this easily.

10.0 DEVICE DRIVER EMULATION

This section is for the most part handled by my comments in section SOS SYSTEM CALL EMULATION. I suspect the programming within the /// emulator for this area could be the most work since there are lots of device drivers that make up a simple Apple /// configuration.

One area of device drivers that the /// emulator may not want to emulate is interrupt handling. Since the emulator does not have physical devices connected to it in any direct fashion I don't think interrupts exist as far as the emulator is concerned. Interrupts dealing with disks or the keyboard can be handled at a lower level by having the /// emulator call the appropriate system call in the target machine. These low-level I/O handlers should set up the appropriate driver data areas so that the rest of the ///'s software (SOS and the interpreter) will work correctly. For example, keyboard I/O should be setup in the /// emulator so that when the keyboard input memory-mapped I/O location is accessed the target machine OS really reads the keyboard and sets up the memory-mapped location as appropriate.

11.0 KEYBOARD SUPPORT

Some Ideas about an 🍏 Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 6 / 23



11.1 User interface support

The /// computer's keyboard layout is basically compatible with modern keyboards. The /// keyboard does have two extra keys, Open Apple and Closed Apple which are positioned to the left of the Apple /// keyboard. Also present on the keyboard are four arrow keys. The emulator should support these keys either directly (i.e., the target machine has similar keys) or associate other keys with the ///'s special keys (e.g., the Macintosh computer's two Option keys could be used to simulate the special Open and Closed Apple keys). The emulator's associated keys need not physically be in the same location as the ///'s special keys but having them in the general area will be beneficial.

Note: The /// Plus keyboard contains an extra key, Delete, compared to the /// keyboard.

11.2 Low-level access

The /// emulator should handle low-level access to the keyboard memory-mapped I/O locations as detailed in section DEVICE DRIVER EMULATION.

12.0 MONITOR SUPPORT

The emulator should support the Apple's built-in ROM Monitor. Entry to the Monitor should be similar to how this is done on a real /// (at startup if Open Apple and Control keys are pressed). The code in the ROM which tests for Monitor entry should work.

13.0 APPLE][EMULATION DISK SUPPORT

It would be nice if the /// emulator supported the Apple][Emulation Disk. I'm not sure of what would be involved here but suspect that if the ///'s 6502 CPU and the memory-mapped I/O locations are robustly supported that the][emulation should work also without any special additional /// emulation features.

Special consideration may need to be given to Apple /// keyboard keys which do not exist in the Apple][world.][emulation details can be found in the Apple /// Owner's Guide and the Apple /// Service Reference Manual.

Note: I have a disassembled listing of the Apple][Emulation Disk ROM source listing which could prove useful in this area.


Further analysis of the][emulation disk's boot sequence needs to be done since I'm unknowledgable about this area. Also, I've heard that the][emulation accesses an I/O location which disables some /// features.

14.0 WHAT LANGUAGE SHOULD THE /// EMULATOR BE WRITTEN IN?

I highly recommend that the /// emulator be written in a high level language such as Pascal or C. This should make the emulator more compatible with different target computers and make development and maintenance of the emulator much easier. I recommend avoiding low-level languages such as assembly.

15.0 WHAT TARGET MACHINES SHOULD BE SUPPORTED?

I recommend that the target machine (or machines) for the emulator be machines that are commonly used today by most computer users. This means either the IBM PC or the Apple Macintosh machine family. For the PC world I recommend the /// emulator run under Windows 95 and Windows NT. For the Macintosh world I recommend the emulator run on most Macintosh models which means support the Macintosh 512 and above. Color display should also be supported by the /// emulator (for the Macintosh this means use Color QuickDraw if the machine supports CQD and if CQD is not supported by a Macintosh model use the Classic B/W QD and maybe use patterns as "colors").

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 7/23



Any of these machines should be fast enough to emulate the /// and most likely will be too fast in many areas. I recommend some type of speed control be built into the emulator so that users can control how fast the emulator works. For many /// programs (e.g. AppleWriter /// and VisiCalc ///) emulation speed will be immaterial since these programs typically wait for the user to enter data and then do their thing. But for programs such as games the user will want to control the emulator speed otherwise the game's actions will be super fast and unplayable.

Some people say that the older machines such as pre-68040 Macintoshes will be too slow for a reasonable /// emulator. I would like to see this /// emulator run on a Mac 512 machine and onwards. Running on a Mac 128 machine seems a problem due to this machine's small memory size and should not be supported (if a virtual memory scheme was used by the emulator the Mac 128 could be supported but I think having this extra level of support in the emulator would not be worth it). I disagree and am willing to wager a small sum that I'm right.

16.0 EMULATOR DEBUGGING FACILITIES

The emulator should support a comprehensive built-in debugger. This debugger's purpose should be to let the sophisticated emulator user access any part of the emulator's /// address space. This should include all of the memory that is allocated to the /// as its memory. This memory would encompass the 256K (or 512K) of /// RAM, the /// ROM (4K), the character set RAM (1K), the 6502 registers, and the special /// registers (e.g. bank register).

This debugger will prove invaluable in diagnosing emulator bugs. Not only will the user be able to type commands for the debugger but the emulator will be able to send messages to the debugger.

Logging of all debugger sessions should be stored to a text file for possible analysis. This text file would be created when the emulator starts. The log file should be appended to by the emulator. Only the user can delete the file.

The debugger should exist as a separate window that does not in any way affect the emulator's main window. This window should display only commands that the user enters or replies returned by the debugger. There should not exist a separate window area showing things such as the 6502 registers since all such information should appear in the debugger log file. The window should support at least 80 columns of text and 24 rows.

The emulator user interface should be based on a simple command line control scheme. All commands and command outputs should be text-based. This scheme could be based on the ///'s Monitor's commands or on a little more readable command scheme such as in Apple's MacsBug debugger. There should be full on-line help that discusses the debugger commands in general and each command should also have on-line help available. The debugger should show at the beginning of each line a prompt character to indicate when it is waiting for a command. I recommend the prompt be the ">" character. The debugger should also show a cursor which I recommend to be a black square.


The debugger should support the standard debugging commands such as displaying/setting memory, displaying/setting registers, and disassembling 6502 instructions. This disassembly should support the special SOS BRK call by listing the word "BRK/SOS" instead of just "BRK" and following this with the SOS command number/name and the parameter list address:

```
SOS C0/CREATE 345A
```

The user should be able to enable or disable this feature.

Note: It may be good to also support the Apple][ProDOS command calling scheme in case this emulator ever becomes an Apple][emulator.

The debugger should support break points, single stepping, and timing buckets. The

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 8 / 23



timing buckets would be used in conjunction with break points to record how long a sequence of 6502 instructions took to execute. This can be very useful in locating emulator bottlenecks. The debugger supports many break point commands since I have a feeling that this facility will be very powerful and useful during the emulator's development.

The debugger should support the collection of statistics about the emulator. I recommend tracking how many times specific 6502 opcodes are executed (obviously, the debugger would need commands to display and clear this information). I would also track memory accesses on at least a page (256 bytes) basis.

The debugger should be accessible at any time that the emulator is running. I recommend some type of key press combination that the emulator would detect and display the debugger window. Once the debugger window is active it should remain on the screen until the user closes the window.

The emulator should also support a special key press combination at emulator startup time that activates the debugger just before the /// ROM is run. This can give the emulator developer a good way of tracing ROM execution.

The emulator should activate the debugger if any fatal emulation errors are detected and the debugger should show a message detailing the reason for the activation. All of these errors display a dump of the 6502 and SOS control registers. Reasons for debugger activation from the emulator are:

1. A program writes to write-protected memory (e.g. SOS's address space). The displayed message is "EMULATOR EXCEPTION: WRITING TO WRITE-PROTECTED MEMORY".
2. A program executes an undefined 6502 instruction (e.g. 6502 opcode \$02). The displayed message is "EMULATOR EXCEPTION: UNDEFINED 6502 OPCODE".

When the debugger is initialized (which should be when the emulator starts) the debugger should check if a text file named "DDT.TXT" exists. If so, the debugger should read each line from this file and execute it. Obviously, this file should contain debugger instructions. This can be very useful for setting up commonly used break points which if you use many would be tedious to type everytime you wanted to use the emulator.

A memory snapshot facility should also exist. When activated by a debugger command this facility would write to the host computer's disk a binary file containing a copy of all the /// memory areas. This snapshot should also be readable by the debugger so that the user could restart a specific emulation session from the snapshot.


I recommend the following emulator debugger commands which are based on the /// Monitor commands so that these debugger commands will be familiar to Monitor users. These commands for the most part have the general syntax of address-command. See my document "Inside the Apple /// Computer ROM" for a list of the /// Monitor commands. For information about the Apple][Monitor commands, which the /// Monitor commands are based upon, see "Apple][Reference Manual" (Chapter 3: The System Monitor, dated 1981).

Addresses appearing in debugger commands may be prefaced by "N/" where N is a bank number. For example, to reference address 2000 of bank 4 use 4/2000. If no bank number precedes an address the current bank is used. To reference a ROM address use a bank "number" of "R", for example "R/F000". To reference a character set address use a bank "number" of "C", for example "C/0000". To reference the SOS system bank use "S", e.g. "S/1400".

Commands should be case-insensitive (none of the UNIX case-sensitivity gobbly-gook).

Commands that display more than a screen full of information should either automatically pause when the screen is full, or the user can use the SPACE key.

Note: Commands using ":" may also use ";" which is easier to type since this

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 9 / 23



character does not need the user of the shift key. Same for "<" and "/".

Most debugger command numeric arguments must be specified in hexadecimal. The exception is the X command which supports hexadecimal, decimal, and binary.

The debugger command parser should be very liberal. This means that users should be able to include extra spaces (or no spaces) and the command should be parsable. For example, if a command needs a list of bytes the user should be able to enter any of the following: "AABBCC", "AA BB CC", " A ABBC C " and the debugger will see these as "AABBCC".

The debugger should also support a command macro facility. This facility allows you to define a macro consisting of other debugger commands. Typing the name of the macro will then type the commands as if you entered them manually.

=====

HELP ^(or ?) *cmd name*

Display debugger on-line help for all commands. Help info should be stored in an external text file for easier modification. I recommend that this section of this document be the help file.

Example: HELP

Note: ? also same

-- shows all help info

cmd name = 1st part or all cmd name

HELP S -- show all cmds starting w/ S

HELP SS -- shows SS cmd

BYE

Return to the emulator.

Example: BYE

CARRIAGE RETURN ^{keypress}

Repeat last command.

Example: If the last command was HELP and you press the CARRIAGE RETURN key then HELP will be displayed and executed again.

SPACE ^{keypress}

Pause current command's output. Press again to continue.

Example: If a command is executing and you press the SPACE key the comand's output will be paused, pressing SPACE again resumes the command's output. Pausing/Resuming are done on an output line basis only.

DELETE ^{keypress}

Stop current command's output.

Example: If a command is executing and you press this key then the command will stop executing and you will be returned to the debugger's prompt.

RD

Display 6502 registers and /// system control registers.

Example: RD

A=04 X=01 Y=D8 P=30/00000011 S=F8 PC=034A : E=77/01110111 Z=1A B=03

1st sp

bit names

bit names

*NV-B DIZC
Latched - shows 0 or 1*

S I W R S R R

Some Ideas about an Apple /// Computer Emulator -- Version 4

David T Craig -- 12 Dec 1997 -- 10/23

- S - System Clock rate
- I - I/O space
- C - Screen (Control) state
- R - Reset enable
- W - Write protect
- S - Stack in use
- R - ROM
- R - ROM

Show table explaining bits in P and E

- N negative
- V overflow
- B break command
- D decimal mode
- I interrupt disable
- Z zero
- C carry



byte:SA

Set 6502 A register to byte.

Example: 45:SA -- —

byte:SX

Set 6502 X register to byte.

Example: 7B:SX -- —

byte:SY

Set 6502 Y register to byte.

Example: FF:SY —

byte:SP

Set 6502 P register to byte.

Example: 56:SP —

byte:SS

Set 6502 S register to byte.

Example: AA:SS ~

word:SPC

Set 6502 PC register to word.

Example: 2000:SPC —

byte:SE

Set /// E system control register to byte.

Example: 34:SE —

byte:SZ


Set /// Z system control register to byte.

Example: 19:SZ /

byte:SB

Set /// B system control register to byte.

Example: 06:SB /

Some Ideas about an  Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 11 / 23



addr1.addr2

Dump memory data to screen from address 1 to address 2 and display ASCII character at the right of the screen.

Example (assumes current bank is bank 4): 300.30F -- (assuming bank 4 is current)
 0 1 2 3
 4/0300- B900 080A 0A0A 9900 08C8 D0F4 A62B A909 [F..d.uy%^&90@..G]

ZPAGE

Dump the contents of the current interpreter's Zero Page (256 bytes). Also supported are commands for the Stack Page and the Extend Page:

- SPAGE - stack page
- EPAGE - extend page

To dump the pages for SOS (and drivers) use the following commands:

- SZPAGE - zero page
- SSPAGE - stack page
- SEPAGE - extend page

Example: ZPAGE

Zero Page (interpreter)

00 1 2 3 4 5 6 7 ...
 1400- 0123456789ABCDEF 0123456789ABCDEF 0123456789ABCDEF 0123456789ABCDEF
 1420- 0123456789ABCDEF 0123456789ABCDEF 0123456789ABCDEF 0123456789ABCDEF
 ...
 14E0- 0123456789ABCDEF 0123456789ABCDEF 0123456789ABCDEF 0123456789ABCDEF

addr:bytes

Store starting at the address the bytes.

Example: 2000:AA BB CC DD EE FF --
 2000:AABBCCDDEEFF --

addr:'text'

Store text starting at address (high bit clear).

Example: 2000:'Hello World' --
 2000:'David's Dog' -- (this stores) David's Dog

addr:"text"

Store text starting at address (high bit set).

Example: 2000:"Hello World" --
 2000:"David's Dog" -- (this stores) David's Dog
 2000: "I said "Hi"" -- I said "Hi"

addr3<addr1.addr2M

Move data in address range to address 3.

Example: 2000<3000.3100M --

Some Ideas about an Apple /// Computer Emulator -- Version 4
 David T Craig -- 12 Dec 1997 -- 12 / 23



addr3<addr1.addr2V

Verify data in address range equals data starting at address 3.

Example: 2000<3000.3100V

Displays either "OK" if the verification succeeds, or "MISMATCH" if the verification fails.

bytes<addr1.addr2S

Search memory in address range for the bytes.

Example: AA<3000.3100S -- searches for byte AA
 AABCC<3000.3100S -- searches for bytes AA BB CC

If a search finds a match then the starting address of the match is displayed, otherwise "PATTERN NOT FOUND" is displayed.

PATTERN FOUND AT 4addr

'text'<addr1.addr2S

Search memory in address range for text (high bit clear).

Example: 'D'<3000.3100S -- =
 'David'<3000.3100S -- =

"text"<addr1.addr2S

Search memory in address range for text (high bit set).

Example: "D"<3000.3100S -- ~
 "David"<3000.3100S -- ~

disk.block<addr1.addr2W

Write address range to disk # disk starting at disk block. If disk # is not present then uses disk .D1. Disk should equal 1, 2, 3, or 4. The address range always ends on a block boundary no matter what you type.

Example: 1.117<2000.21FFW -- write 512 bytes to disk 1 block \$117

Note: Disk /// disks contain 280 blocks (\$118) so the block range is 0-117 (hexadecimal).

disk.block<addr1.addr2R

Read from disk # disk starting at block to the address range. If disk # is not present then uses disk .D1. See the W command for more info.

Example: 1.117<2000.21FFR -- read 512 bytes from disk 1 block \$117

disk.block-block:DISK

Read block range from disk # disk to a special debugger 4K buffer which is not used by the emulator. If the typed block range is greater than 4K then only the first 4K will be read. You can then examine this buffer's contents either with a hex/ascii



dump or with a disassembly (command L). This command is useful when you want to examine a disk's contents. For disassembly purposes, you can specify the logical starting address for the buffer. See the DISKBUFFER command.

To disassemble the special disk buffer (see the L command) use bank X (stands for "extra") as part of the disassembly address parameter (e.g. "X/100"). Same for dumping memory or whatever commands you want to use with this special buffer.

Example: 1.0-7:DISK -- read 8 blocks (0 to 7) from disk 1

addr:DISKBUFFER

Set disk buffer starting logical address. Default address is 2000. See the DISK command.

Example: A000:DISKBUFFER --

Range is 0000-FFFF

addr1.addr2L

Disassemble instructions in address range. If only addr1 appears then disassemble 20 instructions. Disassembly includes the opcode cycle count.

Example: 300L -- assumes bank 4 is current

4/0300-	A9 C1	'X.'	(2)	LDA #\$C1	;
4/0302-	20 ED FD	'...'	(5)	JSR \$FDED	;
4/0305-	18	'.'	(2)	CLC	;
4/0306-	69 0A	'T.'	(4)	ADC #\$01	;
4/0308-	C9 DB	'..'	(3)	CMP #\$DB	;
4/030A-	D0 F6	'..'	(3)	BNE \$0302	;
4/030C-	60	'U'	(4)	RTS	;
1	2	3	4	5	6 (see Note)

Note: Column 1 = bank register/address
 Column 2 = memory bytes
 Column 3 = ASCII for the memory bytes
 Column 4 = opcode cycle count
 Column 5 = disassembled instructions
 Column 6 = remark character ";" (optional, see DISASMREM)

L by itself disassembles the next 20 instructions.

DISASMREM

Display ";" after each disassembly line that is produced by the L command. Default is to not display the remark. Useful if you plan to add comments to a disassembly. See also DISASMREMOFF.

Example: DISASMREM

DISASMREMOFF

Turn off DISASMREM. See also DISASMREM.

Example: DISASMREMOFF

addrG

Some Ideas about an Apple /// Computer Emulator -- Version 4
 David T Craig -- 12 Dec 1997 -- 14 / 23



Call subroutine at the address.

Example: A000G -- ~~~~~

addrJ

Jump to the address.

Example: A000J -- ~~~~~

wordX

Convert word (or up to 4 hex digits) to hexadecimal, decimal, and binary (X stands for "translate"). Prefix character for byte determines its base: no prefix = hex, . = dec, t = binary.

Example: AX	->	A(16)	10(10)	0000	0000	0000	1010(2)
.10X	->	A(16)	10(10)	0000	0000	0000	1010(2)
t1010	->	A(16)	10(10)	0000	0000	0000	1010(2)
FFFFX	->	FFFF(16)	65535(10)	1111	1111	1111	1111(2)

Put in table for easier viewing

addr1.addr2:CS

Calculate and display a checksum for address range. Checksum is a 4 byte quantity which is calculated the same as the disk image file checksums.

Example: 300.500:CS -- ~~~~~
 CHECKSUM=AF897CEE

addrT

Trace instructions starting at the address. Each traced instruction displays register contents. Press the SPACE to pause the trace, press DELETE to stop the trace. The displayed registers contain values after the previously listed command executes.

Example: A000T -- assuming bank 4 is current

```

4/A000-  A9 C1      'X.'  (2)  LDA #$C1
A=C1  X=01  Y=D8  P=30/00000011  S=F8  PC=A002 : E=77/01110111  Z=1A  B=04
4/A002-  20 ED FD  '...' (5)  JSR $FDED
A=C1  X=01  Y=D8  P=30/00000011  S=F6  PC=FDED : E=77/01110111  Z=1A  B=04
    lisp                bit names                bit names
    
```

Note: Press the DELETE key to stop the trace, SPACE to pause/resume.

addrSS

Single step trace starting at the address. After each step pause and wait for user to press SPACE to continue or DELETE to stop the single step.

Example: A000T ^{SS} -- assuming bank 4 is current

```

4/A000-  A9 C1      'X.'  (2)  LDA #$C1
A=C1  X=01  Y=D8  P=30/00000011  S=F8  PC=A002 : E=77/01110111  Z=1A  B=04
    lisp                names                names
    
```

Note: Press SS by itself to single step the next instruction, or press CARRIAGE RETURN to repeat the SS.



addr:BP

Set a break point at address. When address is accessed the debugger is entered and displays the registers. Up to 100 break points should be supported.

Example: A000:BP

addr:BPC

Clear break point at address.

Example: A000:BPC

SOS:BP

Set a break point when a SOS call is made. This means when the BRK opcode is executed. Same as M00:BP.

Example: SOS:BP

Mopcode:BP

Set a break point when opcode is executed.

Example: M60:BP -- set break point when the RTS instruction (60) is executed.

ROM:BP

Set a break point when a call is made to the ROM.

Example: ROM:BP

addr1.addr2:BPW

Set a break point when any address within address range is written to. BPW = Break Point Write.

Example: 300.123AR:BPW

addr1.addr2:BPR

Set a break point when any address within address range is read from. BPR = Break Point Read.

Example: 300.123A:BPR

addr.byte:BPE

Set a break point when the address contents equal the byte value. BPE = Break Point Equals.

Example: 300.AA:BPE

make just 1 BPE command
addr1.addr2.byte1 byte2... : BPE (42 options)
addr1.addr2.byte1-byte2 : BPE (42 options)

addr.byte1-byte2:BPE

Set a break point when the address contents equal a byte value in the byte range. BPE

Some Ideas about an Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 16 / 23



*new cmd
BPNE BP not equals
same syntax as BPE*

= Break Point Equals.

Example: 300.AA-BB:BPE

addr.byte1 byte2 ... :BPEA *e*

Set a break point when the address contents equal byte 1 value, or equals byte 2 value, etc. Supports up to 16 byte values. BPEA = Break Point Equals Any.

Example: 300.AABBCCDD:BPEA
300.AA BB CC DD:BPEA

addr1.addr2.byte1 byte2 ... :BPEA *e*

Set a break point when the address range contains any bytes equalling the byte values. BPEA = Break Point Equals Any.

Example: 300.400.AABBCCDD:BPEA

addr1.addr2.byte1-byte2:BPEA *e*

Set a break point when the address range contents equal the byte range. BPEA = Break Point Equals Any.

Example: 300.400.AA-BB:BPEA

BPD

Display break point table.

Example: BPD

#	Address Range	BP	Setting
1	4/2000-4/21FF	BPEA	AA-BB

BPC

Clear break point table.

Example: BPC

addr1.addr2:TB

Set timing bucket for address range. When address 1 is accessed timing starts. When address 2 is accessed timing stops. Up to 100 timing buckets should be supported.

Example: A000.A1FF:TB

TBD

Display timing bucket table. Shows all set timing buckets and the time in 1/60th of a second and in seconds spent in each bucket.

Example: TBD

#	Address Range	Time (1/60s)	Time (secs)
---	---------------	--------------	-------------

Some Ideas about an Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 17 / 23



1	4/A000-4/A1FF	34	0.567
2	4/A300-4/A310	5	0.083
		39	0.650

addr:TBC

Clear timing bucket starting at address.

Example: A000:TBC

TBC

Clear timing bucket table.

Example: TBC

error:SOSE

List SOS general error message for the error number. If no error number is present then list all general errors. Error info should be stored in an external text file for easier modification. See the SOS Reference Manual for a list of these errors.

Example: 01:SOSE

BADSCNUM - Invalid SOS call number

error:SOSFE

Display SOS fatal error message for the error number. If no error number is present then list all fatal errors. See the SOS Reference Manual for a list of these errors.

Example: 01:SOSFE

BADBRK - Invalid BRK

command:SOS

Display SOS command name and SOS command area (e.g. file system) for the command number. If no command number present then list all SOS command numbers and their names. Command info should be stored in an external text file for easier modification. See the SOS Reference Manual for a list of these commands.

Example: C0:SOS

CREATE (File System)

SOSON

Turn on disassembly of SOS calls which displays SOS followed by the command number and parameter address. The emulator defaults to this.

Example: SOSON

SOSOFF

Some Ideas about an Apple /// Computer Emulator -- Version 4
 David T Craig -- 12 Dec 1997 -- 18 / 23



Turns off SOSON.

Example: SOSOFF

disk:CAT

Display catalog of SOS disk stored in disk # disk. Includes recursive list of all subdirectories. Should show same file info as Apple's System Utilities program.

Note: Other commands that may be supported include CATPASCAL for Apple][Pascal disks and CATDOS for Apple][DOS disks. This may come in handy if you want to see what these disks contain if you have them as disk image files.

Example: 1:CAT

disk.file_name:INFO

Displays information about the specified file in the disk. Information includes standard SOS file information but also block list of all index blocks (if any) associated with the file and block list of all data blocks for the file.

Example: 1.APPLE3.TEXT:INFO

disk.block:DUMP

Display contents of specified disk block in the standard hex/ascii dump format.

Example: 1.0:DUMP

disk:DRIVERS

Display list of contents of the SOS.DRIVER file stored on the disk. List includes driver names, driver information, and other items that are in the driver file (e.g. character sets).

Example: 1:DRIVERS

disk:CHECKIMAGE

Check validity of disk image in disk # disk. Computes header and data part checksums and compares against the image file's listed checksums.

Example: 1:CHECKIMAGE

DIT


Display Driver Information Table (DIT), a data structure maintained by this debugger. Contains list of all loaded drivers, their names, sizes, and entry point addresses.

Example: DIT

MIT

Display Memory Information Table (MIT), a data structure maintained by this debugger. See section EMULATOR MEMORY STRUCTURE for what this structure contains.

Example: MIT

Some Ideas about an  Apple /// Computer Emulator -- Version 4
 David T Craig -- 12 Dec 1997 -- 19/23



OPCODES

Display a histogram of opcode execution counts. Includes the actual number of the counts. Sorted by frequency. Opcodes not executed are listed below the histogram.

Example: OPCODES

```
LDA  2,188,973 *****
STA  12,123  *****
CMP  467     *****
-----
      2,201,563
```

Unexecuted opcodes: TXS NOP

OPCODESCLR

Reset opcode histogram table.

Example: OPCODESCLR

page1.page2:MEMORYR

Display memory write access table. This table lists on a 256 byte page basis counts for each time the page was read. If page1.page2 specified then lists only those pages. If a single page is specified then display only that page's access count.

Example: 0.5:MEMORYR

page1.page2:MEMORYW

Display memory read access table. This table lists on a 256 byte page basis counts for each time the page was written. See MEMORYW for page options.

Example: 0.5:MEMORYW

MEMORYCLR

Reset both memory access tables.

Example: MEMORYCLR

value:SCROLL

Set debugger display scrolling rate interline delay. Value is in 1/10th of a second. Default is no delay (value = 0). Useful if you want to for example dump lots of memory and don't want to mess with the SPACE key to read what is displayed. Set the scrolling delay to a comfortable value, sit back, and enjoy the show.

Example: 10:SCROLL -- sets scrolling delay to 1 second

filename:LOG

Close log file, create a new one with filename, and output all debugger displays to this new file. Useful if you're running the emulator from a write-protected disk and you want to re-direct the output to a writable disk file.

*if SCROLL > 0 then ends showing more than screen of
 14 for do not pause for user when screen full*



Example: MyDiary:LOG

SNAPSHOTW

Write the contents of all of the emulator's memory to binary file on the host computer's hard disk. This snapshot could prove useful in diagnosing an emulator problem. The binary file should be named "Snapshot_YYYYMMDD_HHMMSS.BIN".

Example: SNAPSHOTW

SNAPSHOTRfile-name

Read a snapshot file into the emulator's memory.

Example: SNAPSHOTR Snapshot_19971225_123456.BIN

MACRO name commands

Define a macro name and commands for this macro. You can use any name containing alphanumeric characters or periods with a maximum length of 31 characters. Up to 25 macros may be defined. All commands are verified and if any syntax errors occur you will be told and the macro will not be defined. Macro commands cannot include other macro commands.

Example: MACRO my.dump 300.400 A000.A1FF A000L

MACROL

List all defined macros.

Example: MACROL

```
# Name / Contents
-----
1 my.dump
  300.400 A000.A1FF A000L
```

!macro-name

Execute a macro with the name "macro-name". Each command within the macro is displayed followed by the commands' display.

Example: !my.dump

```
300.400
...
A000.A1FF
...
A000L
...
```

FONT display current font bitmap font?ROM ROM font bitmap

VERSION

Display debugger version information. Includes version number and creation date/time.

=====

Some Ideas about an Apple /// Computer Emulator -- Version 4
 David T Craig -- 12 Dec 1997 -- 21 / 23



17.0 EMULATOR MEMORY STRUCTURE

I recommend that the emulator's internal memory structure for the Apple /// memory resources be structured as follows:

o Memory block containing the size of memory and references to each /// memory bank (the references can be whatever is appropriate -- on the Mac these could be Mac memory pointers or handles):

- number of switchable banks (1..15)
- reference to bank S (32K: 0000-1FFF, A000-FFFF) *
- reference to bank 0/\$0 - switchable (32k: 2000-9FFF)
- reference to bank 1/\$1 - switchable (32k: 2000-9FFF)
- ...
- reference to bank 14/\$E - switchable (32k: 2000-9FFF)
- reference to Boot ROM ROM address space (4k: F000-FFFF)
- reference to Boot ROM RAM address space (4k: F000-FFFF)
- reference to I/O RAM address space (4k: C000-CFFF)

* The system (S) bank is always on-line and is never bank switched. SOS and part of the interpreter reside here.

o Memory block containing the 6502 registers:

- Accumulator (A) 8 bits
- X index (X) 8 bits
- Y index (Y) 8 bits
- Status Register (P) 8 bits
- Stack Pointer (S) 8 bits
- Program Counter (PC) 16 bits

o Memory block containing the special /// System Control Registers:

- E: Environment Register (FFDF) 8 bits
- Z: Zero Page Register (FFD0) 8 bits
- B: Bank Register (FFEF) 8 bits

18.0 WHAT'S NEXT?

Persons seriously interested in creating an Apple /// emulator program should try to obtain as much /// technical information as possible. The author has lots of info which he can copy at minimal charge (10 cents per page plus postage). These persons should also have access to a working Apple /// computer with a fair number of /// programs.

Other areas of compatibility should also be investigated that this document does not address. This includes support for other input devices such as the mouse which does have a 3rd party driver available.

19.0 REFERENCES

- Apple /// Owner's Guide, Apple Computer, 1981
- Apple /// Plus Owner's Guide, Apple Computer, 1982
- Apple /// System Data Sheet, Apple Computer, July 1983
- Apple /// Plus System Data Sheet, Apple Computer, October 1983
- Apple /// Standard Device Drivers Manual, Apple Computer, 1981

Some Ideas about an Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 22 / 23

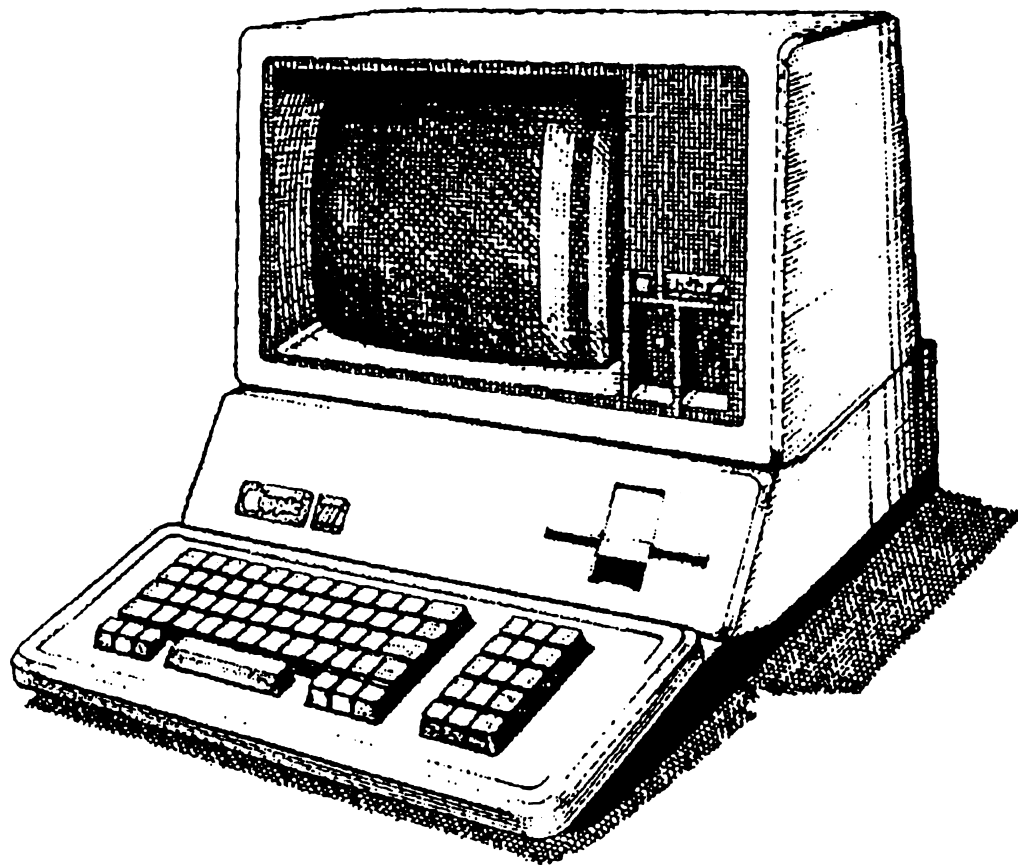


Apple /// SOS Reference Manual, Apple Computer, 1982
Apple /// SOS Device Driver Writer's Guide, Apple Computer, 1982
Apple /// Service Reference Manual (Level 2), Apple Computer, 1983
/// Bits: John Jeppson's Guided Tour of Highway ///, Softalk magazine, May 1983
Bank Switch Razzle-Dazzle, Softalk magazine, August 1982
The Apple Nobody Knows, Apple Orchard magazine, Fall 1981
Apple /// Entry Points, Andy Wells, Call-APPLE, October 1981
Inside the Apple /// Computer ROM, David Craig, November 1997
###

Some Ideas about an 🍏 Apple /// Computer Emulator -- Version 4
David T Craig -- 12 Dec 1997 -- 23 / 23



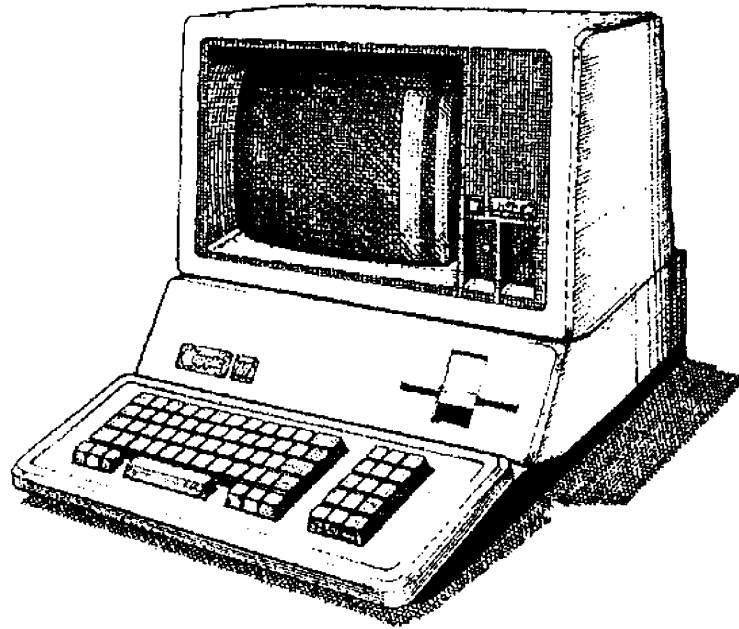
Apple III Computer Information





Apple /// Computer Information

Apple /// Service Reference Manual



THE END

Written by Apple Computer • 1982