

### Apple 2 Technical Manual

### Apple ][ Reference Manual

# Document 030-0004-01 1979

Author: Apple Computer, Inc.

Printed by: Macintosh Picture Printer 0.0.5 1999-01-11

Printed: 2001-07-23 20:15:46

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0000 of 0275

Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

# APPLE JE REFERENCE MANUAL



"A2M 030-0004-01 1-cover1.PICT" 10394 KB 2001-07-23 dpi: 400h x 400v pix: 2127h x 3112v

Author: Apple Computer, Inc. • Document # 030-0004-01 Page 0001 of 0275

Apple 2 Technical Manual • Apple ][ Reference Manual • 1979
NOTICE
Apple Computer Inc. reserves the right to make improvements in the product described in this manual at any time and without notice.
This manual is copyrighted and contains proprietary information. All rights are reserved. This document may not, in whole or part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from Apple Computer Inc.
©1979 by Apple Computer Inc.
10260 Bandley Drive Cupertino, CA 95014 (408) 996-1010
Reorder Apple product number A2L0001A (030-0004-01)
Written by Christopher Espinosa
"Apple" is a trademark of Apple Computer Inc.
"A2M 030-0004-01 1-cover2.PICT" 102 KB 2001-07-23 dpi: 600h x 600v pix: 2982h x 3586v
Author: Apple Computer, Inc. • Document # 030-0004-01 Page 0002 of 0275

A REFERENCE MANUAL FOR THE APPLE II AND THE APPLE II PLUS PERSONAL COMPUTERS

"A2M 030-0004-01 2-title.PICT" 12653 KB 2001-07-23 dpi: 600h x 600v pix: 2783h x 4704v

#### TABLE OF CONTENTS

# CHAPTER 1 APPROACHING YOUR APPLE

- 2 THE POWER SUPPLY
- 3 THE MAIN BOARD
- 4 TALKING TO YOUR APPLE
- 5 THE KEYBOARD
- 6 READING THE KEYBOARD
- 9 THE APPLE VIDEO DISPLAY
- 9 THE VIDEO CONNECTOR
- 10 EURAPPLE (50 HZ) MODIFICATION
- 10 SCREEN FORMAT
- 12 SCREEN MEMORY
- 12 SCREEN PAGES
- 12 SCREEN SWITCHES
- 14 THE TEXT MODE
- 17 THE LOW-RESOLUTION GRAPHICS (LO-RES) MODE
- 19 THE HIGH-RESOLUTION GRAPHICS (HI-RES) MODE
- 20 OTHER INPUT/OUTPUT FEATURES
- 20 THE SPEAKER
- 22 THE CASSETTE INTERFACE
- 23 THE GAME I/O CONNECTOR
- 23 ANNUNCIATOR OUTPUTS
- 24 ONE-BIT INPUTS
- 24 ANALOG INPUTS
- 25 STROBE OUTPUT
- 25 VARIETIES OF APPLES
- 25 AUTOSTART ROM / MONITOR ROM
- 26 REVISION Ø / REVISION 1 BOARD
- 27 POWER SUPPLY CHANGES
- 27 THE APPLE II PLUS

"A2M 030-0004-01 3-toc1.PICT" 341 KB 2001-07-23 dpi: 600h x 600v pix: 3192h x 4985v

## CHAPTER 2 CONVERSATION WITH APPLES

- 30 STANDARD OUTPUT
- 30 THE STOP-LIST FEATURE
- 31 BUT SOFT, WHAT LIGHT THROUGH YONDER WINDOW BREAKS! (OR, THE TEXT WINDOW)
- 32 SEEING IT ALL IN BLACK AND WHITE
- 32 STANDARD INPUT
- 32 RDKEY
- 33 GETLN
- 34 ESCAPE CODES
- 36 THE RESET CYCLE
- 36 AUTOSTART ROM RESET
- 37 AUTOSTART ROM SPECIAL LOCATIONS
- 38 "OLD MONITOR" ROM RESET

## CHAPTER 3 THE SYSTEM MONITOR

- 40 ENTERING THE MONITOR
- 40 ADDRESSES AND DATA
- 41 EXAMINING THE CONTENTS OF MEMORY
- 41 EXAMINING SOME MORE MEMORY
- 43 EXAMINING STILL MORE MEMORY
- 43 CHANGING THE CONTENTS OF A LOCATION
- 44 CHANGING THE CONTENTS OF CONSECUTIVE LOCATIONS
- 44 MOVING A RANGE OF MEMORY
- 46 COMPARING TWO RANGES OF MEMORY
- 46 SAVING A RANGE OF MEMORY ON TAPE
- 47 READING A RANGE FROM TAPE
- 48 CREATING AND RUNNING MACHINE LANGUAGE PROGRAMS
- 49 THE MINI-ASSEMBLER

"A2M 030-0004-01 3-toc2.PICT" 393 KB 2001-07-23 dpi: 600h x 600v pix: 3360h x 5683v

- 51 DEBUGGING PROGRAMS
- 53 EXAMINING AND CHANGING REGISTERS
- 54 MISCELLANEOUS MONITOR COMMANDS
- 55 SPECIAL TRICKS WITH THE MONITOR
- 57 CREATING YOUR OWN COMMANDS
- 59 SUMMARY OF MONITOR COMMANDS
- 61 SOME USEFUL MONITOR SUBROUTINES
- 65 MONITOR SPECIAL LOCATIONS
- 66 MINI-ASSEMBLER INSTRUCTION FORMATS

# CHAPTER 4 MEMORY ORGANIZATION

- 68 RAM STORAGE
- 70 RAM CONFIGURATION BLOCKS
- 72 ROM STORAGE
- 73 I/O LOCATIONS
- 74 ZERO PAGE MEMORY MAPS

"A2M 030-0004-01 3-toc3.PICT" 204 KB 2001-07-23 dpi: 600h x 600v pix: 2887h x 3383v

## CHAPTER 5 INPUT/OUTPUT STRUCTURE

- 78 BUILT-IN I/O
- 79 PERIPHERAL BOARD I/O
- 80 PERIPHERAL CARD I/O SPACE
- 80 PERIPHERAL CARD ROM SPACE
- 81 I/O PROGRAMMING SUGGESTIONS
- 82 PERIPHERAL SLOT SCRATCHPAD RAM
- 83 THE CSW/KSW SWITCHES
- 84 EXPANSION ROM

# CHAPTER 6 HARDWARE CONFIGURATION

- 88 THE MICROPROCESSOR
- 90 SYSTEM TIMING
- 92 POWER SUPPLY
- 94 ROM MEMORY
- 95 RAM MEMORY
- 96 THE VIDEO GENERATOR
- 97 VIDEO OUTPUT JACKS
- 98 BUILT-IN I/O
- 99 "USER 1" JUMPER
- 100 THE GAME I/O CONNECTOR
- 100 THE KEYBOARD
- 102 KEYBOARD CONNECTOR
- 103 CASSETTE INTERFACE JACKS
- 104 POWER CONNECTOR
- 105 SPEAKER
- 105 PERIPHERAL CONNECTORS

"A2M 030-0004-01 3-toc4.PICT" 313 KB 2001-07-23 dpi: 600h x 600v pix: 3349h x 5514v

### 117 APPENDIX A THE 6502 INSTRUCTION SET

APPENDIX B
SPECIAL LOCATIONS

135 APPENDIX C ROM LISTINGS

177 GLOSSARY

185 BIBLIOGRAPHY

"A2M 030-0004-01 3-toc5.PICT" 182 KB 2001-07-23 dpi: 600h x 600v pix: 3394h x 4138v

# INDEX

190	GENERAL INDEX
194	INDEX OF FIGURES
195	INDEX OF PHOTOS
195	INDEX OF TABLES
195	CAST OF CHARACTERS

"A2M 030-0004-01 3-toc6.PICT" 47 KB 2001-07-23 dpi: 600h x 600v pix: 1353h x 1399v

### INTRODUCTION

This is the User Reference Manual for the Apple II and Apple II Plus personal computers. Like the Apple itself, this book is a tool. As with all tools, you should know a little about it before you start to use it.

This book will not teach you how to program. It is a book of facts, not methods. If you have just unpacked your Apple, or you do not know how to program in any of the languages available for it, then before you continue with this book, read one of the other manuals accompanying your Apple. Depending upon which variety of Apple you have purchased, you should have received one of the following:

Apple II BASIC Programming Manual (part number A2L0005)

The Applesoft Tutorial (part number A2L0018)

These are tutorial manuals for versions of the BASIC language available on the Apple. They also include complete instructions on setting up your Apple. The Bibliography at the end of this manual lists other books which may interest you.

There are a few different varieties of Apples, and this manual applies to all of them. It is possible that some of the features noted in this manual will not be available on your particular Apple. In places where this manual mentions features which are not universal to all Apples, it will use a footnote to warn you of these differences.

This manual describes the Apple II computer and its parts and procedures. There are sections on the System Monitor, the input/output devices and their operation, the internal organization of memory and input/output devices, and the actual electronic design of the Apple itself. For information on any other Apple hardware or software product, please refer to the manual accompanying that product.

"A2M 030-0004-01 4-intro.PICT" 270 KB 2001-07-23 dpi: 600h x 600v pix: 3005h x 3132v

Author: Apple Computer, Inc. • Document # 030-0004-01 Page 0010 of 0275

### CHAPTER 1 APPROACHING YOUR APPLE

- 2 THE POWER SUPPLY
- 3 THE MAIN BOARD
- 4 TALKING TO YOUR APPLE
- 5 THE KEYBOARD
- 6 READING THE KEYBOARD
- 9 THE APPLE VIDEO DISPLAY
- 9 THE VIDEO CONNECTOR
- 10 EURAPPLE (50 HZ) MODIFICATION
- 10 SCREEN FORMAT
- 12 SCREEN MEMORY
- 12 SCREEN PAGES
- 12 SCREEN SWITCHES
- 14 THE TEXT MODE
- 17 THE LOW-RESOLUTION GRAPHICS (LO-RES) MODE
- 19 THE HIGH-RESOLUTION GRAPHICS (HI-RES) MODE
- OTHER INPUT/OUTPUT FEATURES
- 20 THE SPEAKER
- 22 THE CASSETTE INTERFACE
- 23 THE GAME I/O CONNECTOR
- 23 ANNUNCIATOR OUTPUTS
- 24 ONE-BIT INPUTS
- 24 ANALOG INPUTS
- 25 STROBE OUTPUT
- 25 VARIETIES OF APPLES
- 25 AUTOSTART ROM / MONITOR ROM
- 26 REVISION Ø / REVISION 1 BOARD
- 27 POWER SUPPLY CHANGES
- 27 THE APPLE II PLUS

"A2M 030-0004-01 5-001.PICT" 13811 KB 2001-07-23 dpi: 600h x 600v pix: 3076h x 4723v

For detailed information on setting up your Apple, refer to Chapter 1 of either the Apple BASIC Programming Manual or The Applesoft Tutorial.

In this manual, all directional instructions will refer to this orientation: with the Apple's typewriter-like keyboard facing you, "front" and "down" are towards the keyboard, "back" and "up" are away. Remove the lid of the Apple by prying up the back edge until it "pops", then pull straight back on the lid and lift it off.

This is what you will see:

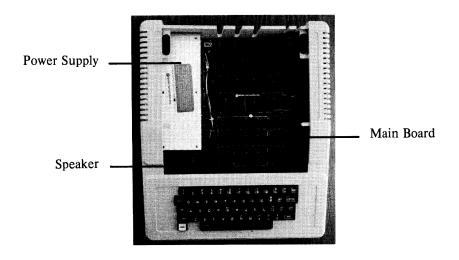


Photo 1. The Apple II.

#### THE POWER SUPPLY

The metal box on the left side of the interior is the Power Supply. It supplies four voltages: +5v, -5.2v, +11.8v, and -12.0v. It is a high-frequency "switching"-type power supply, with many protective features to ensure that there can be no imbalances between the different supplies. The main power cord for the computer plugs directly into the back of the power supply. The power-on switch is also on the power supply itself, to protect you and your fingers from accidentally becoming part of the high-voltage power supply circuit.

2

"A2M 030-0004-01 5-002.PICT" 272 KB 2001-07-23 dpi: 600h x 600v pix: 2968h x 4627v

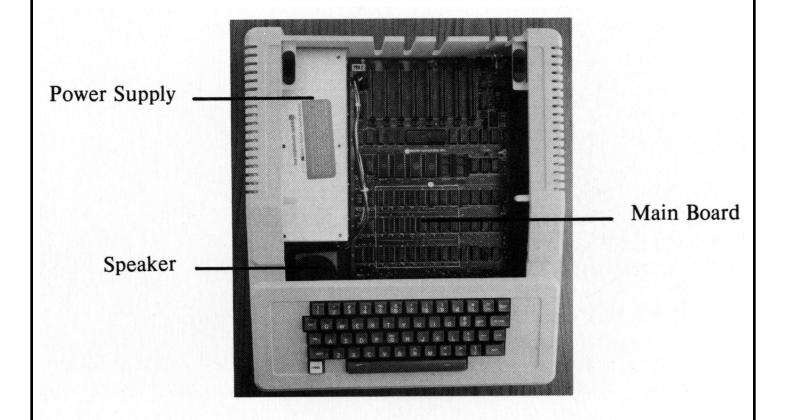
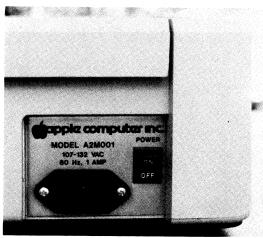


Photo 1. The Apple II.

"A2M 030-0004-01 5-002p.PICT" 9905 KB 2001-07-23 dpi: 1200h x 1200v pix: 4468h x 3699v

Author: Apple Computer, Inc. • Document # 030-0004-01 Page 0013 of 0275





110 volt model

110/220 volt model

Photo 2. The back of the Apple Power Supply.

#### THE MAIN BOARD

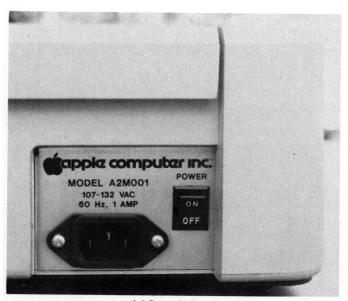
The large green printed circuit board which takes up most of the bottom of the case is the computer itself. There are two slightly different models of the Apple II main board: the original (Revision  $\emptyset$ ) and the Revision 1 board. The slight differences between the two lie in the electronics on the board. These differences are discussed throughout this book. A summary of the differences appears in the section "Varieties of Apples" on page 25.

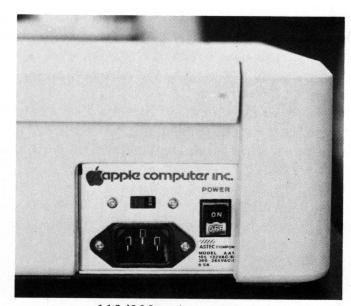
On this board there are about eighty integrated circuits and a handful of other components. In the center of the board, just in front of the eight gold-toothed edge connectors ("slots") at the rear of the board, is an integrated circuit larger than all others. This is the brain of your Apple. It is a Synertek/MOS Technology 6502 microprocessor. In the Apple, it runs at a rate of 1,023,000 machine cycles per second and can do over five hundred thousand addition or subtraction operations in one second. It has an addressing range of 65,536 eight-bit bytes. Its repertory includes 56 instructions with 13 addressing modes. This microprocessor and other versions of it are used in many computers systems, as well as other types of electronic equipment.

Just below the microprocessor are six sockets which may be filled with from one to six slightly smaller integrated circuits. These ICs are the Read-Only Memory (ROM) "chips" for the Apple. They contain programs for the Apple which are available the moment you turn on the power. Many programs are available in ROM, including the Apple System Monitor, the Apple Autostart Monitor, Apple Integer BASIC and Applesoft II BASIC, and the Apple Programmer's Aid #1 utility subroutine package. The number and contents of your Apple's ROMs depend upon which type of Apple you have, and the accessories you have purchased.

Right below the ROMs and the central mounting nut is an area marked by a white square on the board which encloses twenty-four sockets for integrated circuits. Some or all of these may be filled with ICs. These are the main Random Access Memory (RAM) "chips" for your Apple. An Apple can hold 4,096 to 49,152 bytes of RAM memory in these three rows of components.\* Each row can hold eight ICs of either the 4K or 16K variety. A row must hold eight of the same

<sup>\*</sup> You can extend your RAM memory to 64K by purchasing the Apple Language Card, part of the Apple Language System (part number A2B0006).





110 volt model

110/220 volt model

Photo 2. The back of the Apple Power Supply.

"A2M 030-0004-01 5-003p.PICT" 14561 KB 2001-07-23 dpi: 1200h x 1200v pix: 5846h x 3007v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0015 of 0275

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

type of memory components, but the two types can both be used in various combinations on different rows to give nine different memory sizes.\* The RAM memory is used to hold all of the programs and data which you are using at any particular time. The information stored in RAM disappears when the power is turned off.

The other components on the Apple II board have various functions: they control the flow of information from one part of the computer to another, gather data from the outside world, or send information to you by displaying it on a television screen or making a noise on a speaker.

The eight long peripheral slots on the back edge of the Apple's board can each hold a peripheral card to allow you to extend your RAM or ROM memory, or to connect your Apple to a printer or other input/output device. These slots are sometimes called the Apple's "backplane" or "mother board".

#### TALKING TO YOUR APPLE

Your link to your Apple is at your fingertips. Most programs and languages that are used with the Apple expect you to talk to them through the Apple's keyboard. It looks like a normal type-writer keyboard, except for some minor rearrangement and a few special keys. For a quick review on the keyboard, see pages 6 through 12 in the Apple II BASIC Programming Manual or pages 5 through 11 in The Applesoft Tutorial.

Since you're talking with your fingers, you might as well be hearing with your eyes. The Apple will tell you what it is doing by displaying letters, numbers, symbols, and sometimes colored blocks and lines on a black-and-white or color television set.

4

<sup>\*</sup> The Apple II is designed to use both the 16K and the less expensive 4K RAMs. However, due to the greater availability and reduced cost of the 16K chips, Apple now supplies only the 16K RAMs.

#### THE KEYBOARD

The Apple Keyboard

Number of Keys: 52

Coding: Upper Case ASCII

Number of codes: 91

Output: Seven bits, plus strobe

Power requirements: +5v at 120mA

-12v at 50mA

Rollover: 2 key

Special keys: CTRL

ESC RESET REPT ← →

Memory mapped locations: Hex Decimal

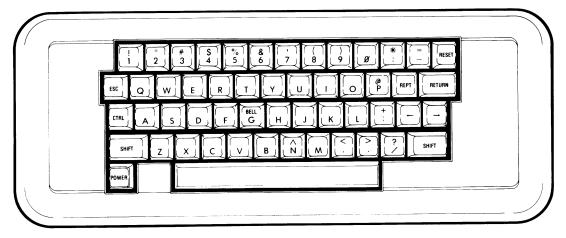
Data \$C000 49152 -16384 Clear \$C010 49168 -16368

The Apple II has a built-in 52-key typewriter-like keyboard which communicates using the American Standard Code for Information Interchange (ASCII)\*. Ninety-one of the 96 upper-case ASCII characters can be generated directly by the keyboard. Table 2 shows the keys on the keyboard and their associated ASCII codes. "Photo" 3 is a diagram of the keyboard.

The keyboard is electrically connected to the main circuit board by a 16-conductor cable with plugs at each end that plug into standard integrated circuit sockets. One end of this cable is connected to the keyboard; the other end plugs into the Apple board's keyboard connector, near the very front edge of the board, under the keyboard itself. The electrical specifications for this connector are given on page 102.

Most languages on the Apple have commands or statements which allow your program to accept input from the keyboard quickly and easily (for example, the INPUT and GET statements in BASIC). However, your programs can also read the keyboard directly.

<sup>\*</sup> All ASCII codes used by the Apple normally have their high bit set. This is the same as standard mark-parity ASCII.



"Photo" 3. The Apple Keyboard.

#### READING THE KEYBOARD

The keyboard sends seven bits of information which together form one character. These seven bits, along with another signal which indicates when a key has been pressed, are available to most programs as the contents of a memory location. Programs can read the current state of the keyboard by reading the contents of this location. When you press a key on the keyboard, the value in this location becomes 128 or greater, and the particular value it assumes is the numeric code for the character which was typed. Table 3 on page 8 shows the ASCII characters and their associated numeric codes. The location will hold this one value until you press another key, or until your program tells the memory location to forget the character it's holding.

Once your program has accepted and understood a keypress, it should tell the keyboard's memory location to "release" the character it is holding and prepare to receive a new one. Your program can do this by referencing another memory location. When you reference this other location, the value contained in the first location will drop below 128. This value will stay low until you press another key. This action is called "clearing the keyboard strobe". Your program can either read or write to the special memory location; the data which are written to or read from that location are irrelevant. It is the mere *reference* to the location which clears the keyboard strobe. Once you have cleared the keyboard strobe, you can still recover the code for the key which was last pressed by adding 128 (hexadecimal \$80) to the value in the keyboard location.

These are the special memory locations used by the keyboard:

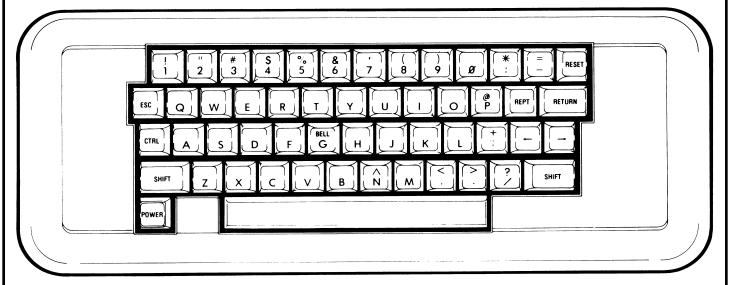
T	able 1:	Keyboard S	Special Locations
Location			Description
Hex	Hex Decimal		
\$CØØØ	49152	-16384	Keyboard Data
\$CØ1Ø	49168	-16368	Clear Keyboard Strobe

The **RESET** key at the upper right-hand corner does not generate an ASCII code, but instead is directly connected to the microprocessor. When this key is pressed, all processing stops. When the key is released, the computer starts a reset cycle. See page 36 for a description of the RESET

6

"A2M 030-0004-01 5-006.PICT" 457 KB 2001-07-23 dpi: 600h x 600v pix: 3013h x 4636v

Apple 2 Technical Manual • Apple ][ Reference Manual • 1979



"Photo" 3. The Apple Keyboard.

"A2M 030-0004-01 5-006p.PICT" 322 KB 2001-07-23 dpi: 1200h x 1200v pix: 5847h x 2624v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0019 of 0275

function.

The CTRL and SHIFT keys generate no codes by themselves, but only alter the codes produced by other keys.

The **REPT** key, if pressed alone, produces a duplicate of the last code that was generated. If you press and hold down the **REPT** key while you are holding down a character key, it will act as if you were pressing that key repeatedly at a rate of 10 presses each second. This repetition will cease when you release either the character key or **REPT**.

The POWER light at the lower left-hand corner is an indicator lamp to show when the power to the Apple is on.

	Table 2: Keys and Their Associated ASCII Codes								
Key	Alone	CTRL	SHIFT	Both	Key	Alone	CTRL	SHIFT	Both
space	\$AØ	\$AØ	\$AØ	\$AØ	RETURN	\$8D	\$8D	\$8D	\$8D
Ø	\$BØ	\$BØ	\$BØ	\$BØ	G	\$C7	\$87	<b>\$</b> C7	\$87
1!	\$B1	\$B1	<b>\$</b> A1	\$A1	Н	\$C8	\$88	\$C8	\$88
2"	\$B2	\$B2	\$A2	\$A2	I	\$C9	\$89	\$C9	\$89
3#	\$B3	\$B3	\$A3	<b>\$A3</b>	J	\$CA	\$8A	\$CA	\$8A
4\$	\$B4	\$B4	\$A4	\$A4	K	\$CB	\$8B	\$CB	\$8B
5%	\$B5	\$B5	\$A5	\$A5	L	\$CC	\$8C	\$CC	\$8C
6&	\$B6	<b>\$B6</b>	\$A6	\$A6	M	\$CD	\$8D	\$DD	\$9D
7'	<b>\$B</b> 7	<b>\$B</b> 7	<b>\$A7</b>	\$A7	N^	\$CE	\$8E	\$DE	\$9E
8(	<b>\$B8</b>	\$B8	<b>\$A8</b>	\$A8	О	\$CF	<b>\$8</b> F	\$CF	\$8F
9)	\$B9	\$B9	\$A9	\$A9	P@	\$DØ	<b>\$9</b> Ø	\$CØ	\$80
:*	\$BA	\$BA	\$AA	\$AA	Q	\$D1	<b>\$91</b>	<b>\$</b> D1	\$91
;+	\$BB	\$BB	\$AB	\$AB	R	\$D2	\$92	\$D2	\$92
,<	\$AC	\$AC	\$BC	\$BC	S	\$D3	\$93	\$D3	\$93
-=	\$AD	\$AD	\$BD	\$BD	T	<b>\$D4</b>	\$94	\$D4	\$94
.>	\$AE	\$AE	\$BE	\$BE	U	\$D5	<b>\$95</b>	<b>\$</b> D5	\$95
/?	\$AF	\$AF	\$BF	\$BF	V	\$D6	\$96	<b>\$</b> D6	\$96
A	\$C1	\$81	\$C1	\$81	W	<b>\$</b> D7	\$97	<b>\$</b> D7	\$97
В	\$C2	\$82	\$C2	\$82	X	<b>\$D8</b>	\$98	\$D8	\$98
C	\$C3	\$83	\$C3	\$83	Y	<b>\$</b> D9	\$99	\$D9	\$99
D	\$C4	\$84	\$C4	\$84	Z	\$DA	\$9A	\$DA	\$9A
E	\$C5	\$85	\$C5	\$85	→	\$88	\$88	\$88	\$88
F	\$C6	\$86	<b>\$</b> C6	\$86	←	\$95	\$95	\$95	\$95
					ESC	\$9B	\$9B	\$9B	\$9B

All codes are given in hexadecimal. To find the decimal equivalents, use Table 3.

		Tal	ble 3:	The AS	The ASCII Character Set				
Dec	imal:	128	144	160	176	192	208	224	240
	Hex:	\$8Ø	<b>\$90</b>	\$AØ	\$BØ	\$CØ	\$DØ	\$EØ	\$FØ
Ø	\$Ø	nul	dle		Ø	@	P		р
1	\$1	soh	dc1	!	1	Α	Q	a	q
2	\$2	stx	dc2	"	2	В	R	b	r
3	\$3	etx	dc3	#	3	C	S	c	s
4	\$4	eot	dc4	\$	4	D	T	d	t
5	\$5	enq	nak	%	5	E	U	е	u
6	\$6	ack	syn	&	6	F	V	f	v
7	<b>\$</b> 7	bel	etb	,	7	G	$\mathbf{W}$	g	w
8	\$8	bs	can	(	8	Н	X	h	х
9	<b>\$</b> 9	ht	em	)	9	I	Y	i	у
10	\$A	lf	sub	*	:	J	Z	j	Z
11	\$B	vt	esc	+	;	K	[	k	{
12	\$C	ff	fs	,	<	L	\	1	
13	\$D	cr	gs	_	_	M	]	m	}
14	\$E	so	rs	•	>	N	^	n	~
15	\$F	si	us	/	?	O	_	0	rub

Groups of two and three lower case letters are abbreviations for standard ASCII control characters.

Not all the characters listed in this table can be generated by the keyboard. Specifically, the characters in the two rightmost columns (the lower case letters), the symbols [ (left square bracket), \ (backslash), \_ (underscore), and the control characters "fs", "us", and "rub", are not available on the Apple keyboard.

The decimal or hexadecimal value for any character in the above table is the sum of the decimal or hexadecimal numbers appearing at the top of the column and the left side of the row in which the character appears.

#### THE APPLE VIDEO DISPLAY

#### The Apple Video Display

Display type: Memory mapped into system RAM

Display modes: Text, Low-Resolution Graphics,

High-Resolution Graphics

Text capacity: 960 characters (24 lines, 40 columns)

Character type:  $5 \times 7$  dot matrix

Character set: Upper case ASCII, 64 characters

Character modes: Normal, Inverse, Flashing

Graphics capacity: 1,920 blocks (Low-Resolution)

in a 40 by 48 array

53,760 dots (High-Resolution) in a 280 by 192 array

Number of colors: 16 (Low-Resolution Graphics)

6 (High-Resolution Graphics)

#### THE VIDEO CONNECTOR

In the right rear corner of the Apple II board, there is a metal connector marked "VIDEO". This connector allows you to attach a cable between the Apple and a closed-circuit video monitor. One end of the connecting cable should have a male RCA phono jack to plug into the Apple, and the other end should have a connector compatible with the particular device you are using. The signal that comes out of this connector on the Apple is similar to an Electronic Industries Association (EIA)-standard, National Television Standards Committee (NTSC)-compatible, positive composite color video signal. The level of this signal can be adjusted from zero to 1 volt peak by the small round potentiometer on the right edge of the board about three inches from the back of the board.

A non-adjustable, 2 volts peak version of the same video signal is available in two other places: on a single wire-wrap pin\* on the left side of the board about two inches from the back of the board, and on one pin of a group of four similar pins also on the left edge near the back of the board. The other three pins in this group are connected to -5 volts, +12 volts, and ground. See page 97 for a full description of this auxiliary video connector.

<sup>\*</sup> This pin is not present in Apple II systems with the Revision Ø board.

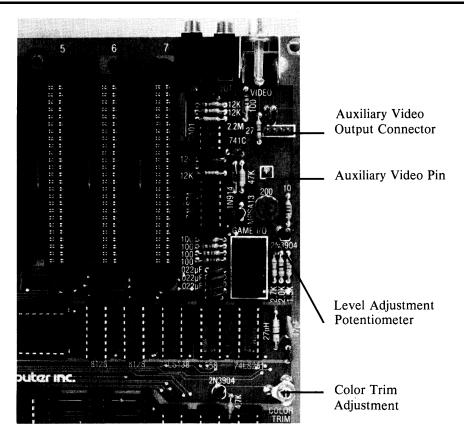


Photo 4. The Video Connectors and Potentiometer.

#### **EURAPPLE (50 HZ) MODIFICATION**

Your Apple can be modified to generate a video signal compatible with the CCIR standard used in many European countries. To make this modification, just cut the two X-shaped pads on the right edge of the board about nine inches from the back of the board, and solder together the three O-shaped pads in the same locations (see photo 5). You can then connect the video connector of your Apple to a European standard closed-circuit black-and-white or color video monitor. If you wish, you can obtain a "Eurocolor" encoder to convert the video signal into a PAL or SECAM standard color television signal suitable for use with any European television receiver. The encoder is a small printed circuit board which plugs into the rightmost peripheral slot (slot 7) in your Apple and connects to the single auxiliary video output pin.

WARNING: This modification will void the warranty on your Apple and requires the installation of a different main crystal. This modification is not for beginners.

#### **SCREEN FORMAT**

Three different kinds of information can be shown on the video display to which your Apple is connected:

10

"A2M 030-0004-01 5-010.PICT" 376 KB 2001-07-23 dpi: 600h x 600v pix: 3031h x 4636v

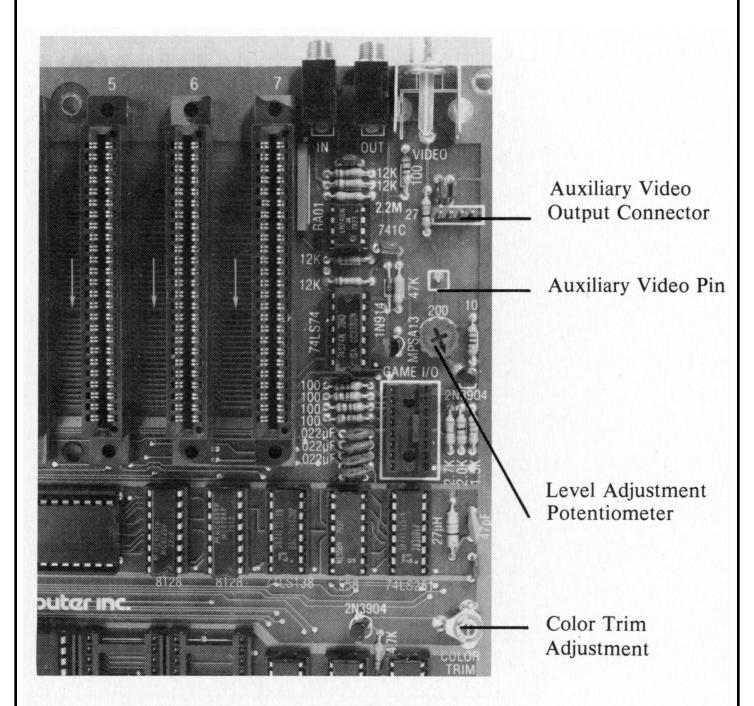


Photo 4. The Video Connectors and Potentiometer.

"A2M 030-0004-01 5-010p.PICT" 18708 KB 2001-07-23 dpi: 1200h x 1200v pix: 4684h x 4649v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0024 of 0275

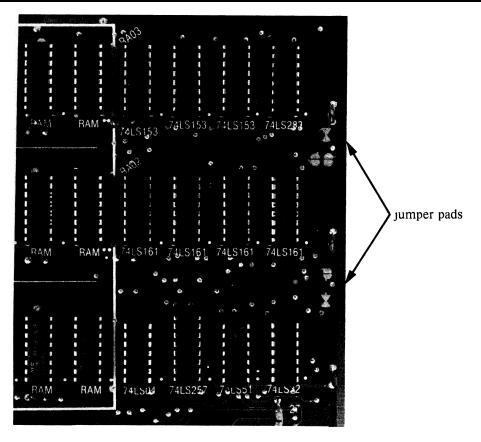


Photo 5. Eurapple (50 hz) Jumper Pads.

- 1) **Text**. The Apple can display 24 lines of numbers, special symbols, and upper-case letters with 40 of these characters on each line. These characters are formed in a dot matrix 7 dots high and 5 dots wide. There is a one-dot wide space on either side of the character and a one-dot high space above each line.
- 2) Low-Resolution Graphics. The Apple can present 1,920 colored squares in an array 40 blocks wide and 48 blocks high. The color of each block can be selected from a set of sixteen different colors. There is no space between blocks, so that any two adjacent blocks of the same color look like a single, larger block.
- 3) **High-Resolution Graphics**. The Apple can also display colored dots on a matrix 280 dots wide and 192 dots high. The dots are the same size as the dots which make up the Text characters. There are six colors available in the High-Resolution Graphics mode: black, white, red, blue, green, and violet.\* Each dot on the screen can be either black, white, or a color, although not all colors are available for every dot.

When the Apple is displaying a particular type of information on the screen, it is said to be in that particular "mode". Thus, if you see words and numbers on the screen, you can reasonably be assured that your Apple is in Text mode. Similarly, if you see a screen full of multicolored blocks, your computer is probably in Low-Resolution Graphics mode. You can also have a four-line "caption" of text at the bottom of either type of graphics screen. These four lines replace

<sup>\*</sup> For Apples with Revision Ø boards, there are four colors: black, white, green, and violet.

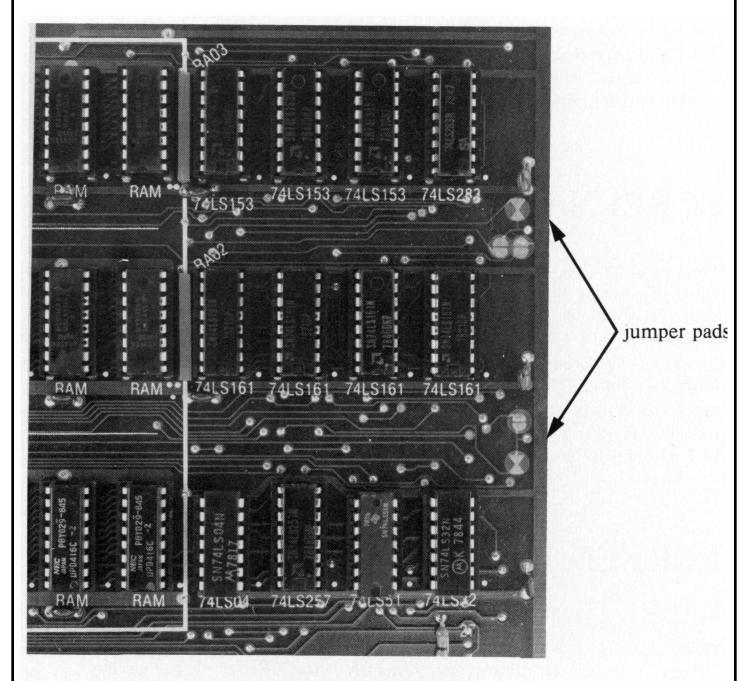


Photo 5. Eurapple (50 hz) Jumper Pads.

"A2M 030-0004-01 5-011p.PICT" 20200 KB 2001-07-23 dpi: 1200h x 1200v pix: 4779h x 4684v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0026 of 0275

the lower 8 rows of blocks in Low-Resolution Graphics, leaving a 40 by 40 array. In High-Resolution Graphics, they replace the bottom 32 rows of dots, leaving a 280 by 160 matrix. You can use these "mixed modes" to display text and graphics simultaneously, but there is no way to display both graphics modes at the same time.

#### **SCREEN MEMORY**

The video display uses information in the system's RAM memory to generate its display. The value of a single memory location controls the appearance of a certain, fixed object on the screen. This object can be a character, two stacked colored blocks, or a line of seven dots. In Text and Low-Resolution Graphics mode, an area of memory containing 1,024 locations is used as the source of the screen information. Text and Low-Resolution Graphics share this memory area. In High-Resolution Graphics mode, a separate, larger area (8,192 locations) is needed because of the greater amount of information which is being displayed. These areas of memory are usually called "pages". The area reserved for High-Resolution Graphics is sometimes called the "picture buffer" because it is commonly used to store a picture or drawing.

#### SCREEN PAGES

There are actually *two* areas from which each mode can draw its information. The first area is called the "primary page" or "Page 1". The second area is called the "secondary page" or "Page 2" and is an area of the same size immediately following the first area. The secondary page is useful for storing pictures or text which you want to be able to display instantly. A program can use the two pages to perform animation by drawing on one page while displaying the other and suddenly flipping pages.

Text and Low-Resolution Graphics share the same memory range for the secondary page, just as they share the same range for the primary page. Both mixed modes which were described above are also available on the secondary page, but there is no way to mix the two pages on the same screen.

Ta	ıble 4: Video	Display	Memory Ra	anges	
Saraan	Page	Begins	at:	Ends at:	
Screen	rage	Hex	Decimal		
Text/Lo-Res	Primary	\$400	1024	\$7FF	2047
	Secondary	\$800	2048	\$BFF	3Ø71
Hi-Res	Primary	\$2000	8192	\$3FFF	16383
	Secondary	\$4000	16384	\$5FFF	24575

#### **SCREEN SWITCHES**

The devices which decide between the various modes, pages, and mixes are called "soft switches". They are switches because they have two positions (for example: on or off, text or graphics) and they are called "soft" because they are controlled by the software of the computer.

12

"A2M 030-0004-01 5-012.PICT" 423 KB 2001-07-23 dpi: 600h x 600v pix: 2994h x 4654v

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

A program can "throw" a switch by referencing the special memory location for that switch. The data which are read from or written to the location are irrelevant; it is the *reference to the address* of the location which throws the switch.

There are eight special memory locations which control the setting of the soft switches for the screen. They are set up in pairs; when you reference one location of the pair you turn its corresponding mode "on" and its companion mode "off". The pairs are:

	7	Гable 5: S	Screen Soft Switches
Location	n:		Description:
Hex	Decimal		Description.
\$CØ5Ø	49232	-16304	Display a GRAPHICS mode.
\$CØ51	49233	-163Ø3	Display TEXT mode.
\$CØ52	49234	-16302	Display all TEXT or GRAPHICS.
\$CØ53	49235	-163Ø1	Mix TEXT and a GRAPHICS mode.*
\$CØ54	49236	-16300	Display the Primary page (Page 1).
\$CØ55	49237	-16299	Display the Secondary page (Page 2).
\$CØ56	49238	-16298	Display LO-RES GRAPHICS mode.*
\$CØ57	49239	-16297	Display HI-RES GRAPHICS mode.*

There are ten distinct combinations of these switches:

	Table 6: Screen Mode Combinations									
Prim	nary Page		Secor	ndary Page	;					
Screen	Switches	S	Screen	Switches	S					
All Text	\$CØ54	\$CØ51	All Text	\$CØ55	\$CØ51					
All Lo-Res	\$CØ54	\$CØ56	All Lo-Res	\$CØ55	\$CØ56					
Graphics	\$CØ52	\$CØ5Ø	Graphics	\$CØ52	\$CØ5Ø					
All Hi-Res	\$CØ54	\$CØ57	All Hi-Res	\$CØ55	\$CØ57					
Graphics	\$CØ52	\$CØ5Ø	Graphics	\$CØ52	\$CØ5Ø					
Mixed Text	\$CØ54	\$CØ56	Mixed Text	\$CØ55	\$CØ56					
and Lo-Res	\$CØ53	\$CØ5Ø	and Lo-Res	\$CØ53	\$CØ5Ø					
Mixed Text	\$CØ54	\$CØ57	Mixed Text	\$CØ55	\$CØ57					
and Hi-Res	\$CØ53	\$CØ5Ø	and Hi-Res	\$CØ53	\$CØ5Ø					

(Those of you who are learned in the ways of binary will immediately cry out, "Where's the other six?!", knowing full well that with 4 two-way switches there are indeed sixteen possible combinations. The answer to the mystery of the six missing modes lies in the TEXT/GRAPHICS switch. When the computer is in Text mode, it can also be in one of six combinations of the Lo-Res/Hi-Res graphics mode, "mix" mode, or page selection. But since the Apple is displaying text, these different graphics modes are invisible.)

To set the Apple into one of these modes, a program needs only to refer to the addresses of the memory locations which correspond to the switches that set that mode. Machine language programs should use the hexadecimal addresses given above; BASIC programs should PEEK or POKE their decimal equivalents (given in Table 5, "Screen Soft Switches", above). The switches may be thrown in any order; however, when switching into one of the Graphics modes, it is helpful to throw the TEXT/GRAPHICS switch last. All the other changes in mode will then take place invisibly behind the text, so that when the Graphics mode is set, the finished graphics

<sup>\*</sup> These modes are only visible if the "Display GRAPHICS" switch is "on".

screen appears all at once.

#### THE TEXT MODE

In the Text mode, the Apple can display 24 lines of characters with up to 40 characters on each line. Each character on the screen represents the contents of one memory location from the memory range of the page being displayed. The character set includes the 26 upper-case letters, the 10 digits, and 28 special characters for a total of 64 characters. The characters are formed in a dot matrix 5 dots wide and 7 dots high. There is a one-dot wide space on both sides of each character to separate adjacent characters and a one-dot high space above each line of characters to separate adjacent lines. The characters are normally formed with white dots on a dark background; however, each character on the screen can also be displayed using dark dots on a white background or alternating between the two to produce a flashing character. When the Video Display is in Text mode, the video circuitry in the Apple turns off the color burst signal to the television monitor, giving you a clearer black-and-white display.\*

The area of memory which is used for the primary text page starts at location number 1024 and extends to location number 2047. The secondary screen begins at location number 2048 and extends up to location 3071. In machine language, the primary page is from hexadecimal address \$400 to address \$7FF; the secondary page is from \$800 to \$BFF. Each of these pages is 1,024 bytes long. Those of you intrepid enough to do the multiplication will realize that there are only 960 characters displayed on the screen. The remaining 64 bytes in each page which are not displayed on the screen are used as temporary storage locations by programs stored in PROM on Apple Intelligent Interface® peripheral boards (see page 82).

Photo 6 shows the sixty-four characters available on the Apple's screen.

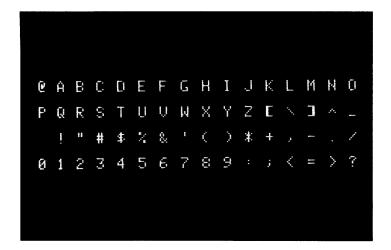


Photo 6. The Apple Character Set.

Table 7 gives the decimal and hexadecimal codes for the 64 characters in normal, inverse, and flashing display modes.

<sup>\*</sup> This feature is not present on the Revision Ø board.



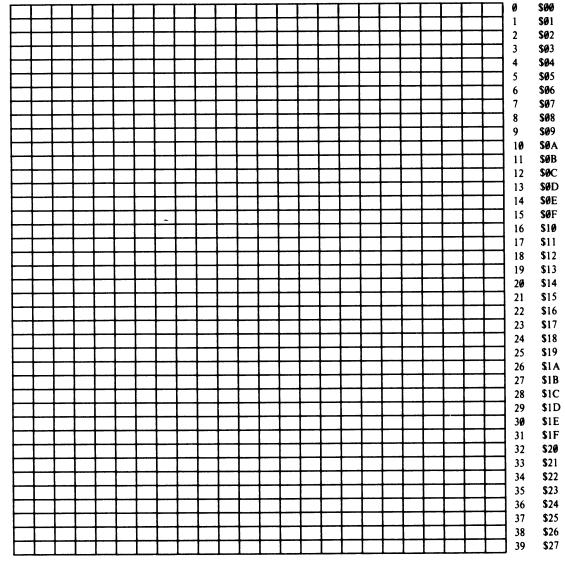
Photo 6. The Apple Character Set.

"A2M 030-0004-01 5-014p.PICT" 10336 KB 2001-07-23 dpi: 1200h x 1200v pix: 3808h x 2834v

Author: Apple Computer, Inc. • Document # 030-0004-01 Page 0030 of 0275

				Table	le 7:		ASCII	Scre	en C	hara	Screen Characters					
						i						Normal	mal			
		Inverse	rse			Flashing	guit		(Control)	trol)					(Lowercase)	rcase)
Decimal	s	16	32	48	64	98	96	112	128	144	160	176	192	208	224	240
Hex	800	810	\$20	830	840	850	860	870	880	890	\$A0	\$B0	\$CØ	\$D0	\$E0	\$F0
08 0	@	Ь		0	(9)	Ь		0	0	Ь		0	(8)	Ь		0
1 \$1	<	0		_	<b>V</b>	0			<b>∀</b>	0		_	∢	0		_
2 \$2	В	~	=	2	В	~	=	7	В	~	=	7	В	R	=	7
3 \$3	C	S	#	3	C	S	#	3	၁	S	#	3	C	S	#	3
4 \$4	D	L	s	4	D	Н	<b>∽</b>	4	D	Г	S	4	D	П	<del>⇔</del>	4
5 \$ \$	Щ	Ω	%	S	Э	ņ	%	S	Э	n	%	2	ш	Ω	%	2
9\$ 9	ш	>	ઝ	9	ц	>	ઝ	9	ц	>	ઝ	9	щ	>	ઋ	9
7.87	Ð	≽		7	Ŋ	≱		7	Ŋ	≽		7	Ö	≽	•	7
8 \$ 8	Н	×	$\smile$	∞	Н	×	$\smile$	∞	Η	×	$\smile$	<b>∞</b>	Η	×	<u> </u>	<b>∞</b>
6\$ 6	ı	>	_	6	_	<b>X</b>	_	6	_	X	_	6	_	<b>X</b>		6
10 SA	,	7	*		_	7	*		_	7	*		'n	7	*	
11 \$B	×		+		×	_	+		×	_	+	. •	×	_	+	
12 SC	Γ	_	•	V	Γ	_	•	V	_	_	•	٧	Γ	_	•	V 1
13 \$D	Σ	_	ı	11	Σ	_	1	II	Σ	_	1	11	Σ	_	I	11
14 SE	Z	•	٠	٨	Z	•	•	٨	Z	•		٨	Z	•	•	٨
15 SF	0	i	_	ن	0	I	/	?	0	ł	_	ç.	0	I	\	6

Figure 1. Map of the Text Screen



16

"A2M 030-0004-01 5-016.PICT" 417 KB 2001-07-23 dpi: 600h x 600v pix: 3039h x 4257v

Figure 1 is a map of the Apple's display in Text mode, with the memory location addresses for each character position on the screen.

### THE LOW-RESOLUTION GRAPHICS (LO-RES) MODE

In the Low-Resolution Graphics mode, the Apple presents the contents of the same 1,024 locations of memory as is in the Text mode, but in a different format. In this mode, each byte of memory is displayed not as an ASCII character, but as two colored blocks, stacked one atop the other. The screen can show an array of blocks 40 wide and 48 high. Each block can be any of sixteen colors. On a black-and-white television set, the colors appear as patterns of grey and white dots.

Since each byte in the page of memory for Low-Resolution Graphics represents two blocks on the screen, stacked vertically, each byte is divided into two equal sections, called (appropriately enough) "nybbles". Each nybble can hold a value from zero to 15. The value which is in the lower nybble of the byte determines the color for the upper block of that byte on the screen, and the value which is in the upper nybble determines the color for the lower block on the screen. The colors are numbered zero to 15, thus:

	Table	8: Low-Resolu	tion Graphi	cs Colo	ors
Decimal	Hex	Color	Decimal	Hex	Color
Ø	\$Ø	Black	8	\$8	Brown
1	<b>\$</b> 1	Magenta	9	<b>\$9</b>	Orange
2	\$2	Dark Blue	10	\$A	Grey 2
3	\$3	Purple	11	<b>\$B</b>	Pink
4	\$4	Dark Green	12	\$C	Light Green
5	\$5	Grey 1	13	\$D	Yellow
6	\$6	Medium Blue	14	\$E	Aquamarine
7	\$7	Light Blue	15	\$F	White

(Colors may vary from television to television, particularly on those without hue controls. You can adjust the tint of the colors by adjusting the COLOR TRIM control on the right edge of the Apple board.)

So, a byte containing the hexadecimal value \$D8 would appear on the screen as a brown block on top of a yellow block. Using decimal arithmetic, the color of the lower block is determined by the quotient of the value of the byte divided by 16; the color of the upper block is determined by the remainder.

Figure 2 is a map of the Apple's display in Low-Resolution Graphics mode, with the memory location addresses for each block on the screen.

Since the Low-Resolution Graphics screen displays the same area in memory as is used for the Text screen, interesting things happen if you switch between the Text and Low-Resolution Graphics modes. For example, if the screen is in the Low-Resolution Graphics mode and is full of colored blocks, and then the TEXT/GRAPHICS screen switch is thrown to the Text mode, the screen will be filled with seemingly random text characters, sometimes inverse or flashing. Similarly, a screen full of text when viewed in Low-Resolution Graphics mode appears as long horizontal grey, pink, green or yellow bars separated by randomly colored blocks.

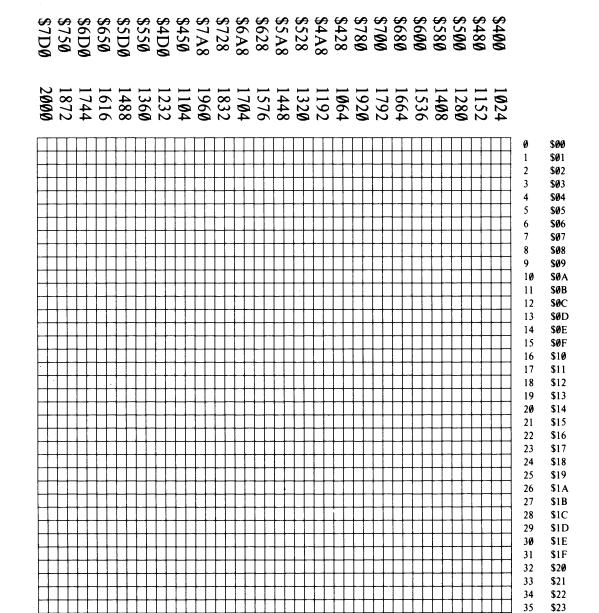


Figure 2. Map of the Low-Resolution Graphics Mode

\$24 \$25

\$26

37

### THE HIGH-RESOLUTION GRAPHICS (HI-RES) MODE

The Apple has a second type of graphic display, called High-Resolution Graphics (or sometimes "Hi-res"). When your Apple is in the High-Resolution Graphics mode, it can display 53,760 dots in a matrix 280 dots wide and 192 dots high. The screen can display black, white, violet, green, red, and blue dots, although there are some limitations concerning the color of individual dots.

The High-Resolution Graphics mode takes its data from an 8,192-byte area of memory, usually called a "picture buffer". There are two separate picture buffers: one for the primary page and one for the secondary page. Both of these buffers are independent of and separate from the memory areas used for Text and Low-Resolution Graphics. The primary page picture buffer for the High-Resolution Graphics mode begins at memory location number 8192 and extends up to location number 16383; the secondary page picture buffer follows on the heels of the first at memory location number 16384, extending up to location number 24575. For those of you with sixteen fingers, the primary page resides from \$2000 to \$3FFF and the secondary page follows in succession at \$4000 to \$5FFF. If your Apple is equipped with 16K (16,384 bytes) or less of memory, then the secondary page is inaccessible to you; if its memory size is less than 16K, then the entire High-Resolution Graphics mode is unavailable to you.

Each dot on the screen represents one bit from the picture buffer. Seven of the eight bits in each byte are displayed on the screen, with the remaining bit used to select the colors of the dots in that byte. Forty bytes are displayed on each line of the screen. The least significant bit (first bit) of the first byte in the line is displayed on the left edge of the screen, followed by the second bit, then the third, etc. The most significant (eighth) bit is not displayed. Then follows the first bit of the next byte, and so on. A total of 280 dots are displayed on each of the 192 lines of the screen.

On a black-and-white monitor or TV set, the dots whose corresponding bits are "on" (or equal to 1) appear white; the dots whose corresponding bits are "off" or (equal to  $\emptyset$ ) appear black. On a color monitor or TV, it is not so simple. If a bit is "off", its corresponding dot will always be black. If a bit is "on", however, its color will depend upon the *position* of that dot on the screen. If the dot is in the leftmost column on the screen, called "column  $\emptyset$ ", or in any even-numbered column, then it will appear violet. If the dot is in the rightmost column (column 279) or any odd-numbered column, then it will appear green. If two dots are placed side-by-side, they will both appear white. If the undisplayed bit of a byte is turned on, then the colors blue and red are substituted for violet and green, respectively.\* Thus, there are six colors available in the High-Resolution Graphics mode, subject to the following limitations:

- 1) Dots in even columns must be black, violet, or blue.
- 2) Dots in odd columns must be black, green, or red.
- 3) Each byte must be either a violet/green byte or a blue/red byte. It is not possible to mix green and blue, green and red, violet and blue, or violet and red in the same byte.

<sup>\*</sup> On Revision Ø Apple boards, the colors red and blue are unavailable and the setting of the eighth bit is irrelevant.

#### Apple 2 Technical Manual • Apple | Reference Manual • 1979

- 4) Two colored dots side by side always appear white, even if they are in different bytes.
- 5) On European-modified Apples, these rules apply but the colors generated in the High-Resolution Graphics mode may differ.

Figure 3 shows the Apple's display screen in High-Resolution Graphics mode with the memory addresses of each line on the screen.

# OTHER INPUT/OUTPUT FEATURES

#### Apple Input/Output Features

Inputs: Cassette Input

Three One-bit Digital Inputs

Four Analog Inputs

Outputs: Cassette Output

Built-In Speaker

Four "Annunciator" Outputs

Utility Strobe Output

# THE SPEAKER

Inside the Apple's case, on the left side under the keyboard, is a small 8 ohm speaker. It is connected to the internal electronics of the Apple so that a program can cause it to make various sounds.

The speaker is controlled by a soft switch. The switch can put the paper cone of the speaker in two positions: "in" and "out". This soft switch is not like the soft switches controlling the various video modes, but is instead a *toggle* switch. Each time a program references the memory address associated with the speaker switch, the speaker will change state: change from "in" to "out" or vice-versa. Each time the state is changed, the speaker produces a tiny "click". By referencing the address of the speaker switch frequently and continuously, a program can generate a steady tone from the speaker.

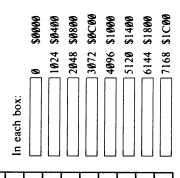
The soft switch for the speaker is associated with memory location number 49200. Any reference to this address (or the equivalent addresses -16336 or hexadecimal \$C030) will cause the speaker to emit a click.

A program can "reference" the address of the special location for the speaker by performing a "read" or "write" operation to that address. The data which are read or written are irrelevant, as it is the address which throws the switch. Note that a "write" operation on the Apple's 6502 microprocessor actually performs a "read" before the "write", so that if you use a "write" operation to flip any soft switch, you will actually throw that switch twice. For toggle-type soft switches, such as the speaker switch, this means that a "write" operation to the special location

20

"A2M 030-0004-01 5-020.PICT" 364 KB 2001-07-23 dpi: 600h x 600v pix: 3057h x 4681v





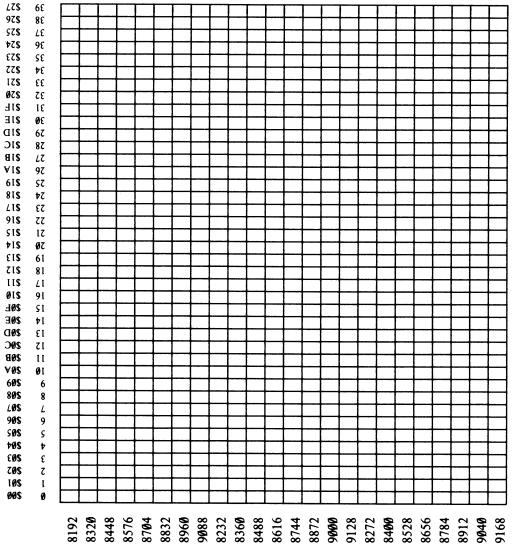


Figure 3. Map of the High-Resolution Graphics Screen

To obtain the address for any byte, add the addresses for that byte's box row, box column, and position in box.

21

\$2000 \$2100 \$2100 \$2100 \$2180 \$2280 \$2380 \$2380 \$2380 \$2328 \$2328 \$2328 \$2328 \$2320 \$2320 \$2320 \$2320 \$2320 \$2320

"A2M 030-0004-01 5-021.PICT" 461 KB 2001-07-23 dpi: 600h x 600v pix: 2985h x 4690v

controlling the switch will leave the switch in the same state it was in before the operation was performed.

#### THE CASSETTE INTERFACE

On the back edge of the Apple's main board, on the right side next to the VIDEO connector, are two small black packages labelled "IN" and "OUT". These are miniature phone jacks into which you can plug a cable which has a pair of miniature phono plugs on each end. The other end of this cable can be connected to a standard cassette tape recorder so that your Apple can save information on audio cassette tape and read it back again.

The connector marked "OUT" is wired to yet another soft switch on the Apple board. This is another toggle switch, like the speaker switch (see above). The soft switch for the cassette output plug can be toggled by referencing memory location number 49184 (or the equivalent -16352 or hexadecimal \$CØ2Ø). Referencing this location will make the voltage on the OUT connector swing from zero to 25 millivolts (one fortieth of a volt), or return from 25 millivolts back to zero. If the other end of the cable is plugged into the MICROPHONE input of a cassette tape recorder which is recording onto a tape, this will produce a tiny "click" on the recording. By referencing the memory location associated with the cassette output soft switch repeatedly and frequently, a program can produce a tone on the recording. By varying the pitch and duration of this tone, information may be encoded on a tape and saved for later use. Such a program to encode data on a tape is included in the System Monitor and is described on page 46.

Be forewarned that if you attempt to flip the soft switch for the cassette output by writing to its special location, you will actually generate *two* "clicks" on the recording. The reason for this is mentioned in the description of the speaker (above). You should only use "read" operations when toggling the cassette output soft switch.

The other connector, marked "IN", can be used to "listen" to a cassette tape recording. Its main purpose is to provide a means of listening to tones on the tape, decoding them into data, and storing them in memory. Thus, a program or data set which was stored on cassette tape may be read back in and used again.

The input circuit takes a 1 volt (peak-to-peak) signal from the cassette recorder's EARPHONE jack and converts it into a string of ones and zeroes. Each time the signal applied to the input circuit swings from positive to negative, or vice-versa, the input circuit changes state: if it was sending ones, it will start sending zeroes, and vice versa. A program can inspect the state of the cassette input circuit by looking at memory location number 49248 or the equivalents -16288 or hexadecimal \$C060. If the value which is read from this location is greater than or equal to 128, then the state is a "one"; if the value in the memory location is less than 128, then the state is a "zero". Although BASIC programs can read the state of the cassette input circuit, the speed of a BASIC program is usually much too slow to be able to make any sense out of what it reads. There is, however, a program in the System Monitor which will read the tones on a cassette tape and decode them. This is described on page 47.

# THE GAME I/O CONNECTOR

The purpose of the Game I/O connector is to allow you to connect special input and output devices to heighten the effect of programs in general, and specifically, game programs. This connector allows you to connect three one-bit inputs, four one-bit outputs, a data strobe, and four analog inputs to the Apple, all of which can be controlled by your programs. Supplied with your Apple is a pair of Game Controllers which are connected to cables which plug into the Game I/O connector. The two rotary dials on the Controllers are connected to two analog inputs on the Connector; the two pushbuttons are connected to two of the one-bit inputs.

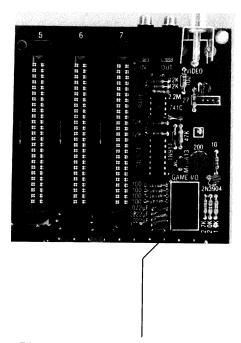


Photo 7. The Game I/O Connector.

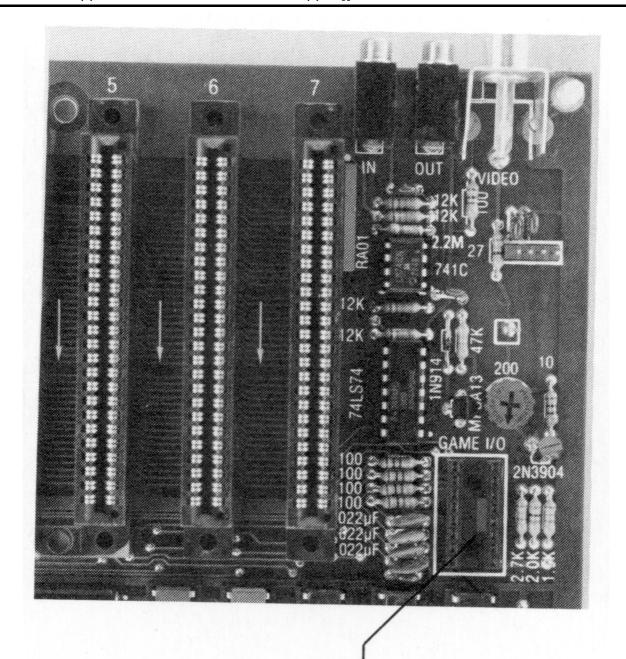
# ANNUNCIATOR OUTPUTS

The four one-bit outputs are called "annunciators". Each annunciator output can be used as an input to some other electronic device, or the annunciator outputs can be connected to circuits to drive lamps, relays, speakers, etc.

Each annunciator is controlled by a soft switch. The addresses of the soft switches for the annunciators are arranged into four pairs, one pair for each annunciator. If you reference the first address in a pair, you turn the output of its corresponding annunciator "off"; if you reference the second address in the pair, you turn the annunciator's output "on". When an annunciator is

23

"A2M 030-0004-01 5-023.PICT" 311 KB 2001-07-23 dpi: 600h x 600v pix: 3013h x 4663v



# Photo 7. The Game I/O Connector.

"A2M 030-0004-01 5-023p.PICT" 6910 KB 2001-07-23 dpi: 1200h x 1200v pix: 2503h x 3735v

"off", the voltage on its pin on the Game I/O Connector is near 0 volts; when an annunciator is "on", the voltage is near 5 volts. There are no inherent means to determine the current setting of an annunciator bit. The annunciator soft switches are:

Table 9: Annunciator Special Locations						
A	State	Address	S:			
Ann.	State	Decimal		Hex		
Ø	off	49240	-16296	\$CØ58		
	on	49241	-16295	\$CØ59		
1	off	49242	-16294	\$CØ5A		
	on	49243	-16293	\$CØ5B		
2	off	49244	-16292	\$CØ5C		
	on	49245	-16291	\$CØ5D		
3	off	49246	-16290	\$CØ5E		
	on	49247	-16289	\$CØ5F		

#### **ONE-BIT INPUTS**

The three one-bit inputs can each be connected to either another electronic device or to a push-button. You can read the state of any of the one-bit inputs from a machine language or BASIC program in the same manner as you read the Cassette Input, above. The locations for the three one-bit inputs have the addresses 49249 through 49251 (-16287 through -16285 or hexadecimal \$CØ61 through \$CØ63).

# **ANALOG INPUTS**

The four analog inputs can be connected to 150K Ohm variable resistors or potentiometers. The variable resistance between each input and the +5 volt supply is used in a one-shot timing circuit. As the resistance on an input varies, the timing characteristics of its corresponding timing circuit change accordingly. Machine language programs can sense the changes in the timing loops and obtain a numerical value corresponding to the position of the potentiometer.

Before a program can start to read the setting of a potentiometer, it must first reset the timing circuits. Location number 49264 (-16272 or hexadecimal \$C070) does just this. When you reset the timing circuits, the values contained in the four locations 49252 through 49255 (-16284 through -16281 or \$C064 through \$C067) become greater than 128 (their high bits are set). Within 3.060 milliseconds, the values contained in these four locations should drop below 128. The exact time it takes for each location to drop in value is directly proportional to the setting of the game paddle associated with that location. If the potentiometers connected to the analog inputs have a greater resistance than 150K Ohms, or there are no potentiometers connected, then the values in the game controller locations may never drop to zero.

# **STROBE OUTPUT**

There is an additional output, called  $\overline{C040}$  STROBE, which is normally +5 volts but will drop to zero volts for a duration of one-half microsecond under the control of a machine language or BASIC program. You can trigger this "strobe" by referring to location number 49216 (-16320 or \$C04F). Be aware that if you perform a "write" operation to this location, you will trigger the strobe *twice* (see a description of this phenomenon in the section on the Speaker).

Table 10: Input/Output Special Locations						
Function:	Address:	Read/Write				
Speaker	49200	-16336	\$CØ3Ø	R		
Cassette Out	49184	-16352	\$CØ2Ø	R		
Cassette In	49256	-16288	\$CØ6Ø	R		
Annunciators*	49240	-16296	\$CØ58	R/W		
	through	through	through			
	49247	-16289	\$CØ5F			
Flag inputs	49249	-16287	\$CØ61	R		
	49250	-16286	\$CØ62	R		
	49251	-16285	\$CØ63	R		
Analog Inputs	49252	-16284	\$CØ64	R		
	49253	-16283	\$CØ65			
	49254	-16282	\$CØ66			
	49255	-16281	\$CØ67			
Analog Clear	49264	-16272	\$CØ7Ø	R/W		
Utility Strobe	49216	-16320	\$CØ4Ø	R		

# **VARIETIES OF APPLES**

There are a few variations on the basic Apple II computer. Some of the variations are revisions or modifications of the computer itself; others are changes to its operating software. These are the basic variations:

# **AUTOSTART ROM / MONITOR ROM**

All Apple II Plus Systems include the Autostart Monitor ROM. All other Apple systems do not contain the Autostart ROM, but instead have the Apple System Monitor ROM. This version of the ROM lacks some of the features present in the Autostart ROM, but also has some features which are not present in that ROM. The main differences in the two ROMs are listed on the following pages.

<sup>\*</sup> See the previous table.

#### Apple 2 Technical Manual • Apple | Reference Manual • 1979

- Editing Controls. The ESC-I, J, K, and M sequences, which move the cursor up, left, right, and down, respectively, are not available in the Old Monitor ROM.
- Stop-List. The Stop-List feature (invoked by a CTRL S), which allows you to introduce a pause into the output of most BASIC or machine language programs or listings, is not available in the Old Monitor ROM.
- The RESET cycle. When you first turn on your Apple or press RESET, the Old Monitor ROM will send you directly into the Apple System Monitor, instead of initiating a warm or cold start as described in "The RESET Cycle" on page 36.

The Old Monitor ROM does, however, support the STEP and TRACE debugging features of the System Monitor, described on page 51. The Autostart ROM does not recognize these Monitor commands.

# REVISION Ø / REVISION 1 BOARD

The Revision Ø Apple II board lacks a few features found on the current Revision 1 version of the Apple II main board. To determine which version of the main board is in your Apple, open the case and look at the upper right-hand corner of the board. Compare what you see to Photo 4 on page 10. If your Apple does not have the single metal video connector pin between the four-pin video connector and the video adjustment potentiometer, then you have a Revision Ø Apple.

The differences between the Revision Ø and Revision 1 Apples are summarized below.

- Color Killer. When the Apple's Video Display is in Text mode, the Revision Ø Apple board leaves the color burst signal active on the video output circuit. This causes text characters to appear tinted or with colored fringes.
- Power-on RESET. Revision Ø Apple boards have no circuit to automatically initiate a RESET cycle when you turn the power on. Instead, you must press RESET once to start using your Apple.

Also, when you turn on the power to an Apple with a Revision Ø board, the keyboard will become active, as if you had typed a random character. When the Apple starts looking for input, it will accept this random character as if you had typed it. In order to erase this character, you should press CTRL X after you RESET your Apple when you turn on its power.

- Colors in High-Resolution Graphics. Apples with Revision Ø boards can generate only four colors in the High-Resolution Graphics mode: black, white, violet, and green. The high bit of each byte displayed on the Hi-Res screen (see page 19) is ignored.
- 24K Memory Map problem. Systems with a Revision Ø Apple II board which contain 20K or 24K bytes of RAM memory appear to BASIC to have more memory than they actually do. See "Memory Organization", page 72, for a description of this problem.
- 50 Hz Apples. The Revision Ø Apple II board does not have the pads and jumpers which you can cut and solder to convert the VIDEO OUT signal of your Apple to conform to the European PAL/SECAM television standard. It also lacks the third VIDEO connector, the single metal pin in front of the four-pin video connector.

26

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

- Speaker and Cassette Interference. On Apples with Revision Ø boards, any sound generated by the internal speaker will also appear as a signal on the Cassette Interface's OUT connector. If you leave the tape recorder in RECORD mode, then any sound generated by the internal speaker will also appear on the tape recording.
- Cassette Input. The input circuit for the Cassette Interface has been modified so that it will respond with more accuracy to a weaker input signal.

#### **POWER SUPPLY CHANGES**

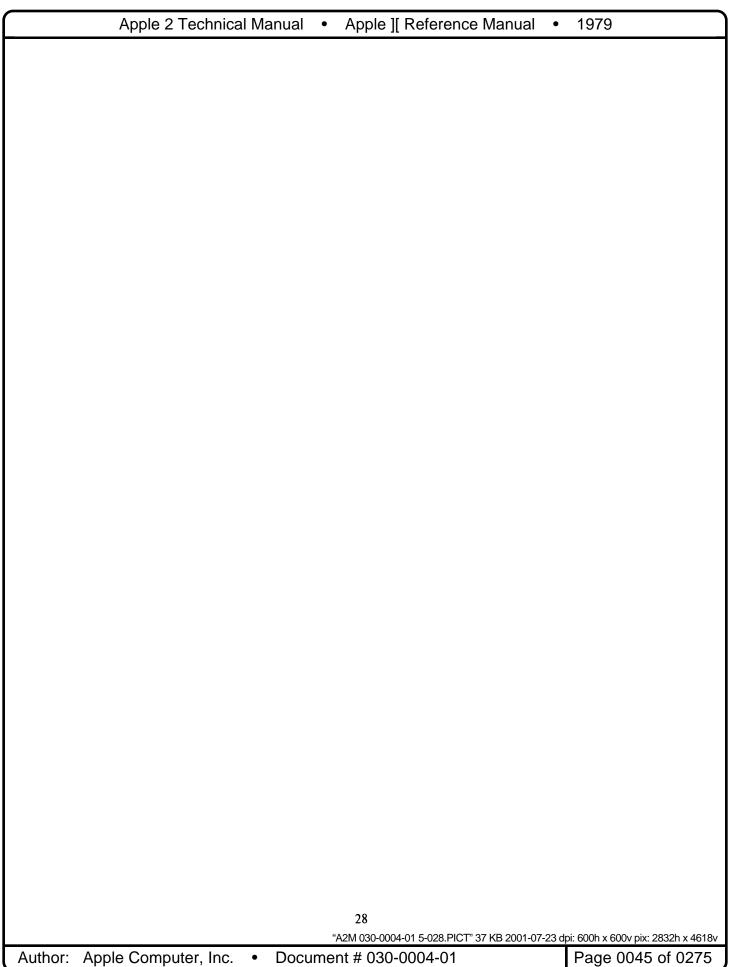
In addition, some Apples have a version of the Apple Power Supply which will accept only a 110 volt power line input. These are not equipped with the voltage selector switch on the back of the supply.

#### THE APPLE II PLUS

The **Apple II Plus** is a standard Apple II computer with a Revision 1 board, an Autostart Monitor ROM, and the Applesoft II BASIC language in ROM in lieu of Apple Integer BASIC. European models of the Apple II Plus are equipped with a 110/220 volt power supply. The Apple Mini-Assembler, the Floating-Point Package, and the SWEET-16 interpreter, stored in the Integer BASIC ROMs, are not available on the Apple II Plus.

27

"A2M 030-0004-01 5-027.PICT" 216 KB 2001-07-23 dpi: 600h x 600v pix: 2995h x 4681v



# CHAPTER 2 CONVERSATION WITH APPLES

- 30 STANDARD OUTPUT
- 30 THE STOP-LIST FEATURE
- 31 BUT SOFT, WHAT LIGHT THROUGH YONDER WINDOW BREAKS! (OR, THE TEXT WINDOW)
- 32 SEEING IT ALL IN BLACK AND WHITE
- 32 STANDARD INPUT
- 32 RDKEY
- 33 GETLN
- 34 ESCAPE CODES
- 36 THE RESET CYCLE
- 36 AUTOSTART ROM RESET
- 37 AUTOSTART ROM SPECIAL LOCATIONS
- 38 "OLD MONITOR" ROM RESET

29

"A2M 030-0004-01 5-029.PICT" 13981 KB 2001-07-23 dpi: 600h x 600v pix: 3067h x 4732v

#### Apple 2 Technical Manual • Apple | Reference Manual • 1979

Almost every program and language on the Apple needs some sort of input from the keyboard, and some way to print information on the screen. There is a set of subroutines stored in the Apple's ROM memory which handle most of the standard input and output from all programs and languages on the Apple.

The subroutines in the Apple's ROM which perform these input and output functions are called by various names. These names were given to the subroutines by their authors when they were written. The Apple itself does not recognize or remember the names of its own machine language subroutines, but it's convenient for us to call these subroutines by their given names.

# STANDARD OUTPUT

The standard output subroutine is called COUT. COUT will display upper-case letters, numbers, and symbols on the screen in either Normal or Inverse mode. It will ignore control characters except RETURN, the bell character, the line feed character, and the backspace character.

The COUT subroutine maintains its own invisible "output cursor" (the position at which the next character is to be placed). Each time COUT is called, it places one character on the screen at the current cursor position, replacing whatever character was there, and moves the cursor one space to the right. If the cursor is bumped off the right edge of the screen, then COUT shifts the cursor down to the first position on the next line. If the cursor passes the bottom line of the screen, the screen "scrolls" up one line and the cursor is set to the first position on the newly blank bottom line.

When a RETURN character is sent to COUT, it moves the cursor to the first position of the next line. If the cursor falls off the bottom of the screen, the screen scrolls as described above.

# THE STOP-LIST FEATURE

When any program or language sends a RETURN code to COUT, COUT will take a quick peek at the keyboard. If you have typed a CTRLS since the last time COUT looked at the keyboard, then it will stop and wait for you to press another key. This is called the *Stop-List* feature.\*\* When you press another key, COUT will then output the RETURN code and proceed with normal output. The code of the key which you press to end the Stop-List mode is ignored unless it is a CTRLC. If it is, then COUT passes this character code back to the program or language which is sending output. This allows you to terminate a BASIC program or listing by typing CTRLC while you are in Stop-List mode.

A line feed character causes COUT to move its mythical output cursor down one line without any horizontal motion at all. As always, moving beyond the bottom of the screen causes the screen to scroll and the cursor remains at its same position on a fresh bottom line.

A backspace character moves the imaginary cursor one space to the left. If the cursor is bumped off the left edge, it is reset to the rightmost position on the previous line. If there is no previous line (if the cursor was at the top of the screen), the screen does *not* scroll downwards, but instead

<sup>\*</sup> From latin cursus, "runner"

<sup>\*\*</sup> The Stop-list feature is not present on Apples without the Autostart ROM.

the cursor is placed again at the rightmost position on the top line of the screen.

When COUT is sent a "bell" character (CTRL G), it does not change the screen at all, but instead produces a tone from the speaker. The tone has a frequency of 100Hz and lasts for 1/10th of a second. The output cursor does not move for a bell character.

# BUT SOFT, WHAT LIGHT THROUGH YONDER WINDOW BREAKS!

# (OR, THE TEXT WINDOW)

In the above discussions of the various motions of the output cursor, the words "right", "left", "top", and "bottom" mean the physical right, left, top, and bottom of the standard 40-character wide by 24-line tall screen. There is, however, a way to tell the COUT subroutine that you want it to use only a section of the screen, and not the entire 960-character display. This segregated section of the text screen is called a "window". A program or language can set the positions of the top, bottom, left side, and width of the text window by storing those positions in four locations in memory. When this is done, the COUT subroutine will use the new positions to calculate the size of the screen. It will never print any text outside of this window, and when it must scroll the screen, it will only scroll the text within the window. This gives programs the power to control the placement of text, and to protect areas of the screen from being overwritten with new text.

Location number 32 (hexadecimal \$20) in memory holds the column position of the leftmost column in the window. This position is normally position 0 for the extreme left side of the screen. This number should never exceed 39 (hexadecimal \$27), the leftmost column on the text screen. Location number 33 (hexadecimal \$21) holds the width, in columns, of the cursor window. This number is normally 40 (hexadecimal \$28) for a full 40-character screen. Be careful that the sum of the window width and the leftmost window position does not exceed 40! If it does, it is possible for COUT to place characters in memory locations not on the screen, endangering your programs and data.

Location 34 (hexadecimal \$22) contains the number of the top line of the text window. This is also normally  $\emptyset$ , indicating the topmost line of the display. Location 35 (hexadecimal \$23) holds the number of the bottom line of the screen (plus one), thus normally 24 (hexadecimal \$18) for the bottommost line of the screen. When you change the text window, you should take care that you know the whereabouts of the output cursor, and that it will be inside the new window.

Table 11: Text Window Special Locations						
F4:	Location:		Minimum/Normal/Maximum Val			
Function:	Decimal	Hex	Decimal	Hex		
Left Edge	32	\$20	0/0/39	\$0/\$0/\$17		
Width	33	\$21	0/40/40	\$0/\$28/\$28		
Top Edge	34	\$22	0/0/24	\$0/\$0/\$18		
Bottom Edge	35	\$23	0/24/24	\$0/\$18/\$18		

# SEEING IT ALL IN BLACK AND WHITE

The COUT subroutine has the power to print what's sent to it in either Normal or Inverse text modes (see page 14). The particular form of its output is determined by the contents of location number 50 (hexadecimal \$32). If this location contains the value 255 (hexadecimal \$FF), then COUT will print characters in Normal mode; if the value is 63 (hexadecial \$3F), then COUT will present its display in Inverse mode. Note that this mode change only affects the characters printed after the change has been made. Other values, when stored in location 50, do unusual things: the value 127 prints letters in Flashing mode, but all other characters in Inverse; any other value in location 50 will cause COUT to ignore some or all of its normal character set.

Table 12: Normal/Inverse Control Values					
Value:		Effect:			
Decimal	Hex				
255	\$FF	COUT will display characters in Normal mode.			
63	\$3F	COUT will display characters in Inverse mode.			
127	\$7F	COUT will display letters in Flashing mode, all other characters in Inverse mode.			

The Normal/Inverse "mask" location, as it is called, works by performing a logical "AND" between the bits contained in location 50 and the bits in each outgoing character code. Every bit in location 50 which is a logical "zero" will force the corresponding bit in the character code to become "zero" also, regardless of its former setting. Thus, when location 50 contains 63 (hexadecimal \$3F or binary 00111111), the top two bits of every output character code will be turned "off". This will place characters on the screen whose codes are all between 0 and 63. As you can see from the ASCII Screen Character Code table (Table 7 on page 15), all of these characters are in Inverse mode.

# STANDARD INPUT

There are actually two subroutines which are concerned with the gathering of standard input: RDKEY, which fetches a single keystroke from the keyboard, and GETLN, which accumulates a number of keystrokes into a chunk of information called an *input line*.

# **RDKEY**

The primary function of the RDKEY subroutine is to wait for the user to press a key on the key-board, and then report back to the program which called it with the code for the key which was pressed. But while it does this, RDKEY also performs two other helpful tasks:

1). Input Prompting. When RDKEY is activated, the first thing it does is make visible the hidden output cursor. This accomplishes two things: it reminds the user that the Apple is waiting for a key to be pressed, and it also associates the input it wants with a particular place on the screen. In most cases, the input prompt appears near a word or phrase describing what is being requested by the particular program or language currently in use. The input cursor itself is a flashing representation of whatever character was at the position of the output cursor. Usually this is the blank character, so the input cursor most often appears to be a flashing square.

32

When the user presses a key, RDKEY dutifully removes the input cursor and returns the value of the key which was pressed to the program which requested it. Remember that the output cursor is just a position on the screen, but the input cursor is a flashing character on the screen. They usually move in tandem and are rarely separated from each other, but when the input cursor disappears, the output cursor is still active.

2). Random Number Seeding. While it waits for the user to press a key, RDKEY is continually adding 1 to a pair of numbers in memory. When a key is finally pressed, these two locations together represent a number from Ø to 65,535, the exact value of which is quite unpredictable. Many programs and languages use this number as the base of a random number generator. The two locations which are randomized during RDKEY are numbers 78 and 79 (hexadecimal \$4E and \$4F).

#### **GETLN**

The vast majority of input to the Apple is gathered into chunks called *input lines*. The subroutine in the Apple's ROM called GETLN requests an input line from the keyboard, and after getting one, returns to the program which called it. GETLN has many features and nuances, and it is good to be familiar with the services it offers.

When called, GETLN first prints a prompting character, or "prompt". The prompt helps you to identify which program has called GETLN requesting input. A prompt character of an asterisk (\*) represents the System Monitor, a right caret (>) indicates Apple Integer BASIC, a right bracket (]) is the prompt for Applesoft II BASIC, and an exclamation point (!) is the hallmark of the Apple Mini-Assembler. In addition, the question-mark prompt (?) is used by many programs and languages to indicate that a user program is requesting input. From your (the user's) point of view, the Apple simply prints a prompt and displays an input cursor. As you type, the characters you type are printed on the screen and the cursor moves accordingly. When you press RETURN, the entire line is sent off to the program or language you are talking to, and you get another prompt.

Actually, what really happens is that after the prompt is printed, GETLN calls RDKEY, which displays an input cursor. When RDKEY returns with a keycode, GETLN stores that keycode in an *input buffer* and prints it on the screen where the input cursor was. It then calls RDKEY again. This continues until the user presses **RETURN**. When GETLN receives a RETURN code from the keyboard, it sticks the RETURN code at the end of the input buffer, clears the remainder of the screen line the input cursor was on, and sends the RETURN code to COUT (see above). GETLN then returns to the program which called it. The program or language which requested input may now look at the entire line, all at once, as saved in the input buffer.

At any time while you are typing a line, you can type a CTRL X and cancel that entire line. GETLN will simply forget everything you have typed, print a backslash (\), skip to a new line, and display another prompt, allowing you to retype the line. Also, GETLN can handle a maximum of 255 characters in a line. If you exceed this limit, GETLN will cancel the entire line and you must start over. To warn you that you are approaching the limit, GETLN will sound a tone every keypress starting with the 249th character.

GETLN also allows you to edit and modify the line you are typing in order to correct simple typographical errors. A quick introduction to the standard editing functions and the use of the two arrow keys, — and —, appears on pages 28-29 and 53-55 of the Apple II BASIC Programming Manual, or on pages 27-28, 52-53 and Appendix C of The Applesoft Tutorial, at least one

#### Apple 2 Technical Manual • Apple | Reference Manual • 1979

of which you should have received. Here is a short description of GETLN's editing features:

#### THE BACKSPACE ( ) KEY

Each press of the backspace key makes GETLN "forget" one previous character in the input line. It also sends a backspace character to COUT (see above), making the cursor move back to the character which was deleted. At this point, a character typed on the keyboard will replace the deleted character both on the screen and in the input line. Multiple backspaces will delete successive characters; however, if you backspace over more characters than you have typed, GETLN will forget the entire line and issue another prompt.

#### THE RETYPE ( ) KEY

Pressing the retype key has exactly the same effect as typing the character which is under the cursor. This is extremly useful for re-entering the remainder of a line which you have backspaced over to correct a typographical error. In conjunction with *pure cursor moves* (which follow), it is also useful for recopying and editing data which is already on the screen.

#### **ESCAPE CODES**

When you press the key marked ESC on the keyboard, the Apple's input subroutines go into escape mode. In this mode, eleven keys have separate meanings, called "escape codes". When you press one of these eleven keys, the Apple will perform the function associated with that key. After it has performed the function, the Apple will either continue or terminate escape mode, depending upon which escape code was performed. If you press any key in escape mode which is not an escape code, then that keypress will be ignored and escape mode will be terminated.

The Apple recognizes eleven escape codes, eight of which are *pure cursor moves*, which simply move the cursor without altering the screen or the input line, and three of which are *screen clear codes*, which simply blank part or all of the screen. All of the screen clear codes and the first four pure cursor moves (escape codes @, A, B, C, D, E, and F) terminate the escape mode after operating. The final four escape codes (I, K, M, and J) complete their functions with escape mode active.\*

- ESC A press of the ESC key followed by a press of the A key will move the cursor one space to the right without changing the input line. This is useful for skipping over unwanted characters in an input line: simply backspace back over the unwanted characters, press ESC A to skip each offending symbol, and use the retype key to re-enter the remainder of the line.
- ESC B Pressing ESC followed by B moves the cursor back one space, also without disturbing the input line. This may be used to enter something twice on the same line without retyping it: just type it once, press ESC B repeatedly to get back to the beginning of the phrase, and use the retype key to enter it again.

<sup>\*</sup> These four escape codes are not available on Apples without the Autostart Monitor ROM.

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

- ESC C The key sequence ESC C moves the cursor one line directly down, with no horizontal movement. If the cursor reaches the bottom of the text window, then the cursor remains on the bottom line and the text in the window scrolls up one line. The input line is not modified by the ESC C sequence. This, and ESC D (below), are useful for positioning the cursor at the beginning of another line on the screen, so that it may be re-entered with the retype key.
- ESC D The ESC D sequence moves the cursor directly up one line, again without any horizontal movement. If the cursor reaches the top of the window, it stays there. The input line remains unmodified. This sequence is useful for moving the cursor to a previous line on the screen so that it may be re-entered with the retype key.
- ESC E sequence is called "clear to end of line". When COUT detects this sequence of keypresses, it clears the remainder of the screen line (not the input line!) from the cursor position to the right edge of the text window. The cursor remains where it is, and the input line is unmodified. ESC E always clears the rest of the line to blank spaces, regardless of the setting of the Normal/Inverse mode location (see above).
- ESC F This sequence is called "clear to end of screen". It does just that: it clears everything in the window below or to the right of the cursor. As before, the cursor does not move and the input line is not modified. This is useful for erasing random garbage on a cluttered screen after a lot of cursor moves and editing.
- ESC @ Sequence is called "home and clear". It clears the entire window and places the cursor in the upper left-hand corner. The screen is cleared to blank spaces, regardless of the setting of the Normal/Inverse location, and the input line is not changed (note that "@" is SHIFT P).
- ESC K These four escape codes are synonyms for the four pure cursor moves given above.

  ESC J When these four escape codes finish their respective functions, they do not turn off the ESC M escape mode: you can continue typing these escape codes and moving the cursor around ESC I the screen until you press any key other than another escape code. These four keys are placed in a "directional keypad" arrangement, so that the direction of each key from the center of the keypad corresponds to the direction which that escape code moves the cursor.

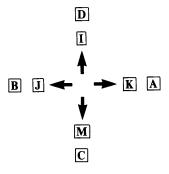


Figure 4. Cursor-moving Escape Codes.

#### THE RESET CYCLE

When you turn your Apple's power switch on\* or press and release the **RESET** key, the Apple's 6502 microprocessor initiates a RESET cycle. It begins by jumping into a subroutine in the Apple's Monitor ROM. In the two different versions of this ROM, the Monitor ROM and the Autostart ROM, the RESET cycle does very different things.

# AUTOSTART ROM RESET

Apples with the Autostart ROM begin their RESET cycles by flipping the soft switches which control the video screen to display the full primary page of Text mode, with Low-Resolution Graphics mixed mode lurking behind the veil of text. It then opens the text window to its full size, drops the output cursor to the bottom of the screen, and sets Normal video mode. Then it sets the COUT and KEYIN switches to use the Apple's internal keyboard and video display as the standard input and output devices. It flips annunciators  $\emptyset$  and 1 ON and annunciators 2 and 3 OFF on the Game I/O connector, clears the keyboard strobe, turns off any active I/O Expansion ROM (see page 84), and sounds a "beep!".

These actions are performed every time you press and release the **RESET** key on your Apple. At this point, the Autostart ROM peeks into two special locations in memory to see if it's been RESET before or if the Apple has just been powered up (these special locations are described below). If the Apple has just been turned on, then the Autostart ROM performs a "cold start"; otherwise, it does a "warm start".

1) Cold Start. On a freshly activated Apple, the RESET cycle continues by clearing the screen and displaying "APPLE II" top and center. It then sets up the special locations in memory to tell itself that it's been powered up and RESET. Then it starts looking through the rightmost seven slots in your Apple's backplane, looking for a Disk II Controller Card. It starts the search with Slot 7 and continues down to Slot 1. If it finds a disk controller card, then it proceeds to bootstrap the Apple Disk Operating System (DOS) from the diskette in the disk drive attached to the controller card it discovered. You can find a description of the disk bootstrapping procedure in Do's and Don'ts of DOS, Apple part number A2L0012, page 11.

If the Autostart ROM cannot find a Disk II controller card, or you press **RESET** again before the disk booting procedure has completed, then the RESET cycle will continue with a "lukewarm start". It will initialize and jump into the language which is installed in ROM on your Apple. For a Revision Ø Apple, either without an Applesoft II Firmware card or with such a card with its controlling switch in the DOWN position, the Autostart ROM will start Apple Integer BASIC. For Apple II-Plus systems, or Revision Ø Apple IIs with the Applesoft II Firmware card with the switch in the UP position, the Autostart ROM will begin Applesoft II Floating-Point BASIC.

2) Warm Start. If you have an Autostart ROM which has already performed a cold start cycle, then each time you press and release the **RESET** key, you will be returned to the language you were using, with your program and variables intact.

<sup>\*</sup> Power-on RESET cycles occur only on Revision 1 Apples or Revision Ø Apples with at least one Disk II controller card.

# **AUTOSTART ROM SPECIAL LOCATIONS**

The three "special locations" used by the Autostart ROM all reside in an area of RAM memory reserved for such system functions. Following is a table of the special locations used by the Autostart ROM:

Table 13: Autostart ROM Special Locations				
Location: Decimal	Hex	Contents:		
1010 1011	\$3F2 \$3F3	Soft Entry Vector. These two locations contain the address of the reentry point for whatever language is in use. Normally contains \$E003.		
1012	\$3F4	Power-Up Byte. Normally contains \$45. See below.		
64367 (-1169)	\$FB6F	This is the beginning of a machine language subroutine which sets up the power-up location.		

When the Apple is powered up, the Autostart ROM places a special value in the power-up location. This value is the Exclusive-OR of the value contained in location 1011 with the constant value 165. For example, if location 1011 contains 224 (its normal value), then the power-up value will be:

	Decimal	Hex	Binary
Location 1011	224	\$EØ	11100000
Constant	165	<b>\$A5</b>	10100101
Power-Up Value	69	\$45	01000101

Your programs can change the soft entry vector, so that when you press **RESET** you will go to some program other than a language. If you change this soft entry vector, however, you should make sure that you set the value of the power-up byte to the Exclusive-OR of the high part of your new soft entry vector with the constant decimal 165 (hexadecimal \$A5). If you do not set this power-up value, then the next time you press **RESET** the Autostart ROM will believe that the Apple has just been turned on and it will do another cold start.

For example, you can change the soft entry vector to point to the Apple System Monitor, so that when you press **RESET** you will be placed into the Monitor. To make this change, you must place the address of the beginning of the Monitor into the two soft entry vector locations. The Monitor begins at location \$FF69, or decimal 65385. Put the last two hexadecimal digits of this address (\$69) into location \$3F2 and the first two digits (\$FF) into location \$3F3. If you are working in decimal, put 105 (which is the remainder of 65385/256) into location 1010 and the value 255 (which is the integer quotient of 65385/256) into location 1011.

Now you must set up the power-up location. There is a machine-language subroutine in the Autostart ROM which wil automatically set the value of this location to the Exclusive-OR mentioned above. Al you need to do is to execute a JSR (Jump to SubRoutine) instruction to the address \$FB6F. If you are working in BASIC, you should perform a CALL -1169. Now everything is set, and the next time you press **RESET**, you will enter the System Monitor.

To make the **RESET** key work in its usual way, just restore the values in the soft entry vector to their former values (\$EØØ3, or decimal 57347) and again call the subroutine described above.

# "OLD MONITOR" ROM RESET

A RESET cycle in the Apple II Monitor ROM begins by setting Normal video mode, a full screen of Primary Page text with the Color Graphics mixed mode behind it, a fully-opened text window, and the Apple's standard keyboard and video screen as the standard input and output devices. It sounds a "beep!", the cursor leaps to the bottom line of the uncleared text screen, and you find yourself facing an asterisk (\*) prompt and talking to the Apple System Monitor.

38

"A2M 030-0004-01 5-038.PICT" 123 KB 2001-07-23 dpi: 600h x 600v pix: 2986h x 4663v

Buried deep within the recesses of the Apple's ROM is a masterful program called the System Monitor. It acts as both a supervisor of the system and a slave to it; it controls all programs and all programs use it. You can use the powerful features of the System Monitor to discover the hidden secrets in all 65,536 memory locations. From the Monitor, you may look at one, some, or all locations; you may change the contents of any location; you can write programs in Machine and Assembly languages to be executed directly by the Apple's microprocessor; you can save vast quantities of data and programs onto cassette tape and read them back in again; you can move and compare thousands of bytes of memory with a single command; and you can leave the Monitor and enter any other program or language on the Apple.

# ENTERING THE MONITOR

The Apple System Monitor program begins at location number \$FF69 (decimal 65385 or -151) in memory. To enter the Monitor, you or your BASIC program can CALL this location. The Monitor's prompt (an asterisk [\*]) will appear on the left edge of the screen, with a flashing cursor to its right. The Monitor accepts standard input lines (see page 32) just like any other system or language on the Apple. It will not take any action until you press **RETURN**. Your input lines to the Monitor may be up to 255 characters in length. When you have finished your stay in the Monitor, you can return to the language you were previously using by typing **CTRL C RETURN** (or, with the Apple DOS, 3 **D G RETURN**), or simply press **RESET**.\*

### ADDRESSES AND DATA

Talking to the Monitor is somewhat like talking to any other program or language on the Apple: you type a line on the keyboard, followed by a **RETURN**, and the Monitor will digest what you typed and act according to those instructions. You will be giving the Monitor three types of information: addresses, values, and commands. Addresses and values are given to the Monitor in hexadecimal notation. Hexadecimal notation uses the ten decimal digits (\$\vartheta\$-9) to represent themselves and the first six letters (A-F) to represent the numbers 10 through 15. A single hexadecimal digit can, therefore, have one of sixteen values from 0 to 15. A pair of hex digits can assume any value from 0 to 255, and a group of four hex digits can denote any number from 0 to 65,536. It so happens that any address in the Apple can be represented by four hex digits, and any value by two hex digits. This is how you tell the Monitor about addresses and values. When the Monitor is looking for an address, it will take any group of hex digits. If there are fewer than four digits in the group, it will prepend leading zeroes; if there are more than four hex digits, the Monitor will truncate the group and use only the last four hex digits. It follows the same procedure when looking for two-digit data values.

The Monitor recognizes 22 different command characters. Some of these are punctuation marks, others are upper-case letters or control characters. In the following sections, the full name of a command will appear in capital letters. The Monitor needs only the first letter of the command name. Some commands are invoked with control characters. You should note that although the Monitor recognizes and interprets these characters, a control character typed on an input line will not appear on the screen.

<sup>\*</sup> This does not work on Apples without the Autostart ROM.

The Monitor remembers the addresses of up to five locations. Two of these are special: they are the addresses of the last location whose value you inquired about, and the location which is next to have its value changed. These are called the *last opened location* and the *next changeable location*. The usefulness of these two addresses will be revealed shortly.

### **EXAMINING THE CONTENTS OF MEMORY**

When you type the address of a location in memory alone on an input line to the Monitor, it will reply\* with the address you typed, a dash, a space, and the value\*\* contained in that location, thus:

\*E000 E000- 20 \*300 0300- 99

Each time the Monitor displays the value contained in a location, it remembers that location as the *last opened location*. For technical reasons, it also considers that location as the *next change-able location*.

## **EXAMINING SOME MORE MEMORY**

If you type a period (.) on an input line to the Monitor, followed by an address, the Monitor will display a *memory dump*: the values contained in all locations from the last opened location to the location whose address you typed following the period. The Monitor then considers the last location displayed to be both the last opened location and the next changeable location.

<sup>\*</sup> In the examples, your queries are in normal type and the Apple replies in boldface.

<sup>\*\*</sup> The values printed in these examples may differ from the values displayed by your Apple for the same instructions.

```
* 2 Ø
ØØ2Ø- ØØ
* . 2B
ØØ21- 28 ØØ 18 ØF ØC ØØ ØØ
ØØ28- A8 Ø6 DØ Ø7
*300
Ø3ØØ- 99
* . 315
Ø3Ø1- B9
             Ø8 ØA ØA ØA 99
Ø3Ø8- ØØ Ø8 C8 DØ F4 A6 2B A9
Ø31Ø- Ø9 85 27 AD CC Ø3
* . 32A
\emptyset 316 - 85 41
Ø318- 84 4Ø 8A 4A 4A 4A Ø9
Ø32Ø- CØ 85 3F A9 5D 85 3E 2Ø
Ø328- 43 Ø3 2Ø
```

You should notice several things about the format of a memory dump. First, the first line in the dump begins with the address of the location following the last opened location; second, all other lines begin with addresses which end alternately in zeroes and eights; and third, there are never more than eight values displayed on a single line in a memory dump. When the Monitor does a memory dump, it starts by displaying the address and value of the location following the last opened location. It then proceeds to the next successive location in memory. If the address of that location ends in an 8 or a Ø, the Monitor will "cut" to a new line and display the address of that location and continue displaying values. After it has displayed the value of the location whose address you specified, it stops the memory dump and sets the address of both the last opened and the next changeable location to be the address of the last location in the dump. If the address specified on the input line is less than the address of the last opened location, the Monitor will display the address and value of only the location following the last opened location.

You can combine the two commands (opening and dumping) into one operation by concatenating the second to the first; that is, type the first address, followed by a period and the second address. This two-addresses-separated-by-a-period form is called a *memory range*.

```
*300.32F
Ø3ØØ− 99 B9 ØØ
                 Ø8 ØA ØA ØA 99
          Ø8 C8 DØ F4 A6 2B A9
\emptyset 31\emptyset - \emptyset 9
         85 27 AD CC 03 85 41
Ø318- 84 4Ø 8A 4A 4A
                        4 A
Ø32Ø- CØ 85 3F A9 5D 85
Ø328- 43 Ø3 2Ø
                46 Ø3 A5
*30.40
0030- AA 00 FF AA 05 C2 05 C2
9938- 1B FD D9 93 3C 99 49 99
9949- 39
*EØ15.EØ25
```

```
E Ø 15- 4C ED FD
E Ø 18- A9 2Ø C5 24 BØ ØC A9 8D
E Ø 2Ø- AØ Ø 7 2Ø ED FD A9
```

### **EXAMINING STILL MORE MEMORY**

A single press of the **RETURN** key will cause the Monitor to respond with one line of a memory dump; that is, a memory dump from the location following the last opened location to the next eight-location "cut". Once again, the last location displayed is considered the last opened and next changeable location.

# CHANGING THE CONTENTS OF A LOCATION

You've heard all about the "next changeable location"; now you're going to use it. Type a colon followed by a value.

Presto! The contents of the next changeable location have just been changed to the value you typed. Check this by examining that location again:

\*0 9999- 5F

43

"A2M 030-0004-01 5-043.PICT" 211 KB 2001-07-23 dpi: 600h x 600v pix: 3013h x 4654v

You can also combine opening and changing into one operation:

```
*302:42
*302
#302 42
```

When you change the contents of a location, the old value which was contained in that location disappears, never to be seen again. The new value will stick around until it is replaced by another hexadecimal value.

# CHANGING THE CONTENTS OF CONSECUTIVE LOCATIONS

You don't have to type an address, a colon, a value, and press **RETURN** for each and every location you wish to change. The Monitor will allow you to change the values of up to eighty-five locations at a time by typing only the initial address and colon, and then all the values separated by spaces. The Monitor will duly file the consecutive values in consecutive locations, starting at the next changeable location. After it has processed the string of values, it will assume that the location following the last changed location is the next changeable location. Thus, you can continue changing consecutive locations without breaking stride on the next input line by typing another colon and more values.

```
*300:69 01 20 ED FD 4C 0 3

*300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#300

#3
```

# **MOVING A RANGE OF MEMORY**

You can treat a range of memory (specified by two addresses separated by a period) as an entity

44

"A2M 030-0004-01 5-044.PICT" 260 KB 2001-07-23 dpi: 600h x 600v pix: 2958h x 4645v

unto itself and move it from one place to another in memory by using the Monitor's MOVE command. In order to move a range of memory from one place to another, the Monitor must be told both where the range is situated in memory and where it is to be moved. You give this information to the Monitor in three parts: the address of the destination of the range, the address of the first location in the range proper, and the address of the last location in the range. You specify the starting and ending addresses of the range in the normal fashion, by separating them with a period. You indicate that this range is to be placed somewhere else by separating the range and the destination address with a left caret (<). Finally, you tell the Monitor that you want to move the range to the destination by typing the letter M, for "MOVE". The final command looks like this:

```
{destination} < {start} . {end} M
```

When you type this line to the Monitor, of course, the words in curly brackets should be replaced by hexadecimal addresses and the spaces should be omitted. Here are some real examples of memory moves:

```
* Ø . F
9999- 5F 99 95 97
9998- 99 99 99 99 99
*300:A9 8D 20 ED FD A9 45 20 DA FD 4C 00 03
*300.30C
Ø3ØØ- A9 8D 2Ø ED FD A9 45 2Ø
0308- DA FD 4C 00 03
*Ø<3ØØ.3ØCM
* Ø . C
ØØØØ- A9 8D 2Ø ED FD A9 45 2Ø
0008- DA FD 4C 00 03
* 31Ø<8.AM
*310.312
Ø31Ø- DA FD 4C
* 2<7.9M
* Ø . C
0000- A9 8D 20 DA FD A9 45 20
ØØØ8- DA FD 4C ØØ Ø3
```

The Monitor simply makes a copy of the indicated range and moves it to the specified destination. The original range is left undisturbed. The Monitor remembers the last location in the original range as the last opened location, and the first location in the original range as the next changeable location. If the second address in the range specification is less than the first, then only one value (that of the first location in the range) will be moved.

If the destination address of the MOVE command is inside the original range, then strange and (sometimes) wonderful things happen: the locations between the beginning of the range and the

destination are treated as a sub-range and the values in this sub-range are replicated throughout the original range. See "Special Tricks", page 55, for an interesting application of this feature.

# COMPARING TWO RANGES OF MEMORY

You can use the Monitor to compare two ranges of memory using much the same format as you use to move a range of memory from one place to another. In fact, the VERIFY command can be used immediately after a MOVE to make sure that the move was successful.

The VERIFY command, like the MOVE command, needs a range and a destination. In short-hand:

```
\{destination\} < \{start\} \cdot \{end\} V
```

The Monitor compares the range specified with the range beginning at the destination address. If there is any discrepancy, the Monitor displays the address at which the difference was found and the two offending values.

- \*Ø:D7 F2 E9 F4 F4 E5 EE AØ E2 F9 AØ C3 C4 C5
- \*300<0.DM
- \*300<0.DV
- \*6:E4
- \*300<0.DV

 $\emptyset \emptyset \emptyset 6-E4$  (EE)

Notice that the VERIFY command, if it finds a discrepancy, displays the address of the location in the original range whose value differs from its counterpart in the destination range. If there is no discrepancy, VERIFY displays nothing. It leaves both ranges unchanged. The last opened and next changeable locations are set just as in the MOVE command. As before, if the ending address of the range is less than the starting address, the values of only the first locations in the ranges will be compared. VERIFY also does unusual things if the destination is within the original range; see "Special Tricks", page 55.

# SAVING A RANGE OF MEMORY ON TAPE

The Monitor has two special commands which allow you to save a range of memory onto cassette tape and recall it again for later use. The first of these two commands, WRITE, lets you save the contents of one to 65,536 memory locations on standard cassette tape.

To save a range of memory to tape, give the Monitor the starting and ending addresses of the range, followed by the letter W (for WRITE):

46

"A2M 030-0004-01 5-046.PICT" 352 KB 2001-07-23 dpi: 600h x 600v pix: 3022h x 4663v

```
{start} . {end} W
```

To get an accurate recording, you should put the tape recorder in *record* mode before you press **RETURN** on the input line. Let the tape run a few seconds, then press **RETURN**. The Monitor will write a ten-second "leader" tone onto the tape, followed by the data. When the Monitor is finished, it will sound a "beep! and give you another prompt. You should then rewind the tape, and label the tape with something intelligible about the memory range that's on the tape and what it's supposed to be.

```
*0.FF FF AD 30 C0 88 D0 04 C6 01 F0 08 C
A D0 F6 A6 00 4C 02 00 60

*0.14

$000 - FF FF AD 30 C0 88 D0 04

$0008 - C6 01 F0 08 CA D0 F6 A6

$010 - 00 4C 02 00 60

*0.14W
```

It takes about 35 seconds total to save the values of 4,096 memory locations preceded by the ten-second leader onto tape. This works out to a speed of about 1,350 bits per second, average. The WRITE command writes one extra value on the tape after it has written the values in the memory range. This extra value is the *checksum*. It is the partial sum of all values in the range. The READ subroutine uses this value to determine if a READ has been successful (see below).

# READING A RANGE FROM TAPE

Once you've saved a memory range onto tape with the Monitor's WRITE command, you can read that memory range back into the Apple by using the Monitor's READ command. The data values which you've stored on the tape need not be read back into the same memory range from whence they came; you can tell the Monitor to put those values into any similarly sized memory range in the Apple's memory.

The format of the READ command is the same as that of the WRITE command, except that the command letter is R, not W:

```
{start} . {end} R
```

Once again, after typing the command, don't press **RETURN**. Instead, start the tape recorder in PLAY mode and wait for the tape's nonmagnetic leader to pass by. Although the WRITE command puts a ten-second leader tone on the beginning of the tape, the READ command needs only three seconds of this leader in order to lock on to the frequency. So you should let a few seconds of tape go by before you press **RETURN**, to allow the tape recorder's output to settle down to a steady tone.

47

After the Monitor has read in and stored all the values on the tape, it reads in the extra checksum value. It compares the checksum on the tape to its own checksum, and if the two differ, the Monitor beeps the speaker and displays "ERR". This warns you that there was a problem during the READ and that the values stored in memory aren't the values which were recorded on the tape. If, however, the two checksums match, the Monitor will give you another prompt.

# CREATING AND RUNNING MACHINE LANGUAGE PROGRAMS

Machine language is certainly the most efficient language on the Apple, albeit the least pleasant in which to code. The Monitor has special facilities for those of you who are determined to use machine language to simplify creating, writing, and debugging machine language programs.

You can write a machine language program, take the hexadecimal values for the opcodes and operands, and store them in memory using the commands covered above. You can get a hexadecimal dump of your program, move it around in memory, or save it to tape and recall it again simply by using the commands you've already learned. The most important command, however, when dealing with machine language programs is the GO command. When you open a location from the Monitor and type the letter G, the Monitor will cause the 6502 microprocessor to start executing the machine language program which begins at the last opened location. The Monitor treats this program as a subroutine: when it's finished, all it need do is execute an RTS (return from subroutine) instruction and control will be transferred back to the Monitor.

Your machine language programs can call many subroutines in the Monitor to do various things. Here is an example of loading and running a machine language program to display the letters A through Z:

```
*300:A9 C1 20 ED FD 18 69 1 C9 DB D0 F6 60

*300.30C

0300- A9 C1 20 ED FD 18 69 01

0308- C9 DB D0 F6 60

*300G

ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

(The instruction set of the Apple's 65\( \text{02} \) microprocessor is listed in Appendix A of this manual.)

48

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

Now, straight hexadecimal code isn't the easiest thing in the world to read or debug. With this in mind, the creators of the Apple's Monitor neatly included a command to list machine language programs in assembly language form. This means that instead of having one, two, or three bytes of unformatted hexadecimal gibberish per instruction you now have a three-letter mnemonic and some formatted hexadecimal gibberish to comprehend for each instruction. The LIST command to the Monitor will start at the specified location and display a screenfull (20 lines) of instructions:

* 3 Ø Ø L				
Ø3ØØ—	A 9	C1	LDA	#\$C1
Ø3Ø2-	2 Ø	ED FD	JSR	\$FDED
Ø3Ø5—	18		CLC	
Ø3Ø6-	69	Ø 1	ADC	#\$Ø1
Ø3Ø8 <b>-</b>	C9	DB	CMP	#\$DB
Ø3ØA-	DØ	F 6	BNE	\$0302
Ø3ØC-	6 Ø		RTS	
Ø3ØD-	øø		BRK	
Ø3ØE-	øø		BRK	
Ø3ØF-	ØØ		BRK	
Ø31 <b>Ø</b> —	øø		BRK	
Ø311-	øø		BRK	
Ø312-	øø		BRK	
Ø313 <del>-</del>	øø		BRK	
Ø314 <del>-</del>	øø		BRK	
Ø315 <del>-</del>	Ø Ø		BRK	
Ø316 <del>-</del>	øø		BRK	
Ø317 <del>-</del>	øø		BRK	
Ø318-	øø		BRK	
Ø319 <b>-</b>	øø		BRK	
*				

Recognize those first few lines? They're the assembly language form of the program you typed in a page or so ago. The rest of the lines (the BRK instructions) are just there to fill up the screen. The address that you specify is remembered by the Monitor, but not in one of the ways explained before. It's put in the *Program Counter*, which is used solely to point to locations within programs. After a LIST command, the Program Counter is set to point to the location immediately following the last location displayed on the screen, so that if you do another LIST command it will continue with another screenfull of instructions, starting where the first screen left off.

# THE MINI-ASSEMBLER

There is another program within the Monitor\* which allows you to type programs into the Apple in the same assembly format which the LIST command displays. This program is called the Apple Mini-Assembler. It is a "mini"-assembler because it cannot understand symbolic labels, something that a full-blown assembler must do. To run the Mini-Assembler, type:

<sup>\*</sup> The Mini-Assembler does not actually reside in the Monitor ROM, but is part of the Integer BASIC ROM set. Thus, it is not available on Apple II Plus systems or while Firmware Applesoft II is in use.

\*F666G

You are now in the Mini-Assembler. The exclamation point (!) is the prompt character. During your stay in the Mini-Assembler, you can execute any Monitor command by preceding it with a dollar sign (\$). Aside from that, the Mini-Assembler has an instruction set and syntax all its own.

The Mini-Assembler remembers one address, that of the Program Counter. Before you start to enter a program, you must set the Program Counter to point to the location where you want your program to go. Do this by typing the address followed by a colon. Follow this with the mnemonic for the first instruction in your program, followed by a space. Now type the operand of the instruction (Formats for operands are listed on page 66). Now press RETURN. The Mini-Assembler converts the line you typed into hexadecimal, stores it in memory beginning at the location of the Program Counter, and then disassembles it again and displays the disassembled line on top of your input line. It then poses another prompt on the next line. Now it's ready to accept the second instruction in your program. To tell it that you want the next instruction to follow the first, don't type an address or a colon, but only a space, followed by the next instruction's mnemonic and operand. Press RETURN. It assembles that line and waits for another.

If the line you type has an error in it, the Mini-Assembler will beep loudly and display a circumflex (^) under or near the offending character in the input line. Most common errors are the result of typographical mistakes: misspelled mnemonics, missing parentheses, etc. The Mini-Assembler also will reject the input line if you forget the space before or after a mnemonic or include an extraneous character in a hexadecimal value or address. If the destination address of a branch instruction is out of the range of the branch (more than 127 locations distant from the address of the instruction), the Mini-Assembler will also flag this as an error.

! 300:1	LDX #02		
<b>Ø3ØØ</b> — ! LDA	A2 Ø2 \$Ø,X	LDX	#\$\$2
	B5 ØØ \$10,X	LDA	\$ØØ,X
<b>Ø3Ø4</b> - ! DEX	95 10	STA	\$1Ø,X
<b>Ø3Ø6</b> - ! STA		DEX	
Ø3Ø7- ! BPL		CØ STA	\$CØ3Ø
Ø3ØA— ! BRK	10 F6	BPL	\$0302
Ø3ØC− !	99	BRK	

To exit the Mini-Assembler and re-enter the Monitor, either press RESET or type the Monitor

command (preceded by a dollar sign) FF69G:

!\$FF69G

\*

. 2 4 4 1

Your assembly language program is stored in memory. You can look at it again with the LIST command:

*300L					
Ø3ØØ—	A2	<b>Ø</b> 2		LDX	#\$02
Ø3Ø2-	<b>B5</b>	Ø Ø		LDA	\$00,X
Ø3Ø4 <del>-</del>	95	10		STA	\$10,X
Ø3Ø6-	CA			DEX	
Ø3Ø7—	<b>8D</b>	3 Ø	СØ	STA	\$CØ3Ø
Ø3ØA-	1 Ø	F 6		$\mathbf{BPL}$	\$0302
Ø3ØC-	øø			BRK	
Ø3 ØD-	ØØ			BRK	
Ø3ØE-	ØØ			BRK	
Ø3ØF-	øø			BRK	
Ø31Ø—	øø			BRK	
Ø311-	ØØ			BRK	
Ø312-	øø			BRK	
<b>Ø313</b> —	Ø Ø			BRK	
Ø314 <del>-</del>	Ø Ø			BRK	
Ø315 <del>-</del>	øø			BRK	
Ø316 <del>-</del>	øø			BRK	
Ø317 <b>-</b>	Ø Ø			BRK	
Ø318-	øø			BRK	
Ø319 <b>-</b>	øø			BRK	
*					

# **DEBUGGING PROGRAMS**

As put so concisely by Lubarsky\*, "There's always one more bug." Don't worry, the Monitor provides facilities for stepping through ornery programs to find that one last bug. The Monitor's STEP\*\* command decodes, displays, and executes one instruction at a time, and the TRACE\*\* command steps quickly through a program, stopping when a BRK instruction is executed.

Each STEP command causes the Monitor to execute the instruction in memory pointed to by the Program Counter. The instruction is displayed in its disassembled form, then executed. The contents of the 6502's internal registers are displayed after the instruction is executed. After execution, the Program Counter is bumped up to point to the next instruction in the program.

Here's what happens when you STEP through the program you entered using the Mini-Assembler, above:

<sup>\*</sup> In Murphy's Law, and Other Reasons why Things Go Wrong, edited by Arthur Bloch. Price/Stern/Sloane 1977.

<sup>\*\*</sup> The STEP and TRACE commands are not available on Apples with the Autostart ROM.

Notice that after the third instruction was executed, we examined the contents of location 12. They were as we expected, and so we continued stepping. The Monitor keeps the Program Counter and the last opened address separate from one another, so that you can examine or change the contents of memory while you are stepping through your program.

The TRACE command is just an infinite STEPper. It will stop TRACEing the execution of a program only when you push **RESET** or it encounters a BRK instruction in the program. If the TRACE encounters the end of a program which returns to the Monitor via an RTS instruction, the TRACEing will run off into never-never land and must be stopped with the **RESET** button.

```
A=\emptyset B X=\emptyset \emptyset Y=D8 P=32 S=F8
Ø3Ø2-
             B5 ØØ
                                           $00,X
                                 LDA
 A=\emptyset A X=\emptyset \emptyset Y=D8 P=3\emptyset S=F8
Ø3Ø4-
             95
                 10
                                 STA
                                           $10,X
 A=\emptyset A X=\emptyset \emptyset Y=D8 P=3\emptyset S=F8
Ø3Ø6-
            CA
                                 DEX
 A=\emptyset A X=FF Y=D8 P=B\emptyset S=F8
0307-
             8D 30 C0
                                 STA
                                           $CØ3Ø
 A=\emptyset A X=FF Y=D8 P=B\emptyset S=F8
Ø3ØA-
             10 F6
                                 BPL
                                           $0302
 A=\emptyset A X=FF Y=D8 P=B\emptyset S=F8
Ø3ØC-
             99
                                 BRK
Ø3 ØC-
              A=\emptyset A X=FF Y=D8 P=B\emptyset S=F8
```

# **EXAMINING AND CHANGING REGISTERS**

As you saw above, the STEP and TRACE commands displayed the contents of the 65\(\textit{0}\)2's internal registers after each instruction. You can examine these registers at will or pre-set them when you TRACE, STEP, or GO a machine language program.

The Monitor reserves five locations in memory for the five 6502 registers: A, X, Y, P (processor status register), and S (stack pointer). The Monitor's EXAMINE command, invoked by a CTRL E, tells the Monitor to display the contents of these locations on the screen, and lets the location which holds the 6502's A-register be the next changeable location. If you want to change the values in these locations, just type a colon and the values separated by spaces. Next time you give the Monitor a GO, STEP, or TRACE command, the Monitor will load these five locations into their proper registers inside the 6502 before it executes the first instruction in your program.

```
* CTRL E
 A=\emptyset A X=FF Y=D8 P=B\emptyset S=F8
*:BØ Ø2
* CTRL E
 A=B\emptyset X=\emptyset 2 Y=D8 P=B\emptyset S=F8
*306S
Ø3Ø6-
             CA
                                 DEX
 A=B\emptyset X=\emptyset 1 Y=D8 P=3\emptyset S=F8
* S
Ø3Ø7-
             8D 30 C0
                                 STA
                                           $CØ3Ø
 A=B\emptyset X=\emptyset1 Y=D8 P=3\emptyset S=F8
* S
Ø3ØA-
             10 F6
                                 BPL
                                           $0302
 A=B\emptyset X=\emptyset 1 Y=D8 P=3\emptyset S=F8
```

53

# MISCELLANEOUS MONITOR COMMANDS

You can control the setting of the Inverse/Normal location used by the COUT subroutine (see page 32) from the Monitor so that all of the Monitor's output will be in Inverse video. The INVERSE command does this nicely. Input lines are still displayed in Normal mode, however. To return the Monitor's output to Normal mode, use the NORMAL command.

The BASIC command, invoked by a CTRL B, lets you leave the Monitor and enter the language installed in ROM on your Apple, usually either Apple Integer or Applesoft II BASIC. Any program or variables that you had previously in BASIC will be lost. If you've left BASIC for the Monitor and you want to re-enter BASIC with your program and variables intact, use the CTRL C (CONTINUE BASIC) command. If you have the Apple Disk Operating System (DOS) active, the '3DØG' command will return you to the language you were using, with your program and variables intact.

The PRINTER command, activated by a CTRL P, diverts all output normally destined for the screen to an Apple Intelligent Interface® in a given slot in the Apple's backplane. The slot number should be from 1 to 7, and there should be an interface card in the given slot, or you will lose control of your Apple and your program and variables may be lost. The format for the command is:

{slot number} CTRL P

A PRINTER command to slot number Ø will reset the flow of printed output back to the Apple's video screen.

The KEYBOARD command similarly substitutes the device in a given backplane slot for the Apple's keyboard. For details on how these commands and their BASIC counterparts PR# and IN# work, please refer to "CSW and KSW Switches", page 83. The format for the KEYBOARD command is:

{slot number} CTRL K

A slot number of  $\emptyset$  for the KEYBOARD command will force the Monitor to listen for input from the Apple's built-in keyboard.

The Monitor will also perform simple hexadecimal addition and subtraction. Just type a line in the format:

```
{value} + {value}
{value} - {value}
```

The Apple will perform the arithmetic and display the result:

```
*20+13
=33
*4A-C
=3E
*FF+4
=03
*3-4
```

# SPECIAL TRICKS WITH THE MONITOR

You can put as many Monitor commands on a single line as you like, as long as you separate them with spaces and the total number of characters in the line is less than 254. You can intermix any and all commands freely, except the STORE (:) command. Since the Monitor takes all values following a colon and places them in consecutive memory locations, the last value in a STORE must be followed by a letter command before another address is encountered. The NORMAL command makes a good separator; it usually has no effect and can be used anywhere.

Single-letter commands such as L, S, I, and N need not be separated by spaces.

If the Monitor encounters a character in the input line which it does not recognize as either a hexadecimal digit or a valid command character, it will execute all commands on the input line up to that character, and then grind to a halt with a noisy beep, ignoring the remainder of the input line.

The MOVE command can be used to replicate a pattern of values throughout a range in memory.

To do this, first store the pattern in its first position in the range:

```
*300:11 22 33
```

Remember the number of values in the pattern: in this case, 3. Then use this special arrangement of the MOVE command:

```
\{ \text{start} + \text{number} \} < \{ \text{start} \}. \{ \text{end} - \text{number} \} M
```

This MOVE command will first replicate the pattern at the locations immediately following the original pattern, then re-replicate that pattern following itself, and so on until it fills the entire range.

```
*303<300.32DM

*300.32F

#309-11 22 33 11 22 33 11 22

#3398-33 11 22 33 11 22 33 11

#319-22 33 11 22 33 11 22 33

#318-11 22 33 11 22 33 11 22

#329-33 11 22 33 11 22 33 11

#328-22 33 11 22 33 11 22 33
```

A similar trick can be done with the VERIFY command to check whether a pattern repeats itself through memory. This is especially useful to verify that a given range of memory locations all contain the same value:

```
*300:0

*301<300.31FM

*301<300.31FV

*304:02

*301<300.31FV

9303-90.31FV

9303-90.000
```

You can create a command line which will repeat all or part of itself indefinitely by beginning the part of the command line which is to be repeated with a letter command, such as N, and ending it with the sequence 34:n, where n is a hexadecimal number specifying the character position of the command which begins the loop; for the first character in the line,  $n=\emptyset$ . The value for n must be followed with a space in order for the loop to work properly.

```
*N 300 302 34:0

9309-11
```

Apple 2 Technical Manual	•	Apple If Reference Manual	•	1979
ADDIC Z I CUITICAI MATICAI		ANDRE II INCICIOS MAINAI		

 $\emptyset 3 \emptyset 2 - 33$ 0300 - 11 $\emptyset 3 \emptyset 2 - 33$ 0300 - 110302 - 330300 - 110302 - 330300 - 110302 - 330300 - 11Ø3Ø2- 33 Ø3Ø

The only way to stop a loop like this is to press **RESET**.

## CREATING YOUR OWN COMMANDS

The USER (CTRLY) command, when encountered in the input line, forces the Monitor to jump to location number \$3F8 in memory. You can put your own JMP instruction in this location which will jump to your own program. Your program can then either examine the Monitor's registers and pointers or the input line itself. For example, here is a program which will make the CTRL Y command act as a "comment" indicator: everything on the input line following the CTRL Y will be displayed and ignored.

#### \*F666G

!300:LDY \$34

LDY \$34 Ø3ØØ-A4 34 ! LDA 200, Y

LDA \$ \\ \mathbf{9} \, \mathbf{9} 0302-B9 ØØ Ø2

! JSR FDED

\$FDED Ø3Ø5-**JSR** 2 Ø ED FD

! INY

**Ø3Ø8**-INY **C8** 

! CMP #\$8D

Ø3Ø9-C9 8D **CMP** #\$8D

! BNE 302

Ø3ØB-DØ F5 **BNE** \$0302

! JMP \$FF69

4C 69 FF \$FF69 Ø3 ØD− **JMP** 

!3F8:JMP \$300

**JMP** \$0300 Ø3F8-4C ØØ Ø3

57

"A2M 030-0004-01 5-057.PICT" 214 KB 2001-07-23 dpi: 600h x 600v pix: 2949h x 4681v

!\$FF69G

\*CTRLY THIS IS A TEST.
THIS IS A TEST.

# **SUMMARY OF MONITOR COMMANDS**

#### Summary of Monitor Commands.

Examining Memory.

{adrs} Examines the value contained in one location.

{adrs1}.{adrs2} Displays the values contained in all locations

between {adrs1} and {adrs2}.

**RETURN** Displays the values in up to eight locations fol-

lowing the last opened location.

Changing the Contents of Memory.

{adrs}:{val} {val} ... Stores the values in consecutive memory loca-

tions starting at {adrs}.

:{val} {val} ... Stores values in memory starting at the next

changeable location.

Moving and Comparing.

{dest} < {start}.{end}M Copies the values in the range {start}.{end} into

the range beginning at {dest}.

{dest} < {start}.{end}V Compares the values in the range {start}.{end}

to those in the range beginning at {dest}.

Saving and Loading via Tape.

{start}.{end}W Writes the values in the memory range

{start}.{end} onto tape, preceded by a ten-

second leader.

{start}.{end}R Reads values from tape, storing them in

memory beginning at {start} and stopping at

{end}. Prints "ERR" if an error occurs.

Running and Listing Programs.

{adrs}G Transfers control to the machine language pro-

gram beginning at {adrs}.

{adrs}L Disassembles and displays 20 instructions, start-

ing at {adrs}. Subsequent L's will display 20

more instructions each.

59

"A2M 030-0004-01 5-059.PICT" 237 KB 2001-07-23 dpi: 600h x 600v pix: 3040h x 4654v

#### Summary of Monitor Commands.

#### The Mini-Assembler

F666G Invoke the Mini-Assembler.\*

\${command} Execute a Monitor command from the Mini-

Assembler.

\$FF69G Leave the Mini-Assembler.

{adrs} S Disassemble, display, and execute the instruc-

tion at {adrs}, and display the contents of the 6502's internal registers. Subsequent S's will display and execute successive instructions.\*\*

{adrs} T Step infinitely. The TRACE command stops

only when it executes a BRK instruction or

when you press **RESET**.\*\*

CTRL E Display the contents of the 65\(\theta\)2's registers.

Miscellaneous.

I Set Inverse display mode.

N Set Normal display mode.

Enter the language currently installed in the

Apple's ROM.

Reenter the language currently installed in the

Apple's ROM.

 $\{val\}+\{val\}$  Add the two values and print the result.

{val}-{val} Subtract the second value from the first and

print the result.

{slot} CTRL P Divert output to the device whose interface

card is in slot number  $\{slot\}$ . If  $\{slot\} = \emptyset$ , then

route output to the Apple's screen.

{slot} CTRL K Accept input from the device whose interface

card is in slot number  $\{slot\}$ . If  $\{slot\} = \emptyset$ , then

accept input from the Apple's keyboard.

Jump to the machine language subroutine at

location \$3F8.

<sup>\*</sup> Not available in the Apple II Plus.

<sup>\*\*</sup> Not available in the Autostart ROM.

## SOME USEFUL MONITOR SUBROUTINES

Here is a list of some useful subroutines in the Apple's Monitor and Autostart ROMs. To use these subroutines from machine language programs, load the proper memory locations or 6502 registers as required by the subroutine and execute a JSR to the subroutine's starting address. It will perform the function and return with the 6502's registers set as described.

#### \$FDED COUT Output a character

COUT is the standard character output subroutine. The character to be output should be in the accumulator. COUT calls the current character output subroutine whose address is stored in CSW (locations \$36 and \$37), usually COUT1 (see below).

#### \$FDFØ COUT1 Output to screen

COUT1 displays the character in the accumulator on the Apple's screen at the current output cursor position and advances the output cursor. It places the character using the setting of the Normal/Inverse location. It handles the control characters RETURN, linefeed, and bell. It returns with all registers intact.

#### **SFE80 SETINV** Set Inverse mode

Sets Inverse video mode for COUT1. All output characters will be displayed as black dots on a white background. The Y register is set to \$3F, all others are unchanged.

#### \$FE84 SETNORM Set Normal mode

Sets Normal video mode for COUT1. All output characters will be displayed as white dots on a black background. The Y register is set to \$FF, all others are unchanged.

#### \$FD8E CROUT Generate a RETURN

CROUT sends a RETURN character to the current output device.

#### \$FD8B CROUT1 RETURN with clear

CROUT1 clears the screen from the current cursor position to the edge of the text window, then calls CROUT.

#### \$FDDA PRBYTE Print a hexadecimal byte

This subroutine outputs the contents of the accumulator in hexadecimal on the current output device. The contents of the accumulator are scrambled.

#### \$FDE3 PRHEX Print a hexadecimal digit

This subroutine outputs the lower nybble of the accumulator as a single hexadecimal digit. The contents of the accumulator are scrambled.

#### \$F941 PRNTAX Print A and X in hexadecimal

This outputs the contents of the A and X reisters as a four-digit hexadecimal value. The accumulator contains the first byte output, the X register contains the second. The contents of the

accumulator are usually scrambled.

#### \$F948 PRBLNK Print 3 spaces

Outputs three blank spaces to the standard output device. Upon exit, the accumulator usually contains  $A\emptyset$ , the X register contains  $\emptyset$ .

#### \$F94A PRBL2 Print many blank spaces

This subroutine outputs from 1 to 256 blanks to the standard output device. Upon entry, the X register should contain the number of blanks to be output. If X=\$00, then PRBL2 will output 256 blanks.

#### \$FF3A BELL Output a "bell" character

This subroutine sends a bell (CTRL G) character to the current output device. It leaves the accumulator holding \$87.

#### \$FBDD BELL1 Beep the Apple's speaker

This subroutine beeps the Apple's speaker for .1 second at 1KHz. It scrambles the A and X registers.

## \$FD\( C \) RDKEY Get an input character

This is the standard character input subroutine. It places a flashing input cursor on the screen at the position of the output cursor and jumps to the current input subroutine whose address is stored in KSW (locations \$38 and \$39), usually KEYIN (see below).

#### \$FD35 RDCHAR Get an input character or ESC code

RDCHAR is an alternate input subroutine which gets characters from the standard input, but also interprets the eleven escape codes (see page 34).

#### \$FD1B KEYIN Read the Apple's keyboard

This is the keyboard input subroutine. It reads the Apple's keyboard, waits for a keypress, and randomizes the random number seed (see page 32). When it gets a keypress, it removes the flashing cursor and returns with the keycode in the accumulator.

#### \$FD6A GETLN Get an input line with prompt

GETLN is the subroutine which gathers input lines (see page 33). Your programs can call GETLN with the proper prompt character in location \$33; GETLN will return with the input line in the input buffer (beginning at location \$200) and the X register holding the length of the input line.

#### \$FD67 GETLNZ Get an input line

GETLNZ is an alternate entry point for GETLN which issues a carriage return to the standard output before falling into GETLN (see above).

#### \$FD6F GETLN1 Get an input line, no prompt

GETLN1 is an alternate entry point for GETLN which does not issue a prompt before it gathers the input line. If, however, the user cancels the input line, either with too many backspaces or with a CTRL X, then GETLN1 will issue the contents of location \$33 as a prompt when it gets another line.

#### \$FCA8 WAIT Delay

This subroutine delays for a specific amount of time, then returns to the program which called it. The amount of delay is specified by the contents of the accumulator. With A the contents of the accumulator, the delay is  $\frac{1}{2}(26+27A+5A^2)$  µseconds. WAIT returns with the A register zeroed and the X and Y registers undisturbed.

#### \$F864 SETCOL Set Low-Res Graphics color

This subroutine sets the color used for plotting on the Low-Res screen to the color passed in the accumulator. See page 17 for a table of Low-Res colors.

#### \$F85F NEXTCOL Increment color by 3

This adds 3 to the current color used for Low-Res Graphics.

#### \$F800 PLOT Plot a block on the Low-Res screen

This subroutine plots a single block on the Low-Res screen of the prespecified color. The block's vertical position is passed in the accumulator, its horizontal position in the Y register. PLOT returns with the accumulator scrambled, but X and Y unmolested.

#### \$F819 HLINE Draw a horizontal line of blocks

This subroutine draws a horizontal line of blocks of the predetermined color on the Low-Res screen. You should call HLINE with the vertical coordinate of the line in the accumulator, the leftmost horizontal coordinate in the Y register, and the rightmost horizontal coordinate in location \$2C. HLINE returns with A and Y scrambled, X intact.

#### \$F828 VLINE Draw a vertical line of blocks

This subroutine draws a vertical line of blocks of the predetermined color on the Low-Res screen. You should call VLINE with the horizontal coordinate of the line in the Y register, the top vertical coordinate in the accumulator, and the bottom vertical coordinate in location \$2D. VLINE will return with the accumulator scrambled.

#### \$F832 CLRSCR Clear the entire Low-Res screen

CLRSCR clears the entire Low-resolution Graphics screen. If you call CLRSCR while the video display is in Text mode, it will fill the screen with inverse-mode "@" characters. CLRSCR destroys the contents of A and Y.

#### \$F836 CLRTOP Clear the top of the Low-Res screen

CLRTOP is the same as CLRSCR (above), except that it clears only the top 40 rows of the screen.

#### \$F871 SCRN Read the Low-Res screen

This subroutine returns the color of a single block on the Low-Res screen. Call it as you would call PLOT (above). The color of the block will be returned in the accumulator. No other registers are changed.

#### \$FB1E PREAD Read a Game Controller

PREAD will return a number which represents the position of a game controller. You should pass the number of the game controller (0 to 3) in the X register. If this number is not valid, strange things may happen. PREAD returns with a number from \$00 to \$FF in the Y register. The accumulator is scrambled.

#### \$FF2D PRERR Print "ERR"

Sends the word "ERR", followed by a bell character, to the standard output device. The accumulator is scrambled.

#### \$FF4A IOSAVE Save all registers

The contents of the 65\( \text{02}'\) s internal registers are saved in locations \$45 through \$49 in the order A-X-Y-P-S. The contents of A and X are changed; the decimal mode is cleared.

#### **\$FF3F** IOREST Restore all registers

The contents of the 65\( \text{02} \)'s internal registers are loaded from locations \$45 through \$49.

# MONITOR SPECIAL LOCATIONS

T	able 14:	Page Three Mon	itor Locations							
Address:		Use:								
Decimal	Hex	Monitor ROM	Autostart ROM							
1008	\$3FØ		Holds the address							
1009	\$3F1		of the subroutine							
	′	None.	which handles							
]		i vone.	machine language							
			"BRK" requests							
			(normally \$FA59).							
1010	\$3F2	None.	Soft Entry Vector.							
1011	\$3F3	rvone.	Soft Entry Vector.							
1012	\$3F4	None. Power-up Byte								
1013	\$3F5	Holds a "JuMP" instruction to t								
1014	\$3F6		h handles Applesoft II							
1015	\$3F7		.* Normally \$4C \$58							
		\$FF.								
1016	\$3F8		P" instruction to the							
1017	\$3F9	subroutine which								
1018	\$3FA	(CTRL Y) com	mands.							
1019	\$3FB	Holds a ''JuMI	e" instruction to the							
1020	\$3FC		ich handles Non-							
1021	\$3FD	Maskable Interru	ipts.							
1022	\$3FE	Holds the address of the subroutine								
1023	\$3FF	which handles In	iterrupt ReQuests.							

<sup>\*</sup> See page 123 in the Applesoft II BASIC Reference Manual.

## MINI-ASSEMBLER INSTRUCTION FORMATS

The Apple Mini-Assembler recognizes 56 mnemonics and 13 addressing formats used in 6502 Assembly language programming. The mnemonics are standard, as used in the MOS Technology/Synertek 6500 Programming Manual (Apple part number A2L0003), but the addressing formats are different. Here are the Apple standard address mode formats for 6502 Assembly Language:

Table 15: Mini-Ass	embler Address Formats
Mode:	Format:
Accumulator	None.
Immediate	#\${value}
Absolute	\${address}
Zero Page	\${address}
Indexed Zero Page	\${address},X
	\${address},Y
Indexed Absolute	\${address},X
	\${address},Y
Implied	None.
Relative	\${address}
Indexed Indirect	(\${address},X)
Indirect Indexed	(\${address}),Y
Absolute Indirect	(\${address})

An {address} consists of one or more hexadecimal digits. The Mini-Assembler interprets addresses in the same manner that the Monitor does: if an address has fewer than four digits, it adds leading zeroes; if it has more than four digits, then it uses only the last four.

All dollar signs (\$), signifying that the addresses are in hexadecimal notation, are ignored by the Mini-Assembler and may be omitted.

There is no syntactical distinction between the Absolute and Zero Page addressing modes. If you give an instruction to the Mini-Assembler which can be used in both Absolute and Zero-Page mode, then the Mini-Assembler will assemble that instruction in Absolute mode if the operand for that instruction is greater than \$FF, and it will assemble that instruction in Zero Page mode if the operand for that instruction is less than \$0100.

Instructions with the Accumulator and Implied addressing modes need no operand.

Branch instructions, which use the Relative addressing mode, require the *target address* of the branch. The Mini-Assembler will automatically figure out the relative distance to use in the instruction. If the target address is more than 127 locations distant from the instruction, then the Mini-Assembler wil sound a "beep", place a circumfex (^) under the target address, and ignore the line.

If you give the Mini-Assembler the mnemonic for an instruction and an operand, and the addressing mode of the operand cannot be used with the instruction you entered, then the Mini-Assembler will not accept the line.

# CHAPTER 4 MEMORY ORGANIZATION

- 68 RAM STORAGE
- 70 RAM CONFIGURATION BLOCKS
- 72 ROM STORAGE
- 73 I/O LOCATIONS
- 74 ZERO PAGE MEMORY MAPS

"A2M 030-0004-01 5-067.PICT" 14311 KB 2001-07-23 dpi: 600h x 600v pix: 3091h x 4785v

The Apple's 6502 microprocessor can directly reference a total of 65,536 distinct memory locations. You can think of the Apple's memory as a book with 256 "pages", with 256 memory locations on each page. For example, "page \$30" is the 256 memory locations beginning at location \$3000 and ending at location \$30FF. Since the 6502 uses two eight-bit bytes to form the address of any memory location, you can think of one of the bytes as the *page number* and the other as the *location within the page*.

The Apple's 256 pages of memory fall into three categories: Random Access Memory (RAM), Read-Only Memory (ROM), and Input/Output locations (I/O). Different areas of memory are dedicated to different functions. The Apple's basic memory map looks like this:

Sys	tem Me	emory Map
Page Num	ber:	
Decimal	Hex	
Ø	\$ØØ	
1	<b>\$Ø</b> 1	
2	<b>\$Ø</b> 2	
•		RAM (48K)
		K/KW (40K)
190	\$BE	
191	\$BF	
192	\$CØ	
193	\$C1	
		- (- ()
		I/O (2K)
198	\$C6	
199	\$C7	
200	\$C8	
201	\$C9	
•	•	LO DOM (NY)
	•	I/O ROM (2K)
206	\$СЕ	
200	\$CE	
207	\$DØ	
208	\$D0 \$D1	
207	וענ	
	• .	ROM (12K)
•	•	KOWI (121X)
254	\$FE	
255	\$FF	
L		

Figure 5. System Memory Map

## **RAM STORAGE**

The area in the Apple's memory map which is allocated for RAM memory begins at the bottom

of Page Zero and extends up to the end of Page 191. The Apple has the capacity to house from 4K (4,096 bytes) to 48K (49,152 bytes) of RAM on its main circuit board. In addition, you can expand the RAM memory of your Apple all the way up to 64K (65,536 bytes) by installing an Apple Language Card (part number A2B0006). This extra 16K of RAM takes the place of the Apple's ROM memory, with two 4K segments of RAM sharing the 4K range from \$D000 to \$DFFF.

Most of your Apple's RAM memory is available to you for the storage of programs and data. The Apple, however, does reserve some locations in RAM for use of the System Monitor, various languages, and other system functions. Here is a map of the available areas in RAM memory:

	7	able 16: RAM Organization and Usage
Page Num		Used For:
Decimal	Hex	
Ø	\$ØØ	System Programs
_1	\$Ø1	System Stack
2	\$02	GETLN Input Buffer
3	\$Ø3	Monitor Vector Locations
4	\$04	
5	<b>\$Ø</b> 5	Text and Lo-Res Graphics
6	<b>\$Ø</b> 6	Primary Page Storage
7	<b>\$Ø</b> 7	
8	\$Ø8	
9	\$09	Text and Lo-Res Graphics
10	\$ØA	Secondary Page Storage
11	\$ØB	
12	\$ØC	FREE
through	SWC	
31	\$1F	
	——————————————————————————————————————	RAM
32	\$2Ø	Hi-Res Graphics
through		Primary Page
63	\$3F	Storage
64	\$40	Hi-Res Graphics
through		Secondary Page
95	\$5F	Storage
96	\$60	
through		
191	\$BF	

Following is a breakdown of which ranges are assigned to which functions:

Zero Page. Due to the construction of the Apple's 65\( \text{0}2 \) microprocessor, the lowermost page in the Apple's memory is prime real estate for machine language programs. The System Monitor uses about 20 locations on Page Zero; Apple Integer BASIC uses a few more; and Applesoft II BASIC and the Apple Disk Operating System use the rest. Tables 18, 19, 20, and 21 show the locations on zero page which are used by these system functions.

Page One. The Apple's 6502 microprocessor reserves all 256 bytes of Page 1 for use as a "stack". Even though the Apple usually uses less than half of this page at any one time, it is not easy to determine just what is being used and what is lying fallow, so you shouldn't try to use

Page 1 to store any data.

Page Two. The GETLN subroutine, which is used to get input lines by most programs and languages, uses Page 2 as its input buffer. If you're sure that you won't be typing any long input lines, then you can (somewhat) safely store temporary data in the upper regions of Page 2.

Page Three. The Apple's Monitor ROM (both the Autostart and the original) use the upper sixteen locations in Page Three, from location \$3FØ to \$3FF (decimal 1008 to 1023). The Monitor's use of these locations is outlined on page 62.

**Pages Four through Seven**. This 1,024-byte range of memory locations is used for the Text and Low-Resolution Graphics Primary Page display, and is therefore unusable for storage purposes. There are 64 locations in this range which are not displayed on the screen. These 64 locations are reserved for use by the peripheral cards (see page 82).

## RAM CONFIGURATION BLOCKS

The Apple's RAM memory is composed of eight to 24 integrated circuits. These IC's reside in three rows of sockets on the Apple board. Each row can hold eight chips of either the 4,096-bit (4K) or 16,384-bit (16K) variety. The 4K RAM chips are of the Mostek "4096" family, and may be marked "MK4096" or "MCM6604". The 16K chips are of the "4116" type, and may have the denomination "MK4116" or "UPD4160". Each row must have eight of the same type of chip, although different rows may hold different types.

A row of eight 16K IC's represents 16,384 eight-bit bytes of RAM. The leftmost IC in a row represents the lowermost (least significant) bit of every byte in that range, and the rightmost IC in a row represents the uppermost (most significant) bit of every byte. The row of RAM IC's which is frontmost on the Apple board holds the RAM memory which begins at location  $\emptyset$  in the memory map; the next row back continues where the first left off.

You can tell the Apple how much memory it has, and of what type it is, by plugging *Memory Configuration Blocks* into three IC sockets on the left side of the Apple board. These configuration blocks are three 14-legged critters which look like big, boxy integrated circuits. But there are no chips inside of them; only three jumper wires in each. The jumper wires "strap" each row of RAM chips into a specific place in the Apple's memory map. All three configuration blocks should be strapped the same way. Apple supplies several types of standard configuration blocks for the most common system sizes. A set of these was installed in your Apple when it was built, and you get a new set each time you purchase additional memory for your Apple. If, however, you want to expand your Apple's memory with some RAM chips that you did not purchase from Apple, you may have to construct your own configuration blocks (or modify the ones already in your Apple).

There are nine different RAM memory configurations possible in your Apple. These nine memory sizes are made up from various combinations of 4K and 16K RAM chips in the three rows of sockets in your Apple. The nine memory configurations are:

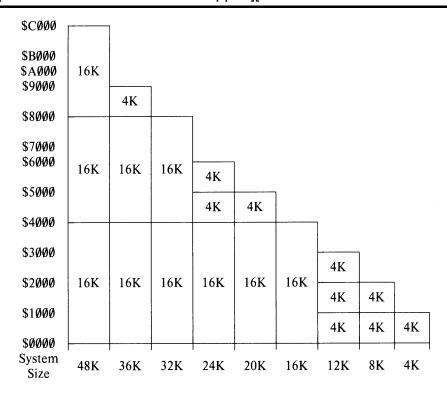


Figure 6. Memory Configurations

Of the fourteen "legs" on each controller block, the three in the upper-right corner (looking at it from above) represent the three rows of RAM chips on the Apple's main board. There should be a wire jumper from each one of these pins to another pin in the configuration block. The "other pin" corresponds to a place in the Apple's memory map where you want the RAM chips in each row to reside. The pins on the configuration block are represented thus:

4K range \$0000-\$0FFF	10	14	Frontmost row ("C")
4K range \$1000-\$1FFF	2	13	Middle row ("D")
4K range \$2000-\$2FFF	3	12	Backmost row ("E")
4K range \$3000-\$3FFF	4	11	No connection.
4K range \$4000-\$4FFF	5	10	16K range \$0000-\$3FFF
4K range \$5000-\$5FFF	6	9	16K range \$4000-\$7FFF
4K range \$8000-\$8FFF	7	8	16K range \$8000-\$BFFF
<del>-</del>			_

Figure 7. Memory Configuration Block Pinouts

If a row contains eight chips of the 16K variety, then you should connect a jumper wire from the pin corresponding to that row to a pin corresponding to a 16K range of memory. Similarly, if a row contains eight 4K chips, you should connect a jumper wire from the pin for that row to a pin corresponding to a 4K range of memory. You should *never* put 4K chips in a row strapped for 16K, or vice versa. It is also not advisable to leave a row unstrapped, or to strap two rows into the same range of memory.

You should always make sure that there is some kind of memory beginning at location  $\emptyset$ . Your Apple's memory should be in one contiguous block, but it does not need to be. For example, if you have only three sets of 4K chips, but you want to use the primary page of the High-

Resolution Graphics mode, then you would strap one row of 4K chips to the beginning of memory (4K range \$0000 through \$0FFF), and strap the other two rows to the memory range used by the High-Resolution Graphics primary page (4K ranges \$2000 through \$2FFF and \$3000 through \$3FFF). This will give you 4K bytes of RAM memory to work with, and 8K bytes of RAM to be used as a picture buffer.

Notice that the configuration blocks are installed into the Apple with their front edges (the edge with the white dot, representing pin 1) towards the front of the Apple.

There is a problem in Apples with Revision Ø boards and 20K or 24K of RAM. In these systems, the 8K range of the memory map from \$4000 to \$5FFF is duplicated in the memory range \$6000 to \$7FFF, regardless of whether it contains RAM or not. So systems with only 20K or 24K of RAM would appear to have 24K or 36K, but this extra RAM would be only imaginary. This has been changed in the Revision 1 Apple boards.

## ROM STORAGE

The Apple, in its natural state, can hold from 2K (2,048 bytes) to 12K (12,288 bytes) of Read-Only memory on its main board. This ROM memory can include the System Monitor, a couple of dialects of the BASIC language, various system and utility programs, or pre-packaged subroutines such as are included in Apple's *Programmer's Aid #1* ROM.

The Apple's ROM memory resides in the top 12K (48 pages) of the memory map, beginning at location \$D000. For proper operation of the Apple, there must be some kind of ROM in the upppermost locations of memory. When you turn on the Apple's power supply, the microprocessor must have some program to execute. It goes to the top locations in the memory map for the address of this program. In the Apple, this address is stored in ROM, and is the address of a program within the same ROM. This program initializes the Apple and lets you start to use it. (For a description of the startup cycle, see "The RESET Cycle", page 36.)

Here is a map of the Apple's ROM memory, and of the programs and packages that Apple currently supports in ROM:

	Table	17: ROM Organization	and Usage
Page Nui	mber:	Used By:	
Decimal	Hex	Osed by.	
208	\$DØ	Programmer's Aid #1	
212	\$D4	riogianinei s Aiu #1	
216	<b>\$D8</b>		
220	\$DC		Applesoft
224	\$EØ		II
228	\$E4		BASIC
232	\$E8	Integer BASIC	
236	\$EC		
240	\$FØ		
244	\$F4	Utility Subroutines	
248	\$F8	Monitor ROM	Autostart ROM
252	\$FC	IVIOLITIOF ROW	Autostatt KOM

Six 24-pin IC sockets on the Apple's board hold the ROM integrated circuits. Each socket can hold one of a type 9316B 2,048-byte by 8-bit Read-Only Memory. The leftmost ROM in the Apple's board holds the upper 2K of ROM in the Apple's memory map; the rightmost ROM IC holds the ROM memory beginning at page \$DØ in the memory map. If a ROM is not present in a given socket, then the values contained in the memory range corresponding to that socket will be unpredictable.

The Apple Firmware card can disable some or all of the ROMs on the Apple board, and substitute its own ROMs in their place. When you have an Apple Firmware card installed in any slot in the Apple's board, you can disable the Apple's on-board ROMs by flipping the card's controller switch to its UP position and pressing and releasing the RESET button, or by referencing location \$C080 (decimal 49280 or -16256). To enable the Apple's on-board ROMs again, flip the controller switch to the DOWN position and press RESET, or reference location \$C081 (decimal 49281 or -16255). For more information, see Appendix A of the Applesoft II BASIC Programming Reference Manual.

## I/O LOCATIONS

4,096 memory locations (16 pages) of the Apple's memory map are dedicated to input and output functions. This 4K range begins at location \$C000 (decimal 49152 or -16384) and extends on up to location \$CFFF (decimal 53247 or -12289). Since these functions are somewhat intricate, they have been given a chapter all to themselves. Please see Chapter 5 for information on the allocation of Input/Output locations.

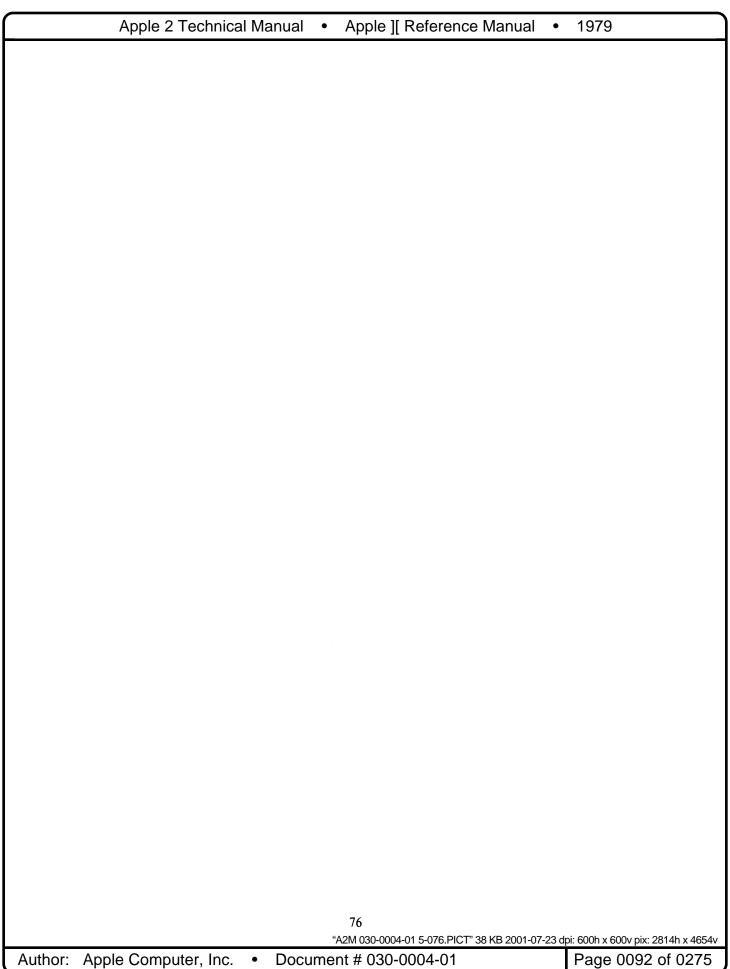
					Tab	le 18	: Me	onito	r Zer	o Pag	e Us	age					
Deci	mal	Ø	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Hex	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	<b>\$B</b>	\$C	\$D	\$E	\$F
Ø	\$ØØ															-	
16	<b>\$10</b>						-										
32	\$20	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
48	<b>\$3Ø</b>	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
64	\$40	•	•	•	•	•	•	•	•	•	•					•	•
8Ø	<b>\$50</b>	•	•	•	•	•	•										
96	\$6Ø																
112	\$7Ø																
128	\$8Ø																
144	\$9Ø																
16Ø	\$AØ																
176	\$BØ																
192	\$CØ																
2Ø8	\$DØ																
224	\$EØ																
240	\$FØ																

				Tabl	e 19:	App	lesof	t II l	BASI	C Ze	ro Pa	ge Us	age				
Deci	mal	Ø	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Hex	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	\$B -	\$C	\$D	\$E	\$F
Ø	\$ØØ	•	•	•	•	•	•					•	•	•	•	•	•
16	\$10	•	•	•	•	•	•	•	•	•							
32	\$20																
48	<b>\$3Ø</b>																
64	\$40																
8Ø	<b>\$50</b>	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
96	<b>\$60</b>	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
112	<b>\$7Ø</b>	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
128	\$8Ø	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
144	<b>\$9</b> Ø	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
16Ø	\$AØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
176	\$BØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
192	\$CØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•		
208	\$DØ	•	•	•	•	•	•			•	•	•	•	•	•	•	•
224	\$EØ	•	•	•		•	•	•	•	•	•	•					
240	\$FØ	•	•	•	•	•	•	•	•	•							

Apple 2 Technical Manual	•	Apple 1	If Reference Manual	•	1979
ADDIE Z I ECHILICAL MALIUAL	•	Apple 1	ii ivelelelle mallual	•	131

				Ta	able 2	20: A	pple	DOS	3.2	Zero	Page	Usag	e				
Deci	mal	Ø	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Hex	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	<b>\$</b> 9	\$A	\$B	\$C	\$D	\$E	\$F
Ø	\$ØØ																
16	\$10																
32	\$2Ø							•	•			•	•	•	•	•	•
48	<b>\$3Ø</b>						•	•	•	•	•					•	•
64	\$40	•	•	•	•	•	•	•	•	•		•	•	•	•		
8Ø	\$5Ø																
96	\$6Ø								•	•	•	•					•
112	\$7Ø	•															
128	\$8Ø																
144	<b>\$9</b> Ø																
16 <b>Ø</b>	\$AØ																•
176	\$BØ	•															
192	\$CØ											•	•	•	•		
2Ø8	\$DØ									•							
224	\$EØ																
240	\$FØ																

				T	able 2	21: I	ntege	r BA	SIC	Zero	Page	Usage	e				
Deci	mal	Ø	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	Hex	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	<b>\$B</b>	\$C	\$D	\$E	\$F
Ø	\$ØØ																
16	<b>\$10</b>																
32	\$20																
48	\$30																
64	\$40											•	•	•	•		
8Ø	<b>\$50</b>						•	•	•	•	•	•	•	•	•	•	•
96	\$60	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
112	<b>\$7Ø</b>	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
128	\$8Ø	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
144	<b>\$9</b> Ø	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
160	\$AØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
176	\$BØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
192	\$CØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
2Ø8	\$DØ	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•	•
224	\$EØ																
240	\$FØ																



# CHAPTER 5 INPUT/OUTPUT STRUCTURE

- 78 BUILT-IN I/O
- 79 PERIPHERAL BOARD I/O
- 80 PERIPHERAL CARD I/O SPACE
- 80 PERIPHERAL CARD ROM SPACE
- 81 I/O PROGRAMMING SUGGESTIONS
- 32 PERIPHERAL SLOT SCRATCHPAD RAM
- 83 THE CSW/KSW SWITCHES
- 84 EXPANSION ROM

77

"A2M 030-0004-01 5-077.PICT" 13988 KB 2001-07-23 dpi: 600h x 600v pix: 3055h x 4767v

The Apple's Input and Output functions fall into two basic categories: those functions which are performed on the Apple's board itself, and those functions which are performed by peripheral interface cards plugged into the Apple's eight peripheral "slots". Both of these functions communicate to the microprocessor and your programs via 4,096 locations in the Apple's memory map. This chapter describes the memory mapping and operation of the various input and output controls and functions; the hardware which executes these functions is described in the next chapter.

## **BUILT-IN I/O**

Most of the Apple's inherent I/O facilities are described briefly in Chapter 1, "Approaching your Apple". For a short description of these facilities, please see that chapter.

The Apple's on-board I/O functions are controlled by 128 memory locations in the Apple's memory map, beginning at location \$C000 and extending up through location \$C07F (decimal 49152 through 49279, or -16384 through -16257). Twenty-seven different functions share these 128 locations. Obviously, some functions are affected by more than one location: in some instances, as many as sixteen different locations all can perform exactly the same function. These 128 locations fall into five types: Data Inputs, Strobes, Soft Switches, Toggle Switches, and Flag Inputs.

**Data Inputs**. The only Data Input on the Apple board is a location whose value represents the current state of the Apple's built-in keyboard. The uppermost bit of this input is akin to the Flag Inputs (see below); the lower seven bits are the ASCII code of the key which was most recently pressed on the keyboard.

Flag Inputs. Most built-in input locations on the Apple are single-bit 'flags'. These flags appear in the highest (eighth) bit position in their respective memory locations. Flags have only two values: 'on' and 'off'. The setting of a flag can be tested easily from any language. A higher-level language can use a "PEEK" or similar command to read the value of a flag location: if the PEEKed value is greater than or equal to 128, then the flag is on; if the value is less than 128, the flag is off. Machine language programs can load the contents of a flag location into one of the 6502's internal registers (or use the BIT instruction) and branch depending upon the setting of the N (sign) flag. A BMI instruction will cause a branch if the flag is off.

The Single-Bit (Pushbutton) inputs, the Cassette input, the Keyboard Strobe, and the Game Controller inputs are all of this type.

Strobe Outputs. The Utility Strobe, the Clear Keyboard Strobe, and the Game Controller Strobe are all controlled by memory locations. If your program reads the contents of one of these locations, then the function associated with that location will be activated. In the case of the Utility Strobe, pin 5 on the Game I/O connector will drop from +5 volts to 0 volts for a period of .98 microseconds, then rise back to +5 again; in the case of the Keyboard Strobe, the Keyboard's flag input (see above) will be turned off; and in the case of the Game Controller Strobe, all of the flag inputs of the Game Controllers will be turned off and their timing loops restarted.

Your program can also trigger the Keyboard and Game Controller Strobes by writing to their controlling locations, but you should not write to the Utility Strobe location. If you do, you will produce two .98 microsecond pulses, about 24.43 nanoseconds apart. This is due to the method in which the 6502 writes to a memory location: first it reads the contents of that location, then it

writes over them. This double pulse will go unnoticed for the Keyboard and Game Controller Strobes, but may cause problems if it appears on the Utility Strobe.

**Toggle Switches**. Two other strobe outputs are connected internally to two-state "flip-flops". Each time you read from the location associated with the strobe, its flip-flop will "toggle" to its other state. These toggle switches drive the Cassette Output and the internal Speaker. There is no practical way to determine the setting of an internal toggle switch. Because of the nature of the toggle switches, you should only read from their controlling locations, and not write to them (see Strobe Outputs, above).

**Soft Switches**. Soft Switches are two-position switches in which each side of the switch is controlled by an individual memory location. If you reference the location for one side of the switch, it will throw the switch that way; if you reference the location for the other side, it will throw the switch the other way. It sets the switch without regard to its former setting, and there is no way to determine the position a soft switch is in. You can safely write to soft switch controlling locations: two pulses are as good as one (see Strobe Outputs, above). The Annunciator outputs and all of the Video mode selections are controlled by soft switches.

The special memory locations which control the built-in Input and Output functions are arranged thus:

				1	able	22:	Built-	ln I/O	Loca	ation	s					
	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	\$B	\$C	\$D	\$E	\$F
\$CØØØ	Key	board	d Data I	nput												
\$CØ1Ø	Cle	ar Ke	yboard	Strobe											-	
\$CØ2Ø	Cas	Cassette Output Toggle														
\$CØ3Ø	Spe	aker '	Toggle													
\$CØ4Ø	Util	ity St	robe													
\$CØ5Ø	gr	tx	nomix	mix	pri	sec	lores	hires	aı	nØ	ar	1	aı	n2	ar	13
\$CØ6Ø	cin	cin pb1 pb2 pb3 gc0 gc1 gc2 gc3 repeat \$C060-\$C067														
\$CØ7Ø	Gar	ne Co	ontroller	Strol	be				•							

#### Key to abbreviations:

gr	Set GRAPHICS mode	tx	Set TEXT mode
nomix	Set all text or graphics	mix	Mix text and graphics
pri	Display primary page	sec	Display secondary page
lores	Display Low-Res Graphics	hires	Display Hi-Res Graphics
an	Annunciator outputs	pb	Pushbutton inputs
gc	Game Controller inputs	cin	Cassette Input

## PERIPHERAL BOARD I/O

Along the back of the Apple's main board is a row of eight long "slots", or Peripheral Connectors. Into seven of these eight slots, you can plug any of many Peripheral Interface boards designed especially for the Apple. In order to make the peripheral cards simpler and more versatile, the Apple's circuitry has allocated a total of 280 byte locations in the memory map for each

of seven slots. There is also a 2K byte "common area", which all peripheral cards in your Apple can share.

Each slot on the board is individually numbered, with the leftmost slot called "Slot 0" and the rightmost called "Slot 7". Slot 0 is special: it is meant for RAM, ROM, or Interface expansion. All other slots (1 through 7) have special control lines going to them which are active at different times for different slots.

## PERIPHERAL CARD I/O SPACE

Each slot is given sixteen locations beginning at location \$C080 for general input and output purposes. For slot 0, these sixteen locations fall in the memory range \$C080 through \$C08F; for slot 1, they're in the range \$C090 through \$C09F, et cetera. Each peripheral card can use these locations as it pleases. Each peripheral card can determine when it is being selected by listening to pin 41 (called DEVICE SELECT) on its peripheral connector. Whenever the voltage on this pin drops to 0 volts, the address which the microprocessor is calling is somewhere in that peripheral card's 16-byte allocation. The peripheral card can then look at the bottom four address lines to determine which of its sixteen addresses is being called.

				Table	e 23:	Peripl	ieral C	ard I	O L	ocati	ons					
	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	\$B	\$C	\$D	\$E	\$F
\$CØ8Ø									1	Ø						
\$CØ9Ø									İ	1						
\$CØAØ									1	2						
\$CØBØ				Input	/Outpi	at for s	slot nu	mber	{	3						
\$CØCØ									1	4						
\$CØDØ									1	5						
\$CØEØ									ĺ	6						
\$CØFØ										7						

## PERIPHERAL CARD ROM SPACE

Each peripheral slot also has reserved for it one 256-byte page of memory. This page is usually used to house 256 bytes of ROM or Programmable ROM (PROM) memory, which contains driving programs or subroutines for the peripheral card. In this way, the peripheral interface cards can be "intelligent": they contain their own driving software; you do not need to load separate programs in order to use the interface cards.

The page of memory reserved for each peripheral slot has the page number Cn, where n is the slot number. Slot 0 does not have a page reserved for it, so you cannot use most Apple interface cards in that slot. The signal on Pin 1 (called  $\overline{I/O}$   $\overline{SELECT}$ ) of each peripheral slot will become active (drop from +5 volts to ground) when the microprocessor is referencing an address within that slot's reserved page. Peripheral cards can use this signal to enable their PROMs, and use the lower eight address lines to address each byte in the PROM.

	Table 24: Peripheral Card PROM Locations															
	\$ØØ	\$10	\$20	\$30	\$40	\$50	\$60	<b>\$70</b>	\$80	\$90	\$AØ	\$BØ	\$CØ	\$DØ	\$EØ	\$FØ
\$C100									1	1						
\$C200										2						
\$C300									- 1	3						
\$C400			PF	ROM:	space	for sl	ot nu	mber	-{	4						
\$C500										5						
\$C600										6						
\$C700									(	7						

## I/O PROGRAMMING SUGGESTIONS

The programs in peripheral card PROMs should be portable; that is, they should be able to function correctly regardless of where they are placed in the Apple's memory map. They should contain no absolute references to themselves. They should perform all JuMPs with conditional or forced branches.

Of course, you can fill a peripheral card PROM with subroutines which are *not* portable, and your only loss would be that the peripheral card would be slot-dependent. If you're cramped for space in a peripheral card PROM, you can save many bytes by making the subroutines slot-dependent.

The first thing that a subroutine in a peripheral card PROM should do is to save the values of *all* of the 6502's internal registers. There is a subroutine called IOSAVE in the Apple's Monitor ROM which does just this. It saves the contents of all internal registers in memory locations \$45 through \$49, in the order A-X-Y-P-S. This subroutine starts at location \$FF4A. A companion subroutine, called IORESTORE, restores *all* of the internal registers from these storage locations. You should call this subroutine, located at \$FF3F, before your PROM subroutine finishes.

Most single-character input and output is passed in the 6502's Accumulator. During output, the character to be displayed is in the Accumulator, with its high bit set. During input, your subroutine should pass the character received from the input device in the Accumulator, also with its high bit set.

A program in a peripheral card's PROM can determine which slot the card is plugged into by executing this sequence of instructions:

0300-	2 Ø	4 A	FF	JSR	\$FF4A
0303-	78			SEI	
0304-	2 Ø	58	FF	JSR	\$FF58
0307-	BA			TSX	
0308-	BD	ØØ	Ø 1	LDA	\$Ø1ØØ,X
Ø3ØB-	8D	F8	Ø 7	STA	<b>\$07F8</b>
Ø3ØE-	29	ØF		AND	#\$ØF
0310-	A8			TAY	

After a program executes these steps, the slot number which its card is in will be stored in the 6502's Y index register in the format \$0n, where n is the slot number. A program in the ROM can further process this value by shifting it four bits to the left, to obtain \$n0.

Ø311- 98 TYA

81

"A2M 030-0004-01 5-081.PICT" 403 KB 2001-07-23 dpi: 600h x 600v pix: 2994h x 4654v

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979
--------------------------	---	---------------------------	---	------

0312-	GL A	ACI
W 3 1 2 -	ØA	ASL
Ø313-	ØA	ASL
Ø314-	ØA	ASL
Ø315-	ØA	ASL
Ø316-	AA	TAX

A program can use this number in the X index register with the 65%2's indexed addressing mode to refer to the sixteen I/O locations reserved for each card. For example, the instruction

Ø317- BD 80 C0 LDA \$C080, X

will load the 6502's accumulator with the contents of the first I/O location used by the peripheral card. The address C080 is the *base address* for the first location used by all eight peripheral slots. The address C081 is the base address for the second I/O location, and so on. Here are the base addresses for all sixteen I/O locations on each card:

	Table 25: I/O Location Base Addresses										
Base				S	lot						
Address	Ø	1	2	3	4	5	6	7			
\$CØ8Ø	\$CØ8Ø	\$CØ9Ø	\$CØAØ	\$CØBØ	\$CØCØ	\$CØDØ	\$CØEØ	\$CØFØ			
\$CØ81	\$CØ81	\$CØ91	\$CØA1	\$CØB1	\$CØC1	\$CØD1	\$CØE1	\$CØF1			
\$CØ82	\$CØ82	\$CØ92	\$CØA2	\$CØB2	\$CØC2	\$CØD2	\$CØE2	\$CØF2			
\$CØ83	\$CØ83	\$CØ93	\$CØA3	\$CØB3	\$CØC3	\$CØD3	\$CØE3	\$CØF3			
\$CØ84	\$CØ84	\$CØ94	\$CØA4	\$CØB4	\$CØC4	\$CØD4	\$CØE4	\$CØF4			
\$CØ85	\$CØ85	\$CØ95	\$CØA5	\$CØB5	\$CØC5	\$CØD5	\$CØE5	\$CØF5			
\$CØ86	\$CØ86	\$CØ96	\$CØA6	\$CØB6	\$CØC6	\$CØD6	\$CØE6	\$CØF6			
\$CØ87	\$CØ87	\$CØ97	\$CØA7	\$CØB7	\$CØC7	\$CØD7	\$CØE7	\$CØF7			
\$CØ88	\$CØ88	\$CØ98	\$CØA8	\$CØB8	\$CØC8	\$CØD8	\$CØE8	\$CØF8			
\$CØ89	\$CØ89	\$CØ99	\$CØA9	\$CØB9	\$CØC9	\$CØD9	\$CØE9	\$CØF9			
\$CØ8A	\$CØ8A	\$CØ9A	\$CØAA	\$CØBA	\$CØCA	\$CØDA	\$CØEA	\$CØFA			
\$CØ8B	\$CØ8B	\$CØ9B	\$CØAB	\$CØBB	\$CØCB	\$CØDB	\$CØEB	\$CØFB			
\$CØ8C	\$CØ8C	\$CØ9C	\$CØAC	\$CØBC	\$CØCC	\$CØDC	\$CØEC	\$CØFC			
\$CØ8D	\$CØ8D	\$CØ9D	\$CØAD	\$CØBD	\$CØCD	\$CØDD	\$CØED	\$CØFD			
\$CØ8E	\$CØ8E	\$CØ9E	\$CØAE	\$CØBE	\$CØCE	\$CØDE	\$CØEE	\$CØFE			
\$CØ8F	\$CØ8F	\$CØ9F	\$CØAF	\$CØBF	\$CØCF	\$CØDF	\$CØEF	\$CØFF			
				I/O Lo	ocations						

# PERIPHERAL SLOT SCRATCHPAD RAM

Each of the eight peripheral slots has reserved for it 8 locations in the Apple's RAM memory. These 64 locations are actually in memory pages \$04 through \$07, inside the area reserved for the Text and Low-Resolution Graphics video display. The contents of these locations, however, are not displayed on the screen, and their contents are not changed by normal screen operations.\* The peripheral cards can use these locations for temporary storage of data while the cards are in operation. These "scratchpad" locations have the following addresses:

<sup>\*</sup> See "But Soft...", page 31.

	Table 26: I/O Scratchpad RAM Addresses								
Base			S	lot Numb	er				
Address	1	2	3	4	5	6	7		
\$Ø478	\$Ø479	\$Ø47A	\$Ø47B	\$047C	\$Ø47D	\$ <b>Ø</b> 47E	<b>\$</b> Ø47F		
<b>\$Ø4F8</b>	\$Ø4F9	<b>\$04FA</b>	\$Ø4FB	\$Ø4FC	\$Ø4FD	<b>\$04</b> FE	\$Ø4FF		
<b>\$Ø</b> 578	\$Ø579	\$Ø57A	\$Ø57B	\$Ø57C	\$Ø57D	<b>\$</b> Ø57E	<b>\$</b> Ø57F		
<b>\$0</b> 5F8	\$Ø5F9	<b>\$05FA</b>	\$Ø5FB	\$Ø5FC	\$Ø5FD	<b>\$05FE</b>	\$Ø5FF		
<b>\$Ø</b> 678	<b>\$Ø</b> 679	\$Ø67A	\$Ø67B	\$Ø67C	\$Ø67D	<b>\$0</b> 67E	<b>\$</b> Ø67F		
\$Ø6F8	\$Ø6F9	<b>\$0</b> 6FA	\$Ø6FB	<b>\$06FC</b>	\$Ø6FD	<b>\$06FE</b>	<b>\$0</b> 6FF		
<b>\$Ø</b> 778	<b>\$</b> 0779	\$Ø77A	\$Ø77B	\$Ø77C	\$Ø77D	<b>\$Ø77E</b>	<b>\$Ø77</b> F		
<b>\$Ø</b> 7F8	\$Ø7F9	<b>\$0</b> 7FA	\$Ø7FB	\$Ø7FC	\$Ø7FD	<b>\$07FE</b>	<b>\$</b> Ø7FF		

Slot Ø does not have any scratchpad RAM addresses reserved for it. The Base Address locations are used by Apple DOS 3.2 and are also shared by all peripheral cards. Some of these locations have dedicated functions: location \$7F8 holds the slot number (in the format \$Cn) of the peripheral card which is currently active, and location \$5F8 holds the slot number of the disk controller card from which any active DOS was booted.

By using the slot number \$0n, derived in the program example above, a subroutine can directly reference any of its eight scratchpad locations:

Ø31A-	В9	78	Ø 4	LDA	\$Ø478,Y
Ø31D-	99	F8	Ø 4	STA	\$Ø4F8,Y
0320-	В9	78	Ø 5	LDA	\$Ø578,Y
Ø323-	99	F8	Ø 5	STA	\$Ø5F8,Y
Ø326-	B9	78	Ø6	LDA	\$0678,Y
0329-	99	F8	Ø 6	STA	\$06F8,Y
Ø32C-	В9	78	Ø 7	LDA	\$Ø778,Y
Ø32F-	99	F8	Ø 7	STA	\$07F8.Y

## THE CSW/KSW SWITCHES

The pair of locations \$36 and \$37 (decimal 54 and 55) is called CSW, for "Character output SWitch". Individually, location \$36 is called CSWL (CSW Low) and location \$37 is called CSWH (CSW High). This pair of locations holds the address of the subroutine which the Apple is currently using for single-character output. This address is normally \$FDFØ, the address of the COUT subroutine (see page 30). The Monitor's PRINTER (CTRL P) command, and the BASIC command PR#, can change this address to be the address of a subroutine in a PROM on a peripheral card. Both of these commands put the address \$CnØØ into this pair of locations, where n is the slot number given in the command. This is the address of the first location in whatever PROM happens to be on the peripheral card plugged into that slot. The Apple will then call this subroutine every time it wishes to output one character. This subroutine can use the instruction sequences given above to find its slot number and use the I/O and RAM scratchpad locations for its slot. When it is finished, it can either execute an RTS (ReTurn from Subroutine) instruction, to return to the program or language which is sending the output, or it can jump to the COUT subroutine at location \$FDFØ, to display the character on the screen and then return to the program which is producing output.

Similarly, locations \$38 and 39 (decimal 56 and 57), called KSWL and KSWH separately or KSW

(Keyboard input SWitch) together, hold the address of the subroutine the Apple is currently using for single-character input. This address is normally \$FD1B, the address of the KEYIN subroutine. The Monitor's KEYBOARD command ( $\overline{CTRL K}$ ) and the BASIC command IN# both change this address to \$Cn00, again with n the slot number given in the command. The Apple will call the subroutine at the beginning of the PROM on the peripheral card in this slot whenever it wishes to get a single character from the input device. The subroutine should place the input character into the 6502's accumulator and ReTurn from Subroutine (RTS). The subroutine should set the high bit of the character before it returns.

The subroutines in a peripheral card's PROM can change the addresses in the CSW and KSW switches to point to places in the PROM other than the very beginning. For example, a certain PROM could begin with a segment of code to determine what slot it is in and do some initialization, and then jump in to the actual character handling subroutine. As part of its initialization sequence, it could change KSW or CSW (whichever is applicable) to point directly to the beginning of the character handling subroutine. Then the next time the Apple asks for input or output from that card, the handling subroutines will skip the already-done initialization sequence and go right in to the task at hand. This can save time in speed-sensitive situations.

A peripheral card can be used for both input and output if its PROM has seperate subroutines for the separate functions and changes CSW and KSW accordingly. The initialization sequence in a peripheral card PROM can determine if it is being called for input or output by looking at the high parts of the CSW and KSW switches. Whichever switch contains Cn is currently calling that card to perform its function. If both switches contain Cn, then your subroutine should assume that it is being called for output.

## **EXPANSION ROM**

The 2K memory range from location \$C800 to \$CFFF is reserved for a 2K ROM or PROM on a peripheral card, to hold large programs or driving subroutines. The expansion ROM space also has the advantage of being absolutely located in the Apple's memory map, which gives you more freedom in writing your interface programs.

This PROM space is available to all peripheral slots, and more than one card in your Apple can have an expansion ROM. However, only one expansion ROM can be active at one time.

Each peripheral card's expansion ROM should have a flip-flop to enable it. This flip-flop should be turned "on" by the  $\overline{DEVICE}$  SELECT signal (the one which enables the 256-byte PROM). This means that the expansion ROM on any card will be partially enabled after you first reference the card it is on. The other enable to the expansion ROM should be the  $\overline{I/O}$  STROBE line, pin 20 on each peripheral connector. This line becomes active whenever the Apple's microprocessor is referencing a location inside the expansion ROM's domain. When this line becomes active, and the aforementioned flip-flop has been turned "on", then the Apple is referencing the expansion ROM on this particular board (see figure 8).

A peripheral card's 256-byte PROM can gain sole access to the expansion ROM space by referring to location \$CFFF in its initialization subroutine. This location is a special location, and all peripheral cards should recognize it as a signal to turn their flip-flops "off" and disable their expansion ROMs. Of course, this will also disable the expansion ROM on the card which is trying to grab the ROM space, but the ROM will be enabled again when the microprocessor gets another instruction from the 256-byte driving PROM. Now the expansion ROM is enabled, and its space is clear. The driving subroutines can then jump directly into the programs in the ROM, where

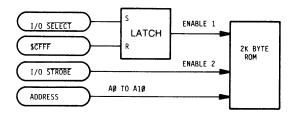


Figure 8. Expansion ROM Enable Circuit

they can enjoy the 2K of unobstructed, absolutely located memory space:

Ø332-Ø35-Ø60Ø70Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80Ø80</li

It is possible to save circuitry (at the expense of ROM space) on the peripheral card by not fully decoding the special location address, \$CFFF. In fact, if you can afford to lose the last 256 bytes of your ROM space, the following simple circuit will do just fine:

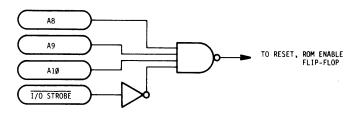
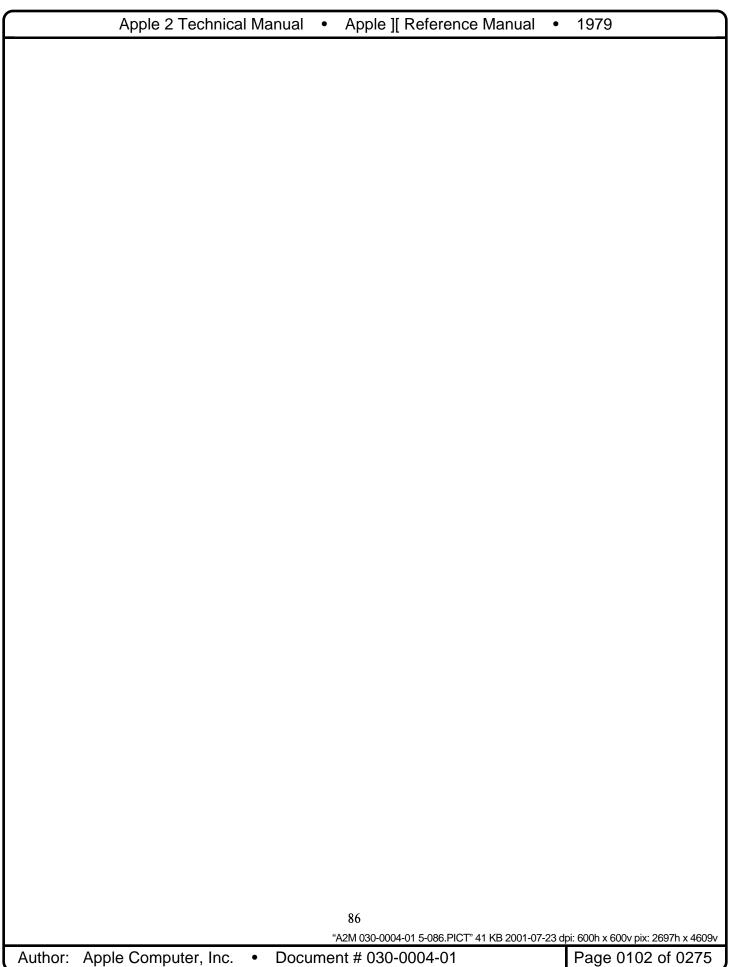


Figure 9. \$CFXX Decoding

85

"A2M 030-0004-01 5-085.PICT" 129 KB 2001-07-23 dpi: 600h x 600v pix: 3012h x 4230v



# CHAPTER 6 HARDWARE CONFIGURATION

- 88 THE MICROPROCESSOR
- 90 SYSTEM TIMING
- 92 POWER SUPPLY
- 94 ROM MEMORY
- 95 RAM MEMORY
- 96 THE VIDEO GENERATOR
- 97 VIDEO OUTPUT JACKS
- 98 BUILT-IN I/O
- 99 "USER 1" JUMPER
- 100 THE GAME I/O CONNECTOR
- 100 THE KEYBOARD
- 102 KEYBOARD CONNECTOR
- 103 CASSETTE INTERFACE JACKS
- 104 POWER CONNECTOR
- 105 SPEAKER
- 105 PERIPHERAL CONNECTORS

87

"A2M 030-0004-01 5-087. PICT" 13952 KB 2001-07-23 dpi: 600h x 600v pix: 3027h x 4785v

## THE MICROPROCESSOR

The 6502 Microprocessor

Model:

MCS6502/SY6502

Manufactured by:

MOS Technology, Inc.

Synertek

Rockwell

Number of instructions:

56

Addressing modes:

13

Accumulators:

1 (A)

Index registers:

2(X,Y)

Other registers:

Stack pointer (S)

Processor status (P)

Stack:

256 bytes, fixed

Status flags:

N (sign)

C (carry)
V (overflow)

I (Interrupt disable)

D (Decimal arithmetic)

B (Break)

Interrupts:

Other flags:

2 (IRQ, NMI)

Resets:

1 (RES)

Addressing range:

2<sup>16</sup> (64K) locations

Address bus:

16 bits, parallel

Data bus:

8 bits, parallel

**Bidirectional** 

Voltages:

+5 volts

Power dissipation:

.25 watt

Clock frequency:

1.023MHz

The microprocessor gets its main timing signals,  $\Phi\emptyset$  and  $\Phi1$ , from the timing circuits described below. These are complimentary 1.023MHz clock signals. Various manuals, including the MOS

88

"A2M 030-0004-01 5-088.PICT" 192 KB 2001-07-23 dpi: 600h x 600v pix: 2967h x 4654v

Figure 10. The Apple Main Board

89

"A2M 030-0004-01 5-089.PICT" 333 KB 2001-07-23 dpi: 600h x 600v pix: 3130h x 4609v

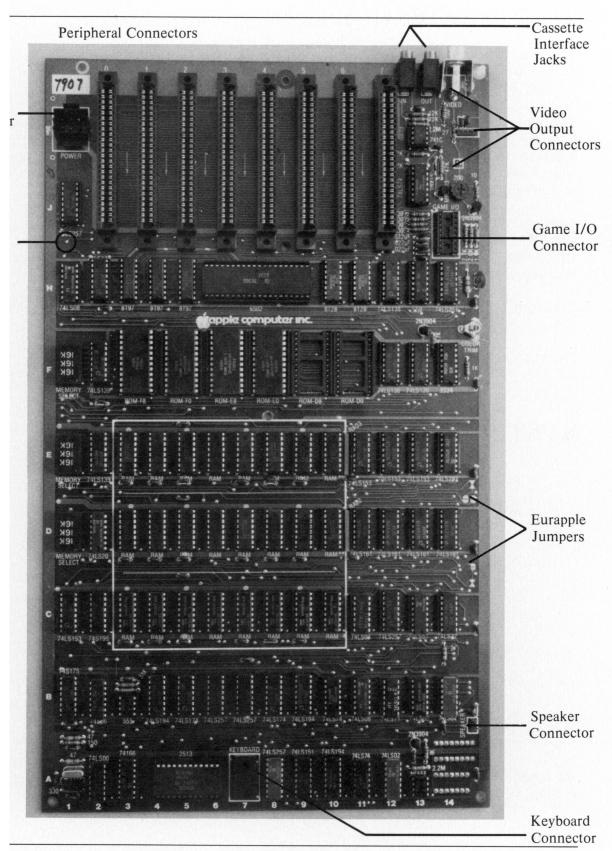


Figure 10. The Apple Main Board

"A2M 030-0004-01 5-089p.PICT" 19538 KB 2001-07-23 dpi: 800h x 800v pix: 3800h x 5534v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0106 of 0275

Technology Hardware manual, use the designation Φ2 for the Apple's ΦØ clock.

The microprocessor uses its address and data buses only during the time period when  $\Phi\emptyset$  is active. When  $\Phi\emptyset$  is low, the microprocessor is doing internal operations and does not need the data and address buses.

The microprocessor has a 16-bit address bus and an 8-bit bidirectional data bus. The Address bus lines are buffered by three 8T97 three-state buffers at board locations H3, H4, and H5. The address lines are held open only during a DMA cycle, and are active at all other times. The address on the address bus becomes valid about 300ns after  $\Phi$ 1 goes high and remains valid through all of  $\Phi$ 0.

The data bus is buffered through two 8T28 bidirectional three-state buffers at board locations H10 and H11. Data from the microprocessor is put onto the bus about 300ns after  $\Phi$ 1 and the READ/WRITE signal  $(R/\overline{W})$  both drop to zero. At all other times, the microprocessor is either listening to or ignoring the data bus.

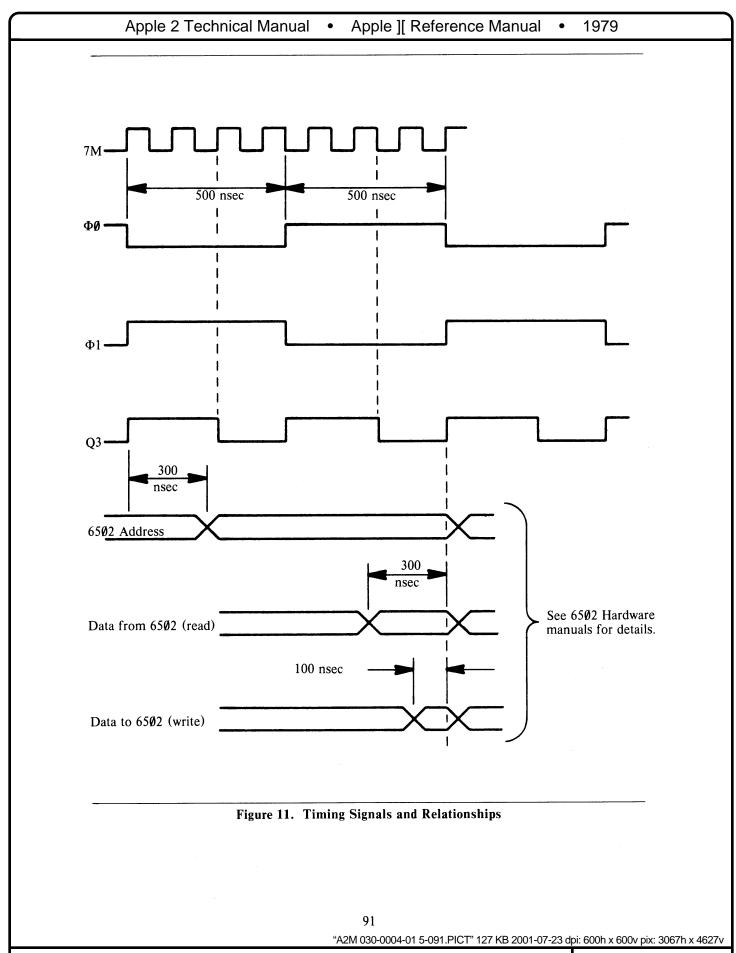
The RDY,  $\overline{RES}$ ,  $\overline{IRQ}$ , and  $\overline{NMI}$  lines to the microprocessor are all held high by 3.3K Ohm resistors to +5v. These lines also appear on the peripheral connectors (see page 105).

The SET OVERFLOW (SO) line to the microprocessor is permanently tied to ground.

## SYSTEM TIMING

	Table 27: Timing Signal Descriptions
14M:	Master Oscillator output, 14.318 MHz. All timing signals are derived from this signal.
7M:	Intermediate timing signal, 7.159 MHz.
COLOR REF:	Color reference frequency, 3.580MHz. Used by the video generation circuitry.
ФØ (Ф2):	Phase Ø system clock, 1.023MHz, compliment to Φ1.
Ф1:	Phase 1 system clock, 1.023 MHz, compliment to ΦØ.
Q3:	A general-purpose timing signal, twice the frequency of the system clocks, but asymmetrical.

All peripheral connectors get the timing signals 7M,  $\Phi\emptyset$ ,  $\Phi$ 1, and Q3. The timing signals 14M and COLOR REF are not available on the peripheral connectors.



#### POWER SUPPLY

The Apple Power Supply (U. S. Patent #4,130,862)

Input voltage:

107 VAC to 132 VAC, or

214 VAC to 264 VAC (switch selectable\*)

Supply voltages:

+5.0

+11.8

-12.0 -5.2

Power Consumption:

60 watts max. (full load)

79 watts max. (intermittent\*\*)

Full load power output:

+5v: 2.5 amp

-5v: 250ma

+12v: 1.5 amp ( $\sim$  2.5 amp intermittent\*\*)

-12v: 250ma

Operating temperature:

55c (131° Farenheit)

The Apple Power Supply is a high-voltage "switching" power supply. While most other power supplies use a large transformer with many windings to convert the input voltage into many lesser voltages and then rectify and regulate these lesser voltages, the Apple power supply first converts the AC line voltage into a DC voltage, and then uses this DC voltage to drive a high-frequency oscillator. The output of this oscillator is fed into a small transformer with many windings. The voltages on the secondary windings are then regulated to become the output voltages.

The +5 volt output voltage is compared to a reference voltage, and the difference error is fed back into the oscillator circuit. When the power supply's output starts to move out of its tolerances, the frequency of the oscillator is altered and the voltages return to their normal levels.

If by chance one of the output voltages of the power supply is short-circuited, a feedback circuit in the power supply stops the oscillator and cuts all output circuits. The power supply then pauses for about ½ second and then attempts to restart the oscillations. If the output is still shorted, it will stop and wait again. It will continue this cycle until the short circuit is removed or the power is turned off.

If the output connector of the power supply is disconnected from the Apple board, the power supply will notice this "no load" condition and effectively short-circuit itself. This activates the protection circuits described above, and cuts all power output. This prevents damage to the power supply's internals.

<sup>\*</sup> The voltage selector switch is not present on some Apples.

<sup>\*\*</sup> The power supply can run 20 minutes with an intermittent load if followed by 10 minutes at normal load without damage.

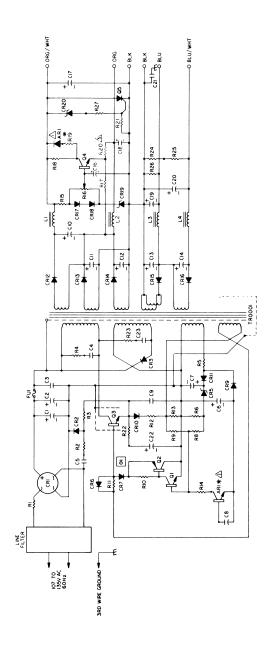


Figure 12. Power Supply Schematic Drawing

"A2M 030-0004-01 5-093.PICT" 417 KB 2001-07-23 dpi: 1200h x 1200v pix: 5976h x 9207v

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

If one of the output voltages leaves its tolerance range, due to any problem either within or external to the power supply, it will again shut itself down to prevent damage to the components on the Apple board. This insures that all voltages will either be correct and in proportion, or they will be shut off.

When one of the above fault conditions occurs, the internal protection circuits will stop the oscillations which drive the transformer. After a short while, the power supply will perform a restart cycle, and attempt to oscillate again. If the fault condition has not been removed, the supply will again shut down. This cycle can continue infinitely without damage to the power supply. Each time the oscillator shuts down and restarts, its frequency passes through the audible range and you can hear the power supply squeal and squeak. Thus, when a fault occurs, you will hear a steady "click click click" emanating from the power supply. This is your warning that something is wrong with one of the voltage outputs.

Under no circumstances should you apply more than 140 VAC to the input of the transformer (or more than 280 VAC when the supply's switch is in the 220V position). Permanent damage to the supply will result.

You should connect your Apple's power supply to a properly grounded 3-wire outlet. It is very important that the Apple be connected to a good earth ground.

CAUTION: There are dangerous high voltages inside the power supply's case. Much of the internal circuitry is *not* isolated from the power line, and special equipment is needed for service. **DO NOT ATTEMPT TO REPAIR YOUR POWER SUPPLY!** Send it to your Apple dealer for service.

#### **ROM MEMORY**

The Apple can support up to six 2K by 8 mask programmed Read-Only Memory ICs. One of these six ROMs is enabled by a 74LS138 at location F12 on the Apple's board whenever the microprocessor's address bus holds an address between \$D000 and \$FFFF. The eight Data outputs of all ROMs are connected to the microprocessor's data line buffers, and the ROM's address lines are connected to the buffers driving the microprocessor's address lines A0 through A10.

The ROMs have three "chip select" lines to enable them. CS1 and CS3, both active low, are connected together to the 74LS138 at location F12 which selects the individual ROMs. CS2, which is active high, is common to all ROMs and is connected to the INH (ROM Inhibit) line on the peripheral connectors. If a card in any peripheral slot pulls this line low, all ROMs on the Apple board will be disabled.

The ROMs are similar to type 2316 and 2716 programmable ROMs. However, the chip selects on most of these PROMs are of a different polarity, and they cannot be plugged directly into the Apple board.

Apple 2 Technical Manual • Apple If Reference Manual • 197	1979	•	e Manual	[ Reference	Apple 1	•	Manual	<b>Technical</b>	Apple 2
------------------------------------------------------------	------	---	----------	-------------	---------	---	--------	------------------	---------

<b>A</b> 7	1 0	24	+5v
<b>A6</b>	2	23	A8
<b>A</b> 5	3	22	A9
<b>A4</b>	4	21	CS3
<b>A3</b>	5	20	CS1
A2	6	19	A10
<b>A</b> 1	7	18	CS2
ΑØ	8	17	D7
DØ	9	16	D6
D1	10	15	D5
D2	11	14	D4
Gnd	12	13	D3

Figure 13. 9316B ROM Pinout.

#### RAM MEMORY

The Apple uses 4K and 16K dynamic RAMs for its main RAM storage. This RAM memory is used by both the microprocessor and the video display circuitry. The microprocessor and the video display interleave their use of RAM: the microprocessor reads from or writes to RAM only during  $\Phi\emptyset$ , and the video display refreshes its screen from RAM memory during  $\Phi1$ .

The three 74LS153s at E11, E12, and E13, the 74LS283 at E14, and half of the 74LS257 at C12 make up the address multiplexer for the RAM memory. They take the addresses generated by the microprocessor and the video generator and multiplex them onto six RAM address lines. The other RAM addressing signals,  $\overline{RAS}$  and  $\overline{CAS}$ , and the signal which is address line 6 for 16K RAMs and  $\overline{CS}$  for 4K RAMs, are generated by the RAM select circuit. This circuit is made up of two 74LS139s at E2 and F2, half of a 74LS153 at location C1, one and a half 74LS257s at C12 and J1, and the three Memory Configuration blocks at D1, E1, and F1. This circuit routes signals to each row of RAM, depending upon what type of RAM (4K or 16K) is in that row.

The dynamic RAMs are refreshed automatically during  $\Phi 1$  by the video generator circuitry. Since the video screen is always displaying at least a 1K range of memory, it needs to cycle through every location in that 1K range sixty times a second. It so happens that this action automatically refreshes every bit in all 48K bytes of RAM. This, in conjunction with the interleaving of the video and microprocessor access cycles, lets the video display, the microprocessor, and the RAM refresh run at full speed, without interfering with each other.

The data inputs to the RAMs are drawn directly off of the system's data bus. The data outputs of the RAMs are latched by two 74LS174s at board locations B5 and B8, and are multiplexed with the seven bits of data from the Apple's keyboard. These latched RAM outputs are fed directly to the video generator's character, color, and dot generators, and also back onto the system data bus by two 74LS257s at board locations B6 and B7.

Apple 2 Technical Manual	•	Apple 1	[ Reference Manual	•	1979

					_		
-5v	10	16	Gnd	-5v	10	16	Gnd
Data In	2	15	$\overline{CAS}$	Data In	2	15	$\overline{CAS}$
$R/\overline{W}$	3	14	Data Out	$R/\overline{W}$	3	14	Data Out
$\overline{RAS}$	4	13	$\overline{\text{CS}}$	$\overline{R}AS$	4	13	A6
A5	5	12	A2	A5	5	12	A2
A4	6	11	<b>A</b> 1	A4	6	11	A1
A3	7	10	ΑØ	A3	7	10	ΑØ
+12v	8	9	+5v	+12v	8	9	+5v
4096 4K RAM					4116 161	K RAM	
	Pinou	ıt			Pino	out	

Figure 14. RAM Pinouts

#### THE VIDEO GENERATOR

There are 192 scan lines on the video screen, grouped in 24 lines of eight scan lines each. Each scan line displays some or all of the contents of forty bytes of memory.

The video generation circuitry derives its synchronization and timing signals from a chain of 74LS161 counters at board locations D11 through D14. These counters generate fifteen synchronization signals:

The "H" family of signals is the horizontal byte position on the screen, from 000000 to binary 100111 (decimal 39). The signals V0 through V4 are the vertical line position on the screen, from binary 00000 to binary 10111 (decimal 23). The VA, VB, and VC signals are the vertical scan line position within the vertical screen line, from binary 000 to 111 (decimal 7).

These signals are sent to the RAM address multiplexer, which turns them into the address of a single RAM location, dependent upon the setting of the video display mode soft switches (see below). The RAM multiplexer then sends this address to the array of RAM memory during  $\Phi$ 1. The latches which hold the RAM data sent by the RAM array reroute it to the video generation circuit. The 74LS283 at location rearranges the memory addresses so that the memory mapping on the screen is scrambled.

If the current area on the screen is to be a text character, then the video generator will route the lower six bits of the data to a type 2513 character generator at location A5. The seven rows in each character are scanned by the VA, VB, and VC signals, and the output of the character generator is serialized into a stream of dots by a 74166 at location A3. This bit stream is routed to an exclusive-OR gate, where it is inverted if the high bit of the data byte is off and either the sixth bit is low or the 555 timer at location B3 is high. This produces inverse and flashing characters. The text bit stream is then sent to the video selector/multiplexer (below).

If the Apple's video screen is in a graphics mode, then the data from RAM is sent to two 74LS194 shift registers at board locations B4 and B9. Here each nybble is turned into a serial data stream. These two data streams are also sent to the video selector/multiplexer.

96

"A2M 030-0004-01 5-096.PICT" 414 KB 2001-07-23 dpi: 600h x 600v pix: 2994h x 4690v

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

The 74LS257 multiplexer at board position A8 selects between Color and High-Resolution graphics displays. The serialized Hi-res dot stream is delayed one-half clock cycle by the 74LS74 at location A11 if the high bit of the byte is set. This produces the alternate color set in High-Resolution graphics mode.

The video selector/multiplexer mixes the two data streams from the above sources according to the setting of the video screen soft switches. The 74LS194 at location A10 and the 74LS151 at A9 select one of the serial bit streams for text, color graphics, or high-resolution graphics depending upon the screen mode. The final serial output is mixed with the composite synchronization signal and the color burst signal generated by the video sync circuits, and sent to the video output connectors.

The video display soft switches, which control the video modes, are decoded as part of the Apple's on-board I/O functions. Logic gates in board locations B12, B13, B11, A12, and A11 are used to control the various video modes.

The color burst signal is created by logic gates at B12, B13, and C13 and is conditioned by R5, coil L1, C2, and trimmer capacitor C3. This trimmer capacitor can be tuned to vary the tint of colors produced by the video display. Transistor Q6 and its companion resistor R27 disable the color burst signal when the Apple is displaying text.

#### VIDEO OUTPUT JACKS

The video signal generated by the aforementioned circuitry is an NTSC compatible, similar to an EIA standard, positive composite video signal which can be fed to any standard closed-circuit or studio video monitor. This signal is available in three places on the Apple board:

**RCA Jack**. On the back of the Apple board, near the right edge, is a standard RCA phono jack. The sleeve of this jack is connected to the Apple's common ground and the tip is connected to the video output signal through a 200 Ohm potentiometer. This potentiometer can adjust the voltage on this connector from 0 to 1 volt peak.

**Auxiliary Video Connector**. On the right side of the Apple board near the back is a Molex KK100 series connector with four square pins, .25" tall, on .10" centers. This connector supplies the composite video output and two power supply voltages. This connector is illustrated in figure 15.

	Table 28: Auxiliary Video Output Connector Signal Descriptions					
Pin	Name	Description				
1	GROUND	System common ground; 0 volts.				
2	VIDEO	NTSC compatible positive composite video. Black level is about .75 volt, white level about 2.0 volt, sync tip level is 0 volts. Output level is not adjustable. This is not protected against short circuits.				
3	+12v	+12 volt power supply.				
4	-5v	-5 volt line from power supply.				

**Auxiliary Video Pin**. This single metal wire-wrap pin below the Auxiliary Video Output Connector supplies the same video signal available on that connector. It is meant to be a connection point for Eurapple PAL/SECAM encoder boards.

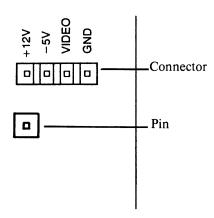


Figure 15. Auxiliary Video Output Connector and Pin.

#### **BUILT-IN I/O**

The Apple's built-in I/O functions are mapped into 128 memory locations beginning at \$C000. On the Apple board, a 74LS138 at location F13 called the I/O selector decodes these 128 special addresses and enables the various functions.

The 74LS138 is enabled by another '138 at location H12 whenever the Apple's address bus contains an address between \$C000 and \$C0FF. The I/O selector divides this 256-byte range into eight sixteen-byte ranges, ignoring the range \$C080 through \$C0FF. Each output line of the '138 becomes active (low) when its associated 16-byte range is being referenced.

The "O" line from the I/O selector gates the data from the keyboard connector into the RAM data multiplexer.

The "1" line from the I/O selector resets the 74LS74 flip-flop at B10, which is the keyboard flag,

The "2" line toggles one half of a 74LS74 at location K13. The output of this flip-flop is connected through a resistor network to the tip of the cassette output jack.

The "3" line toggles the other half of the 74LS74 at K13. The output of this flip-flop is connected through a capacitor and Darlington amplifier circuit to the Apple's speaker connector on the right edge of the board under the keyboard.

The "4" line is connected directly to pin 5 of the Game I/O connector. This pin is the utility  $\overline{C040}$   $\overline{STROBE}$ .

The "5" line is used to enable the 74LS259 at location F14. This IC contains the soft switches for the video display and the Game I/O connector annunciator outputs. The switches are selected

98

by the address lines 1 through 3 and the setting of each switch is controlled by address line Ø.

The "6" line is used to enable a 74LS251 eight-bit multiplexer at location H14. This multiplexer, when enabled, connects one of its eight input lines to the high order bit (bit 7) of the three-state system data bus. The bottom three address lines control which of the eight inputs the multiplexer chooses. Four of the mux's inputs come from a 553 quad timer at location H13. The inputs to this timer are the game controller pins on the Game I/O connector. Three other inputs to the multiplexer come from the single-bit (pushbutton) inputs on the Game I/O connector. The last multiplexer input comes from a 741 operational amplifier at location K13. The input to this op amp comes from the cassette input jack.

The "7" line from the I/O selector resets all four timers in the 553 quad timer at location H13. The four inputs to this timer come from an RC network made up of four  $0.022\mu$ F capacitors, four 100 Ohm resistors, and the variable resistors in the game controllers attached to the Game I/O connector. The total resistance in each of the four timing circuits determines the timing characteristics of that circuit.

#### "USER 1" JUMPER

There is an unlabeled pair of solder pads on the Apple board, to the left of slot  $\emptyset$ , called the "User 1" jumper. This jumper is illustrated in Photo 8. If you connect a wire between these two pads, then the USER 1 line on each peripheral connectors becomes active. If any peripheral card pulls this line low, *all* internal I/O decoding is disabled. The  $\overline{\text{I/O}}$  SELECT and the  $\overline{\text{DEVICE}}$  SELECT lines all go high and will remain high while USER 1 is low, regardless of the address on the address bus.

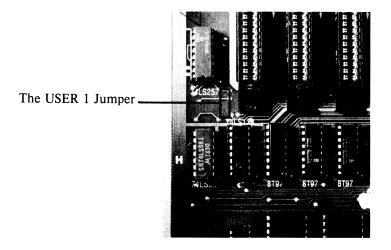


Photo 8. The USER 1 Jumper.

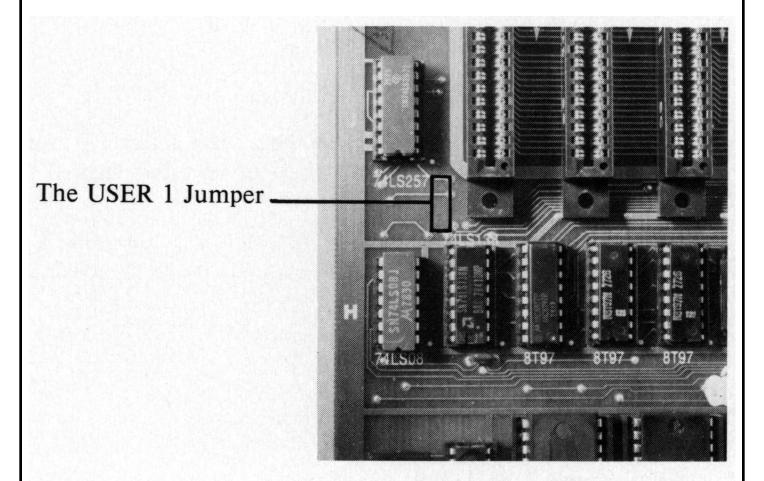


Photo 8. The USER 1 Jumper.

"A2M 030-0004-01 5-099p.PICT" 9970 KB 2001-07-23 dpi: 1200h x 1200v pix: 3843h x 2848v

#### THE GAME I/O CONNECTOR

+5v	10	16	NC
PB∅	2	15	ANØ
PB1	3	14	AN1
PB2	4	13	AN2
CØ4Ø STROBE	5	12	AN3
GCØ	6	11	GC3
GC2	7	10	GC1
Gnd	8	9	NC

Figure 16.
Game I/O Connector Pinouts

	Table 29: Game I/O Connector Signal Descriptions				
Pin:	Name:	Description:			
1	+5v	+5 volt power supply. Total current drain on this pin must be less than 100mA.			
2-4	PBØ-PB2	Single-bit (Pushbutton) inputs. These are standard 74LS series TTL inputs.			
5	CØ4Ø STROBE	A general-purpose strobe. This line, normally high, goes low during $\Phi\emptyset$ of a read or write cycle to any address from \$C040 through \$C04F. This is a standard 74LS TTL output.			
6,7,10,11	GCØ-GC3	Game controller inputs. These should each be connected through a 150K Ohm variable resistor to +5v.			
8	Gnd	System electrical ground.			
12-15	ANØ-AN3	Annunciator outputs. These are standard 74LS series TTL outputs and must be buffered if used to drive other than TTL inputs.			
9,16	NC	No internal connection.			

#### THE KEYBOARD

The Apple's built-in keyboard is built around a MM5740 monolithic keyboard decoder ROM. The inputs to this ROM, on pins 4 through 12 and 22 through 31, are connected to the matrix of keyswitches on the keyboard. The outputs of this ROM are buffered by a 7404 and are connected to the Apple's Keyboard Connector (see below).

The keyboard decoder rapidly scans through the array of keys on the keyboard, looking for one which is pressed. This scanning action is controlled by the free-running oscillator made up of three sections of a 7400 at keyboard location U4. The speed of this oscillation is controlled by C6, R6, and R7 on the keyboard's printed-circuit board.

100

"A2M 030-0004-01 5-100.PICT" 315 KB 2001-07-23 dpi: 600h x 600v pix: 3003h x 4654v

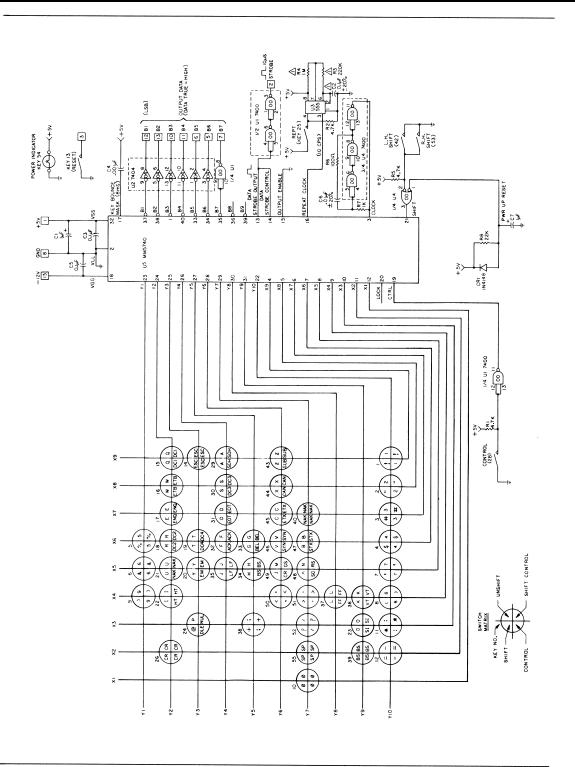


Figure 17. Schematic of the Apple Keyboard

"A2M 030-0004-01 5-101.PICT" 902 KB 2001-07-23 dpi: 1200h x 1200v pix: 5940h x 9226v

#### Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

The **REPT** key on the keyboard is connected to a 555 timer circuit at board location U3 on the keyboard. This chip and the capacitor and three resistors around it generate the 10Hz "REPeaT" signal. If the 220K Ohm resistor R3 is replaced with a resistor of a lower value, then the **REPT** key will repeat characters at a faster rate.

See Figure 17 for a schematic diagram of the Apple Keyboard.

#### **KEYBOARD CONNECTOR**

The data from the Apple's keyboard goes directly to the RAM data multiplexers and latches, the two 74LS257s at locations B6 and B7. The STROBE line on the keyboard connector sets a 74LS74 flip-flop at location B10. When the I/O selector activates its "Ø" line, the data which is on the seven inputs on the keyboard connector, and the state of the strobe flip-flop, are multiplexed onto the Apple's data bus.

	Table	30: Keyboard Connector Signal Descriptions
Pin:	Name:	Description:
1	+5v	+5 volt power supply. Total current drain on this pin must be less than 120mA.
2	STROBE	Strobe output from keyboard. This line should be given a pulse at least $10\mu s$ long each time a key is pressed on the keyboard. The strobe can be of either polarity.
3	RESET	Microprocessor's RESET line. Normally high, this line should be pulled low when the RESET button is pressed.
4,9,16	NC	No connection.
5-7, 10-13	Data	Seven bit ASCII keyboard data input.
8	Gnd	System electrical ground.
15	-12v	-12 volt power supply. Keyboard should draw less than 50mA.

+5v	10	16	NC
STROBE	2	15	-12v
RESET	3	14	NC
NC	4	13	Data 1
Data 5	5	12	Data Ø
Data 4	6	11	Data 3
Data 6	7	10	Data 2
Gnd	8	9	NC

Figure 18. Keyboard Connector Pinouts

#### **CASSETTE INTERFACE JACKS**

The two female miniature phone jacks on the back of the Apple II board can connect your Apple to a normal home cassette tape recorder.

Cassette Input Jack: This jack is designed to be connected to the "Earphone" or "Monitor" output jacks on most tape recorders. The input voltage should be 1 volt peak-to-peak (nominal). The input impedance is 12K Ohms.

Cassette Output Jack: This jack is designed to be connected to the "Microphone" input on most tape recorders. The output voltage is 25mv into a 100 Ohm impedance load.

103

## **POWER CONNECTOR**

This connector mates with the cable from the Apple Power Supply. This is an AMP #9-35028-1 six-pin male connector.

	Table 31: Power Connector Pin Descriptions						
Pin:	Name:	Description:					
1,2	Ground	Common electrical ground for Apple board.					
3	+5v	$+5.0$ volts from power supply. An Apple with 48K of RAM and no peripherals draws $\sim 1.5$ amp from this supply.					
4	+12v	+12.0 volts from power supply. An Apple with 48K of RAM and no peripherals draws ~400ma from this supply.					
5	-12v	$-12.0$ volts from power supply. An Apple with 48K of RAM and no peripherals draws $\sim 12.5$ ma from this supply.					
6	-5v	$-5.0$ volts from power supply. An Apple with 48K of RAM and no peripherals draws $\sim 0.0$ ma from this supply.					

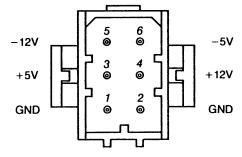


Figure 19. Power Connector

#### **SPEAKER**

The Apple's internal speaker is driven by half of a 74LS74 flip-flop through a Darlington amplifier circuit. The speaker connector is a Molex KK100 series connector, with two square pins, .25" tall, on .10" centers.

	Table 32: Speaker Connector Signal Descriptions							
Pin:	Name:	Description:						
1	SPKR	Speaker signal. This line will deliver about .5 watt into an 8 Ohm load.						
2	+5v	+5 volt power supply.						



Figure 20. Speaker Connector

## PERIPHERAL CONNECTORS

The eight peripheral connectors along the back edge of the Apple's board are Winchester #2HW25C0-111 50-pin PC card edge connectors with pins on .10" centers. The pinout for these connectors is given in Figure 21, and the signal descriptions are given on the following pages.

105

"A2M 030-0004-01 5-105.PICT" 167 KB 2001-07-23 dpi: 600h x 600v pix: 3013h x 4636v

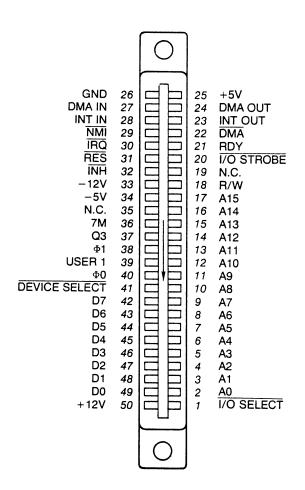


Figure 21. Peripheral Connector Pinout

	Table 33: Pe	ripheral Connector Signal Description
Pin:	Name:	Description:
1	I/O SELECT	This line, normally high, will become low when the microprocessor references page $Cn$ , where $n$ is the individual slot number. This signal becomes active during $D$ and will drive 10 LSTTL loads*. This signal is not present on peripheral connector $D$ .
2-17	AØ-A15	The buffered address bus. The address on these lines becomes valid during $\Phi 1$ and remains valid through $\Phi \emptyset$ . These lines will each drive 5 LSTTL loads*.
18	R/W	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 2 LSTTL loads*.
19	SYNC	On peripheral connector 7 <i>only</i> , this pin is connected to the video timing generator's SYNC signal.
20	I/O STROBE	This line goes low during $\Phi\emptyset$ when the address bus contains an address between \$C800 and \$CFFF. This line will drive 4 LSTTL loads*.
21	RDY	The 6502's RDY input. Pulling this line low during $\Phi$ 1 will halt the microprocessor, with the address bus holding the address of the current location being fetched.
22	DМА	Pulling this line low disables the 6502's address bus and halts the microprocessor. This line is held high by a $3K\Omega$ resistor to $+5v$ .
23	INT OUT	Daisy-chained interrupt output to lower priority devices. This pin is usually connected to pin 28 (INT IN).
24	DMA OUT	Daisy-chained DMA output to lower priority devices. This pin is usually connected to pin 22 (DMA IN).
25	+5v	+5 volt power supply. 500mA current is available for <i>all</i> peripheral cards.
26	GND	System electrical ground.

<sup>\*</sup> Loading limits are for each peripheral card.

	Table 33 (cont'd):	Peripheral Connector Signal Description
Pin:	Name:	Description:
27	DMA IN	Daisy-chained DMA input from higher priority devices. Usually connected to pin 24 (DMA OUT).
26	INT IN	Daisy-chained interrupt input from higher priority devices. Usually connected to pin 23 (INT OUT).
29	NMI	Non-Maskable Interrupt. When this line is pulled low the Apple begins an interrupt cycle and jumps to the interrupt handling routine at location \$3FB.
30	ĪRQ	Interrupt ReQuest. When this line is pulled low the Apple begins an interrupt cycle only if the 6502's I (Interrupt disable) flag is not set. If so, the 6502 will jump to the interrupt handling subroutine whose address is stored in locations \$3FE and \$3FF.
31	RES	When this line is pulled low the microprocessor begins a RESET cycle (see page 36).
32	ĪNĦ	When this line is pulled low, all ROMs on the Apple board are disabled. This line is held high by a $3K\Omega$ resistor to $+5v$ .
33	-12v	-12 volt power supply. Maxmum current is 200mA for all peripheral boards.
34	-5v	-5 volt power supply. Maximum current is 200mA for all peripheral boards.
35	COLOR REF	On peripheral connector 7 <i>only</i> , this pin is connected to the 3.5MHz COLOR REFerence signal of the video generator.
36	7M	7MHz clock. This line will drive 2 LSTTL loads*.
37	Q3	2MHz asymmetrical clock. This line will drive 2 LSTTL loads*.
38	Φ1	Microprocessor's phase one clock. This line will drive 2 LSTTL loads*.
39	USER 1	This line, when pulled low, disables all internal I/O address decoding**.

<sup>\*</sup> Loading limits are for each peripheral card.
\*\* See page 99.

	Table 33 (cont'd):	Peripheral Connector Signal Description
Pin:	Name:	Description:
40	ФØ	Microprocessor's phase zero clock. This line will drive 2 LSTTL loads*.
41	DEVICE SELECT	This line becomes active (low) on each peripheral connector when the address bus is holding an address between $C0n$ and $C0n$ , where $n$ is the slot number plus \$8. This line will drive 10 LSTTL loads*.
42-49	DØ-D7	Buffered bidirectional data bus. The data on this line becomes valid 300nS into $\Phi\emptyset$ on a write cycle, and should be stable no less than 100ns before the end of $\Phi\emptyset$ on a read cycle. Each data line can drive one LSTTL load.
50	+12v	+12 volt power supply. This can supply up to 250mA total for all peripheral cards.

<sup>\*</sup> Loading limits are for each peripheral card.

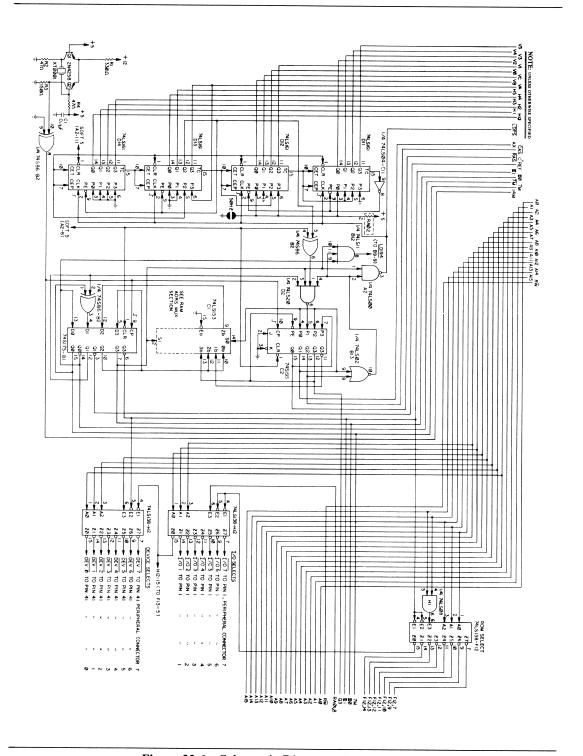


Figure 22-1. Schematic Diagram of the Apple II

"A2M 030-0004-01 5-110.PICT" 1270 KB 2001-07-23 dpi: 1200h x 1200v pix: 5940h x 9280v

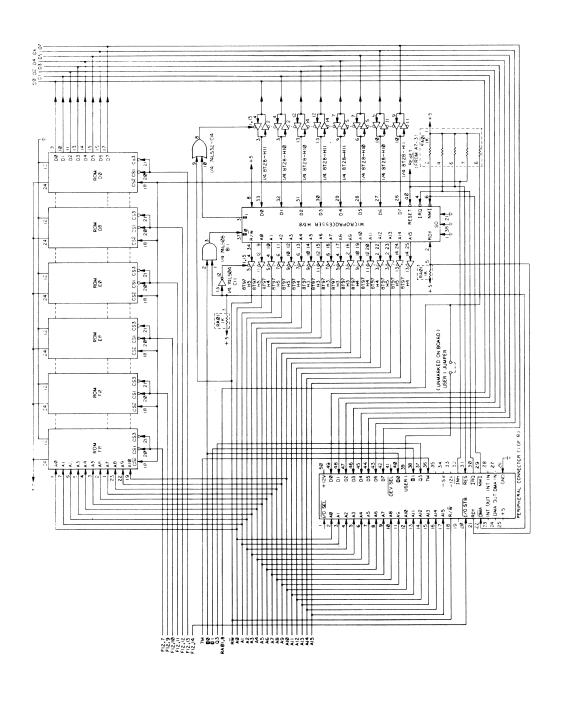


Figure 22-2. Schematic Diagram of the Apple II

"A2M 030-0004-01 5-111.PICT" 1287 KB 2001-07-23 dpi: 1200h x 1200v pix: 5939h x 9225v

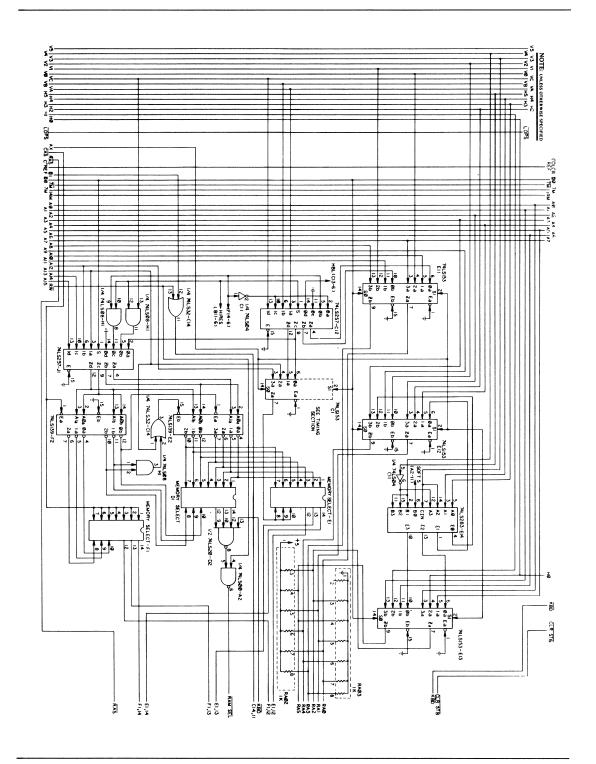


Figure 22-3. Schematic Diagram of the Apple II

"A2M 030-0004-01 5-112.PICT" 1240 KB 2001-07-23 dpi: 1200h x 1200v pix: 5976h x 9225v

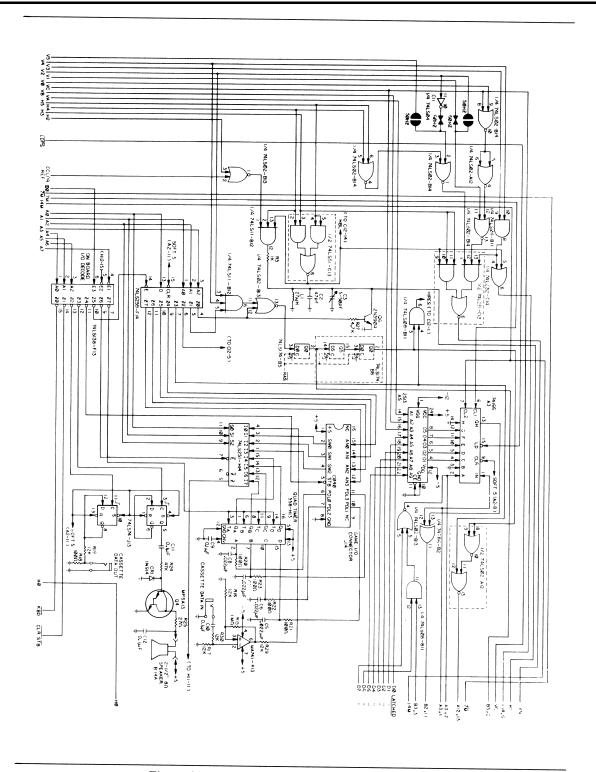


Figure 22-5. Schematic Diagram of the Apple II

"A2M 030-0004-01 5-114.PICT" 1215 KB 2001-07-23 dpi: 1200h x 1200v pix: 5940h x 9243v

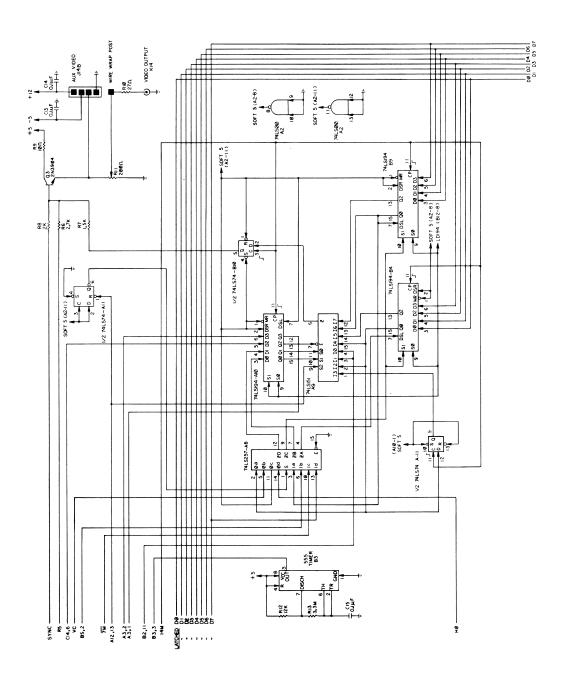
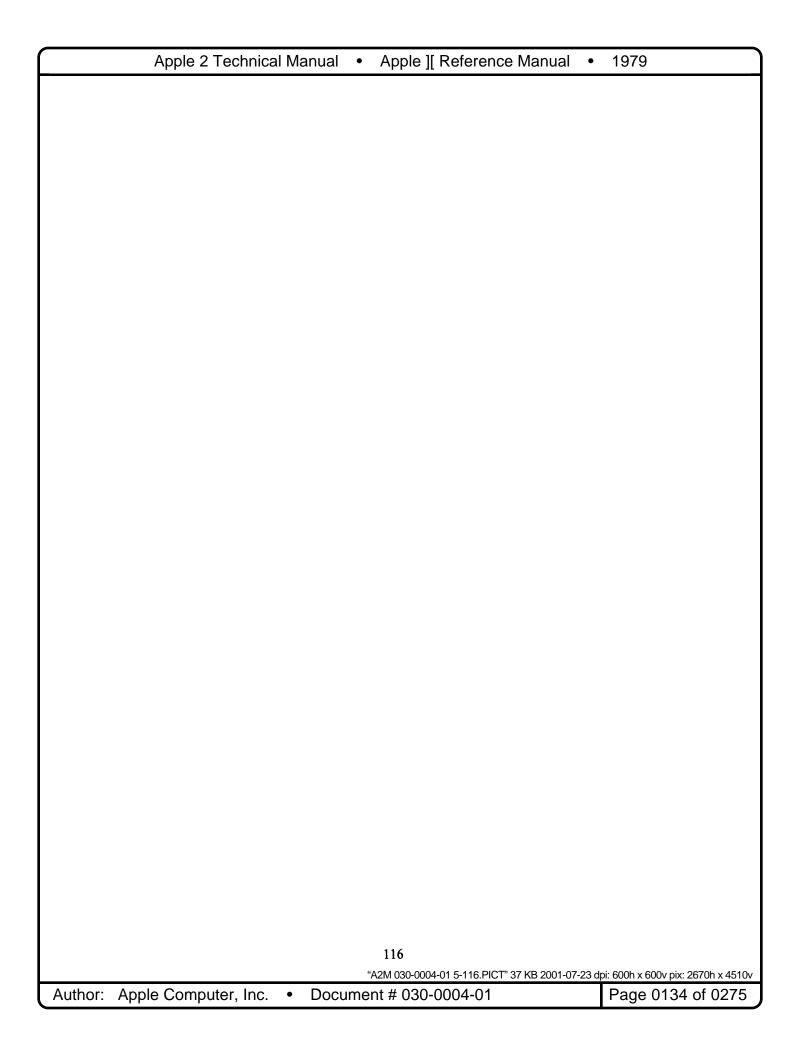


Figure 22-6. Schematic Diagram of the Apple II

"A2M 030-0004-01 5-115.PICT" 930 KB 2001-07-23 dpi: 1200h x 1200v pix: 5921h x 9226v



# APPENDIX A THE 6502 INSTRUCTION SET

117

"A2M 030-0004-01 5-117.PICT" 13804 KB 2001-07-23 dpi: 600h x 600v pix: 3100h x 4740v

# 6502 MICROPROCESSOR INSTRUCTIONS

ADC	Add Memory to Accumulator with	LDA	Load Accumulator with Memory
	Carry	LDX	Load Index X with Memory
AND	"AND" Memory with Accumulator	LDY	Load Index Y with Memory
ASL	Shift Left One Bit (Memory or	LSR	Shift Right one Bit (Memory or
	Accumulator)		Accumulator)
BCC	Branch on Carry Clear	NOP	No Operation
BCS	Branch on Carry Set	ORA	"OR" Memory with Accumulator
BEQ	Branch on Result Zero	• • • • • • • • • • • • • • • • • • • •	•
BIT	Test Bits in Memory with	PHA	Push Accumulator on Stack
	Accumulator	PHP	Push Processor Status on Stack
BMI	Branch on Result Minus	PLA	Pull Accumulator from Stack
BNE	Branch on Result not Zero	PLP	Pull Processor Status from Stack
BPL	Branch on Result Plus	ROL	Rotate One Bit Left (Memory or
BRK	Force Break		Accumulator)
BVC	Branch on Overflow Clear	ROR	Rotate One Bit Right (Memory or
BVS	Branch on Overflow Set		Accumulator)
CLC	Clear Carry Flag	RTI	Return from Interrupt
CLD	Clear Decimal Mode	RTS	Return from Subroutine
CLI	Clear Interrupt Disable Bit	SBC	Subtract Memory from Accumulato
CLV	Clear Overflow Flag		with Borrow
CMP	Compare Memory and Accumulator	SEC	Set Carry Flag
CPX	Compare Memory and Index X	SED	Set Decimal Mode
CPY	Compare Memory and Index Y	SEI	Set Interrupt Disable Status
DEC	Decrement Memory by One	STA	Store Accumulator in Memory
DEX	Decrement Index X by One	STX	Store Index X in Memory
DEY	Decrement Index Y by One	STY	Store Index Y in Memory
EOR	"Exclusive-Or" Memory with	TAX	Transfer Accumulator to Index X
	Accumulator	TAY	Transfer Accumulator to Index Y
INC	Increment Memory by One	TSX	Transfer Stack Pointer to Index X
INX	Increment Index X by One	TXA	Transfer Index X to Accumulator
INY	increment Index Y by One	TXS	Transfer Index X to Stack Pointer
		TYA	Transfer Index Y to Accumulator
JMP	Jump to New Location		
JSR	Jump to New Location Saving		

Return Address

# THE FOLLOWING NOTATION APPLIES TO THIS SUMMARY:

Accumulator Index Registers Memory Borrow **Processor Status Register** Stack Pointer Change No Change Add Logical AND Subtract Logical Exclusive Or Transfer From Stack Transfer To Stack Transfer To Transfer To Logical OR PC Program Counter PCH Program Counter High PCL Program Counter Low Immediate Addressing Mode FIGURE 1. ASL-SHIFT LEFT ONE BIT OPERATION

C 7 6 5 4 3 2 1 0 0

FIGURE 2. ROTATE ONE BIT LEFT (MEMORY OR ACCUMULATOR)

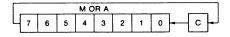
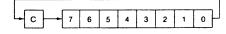


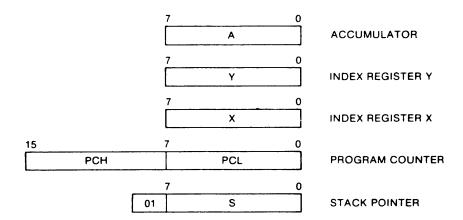
FIGURE 3.

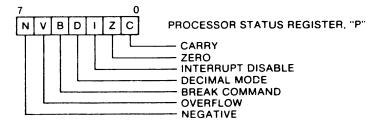


NOTE 1: BIT - TEST BITS

Bit 6 and 7 are transferred to the status register. If the result of A  $\Lambda$  M is zero then Z=1, otherwise Z=0.

#### PROGRAMMING MODEL





120

"A2M 030-0004-01 5-120.PICT" 76 KB 2001-07-23 dpi: 600h x 600v pix: 2182h x 3788v

#### **INSTRUCTION CODES**

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
ADC						
Add memory to accumulator with carry	A-M-CA.C	Immediate Zero Page Zero Page,X Absolute Absolute,X Absolute,Y (indirect,X) (Indirect),Y	ADC #Oper ADC Oper,X ADC Oper,X ADC Oper,X ADC Oper,X ADC Oper,X ADC (Oper,X) ADC (Oper,X)	69 65 75 60 70 79 61 71	2 2 2 3 3 3 2 2 2	<b>√√√</b> √
AND						
"AND" memory with accumulator	A A M A	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	AND #Oper AND Oper,X AND Oper,X AND Oper,X AND Oper,X AND Oper,Y AND (Oper,X) AND (Oper,X)	29 25 35 20 30 39 21 31	2 2 2 3 3 3 2 2	<b>V</b> V
ASL						
Shift left one bit (Memory or Accumulator)	(See Figure 1)	Accumulator Zero Page Zero Page.X Absolute Absolute.X	ASL A ASL Oper ASL Oper,X ASL Oper ASL Oper,X	0A 06 16 0E 1E	1 2 2 3 3	<b>VV</b>
BCC						
Branch on carry clear	Branch on C=0	Relative	BCC Oper	90	2	
BCS Branch on carry set	Branch on C=1	Relative	BCS Oper	В0	2	
BEQ						
Branch on result zero	Branch on Z=1	Relative	BEQ Oper	F0	2	
BIT Test bits in memory with accumulator	ΑΛΜ, M <sub>7</sub> N, M <sub>6</sub> V	Zero Page Absolute	BIT* Oper BIT* Oper	24 2C	2	M <sub>7</sub> √M <sub>6</sub>
BMI						
Branch on result minus	Branch on N=1	Relative	BMI Oper	30	2	
BNE						
Branch on result not zero	Branch on Z=0	Relative	BNE Oper	D0	2	
BPL						
Branch on result plus	Branch on N=0	Relative	BPL oper	10	2	
BRK Force Break	Forced Interrupt PC+2 † P †	Implied	BRK*	00	1	1
BVC						
Branch on overflow clear	Branch on V=0	Relative	BVC Oper	50	2	

Note 1 1 4 5 and 7 are transferred to the status register. If the result of A.V.M. is

Note 2. A BRK command cannot be masked by setting

121

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
BVS						
Branch on overflow set	Branch on V=1	Relative	BVS Oper	70	2	
CLC						
Clear carry flag	0 C	Implied	CLC	18	1	0
CLD						
Clear decimal mode	0 D	Implied	CLD	D8	1	-0
CLI						
	0 1	Implied	CLI	58	1	0
CLV						
Clear overflow flag	0 V	Implied	CLV	B8	1	0
CMP		·				
Compare memory and accumulator	A — M	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	CMP #Oper CMP Oper,X CMP Oper CMP Oper,X CMP Oper,X CMP (Oper,X) CMP (Oper,X)	C9 C5 D5 CD DD D9 C1	2 2 2 3 3 2 2	<b>VV</b>
CPX Compare memory and index X	X — M	Immediate Zero Page Absolute	CPX #Oper CPX Oper CPX Oper	E0 E4 EC	2 2 3	<b>VV</b>
CPY						
Compare memory and index Y	Y — M	Immediate Zero Page Absolute	CPY #Oper CPY Oper CPY Oper	C0 C4 CC	2 2 3	<b>\</b> \\
DEC						
Decrement memory by one	M — 1 → M	Zero Page Zero Page,X Absolute Absolute,X	DEC Oper DEC Oper,X DEC Oper DEC Oper,X	C6 D6 CE DE	2 2 3 3	<b>\</b> \
DEX						
Decrement index X by one	X — 1 → X	Implied	DEX	CA	1	<b>\</b> \
DEY						
Decrement index Y by one	Y — 1 → Y	Implied	DEY	88	1	<b>\</b> \

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
EOR						
"Exclusive-Or" memory with accumulator	A V M -+ A	Immediate Zero Page Zero Page, X Absolute Absolute, X Absolute, Y (Indirect, X) (Indirect), Y	EOR #Oper EOR Oper,X EOR Oper,X EOR Oper,X EOR Oper,Y EOR (Oper,X) EOR (Oper),Y	49 45 55 4D 5D 59 41 51	2 2 2 3 3 3 2 2	√√·
INC	1					
Increment memory by one	M + 1 - <b>→</b> M	Zero Page Zero Page,X Absolute Absolute,X	INC Oper INC Oper.X INC Oper INC Oper,X	E6 F6 EE FE	2 2 3 3	<b>\</b> \
INX						
Increment index X by one	X + 1 X	Implied	INX	E8	1	\\\
INY						
Increment index Y by one	Y + 1 → Y	Implied	INY	C8	1	<b>\</b> \
JMP						
Jump to new location	(PC+1) → PCL (PC+2) → PCH	Absolute Indirect	JMP Oper JMP (Oper)	4C 6C	3 3	
JSR						
Jump to new location saving return address	PC+2 ♥ . (PC+1) → PCL (PC+2) → PCH	Absolute	JSR Oper	20	3	
LDA						
Load accumulator with memory	M A	Immediate Zero Page Zero Page,X Absolute Absolute,X Absolute,Y (Indirect,X) (Indirect),Y	LDA #Oper LDA Oper, X LDA Oper, X LDA Oper, X LDA Oper, X LDA (Oper, X) LDA (Oper, X) LDA (Oper), Y	A9 A5 B5 AD BD B9 A1 B1	2 2 2 3 3 3 2 2	<b>\</b> \
LDX						
Load index X with memory	M →X	Immediate Zero Page Zero Page,Y Absolute Absolute,Y	LDX #Oper LDX Oper LDX Oper,Y LDX Oper LDX Oper,Y	A2 A6 B6 AE BE	2 2 2 3 3	<b>√√</b>
LDY Load index Y with memory	M Y	Immediate Zero Page Zero Page,X Absolute Absolute,X	LDY #Oper LDY Oper LDY Oper,X LDY Oper LDY Oper,X	A0 A4 B4 AC BC	2 2 2 3 3	√√

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
LSR Shift right one bit (memory or accumulator)	(See Figure 1)	Accumulator Zero Page	LSR A LSR Oper	4A 46	1 2	0 🗸
(memory or accumulator)		Zero Page,X Absolute Absolute,X	LSR Oper,X LSR Oper LSR Oper	56 4E 5E	2 3 3	
NOP					<b>†</b>	
No operation.	No Operation	Implied	NOP	EA	1	
ORA						
"OR" memory with accumulator	AVM -A	Immediate Zero Page Zero Page,X Absolute,X Absolute,Y (Indirect,X) (Indirect),Y	ORA #Oper ORA Oper ORA Oper,X ORA Oper ORA Oper,X ORA Oper,Y ORA (Oper,X) ORA (Oper,Y)	09 05 15 00 1D 19 01	2 2 2 3 3 3 2 2	√√
PHA						
Push accumulator on stack	A +	Implied	PHA	48	1	
PHP						
Push processor status on stack	P #	Implied	PHP	08	1	
PLA						
Pull accumulator from stack	At	Implied	PLA	68	1	<b>/</b> /
PLP						
Pull processor status from stack	P #	Implied	PLP	28	1	From Stack
ROL						
Rotate one bit left (memory or accumulator)	(See Figure 2)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	ROL A ROL Oper ROL Oper,X ROL Oper ROL Oper,X	2A 26 36 2E 3E	1 2 2 3 3	<b>VV</b>
ROR						
Rotate one bit right (memory or accumulator)	(See Figure 3)	Accumulator Zero Page Zero Page,X Absolute Absolute,X	ROR A ROR Oper ROR Oper,X ROR Oper ROR Oper,X	6A 66 76 6E 7E	1 2 2 3 3	<b>///</b>

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
RTI						
Return from interrupt	P∮PC∮	Implied	RTI	40	1	From Stack
RTS				60		
Return from subroutine	PC €, PC+1 → PC	Implied	RTS	00	1	
SBC Subtract memory from accumulator with borrow	A - M - Ĉ -←A	Immediate Zero Page Zero Page,X Absolute Absolute,X Absolute,Y (Indirect,X)	SBC #Oper SBC Oper SBC Oper,X SBC Oper,X SBC Oper,X SBC Oper,Y SBC (Oper,X)	E9 E5 F5 ED FD F9 E1	2 2 2 3 3 3 2	<b>√√√√</b>
		(Indirect).Y	SBC (Oper),Y	F1	2	
SEC						
Set carry flag	1 <b>→</b> C	Implied	SEC	38	1	1
SED						
Set decimal mode	1 - <b>→</b> D	Implied	SED	F8	1	1-
SEI Set interrupt disable status	11	Implied	SEI	78	1	1
STA Store accumulator in memory	A M	Zero Page Zero Page,X Absolute Absolute,X Absolute,Y (Indirect,X) (indirect),Y	STA Oper STA Oper,X STA Oper STA Oper,Y STA Oper,Y STA (Oper,X) STA (Oper),Y	85 95 8D 9D 99 81 91	2 2 3 3 3 2 2	
STX Store index X in memory	X M	Zero Page Zero Page,Y Absolute	STX Oper STX Oper,Y STX Oper	86 96 8E	2 2 3	
STY Store index Y in memory	Y M	Zero Page Zero Page,X Absolute	STY Oper STY Oper X STY Oper	84 94 8C	2 2 3	
TAX						
Transfer accumulator to index X	A X	Implied	TAX	AA	1	<b>/</b> /
TAY Transfer accumulator to index Y	AY	Implied	TAY	A8	1	<b>/</b> /
TSX Transfer stack pointer to index X	S X	Implied	TSX	ВА	1	<b>/</b> /

## Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

Name Description	Operation	Addressing Mode	Assembly Language Form	HEX OP Code	No. Bytes	"P" Status Reg. N Z C I D V
TXA						
Transfer index X to accumulator	X <del></del> A	Implied	TXA	8A	1	<b>V</b> V
TXS						
Transfer index X to stack pointer	X S	Implied	TXS	9A	1	
TYA						
Transfer index Y to accumulator	Y A	Implied	TYA	98	1	<b>/</b> /

## HEX OPERATION CODES

```
00 - BRK
                             2F - NOP
                                                           5E - LSR - Absolute, X
01 - ORA - (Indirect, X)
                                                          5F - NOP
                             30 - BMI
02 - NOP
                             31 - AND - (Indirect), Y
                                                          60 - RTS
03 - NOP
                             32 - NOP
                                                           61 - ADC - (Indirect, X)
04 - NOP
                             33 - NOP
                                                           62 - NOP
05 - ORA - Zero Page
                           34 — NOP
                                                          63 — NOP
06 - ASL - Zero Page
                             35 - AND - Zero Page, X 64 - NOP
36 - ROL - Zero Page, X 65 - ADC
07 - NOP
                             36 -- ROL -- Zero Page, X
                                                          65 - ADC - Zero Page
08 -- PHP
                             37 - NOP
                                                          66 - ROR - Zero Page
09 - ORA - Immediate
                             38 - SEC
                                                          67 - NOP
0A - ASL - Accumulator
                             39 - AND - Absolute, Y
                                                           68 — PLA
0B - NOP
                             3A - NOP
                                                          69 - ADC - Immediate
0C - NOP
                             3B - NOP
                                                          6A - ROR - Accumulator
0D — ORA — Absolute
0E — ASL — Absolute
                             3C - NOP
                                                          6B - NOP
                             3D — AND — Absolute, X
3E — ROL — Absolute, X
                                                          6C - JMP - Indirect
0F - NOP
                                                          6D - ADC - Absolute
10 - BPL
                             3F - NOP
                                                          6E - ROR - Absolute
11 - ORA - (Indirect), Y
                             40 — RTI
                                                           6F - NOP
12 — NOP
                             41 - EOR - (Indirect, X)
                                                          70 - BVS
13 - NOP
                             42 -- NOP
                                                          71 - ADC - (Indirect), Y
14 - NOP
                             43 - NOP
                                                          72 - NOP
15 — ORA — Zero Page, X
                             44 - NOP
                                                           73 - NOP
                             45 — EOR — Zero Page
16 — ASL — Zero Page, X
                                                          74 -- NOP
                             46 - LSR - Zero Page
17 - NOP
                                                         75 - ADC - Zero Page, X
                             47 - NOP
18 - CLC
                                                           76 - ROR - Zero Page, X
19 - ORA - Absolute, Y
                             48 - PHA
                                                          77 - NOP
1A - NOP
                             49 - EOR - Immediate
                                                         78 - SEI
18 - NOP
                             4A — LSR — Accumulator
                                                          79 - ADC - Absolute, Y
1C - NOP
                             4B - NOP
                                                          7A - NOP
1D - ORA - Absolute, X
                             4C - JMP - Absolute
                                                          7B — NOP
1E - ASL - Absolute, X
                             4D — EOR — Absolute
                                                          7C - NOP
1F - NOP
                             4E — LSR — Absolute
                                                          7D - ADC - Absolute, X NOP
20 - JSR
                             4F - NOP
                                                          7E - ROR - Absolute, X NOP
21 - AND - (Indirect, X)
                             50 - BVC
                                                          7F - NOP
22 - NOP
                             51 — EOR (Indirect), Y
                                                          80 - NOP
23 - NOP
                             52 - NOP
                                                          81 - STA - (Indirect, X)
24 - BIT - Zero Page
                             53 — NOP
                                                          82 - NOP
25 — AND — Zero Page
                             54 - NOP
                                                          83 - NOP
26 - ROL - Zero Page
                            55 — EOR — Zero Page, X
                                                          84 -STY - Zero Page
                             56 - LSR - Zero Page, X
27 - NOP
                                                          85 - STA - Zero Page
28 - PLP
                             57 - NOP
                                                          86 - STX - Zero Page
29 - AND - Immediate
                             58 - CLI
                                                          87 - NOP
2A - ROL - Accumulator
                           59 — EOR — Absolute, Y
                                                          88 - DEY
2B - NOP
                             5A - NOP
                                                          89 - NOP
2C - BIT - Absolute
                             5B - NOP
                                                           8A - TXA
2D - AND - Absolute
                             5C -- NOP
                                                          8B - NOP
2E - ROL - Absolute
                             5D — EOR — Absolute, X
                                                          8C - STY - Absolute
```

## Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

8D — STA — Absolute	B4 — LDY — Zero Page, X	DB - NOP
8E - STX - Absolute	B5 — LDA — Zero Page, X	DC - NOP
8F - NOP	B6 — LDX — Zero Page, Y	DD - CMP - Absolute, X
90 — BCC	B7 — NOP	DE — DEC — Absolute, X
91 - STA - (Indirect), Y	B8 — CLV	DF — NOP
92 — NOP	B9 - LDA - Absolute, Y	E0 — CPX — Immediate
93 — NOP	BA — TSX	E1 - SBC - (Indirect, X)
94 — STY — Zero Page, X	BB NOP	E2 - NOP
95 — STA — Zero Page, X	BC — LDY — Absolute, X	E3 — NOP
96 — STX — Zero Page, Y	BD — LDA — Absolute, X	E4 — CPX — Zero Page
97 — NOP	BE — LDX — Absolute, Y	E5 — SBC — Zero Page
98 — TYA	BF — NOP	E6 — INC — Zero Page
99 — STA — Absolute, Y	C0 — CPY — Immediate	E7 — NOP
9A — TXS	C1 - CMP - (Indirect, X)	E8 — INX
9B — NOP	C2 — NOP	E9 — SBC — Immediate
9C - NOP	C3 — NOP	EA NOP
9D - STA - Absolute, X	C4 — CPY — Zero Page	EB — NOP
9E - NOP	C5 — CMP — Zero Page	EC - CPX - Absolute
9F NOP	C6 — DEC — Zero Page	ED — SBC — Absolute
A0 - LDY - Immediate	C7 — NOP	EE - INC - Absolute
A1 — LDA — (Indirect, X)	C8 — INY	EF - NOP
A2 — LDX — Immediate	C9 — CMP — Immediate	F0 — BEQ
A3 — NOP	CA — DEX	F1 - SBC - (Indirect), Y
A4 — LDY — Zero Page	CB - NOP	F2 — NOP
A5 — LDA — Zero Page	CC — CPY — Absolute	F3 NOP
A6 — LDX — Zero Page	CD — CMP — Absolute	F4 — NOP
A7 - NOP	CE — DEC — Absolute	F5 - SBC - Zero Page, X
A8 TAY	CF — NOP	F6 — INC — Zero Page, X
A9 — LDA — Immediate	D0 — BNE	F7 - NOP
AA — TAX	D1 — CMP — (Indirect), Y	F8 - SED
AB - NOP	D2 — NOP	F9 - SBC - Absolute, Y
AC - LDY - Absolute	D3 — NOP	FA — NOP
AD — Absolute	D4 - NOP	FB — NOP
AE — LDX — Absolute	D5 — CMP — Zero Page, X	FC - NOP
AF NOP	D6 — DEC — Zero Page, X	FD — SBC — Absolute, X
BO - BCS	D7 — NOP	FE - INC - Absolute, X
B1 — LĎA — (Indirect), Y	D8 CLD	FF — NOP
B2 NOP	D9 — CMP — Absolute, Y	
B3 — NOP	DA — NOP	

# APPENDIX B SPECIAL LOCATIONS

129

"A2M 030-0004-01 5-129.PICT" 14156 KB 2001-07-23 dpi: 600h x 600v pix: 3073h x 4749v

Table 1: Keyboard Special Locations								
Location Hex		cimal	Description:					
\$CØØØ	49152	-16384	Keyboard Data					
\$CØ1Ø	49168	-16368	Clear Keyboard Strobe					

Table 4: Video Display Memory Ranges								
Screen	Page	Begins	Begins at:					
Serecii	1 age	Hex	Decimal	Hex	Decimal			
Text/Lo-Res	Primary	\$400	1024	\$7FF	2047			
	Secondary	\$800	2048	\$BFF	3Ø71			
Hi-Res	Primary	\$2000	8192	\$3FFF	16383			
	Secondary	\$4000	16384	\$5FFF	24575			

		Table 5:	Screen Soft Switches
Location	1:		Description
Hex	Decimal		Description:
\$CØ5Ø	49232	-16304	Display a GRAPHICS mode.
\$CØ51	49233	-163Ø3	Display TEXT mode.
\$CØ52	49234	-163Ø2	Display all TEXT or GRAPHICS.
\$CØ53	49235	-163Ø1	Mix TEXT and a GRAPHICS mode.
\$CØ54	49236	-16300	Display the Primary page (Page 1).
\$CØ55	49237	-16299	Display the Secondary page (Page 2).
\$CØ56	49238	-16298	Display LO-RES GRAPHICS mode.
\$CØ57	49239	-16297	Display HI-RES GRAPHICS mode.

Table	9: Ann	unciator	Special L	ocations
Ann.	State	Address	s:	
Aiiii.	State	Dec	cimal	Hex
Ø	off	49240	-16296	\$CØ58
	on	49241	-16295	\$CØ59
1	off	49242	-16294	\$CØ5A
	on	49243	-16293	\$CØ5B
2	off	49244	-16292	\$CØ5C
	on	49245	-16291	\$CØ5D
3	off	49246	-16290	\$CØ5E
	on	49247	-16289	\$CØ5F

Table 10: Input/Output Special Locations								
Function	Address: Dec	imal	Hex	Read/Write				
Speaker	49200	-16336	\$CØ3Ø	R				
Cassette Out Cassette In	49184 49256	-16352 -16288	\$CØ2Ø \$CØ6Ø	R R				
Annunciators	4924Ø through 49247	-16296 through -16289	\$CØ58 through \$CØ5F	R/W				
Flag inputs	49249 4925 <b>0</b> 49251	-16287 -16286 -16285	\$CØ61 \$CØ62 \$CØ63	R R R				
Analog Inputs	49252 49253 49254 49255	-16284 -16283 -16282 -16281	\$CØ64 \$CØ65 \$CØ66 \$CØ67	R				
Analog Clear	49264	-16272	\$CØ7Ø	R/W				
Utility Strobe	49216	-16320	\$CØ4Ø	R				

Table 11: Text Window Special Locations								
Function	Location:		Minimum/Normal/Maximum Va					
	Decimal	Hex	Decimal	Hex				
Left Edge	32	\$20	0/0/39	\$0/\$0/\$17				
Width	33	\$21	0/40/40	\$0/\$28/\$28				
Top Edge	34	\$22	0/0/24	\$0/\$0/\$18				
Bottom Edge	35	\$23	0/24/24	\$0/\$18/\$18				

Table 12: Normal/Inverse Control Values						
Value: Decimal	Hex	Effect:				
255	\$FF	COUT will display characters in Normal mode.				
63	\$3F	COUT will display characters in Inverse mode.				
127	\$7F	COUT will display letters in Flashing mode, all other characters in Inverse mode.				

Table 13: Autostart ROM Special Locations						
Location: Decimal	Hex	Contents:				
1010 1011	\$3F2 \$3F3	Soft Entry Vector. These two locations contain the address of the reentry point for whatever language is in use. Normally contains \$E003.				
1012	\$3F4	Power-Up Byte. Normally contains \$45.				
64367 (-1169)	\$FB6F	This is the beginning of a machine language subroutine which sets up the power-up location.				

	Table 14:	Page Three Mon	itor Locations
Address	S:	Use:	
Decima	l Hex	Monitor ROM	Autostart ROM
1008	\$3FØ		Holds the address
1009	\$3F1		of the subroutine
		None.	which handles
		TAOIIC.	machine language
			"BRK" requests
			(normaly \$FA59).
1010	\$3F2	None.	Soft Entry Vector.
1011	\$3F3	TVOIIC.	Soft Entry Vector.
1012	\$3F4	None.	Power-up byte.
1013	\$3F5	Holds a "JuMI	e" instruction to the
1014	\$3F6		h handles Applesoft II
1015	\$3F7		s. Normaly \$4C \$58
		\$FF.	
1016	\$3F8	Holds a "JuMI	e" instruction to the
1017	\$3F9	subroutine whi	ch handles "User"
1018	\$3FA	(CTRL Y) com	nands.
1019	\$3FB	Holds a "JuMI	" instruction to the
1020	\$3FC	subroutine wh	ich handles Non-
1021	\$3FD	Maskable Interru	ipts.
1022	\$3FE		ess of the subroutine
1023	\$3FF	which handles In	terrupt ReQuests.

				1	able	22:	Built-	n I/O	Loca	ation	S					
	\$Ø	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	\$B	\$C	\$D	\$E	\$F
\$C000	Key	board	l Data I	nput												
\$CØ1Ø	Cle	ar Ke	yboard :	Strobe	•											
\$CØ2Ø	Cas	sette	Output	Toggl	e										***	
\$CØ3Ø	Spe	aker [	Toggle													
\$CØ4Ø	Util	ity St	robe													
\$CØ5Ø	gr	tx	nomix	mix	pri	sec	lores	hires	ar	nØ	an	1	a	n2	ar	13
\$CØ6Ø	cin	pb1	pb2	pb3	gcØ	gc1	gc2	gc3	repeat \$C060-\$C067							
\$CØ7Ø	Gar	ne Co	ntroller	Strol	oe .									-		

### Key to abbreviations:

gr	Set GRAPHICS mode	tx	Set TEXT mode Mix text and graphics Display secondary page Display Hi-Res Graphics
nomix	Set all text or graphics	mix	
pri	Display primary page	sec	
lores	Display Low-Res Graphics	hires	
an gc	Annunciator outputs Game Controller inputs	pb cin	Pushbutton inputs Cassette Input

## Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

Table 23: Peripheral Card I/O Locations																
	\$0	\$1	\$2	\$3	\$4	\$5	\$6	\$7	\$8	\$9	\$A	\$B	\$C	\$D	\$E	\$F
\$CØ8Ø									1	Ø						
\$CØ9Ø										1						
\$CØAØ										2						
\$CØBØ				Input	/Outpi	it for s	slot nu	mber	$\downarrow$	3						
\$CØCØ				•	•				İ	4						
\$CØDØ									Ì	5						
\$CØEØ										6						
\$CØFØ									l	7						

Table 24: Peripheral Card PROM Locations																
	\$00	\$10	\$20	\$30	\$40	\$50	\$60	\$7Ø	\$80	\$90	\$AØ	\$BØ	\$CØ	\$DØ	\$EØ	\$FØ
\$C100									1	1						
\$C200									ļ	2						
\$C300									- 1	3						
\$C400			PF	ROM	space	for sl	ot nu	mber	- {	4						
\$C500										5						
\$C600									1	6						
\$C700									l	7						

	Table 25: I/O Location Base Addresses													
Base				S	lot									
Address	Ø	1	2	3	4	5	6	7						
\$CØ8Ø	\$CØ8Ø	\$CØ9Ø	\$CØAØ	\$CØBØ	\$CØCØ	\$CØDØ	\$CØEØ	\$CØFØ						
\$CØ81	\$CØ81	\$CØ91	\$CØA1	\$CØB1	\$CØC1	\$CØD1	\$CØE1	\$CØF1						
\$CØ82	\$CØ82	\$CØ92	\$CØA2	\$CØB2	\$CØC2	\$CØD2	\$CØE2	\$CØF2						
\$CØ83	\$CØ83	\$CØ93	\$CØA3	\$CØB3	\$CØC3	\$CØD3	\$CØE3	\$CØF3						
\$CØ84	\$CØ84	\$CØ94	\$CØA4	\$CØB4	\$CØC4	\$CØD4	\$CØE4	\$CØF4						
\$CØ85	\$CØ85	\$CØ95	\$CØA5	\$CØB5	\$CØC5	\$CØD5	\$CØE5	\$CØF5						
\$CØ86	\$CØ86	\$CØ96	\$CØA6	\$CØB6	\$CØC6	\$CØD6	\$CØE6	\$CØF6						
\$CØ87	\$CØ87	\$CØ97	\$CØA7	\$CØB7	\$CØC7	\$CØD7	\$CØE7	\$CØF7						
\$CØ88	\$CØ88	\$CØ98	\$CØA8	\$CØB8	\$CØC8	\$CØD8	\$CØE8	\$CØF8						
\$CØ89	\$CØ89	\$CØ99	\$CØA9	\$CØB9	\$CØC9	\$CØD9	\$CØE9	\$CØF9						
\$CØ8A	\$CØ8A	\$CØ9A	\$CØAA	\$CØBA	\$CØCA	\$CØDA	\$CØEA	\$CØFA						
\$CØ8B	\$CØ8B	\$CØ9B	\$CØAB	\$CØBB	\$CØCB	\$CØDB	\$CØEB	\$CØFB						
\$CØ8C	\$CØ8C	\$CØ9C	\$CØAC	\$CØBC	\$CØCC	\$CØDC	\$CØEC	\$CØFC						
\$CØ8D	\$CØ8D	\$CØ9D	\$CØAD	\$CØBD	\$CØCD	\$CØDD	\$CØED	\$CØFD						
\$CØ8E	\$CØ8E	\$CØ9E	\$CØAE	\$CØBE	\$CØCE	\$CØDE	\$CØEE	\$CØFE						
\$CØ8F	\$CØ8F	\$CØ9F	\$CØAF	\$CØBF	\$CØCF	\$CØDF	\$CØEF	\$CØFF						
				I/O Lo	ocations									

133

"A2M 030-0004-01 5-133.PICT" 355 KB 2001-07-23 dpi: 600h x 600v pix: 3021h x 4654v

## Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

	Table 26: I/O Scratchpad RAM Addresses												
Base		Slot Number											
Address	1	2	3	4	5	6	7						
\$Ø478	\$Ø479	\$Ø47A	\$Ø47B	\$Ø47C	\$Ø47D	\$ <b>0</b> 47E	\$047F						
\$Ø4F8	\$Ø4F9	\$Ø4FA	\$Ø4FB	\$Ø4FC	\$Ø4FD	\$04FE	\$04FF						
<b>\$</b> Ø578	\$Ø579	\$Ø57A	\$Ø57B	\$Ø57C	\$Ø57D	\$Ø57E	\$057F						
\$Ø5F8	\$Ø5F9	\$Ø5FA	\$Ø5FB	\$Ø5FC	\$Ø5FD	\$05FE	\$05FF						
\$Ø678	\$Ø679	\$Ø67A	\$Ø67B	\$Ø67C	\$Ø67D	\$067E	\$067F						
\$Ø6F8	\$Ø6F9	\$Ø6FA	\$Ø6FB	\$Ø6FC	\$06FD	\$06FE	\$06FF						
\$Ø778	\$Ø779	\$Ø77A	\$Ø77B	\$Ø77C	\$Ø77D	\$077E	\$Ø77F						
\$Ø7F8	<b>\$</b> 07F9	\$Ø7FA	\$Ø7FB	\$07FC	\$Ø7FD	\$07FE	\$Ø7FF						

# APPENDIX C ROM LISTINGS

136 AUTOSTART ROM LISTING

155 MONITOR ROM LISTING

135

"A2M 030-0004-01 5-135.PICT" 14155 KB 2001-07-23 dpi: 600h x 600v pix: 3073h x 4776v

# **AUTOSTART ROM LISTING**

```
0000:
                       *****
 0000
 0000:
                     4 * APPLE II
 0000:
                     5 * MONITOR II
0000:
                    6 * 7 * COPYRIGHT 1978 BY
0000:
0000:
                     8 * APPLE COMPUTER, INC.
0000:
0000:
                    10 * ALL RIGHTS RESERVED
                   11 *
0000:
                   12 * STEVE WOZNIAK
0000:
                   13 *
0000:
                   14 ***********
0000:
                   15 *
0000:
                   16 * MODIFIED NOV 1978
0000
                   17 * BY JOHN A
0000:
                   18 *
0000:
                   19 ***********
F800:
                   20
                              ORG $F800
F800:
                              OBJ $2000
                   21
F800:
                   22 ****************
F800:
                   23 LOCO EQU $00
F800:
                   24 LOC1
                              EGU $01
                   25 WNDLFT EQU $20
F800:
F800:
                   26 WNDWDTH EQU $21
F800:
                   27 WNDTOP
                              EQU $22
F800:
                   28 WNDBTM EQU $23
F800:
                   29 CH
                              EQU $24
F800:
                   30 CV
                              EQU $25
F800:
                   31 GBASL
                              EQU $26
F800:
                   32 GBASH
                              EQU $27
F800:
                   33 BASL
                              EQU $28
F800:
                   34 BASH
                              EQU $29
F800:
                   35 BAS2L
                              EQU $2A
FB00:
                   36 BAS2H
                              EQU $2B
F800:
                   37 H2
                              EQU $20
F800:
                   38 LMNEM
                              EQU $20
F800:
                   39 V2
                              EQU $2D
F800:
                   40 RMNEM
                              EQU $2D
FB00:
                   41 MASK
                              EQU $2E
F800:
                   42 CHKSUM
                              EQU $2E
F800:
                   43 FORMAT
                              EQU $2E
F800:
                   44 LASTIN
                              EQU $2F
FB00:
                   45 LENGTH
                              EQU $2F
FB00:
                   46 SIGN
                              EQU $2F
F800:
                   47 COLOR
                              EQU $30
F800:
                   48 MODE
                              EQU $31
                   49 INVFLG
                              EQU $32
F800:
                   50 PROMPT
                              EQU $33
F800:
                   51 YSAV
                              EQU $34
F800:
                   52 YSAV1
                              EQU $35
F800:
                   53 CSWL
                              EQU $36
F800:
                   54 CSWH
                              EQU $37
F800:
                   55 KSWL
                              EQU $38
F800:
                   56 KSWH
                              EQU $39
F800:
                   57 PCL
                              AE# UDB
F800:
                   58 PCH
                              EQU $3B
FB00:
                   59 A1L
                              EQU $30
F800
                   60 A1H
                              EQU $3D
F800
                  61 A2L
                              EQU $3F
F800:
                   62 A2H
                              EQU $3F
F800:
                  63 A3L
                              EQU $40
F800:
                  64 A3H
                              EQU $41
                  65 A4L
                              EQU $42
F800:
                  66 A4H
                              EQU $43
FB00:
                  67 A5L
                              EQU $44
F800:
                  68 A5H
                              EQU $45
```

### Apple 2 Technical Manual • Apple ][ Reference Manual 1979 F800 69 ACC EQU \$45 ; NOTE OVERLAP WITH A5H! F800 70 XREG EQU \$46 F800: 71 YREG EQU \$47 72 STATUS F800 EQU \$48 73 SPNT F800 EQU \$49 F800: 74 RNDL EQU \$4E F800 75 RNDH EQU \$4F 76 PICK F800 EQU \$95 F800 77. IN EGU \$0200 78 BRKV ; NEW VECTOR FOR BRK F800: EQU \$3F0 79 SOFTEV EQU \$3F2 VECTOR FOR WARM START F800: ; THIS MUST = EOR #\$A5 OF SOFTEV+1 F800: 80 PWREDUP EQU \$3F4 F800: 81 AMPERV EQU \$3F5 APPLESOFT & EXIT VECTOR F800: 82 USRADR EQU \$03F8 F800: IMN ES EQU \$03FB F800: 84 IRQLOC EQU \$3FE 85 LINE1 FB00: EQU \$400 FROO: 86 MSLOT EQU \$07F8 87 IOADR F800: EQU \$0000 F800: 88 KBD EQU \$C000 F800: 89 KBDSTRB EQU \$C010 F800: 90 TAPEOUT EQU \$CO20 F800: 91 SPKR EGU \$C030 92 TXTCLR F800: FQU \$0050 F800: 93 TXTSET EQU \$C051 F800 94 MIXCLR EQU \$C052 F800: 95 MIXSET EQU \$C053 F800: 96 LOWSCR EQU \$C054 F800: 97 HISCR EQU \$C055 98 LORES FB00: EQU \$0056 99 HIRES EQU \$C057 F800: F800: 100 SETANO EQU \$0058 F800: 101 CLRANO EQU \$C059 F800: 102 SETAN1 EQU \$CO5A F800: 103 CLRAN1 EQU \$CO5B 104 SETAN2 F800: EQU \$CO5C F800: 105 CLRAN2 EQU \$CO5D F800: 106 SETAN3 EQU \$COSE EQU \$CO5F FB00: 107 CLRAN3 F800: 108 TAPEIN EQU \$C060 F800: 109 PADDLO EQU \$C064 F800: 110 PTRIG EQU \$C070 F800: 111 CLRROM EQU \$CFFF FB00: 112 BASIC EQU \$E000 F800: 113 BASIC2 EQU \$E003 F800: 114 PAGE F800: 4A 115 PLOT LSR A FB01: 08 116 PHP F802: 20 47 FB 117 JSR GBASCALC F805: 28 118 PLP F806: A9 OF 119 LDA #\$OF F808: 90 02 120 BCC RTMASK F80A: 69 E0 121 ADC #\$EO FB0C: 85 2E 122 RTMASK STA MASK F80E: B1 26 123 PLOT1 LDA (GBASL), Y F810: 45 30 124 EOR COLOR F812: 25 2E 125 AND MASK FB14: 51 26 EDR (GBASL), Y 126 F816: 91 26 127 STA (GBASL), Y F818: 60 128 RTS F819: 20 00 F8 129 HLINE JSR PLOT F81C: C4 2C 130 HLINE1 CPY H2 F81E: BO 11 131 BCS RTS1 F820: CB 132 INY F821: 20 OE F8 JSR PLOT1 133 90 F6 F824: 134 BCC HLINE1 F826: 69 01 135 VLINEZ ADC #\$01 FB28: 48 136 VLINE PHA F829: 20 00 F8 137 JSR PLOT F82C: 68 138 PLA F82D: C5 2D 139 CMP V2 F82F: 90 F5 140 BCC VLINEZ 141 RTS1 FB31: 60 RTS

### Apple 2 Technical Manual • Apple | Reference Manual 1979 F832: A0 2F 142 CLRSCR LDY #\$2F F834: D0 02 143 BNE CLRSC2 F836: A0 27 144 CLRTOP LDY #\$27 F838: 84 2D 145 CLRSC2 STY V2 FB3A: A0 27 146 LDY #\$27 FB3C: A9 00 147 CLRSC3 LDA #\$00 F83E: 85 30 148 STA COLOR 20 28 F8 F840: 149 JSR VLINE F843: 88 150 DEY F844: 10 F6 151 BPL CLRSC3 F846: 60 152 153 RTS F847 PAGE F847: 48 154 GBASCALC PHA F848: 44 155 LSR A 29 03 F849 156 AND #\$03 09 04 F84B: 157 DRA #\$04 F84D: 85 27 158 STA GBASH F84F: 68 159 PLA F850: 29 18 160 AND #\$18 F852: 90 02 161 BCC GBCALC F854: 69 7F 162 ADC #\$7F 163 GBCALC STA GBASL F856: 85 26 F858 OA 164 ASL A F859: OA 165 ASL A 05 26 F85A: 166 ORA GBASL F850: 85 26 167 STA GBASL F85E: 60 168 RTS F85F: A5 30 169 LDA COLOR F861: 18 170 CLC F862: 69 03 171 ADC #\$03 29 OF F864: 172 SETCOL AND #\$OF F866: 85 30 173 STA COLOR F868: 174 ASL A F869: OΑ 175 ASI A 176 177 F86A OA ASL A F86B: OA ASL A 178 05 30 FB6C: ORA COLOR F86E: 85 30 179 STA COLOR F870: 60 180 RTS F871: 44 181 SCRN LSR A F872: 08 182 PHP F873: 20 47 F8 183 JSR GBASCALC F876: B1 26 184 LDA (GBASL), Y F878: 28 185 PLP F879: 90 04 186 SCRN2 BCC RTMSKZ F87B: 44 187 LSR A F87C 4Δ 188 LSR F87D: 44 189 LSR A F87E: 44 190 LSR A F87F: 29 OF 191 RTMSKZ AND #\$OF F881: 60 192 RTS F882: 193 PAGE F882: 194 INSDS1 A6 3A LDX PCL 195 A4 3B LDY PCH F884: 20 96 FD F886: 196 JSR PRYX2 JSR PRBLNK F889: 20 48 F9 197 F880: A1 3A 198 INSDS2 LDA (PCL, X) F88E: AB 199 TAY F88F: 44 200 LSR A F890: 90 09 201 ECC IEVEN F892 6A 202 ROR A F893: BO 10 203 BCS ERR F895: C9 A2 204 CMP #\$A2 F897: FO OC 205 BEG ERR F899: 29 87 206 AND #\$87 F89B: 44 207 IEVEN LSR A F890: AA 208 TAX F89D: BD 62 F9 209 LDA FMT1, X

JSR SCRN2

LDY #\$80

LDA #\$00

TAX

BNE GETFMT

20 79 FB

210

211

213

212 ERR

214 GETFMT

FBAO:

F8A3: D0 04

F8A5: A0 B0

FBA7: A9 00

FBA9: AA

### Apple 2 Technical Manual • Apple ][ Reference Manual 1979 LDA FMT2, X FBAA: BD A6 F9 215 STA FORMAT 85 2E 216 FBAD: 217 AND #\$03 FBAF: 29 03 STA LENGTH 218 FBB1: 85 2F 219 TYA F8B3: 98 AND #\$8F 220 F8B4: 29 BF TAX 221 FBB6: 222 TYA F8B7: **9**8 LDY #\$03 223 E0 0A FBB8: 224 CPX #\$BA EO BA FBBA: 225 BEG MNNDX3 FBBC: FO OB 226 MNNDX1 LSR A F8BE: 4A BCC MNNDX3 F8BF: 90 08 227 F801: 44 228 LSR A 229 MNNDX2 LSR A 44 FBC2: 09 20 230 ORA #\$20 FBC3: DEY 231 F8C5: 88 BNE MNNDX2 F8C6: DO FA 232 INY FBC8: C8 233 234 MNNDX3 FBC9: 88 DEY BNE MNNDX1 235 FBCA: DO F2 FBCC: 60 236 RTS FBCD: FF FF FF DFB \$FF, \$FF, \$FF 237 238 PAGE FRDO: 239 INSTOSP JSR INSDS1 20 82 F8 FBD0: 240 PHA FBD3: 48 241 PRNTOP LDA (PCL), Y FBD4: B1 3A JSR PRBYTE F806: 20 DA FD 242 243 LDX #\$01 FBD9: A2 01 20 4A F9 244 PRNTBL JSR PRBL2 FRDB: CPY LENGTH 245 C4 2F FBDE: INY F8E0: CB 246 BCC PRNTOP F8E1: 90 F1 247 248 LDX #\$03 F8E3: A2 03 249 CPY #\$04 CO 04 F8E5: 250 BCC PRNTBL 90 F2 FBF7: 251 PLA F8E9: 68 TAY 252 FBEA: A8 LDA MNEML, Y FBEB: B9 CO F9 253 STA LMNEM 85 254 FBEE: FBF0: B9 00 FA 255 LDA MNEMR, Y 85 2D 256 STA RMNEM FBF3: 257 NXTCOL LDA #\$00 A9 00 F8F5: LDY ##05 258 FBF7: A0 05 259 PRMN2 ASL RMNEM F8F9: 06 2D ROL LMNEM F8FB: 26 2C 260 FBFD: 261 ROL A DEY FBFE: 88 262 BNE PRMN2 DO FB 263 FRFF: ADC #\$BF 69 BF F901: 264 20 ED FD JSR COUT F903: 265 DEX F906: CA 266 BNE NXTCOL F907: DO EC 267 20 48 F9 268 JSR PRBLNK F909: F900: A4 2F 269 LDY LENGTH LDX #\$06 270 F90E: A2 06 271 PRADR1 CPX #\$03 F910: E0 03 272 BEG PRADR5 F912: FO 1C F914: 06 2E 273 PRADR2 ASL FORMAT 274 BCC PRADR3 F916: 90 OE LDA CHAR1-1, X F918: BD B3 F9 275 JSR COUT F91B: 20 ED FD 276 277 LDA CHAR2-1, X F91E: BD B9 F9 278 BEG PRADR3 F921: F0 03 JSR COUT F923: 20 ED FD 279 280 PRADR3 DEX F926: CA BNE PRADR1 DO E7 281 F927: RTS 282 F929: 60 283 PRADR4 DEY F92A: 88 BMI PRADR2 F92B: 30 E7 284 JSR PRBYTE F92D: 20 DA FD 285 LDA FORMAT 286 PRADR5 F930: A5 2E F932: C9 E8 287 CMP #\$E8

Apple 2 Tech	nical Manu	al • Apple ][ Reference Manual • 1979
F934: B1 3A	288	LDA (PCL), Y
<b>F93</b> 6: 90 F2 F938:	<b>287</b> 290	BCC PRADR4
F938: 20 56 F9	291 RELADR	PAGE JSR PCADJ3
F93B: AA F93C: E8	292 293	TAX
F93D: D0 01	294	INX BNE PRNTYX
F93F: C8 F940: 98	295	INY
F941: 20 DA FD	296 PRNTYX 297 PRNTAX	TYA JSR PRBYTE
F944: BA F945: 4C DA FD	298 PRNTX	TXA
F948: A2 03	299 300 PRBLNK	JMP PRBYTE LDX #\$03
F94A: A9 A0 F94C: 20 ED FD	301 PRBL2	LDA #\$AO
F94C: 20 ED FD F94F: CA	302 PRBL3 303	JSR COUT DEX
F950 DO F8	304	BNE PRBL2
F952: 60 F953: 38	305 306 PCADJ	RTS SEC
F954: A5 2F	307 PCADJ2	LDA LENGTH
F956: A4 3B F958: AA	308 PCADJ3 307	LDY PCH TAX
F959: 10 01	310	BPL PCADJ4
F95B: 88 F95C: 65 3A	311 312 PCADJ4	DEY ADC PCL
F95E: 90 01	313	BCC RTS2
F960: C8 F961: 60	314 315 RTS2	INY RTS
F962: 04	316 FMT1	DFB \$04
F963: 20 F964: 54	317 318	DFB \$20 DFB \$54
F965: 30	319	DFB \$30
F966: OD F967: 80	320 321	DFB \$0D DFB \$80
F968: 04	322	DFB \$04
F969: 90 F96A: 03	323 324	DFB \$90 DFB \$03
F96B: 22	325	DFB \$22
F96C: 54 F96D: 33	326 327	DFB \$54 DFB \$33
F96E: OD	328	DFB \$OD
F96F: 80 F970: 04	329 330	DFB \$80 DFB \$04
F971: 90	331	DFB \$90
F972: 04 F973: 20	332 333	DFB \$04 DFB \$20
F974: 54	334	DFB \$54
F975: 33 F976: OD	335 336	DFB \$33 DFB \$0D
F977: 80	337	DFB \$80
F978: 04 F979: 90	338 339	DFB \$04 DFB \$90
F97A: 04	340	DFE \$04
F97B: 20 F97C: 54	341 342	DFB \$20 DFB \$54
F97D: 3B	343	DFB \$3B
F97E: OD F97F: 80	344 345	DFB \$80D DFB \$80
F980: 04	346	DFB \$04
F9B1: 90 F9B2: 00	347 348	DFB \$90 DFB \$00
F983: 22	349	DFB \$22
F984: 44 F985: 33	350 351	DFB \$44 DFB \$33
F986: OD	352	DFB \$OD
F987: C8 F988: 44	353 354	DFB \$C8 DFB \$44
F989: 00	355	DFE \$00
F98A: 11 F98B: 22	356 357	DFB \$11 DFB \$22
F98C: 44	358	DFB \$44
F98D: 33 F98E: OD	359 360	DFB \$33 DFB \$0D

Apı	ple	2 Technical	Manual	•	,	Apple ][	Reference	Manual	•	1979
F98F:		361		DFB						
F990:	44	362		DFB						
F991: F992:	A9 01	363 364		DFB DFB						
F993:		365		DFE						
F994:	44	366		DFB						
F995:	33	367		DFB DFB						
F996: F997:	80 80	368 369		DFB						
F998:	04	370		DFB	<b>\$</b> 0	4				
F999:	90	371		DFB						
F99A: F99B:	01 22	372 373		DFB DFB						
F990:	44	374		DFB						
F99D:	33	375		DFB						
F99E: F99F:	0D 80	376 377		DFB DFB						
F9A0:	04	378		DFB						
F9A1:	90	379		DFB	\$9	0				
F9A2:		380		DFB						
F9A3: F9A4:	31 87	381 382		DFB DFB						
F9A5:	9A	383		DFB	\$9	'A				
F9A6:	00		FMT2	DFB						4
F9A7: F9A8:		385 386		DF B						
F9A9:		387		DFB						
F9AA:	00	388		DFB	<b>\$</b> 0	00				
F9AB: F9AC:	00	389		DFB DFB						
F9AU:	59 4D	390 391		DFB						
F9AE:	91	392		DFB	\$9	1				
F9AF:		393		DFB						
F9B0: F9B1:		394 395		DFB DFB						
F9B2:		396		DFB						
F9B3:	9D	397	5114E *	DFB	\$9	D O				
F9B4: F9B5:		3 <del>7</del> 8 399	CHAR1	DFB DFB						
F9B6:		400		DFB						
F9B7:	ΑЗ	401		DFB	\$A	<b>13</b>				
F9B8: F9B9:		402 403		DFB DFB						
F9BA:			CHAR2	DFB						
F9BB:	00	405		DFB	<b>\$</b> 0	00				
F9BC:		<b>4</b> 06		DFB						
F9BD: F9BE:		407 408		DFB DFB						
F9BF:	00	409	=:::	DFB	\$C	00				
F900:			MNEML	DFB						
F9C1:		411 412		DFB DFB						
F903:		413		DFB	\$2	23				
F904:	5D	414		DFE						
F905: F906:		415 416		DFB DFB						
F907:		417		DFB						
F908:	9D	418		DFB						
F909:		419 420		DFB DFB						
F9CA:		420 421		DFB						
F9CC:	<b>9</b> D	422		DFB	\$9	7D				
F9CD:		423		DFB						
F9CE:		424 425		DFB DFB						
F9D0:		426		DFB	\$0	00				
F9D1:	29	427		DFB						
F9D2: F9D3:		428 429		DFB DFB						
F9D3:		430		DFB						
F9D5:	A8	431		DFB	\$4	48				
F9D6:		432		DFB						
F9D7:	23	433		DFB	) \$PE	±J.				
						141				

Apple	2 Technical Man	ual •	Apple ][ Reference Manual	•	1979
F9D8: 24	434	DFB \$2			
F9D9: 53 F9DA: 1B	435	DFB \$5			
F9DB: 23	436 437	DFB \$1 DFB \$2			
F9DC: 24	438	DFB \$2	4		
F9DD: 53 F9DE: 19	43 <del>9</del> 440	DFB \$5			
F9DF: A1	441	DFB \$1			
F9E0: 00	442	DFB \$0	0		
F9E1: 1A F9E2: 5B	443 444	DFB \$1.			
F9E3: 5B	445	DFB \$5			
F9E4: A5 F9E5: 69	446	DFB \$A	5		
F9E6: 24	<b>44</b> 7 448	DFB \$69			
F9E7: 24	449	DFB \$24	4		
F9E8: AE F9E9: AE	450 451	DFB \$A			
F9EA: AB	452	DFB \$AB			
F9EB: AD	453	DFB \$AI	)		
<b>F9E</b> C: 29 <b>F9</b> ED: 00	454 455	DFB \$29			
F9EE: 7C	456	DFB \$00 DFB \$70			
F9EF: 00	457	DFB \$00			
F9F0: 15 F9F1: 9C	458 459	DFB \$15			
F9F2: 6D	460	DFB \$9( DFB \$61			
F9F3: 9C	461	DFB \$90			
F9F4: A5 F9F5: 69	462 463	DFB \$69			
F9F6: 29	464	DFD \$29	, ,		
F9F7: 53	465	DFB \$53	3		
F9F8: 84 F9F9: 13	466 467	DFB \$84 DFB \$13			
F9FA: 34	468	DFB \$34			
F9FB: 11 F9FC: A5	469 470	DFB \$11			
F9FD: 69	471	DFB \$45 DFB \$69			
F9FE: 23	472	DFB \$23	3		
F9FF: A0 FA00: DB	473 474 MNEMR	DFB \$AC			
FA01: 62	475	DFB \$62			
FA02: 5A FA03: 48	476 477	DFB \$5A			
FA04: 26	477 478	DFB \$48			
FA05: 62	479	DFB \$62	?		
FA06: 94 FA07: 88	480 481	DFB \$94			
FA08: 54	481 482	DFB \$88 DFB \$54			
FA09: 44	483	DFE \$44			
FAOA: C8 FAOB: 54	484 <b>48</b> 5	DFB \$C8			
FA0C: 68	485	DFB \$68			
FAOD: 44 FAOE: E8	487 480	DFB \$44			
FAOF: 94	488 489	DFB \$E8 DFB \$94			
FA10: 00	490	DFB \$00			
FA11: B4 FA12: OB	491 492	DFB \$B4			
FA13: 84	472 493	DFB \$08 DFB \$84			
FA14: 74	494	DFB \$74			
FA15: B4 FA16: 28	495 496	DFB \$84 DFB \$28			
FA17: 6E	<b>4</b> 97	DFB \$6E			
FA18: 74 FA19: F4	498 499	DFB \$74			
FA1A: CC	500	DFB \$F4 DFB \$CC			
FA1B: 4A	501	DFB \$4A			
FA1C: 72 FA1D: F2	502 503	DFB \$72			
FA1E: A4	504	DFB \$F2 DFB \$A4			
FA1F: 8A	505	DFB \$8A			
FA20: 00	506	DFB \$00			
			142		

```
Apple 2 Technical Manual • Apple ][ Reference Manual
                                                                            1979
FA21 ·
      AA
                   507
                                DFB $AA
FA22:
      A2
                   508
                                DFB $A2
FA23:
                   509
                                DFB $A2
FA24:
      74
                   510
                                DFB $74
FA25:
                   511
                                DFB $74
FA26:
      74
                   512
                                DFB $74
FA27:
      72
                                DER $72
                   513
FA28:
      44
                   514
                                DFB $44
FA29:
      68
                   515
                                DFB $68
FA2A:
      B2
                   516
                                DFB $B2
FA2B:
      32
                   517
                                DFB $32
FA2C: B2
                   518
                               DFB $B2
FA2D:
      00
                   519
                                DFB $00
FA2E:
      22
                   520
                                DFB $22
FA2F:
      00
                   521
                                DFB $00
FA30:
      1 A
                   522
                                DFB $1A
FA31:
                   523
                                DFB $1A
FA32:
      26
                   524
                                DFB $26
FA33:
                   525
      26
                               DFB $26
FA34:
      72
                   526
                                DFB $72
FA35:
      72
                   527
                                DFB $72
FA36:
      88
                   528
                                DFB $88
FA37:
      C8
                   529
                                DFB $C8
FA38: C4
                   530
                                DFB $C4
FA39:
                   531
                                DFB $CA
      CA
FA3A:
                                DFB $26
      26
                   532
FA3B: 48
                   533
                                DFB $48
FA3C: 44
                   534
                                DFB $44
FA3D: 44
                                DFB $44
                   535
FA3E:
      A2
                   536
                                DFB $A2
FASF
                   537
                                DFB $CB
      C8
FA40:
                   538
                                PAGE
FA40: 85 45
                   539 IRQ
                                STA ACC
FA42: 68
                   540
                                PLA
FA43: 48
                   541
                                PHA
FA44:
      OΑ
                   542
                                ASL A
FA45:
      OΑ
                   543
                                ASL A
FA46:
      OA
                   544
                                ASL A
                   545
FA47:
                                BMI BREAK
      30 03
FA49:
      6C FE 03
                                    (IRQLOC)
                   546
                                JMP
FA4C:
      28
                   547 BREAK
                                PLP
FA4D:
      20 4C FF
                   548
                                JSR SAV1
FA50:
                   549
      68
                                PLA
FA51:
      85 3A
                   550
                                STA PCL
FA53:
      68
                   551
                                PLA
FA54 ·
      85 3B
                   552
                                STA PCH
FA56:
      6C FO 03
                   553
                                JMP (BRKV) ; BRKV WRITTEN OVER BY DISK BOOT
FA59:
      20 82 F8
                   554 OLDBRK
                                JSR INSDS1
FA5C:
      20 DA FA
                   555
                                JSR RGDSP1
FA5F:
      4C 65 FF
                   556
                                JMP MON
                                            ; DO THIS FIRST THIS TIME
FA62:
      D8
                   557 RESET
                                CLD
      20 84 FE
                                JSR SETNORM
FA63:
                   558
FA66:
      20 2F FB
                   559
                                JSR INIT
FA69:
      20 93 FE
                   560
                                JSR SETVID
FA6C:
      20 89 FE
                   561
                                JSR SETKBD
FA6F:
      AD 58 CO
                   562 INITAN
                               LDA SETANO ; ANO = TTL HI
                                LDA SETAN1 ; AN1 = TTL HI
FA72:
      AD 5A CO
                   563
FA75:
      AD 5D CO
                   564
                                LDA CLRAN2 ;
                                             AN2 = TTL LO
FA78:
      AD 5F CO
                   565
                                LDA CLRAN3 ;
                                              AN3 = TTL LO
FA7B:
      AD FF CF
                                LDA CLRROM; TURN OFF EXTNSN ROM
                   566
FA7E:
      2C 10 CO
                   567
                                BIT KBDSTRB ; CLEAR KEYBOARD
FA81:
      D8
                   568 NEWMON CLD
FA82:
      20 3A FF
                                JSR BELL
                                           ; CAUSES DELAY IF KEY BOUNCES
                   569
FA85:
      AD F3 03
                   570
                                LDA SOFTEV+1 ; IS RESET HI
      49 A5
                   571
FA88:
                                EOR #$A5
                                          ; A FUNNY COMPLEMENT OF THE
                   572
FA8A:
      CD F4 03
                                CMP PWREDUP ; PWR UP BYTE ???
FABD: DO 17
                   573
                                BNE PWRUP ; NO SO PWRUP
                   574
FABF:
      AD F2 03
                                LDA SOFTEV; YES SEE IF COLD START
FA92:
      DO OF
                   575
                                BNE NOFIX ; HAS BEEN DONE YET?
FA94:
      A9 E0
                   576
                                LDA #$EO
                                           ; ??
                                CMP SOFTEV+1 ; ??
                   577
FA96: CD F3 03
FA99: DO 08
                   578
                                BNE NOFIX ; YES SO REENTER SYSTEM
FA9B: A0 03
                   579 FIXSEV LDY #3
                                           ; NO SO POINT AT WARM START
```

```
FA9D:
       8C F2 03
                    580
                                 STY SOFTEV ; FOR NEXT RESET
       4C 00 E0
                                 JMP BASIC ; AND DO THE COLD START JMP (SOFTEV) ; SOFT ENTRY VECTOR
FAA0:
                    581
       6C F2 03
                    582 NOFIX
FAA3:
FAA6:
                    583 ******************
FAA6:
      20 60 FB
                    584 PWRUP
                                  JSR APPLEII
                    585 SETPG3
FAA9:
                                 EQU *
                                             ; SET PAGE 3 VECTORS
       A2 05
FAA9:
                    586
                                 LDX #5
FAAB: BD FC FA
                    587 SETPLP
                                 LDA PWRCON-1, X; WITH CNTRL B ADRS
FAAE:
       9D EF 03
                                 STA BRKV-1, X ; OF CURRENT BASIC
                    588
FAB1:
       CA
                    589
                                 DEX
       DO F7
FAB2:
                    590
                                 BNE SETPLP
FAB4:
       A9 C8
                    591
                                 LDA #$CB
                                            ; LOAD HI SLOT +1
FAB6:
       86 00
                    592
                                 STX LOCO
                                            ; SETPG3 MUST RETURN X=0
FAB8: 85 01
                    593
                                 STA LOC1
                                             ; SET PTR H
FABA: A0 07
                    594 SLOOP
                                 LDY #7
                                             ; Y IS BYTE PTR
FABC:
      C6 01
                    595
                                 DEC LOCA
FABE: A5 01
                    596
                                 LDA LOC1
FACO: C9 CO
                    597
                                 CMP #$CO
                                             ; AT LAST SLOT YET?
FAC2:
      FO D7
                    598
                                 BEQ FIXSEV ; YES AND IT CANT BE A DISK
FAC4:
      8D F8 07
                    599
                                 STA MSLOT
FAC7:
      B1 00
                    600 NXTBYT
                                LDA (LOCO), Y ; FETCH A SLOT BYTE CMP DISKID-1, Y ; IS IT A DISK ??
FAC9: D9 01 FB
                    601
FACC:
      DO EC
                    602
                                 BNE SLOOP ; NO SO NEXT SLOT DOWN
FACE:
      88
                    603
                                 DEY
FACE:
      88
                    604
                                 DEY
                                             ; YES SO CHECK NEXT BYTE
FADO:
      10 F5
                                 BPL NXTBYT ; UNTIL 4 CHECKED
                    605
FAD2:
      60 00 00
                                 JMP (LDCO)
                    606
FAD5:
      EΑ
                    607
                                 NOP
FAD6:
      EΑ
                    808
                                 NOP
FAD7:
                    609 * REGDSP MUST ORG $FAD7
      20 8E FD
FAD7:
                    610 REGDSP
                                 JSR CROUT
FADA:
      A9 45
                    611 RGDSP1
                                 LDA #$45
FADC:
      85 40
                    612
                                 STA A3L
FADE:
      A9 00
                    613
                                 LDA #$00
      85 41
FAE0:
                    614
                                 STA A3H
FAE2:
      A2 FB
                    615
                                 LDX #$FB
FAE4:
      A9 A0
                    616 RDSP1
                                 LDA #$A0
FAE6:
      20 FD FD
                    617
                                 JSR COUT
FAE9:
      BD 1E FA
                    618
                                 LDA RTBL-251, X
FAEC:
      20 ED FD
                    619
                                 JSR COUT
FAEF:
      A9 BD
                    620
                                 LDA #$BD
FAF1:
      20 ED FD
                                 JSR COUT
                    621
FAF4:
                    622 * LDA ACC+5, X
FAF4:
      B5 4A
                    623
                                 DFB $B5, $4A
FAFó:
      20 DA FD
                    624
                                 JSR PRBYTE
FAF9:
      E8
                    625
                                 INX
FAFA:
      30 E8
                    626
                                 BMI RDSP1
FAFC: 60
                    627
                                 RTS
FAFD: 59 FA
                    628 PWRCON
                                DW DIDBRK
FAFF:
      00 E0 45
                    629
                                 DFB $00,$E0,$45
FBO2:
      20 FF 00
FBO5: FF
                                DFB $20, $FF, $00, $FF
                    630 DISKID
FB06: 03 FF 3C
                    631
                                 DFB $03, $FF, $30
FB09: C1 D0 D0
                   632 TITLE
                                 DFB $C1,$D0,$D0
FBOC: CC C5 AO
                   633
                                 DFB $CC, $C5, $AO
FBOF: DD DB
                    634
                                 DFB $DD, $DB
FB11:
                    635 XLTBL
                                 EQU *
FB11: C4 C2 C1
                    636
                                 DFB $C4, $C2, $C1
FB14: FF C3
                    637
                                 DFB $FF, $C3
      FF FF FF
FB16:
                    638
                                 DFB $FF, $FF, $FF
FB19:
                    639 * MUST ORG $FB19
FB19: C1 D8 D9
                   640 RTBL
                                DFB $C1,$D8,$D9
FB1C: DO D3
                    641
                                 DFB $D0, $D3
FB1E: AD 70 CO
                   642 PREAD
                                LDA PTRIG
FB21 ·
                    643
                                 LST ON
FB21: A0 00
                    644
                                 LDY
                                     #$00
FB23: EA
                   645
                                 NOP
FB24: EA
                   646
                                 NOP
FB25: BD 64 CO
                   647 PREAD2
                                LDA PADDLO, X
FB28: 10 04
                    648
                                 BPL RTS2D
FB2A: CB
                    649
                                 INY
FB2B: DO F8
                    650
                                 BNE PREAD2
FB2D: 88
                   651
                                 DEY
```

```
FB2E:
     60
                   652 RTS2D
                                RTS
      A9 00
FB2F
                                LDA #$00
                       INIT
      25 48
FB31:
                                STA STATUS
FB33:
      AD 56 CO
                                LDA LORES
FB36:
      AD 54 CO
                     5
                                LDA LOWSCR
                     6 SETTXT
FB39:
      AD 51 CO
                                LDA TXTSET
FB3C:
      A9 00
                                LDA #$00
FB3E:
      FO OB
                     8
                                BEG SETWND
                     9 SETGR
FB40:
      AD 50 CO
                                LDA TXTCLR
FB43:
      AD 53 CO
                    10
                                LDA MIXSET
FB46:
      20 36 F8
                    11
                                JSR CLRTOP
FB49:
      A9 14
                                LDA #$14
FB4B: 85 22
                    13 SETWND
                               STA WNDTOP
      A9 00
FB4D:
                    14
                                LDA #$00
      85 20
FB4F:
                    15
                                STA WNDLFT
FB51:
      A9 28
                    16
                                LDA #$28
FB53:
      85 21
                    17
                                STA WNDWDTH
FB55:
      A9 18
                                LDA #$18
FB57:
      85
                    19
         23
                                STA WNDBTM
FB59:
      A9 17
                    20
                                LDA #$17
FB5B:
      85 25
                    21 TABV
                                STA CV
FB5D:
      40 22 FC
                                JMP VTAB
                    22
                    23 APPLEII JSR HOME
      20 58 FC
FB60:
                                           ; CLEAR THE SCRN
FB63:
      A0 08
                    24
                                LDY #8
                               LDA TITLE-1, Y ; GET A CHAR
FB45:
      B9 08 FB
                    25 STITLE
      99 OE 04
FB68:
                    26
                                STA LINE1+14, Y
FB6B:
      88
                    27
                                DEY
FB6C:
      DO F7
                    28
                                BNE STITLE
FB6E:
      60
                    29
                                RTS
      AD F3 03
                    30 SETPWRC LDA SOFTEV+1
FB6F:
FB72:
      49 A5
                    31
                                EOR #$A5
FB74:
      8D F4 03
                    32
                                STA PWREDUP
FB77:
      60
                    33
                                RTS
FB78:
                    34 VIDWAIT EQU *
                                           ; CHECK FOR A PAUSE
FB78:
      C9 BD
                    35
                                CMP #$8D
                                           ; ONLY WHEN I HAVE A CR
                                BNE NOWAIT ; NOT SO, DO REGULAR
FB7A: DO 18
                    36
FB7C:
      AC 00 C0
                    37
                                LDY KBD
                                           ; IS KEY PRESSED?
FB7F:
      10 13
                    38
                                BPL NOWAIT ; NO
                                CPY #$93
FB81:
      CO 93
                                          ; IS IT CTL S ?
FB83:
      DO OF
                    40
                                BNE NOWAIT ; NO SO IGNORE
                               BIT KBDSTRB ; CLEAR STROBE
FB85:
      2C 10 CO
                    41
                    42 KBDWAIT LDY KBD
FB88:
      AC 00 C0
                                           ; WAIT TILL NEXT KEY TO RESUME
                               BPL KBDWAIT ; WAIT FOR KEYPRESS
FB8B:
     10 FB
                    43
FB8D:
      CO 83
                    44
                                CPY #$83 ; IS IT CONTROL C
FB8F:
      FO 03
                    45
                                BEQ NOWAIT ; YES SO LEAVE IT
FB91:
      2C 10 CO
                                BIT KBDSTRB ; CLR STROBE
FB94:
      4C FD FB
                    47 NOWAIT
                               JMP VIDOUT ; DO AS BEFORE
                               PAGE
FB97:
                    48
FB97:
                    49 ESCOLD
                                           ; INSURE CARRY SET
      38
                               SEC
FB98:
      4C 2C FC
                    50
                                JMP ESC1
FB9B
      A8
                    51 ESCNOW
                               TAY
                                           ; USE CHAR AS INDEX
      B9 48 FA
FB9C:
                    52
                                LDA XLTBL-$C9, Y ; XLATE IJKM TO CBAD
FB9F:
      20 97 FB
                    53
                                JSR ESCOLD ; DO THIS CURSOR MOTION
      20 OC FD
                                JSR RDKEY ; AND GET NEXT
FBA2:
                                           ; IS THIS AN N ?
FBA5:
      C9 CE
                    55 ESCNEW
                               CMP #$CE
      BO EE
                    56
                               BCS ESCOLD ; N OR GREATER DO IT
FBA7:
                                          ; LESS THAN I ?
FRA9.
      C9 C9
                    57
                                CMP #$C9
FBAB:
      90 EA
                    58
                                BCC ESCOLD ; YES SO OLD WAY
FBAD:
      C9 CC
                    59
                                CMP #$CC ; IS IT A L
FBAF: FO E6
                               BEG ESCOLD ; DO NORMAL
                    60
FBB1: DO E8
                               BNE ESCNOW ; GO DO IT
                    61
FBB3: EA
                    62
                               NOP
FBB4: EA
                    63
                                NOP
FBB5:
      EΑ
                    64
                                NOP
FBB6:
                                NOP
      EA
FBB7:
      EΑ
                                NOP
                    66
FBB8: EA
                    67
                                NOP
FBB9: EA
                    68
                                NOP
FBBA: EA
                                NOP
```

```
Apple 2 Technical Manual •
                                       Apple | Reference Manual
                                                                             1979
                    70
71
FBBB: EA
                                NOP
FBBC:
      EΑ
                                NOP
FBBD:
      EΑ
                    72
                                NOP
                    73
FBBE:
      EΑ
                                NOP
FBBF:
                                NOP
FBCO:
                    75
                                NOP
FBC1:
                    76 *
                             MUST DRG $FBC1
                    77 BASCALC PHA
      48
FBC1:
FBC2:
                    78
      44
                                LSR A
                    79
FBC3:
      29 03
                                E0## G/A
FBC5:
      09
                    80
                                ORA #$04
FBC7:
      85
         29
                    81
                                STA BASH
FBC9:
      68
                    82
                                PLA
FBCA:
      29 18
                                AND #$18
                    83
FBCC:
      90 02
                    84
                                BCC BASCLC2
FBCE:
      69
         7F
                    85
                                ADC #$7F
FBDO:
                    86 BASCLC2 STA BASL
FBD2:
      OA
                    87
                                ASL A
FBD3:
      OA
                    88
                                ASL A
      05 28
                                ORA BASL
FBD4:
                    89
FBD6:
      85
         28
                    90
                                STA
                                    BASL
FBD8:
      60
                    91
                                RTS
FBD9:
      C9 87
                    92 BELL1
                                CMP
                                    #$87
FBDB: DO 12
                    93
                                BNE RTS2B
FBDD:
      A9 40
                    94
                                LDA #$40
FBDF:
      20 A8 FC
                    95
                                JSR WAIT
                    96
FBE2:
      AO CO
                                LDY #$CO
                    97 BELL2
FBE4:
      A9 0C
                                LDA #$OC
FBE6:
      20 AB FC
                    98
                                JSR WAIT
FBE9:
      AD 30 CO
                    99
                                LDA SPKR
FBEC:
                   100
                                DEY
FBED:
      DO F5
                   101
                                BNE BELL2
FBEF:
                   102 RTS2B
      60
                                RTS
FBF0:
                   103
                                PAGE
FBF0:
      A4 24
                   104 STORADV LDY CH
      91 28
FBF2:
                   105
                                STA (BASL), Y
FBF4:
      E6 24
                   106 ADVANCE INC CH
FBF6:
      A5 24
                   107
                                LDA CH
FBF8:
      C5 21
                   108
                                CMP
                                    WNDWDTH
FBFA: BO 66
                                BCS CR
                   109
FBFC:
                   110 RTS3
      60
                                RTS
FBFD: C9 A0
                   111 VIDOUT
                                CMP #$AO
FBFF:
      BO EF
                   112
                                BCS STORADV
FCO1:
      A8
                   113
FC02:
      10 EC
                   114
                                BPL STORADY
FCO4:
      C9
                   115
                                CMP
                                    #$8D
      FO 5A
FC06:
                   116
                                BEQ CR
FCOB:
      C9 8A
                   117
                                CMP #$8A
FCOA:
      FO 5A
                   118
                                BEQ LF
FCOC:
      C9 88
                   119
                                CMP #$88
FC0E: D0 C9
                   120
                                BNE BELL1
FC10:
      C6 24
                   121 BS
                                DEC CH
FC12:
      10 E8
                   122
                                BPL RTS3
FC14: A5 21
                   123
                                LDA WNDWDTH
FC16:
      85 24
                   124
                                STA CH
FC18: C6 24
                   125
                                DEC CH
FC1A: A5 22
                   126 UP
                                LDA WNDTOP
FC1C:
      C5
                   127
                                CMP
                                    CV
FC1E: BO OB
                                BCS RTS4
                   128
FC20: C6 25
                   129
                                DEC CV
FC22: A5 25
                   130 VTAB
                                LDA CV
FC24:
      20 C1 FB
                   131 VTABZ
                                JSR BASCALC
FC27:
      65 20
                   132
                                ADC WNDLFT
FC29:
      85 28
                   133
                                STA BASL
FC2B: 60
                   134 RTS4
                                RTS
FC2C: 49 CO
                   135 ESC1
                                EOR #$CO
                                            ; ESC @ ?
FC2E: F0 28
                                            ; IF SO DO HOME AND CLEAR
                   136
                                BEQ HOME
FC30:
      69 FD
                   137
                                ADC #$FD
                                            ; ESC-A OR B CHECK
FC32:
      90 CO
                   138
                                BCC ADVANCE ; A, ADVANCE
FC34: FO DA
                   139
                                BEG BS
                                           ; B, BACKSPACE
FC36: 69 FD
                   140
                                           ; ESC-C OR D CHECK
                                ADC #$FD
FC38: 90 2C
                   141
                                BCC LF
                                           ; C, DOWN
FC3A: FO DE
                   142
                                BEQ UP
                                            ; D, GO UP
```

### Apple 2 Technical Manual • Apple ][ Reference Manual 1979 ADC #\$FD ; ESC-E OR F CKECK FC3C: 69 FD 143 BCC CLREOL ; E, CLEAR TO END OF LINE 144 FC3E: 90 5C ; ELSE NOT F, RETURN BNE RTS4 145 FC40: DO E9 ; ESC F IS CLR TO END OF PAGE FC42: A4 24 146 CLREOP LDY CH FC44: A5 25 147 LDA CV FC46: 48 148 CLEOP1 PHA JSR VTABZ FC47: 20 24 FC 149 20 9E FC 150 JSR CLEOLZ FC4A: LDY #\$00 FC4D: AO OO 151 FC4F: 68 152 PLA FC50: 69 00 153 ADC #\$00 154 CMP WNDBTM FC52: C5 23 90 FO 155 BCC CLEOP1 FC54: BCS VTAB FC56: BO CA 156 LDA WNDTOP 157 HOME FC58: A5 22 STA CV FC5A: 85 25 158 LDY #\$00 FC5C: AO 00 159 STY CH FC5E: 84 24 160 BEG CLEOP 1 FC60: F0 E4 161 PAGE 162 FC62: LDA #\$00 A9 00 163 CR FC62: STA CH FC64: 85 24 164 FC66: E6 25 165 LF INC CV LDA CV FC68: A5 25 166 FC6A: C5 23 CMP WNDBTM 167 BCC VTABZ 168 FC6C: 90 B6 DEC CV FC6E: C6 25 169 LDA WNDTOP 170 SCROLL FC70: A5 22 FC72: 48 171 PHA FC73: 20 24 FC 172 JSR VTABZ FC76: A5 28 FC78: 85 2A 173 SCRL1 LDA BASL 174 STA BASZL 175 LDA BASH FC7A: A5 29 STA BAS2H FC7C: 85 2B 176 FC7E: A4 21 177 LDY WNDWDTH 178 DEY FCB0: 88 179 PLA FC81: 68 69 01 180 ADC #\$01 FC82: C5 23 CMP WNDBTM FC84: 181 BCS SCRL3 FC86: BO OD 182 FC88: 48 183 PHA JSR VTABZ FC89: 20 24 FC 184 185 SCRL2 LDA (BASL), Y FCBC: B1 28 STA (BAS2L), Y 91 2A 186 FCBE: DEY FC90: 88 187 BPL SCRL2 FC91: 10 F9 188 FC93: 30 E1 189 BMI SCRL1 190 SCRL3 LDY #\$00 FC95: A0 00 FC97: 20 9E FC 191 JSR CLEOLZ 192 BCS VTAB FC9A: BO 86 193 CLREOL LDY CH FC9C: A4 24 LDA #\$A0 FC9E: A9 A0 194 CLEOLZ 195 CLEOL2 STA (BASL), Y FCAO: 91 28 INY FCA2: CB 196 WNDWDTH 197 CPY FCA3: C4 21 198 90 F9 BCC CLEOL2 FCA5: 199 FCA7: 60 RTS FCA8: 38 200 WAIT SEC 201 WAIT2 PHA FCA9: 48 FCAA: E9 01 202 WAIT3 SBC #\$01 BNE WAITS 203 FCAC: DO FC 204 PLA FCAE: 68 SBC #\$01 FCAF: E9 01 205 FCB1: DO F6 206 BNE WAIT2 207 RTS FCB3: 60 208 NXTA4 E6 42 INC A4L FCB4: BNE NXTA1 209 FCB6: DO 02 INC A4H FCB8: E6 43 210

FCBA: A5 3C

FCBC: C5 3E

FCBE: A5 3D

FCCO: E5 3F

FCC2: E6 3C

211 NXTA1

212

213

214

215

LDA A1L

CMP A2L LDA A1H

SBC A2H

INC A1L

### Apple 2 Technical Manual • Apple ][ Reference Manual 1979 FCC4: DO 02 BNE RTS4B 216 FCC6: E6 3D 217 INC A1H FCC8: 60 218 RTS4B RTS FCC9 219 PAGE FCC9: AO 4B 220 HEADR LDY #\$4B FCCB: 20 DB FC 221 JSR ZERDLY FCCE: DO F9 222 BNE HEADR FCDO: 223 ADC #\$FE FCD2: BO F5 224 BCS HEADR FCD4: AO 21 225 LDY #\$21 20 DB FC 226 WRBIT FCD6: JSR ZERDLY FCD9: CB 227 INY FCDA: CB INY FCDB: 88 229 ZERDLY DEY FCDC: DO FD 230 BNE ZERDLY FCDE: 90 05 231 BCC WRIAPE FCE0: A0 32 232 LDY #\$32 233 ONEDLY FCE2: 88 DEY DO FD FCE3: 234 BNE ONEDLY FCE5: AC 20 CO 235 WRTAPE LDY TAPEDUT FCEB: A0 20 236 LDY #\$20 FCEA: 237 DEX FCEB: 60 238 RTS FCEC: A2 08 239 RDBYTE LDX #\$OR FCEE: 48 240 RDBYT2 PHA FCEF: 20 FA FC 241 JSR RD2BIT FCF2: 68 242 PLA FCF3: 2A 243 ROL A FCF4: AO 3A 244 LDY #\$3A FCF6: CA 245 DEX FCF7: DO F5 246 BNE RDBYT2 FCF9: 247 60 RTS FCFA: 20 FD FC 248 RD2BIT JSR RDBIT FCFD: 88 249 RDBIT DEY FCFE: AD 60 CO 250 LDA TAPEIN FD01: 45 2F 251 EOR LASTIN FD03: 10 F8 252 BPL RDBIT FD05: 45 2F 253 EOR LASTIN FD07: 85 2F 254 STA LASTIN FD09: CO 80 255 CPY #\$80 FDOB: 60 256 RTS FDOC: A4 24 257 RDKEY LDY CH FD0E: B1 28 258 LDA (BASL), Y FD10: 259 48 PHA FD11: 29 3F AND #\$3F 260 FD13: 09 40 261 DRA #\$40 FD15: 91 28 262 STA (BASL), Y FD17: 68 263 PLA FD18: 6C 38 00 264 JMP (KSWL) FD1B: E6 4E 265 KEYIN INC RNDL FD1D: DO 02 266 BNE KEYIN2 FD1F: E6 4F 267 INC RNDH 268 KEYIN2 FD21: 2C 00 C0 ; READ KEYBOARD BIT KBD FD24: 10 F5 269 BPL KEYIN FD26: 91 28 270 STA (BASL), Y AD 00 C0 FD28: 271 LDA KBD FD2B: 2C 10 CO 272 BIT KBDSTRB FD2E: 60 273 RTS FD2F: 20 OC FD 274 ESC JSR RDKEY FD32: 275 20 A5 FB JSR ESCNEW FD35: 20 OC FD 276 RDCHAR JSR RDKEY FD38: C9 9B 277 CMP #\$9B FD3A: F0 F3 278 BEQ ESC FD3C: 60 FD3D 280 PAGE A5 32 FD3D: 281 NOTCR LDA INVFLG FD3F: 48 282 PHA FD40: A9 FF 283 LDA #\$FF FD42: 85 32 284 STA INVFLG FD44: BD 00 02 285 LDA IN. X FD47: 20 ED FD 286 JSR COUT

PLA

STA INVFLG

287

288

FD4A:

68

FD4B: 85 32

```
Apple 2 Technical Manual •
                                        Apple | Reference Manual
                                                                               1979
FD4D:
      BD 00 02
                    289
                                 LDA IN, X
FD50:
      C9 88
                    290
                                 CMP
                                     #$88
FD52:
      FO 1D
                    291
                                 BEQ BCKSPC
      C9 98
FD54:
                    292
                                 CMP #$98
FD56:
      FO OA
                    293
                                 BEQ CANCEL
FD58:
      EO FB
                    294
                                 CPX #$F8
FD5A:
      90 03
                    295
                                 BCC NOTCR1
FD5C:
      20 3A FF
                    296
                                 JSR BELL
                    297 NOTCR1
FD5F:
      E8
                                 INX
FD60:
      DO 13
                    298
                                 BNE NXTCHAR
      A9 DC
                    299
FD62:
                        CANCEL
                                 LDA #$DC
      20 ED FD
FD64:
                    300
                                 JSR COUT
FD67:
      20 8E FD
                    301
                        GETLNZ
                                 JSR CROUT
FD6A:
      A5 33
                    302
                        GETLN
                                 LDA PROMPT
FD6C:
      20 ED FD
                    303
                                 JSR COUT
FD6F:
      A2 01
                    304
                                 LDX #$01
FD71:
      84
                    305 BCKSPC
                                 TXA
FD72:
      FO F3
                    306
                                 BEG GETLNZ
FD74:
      CA
                    307
                                 DEX
FD75:
      20 35 FD
                    308 NXTCHAR JSR RDCHAR
      C9
FD78:
         95
                   309
                                 CMP
                                     #$95
FD7A:
      DO 02
                   310
                                 BNE CAPTST
      B1 28
FD7C:
                   311
                                 LDA
                                     (BASL), Y
                   312 CAPTST
FD7E:
      C9
         ΕO
                                 CMP
                                     #$E0
FD80:
      90 02
                   313
                                 BCC ADDINA
FD82:
      29
         DF
                   314
                                 AND
                                     #$DF
                                            ; SHIFT TO UPPER CASE
FD84:
      9D 00 02
                   315 ADDINP
                                STA IN, X
FD87:
      C9 8D
                   316
                                 CMP
                                     #$8D
FD89:
      DO B2
                   317
                                 BNE NOTCR
FDSB:
      20 9C FC
                   318
                                 JSR
                                     CLREOL
FD8E:
      A9 8D
                   319 CROUT
                                 LDA
                                     #$8D
FD90:
      DO 5B
                    320
                                 BNE COUT
FD92:
      A4
                    321 PRA1
         ЗD
                                 LDY A1H
FD94:
      Α6
         30
                    322
                                 LDX A1L
                   323 PRYX2
FD96:
      20 8E FD
                                 JSR CROUT
FD99:
      20 40
            F9
                    324
                                 JSR PRNTYX
FD9C:
      A0 00
                    325
                                 LDY #$00
FD9E:
      Α9
         AD
                    326
                                 LDA
                                     #$AD
FDAO:
      4C ED FD
                    327
                                 JMP COUT
FDA3:
                                 PAGE
FDA3:
      A5 30
                    329
                        XAMB
                                 LDA A1L
FDA5:
      09
         07
                   330
                                 ORA #$07
FDA7:
      85 3E
                   331
                                 STA A2L
FDA9:
      A5 3D
                   332
                                 LDA A1H
FDAB:
      85 3F
                    333
                                 STA
                                     A2H
FDAD:
      Α5
         30
                    334 MODBCHK LDA
FDAF:
      29
         07
                   335
                                     #$07
                                 AND
FDB1:
      DO
         03
                   336
                                 BNE DATAOUT
FDB3:
      20 92 FD
                   337
                        XAM
                                 JSR PRA1
                   338 DATACUT LDA #$AO
FDB6:
      A9 A0
FDB8:
      20 ED FD
                   339
                                 JSR COUT
FDBB:
      B1 3C
                   340
                                 LDA
                                     (A1L), Y
FDBD:
      20 DA FD
                   341
                                 JSR PRBYTE
FDCO:
      20 BA FC
                   342
                                 JSR
                                     NXTA1
FDC3:
      90 E8
                   343
                                 BCC
                                     MODBCHK
FDC5:
                   344 RT54C
      60
                                 RTS
FDC6:
      44
                   345 XAMPM
                                 LSR
      90 EA
FDC7:
                   346
                                 BCC
                                     XAM
FDC9:
      4A
                   347
                                 LSR
FDCA:
                   348
                                 LSR A
FDCB:
      A5 3E
                   349
                                 LDA A2L
      90 02
FDCD:
                   350
                                 BCC ADD
FDCF:
      49 FF
                   351
                                 EOR
                                     #$FF
FDD1:
      65
         30
                   352 ADD
                                 ADC
                                     A1L
FDD3:
                   353
                                 PHA
FDD4:
      Α9
         BD
                   354
                                 LDA #$BD
                   355
FDD6:
      20 ED FD
                                 JSR COUT
FDD9:
      68
                   356
                                 PLA
FDDA:
      48
                    357
                        PRBYTE
                                 PHA
FDDB:
      44
                    358
                                 LSR
FDDC:
      44
                    359
                                 LSR
FDDD: 4A
                   360
                                 LSR
FDDE: 4A
                   361
                                 LSR
```

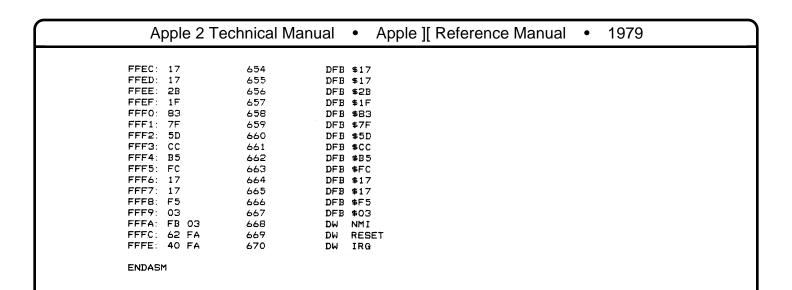
### Apple 2 Technical Manual • Apple | Reference Manual 1979 FDDF: 20 E5 FD 362 JSR PRHEXZ FDE2: 68 363 PLA 364 PRHEX AND #\$OF FDE3: 29 OF 09 365 PRHEXZ FDE5: BO ORA #\$B0 FDE7: C9 BA CMP #\$BA 366 FDE9: 90 02 367 BCC COUT FDFB: 69 06 368 ADC #\$06 369 COUT FDED: 9C 39 00 JMP (CSWL) FDF0: C9 A0 370 CDUT1 CMP #\$A0 FDF2: 90 02 371 BCC COUTZ FDF4: 25 32 372 AND INVFLG FDF6: 84 35 373 COUTZ STY YSAV1 FDF8: 48 374 PHA 20 78 FB FDF9: 375 JSR VIDWAIT ; GO CHECK FOR PAUSE FDFC: 68 376 FDFD: A4 35 377 LDY YSAV1 FDFF: 60 378 RTS FE00: 379 PAGE C6 34 FE00: 380 BL1 DEC YSAV F0 9F FE02: 381 BEG XAMB FE04: CA 382 BLANK DEX FE05: DO 16 383 BNE SETMDZ FE07: C9 384 ΒA CMP #\$BA FEO9: DO BB 385 BNE XAMPM FEOB: 85 386 STOR 31 STA MODE FEOD: A5 3E 387 LDA A2L 91 40 FEOF: 388 STA (A3L), Y FE11: E6 40 389 INC A3L FE13: DO 02 390 BNE RTS5 FE15: E6 41 391 INC A3H FE17: 392 RTS5 60 RTS FE18: A4 34 393 SETMODE LDY YSAV B9 FF 01 FE1A: 394 LDA IN-1, Y 395 SETMDZ FE1D: 85 31 STA MODE FE1F: 60 396 RTS FE20: A2 01 397 LT LDX #\$01 FE22: В5 398 LT2 LDA A2L, X FE24: 95 42 399 STA A4L, X FE26: 95 44 400 STA A5L, X FE28: CA 401 DEX 10 F7 FE29: 402 BPL LT2 FE2B: 60 403 RTS FE2C: B1 3C 404 MOVE LDA (A1L), Y FE2E: 91 42 405 STA (A4L), Y FE30: 20 B4 FC 406 JSR NXTA4 FE33: 90 F7 407 BCC MOVE FE35: 60 408 RTS FE36: B1 3C VFY 409 LDA (A1L), Y FE38: D1 42 410 CMP (A4L), Y FE3A: FO 411 BEG VFYOK FE3C: 20 92 FD 412 JSR PRA1 FE3F: B1 3C 413 LDA (A1L), Y FE41: 20 DA FD 414 JSR PRBYTE FE44: A9 A0 415 LDA #\$A0 FE46: 20 ED FD 416 JSR COUT FE49: Α9 A8 417 LDA #\$A8 FE4B: 20 ED FD 418 JSR COUT FE4E: B1 42 419 LDA (A4L), Y FE50: 20 DA FD 420 JSR PRBYTE FE53: Α9 Α9 421 LDA #\$A9 FE55: 20 ED FD 422 JSR COUT FE58: 423 VFYOK 20 B4 FC JSR NXTA4 FE5B: 90 D9 424 BCC VFY FE5D: 60 425 RTS FE5E: 20 75 FE 426 LIST JSR A1PC FE61: A9 14 427 LDA #\$14 FE63: 48 428 LIST2 PHA FE64: 20 DO F8 429 JSR INSTDSP FE67: 20 53 F9 430 JSR PCADJ FE6A: 85 3A 431 STA PCL FE6C: 84 3B 432 STY PCH FE6E: 68 433 PLA FE6F: 38 434 SEC

### Apple 2 Technical Manual • Apple ][ Reference Manual 1979 FE70: E9 01 435 SBC #\$01 DO EF 435 FE72: BNE LIST2 FE74: 60 437 RTS FE75 438 PAGE FE75: 8A 439 A1PC TXA FE76: FO 07 440 BEQ A1PCRTS FE78: B5 441 A1PCLP LDA A1L, X FE7A: 95 3A 442 STA PCL, X FE7C: CA 443 DEX FE7D: 10 F9 444 BPL A1PCLP FE7F: 60 445 AIPCRTS RTS FE80: A0 3F 446 SETINV LDY FE82: DO 02 447 BNE SETIFLG FE84: AO FF 448 SETNORM LDY #\$FF FE86: 84 32 449 SETIFLG STY INVFLG FF88 60 450 RTS FE89: A9 00 451 SETKBD LDA #\$00 FE8B: 85 3E 452 INPORT STA AZL FEBD: A2 38 453 INPRT LDX #KSWL FE8F: AO 1B 454 LDY #KEYIN FE91: DO 08 455 BNE IOPRT FE93: A9 00 456 SETVID LDA #\$00 457 DUTPORT STA A2L FE95: 85 3E FE97: A2 36 458 OUTPRT LDX #CSWL FE99: A0 F0 459 LDY #COUT1 460 IDPRT FE9B: A5 3E LDA A2L FE9D: 29 OF 461 AND #\$OF FE9F: F0 06 462 BEQ IOPRT1 FEA1: 09 CO 463 ORA #IOADR/256 FEA3: A0 00 464 LDY #\$00 FEA5: F0 02 465 BEG IOPRT2 FEA7: A9 FD 466 IOPRT1 LDA #COUT1/256 467 IOPRT2 FEA9: EQU \* FEA9: 94 00 468 STY LOCO, X ; \$94,\$00 FEAB: 95 01 469 STA LOC1, X ; \$95,\$01 FEAD: 60 470 RTS FEAE: 471 NOP FEAF: EΑ 472 NOP FEBO: 4C 00 E0 473 XBASIC JMP BASIC FEB3: 4C 03 E0 474 BASCONT JMP BASIC2 FEB6: 20 75 FE 475 GD JSR A1PC FEB9: 20 3F FF 476 JSR RESTORE FEBC: 6C 3A 00 477 JMP (PCL) FEBF: 4C D7 FA 478 REGZ JMP REGDSP FEC2: 60 479 TRACE RTS FEC3: 480 \* TRACE IS GONE FEC3: EA 481 NOP FEC4: 60 482 STEPZ RTS ; STEP IS GONE FEC5: EA 483 NOP FEC6: EA 484 NOP FEC7: 485 EΑ NOP FEC8: EΑ 486 NOP FEC9: FΑ 487 NOP JMP USRADR PAGE FECA: 4C FB 03 488 USR FECD: 489 A9 40 FECD: 490 WRITE LDA #\$40 20 C9 FC FECF: 491 JSR HEADR FED2: A0 27 492 LDY #\$27 FED4: A2 00 493 WR1 LDX #\$00 FED6: 41 3C 494 EOR (A1L, X) FED8: 48 495 PHA A1 3C FED9: 496 LDA (A1L, X) FEDB: 20 ED FE 497 JSR WRBYTE FEDE: 20 BA FC 498 JSR NXTA1 FEE1: 499 A0 1D LDY #\$1D FEE3: 68 500 PLA FEE4: 90 EE 501 BCC WR1 FEE6: A0 22 502 LDY #\$22 20 ED FE FEE8: 503 JSR WRBYTE FEEB: FO 4D 504 BEG BELL FEED: A2 10 505 WRBYTE LDX #\$10 FEEF: OA 506 WRBYT2 ASL A FEF0: 20 D6 FC 507 JSR WRBIT

### Apple 2 Technical Manual • Apple | Reference Manual • 1979

```
BNE WRBYT2
FEF3: DO FA
                    508
                    509
                                 RTS
FEF5:
      60
                    510 CRMON
                                 JSR BL1
      20 00 FE
FEF6:
FEF9:
      68
                    511
                                 PLA
                                 PLA
FEFA:
      68
                    512
                                 BNE MONZ
FEFB: DO 6C
                    513
                    514 READ
                                 JSR RD2BIT
      20 FA FC
FEFD:
                                 LDA #$16
FF00: A9 16
                    515
                                 JSR HEADR
      20 C9 FC
                    516
FF02:
                                 STA CHKSUM
FF05:
      85 2E
                    517
                                 JSR RD2BIT
      20 FA FC
FF07:
                    518
FFOA:
      A0 24
                    519 RD2
                                 LDY #$24
FFOC:
      20 FD FC
                    520
                                 JSR RDBIT
FFOF:
      BO F9
                    521
                                 BCS RD2
                                 JSR RDBIT
FF11: 20 FD FC
                    522
                    523
                                 LDY
                                     #$3B
FF14:
      A0 3B
                    524 RD3
                                 JSR RDBYTE
      20 EC FC
FF16:
FF19:
      81 3C
                    525
                                 STA (A1L, X)
FF1B:
      45
          2E
                    526
                                 EOR CHKSUM
FF1D: 85 2E
                    527
                                 STA CHKSUM
                                 JSR NXTA1
FF1F: 20 BA FC
                    528
                    529
                                 LDY #$35
FF22:
      A0 35
                    530
                                 BCC RD3
      90 F0
FF24:
      20 EC FC
                                 JSR RDBYTE
FF26:
                    531
                                 CMP
                                     CHKSUM
FF29:
      C5 2E
                    532
FF2B: FO OD
                    533
                                 BEG BELL
FF2D: A9 C5
                    534 PRERR
                                 LDA #$C5
      20 ED FD
                    535
                                 JSR COUT
FF2F:
                    536
                                 LDA #$D2
FF32:
       A9 D2
                                 JSR COUT
       20 ED FD
FF34:
                    537
                                 JSR COUT
FF37: 20 ED FD
                    538
FF3A: A9 87
                    539 BELL
                                 LDA #$87
                                 JMP COUT
FF3C:
       4C ED FD
                    540
FF3F
                    541
                                 PAGE
FF3F: A5 48
                    542 RESTORE LDA STATUS
                    543
                                 PHA
FF41: 48
FF42: A5 45
                    544
                                 LDA A5H
                    545 RESTR1
                                 LDX XREG
FF44:
       A6 46
                                      YREG
FF46:
       A4 47
                    546
                                 LDY
FF48:
                    547
                                 PLP
FF49:
       60
                    548
                                 RTS
FF4A:
       85 45
                    549 SAVE
                                 STA A5H
                    550 SAV1
                                 STX XREG
FF4C:
       86 46
                                 STY
                                      YREG
FF4E:
       84 47
                    551
                                 PHP
FF50:
       08
                    552
FF51:
       68
                    553
                                 PLA
FF52:
       85 48
                    554
                                 STA STATUS
FF54: BA
                    555
                                  TSX
                                  STX SPNT
FF55:
       86 49
                    556
                    557
                                  CLD
FF57:
       D8
FF58:
       60
                    558
                                  RTS
       20 84 FE
                    559 OLDRST
                                 JSR SETNORM
FF59:
 FF5C:
       20 2F FB
                    560
                                  JSR INIT
       20 93 FE
 FF5F:
                     561
                                  JSR SETVID
 FF62:
       20 B9 FE
                     562
                                  JSR SETKBD
 FF65:
                     563
                                  PAGE
                     564 MON
                                 CLD
 FF65:
       D8
                                  JSR BELL
       20 3A FF
                     565
 FF66:
                     566 MONZ
                                 LDA #$AA
 FF69:
       A9 AA
                                  STA PROMPT
 FF6B:
       85 33
                     567
 FF6D:
       20 67 FD
                     568
                                  JSR GETLNZ
       20 C7 FF
                    569
                                  JSR ZMODE
 FF70:
                     570 NXTITM
                                  JSR
 FF73:
       20 A7 FF
                                      GETNUM
                                  STY YSAV
 FF76:
                     571
       84 34
                                  LDY #$17
 FF78:
       AO 17
                     572
                     573 CHRSRCH DEY
 FF7A:
       88
                                  BMI MON
       30 E8
                     574
 FF7B:
                     575
                                  CMP
                                      CHRTBL, Y
 FF7D:
       D9
          CC FF
 FF80:
       DO F8
                     576
                                  BNE CHRSRCH
                     577
                                  JSR TOSUB
       20 BE FF
 FF82:
                     578
                                  LDY YSAV
 FF85:
       A4 34
                                  JMP NXTITM
       4C 73 FF
 FF87:
                     579
                     580 DIG
 FF8A: A2 03
                                  LDX #$03
```

```
Apple 2 Technical Manual •
                                         Apple | Reference Manual
                                                                                1979
FF8C:
      OA
                    581
                                 ASL A
FF8D:
       OA
                    582
                                 ASL A
FFBE:
                    583
      OA
                                 ASL A
FF8F:
      OA
                    584
                                 ASL
                                     Α
FF90:
      OA
                    585 NXTBIT
                                 ASL
FF91:
      26 3E
                    586
                                 ROL AZL
FF93:
      26
          ЗF
                    587
                                 ROL A2H
FF95:
                    588
                                 DEX
FF96:
      10 F8
                    589
                                 BPL NXTBIT
FF78:
      A5 31
                    590 NXTBAS
                                 LDA MODE
FF9A:
      DO 06
                    591
                                 BNE NXTBS2
FF9C:
                    592
FF9C:
      B5 3F
                    593
                                 LDA A2H, X
FF9E:
                    594
FF9E:
      95 3D
                    595
                                 STA A1H, X
FFA0:
                    596 *
FFA0:
      95 41
                    597
                                 X HEA ATE
FFA2:
      E8
                    598 NXTBS2
                                 INX
      F0 F3
FFA3:
                    599
                                 BEQ NXTBAS
FFA5:
      DO 06
                    600
                                 BNE NXTCHR
FFA7:
      A2 00
                    601 GETNUM
                                 LDX #$00
FFA9:
      86 3E
                    602
                                 STX A2L
FFAB:
      86
         3F
                    603
                                 STX A2H
FFAD:
      B9 00 02
                    604 NXTCHR
                                 LDA IN, Y
FFBO:
      CB
                    605
                                 INY
FFB1:
      49 BO
                    606
                                 EOR
                                     #$B0
      C9 0A
FFB3:
                    607
                                 CMP
                                     #$0A
FFB5:
      90 D3
                    806
                                 BCC DIG
FFB7:
      69 88
                    609
                                 ADC
                                     #$88
FFB9:
      C9 FA
                    610
                                 CMP
                                     #$FA
FFBB: BO CD
                                 BCS DIG
                    611
FFBD:
      60
                    612
                                 RTS
FFBE: A9 FE
                    613 TOSUB
                                 LDA #GD/256
FFCO:
      48
                    614
                                 PHA
FFC1:
      B9 E3 FF
                    615
                                 LDA SUBTBL, Y
FFC4:
      48
                    616
                                 PHA
FFC5:
      A5 31
                    617
                                 LDA MODE
FFC7:
      A0 00
                    618 ZMODE
                                 LDY #$00
FFC9: 84 31
                    619
                                 STY MODE
FFCB:
                    620
                                 RTS
      60
FFCC:
                    621
                                 PAGE
FFCC:
      BC
                    622 CHRTBL
                                 DFB $BC
FFCD:
      B2
                    623
                                 DFB $B2
FFCE:
      BE
                    624
                                 DFB $BE
FECE:
      B2
                    625
                                 DFB $B2
                                             ; T CMD NOW LIKE USR
FFD0:
      EF
                    626
                                 DFB $EF
FFD1:
      C4
                    627
                                 DFB $C4
FFD2:
      B2
                    628
                                 DFB $B2
                                             ; S CMD NOW LIKE USR
FFD3:
      Α9
                    629
                                 DFR $A9
FFD4:
      BB
                    630
                                 DFB $BB
FFD5
      Α6
                    631
                                 DFB $A6
FFD6:
      A4
                    632
                                 DFB $A4
FFD7:
      06
                    633
                                 DFB $06
FFD8:
                    634
                                 DFB $95
FFD9:
      07
                    635
                                 DFB $07
FFDA:
      02
                    636
                                 DFB $02
FFDB:
      05
                    637
                                 DFB $05
FFDC:
      FO
                    638
                                 DFB $FO
FFDD:
      00
                    639
                                 DFB $00
FFDE:
      ΕB
                    640
                                 DFB $EB
FFDF:
                    641
                                 DFB $93
FFE0:
      Α7
                    642
                                 DFB $A7
FFE1:
      C6
                    643
                                 DFB $C6
FFE2:
      99
                    644
                                 DFB $99
FFE3: B2
                    645 SUBTBL
                                 DFB $B2
FFE4:
      C9
                    646
                                 DFB $C9
FFE5:
      ΒE
                    647
                                 DFB $BE
FFE6: C1
                   648
                                 DFB $C1
FFE7:
      35
                    649
                                 DFB $35
FFE8: 8C
                   650
                                 DFB $80
FFE9: C4
                   651
                                 DFB $C4
FFEA: 96
                    652
                                 DFB $96
FFEB: AF
                                 DFB $AF
```



## **MONITOR ROM LISTING**

```
APPLE II
            SYSTEM MONITOR
           COPYRIGHT 1977 BY
          APPLE COMPUTER, INC.
          ALL RIGHTS RESERVED
11
               S. WOZNIAK
               A. BAUM
12
13
14
                                   "APPLE II SYSTEM MONITOR"
15
              TITLE
     LOCO
                       S00
16
                  EPZ
17
     LOCl
                  EPZ
                        $01
     WNDLFT
                  EPZ
                        $20
     WNDWDTH
                  EPZ
                        $21
19
20
     WNDTOP
                  EPZ
                        $22
21
     WNDBTM
                        $23
                  EPZ
22
     CH
                  EPZ
                        $24
     СV
23
                  EPZ
                        $25
     GBASL
24
                  EPZ
                        $26
     GBASH
                  EPZ
                        $27
     BASL
                  EPZ
26
                        $28
     BASH
                  EPZ
                        $29
27
28
     BAS 2L
                  EPZ
                        S2A
29
     BAS 2H
                  EPZ
                        $2B
     H2
                  EPZ
                        $2C
     LMNEM
                  EPZ
                        $2C
     RTNL
                  EPZ
                        $2C
33
     V 2
                  EPZ
                        $2D
34
     RMNEM
                  EP2
                        S2D
     RTNH
35
                  EPZ
                        $2D
36
     MASK
                  EPZ
                        $2E
37
     CHKSUM
                  EPZ
                        $2E
     FORMAT
                  EPZ
                        $2E
3.3
     LASTIN
                  EPZ
                        $2F
40
     LENGTH
                  EPZ
                        $2F
41
     SIGN
                  EPZ
                        $2F
     COLOR
42
                  EPZ
                        $30
43
     MODE
                  EPZ
                        $31
44
     INVFLG
                  EPZ
                        $32
45
     PROMPT
                  EPZ
                        $33
46
     YSAV
                  EPZ
                        $34
47
     YSAV1
                  EPZ
                        $35
48
     CSWL
                  EPZ
                        $36
49
     CSWH
                  EPZ
                        $37
     KSWL
                  EPZ
                        $38
     KSWH
                  EPZ
                        S39
52
     PCL
                  EPZ
                        $3A
53
     PCH
                  EPZ
                        $3B
54
55
     XQT
AlL
                  E PZ
E PZ
                        $3C
$3C
56
     AlH
                  EPZ
                        $3D
57
     A2L
                  EPZ
                        $3E
     A2H
                  EPZ
                        $3F
     A3L
                  EPZ
                        $40
                        $41
     АЗН
                  EPZ
óυ
61
     A4L
                  EPZ
                        $42
                        $43
62
     A4H
                  EPZ
     A5L
                  EPZ
                        $44
64
     A 5 H
                  EPZ
                        $45
65
     ACC
                  EPZ
                        $45
     XREG
                  EPZ
                        $46
66
                        $47
     YREG
                  EPZ
67
     STATUS
                  EPZ
```

69 SRNT

Ap	ple 2 Te	echni	cal Manual	•	Apple ][ Re	eference Manual • 1979
F849:	29 03	143		AND	#\$03	
F84B:	09 04	144		ORA	#\$04	GENERATE GBASH=000001FG
F84D:	85 27	145		STA	GBASH	AND CDACI - UDEDE 000
F84F: F850:	68 29 18	146 147		PLA AND	#\$18	AND GBASL=HDEDE000
F852:	90 02	148		BCC	GBCALC	
F854:	69 7F	149		ADC	#\$7F	
F856:	85 26	150	GBCALC	STA	GBASL	
F858:	0A	151		ASL	A	
F859:	0A	152		ASL	A	
F85A: F85C:	05 26 85 26	153 154		ORA STA	GBASL GBASL	
F85E:	60	155		RTS	COMBB	
F85F:	A5 30	156	NXTCOL		COLOR	INCREMENT COLOR BY 3
F861:	18	157		CTC		
F862:	69 03	158	CEMOOL	ADC	#\$03	CDMC COLOD-1743 MOD 16
F864: F866:	29 OF 85 30	159 160	SETCOL		#\$0F COLOR	SETS COLOR=17*A MOD 16
F868:	0A	161			A	BOTH HALF BYTES OF COLOR EQUAL
F869:	0A	162		ASL	A	•
F86A:	0A	163			A	
F86B:	UA UE 2.3	164		ASL	A	
F86C: F86E:	υ5 30 85 30	165 166			COLOR COLOR	
F870:	.60	167		RTS	COLON	
F871:	4A	168	SCRN	LSR	A	READ SCREEN Y-COORD/2
F872:	08	169		PHP		SAVE LSB (CARRY)
F873:	20 47 F8				GBASCALC	CALC BASE ADDRESS
F876: F878:	Bl 26 28	171 172		LDA PLP	(GBASL),Y	GET BYTE RESTORE LSB FROM CARRY
F879:	90 04	173	SCRN2	BCC	RTMSKZ	IF EVEN, USE LO H
F87B:	4A	174			A	2. 2.2., 622 26 1.
F87C:	4A	175			A	
F87D:	4A	176			A	SHIFT HIGH HALF BYTE DOWN
F87E: F87F:	4A 29 OF	177 178	RTMSKZ	LSR AND	A #\$0F	MASK 4-BITS
F881:	60	179	KINDKE	RTS	# 401	HASK 4-5115
F882:	A6 3A	180	INSDS1		PCL	PRINT PCL, H
F884:	A4 3B	181		LDY	PCH	, .
F886:	20 96 FD			JSR	PRYX2	BOLLOUID DV 1 DL1111
F889: F88C:	20 48 F9 Al 3A	184			PRBLNK (PCL,X)	FOLLOWED BY A BLANK GET OP CODE
F83E:	A1 3A	185	INSDS 2	TAY	(FCD, X)	GET OF CODE
F88F:	4A	186			A	EVEN/ODD TEST
F890:	90 09	187		BCC	IEVEN	
F892:	6A	188		ROR	A	BIT 1 TEST
F893: F895:	B0 10 C9 A2	189 190		BCS CMP	ERR #\$A2	XXXXXX11 INVALID OP
F897:	FO UC	191		BEQ	ERR	OPCODE \$89 INVALID
F899:		192			#\$87	MASK BITS
F89B:	4A		IEVEN	LSR	A	LSB INTO CARRY FOR L/R TEST
F89C:		194		TAX	อพกา v	CET ECOMAT INDEX SAME
	BD 62 F9 20 79 F8			JSR	FMT1,X SCRN2	GET FORMAT INDEX BYTE R/L H-BYTE ON CARRY
F8A3:		197		BNE	GETFMT	My Z II DIII ON GIMMI
F8A5:			ERR	LDY		SUBSTITUTE \$80 FOR INVALID OPS
F8A7:		199		LDA	#\$0	SET PRINT FORMAT INDEX TO 0
F8A9:			GETFMT	TAX	mama v	THEORY INTO DEINT ECONAG GARIE
F8AA: F8AD:	BD A6 F9	201		LUA STA		INDEX INTO PRINT FORMAT TABLE SAVE FOR ADR FIELD FORMATTING
F8AF:		203		AND	#\$03	MASK FOR 2-BIT LENGTH
		204	*			BYTE, 2=3 BYTE)
F8B1:		205		STA	LENGTH	
F8B3:		206		TYA	"COP	OPCODE
F8B4: F8B6:	29 8F AA	207 208		AND TAX	#\$8F	MASK FOR 1XXX1010 TEST SAVE IT
F8B7:	98	208		TYA		OPCODE TO A AGAIN
F8B8:	AU 03	210			#\$03	
F8BA:	E0 8A	211		CFX	#\$8A	
F8BC:		212			MNNDX3	
F8BE:		213 214	MNNDX1	LSR		FORM INDEX INTO MNEMONIC TABLE
	90 08 4A	214		LSR		TORM INDEX INTO MAGMONIC INDEE

Ар	ple 2 Techni	cal Manual	•	Apple ][ Re	ference Manual • 1979
F8C2:	4A 216	MNNDX2	LSR	A	1) 1xxx1010=>001v1xxx
F8C3:	09 20 217		ORA	#\$2U	2) XXXYYY01=>00111XXX
F8C5:	88 218		DEY		3) XXXYYY10=>00110XXX
F8C6:			BNE	MNNDX2	4) XXXYY100=>00100XXX
F8C8:			INY		5) XXXXX0∪0=>000XXXXX
F8C9:	88 221		DEY		
F8CA:			BNE	MNNDX1	
F8CC: F8CD:			RTS	cen eno eno	
F8D0:			DFB	SFF, SFF, SFF	CEN RMD LON DVMDC
F8D3:			JSR PHA	INSDS1	GEN FMT, LEN BYTES SAVE MNEMONIC TABLE INDEX
F8D4:	B1 3A 227			(PCL),Y	SAVE MALMONIC TABLE TABLE
F8D6:	20 DA FD 228			PRBYTE	
F8D9:	A2 01 229		LDX	#\$01	PRINT 2 BLANKS
F8DB:			JSR	PRBL2	
F8DE:				LENGTH	PRINT INST (1-3 BYTES)
F8E0:			INY	DDMMOD	IN A 12 CHR FIELD
F8E1:	90 F1 233 A2 03 234			PRNTOP	CHAD COUNT FOR MAIDMONIC PRIME
F8E5:			CPY	#\$03 #\$04	CHAR COUNT FOR MNEMONIC PRINT
	90 F2 236		BCC	PRNTBL	
F8E9:			PLA		RECOVER MNEMONIC INDEX
F8EA:			TAY		
F8EB:	B9 C0 F9 239		LDA	MNEML, Y	
F8EE:			STA	LMNEM	FETCH 3-CHAR MNEMONIC
F8F0:	B9 00 FA 241		LDA	MNEMR, Y	(PACKED IN 2-BYTES)
F8F3:				RMNEM	
F8F5:			LDA	#\$00	
F8F7: F8F9:				#\$05	CUIDE E DIEC OF
F8FB:	06 2D 245 26 2C 246			RMNEM LMNEM	SHIFT 5 BITS OF CHARACTER INTO A
F8FD:			ROL	A	(CLEARS CARRY)
F8FE:	88 248		DEY	••	(CDDING CIMMI)
F8FF:	DU F8 249			PRMN2	
F901:	69 BF 250			#\$BF	ADD "?" OFFSET
F903:	20 ED FD 251	,	JSR	COUT	OUTPUT A CHAR OF MNEM
F906:	CA 252		DEX		
F907:	D0 EC 253			PRMN1	
F909:	20 48 F9 254			PRBLNK	OUTPUT 3 BLANKS
F90C: F90E:				LENGTH	CME BOD ( BODMAN DIEG
F910:			LDX CPX	#\$06 #\$03	CNT FOR 6 FORMAT BITS
F912:	FO 1C 258			PRADR5	IF X=3 THEN ADDR.
F914:	06 2E 259			FORMAT	It X-3 INDIA ADDR.
F916:	90 UE 260			PRADR3	
F918:	BD B3 F5 '61		LDA	CHAR1-1,X	
F91B:	20 ED FD 262		JSR	COUT	
F91E:	BD B9 F9 263			CHAR2-1,X	
F921:	F0 03 264			PRADR3	
	20 ED FD 265			COUT	
F920:	CA 266 DU E7 267		DEX	ומתגמם	
F929:			BNE RTS	PRADR1	
F92A:			DEY		
	30 E7 270			PRADR2	
	20 DA FD 271			PRBYTE	
				FORMAT	
	C9 E8 273				HANDLE REL ADR MODE
	B1 3A 274				SPECIAL (PRINT TARGET,
	90 F2 275	DELADO			NOT OFFSET)
F938:	20 56 F9 276 AA 277			PCADJ3	DCI DCU_OppCpm_1 mo x v
F93C •	E8 278		TAX INX		PCL, PCH+OFFSET+1 TO A, Y
	DU 01 279			PRNTYX	+1 TO Y.X
F93F:			INY	• ••	,
F940:			TYA		
	20 DA FD 282	PRN'TAX		PRBYTE	OUTPUT TARGET ADR
			TXA		OF BRANCH AND RETURN
	4C DA FD 284	DDD 5	JMP	PRBYTE	
		PRBLNK :	LDX		BLANK COUNT
	A9 A0 286 20 ED FD 287	PRBL2	LUA	# \$ A U	LOAD A SPACE
F94C:			DEX	COUT	OUTPUT A BLANK
	200		JUA		

```
Apple 2 Technical Manual
                                         Apple | Reference Manual
                                                                                1979
F950:
       DU F8
                 289
                                   BNE
                                         PRBL2
                                                       LOOP UNTIL COUNT=0
F952:
       60
                 290
                                   RTS
                                                       0=1-BYTE, 1=2-BYTE,
F953:
       38
                 291
                       PCADJ
                                   SEC
F954:
       A5 2F
                 292
                       PCADJ2
                                   LDA
                                         LENGTH
                                                         2=3-BYTE
F956:
       A4 38
                 293
                       PCADJ3
                                   LDY
                                         PCH
F958:
       AA
                 294
                                   TAX
                                                       TEST DISPLACEMENT SIGN
F959:
       10 01
                 295
                                   BPL
                                        PCADJ4
                                                          (FOR REL BRANCH)
F958:
                 296
                                   DEY
                                                       EXTEND NEG BY DECR PCH
       88
       65 3A
                       PCADJ4
                                        PCL
F95C:
                 297
                                   ADC
                                                       PCL+LENGTH(OR DISPL)+1 TO A
F95E:
       90 01
                 298
                                   BCC
                                        RTS2
F960:
       C٤
                 299
                                   INY
                                                         CARRY INTO Y (PCH)
                       RTS 2
                                   RTS
F961:
       60
                 300
                  301
                                   FMT1 BYTES:
                                                         XXXXXXYO INSTRS
                                                         THEN LEFT HALF BYTE
                 302
                                   IF Y=0
                                   IF Y=1
                                                         THEN RIGHT HALF BYTE
                 303
                                                               (X = INDEX)
                 304
F962:
       04 20 54
F965:
       30 UD
                       FMT1
                                   DFB
                                        $04,$20,$54,$
       80 04 90
F967:
F96A:
       03 22
                                   DFB
                                        $80,$04,$90,$
                 306
       54 33 UD
F96C:
F96F:
       80 04
                                   DFB
                                        $54,$33,$0D,$
                 307
       90 04 20
F971:
F974:
       54 33
                 308
                                   DFB
                                        $90,$04,$20,$
F976:
       UD 80 04
F979:
       90 u4
                 309
                                   DFB
                                        $0D,$80,$04,$
F97B:
       20 54 3B
                 310
                                   DFB
                                        $20,$54,$38,$
F97E:
       0E 30
F980:
       04 90 00
F983:
       22 44
                                   DFB
                                         $04,$90,$00,$
F985:
       33 UD C8
F938:
       44 00
                                        $33,$0D,$C8,$
                                   DFB
       11 22 44
F98A:
F98D:
       33 UD
                                   DFB
                                        $11,$22,$44,$
                 313
       C8 44 A9
F98F:
F992:
       01 22
44 33 0D
                                   DFB
                                         $C8,$44,$A9,$
F994:
F997:
       80 04
                 315
                                   DFB
                                         $44,$33,$0D,$
F999:
       90 01 22
       44 33
                                   DFB
                                        $90,$01,$22,$
F99C:
                 316
F99E:
       0D 80 04
F9A1:
       90
                 317
                                   DFB
                                        $0D,$80,$04,$
F9A2:
       26 31 87
F9A5:
       9A
                 318
                                   DFB
                                         $26,$31,$87,$ZZXXXY01 INSTR'S
F9A6:
       00
                  319
                       FMT2
                                   DFB
                                         $00
                                                       ERR
F9A7:
                                         $21
                                                       IMM
       21
                  320
                                   DFB
F9A8:
       81
                 321
                                   DFB
                                         S81
                                                       Z-PAGE
F9A9:
       82
                  322
                                   DFR
                                         S82
                                                       ABS
F9AA:
       00
                 323
                                   DFB
                                         S00
                                                       IMPLIED
F9AB:
       00
                  324
                                   DFB
                                         SÚO
                                                       ACCUMULATOR
F9AC:
       59
                 325
                                   DFB
                                         $59
                                                       (ZPAG,X)
F9AD:
        4 D
                  326
                                   DFB
                                         $4D
                                                       (ZPAG),Y
F9AE:
        91
                  327
                                   DFB
                                         $91
                                                       ZPAG, X
F9AF:
                  328
                                   DFB
                                         $92
                                                       ABS,X
F9B0:
                                   DFB
                                         $86
                                                       ABS, Y
       86
                 329
F9B1:
        4A
                 330
                                   DFB
                                         S4A
                                                       (ABS)
F9B2:
       85
                 331
                                   DFB
                                         $85
                                                       ZPAG, Y
F9B3:
       9D
                 332
                                   DFB
                                         $9D
                                                       RELATIVE
       AC A9 AC
F9B4:
F9B7:
       A3 A8 A4 333
                       CHAR1
                                   ASC
                                        ",),#($"
F9BA:
       D9 UU D8
                                  DFB $D9,$00,$D8,$
"Y",0,"X$$",0
F9BD:
       A4 A4 00
                 334
                       CHAR2
                  335
                       *CHAR2:
                                  MNEML
                                                 IS OF FORM:
                  336
                                       XXXXX000
                  337
                                  (A)
                  338
                                   (B)
                                       XXXYY100
                  339
                                  (C)
                                        1XXX1010
                  34Û
                                  (D)
                                       XXXYYY10
                  341
                                       XXXYYY01
                                  (E)
                  342
                                        (X = INDEX)
F9C0:
       1C 8A 1C
                                   DFB $1C,$8A,$1C,$
F9C3:
       23 5D 8B
                 343
                      MNEML
F9C6:
       1B A1 9D
```

### Apple 2 Technical Manual • Apple | Reference Manual • 1979 F9C9: 8A 1D 23 344 DFB \$1B,\$A1,\$9D,\$ F9CC: 9D 8B 1D F9CF: Al 00 29 345 DFB \$9D,\$8B,\$1D,\$ F9D2: 19 AE 69 F9D5: A8 19 23 346 DFB \$19,\$AE,\$69,\$ F9D8: 24 53 1B DFB \$24,\$53,\$1B,\$ DFB \$19,\$A1 F9DB: 23 24 53 347 (A) FORMAT ABOVE F9DE: 19 Al 348 F9E0: 00 1A 5B F9E3: 5B A5 69 349 DFB \$00,\$1A,\$5B,\$ F9E6: 24 24 350 DFB \$24,\$24 (B) FORMAT F9E8: AE AE A8 F9EB: AD 29 00 351 DFB \$AE,\$AE,\$A8,\$ F9EE: 7C 00 352 DFB \$7C,\$00 (C) FORMAT F9F0: 15 9C 6D F9F3: 9C A5 69 353 DFB \$15,\$9C,\$6D,\$ F9Fo: 29 53 DFB \$29,\$53 (D) FORMAT 354 F9F8: 84 13 34 F9FB: 11 A5 69 355 DFB \$84,\$13,\$34,\$ F9FE: 23 A0 356 DFB \$23,\$A0 (E) FORMAT FA00: D8 62 5A FA03: 48 26 62 357 MNEMR DFB \$D8,\$62,\$5A,\$ FA'06: 94 88 54 FA09: 44 C8 54 358 DFB \$94,\$88,\$54,\$ FAUC: 68 44 E8 94 UU B4 359 FAOF: DFB \$68,\$44,\$E8,\$ FA12: 08 84 74 FA15: B4 28 6E 360 DFB \$08,\$84,\$74,\$ FA18: 74 F4 CC FAlB: 4A 72 F2 361 DFB \$74,\$F4,\$CC,\$ FAlE: A4 8A 362 DFB \$A4,\$8A (A) FORMAT FA20: 00 AA A2 FA23: A2 74 74 363 DFB \$00,\$AA,\$A2,\$ FA26: 74 72 364 DFB \$74,\$72 (B) FORMAT FA 28: 44 68 B2 FA2B: 32 B2 JU 365 DFB \$44,\$68,\$B2,\$ FA2E: 22 00 DFB \$22,\$00 (C) FORMAT FA30: 1A 1A 26 FA33: 26 72 72 367 DFB \$1A,\$1A,\$26,\$ FA36: 88 C8 368 DFB \$88,\$C8 (D) FORMAT FA38: C4 CA 26 48 44 44 369 FA3B: DFB \$C4,\$CA,\$26,\$ FA3E: A2 C8 370 DFB \$A2,\$C8 (E) FORMAT FF FF FF 371 FA40: SFF, SFF, SFF FA43: 20 D0 F8 372 STEP JSR INSTDSP DISASSEMBLE ONE INST FA46: 373 68 PLA AT (PCL,H) FA47: 85 2C 374 STA RTNL ADJUST TO USER FA49: 375 68 PLA STACK. SAVE FA4A: 85 2D 376 STA RTNH RTN ADR. FA4C: A2 08 377 LDX FA4E: BD 10 FB 378 XQINIT LDA INITBL-1,X INIT XEQ AREA FA51: 95 3C 379 STA XQT,X FA53: CA 380 DEX FA54: D0 F8 381 BNE XQINIT FA56: Al 3A 382 LDA (PCL,X) USER OPCODE BYTE FA58: Fû 42 383 BEQ XBRK SPECIAL IF BREAK FA5A: A4 2F 384 LDY LENGTH LEN FROM DISASSEMBLY C9 20 FA5C: 385 CMP #\$20 FA5E: FO 59 386 BEQ HANDLE JSR, RTS, JMP, XJSR FA60: C9 60 387 CMP #\$60 JMP ( ), RTI SPECIAL FA62: Fu 45 388 BEQ XRTS FA64: C9 4C 389 CMP #\$4C FA66: F0 5C 390 BEQ XJMP FA68: C9 6C 391 CMP #\$6C FA6A: FU 59 392 **XJMPAT** BEC FA6C: C9 40 393 CMP #S40 FA6E: F0 35 394 BEQ XRTI FA70: 29 1F 395 AND #\$1F FA72: 49 14 396 ECR #\$14 FA74: C9 04 397 CMP COPY USER INST TO XEQ AREA #\$04 WITH TRAILING NOPS CHANGE REL BRANCH FA76: F0 02 398 BEQ XQ2 FA78: B1 3A 399 XQ1 FA7A: 99 3C 00 400 XQ2 (PCL),Y STA XQTNZ, Y DISP TO 4 FOR

Ар	ple 2 Tec	hnic	al Manual	•	Apple ][ Re	ference Manual • 1979
FA7D: FA7E:		401 402		DEY BPL	XQ1	JMP TO BRANCH OR NBRANCH FROM XEQ.
FA80:	20 3F FF			JSR		RESTORE USER REG CONTENTS.
FA83:	4C 3C 00			JMP		XEQ USER OP FROM RAM
FA86:			IRQ	STA	ACC	(RETURN TO NBRANCH)
FA88:		406		PLA		
FA89:		407		PHA	•	**IRQ HANDLER
FASA:		408			A	
FA8B:		409		ASL	A	
FA8C: FA8D:		410 411		ASL BMI	A BREAK	TEST FOR BREAK
FASE:	6C FE 03			JMP	(IRQLOC)	USER ROUTINE VECTOR IN RAM
FA92:	28		BREAK	PLP	(======)	
FA93:	20 4C FF	414		JSR	SAV1	SAVE REG'S ON BREAK
FA96:	68	415		PLA		INCLUDING PC
FA97:	85 3A	416		STA	PCL	
FA99:		417		PLA	DCH.	
FA9A:	85 3B 20 82 F8	418	XBRK	STA JSR	PCH INSDS1	PRINT USER PC.
FA9C: FA9F:	20 02 F0 20 DA FA		ADAK	JSR	RGDS P1	AND REG'S
FAA2:	4C 65 FF			JMP	MON	GO TO MONITOR
FAA5:	18		XRTI	CLC		
FAA6:	68	423		PLA		SIMULATE RTI BY EXPECTING
FAA7:	85 48	424		STA	STATUS	STATUS FROM STACK, THEN RTS
FAA9:	68	425	XRTS	PLA	201	RTS SIMULATION
FAAA:	85 3A	426		STA	PCL	EXTRACT PC FROM STACK AND UPDATE PC BY 1 (LEN=0)
FAAC: FAAD:	68 65 20	427 428	PCINC2	PLA STA	PCH	AND OFDATE FC BI I (EEN-O)
FAAF:	85 3B A5 2F		PCINC3		LENGTH	UPDATE PC BY LEN
FAB1:	20 56 F9		10100		PCADJ3	
FAB4:	84 3B	431		STY	PCH	
FAB6:	18	432		CLC		
FAB7:	90 14	433	w.7.0.B	BCC	NEWPCL	
FAB9:	18	434	XJSR	CLC	PCADJ2	UPDATE PC AND PUSH
FABA: FABD:	20 54 F9 AA	436		JSR TAX	FCADU Z	ONTO STACK FOR
FABE:	98	437		TYA		JSR SIMULATE
FABF:	48	438		PHA		
FAC0:	8A	439		TXA		
FAC1:	48	440		PHA		
FAC2:	A0 02	441		LDY	#\$02	
FAC4: FAC5:	18	442	XJMP	C L C L D A	(PCL),Y	
FAC7:	Bl 3A AA	443 444	XJMPAT	TAX	(FCD),I	LOAD PC FOR JMP,
FAC8:	88	445		DEY		(JMP) SIMULATE.
FAC9:	B1 3A	446		LDA	(PCL),Y	` ,
FACB:	86 3B	447		STX	PCH	
FACD:	85 3A	448	NEWPCL	STA	PCL	
FACF:	B0 F3	449	5:m): 7): 5	BCS	XJMP	
	A5 2D		RTNJMP		RTNH	
	48 A5 2C	451 452		PHA	RTNL	
FAD6:	48	453		PHA	3- <b>-</b>	
FAD7:	20 8E FD		REGDSP		CROUT	DISPLAY USER REG
FADA:	A9 45		RGDS P1		#ACC	CONTENTS WITH
FADC:	85 40	456			A3L	LABELS
FADE:	A9 00	457		LDA	#ACC/256	
FAE0: FAE2:	85 41 A2 FB	458 459			A3H #\$FB	
FAE 4:	A2 FB A9 A0		RDSP1		#\$PB #\$A0	
FAE6:	20 ED FD				COUT	
FAE9:	BD 1E FA				RTBL-SFB, X	
FAEC:	20 ED FD				COUT	
FAEF:	A9 BD	464			#\$BD	
FAF1:	20 ED FD				COUT	
FAF4: FAF6:	B5 4A 20 DA FD	466 467			ACC+5,X PRBYTE	
FAF9:	E8	468		INX		
FAFA:	30 E8	469			RDSPl	
FAFC:	60	470		RTS		
FAFD:	18		BRANCH	CLC		BRANCH TAKEN,
FAFE:	AU 01	472			#\$01 (PÇL),Y	ADD LEN+2 TO PC
FBOU:	Bl 3A	473		חחמ	(E CH) , I	

App	ole 2 Tec	hnica	al Manual	•	Apple ][ F	Reference Manual • 1979
FB02:	20 56 F			JSR	PCADJ3	
FB05:	85 3A	475		STA	PCL	
FBU7: FB08:	98 38	476 477		TYA		
FB09:	BU A2	478		SEC BCS	PCINC2	
FBUB:	20 4A F		NBRNCH	JSR		NORMAL RETURN AFTER
FBUE:	38	480		SEC	_	XEQ USER OF
FBUF:	B0 9E	481	T., T., T.	BCS	PC INC 3	GO UPDATE PC
FB11: FB12:	EA EA	482 483	INITBL	NOP NOP		DUMANU DATA DO-
FB13:	4C UB FE			JMP	NBRNCH	DUMMY FILL FOR
FB16:	4C FD FA			JMP	BRANCH	XEÇ AREA
FB19:	C1	486	RTBL	DFB	\$C1	
FBlA: FBlB:	D8 D9	487		DFB	\$D 8	
FB1C:	DÚ	488 489		DFB DFB	\$D9	
FB1D:	D3	490		DFB	\$D0 \$D3	
FBlE:	AD 70 Cu		PREAD	LDA	PTRIG	TRIGGER PADDLES
FB21:	A0 00	492		LDY	#\$00	INIT COUNT
FB23: FB24:	EA EA	493 494		NOP		COMPENSATE FOR 1ST COUNT
FB25:	BD 64 CU	495	PREAD2	NOP LDA	PADDL0,X	COUNT V-DEC EVERY
FB28:	10 04	496		BPL	RTS2D	COUNT Y-REG EVERY 12 USEC
FB2A:	C8	497		INY		
FB2B: FB2D:	DU F8 88	498		BNE	PREAD2	EXIT AT 255 MAX
FB2E:	60	499 500	RTS 2D	DEY RTS		
FB2F:	A9 00	501	INIT	LDA	#\$00	CLR STATUS FOR DEBUG
FB31:	85 48	502		STA	STATUS	SOFTWARE
FB33: FB36:	AD 56 C0 AD 54 C0			LDA	LORES	
FB39:	AD 51 C0		SETTXT	LDA LDA	LOWSCR TXTSET	INIT VIDEO MODE
FB3C:	A9 00	506	SELIMI	LDA	#\$ÛU	SET FOR TEXT MODE FULL SCREEN WINDOW
FB3E:	FÚ ÚB	507		BEQ	SETWND	
FB40: FB43:	AD 50 C0 AD 53 C0	508	SETGR	LDA	TXTCLR	SET FOR GRAPHICS MODE
FB46:	20 36 F8	510		LDA JSR	MIXSET CLRTOP	LOWER 4 LINES AS TEXT WINDOW
FB49:	A9 14	511		LDA	#\$14	IBMI WINDOW
FB48:	85 22	512	SETWND	STA	WNDTOP	SET FOR 40 COL WINDOW
FB4D: FB4F:	A9 00 85 20	513 514		LDA	#\$00	TOP IN A-REG,
FB51:	A9 28	515		STA LDA	WNDLFT #\$28	BTTM AT LINE 24
FB53:	85 21	516		STA	WNDWDTH	
FB55: FB57:	A9 18	517		LDA	#\$18	
FB59:	85 23 A9 17	518 519		STA	WNDETM	VTAB TO ROW 23
FB5B:	85 25	520	TAB V	LDA STA	#\$17 CV	VTABS TO ROW IN A-REG
FB5D:	4C 22 FC	521		JMP	VTAB	TINDS TO NOW IN A-NEG
FBóú:	20 A4 FB		MULPM	JSR	MD1	ABS VAL OF AC AUX
FB63: FB65:	A0 10 A5 50	523 524	MUL MUL2		#\$10	INDEX FOR 16 BITS
FB67:	4A	525	.1002	LDA LSR	ACL A	ACX * AUX + XTND TO AC, XTND
FB68:	90 UC	526		BCC	MUL4	IF NO CARRY,
FB6A: FB6B:	18	527		CTC		NO PARTIAL PROD.
FB6D:	A2 FE B5 54	528 529	MUL3	LDX	#\$FE XTNDL+2,X	ADD MDI CND (AUV)
FB6F:	75 56	530		ADC		ADD MPLCND (AUX) TO PARTIAL PROD
FB71:	95 54	531		STA	XTNDL+2,X	(XTND).
FB73: FB74:	E8 DU F7	532 533		INX	MUT 2	
FB76:	A2 U3		MUL4	B NE LDX	MUL3 #\$03	
FB78:	76	535	MUL5	DFB	#\$03 #\$76	
FB79:	50	536		DFB	#\$50	
FB7A: FB7B:	CA 10 FB	537		DEX		
FB7D:	10 FB	538 539		B P L D E Y	MUL5	
FB7E:	DU E5	540		BNE	MUL2	
FB80:	60	541		RTS		
FB81: FB84:	20 A4 FB		DIVPM		MD1	ABS VAL OF AC, AUX.
FB86:	AU 1U U6 5U	543 544	DIV DIV2	LDY ASL	#\$10 ACL	INDEX FOR 16 BITS
FB88:	26 51	545			ACH	
FB8A:	26 52	54 b		ROL	XTNDL	XTND/AUX

					. P. P. II	Reference Manual • 1979
FB8C: 2	26 53	547		ROL	XTNDH	TO AC.
	38	548		SEC		
	A5 52	549		LDΛ	XTNDL	MOD TO XTND.
	E5 54	550		SBC	AUXL	MCD TO ATMD.
	AA A5 53	551 552		TAX LDA	XTNDH	
	E5 55	553		SBC	AUXH	
	90 06	554		BCC	DIV3	
	86 52	555		STX	XTNDL	
	85 53	556		STA	XTNDH	
FB9E: I	E6 50	557		INC	ACL	
	88	558	DIA3	DEY	2 7 7 7 7	
	DU E3	559		BNE RTS	DIV2	
	60 AU 00	560 561	MDl	LDY	#\$00	ABS VAL OF AC, AUX
	84 2F	562	1101	STY	SIGN	WITH RESULT SIGN
	A2 54	563		LDX	#AUXL	IN LSB OF SIGN.
		FB 564		JSR	MD2	
	A2 50	565		LDX	#ACL	
	B5 01	566	MD2	LDA	LOC1,X	X SPECIFIES AC OR AUX
	10 0D	567		B P L SEC	MDRTS	
	38 98	568 569	MD3	TYA		
	F5 00	570		SBC	LOC0,X	COMPL SPECIFIED REG
	95 0ú	571		STA	LOC0,X	IF NEG.
FBB9:	98	572		TYA		
	F5 01	573		SBC	LOC1,X	
	95 01	5 <b>74</b>		STA INC	LOC1,X SIGN	
	E6 2F 60	575 576	MDRTS	RTS	SIGN	
	48	577	BASCALC	PHA		CALC BASE ADR IN BASL, H
	4A	578		LSR	A	FOR GIVEN LINE NO.
FBC3:	29 03	579		AND	#\$03	0<=LINE NO.<=\$17
	09 04	580		ORA	#\$04	ARG=000ABCDE, GENERATE
	85 29	581		STA	BASH	BASH=000001CD
	68 29 18	582 583		PLA AND	#\$18	AND BASL=EABAB000
	90 02	584		BCC	BSCLC2	DAG E-BADADO O O
	69 7F	585		ADC	#\$7F	
	85 28	586	BSCLC2	STA	BASL	
	0A	587		ASL	A	
	UA	588		ASL	A	
	05 28 85 28	589 590		ORA S'TA	BASL BASL	
	60	591		RTS	2421	
	C9 87	592	BELL1	CMP	#\$87	BELL CHAR? (CNTRL-G)
	D0 12	593		BNE	RTS2B	NO, RETURN
	A9 40	594		LDA	#\$40	DELAY .01 SECONDS
		FC 595		JSR	#SCO	
FBE2: FBE4:		596 597	BELL2	LDY	#\$C0 #\$0C	TOGGLE SPEAKER AT
FBE6:					WAIT	1 KHZ FOR .1 SEC.
FBE9:	AD 30	CO 599			SPKR	
FBEC:	ខន	600		DEY		
FBED:			- ma 2-		BELL2	
	60		RTS 2B	RTS	CH	CURSER H INDEX TO Y-REG
FBF0: FBF2:	91 28	603 604	STOADV		(BASL),Y	
	E6 24		ADVANCE			INCREMENT CURSER H INDEX
	A5 24			LDA	CH CH	(MOVE RIGHT)
FBF8:	C5 21	607		CMP	WNDWDTH	BEYOND WINDOW WIDTH?
	B0 66	608			CR	YES CR TO NEXT LINE NO, RETURN
	60		RTS 3 VIDOUT	RTS	#\$AU	CONTROL CHAR?
	C9 A0 B0 EF	611		BCS		NO, OUTPUT IT.
	A8	612		TAY		INVERSE VIDEO?
	10 EC	613			STOADV	YES, OUTPUT IT.
FC04:	C9 8D	614			#\$8D	CR?
	F0 5A				CR 4 C B X	YES. LINE FEED?
FC08:					#\$8A LF	IF SO, DO IT.
FCUA: FCOC:					#\$88	BACK SPACE? (CNTRL-H)
					BELL1	NO, CHECK FOR BELL.
FCOE:	שט טע	017				

Ар	ple 2 Te	chnic	cal Manual	•	Apple ][ R	eference Manual • 1979
FC10:	C6 24	620	BS	DEC	СН	DECORMENT CURCED II TURGV
FC12:	10 E8	621	55	BPL	RTS3	DECREMENT CURSER H INDEX IF POS, OK. ELSE MOVE UP
FC14:	A5 21	622		LDA	WNDWDTH	SET CH TO WNDWDTH-1
FC16:	85 24	623		STA	CH	out ch to Madadin-1
FC18:	C6 24	624		DEC	CH	(RIGHTMOST SCREEN POS)
FC1A:	A5 22	625	UP	LDA	WNDTOP	CURSER V INDEX
FC1C:	C5 25	626		CMP	CV	
FC1E:	BU JB	627		BCS	RTS4	IF TOP LINE THEN RETURN
FC20: FC22:	C6 25	628	uma n	DEC	CV	DECR CURSER V-INDEX
FC22:	A5 25 20 C1 FB	629	VTAB VTABZ	LDA	CV	GET CURSER V-INDEX
FC27:	65 20	631	V 1 42 4	JSR ADC	BASCALC WNDLFT	GENERATE BASE ADDR
FC29:	85 28	632		STA	BASL	ADD WINDOW LEFT INDEX TO BASL
FC2B:	60	633	RTS 4	RTS	מאמה	10 babb
FC2C:	49 CO	634	ESC1	EOR	#\$C0	ESC?
FC2E:	FO 28	635		BEQ	HOME	IF SO, DO HOME AND CLEAR
FC30:	69 FD	636		ADC	#\$FD	ESC-A OR B CHECK
FC32:	90 C0	637		BCC	ADVANCE	A, ADVANCE
FC34:	FO DA	638		BEQ	BS	B, BACKSPACE
FC36:	69 FD	639		ADC	#\$FD	ESC-C OR D CHECK
FC38: FC3A:	90 2C F0 DE	640		BCC	LF	C, DOWN
FC3C:	69 FD	641 642		BEQ ADC	UP #\$FD	D, GO UP ESC-E OR F CHECK
FC3E:	90 5C	643		BCC	CLREOL	E, CLEAR TO END OF LINE
FC40:	D0 E9	644		BNE	RTS4	NOT F, RETURN
FC42:	A4 24	645	CLREOP	LDY	СН	CURSOR H TO Y INDEX
FC44:	A5 25	646		LDA	CV	CURSOR V TO A-REGISTER
FC46:	48	647	CLEOP1	PHA		SAVE CURRENT LINE ON STK
FC 47:	20 24 FC			JSR	VTABZ	CALC BASE ADDRESS
FC4A:	20 9E FC			JSR	CLEOLZ	CLEAR TO EOL, SET CARRY
FC4D:	AU 00	650		LDY	#\$00	CLEAR FROM H INDEX=0 FOR REST
FC4F: FC50:	68 69 UU	651 652		PLA	F ¢ 0 0	INCREMENT CURRENT LINE
FC52:	C5 23	653		ADC CMP	#\$00 WNDBTM	(CARRY IS SET)
FC54:	90 F0	654		BCC	CLEOPI	DONE TO BOTTOM OF WINDOW? NO, KEEP CLEARING LINES
FC56:	BO CA	655		BCS	VTAB	YES, TAB TO CURRENT LINE
FC58:	A5 22	656	HOME	LDA	WNDTOP	INIT CURSOR V
FC5A:	୪5 25	657		STA	CV	AND H-INDICES
FC5C:	AU 00	658		LDY	#\$00	
FC5E:	84 24	659		STY	СН	THEN CLEAR TO END OF PAGE
FC60:	FO E4	660	<b>G</b> 2	BEQ	CLEOP1	
FC62: FC64:	A9 00 85 24	661	CR	LDA	#\$00 07	CURSOR TO LEFT OF INDEX
FC66:	E6 25	662 663	LF	STA INC	CH CV	(RET CURSOR H=0)
FC68:	A5 25	664	DL	LDA	CV	INCR CURSOR V(DCWN 1 LINE)
FC6A:	C5 23	665		CMP	WNDBTM	OFF SCREEN?
FC6C:	90 B6	666		BCC	VTAEZ	NO, SET BASE ADDR
FC6E:	C6 25	667		DEC	CV	DECR CURSOR V(BACK TO BOTTOM)
FC70:	A5 22	668	SCROLL	LDA	WNDTOP	START AT TOP OF SCRL WNDW
FC72:	48	669		PHA		
FC73:	20 24 FC				VTABZ	GENERATE BASE ADDRESS
FC76:	A5 28	671	SCRL1		BASL	COPY BASL, H
FC78: FC7A:	85 2A A5 29	672 673			BAS2L	TO BAS2L, H
FC7C:	85 2B	674		STA	BASH BAS2H	
FC7E:	A4 21	675		LDY		INIT Y TO RIGHTMOST INDEX
FC80:	88	676		DEY	"""	OF SCROLLING WINDOW
FC81:	68	677		PLA		1 2011022110 11212011
FC82:	69 01	678			#\$01	INCR LINE NUMBER
FC84:	C5 23	679	•	CMP	WNDBTM	DONE?
FC86:	B0 0D	680		BCS	SCRL3	YES, FINISH
FC88:	48	681		PHA		
FC89: FC8C:	20 24 FC Bl 28		SCRL2		VTABZ	FORM BASL, H (BASE ADDR)
FC8E:	91 2A	683 684	SCKEZ	LDA STA	(BASL),Y (BAS2L),Y	MOVE A CHR UP ON LINE
FC90:	88 88	685		DEY	(00020),1	NEXT CHAR OF LINE
FC91:	10 F9	686		BPL	SCRL2	MEAT CHAIN OF BINE
FC93:	30 E1	687		BMI		NEXT LINE
FC95:	AU 00	688	SCRL3		#\$00	CLEAR BOTTOM LINE
FC97:	20 9E FC				CLEOLZ	GET BASE ADDR FOR BOTTOM LINE
FC9A:	BU 86	690		BCS	VTAB	CARRY IS SET
FC9C:	A4 24	691	CLREOL	LDY		CURSOR H INDEX
FC9E:	A9 A0	692	CLEOLZ	LDA	#\$A0	

Apple 2 Tec	nnical Manua	•	Apple ][ R	eference Manual • 1979
FCAu: 91 28 6	93 CLEOL2	STA	(BASL),Y	STORE BLANKS FROM 'HERE'
	94	INY	,,,	TO END OF LINES (WNDWDTH)
	95	CPY	WNDWDTH	
	96	BCC RTS	CLEOL2	
	97 98 WAIT	SEC		
	99 WAIT2	PHA		
	00 WAIT3	SBC	#\$01	
	01	BNE	WAIT3	1.0204 USEC
	02	PLA	#¢01	(13+2712*A+512*A*A)
	03 04	SBC BNE	#\$01 WAIT2	
	ù5	RTS	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	
	06 NXTA4	INC	A4L	INCR 2-BYTE A4
	07	BNE	NXTAl	AND Al
	08 60 NYMX1	INC LDA	A4H A1L	INCR 2-BYTE Al.
	09 NXTAl 10	CMP	A2L	THER 2 BITS AT.
	11	LDA	AlH	AND COMPARE TO A2
	12	SBC	A2H	
FCC2: E6 3C 7	13	INC	AlL	(CARRY SET IF >=)
	14	BNE	RTS4B AlH	
	15 16 RTS 4B	INC RTS	4111	
	17 HEADR	LDY	#\$4B	WRITE A*256 'LONG 1'
FCCB: 20 DB FC 7	18	JSR	ZERDLY	HALF CYCLES
	19	BNE	HEADR	(650 USEC EACH )
	20	ADC BCS	#\$FE	THEN A 'SHORT 0'
	21 22	LDY	HEADR #\$21	(400 USEC)
FCD6: 20 DB FC		JSR	ZERDLY	WRITE TWO HALF CYCLES
FCD9: C8	24	INY		OF 250 USEC ('0')
	25	INY		OR 500 USEC ('0')
	26 ZERDLY 27	DE Y BNE	ZERDLY	
	28	BCC	WRTAPE	Y IS COUNT FOR
	29	LDY	#\$32	TIMING LOOP
	30 ONEDLY	DEY		
	31	BNE	ONEDLY	
FCE5: AC 20 C0 7	32 WRTAPE	LDY	TAPEOUT #\$2C	
	34	DE X	" 7	
FCEB: 60	35	RTS		
	36 RDBYTE	LDX	#\$08	8 BITS TO READ READ TWO TRANSITIONS
FCEE: 48 FCEF: 20 FA FC	'37 RDBYT2	PHA JSR	RD2BIT	(FIND EDGE)
	39	PLA		(
	40	ROL	A	NEXT BIT
	741	LDY	#\$3A	COUNT FOR SAMPLES
	742 743	DE X BNE	RDBYT2	
	144	RTS	RUUITZ	
FCFA: 20 FD FC		JSR	RDBIT	
FCFD: 88	746 RDBIT	DEY	m. n	DECR Y UNTIL
FCFE: AD 60 CO		LDA	TAPEIN	TAPE TRANSITION
	748 749	ECR BPL	LASTIN RDBIT	
	750	EOR	LASTIN	
FD07: 85 2F	751	STA	LASTIN	
	752	CPY	#\$8U	SET CARRY ON Y-REG.
	753 754 RDKEY	RTS LDY	СН	
	754 KDREI 755	LDA	(BASL),Y	SET SCREEN TO FLASH
	756	PHA		
FD11: 29 3F	757	AND	#\$3F	
	758	ORA	#\$40	
	759 760	STA PLA	(BASL),Y	
FD17: 66 FD18: 6C 38 00		JMP	(KSWL)	GO TO USER KEY-IN
	762 KEYIN	INC	RNDL	
FD1D: DU 02	763	BNE	KEYIN2	INCR RND NUMBER
FD1F: E6 4F	764 765 KEYIN2	INC BIT	RNDH KBD	KEY DOWN?
FD21: 2C 00 C0	, CO KEIINZ	011	1.00	

App	ole 2 Technic	cal Manual	•	Apple ][ Re	eference Manual • 1979
nn c i	10 mm =				_
FD24: FD26:	10 F5 766 91 28 767		B P L STA	KEYIN (BASL),Y	LOOP
FD28:	AD 00 CU 768		LDA		REPLACE FLASHING SCREEN GET KEYCODE
FD2B:	2C 10 C0 769		BIT	KBDSTRB	CLR KEY STROBE
FD2E:	60 770	naa	RTS		
FD2F: FD32:	20 UC FD 771 20 2C FC 772	ESC	JSR JSR		GET KEYCODE
FD35:	20 0C FD 773	RDCHAR	JSR		HANDLE ESC FUNC. READ KEY
FD38:	C9 9B 774		CMP		ESC?
FD3A:	F0 F3 775		BEQ	ESC	YES, DON'T RETURN
FD3C: FD3D:	60 776 A5 32 777	NOTER	RTS LDA	INVELC	
FD3F:	48 778	MOTER	PHA	INVFLG	
FD40:	A9 FF 779		LDA	#\$FF	
FD42: FD44:	85 32 780 BD 00 02 781		STA	INVFLG	ECHO USER LINE
FD47:	BD 00 02 781 20 ED FD 782		LDA JSR	IN,X COUT	NON INVERSE
FD4A:	68 783		PLA	C001	
FD4B:	85 32 784		STA	INVFLG	
FD4D: FD6):	BD 00 02 785 C9 38 786		LDA	IN,X	
FD52:	C9 38 786 F0 1D 787		CMP BEQ	#\$88 BCKSPC	CHECK FOR EDIT KEYS
FD54:	C9 98 788		CMP	#\$98	BS, CTRL-X.
FD56:	FO UA 789		BEQ	CANCEL	
FD53: FD5A:	E0 F8 790 90 03 791		CPX	#\$F8	MARGIN?
FD5C:	20 3A FF 792		BCC JSR	NOTCR1 BELL	YES, SOUND BELL
FD5F:	E8 793	NOTCR1	INX	5555	ADVANCE INPUT INDEX
	D0 13 794		BNE	NXTCHAR	
FD62: FD64:	A9 DC 795 20 ED FD 796	CANCEL		#\$DC	BACKSLASH AFTER CANCELLED LIN
	20 8E FD 797	GETLNZ	JSR JSR	COUT CROUT	OUTPUT CR
FD6A:	A5 33 798	GETLN	LDA	PROMPT	COTTOT CI
	20 ED FD 799			COUT	OUTPUT PROMPT CHAR
	A2 01 800 8A 801	BCKSPC	LDX TXA	#\$01	INIT INPUT INDEX
	F0 F3 802	SUMBLE	BEQ	GETLNZ	WILL BACKSPACE TO U
	CA 803		DEX		
FD75: FD78:	20 35 FD 804 C9 95 805	NXTCHAR	JSR	RDCHAR	
	D0 U2 8U6		CMP BNE	#PICK CAPTST	USE SCREEN CHAR FOR CTRL-U
FL7C:	B1 28 807		LDA	(BASL),Y	TOR CIRE 5
	C9 E0 808	CAPTST	CMP	#ŞEÜ	
	90 02 809 29 DF 810		BCC AND	ADDINP	CONVERT TO CAPS
	9D 00 02 811	ADDINP	STA	#SDF IN,X	ADD TO INPUT BUF
	C9 8D 812		CMP	#\$8D	
	D0 B2 813 20 9C FC 814		BNE	NOTCR	
	A9 8D 815	CROUT	JSR LDA	CLREOL #\$8D	CLR TO EOL IF CR
FD90:	DÚ 5B 816			COUT	
		PRA1	LDY		PRINT CR, Al IN HEX
	A6 3C 818 20 8E FD 819	PRYX2	LDX	AlL CROUT	
	20 40 F9 320	TRIAZ		PRNTYX	
FD9C:	A0 00 821			#\$UO	
	A9 AD 822			#\$AD	PRINT '-'
	4C ED FD 823 A5 3C 824	8MAX	JMP LDA	COUT	
	09 07 825	KAHO		#\$07	SET TO FINISH AT
	85 3E 826		STA		MOD 8=7
	A5 3D 827 85 3F 828		LDA		
	A5 3C 829	MOD8CHK	STA LDA	A2H AlL	
FDAF:	29 07 830		AND	#\$07	
	DJ 03 831	VRM	BNE	DATAOUT	
	20 92 FD 832 A9 A0 833	XAM DATACUT		PRA1 #\$A0	
FDB8:	20 ED FD 834	3		COUT	OUTPUT BLANK
	B1 3C 835		LDA	(AlL),Y	
	20 DA FD 836 20 BA FC 837			PRBYTE NXTA1	OUTPUT BYTE IN HEX
			JUK	MAINI	

Ap	ple 2 T	echni	cal Manua	al •	Apple ][	Reference Manual • 1979
FDC3:	90 E8	838		всс	MOD8CHK	CHECK IF TIME TO,
FDC5:	60	839	RTS 4C	RTS		PRINT ADDR
FDC6:	4A	840	XAMPM	LSR	A	DETERMINE IF MON
FDC7:	90 EA	841		BCC	XAM	MODE IS XAM
FDC9:	4A	842		LSR	A	ADD, OR SUB
FDCA:	4A	843		LSR	A	
FDCB:	A5 3E	844		LDA	A2L	
FDCD: FDCF:	90 02 49 FF	845		BCC	ADD	CUD. BODY 31C COMPLEMENT
FDD1:	65 3C	846 847	ADD	EOR ADC	#\$FF	SUB: FORM 2'S COMPLEMENT
FDD3:	48	848	אטט	PHA	AlL	
FDD4:	A9 BD	849		LDA	#\$BD	
FDD6:	20 ED 1			JSR		PRINT '=', THEN RESULT
FDD9:	68	851		PLA	0001	TRINI - / INDIVIDUODI
FDDA:	48	852	PRBYTE	PHA		PRINT BYTE AS 2 HEX
FDDB:	4A	853		LSR	A	DIGITS, DESTROYS A-REG
FDDC:	4A	854		LSR		,
FDDD:	4A	855		LSR	A	
FDDE:	4A	856		LSR	A	
FDDF:	20 E5 I	FD 857		JSR	PRHEXZ	
FDE 2:	68	858		PLA		
FDE3:	29 UF				#\$0F	PRINT HEX DIG IN A-REG
FDE5:	09 30	860	PRHEXZ		#\$B0	LSB'S
FDE7:	C9 BA	861			#\$BA	
FDE9:	90 02	862			COUT	
FDEB: FDED:	69 06 60 36 (	863	COUM	ADC	#\$06	VECTOR TO USER OUTRING ROUTING
FDF0:	C9 A0	865	COUT COUT1	JMP CMP	(CSWL) #\$A0	VECTOR TO USER OUTPUT ROUTINE
FDF2:	90 02	866	COULT	BCC	COUTZ	DON'T OUTPUT CTRL'S INVERSE
FDF4:	25 32	867		AND	INVFLG	MASK WITH INVERSE FLAG
FDF6:	84 35	868	COUTZ	STY		SAV Y-REG
FDF8:	48	869		PHA		SAV A-REG
FDF9:	20 FD I				VIDOUT	CUTPUT A-REG AS ASCII
FDFC:	68	871		PLA		RESTORE A-REG
FDFD:	A4 35	872		LDY	YSAVl	AND Y-REG
FDFF:	6Ú	873		RTS		THEN RETURN
FEUO:	C6 34	874	BLl	DEC	YSAV	
FEÜ2:	F0 9F	875		BEQ	XAM8	
FEU4:	CA DO 16	876	BLANK	DEX	0.00040.0	BLANK TO MON
FEU5: FEU7:	DU 16 C9 BA	877			SETMDZ	AFTER BLANK
FE07:	DO BB	878 879		CMP BNE		DATA STORE MODE?
FE 0B:	85 31	880	STOR	STA	XAMPM MODE	NO, XAM, ADD OR SUB KEEP IN STORE MODE
FEOD:	A5 3E	881	SIOR	LDA	A2L	REEF IN STORE MODE
FEOF:	91 40	882		STA	(A3L),Y	STORE AS LOW BYTE AS (A3)
FE11:	E6 40	883		INC	A3L	313112 113 (113)
FE13:	D0 02	884		BNE	RTS5	INCR A3, RETURN
FE15:	E6 41	885		INC	A3H	•
FE17:	60	886	RTS 5	RTS		
	A4 34	887	SETMODE	LDY	YSAV	SAVE CONVERTED ':', '+',
	B9 FF (				IN-1,Y	'-', '.' AS MODE.
	85 31	889	SETMDZ		MODE	
FElF:	60	89û	. m	RTS		
FE 20:	A2 01	891			#\$01	20 PV 10 (0 PVPPG) #0
FE 22: FE 24:	B5 3E 95 42	892 893	LT2		A2L,X	COPY A2 (2 BYTES) TO
	95 44	894			A4L,X	A4 AND A5
FE 28:	CA CA	895		DEX	A5L,X	
FE 29:	10 F7	896			LT2	
FE2B:	60	897		RTS	012	
FE 2C:	B1 3C	898	MOVE		(AlL).Y	MOVE (A1 TO A2) TO
FE 2E:	91 42	899			(A4L),Y	(A4)
FE 30:	20 B4 B	C 900			NXTA4	()
FE33:	90 F7	901		BCC	MOVE	
FE35:	60	902		RTS		
FE36:	B1 3C	903	VFY	LDA	(AlL),Y	VERIFY (Al TO A2) WITH
FE38:	D1 42	904		CMP	(A4L),Y	(A4)
FE3A:	F0 1C	905			VF YOK	
FE3C:	20 92 i				PRA1	
FE3F:	B1 3C	907			(AlL),Y	
FE41: FE44:	20 DA I A9 A0	909			PRBYTE	
	a = 40	303			#\$AU	
	20 ED 1	010		TSE	COUT	

Apı	ple 2 Tec	hnic	al Manual	•	Apple ][ F	Reference Manual • 1979
FE49:	A9 A8	911	· ·	LDA	#\$A8	
FE4B:	20 ED FD	912		JSR	COUT	
FE4E:	B1 42	913		LDA	(A4L),Y	
FE50:	20 DA FD			JSR	PRBYTE	
FE53:	A9 A9	915		LDA	#\$A9	
FE55: FE58:	20 ED FD 20 B4 FC		UD VOV	JSR		
FE5B:	90 D9	918	VFYOK	JSR BCC	NXTA4 VFY	
FE5D:	6ù	919		RTS	V1 1	
FE5E:	20 75 FE		LIST		AlPC	MOVE A1 (2 BYTES) TO
FE61:	A9 14	921		LDA	#\$14	PC IF SPEC'D AND
FE 63:	48	922	LIST2	PHA		DISSEMBLE 20 INSTRS
FE64: FE67:	20 D0 F8 20 53 F9	923		JSR JSR	INSTDSP PCADJ	ADJUGE DO ENGU INCEP
FE6A:	85 3A	925		STA	PCL	ADJUST PC EACH INSTR
FE6C:	84 3B	926		STY	PCH	
FE6E:	68	927		PLA		
FE6F:	38	928		SEC		
FE70: FE72:	E9 Ul Dú EF	929 930		SBC BNE	#\$01	NEXT OF 20 INSTRE
FE74:	60	931		RTS	LIST2	
FE75:	8A	932	AlPC	ΊXΑ		IF USER SPEC'D ADR
FE 76:	FO 07	933		BEQ	Alperts	COPY FROM A1 TO PC
FE78:	B 5 3C	934	AlPCLP	LDA	AlL,X	
FE7A: FE7C:	95 3A CA	935 936		STA DEX	PCL,X	
FE7D:		937		BPL	AlPCLP	
FE7F:	60	938	Alperts	RTS		
FE80:	A0 3F	939	SETINV	LDY	#\$3F	SET FOR INVERSE VID
FE82: FE84:	D0 02 A0 FF	940 941	SETNORM	BNE LDY	SETIFLG #\$FF	VIA COUT1
FE86:	84 32	942	SETIFLG	STY	INVFLG	SET FOR NORMAL VID
FE88:	60	943		RTS		
FE89:	A9 00	944	SETKBD		#\$00	SIMULATE PORT #0 INPUT
FE8B: FE8D:	85 3E A2 38	945	INPORT	STA	A2L	SPECIFIED (KEYIN ROUTINE)
FE8F:	A0 1B	946 947	INPRT	LDY	#KSWL #KEYIN	
FE91:		948		BNE	IOPRT	
FE93:		949	SETVID	LDA	#\$00	SIMULATE PORT #0 OUTPUT
FE95:		950	OUTPORT	STA	A2L	SPECIFIED (COUT1 ROUTINE)
FE97: FE99:	A2 36 AU FU	951 952	OUTPRT	LDX LDY	#CSWL #COUT1	
FE9B:		953	IOPRT	LDA	A2L	SET RAM IN/OUT VECTORS
FE9D:		954		AND	#\$0F	ODI MMI IN, OOI VEGICKE
FE9F:		955		BEQ	IOPRT1	
FEA1: FEA3:		956		ORA	#IOADR/256	
FEA5:		957 958		LDY BEQ	#\$00 IOPRT2	
FEA7:		959	IOPRT1	LDA	#COUT1/256	
FEA9:	94 00	960	IOPRT2	STY	LOCU, X	
FEAB:	95 01	961		STA	LOC1,X	
FEAD: FEAE:	60 EA	962. 963		RTS NOP		
FEAF:	EA	964		NOP		
FEB0:	4C 00 E0		XBASIC		BASIC	TO BASIC WITH SCRATCH
FEB3:	4C 03 E0		BASCONT	JMP	BASIC2	CONTINUE BASIC
FEB6:	20 75 FE		GO		Alpc	ADR TO PC IF SPEC'D
FEB9: FEBC:	20 3F FF 6C 3A 00			JSR JMP	RESTORE (PCL)	RESTORE META REGS GO TO USER SUBR
FEBF:	4C D7 FA		REGZ		REGDS P	TO REG DISPLAY
FEC2:	C6 34	971	TRACE		YSAV	
FEC4:	20 75 FE		STEPZ		AlPC	ADR TO PC IF SPEC'D
FEC 7: FECA:	4C 43 FA		HCD	JMP	STEP	TAKE ONE STEP
FECD:	4C F8 03 A9 40	974 975	USR WRITE	JMP LDA	USRADR #\$40	TO USR SUBR AT USRADR
FECF:	20 C9 FC			JSR	HEADR	WRITE 10-SEC HEADER
FED2:	A0 27	977		LDY	#\$27	
FED4:	A2 00	978	WRI	LDX	#\$00	
FED6: FED8:	41 3C 48	979 980		EOR	(AlL, X)	
FED9:		980 981		PHA LDA	(AlL,X)	
					, , ,	

Apple 2 Ted	chnical Manual	•	Apple ][ Re	ference Manual • 1979
FEDB: 20 ED FE	932	JSR	WRBYTE	
FEDE: 20 BA FC		JSR	NXTAl	
FEE1: AU 1D	984	LDY	#\$1D	
FEE3: 68	985	PLA		
FEE4: 90 EE	986	BCC	WR1	
FEE6: A0 22	987	LDY	#\$22	
FEE8: 20 ED FE		JSR	WRBYTE	
FEEB: FU 4D	989	BEQ	BELL	
FEED: A2 10 FEEF: UA	990 WRBYTE 991 WRBYT2	LDX	#\$10	
FEFO: 2U D6 FC		AS L JSR	A WRBIT	
FEF3: DO FA	993	BNE	WRBYT2	
FEF5: 60	994	RTS	WWDIIL	
FEF6: 20 JO FE	995 CRMON	JSR	BLl	HANDLE CR AS BLANK
FEF9: 68	996	PLA		THEN POP STACK
FEFA: 68	997	PLA		AND RTN TO MON
FEFB: DO 6C	998	BNE	MONZ	
FEFD: 20 FA FC		JSR	RD2BIT	FIND TAPEIN EDGE
FF00: A9 16 FF02: 20 C9 FC	1000	LDA	#\$16	DRIAN 3 5 ARROUNG
FF05: 85 2E	1001	JSR STA	HEADR CHKSUM	DELAY 3.5 SECONDS
FFU7: 20 FA FC		JSR	RD2BIT	INIT CHKSUM=\$FF FIND TAPEIN EDGE
FFUA: AU 24	1003 1004 RD2	LDY	#\$24	LOOK FOR SYNC BIT
FFUC: 20 FD FC		JSR	RDBIT	(SHORT 0)
FFOF: BO F9	1006	BCS	RD2	LOOP UNTIL FOUND
FF11: 20 FD FC	1007	JSR	RDBIT	SKIP SECOND SYNC H-CYCLE
FF14: AU 3B	1003	LDY	#\$3B	INDEX FOR 0/1 TEST
FF16: 20 EC FC		JSR	RDBYTE	READ A BYTE
FF19: 81 3C	1010	STA	(AlL,X)	STORE AT (Al)
FF1B: 45 2E FF1D: 85 2E	1011	EOR	CHKSUM	URBLAND SUMMERICA COMPANIA
FF1D: 85 2E FF1F: 20 BA FC	1012	STA	CHKSUM	UPDATE RUNNING CHKSUM
FF22: AU 35	1013	JSR LDY	NXTA1 #\$35	INCR A1, COMPARE TO A2 COMPENSATE G/1 INDEX
FF24: 90 F0	1015	BCC	RD3	LOGP UNTIL DONE
FF26: 20 EC FC	1016		RDBYTE	READ CHKSUM BYTE
FF29: C5 2E	1017	CMP	CHKSUM	
FF2B: F0 OD	1018	BEQ	BELL	GOOD, SOUND BELL AND RETURN
FF2D: A9 C5	1019 PRERR	LDA	#\$C5	
FF2F: 20 ED FD		JSR	COUT	PRINT "ERR", THEN BELL
FF32: A9 D2 FF34: 20 ED FD	1021	LDA	#\$D2	
FF37: 20 ED FD		JSR JSR	COUT	
FF3A: A9 87	1024 BELL	LDA	#\$87	OUTPUT BELL AND RETURN
FF3C: 4C ED FD		JMP	COUT	OUT OF BEEF MAD ABTORN
FF3F: A5 48	1026 RESTORE	LDA	STATUS	RESTORE 6502 REG CONTENTS
FF41: 48	1027	PHA		USED BY DEBUG SOFTWARE
FF42: A5 45	1028	LDA	ACC	
FF44: A6 46	1029 RESTRI	LDX	XREG	
FF46: A4 47	1030	LDY	YREG	
FF48: 28 FF49: 60	1031	PLP		
FF4A: 85 45	1032 1033 SAVE	RTS STA	ACC	SAVE 6502 REG CONTENTS
	1034 SAVI		XREG	S 2 JUL REG CONTENTS
FF4E: 84 47	1035	STY	YREG	
FF50: 08	1036	PHP		
FF51: 68	1037	PLA		
FF52: 85 48	1038	STA	STATUS	
FF54: BA	1039	TSX		
FF55: 86 49	1040	STX	SPNT	
FF57: D8 FF58: 60	1041 1042	CLD		
	1042 1043 RESET	RTS JSR	SETNORM	SET SCREEN MODE
FF5C: 20 2F FB			INIT	AND INIT KBD/SCREEN
FF5F: 20 93 FE			SETVID	AS I/O DEV'S
FF62: 20 89 FE			SETKBD	, <del>-</del> -
FF65: D8	1047 MON	CLD		MUST SET HEX MODE!
FF66: 20 3A FF			BELL	•
FF69: A9 AA	1049 MONZ		#\$AA	'*' PROMPT FOR MCN
FF6B: 85 33 FF6D: 20 67 FD	1050		PROMPT	DEAD A CINE
FF70: 20 C7 FF			GETLNZ ZMODE	READ A LINE
	1053 NXTITM		GETNUM	CLEAR MON MODE, SCAN IDX GET ITEM, NON-HEX
FF76: 84 34	1054		YSAV	CHAR IN A-REG
		_		

Ap	ple 2 Te	chnical Manual	•	Apple ][ Re	eference Manual • 1979
FF78:	AU 17	1055	LDY	#\$17	X-REG=0 IF NO HEX INPUT
FF7A:	88	1056 CHRSRCH	DEY	π γ 1 /	X-REG-U IF NO HEX IMPUT
FF7B: FF7D:	30 E8 D9 CC FI	1057 7 1058	BMI CMP	MON CHRTBL, Y	NOT FOUND, GO TO MON FIND CMND CHAR IN TEL
FF80:	D0 F8	1059	BNE	CHRIBE, 1	FIND CHAR IN TEL
FF82:	20 BE FE		JSR	TCSUB	FOUND, CALL CORRESPONDING
FF85: FF87:	A4 34 4C 73 FE	1061 F 1062	LDY JMP	YSAV NXTITM	SUBROUTINE
FF &A:	A2 03	1063 DIG	LDX	#\$03	
FF8C: FF6D:	ua ua	1064 1065	ASL	A	COM UNIX DIG
FF8E:	UA UA	1066	ASL ASL	A A	GOT HEX DIG, SHIFT INTO A2
FF8F:	OΑ	1067	ASL	A	
FF90: FF91:	υΑ 26 3E	1068 NXTBIT 1069	ASL ROL	A A2L	
FF93:	26 3F	1070	ROL	A 2H	
FF95:	CA	1071	DEX		LEAVE X=\$FF IF DIG
FF96: FF98:	10 Fö A5 31	1072 1073 NXTBAS	3 PL LDA	NXTBIT MODE	
FF9A:	D0 06	1074	BNE	NXTBS2	IF MODE IS ZERO
FF9C: FF9E:	B5 3F	1075	LDA	A2H,X	THEN COPY A2 TO
FFAO:	95 3D 95 41	1076 1077	STA STA	A1H,X A3H,X	Al AND A3
FFA2:	E8	1078 NXTBS2	INX		
FFA3: FFA5:	F0 F3 D0 06	1079 1080	BEQ	NXTBAS NXTCHR	
FFA7:	A2 00	1081 GETNUM	BNE LDX	#\$UO	CLEAR A2
FFA9:	86 3E	1082	STX	A2L	
FFAB: FFAD:	86 3F 89 00 02	1083 2 1084 NXTCHR	STX LDA	A2H IN,Y	GET CHAR
FFB0:	C8	1085	INY	111,1	GET CHAR
FFB1: FFB3:	49 BÚ C9 ÚA	1086	EOR	#\$B0	
FFB5:	90 D3	1087 1088	CMP BCC	#\$0A DIG	IF HEX DIG, THEN
FFB7:	69 88	1089	ADC	#\$88	11 11211 220, 111211
FFB9: FFBB:	C9 FA B0 CD	1090 1091	CMP BCS	#\$FA DIG	
FFBD:	60	1092	RTS	DIG	
FFBE:	A9 FE	1093 TOSUE	LDA	#GO/256	PUSH HIGH-ORDER
FFCU: FFC1:	46 B9 E3 FF	1094 1095	PHA LDA	SUBTBL, Y	SUBR ADR ON STK PUSH LOW ORDER
FFC4:	48	1096	PHA	505155,1	SUBR ADR ON STK
FFC5: FFC7:	A5 31 Aù uù	1097 1098 ZMODE	LDA	MODE	CLD HODE OLD HODE
FFC9:	84 31	1098 20002	LDY STY	#\$00 MODE	CLR MODE, OLD MODE TO A-REG
FFCB:	60	1100	RTS		GO TO SUBR VIA RTS
FFCC: FFCD:	BC B2	1101 CHRTBL 1102	DFB DFB	\$BC \$B2	F("CTRL-C") F("CTRL-Y")
FFCE:	BE	1103	DFB	ŞBE	F("CTRL-E")
FFCF:	ED	1104	DFB	SED	F("T")
FFDU: FFD1:	EF C4	1105 1106	DFB	\$E F \$C 4	F("V") F("C'IRL-K")
FFD2:	EC	1107	DFB	ŞEC	F("S")
FFD3: FFD4:	A9 BB	1108 1109	DFB DFB	\$A9 \$BB	F("CTRL-P") F("CTRL-B")
FFD5:	A6	1110	DFB	\$A6	F("-")
FFD6:	A 4	1111	DFB	\$A4	F ("+")
FFD7: FFD8:	06 95	1112 1113	DFB DFB	\$06 \$95	F("M") (F=EX-OR \$B0+\$89) F("<")
FFD9:	07	1114	DFB	\$07	F("N")
FFDA: FFDB:	02	1115 1116	DFB	\$02	F("I")
FFDC:	05 F0	1116	DFB DFB	\$05 \$ <b>F</b> 0	F("L") F("W")
FFDD:	0.0	1118	DFB	\$00	F("G")
FFDE: FFDF:	EB 93	1119 1120	DFB DFB	\$EB \$93	F("R") F(":")
FFE0:	A 7	1121	DFB	\$47	F(".")
FFE1:	C6	1122	DFB	\$C 6	F("CR")
FFE2: FFE3:	99 B2	1123 1124 SUBTBL	DFB DFB	\$99 #BASCONT-1	F(BLANK)
FFE4:	C9	1125	DFB	#USR-1	
FFE5:	BE	1126	DFB	#REGZ-1	

#### Apple 2 Technical Manual Apple ][ Reference Manual 1979 FFE6: 1127 DFB #TRACE-1 FFE7: 35 1128 DFB #VFY-1 FFE8: 1129 DFB #INPRT-1 8C #STEPZ-1 FFE9: C 3 DFB 1130 #OUTPRT-1 FFEA: 96 1131 DFB DFB #XBASIC-1 FFEB: ΑF 1132 FFEC: 17 1133 DFB #SETMODE-1 FFED: 17 1134 DFB #SETMODE-1 FFEE: 1135 DFB #MOVE-1 2B FFEF: 1F 1136 DFB #LT-1 #SETNORM-1 DFB FFF0: 1137 83 #SETINV-1 FFF1: 7F 1138 DFB FFF2: 1139 DFB #LIST-1 FFF3: CC 1140 DFB #WRITE-1 FFF4: В5 1141 DFB #GO-1 #READ-1 FFF5: FC 1142 DFB #SETMODE-1 FFF6: 17 1143 DFB #SETMODE-1 FFF7: 17 1144 DFB FFF8: F 5 1145 DFB #CRMON-1 FFF9: 03 1146 DFB #BLANK-1 FFFA: 1147 DFB #NMI NMI VECTOR FΒ FFFB: 1148 DFB #NMI/256 0.3 RESET VECTOR FFFC: DFB #RESET 1149 59 #RESET/256 FFFD: FF 1150 DFB IRQ VECTOR FFFE: 86 1151 DFB #IRQ #IRQ/256 \$3C FFFF: FA 1153 XQTNZ

# SYMBOL TABLE (NUMERICAL ORDER)

00000000000000000000000000000000000000	FORMAT COLOR YSAV KSWL A1L A3L A5L YREG RNDH SOFTEV NMI IOADR SPKR MIXSET HIRES CLRAN3 CLRAN3 CLRANOM RTMASK VLINEZ CLRTOP GBCALC RTMS VLINEZ CLRTOP GBCALC RTMS NXTCOL PRADR3 PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL2 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRBL3 PCADB PRNTYX PRNTYX PRBL3 PCADB PRNTYX PRNTYX PRNTYX PRBL3 PCADB PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PRNTYX PR	FC9A FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED FCCED F	CANCEL LOC1 WNDBTM GBASH BAS2H RMNEM LASTIN MODE YSAV1 KSWH A1H A3H A5H STATUS PICK PWRELOC KBD TXTCLR LOWSCR SETANO SETAND TAPEIN BASIC PLOT1 VLINE CLRSC2 SETCOL INSDS1 GETFMT INSTDSP PRMN2 PRADR4 PRNTAX PRBL3	FB78 FB98 FB07 FC122 FC26 FC35 FC26 FC35 FC26 FC35 FC36 FC36 FC36 FC36 FC36 FC36 FC36 FC36	ESC1 CR SCRL2 CLEOL2 NXTA4 WRBIT RDKEY RDKHAR GETLNZ WNDLFT CH BASL HASNOTL HASNOTL PCL A4L CN LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR LINUSTR
<b>EAA3</b>	NOFIX	F8F9	PRMN2	E003	BASIC2
FB2E	RTS2D		PCADJ4	F871	
	SETWND	F9B4	CHAR1	F88C	INSDS2
	SETPWRC	FA40	IRQ	F8BE	MNNDX1
	ESCOLD		INITAN	F8D4	PRNTOP
	BASCLC2		PWRUP		PRADR1
	STORADV		NXTBYT		PRADR5
FC10			PWRCON		PRNTX
FC2B		FB19			PCADJ
FC58	HUME	FB2F	INIT	F961	RTS2

Apple 2 Technical Manual	Apple ][ Reference Manual
	PRADR2 FDFO COUT1
	RELADR FEOB STOR
	B PRBLNK FE20 LT
	FE58 VFYOK FE78 A1PCLP
	MNEML FE86 SETIFLG
	OLDBRK FE93 SETVID
	B FIXSEV FEAT IOPRT1
	S SETPLP FEB6 GO
	A RGDSP1 FECA USR
	FEEF WRBYT2
	5 PREAD2 FF16 RD3
FBFC RTS3 FB46	SETGR FF44 RESTR1
FC22 VTAB FB6	5 STITLE FF65 MON
	NOWAIT FF8A DIG
	L BASCALC FFA7 GETNUM
	F RTS2B FFCC CHRTBL
	O VIDOUT FD84 ADDINP
	TOTABZ FDA3 XAM8
	CLEOP1 FDC5 RTS4C
	SCROLL FDE3 PRHEX
	CCLREOL FDF6 COUTZ PWAIT2 FE17 RTS5
	7 WAIT2 FE17 RTS5 3 RTS4B FE22 LT2
	ONEDLY FESE LIST
	A RD2BIT FE7F A1PCRTS
	L KEYIN2 FE89 SETKBD
	NOTCR1 FE95 OUTPORT
	L BCKSPC FEA9 IOPRT2
	NXTCHAR FEBF REGZ
0033 PROMPT FD98	PRA1 FECD WRITE
	3 XAM FEF6 CRMON
	L ADD FF2D PRERR
	COUT FF4A SAVE
	BLANK FF69 MONZ
	SETMDZ FF90 NXTBIT
	VFY FFAD NXTCHR
	5 A1PC FFE3 SUBTBL 4 SETNORM FD8E CROUT
	) INPRT FDAD MODBCHK
	B IOPRT FDC6 XAMPM
	B BASCONT FDE5 PRHEXZ
	STEPZ FEOO BL1
	WRBYTE FE18 SETMODE
	A RD2 FE2C MOVE
CO70 PTRIG FF3	RESTORE FE63 LIST2
F800 PLOT FF5	OLDRST FE80 SETINV
F81C HLINE1 FF7	A CHRSRCH FEBB INPORT
	NXTBS2 FE97 OUTPRT
	7 ZMODE FEBO XBASIC
	CAPTST FEC2 TRACE
	5 PRYX2 FED4 WR1
	5 DATAOUT FEFD READ
FBDB PRNTBL FDD	A PRBYTE FF3A BELL

FF4C SAV1 FF73 NXTITM FF98 NXTBAS FFBE TOSUB

# SYMBOL TABLE (ALPHABETICAL ORDER)

003D		F956	PCADJ3	FEA7	IOPRT1
FE7F	A1PCRTS	0095	PICK	FA40	
0040		F910	PRADR1		KEYIN
0044			PRADR5	002F	LASTIN
	ADVANCE	FDDA	PRBYTE		LIST
002A	BAS2L		PRHEX	0001	LOC1
	BASH	F8DB	PRNTBL	FE20	
	BCKSPC		PROMPT		MNEML
FE00	BL1		PWREDUP		EXQUIM
FC10		FF16		FF65	
	CHAR2	FD35	RDCHAR	03FB	NMI
0024			REGDSP	FB94	NOWAIT
	CLRANO				NXTBIT
	CLREOL	004F	RNDH		NXTCHR
F83C	CLRSC3	F87F	RTMSKZ		OLDRST
FDED	COUT		RTS2		PADDLO
FC62	CR	0030	A1L		PCADJ4
0025	CV	003F	A2H		PLOT1
FBA5		0043	A2H A4H		PRADR2
	ESCOLD		ACC		PRBL2
F9A6	FMT2		AMPERV		PREAD
0026	GBASL		BASCALC		PRHEXZ
FD6A	GETLN	F000	BASIC		PRNTOP
FCC9	HEADR		BELL1		PRYX2
F819	HLINE		BLANK		PWRUP
0200	IN		CANCEL		RDBIT
F882	INSDS1		CHKSUM		RDKEY
C000	IOADR		CLEOL2		REGZ
03FE	IRQLOC		CLRAN1		RESTR1
C000	KBD		CLREOP		RNDL
0038	KSWL		CLRSCR		RTS1
0400	LINE1		COUT 1		RTS3
0000	LOCO		CRMON		AIPCLP
FE22	LT2		DATAOUT	003E	
CO53	MIXSET		ESC1	0042	
F8C2	MNNDX2		ESC		ADDINP
FF69			FORMAT		APPLEII
FA81	NEWMON		GBCALC		BASCLC2
FD5F	NOTCR1		GETNUM		BASIC2
	NXTBAS		HIRES		BELL2
	NXTCHAR		HOME		BREAK
	OLDBRK				CAPTST
	OUTPRT		INIT		CHRSRCH
		L RRC	INSDS2	rr/A	CHKSKCH

Apple 2 Technical	Manual	Apple ]	Reference Manual	1979
FC9E CLEOLZ	FF3A	DE: I	COSC SETAN2	
COSD CLRAN2		BRKV	FE86 SETIFL	^
CFFF CLRROM		CHAR1	FE18 SETMOD	
F836 CLRTOP		CHRTBL	FB6F SETPWR	
FDF6 COUTZ		CLEOP1	002F SIGN	G
0037 CSWH		CLRANS	0049 SPNT	
FFBA DIG		CLRSC2	FEOB STOR	
FBA5 ESCNEW		COLOR	CO60 TAPEIN	
FA9B FIXSEV		CROUT	FEC2 TRACE	
F847 GBASCALC	0036		FECA USR	
F8A9 GETFMT		DISKID	FE58 VFYOK	
FEB6 GO		ESCNOW	F828 VLINE	
CO55 HISCR	F962		FCAB WAIT	
F89B IEVEN	0027	GBASH	0022 WNDTOP	
FE8B INPORT	FD67	GETLNZ	FEEF WRBYT2	
F8DO INSTDSP	0050	H2	FDA3 XAM8	
FEA9 IOPRT2	F81C	HLINE1	FB11 XLTBL	
CO10 KBDSTRB	FA6F	INITAN	0034 YSAV	
FD21 KEYIN2		INPRT	FCBC SCRL2	
002F LENGTH	0032	INVFLG	FC70 SCROLL	
FE63 LIST2	FE9B	IOPRT	COSE SETAN3	
CO56 LORES	FB88	KBDWAIT	FE80 SETINV	
002E MASK	0039	KSWH	FE84 SETNOR	M
FAOO MNEMR	FC66	LF	FB39 SETTXT	
FDAD MODECHK		LMNEM	FABA SLOOP	
FE2C MOVE		LOWSCR	0048 STATUS	
FAA3 NOFIX		MIXCLR	FBFO STORAD	
FCBA NXTA1		MNNDX1	CO20 TAPEOU	Т
FFA2 NXTBS2		MODE	CO50 TXTCLR	
F8F5 NXTCOL		MSLOT	O3F8 USRADR	
FCE2 ONEDLY F954 PCADJ2		NOTCR NXTA4	FBFD VIDOUT	
003B PCH		NXTBYT	FC24 VTABZ	
F800 PLOT		NXTITM	FCAA WAIT3 0021 WNDWDT	Ц
F926 PRADR3		OUTPORT	FEED WRBYTE	П
F94C PRBL3		PCADJ	FDC6 XAMPM	
FB25 PREAD2	003A		0046 XREG	
F8F9 PRMN2		PRA1	FCDB ZERDLY	
F944 PRNTX		PRADR4	FF4C SAV1	
CO70 PTRIG		PRBLNK	FC95 SCRL3	
FCFA RD2BIT		PRERR	CO58 SETANO	
FCEE RDBYT2		PRNTAX	F864 SETCOL	
FAE4 RDSP1	F940	PRNTYX	FE89 SETKBD	
F938 RELADR	FAFD	PWRCON	FAA9 SETPG3	
FADA RGDSP1	FFOA	RD2	FE93 SETVID	
FB19 RTBL	FCEC	RDBYTE	03F2 SOFTEV	
FBEF RTS2B	FEFD	READ	FEC4 STEPZ	
FCC8 RTS4B	FA62	RESET	FFE3 SUBTBL	
FE75 A1PC	0020	RMNEM	FBO9 TITLE	
0041 A3H		RTMASK	CO51 TXTSET	
0045 A5H		RTS2D	002D V2	
FDD1 ADD		RTS4C	FB78 VIDWAI	Τ
002B BAS2H		RTS5	FC22 VTAB	
FEB3 BASCONT	FC2E	RTS4	0023 WNDBTM	

F879 SCRN2

FC76 SCRL1

FED4 WR1

0028 BASL

FECD WRITE FDB3 XAM 0047 YREG FFC7 ZMODE FF4A SAVE F871 SCRN CO5A SETAN1 FB40 SETGR FEID SETMDZ FAAB SETPLP FB4B SETWND CO30 SPKR FB65 STITLE FB5B TABV FFBE TOSUB FC1A UP FE36 VFY F826 VLINEZ FCA9 WAIT2

0020 WNDLFT FCD6 WRBIT

FCE5 WRTAPE

FEBO XBASIC

0035 YSAV1

SYMBOL TABLE SIZE 2589 BYTES USED

2531 BYTES REMAINING

SLIST 4A

"A2M 030-0004-01 5-177.PICT" 13296 KB 2001-07-23 dpi: 600h x 600v pix: 2945h x 4704v

Author: Apple Computer, Inc.

• Document # 030-0004-01

Page 0195 of 0275

65\$2: The manufacturer's name for the microprocessor at the heart of your Apple.

**Address:** As a noun: the particular number associated with each memory location. On the Apple, an address is a number between  $\emptyset$  and 65535 (or \$0000 and \$FFFF hexadecimal). As a verb: to refer to a particular memory location.

**Address Bus:** The set of wires, or the signal on those wires, which carry the binary-encoded address from the microprocessor to the rest of the computer.

Addressing mode: The Apple's 6502 microprocessor has thirteen distinct ways of referring to most locations in memory. These thirteen methods of forming addresses are called addressing modes.

**Analog:** Analog measurements, as opposed to digital measurements, use an continuously variable physical quantity (such as length, voltage, or resistance) to represent values. Digital measurements use precise, limited quantities (such as presence or absence of voltages or magnetic fields) to represent values.

AND: A binary function which is "on" if and only if all of its inputs are "on".

**Apple:** 1. The round fleshy fruit of a Rosaceous tree (Pyrus Malus). 2. A brand of personal computer. 3) Apple Computer, Inc., manufacturer of home and personal computers.

**ASCII:** An acronym for the American Standard Code for Information Interchange (often called "USASCII" or misinterpreted as "ASC-II"). This standard *code* assigns a unique value from  $\emptyset$  to 127 to each of 128 numbers, letters, special characters, and control characters.

Assembler: 1) One who assembes electronic or mechanical equipment. 2) A program which converts the *mnemonics* and *symbols* of assembly language into the *opcodes* and *operands* of machine language.

Assembly language: A language similar in structure to machine language, but made up of *mnemonics* and *symbols*. Programs written in assembly language are slightly less difficult to write and understand than programs in machine language.

**BASIC:** Acronym for "Beginner's All-Purpose Symbolic Instruction Code". BASIC is a *higher-level language*, similar in structure to FORTRAN but somewhat easier to learn. It was invented by Kemney and Kurtz at Dartmouth College in 1963 and has proved to be the most popular language for personal computers.

**Binary:** A number system with two digits, "0" and "1", with each digit in a binary number representing a power of two. Most digital computers are binary, deep down inside. A binary signal is easily expressed by the presence or absence of something, such as an electrical potential or a magnetic field.

**Binary Function:** An operation performed by an electronic circuit which has one or more inputs and only one output. All inputs and outputs are binary signals. See *AND OR*, and *Exclusive-OR*.

**Bit:** A *Binary digIT*. The smallest amount of information which a computer can hold. A single bit specifies a single value: "Ø" or "1". Bits can be grouped to form larger values (see *Byte* and *Nybble*).

Board: See Printed Circuit Board.

**Bootstrap** ("boot"): To get a system running from a *cold-start*. The name comes from the machine's attempts to "pull itsef off the ground by tugging on its own bootstraps."

**Buffer:** A device or area of memory which is used to hold something temporarily. The "picture buffer" contains graphic information to be displayed on the video screen; the "input buffer" holds a partially formed input line.

**Bug:** An error. A *hardware bug* is a physical or electrical malfunction or design error. A *software* bug is an error in programming, either in the logic of the program or typographical in nature. See "feature".

**Bus:** A set of wires or *traces* in a computer which carry a related set of data from one place to another, or the data which is on such a bus.

**Byte:** A basic unit of measure of a computer's memory. A byte usualy comprises eight *bits*. Thus, it can have a value from  $\emptyset$  to 255. Each character in the *ASCII* can be represented in one byte. The Apple's memory locations are all one byte, and the Apple's addresses of these locations consist of two bytes.

Call: As a verb: to leave the program or subroutine which is currently executing and to begin another, usually with the intent to return to the original program or subroutine. As a noun: an instruction which calls a subroutine.

**Character:** Any *graphic* symbol which has a specific meaning to people. Letters (both upper- and lower-case), numbers, and various symbols (such as punctuation marks) are all characters.

Chip: See Integrated Circuit.

Code: A method of representing something in terms of something else. The ASCII code represents characters as binary numbers, the BASIC language represents algorithms in terms of program statements. Code is also used to refer to programs, usually in *low-level languages*.

Cold-start: To begin to operate a computer which has just been turned on.

**Color burst:** A signal which color television sets recognize and convert to the colored dots you see on a color TV screen. Without the color burst signal, all pictures would be black-and-white.

**Computer:** Any device which can receive and store a set of *instructions*, and then act upon those instructions in a predetermined and predictable fashion. The definition implies that both the instruction and the *data* upon which the instructions act can be changed. A device whose instructions cannot be changed is not a computer.

Control (CTRL) character: Characters in the ASCII character set which usually have no graphic representation, but are used to control various functions. For example, the RETURN control character is a signal to the Apple that you have finished typing an *input line* and you wish the computer to act upon it.

CRT: Acronym for "Cathode-Ray Tube", meaning any television screen, or a device containing such a screen.

Cursor: A special symbol which reminds you of a certain position on something. The cursor on a slide rule lets you line up numbers; the cursor on the Apple's screen reminds you of where you are when you are typing.

Data (datum): Information of any type.

**Debug:** To find *bugs* and eliminate them.

**DIP:** Acronym for "Dual In-line Package", the most common container for an Integrated Circuit. DIPs have two parallel rows of *pins*, spaced on one-tenth of an inch centers. DIPs usually come in 14-, 16-, 18-, 20-, 24-, and 40-pin configurations.

**Disassembler:** A program which converts the *opcodes* of *machine language* to the *mnemonics* of assembly language. The opposite of an assembler.

**Display:** As a noun: any sort of output device for a computer, usually a *video* screen. As a noun: to place information on such a screen.

Edge connector: A socket which mates with the edge of a printed circuit board in order to exchange electrical signals.

Entry point: The location used by a machine-language subroutine which contains the first executable instruction in that subroutine; consequently, often the beginning of the subroutine.

**Excusive-OR:** A binary function whose value is "off" only if all of its inputs are "off", or all of its inputs are "on".

**Execute:** To perform the intention of a command or instruction. Also, to run a program or a portion of a program.

**Feature:** A bug as described by the marketing department.

Format: As a noun: the physical form in which something appears. As a verb: to specify such a form.

**Graphic:** Visible as a distinct, recognizable shape or color.

**Graphics:** A system to display graphic items or a collection of such items.

**Hardware:** The physical parts of a computer.

**Hexadecimal:** A number system which uses the ten digits 0 through 9 and the six letters A through F to represent values in base 16. Each hexadecimal digit in a hexadecimal number represents a power of 16. In this manual, all hexadecimal numbers are preceded by a dollar sign (\$).

High-level Language: A language which is more intelligible to humans than it is to machines.

**High-order:** The most important, or item with the highest vaue, of a set of similar items. The high-order bit of a byte is that which has the highest place value.

**High part:** The *high-order* byte of a two-byte address. In decimal, the high part of an address is the quotient of the address divided by 256. In the 6502, as in many other microprocessors, the high part of an address comes last when that address is stored in memory.

**Hz** (Hertz): Cycles per second. A bicycle wheel which makes two revolutions in one second is running at 2Hz. The Apple's microprocessor runs at 1,023,000Hz.

I/O: See Input/Output.

IC: See Integrated Circuit.

**Input:** As a noun: data which flows from the outside world into the computer. As a verb: to obtain data from the outside world.

**Input/Output (I/O):** The software or hardware which exchanges data with the outside word.

**Instruction:** The smallest portion of a program that a computer can execute. In 6502 machine language, an instruction comprises one, two, or three bytes; in a higher-level language, instructions may be many characters long.

**Integrated circuit:** A small (less than the size of a fingernail and about as thin) wafer of a glassy material (usually silicon) into which has been etched an electronic circuit. A single IC can contain from ten to ten thousand discrete electronic components. ICs are usually housed in *DIPs* (see above), and the term IC is sometimes used to refer to both the circuit and its package.

**Interface:** An exchange of information between one thing and another, or the mechanisms which make such an exchange possible.

**Interpreter:** A program, usualy written in machine language, which understands and executes a higher-level language.

**Interrupt:** A physical effect which causes the computer to jump to a special interrupt-handling subroutine. When the interrupt has been taken care of, the computer resumes execution of the interrupted program with no noticeable change. Interrupts are used to signal the computer that a particular device wants attention.

**K:** Stands for the greek prefix "Kilo", meaning one thousand. In common computer-reated usage, "K" usually represents the quantity  $2^{10}$ , or 1024 (hexadecimal \$400).

Kilobyte: 1,024 bytes.

Language: A computer language is a code which (hopefully!) both a programmer and his computer understand. The programmer expresses what he wants to do in this code, and the computer understands the code and performs the desired actions.

Line: On a video screen, a "line" is a horizontal sequence of graphic symbols extending from one edge of the screen to the other. To the Apple, an *input line* is a sequence of up to 254 characters, terminated by the control character RETURN. In most places which do not have personal computers, a line is something you wait in to use the computer.

Low-level Language: A language which is more intelligible to machines than it is to humans.

**Low-order:** The least important, or item with the least vaue, of a set of items. The low-order bit in a byte is the bit with the least place vaue.

Low part: The *low-order* byte of a two-byte address. In decimal, the low part of an address is the remainder of the address divided by 256, also called the "address *modulo* 256." In the 6502, as in many other microprocessors, the low part of an address comes first when that address is stored in memory.

Machine language: The lowest level language which a computer understands. Machine

languages are usually binary in nature. Instructions in machine language are single-byte opcodes sometimes followed by various operands.

**Memory address:** A memory address is a two-byte value which selects a single memory location out of the *memory map*. Memory addresses in the Apple are stored with their low-order bytes first, followed by their high-order bytes.

**Memory location:** The smallest subdivision of the memory map to which the computer can refer. Each memory location has associated with it a unique *address* and a certain *value*. Memory locations on the Apple comprise one byte each.

**Memory Map:** This term is used to refer to the set of all memory locations which the microprocesor can address directly. It is also used to describe a graphic representation of a system's memory.

**Microcomputer:** A term used to described a computer which is based upon a microprocessor.

**Microprocessor:** An integrated circuit which understands and executes machine language programs.

**Mnemonic:** An acronym (or any other symbol) used in the place of something more difficut to remember. In *Assembly Language*, each machine language opcode is given a three letter mnemonic (for example, the opcode \$60 is given the mnemonic RTS, meaning "ReTurn from Subroutine").

Mode: A condition or set of conditions under which a certain set of rules apply.

**Modulo:** An arithmetic function with two operands. *Modulo* takes the first operand, divides it by the second, and returns the remainder of the division.

**Monitor:** 1) A closed-circuit television receiver. 2) A program which allows you to use your computer at a very low level, often with the values and addresses of individual memory locations.

**Multiplexer:** An electronic circuit which has many data inputs, a few selector inputs, and one output. A multiplexer connects one of its many data inputs to its output. The data input it chooses to connect to the output is determined by the selector inputs.

Mux: See Multiplexer.

**Nybble:** Colloquial term for half of a byte, or four bits.

**Opcode:** A machine language instruction, numerical (often binary) in nature.

OR: A binary function whose value is "on" if at least one of its inputs are "on".

**Output:** As a noun, data generated by the computer whose destination is the real world. As a verb, the process of generating or transmitting such data.

**Page:** 1) A screenfull of information on a video display. 2) A quantity of memory locations, addressible with one byte. On the Apple, a "page" of memory contains 256 locations.

Pascal: A noted French scientist.

PC board: See Printed Circuit Board.

**Peripheral:** Something attached to the computer which is not part of the computer itself. Most peripherals are input and/or output devices.

**Personal Computer:** A computer with *memory*, *languages*, and *peripherals* which are well-suited for use in a home, office, or school.

**Pinout:** A description of the function of each pin on an IC, often presented in the form of a diagram.

**Potentiometer:** An electronic component whose resistance to the flow of electrons is proportional to the setting of a dial or knob. Also known as a "pot" or "variable resistor".

**Printed Circuit Board:** A sheet of fiberglass or epoxy onto which a thin layer of metal has been applied, then etched away to form *traces*. Electronic components can then be attatched to the board with molten solder, and they can exchange electronic signals via the etched traces on the board. Small printed circuit boards are often called "cards", especially if they are meant to connect with *edge connectors*.

**Program:** A sequence of instructions which describes a process.

**PROM:** Acronym for "Programmable Read-Only Memory". A PROM is a ROM whose contents can be altered by electrical means. Information in PROMs does not disappear when the power is turned off. Some PROMs can be erased by ultraviolet light and be reprogrammed.

RAM: See Random-Access Memory.

Random-Access Memory (RAM): This is the main memory of a computer. The acronym RAM can be used to refer either to the integrated circuits which make up this type of memory or the memory itself. The computer can store values in distinct locations in RAM and recall them again, or alter and re-store them if it wishes. On the Apple, as with most small computers, the values which are in RAM memory are lost when the power to the computer is turned off.

**Read-Only Memory (ROM):** This type of memory is usually used to hold important programs or data which must be available to the computer when the power is first turned on. Information in ROMs is placed there in the process of manufacturing the ROMs and is unalterable. Information stored in ROMs does not disappear when the power is turned off.

**Reference:** 1) A source of information, such as this manual. 2) As a verb, the action of examining or altering the contents of a memory location. As a noun, such an action.

**Return:** To exit a subroutine and go back to the program which called it.

ROM: See Read-Only Memory.

Run: To follow the sequence of instructions which comprise a program, and to complete the process outlined by the instructions.

Scan line: A single sweep of a cathode beam across the face of a cathode-ray tube.

Schematic: A diagram which represents the electrical interconnections and circuitry of an electronic device.

**Scroll:** To move all the text on a display (usually upwards) to make room for more (usually at the bottom).

**Soft switch:** A two-position switch which can be "thrown" either way by the software of a computer.

**Software:** The *programs* which give the hardware something to do.

**Stack:** A reserved area in memory which can be used to store information temporarily. The information in a stack is referenced not by address, but in the order in which it was placed on the stack. The last datum which was "pushed" onto the stack will be the first one to be "popped" off it.

**Strobe:** A momentary signal which indicates the occurrence of a specific event.

**Subroutine:** A segment of a program which can be executed by a single *call*. Subroutines are used to perform the same sequence of instructions at many different places in one program.

**Syntax:** The structure of instructions in a given *language*. If you make a mistake in entering an instruction and garble the syntax, the computer sometimes calls this a "SYNTAX ERROR."

**Text:** Characters, usually letters and numbers. "Text" usually refers to large chunks of English, rather than computer, language.

**Toggle switch:** A two-position switch which can only flip from one position to the other and back again, and cannot be directly set either way.

**Trace:** An etched conductive path on a *Printed-Circuit Board* which serves to electronically connect components.

Video: 1) Anything visual. 2) Information presented on the face of a cathode-ray tube.

Warm-start: To restart the operation of a computer after you have lost control of its language or operating system.

**Window:** Something out of which you jump when the power fails and you lose a large program. Really: a reserved area on a *display* which is dedicated to some special purpose.



Here are some other publications which you might enjoy:

#### Synertek/MOS Technology 6500 Programming Manual

This manual is an introduction to machine language programming for the MC6502 microprocessor. It describes the machine lanuage operation of the Apple's microprocessor in meticulous detail. However, it contains no specific information about the Apple.

This book is available from Apple. Order part number A2L0003.

#### Synertek/MOS Technology 6500 Hardware Manual

This manual contains a detailed description of the internal operations of the Apple's 6502 microprocessor. It also has much information regarding interfacing the microprocessor to external devices, some of which is pertinent to the Apple.

This book is also available from Apple. Order part number A2L0002.

#### The Apple II Monitor Peeled

This book contains a thorough, well-done description of the operating subroutines within the Apple's original Monitor ROM.

This is available from the author:

William E. Dougherty 14349 San Jose Street Los Angeles, CA 91345

#### Programming the 65\( \text{02} \)

This book, written by Rodnay Zaks, is an excellent tutorial manual on machine and assembly-language programming for the Apple's 6502 microprocessor.

This manual is available from Sybex Incorporated, 2020 Milvia, Berkeley, CA 94704. It should also be available at your local computer retailer or bookstore. Order book number C202.

#### 6502 Applications

This book, also written by Rodnay Zaks, describes many applications of the Apple's 6502 microprocessor.

This is also available from Sybex. Order book number D302.

#### System Description: The Apple II

Written by Steve Wozniak, the designer of the Apple computers, this article describes the basic construction and operation of the Apple II.

This article was originally published in the May, 1977 issue of BYTE magazine, and is available from BYTE Publications, Inc. Peterborough, NH 30458.

#### SWEET16: The 65\( \textit{0} \)2 Dream Machine

Also written by Steve Wozniak, this article describes the SWEET16® interpretive machine language enclosed in the Apple's Integer BASIC ROMs.

This article appeared in the October, 1977 issue of BYTE magazine, and is available from BYTE Publications, Inc. Peterborough, NH 30458.

#### More Colors for your Apple

This article, written by Allen Watson III, describes in detail the Apple High-Resolution Graphics mode. Also included is a reply by Steve Wozniak, the designer of the Apple, describing a modification you can make to update your Revision Ø Apple to add the two extra colors available on the Revision 1 board.

This article appeared in the June, 1979 issue of BYTE magazine, and is available from BYTE Publications, Inc. Peterborough, NH 30458.

#### Call APPLE (Apple Puget Sound Program Library Exchange)

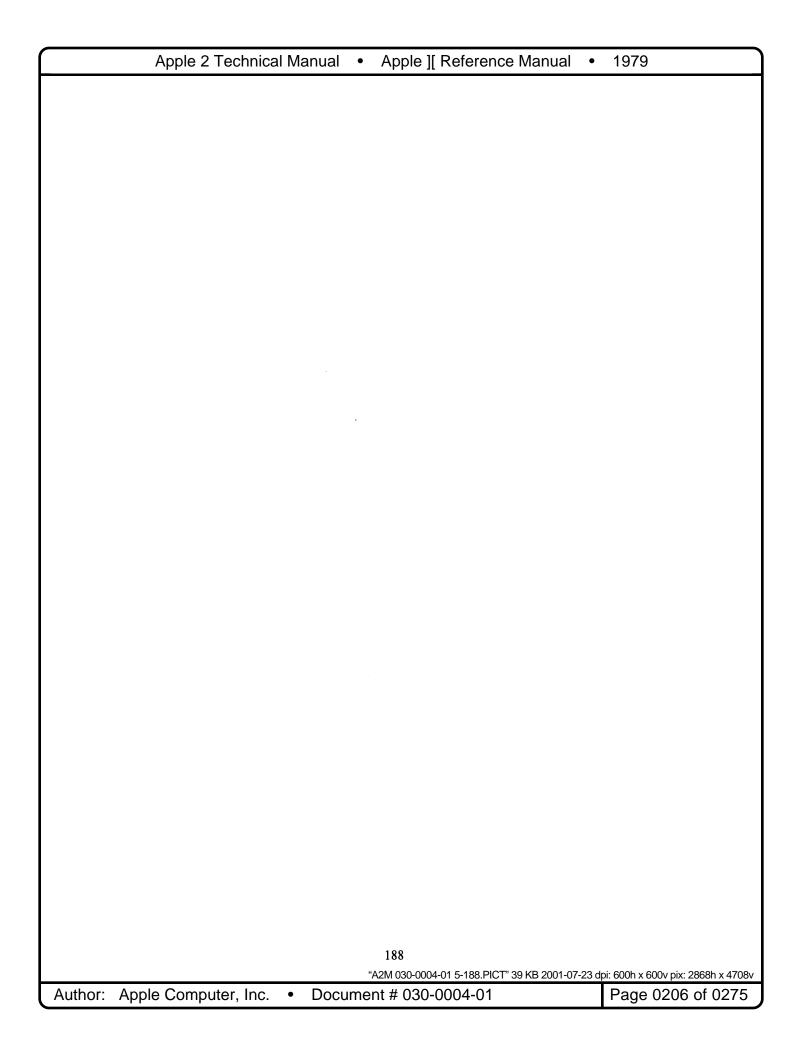
This is one of the largest Apple user group newsletters. For information, write:

Apple Puget Sound Program Library Exchange 6708 39th Ave. Southwest Seatte, Wash., 98136

#### The Cider Press

This is another large club newsletter. For information, write:

The Cider Press c/o The Apple Core of San Francisco Box 4816 San Francisco, CA 94101



194 INDEX OF FIGURES

195 INDEX OF PHOTOS

195 INDEX OF TABLES

195 CAST OF CHARACTERS

"A2M 030-0004-01 5-189.PICT" 13963 KB 2001-07-23 dpi: 600h x 600v pix: 3100h x 4705v

## **GENERAL INDEX**

Ø boards, Revision3, 26	buses, address and data	88. 9
1 board, Revision3, 26	byte, power-up	37. 6:
2716 type PROMs94		
50Hz modification, Eurapple10	C	
6502 instruction setAppendix A		
65Ø2 internal registers53, 81	card, Apple Language	4
65Ø2 microprocessor3, 88	card, Apple Firmware	7
,	cassette interface jacks	22 103
A	cassette interface	22, 10.
	cassette tape, saving to	Λ
Access Memory (RAM), Random3	cassette tape, reading from	40
address and data buses88, 90	changing memory	47
address multiplexer, RAM96	character code, ASCII	7 0 14
addresses and data40	character, backspace	1, 7, 0, 13
addressing modes66	character line feed	30
analog inputs24	character, line-feed	30
annunciator outputs23, 36, 100	character, RETURN	30
annunciator special locations24	character, bell	31
Apple Firmware card	characters, prompting	
Apple Language card	characters, keyboard	
Apple main board, the	characters, control	7
Apple Mani coambler 40	clearing the keyboard strobe	6
Apple Mini-assembler	code, ASCII character5, 6	, 7, 8, 15
Apple, photo of the2	codes, escape	34
Apple power supply, the	codes, keys and ASCII	7
Apple, setting up the2	cold start	36
Apples, varieties of	colors, Low-Res	11, 17
ASCII character code5, 6, 7, 8, 15	colors, High-Res1	
ASCII codes, keys and7	colors, European High-Res	
Autostart ROM listingAppendix C	command loops, Monitor	56
Autostart ROM Reset36	commands, creating your own	57
Autostart ROM special locations37	commands, summary of Monitor	59
Autostart ROM25	comparing memory	46
auxiliary video connector9	configuration block pinout	71
	configuration blocks, RAM	70
B	configuration, RAM memory	70
	connector pinout, peripheral	106
backspace character30	connector, keyboard	5, 102
backspace key34	connector, power	104
BASIC, entering34, 54	connector, speaker	105
BASIC, reentering34, 54	connector, Game I/O	23 100
bell character31	connector, auxiliary video	9
block pinout, configuration71	connector, video	
blocks, RAM configuration70	connectors, peripheral	3 105
board I/O, peripheral79	connnector pinouts, keyboard	103
board, Revision Ø3, 26	control characters	7
board, Revision 13, 26	control values, Normal/Inverse	22
board, the Apple main3, 89	Controllers, Game	
board schematic, main110	COUT, KEYIN switches	02 , 44,
buffer, picture12		
buffer, input	COUT standard output subroutine	
built-in I/O78, 98	creating your own commands	
11 11 0	CSW/KSW switches	83

190

"A2M 030-0004-01 5-190.PICT" 453 KB 2001-07-23 dpi: 600h x 600v pix: 3004h x 4672v

Apple 2 Technical Manual • App	ole ][ Reference Manual • 1979
cursor30	High-Res graphics19
cursor, output30	High-Res screen, the21
cycle, the RESET36	High-Res video mode, the
cycle, the RESET	High-Res colors
D	111gn-1CG CO101319, 20
D	I
data buses, address and90	•
data, addresses and40	input buffer33
debugging programs51	input line, editing an
display special locations, video	input lines, GETLN and33
display, video9	input prompting32
display, video	input subroutine, RDKEY standard32
E	input/output features20
<b>L</b>	input/output special locations
editing an input line33	input/output
editing features25	inputs, data
entering BASIC34, 54	inputs, one-bit ("flag")24, 78
entering the Monitor40	inputs, one-off ( hag )24, 78
entry vector, soft	inputs, single-bit pushbutton
escape (ESC) codes	instruction set, 6502
Eurapple 50Hz modification	instructions, Mini-Assembler
European High-Res colors	interface jacks, cassette
examining memory41	interface, cassette
expansion ROM84	internal registers, 650253, 81
T.	interrupts
F	inverse text mode
C	I/O connector, Game23, 100
feature, the Stop-List26, 30	I/O programming suggestions80
features, input/output20	I/O special locations79
features, editing25	I/O, built-in78, 98
features, keyboard5	I/O, peripheral board79
features, microprocessor88	I/O, peripheral slot79
features, power supply92	<b>.</b>
Firmware card, Apple	J
("flag") inputs, one-bit24, 78	
format, Text screen16	jacks, cassette interface22, 103
format, Low-Res screen	jacks, video output97
format, High-Res screen21	jumper, "USER 1"99
from cassette tape, reading47	
G	К
	key, backspace34
Game Controllers24	key, retype34
Game I/O connector23, 100	keyboard characters7, 8
generator, the video96	keyboard connector5, 102
GETLN and input lines33	keyboard connnector pinouts103
graphics modes	keyboard features5
graphics, High-Res19	keyboard schematic
graphics, Low-Res	keyboard special locations6
5	keyboard strobe6, 78, 79, 98, 102
H	keyboard strobe, clearing the6
<del></del>	keyboard, review of the4, 100
hexadecimal notation40	keyboard, review of the
High-Res colors, European20	KEYIN switches, COUT,83
Tagai Res Colors, Daropean20	122111 0 11101100, 0001,

keys and ASCII codes7	mode, the Low-Res video	17
	mode, the High-Res video	
L	mode, inverse text	
	mode, normal text	
Language card, Apple3, 69	modes, addressing	
leaving the Mini-Assembler50	modes, graphics	
line, editing an input33	modification, Eurapple 50Hz	
line-feed character30	Monitor command loops	
lines, GETLN and input33	Monitor commands, summary of	
listing, Autostart ROMAppendix C	Monitor prompt (*)	
listing, Monitor ROMAppendix C	Monitor ROM RESET	
listing machine language programs49	Monitor ROM listingAppendix	
list of special locationsAppendix B	Monitor ROM	
locations, list of specialAppendix B	Monitor special locations	
locations, annunciator special24	Monitor subroutines, some useful	
locations, video display special	Monitor, entering the	
locations, input/output special25	moving memory	
locations, text window special	multiplexer, RAM address	
locations, Autostart ROM special	multiplexel, K/M/I address	70
locations, Monitor special	N	
locations, keyboard special	14	
locations, I/O special	normal text mode	27
loops, Monitor command56	Normal/Inverse control values	
Low-Res colors	notation, hexadecimal	
Low-Res screen, the	number, random	
Low-Res video mode, the	number, random	33
lukewarm start	0	
iukewaiiii stait50	0	
M	one (system stack), page	69
	one-bit ("flag") inputs24, 25,	
machine language programs, listing49	output cursor	
main board, the Apple3, 89	output jacks, video	
main board schematic	output subroutine, COUT standard	
map, system memory68	output, utility strobe	
maps, zero page memory74	outputs, annunciator	
Memory (RAM), Random Access3	outputs, strobe	
Memory (ROM), Read-Only3	own commands, creating your	
memory configuration, RAM70	own communes, creating your	, ,
memory map, system68	_	
memory maps, zero page	P	
memory maps, zero page	P	
memory pages 68	-	74
memory pages	page memory maps, zero	
memory, examining41	page memory maps, zero	69
memory, examining	page memory maps, zero	69 74
memory, examining	page memory maps, zero	69 74 12
memory, examining	page memory maps, zero	69 74 12 68
memory, examining.       41         memory, changing.       43         memory, moving.       44         memory, comparing.       46         memory, RAM.       68, 95	page memory maps, zero	69 74 12 68 79
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94	page memory maps, zero	69 74 12 68 79 06
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94         microprocessor features       88	page memory maps, zero	69 74 12 68 79 06 05
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94         microprocessor features       88         microprocessor, 6502       3, 88	page memory maps, zero	69 74 12 68 79 06 05 79
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94         microprocessor features       88         microprocessor, 6502       3, 88         Mini-Assembler instructions       66	page memory maps, zero	69 74 12 68 79 06 05 79 82
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94         microprocessor features       88         microprocessor, 6502       3, 88         Mini-Assembler instructions       66         Mini-Assembler prompt (!)       50	page memory maps, zero	69 74 12 68 79 06 05 79 82 80
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94         microprocessor features       88         microprocessor, 6502       3, 88         Mini-Assembler instructions       66         Mini-Assembler prompt (!)       50         Mini-Assembler, Apple       49	page memory maps, zero	69 74 12 68 79 06 05 79 82 80
memory, examining       41         memory, changing       43         memory, moving       44         memory, comparing       46         memory, RAM       68, 95         memory, ROM       72, 94         microprocessor features       88         microprocessor, 6502       3, 88         Mini-Assembler instructions       66         Mini-Assembler prompt (!)       50	page memory maps, zero	69 74 12 68 79 06 05 79 82 80 .2

Apple 2 Technical Manual • Apple 2 Technical Manual	ople ][ Reference Manual • 1979
pinout, configuration block71	ROM, Autostart25
pinout, ROM95	ROM, Monitor25
pinout, RAM96	ROM, peripheral slot80
pinouts, keyboard connnector103	ROM or PROM, expansion84
power connector104	running machine language programs48
power supply features92	
power supply schematic93	S
power supply, the Apple2, 28, 92	
power-up byte37, 65	saving to cassette tape46
programming suggestions, I/O80	schematic, keyboard101
programs, running machine language48	schematic, power supply93
programs, listing machine language48	schematic, main board110
programs, debugging51	screen format11
PROM, peripheral card80	screen format, Text16
PROM, expansion ROM or84	screen format, High-Res21
PROMs, 2716 type94	screen format, Low-Res18
prompt (*), Monitor40	screen pages12
prompt (!), Mini-Assembler50	screen soft switches12
prompting characters33	screen, the text16
prompting, input32	screen, the Low-Res
pushbutton inputs, single-bit78	screen, the High-Res21
	set, 6502 instructionAppendix A
R	setting up the Apple2
	signals and relationships, timing91
RAM address multiplexer96	single-bit pushbutton inputs
RAM configuration blocks70	slot I/O, peripheral79
RAM memory configuration70	slot RAM, peripheral82
RAM memory68, 95	slot ROM, peripheral80
RAM pinout96	soft entry vector37
RAM, peripheral slot82	soft switches12, 79, 98
random access memory (RAM)3	soft switches, screen
random number	speaker connector
RDKEY standard input subroutine32	special locations, list ofAppendix B
reading from cassette tape47	special locations, video display
reading the keyboard6	special locations, input/output
read-only memory (ROM)3	special locations, text window
reentering BASIC	special locations, Autostart ROM37
registers, 6502 internal53, 81	special locations, Monitor
relationships, timing signals and91	special locations, keyboard6
RESET cycle, the	special locations, I/O
RESET, Autostart ROM	stack), page one (system
RESET, Monitor ROM	standard input subroutine, RDKEY32
return character	standard output subroutine, COUT30 start, cold36
retype key	start, lukewarm36
review of the keyboard4, 100	start, warm
Revision Ø boards	STEP and TRACE
ROM listing, AutostartAppendix C	Stop-List feature, the
ROM listing, AutostartAppendix C ROM listing, MonitorAppendix C	strobe output, utility25
ROM memory72, 94	strobe output, utility
	strobe, clearing the keyboard6
ROM pinout         95           ROM RESET, Autostart         36	subroutine, COUT standard output30
ROM RESET, Autostart	subroutine, RDKEY standard input32
ROM special locations, Autostart37	subroutines, some useful Monitor61
Rom special locations, Autostalt	Substitution, some assist monitorini.

#### suggestions, I/O programming ......80 window, the text.....31 summary of Monitor commands ......59 supply features, power......92 -- Y -supply schematic, power......93 supply, the Apple power ......2, 28, 92 your own commands, creating.....57 switches, soft......12, 79, 98 switches, screen soft......12 switches, toggle.....79 switches, COUT, KEYIN .....83 zero page memory maps......74 switches, CSW/KSW......83 zero, page ......69, 74 system memory map......68 (system stack), page one ......69 system timing ......90 INDEX OF FIGURES -- T -tape, saving to cassette ......46 Figure 1. Map of the Text screen.....16 tape, reading from cassette......47 Figure 2. Map of the Low-Res mode .......18 text mode, inverse.....32 Figure 3. Map of the High-Res screen ......21 text mode, normal.....32 Figure 4. Cursor-moving escape codes......35 text screen, the......11, 16 Figure 5. System Memory Map......68 text video mode, the ......14 Figure 6. Memory Configurations ......71 text window special locations.....31 Figure 7. Configuration Block Pinouts ......71 text window, the......31 Figure 8. Expansion ROM Enable circuit...85 timing signals and relationships......91 Figure 9. \$CFXX decoding ......85 timing, system ......90 Figure 10. The Apple Main Board.....89 toggle switches......79 Figure 11. Timing Signals.....91 TRACE, STEP and......26, 51 Figure 12. Power Supply Schematic .......93 Figure 13. ROM Pinout......95 -- U --Figure 14. RAM Pinouts.....96 Figure 15. Auxiliary Video Connector ......98 "USER 1" jumper......99 Figure 16. Game I/O Connector Pinout...100 useful Monitor subroutines, some......61 Figure 17. Keyboard Schematic Drawing .101 utility strobe output......25 Figure 18. Keyboard connector Pinout....103 Figure 19. Power Connector......104 -- V --Figure 20. Speaker Connector ......105 Figure 21. Peripheral Connector Pinout...106 values, Normal/Inverse control.....32 Figure 22. Main Board Schematic.....110-115 varieties of Apples......25 vector, soft entry.....37 video connector.....9 video connector, auxiliary ......9 video display.....9 video display special locations ......13 video generator, the ......96 video mode, the text .....14 video mode, the Low-Res ......17 video mode, the High-Res .....19 video output jacks......97 -- W -warm start......36 window special locations, text ......31

Apple | Reference Manual

1979

194

Apple 2 Technical Manual

## **INDEX OF PHOTOS**

Photo 1.	The Apple II	2
	The Apple Power Supply	
	The Apple Keyboard	
	The Video Connectors	
	Eurapple jumper pads	
	The Apple Character Set	
	The Game I/O Connector	
	The USER 1 Jumper	

## **INDEX OF TABLES**

Table 1.	Keyboard Special Locations	6
Table 2.	Keys and their ASCII codes	
Table 3.	The ASCII Character Set	8
Table 4.	Video Display Memory Ranges	.12
Table 5.	Screen Soft Switches	.13
Table 6.	Screen Mode Combinations	13
Table 7.	ASCII Screen Character Set	15
Table 8.	Low-Resolution Colors	17
Table 9.	Annunciator Special Locations	
Table 10		25
Table 11		31
Table 12		
Table 13		
Table 14		
Table 15		
Table 16		
Table 17	0	
Table 18		
Table 19		74
Table 20		75
Table 21		75
Table 22	. Built-In I/O Locations	79
Table 23		
Table 24		
Table 25		
Table 26	. I/O Scratchpad RAM Addresses	83
Signal D	Descriptions:	
Table 27	• • •	
Table 28		
Table 29		100
Table 30		102
Table 31	. Power Connector	104
Table 32		
Table 33	. Peripheral Connector1	07fl

## **CAST OF CHARACTERS**

!			.33
#			.66
\$		50,	66
&			65
*	33,	38,	40
+			
: (colon)			.43
. (period)			
<			
>			
?			
@			
A			
В			
C			
D			
E			
F			
G			
I			
J			
K		.23,	33
L			.49 45
M	25,	<i>5</i> 5,	43
N			
R			
S			
T			
V			
W			
CTRL B		• • • • • •	.54
CTRL C			
CTRL E			
CTRL G (bell)			.30
CTRL H (←)	.30,	33,	34
CTRL J (line feed)			
CTRL K			
CTRL P			.54
CTRL S CTRL U (→)		.26,	30
CTRL U (→)		.33,	34
CTRL X			.33
CTRL Y		.57,	58
ESC		.25.	34
RETURN	.30.	33.	43
\		8	33
]			.33
^			.50
		•••••	

195

"A2M 030-0004-01 5-195.PICT" 438 KB 2001-07-23 dpi: 600h x 600v pix: 3022h x 4690v



# Apple II Computer Reference Manual Information

# Photo Collection

Apple Part # 030-0004-01 (1979)

"A2M 030-0004-01 7--0a.PICT" 201 KB 2001-07-23 dpi: 600h x 600v pix: 3877h x 4917v

# INDEX OF PHOTOS

Photo 1.	The Apple II	2
Photo 2.	The Apple Power Supply	3
Photo 3.	The Apple Keyboard	6
Photo 4.	The Video Connectors	10
Photo 5.	Eurapple jumper pads	11
Photo 6.	The Apple Character Set	14
Photo 7.	The Game I/O Connector	23
Photo 8.	The USER 1 Jumper	99
	<del>-</del>	

"A2M 030-0004-01 7--0b.PICT" 182 KB 2001-07-23 dpi: 1200h x 1200v pix: 2999h x 1795v

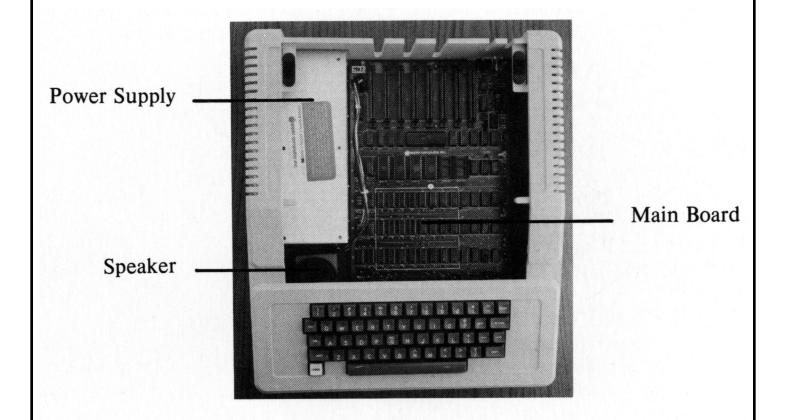
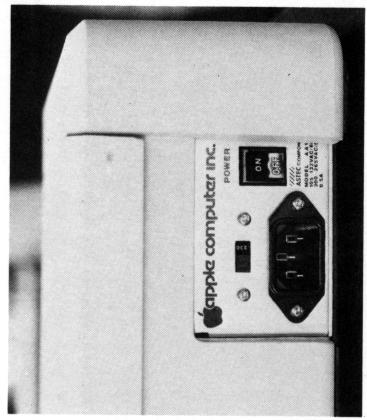


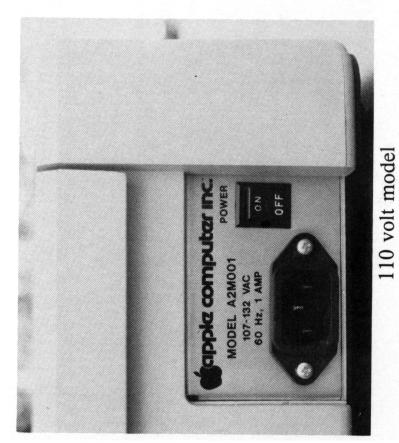
Photo 1. The Apple II.

"A2M 030-0004-01 7-02.PICT" 9905 KB 2001-07-23 dpi: 1200h x 1200v pix: 4468h x 3699v



110/220 volt model

Photo 2. The back of the Apple Power Supply.



"A2M 030-0004-01 7-03.PICT" 14911 KB 2001-07-23 dpi: 1200h x 1200v pix: 3007h x 5846v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0217 of 0275

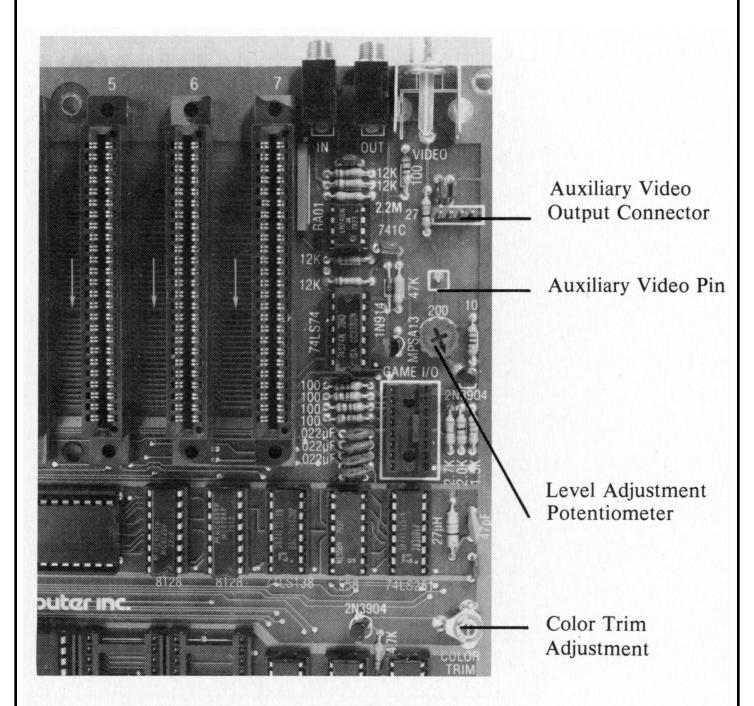


Photo 4. The Video Connectors and Potentiometer.

"A2M 030-0004-01 7-10.PICT" 18708 KB 2001-07-23 dpi: 1200h x 1200v pix: 4684h x 4649v

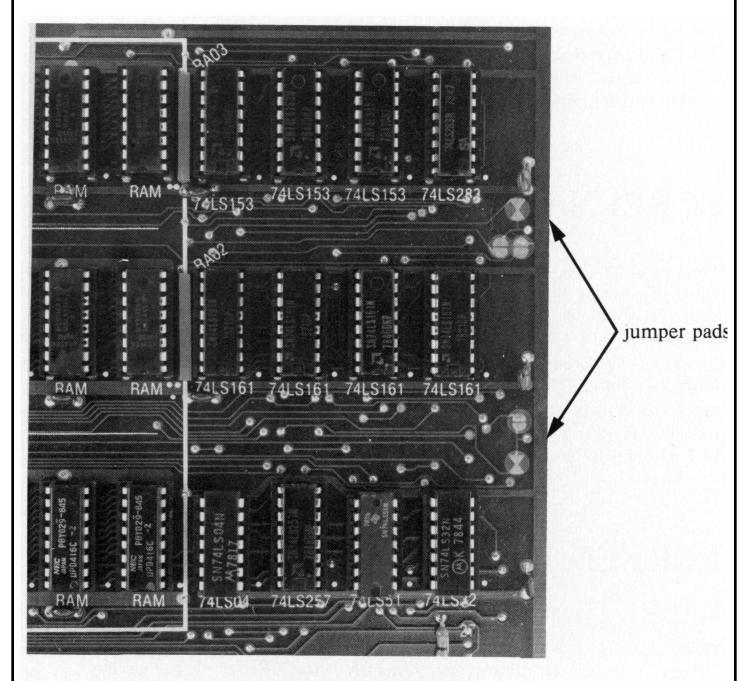


Photo 5. Eurapple (50 hz) Jumper Pads.

"A2M 030-0004-01 7-11.PICT" 20200 KB 2001-07-23 dpi: 1200h x 1200v pix: 4779h x 4684v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0220 of 0275

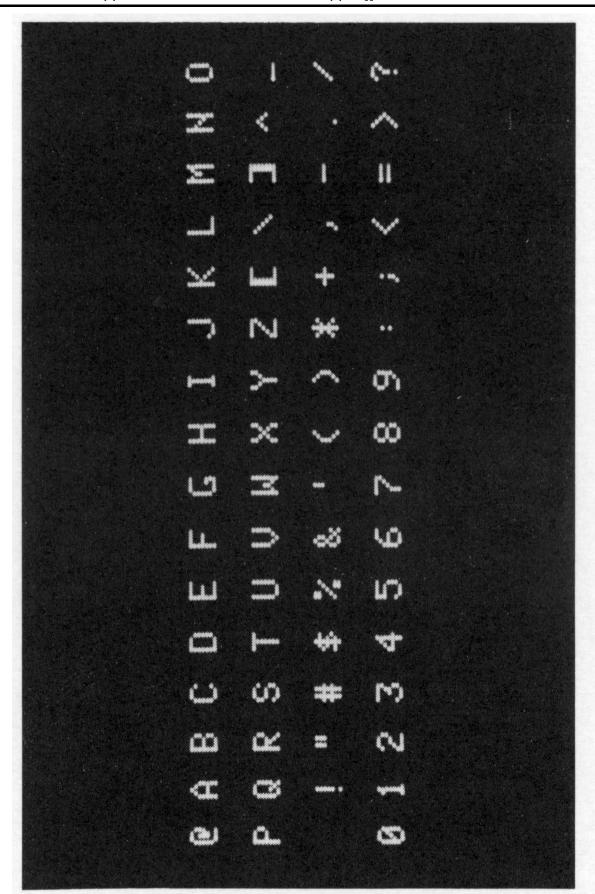
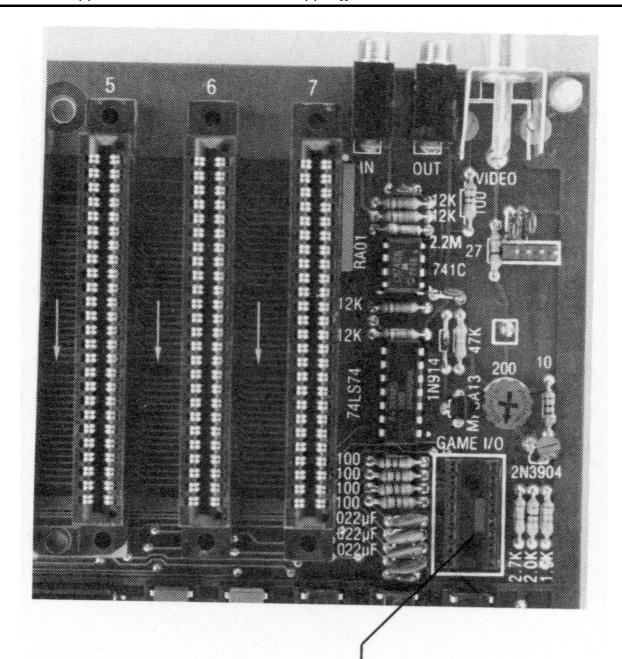


Photo 6. The Apple Character Set

"A2M 030-0004-01 7-14.PICT" 10483 KB 2001-07-23 dpi: 1200h x 1200v pix: 2834h x 3808v



## Photo 7. The Game I/O Connector.

"A2M 030-0004-01 7-23.PICT" 6910 KB 2001-07-23 dpi: 1200h x 1200v pix: 2503h x 3735v

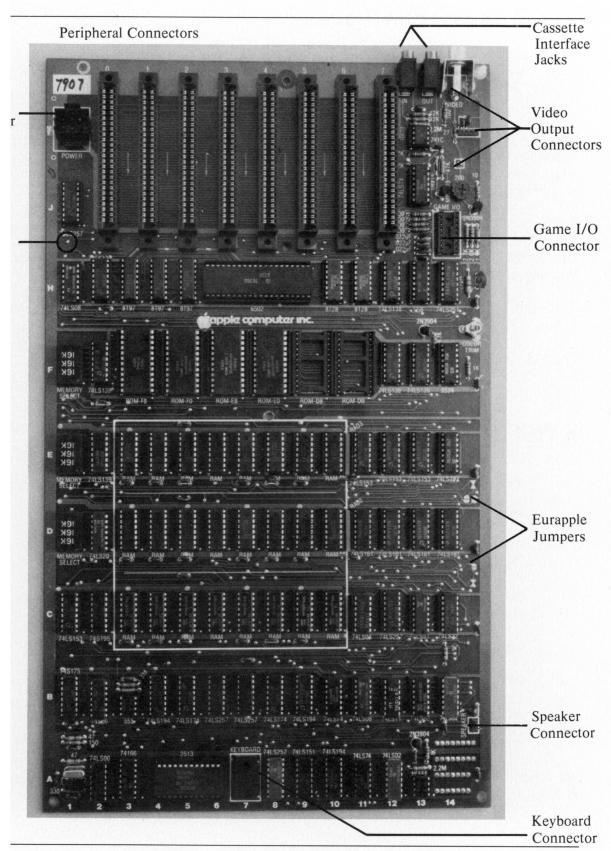


Figure 10. The Apple Main Board

"A2M 030-0004-01 7-89.PICT" 19538 KB 2001-07-23 dpi: 800h x 800v pix: 3800h x 5534v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0223 of 0275

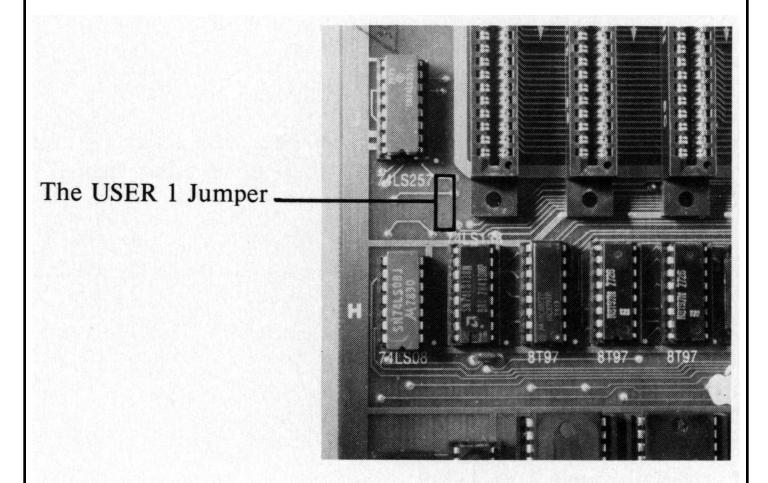


Photo 8. The USER 1 Jumper.

"A2M 030-0004-01 7-99.PICT" 9970 KB 2001-07-23 dpi: 1200h x 1200v pix: 3843h x 2848v



## Apple II Computer Reference Manual Information

## Table Collection

Apple Part # 030-0004-01 (1979)

"A2M 030-0004-01 8-t-1a.PICT" 194 KB 2001-07-23 dpi: 600h x 600v pix: 3866h x 4917v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0225 of 0275

1979

## **INDEX OF TABLES**

Table 1.	Keyboard Special Locations	6
Table 2.	Keys and their ASCII codes	7
Table 3.	The ASCII Character Set	8
Table 4.	Video Display Memory Ranges	12
	Screen Soft Switches	
Table 6.	Screen Mode Combinations	13
Table 7.	ASCII Screen Character Set	15
Table 8.	Low-Resolution Colors	17
Table 9.	Annunciator Special Locations	24
Table 10.		
Table 11.	Text Window Special Locations	31
Table 12.	Normal/Inverse Control Values	32
Table 13.	Autostart ROM Special Locations	37
Table 14.	Page Three Monitor Locations	65
Table 15.	Mini-Assembler Address Formats	66
Table 16.	RAM Organization and Usage	69
Table 17.	ROM Organization and Usage	72
Table 18.	Monitor Zero Page Usage	74
Table 19.	Applesoft II Zero Page Usage	74
Table 20.	DOS 3.2 Zero Page Usage	75
Table 21.	Integer BASIC Zero Page Usage	75
Table 22.	Built-In I/O Locations	79
Table 23.	Peripheral Card I/O Locations	80
Table 24.	Peripheral Card PROM Locations	81
Table 25.	I/O Location Base Addresses	82
Table 26.	I/O Scratchpad RAM Addresses	83
	escriptions:	
Table 27.	Timing	90
Table 28.	Auxiliary Video Output	97
Table 29.	Game I/O Connector	100
Table 30.	Keyboard Connector	102
	Power Connector	
Table 32.	Speaker Connector	105
Table 33.	Peripheral Connector	107ff

"A2M 030-0004-01 8-t-1b.PICT" 735 KB 2001-07-23 dpi: 1200h x 1200v pix: 3074h x 6002v

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979
--------------------------	---	---------------------------	---	------

T	Table 1: I	Keyboard S	Keyboard Special Locations
Location:			Decription
Hex	Dec	Decimal	Description
\$C000	49152	-16384	Keyboard Data
\$CØ10	49168	-16368	Clear Keyboard Strobe

"A2M 030-0004-01 8-t01.PICT" 64 KB 2001-07-23 dpi: 600h x 600v pix: 501h x 1668v

Key	Alone	CTRL	SHIFT	Both	Key	Alone	CTRL	SHIFT	Both
space	8A0	\$A0	\$A0	\$A0	RETURN	\$8D	\$8D	\$8D	\$8D
0	\$B0	<b>\$B</b> 0	<b>\$B0</b>	\$B0	Ü	\$C7	287	\$C7	287
1:	\$B1	<b>\$B</b> 1	<b>\$</b> A1	\$A1	H	\$C8	888	\$C8	888
2,,	\$B2	<b>\$B2</b>	\$A2	\$A2	H	\$C6	886	\$C6	68\$
3#	\$B3	<b>\$B3</b>	\$A3	\$A3	<b>-</b>	\$CA	\$8A	\$CA	\$8A
4\$	<b>\$B4</b>	<b>\$B4</b>	\$A4	\$A4	¥	\$CB	\$8B	\$CB	\$8B
2%	<b>\$B</b> 5	<b>\$B</b> 5	\$A5	\$A5	T	\$CC	\$8C	\$CC	\$8C
89	<b>\$B</b> 6	<b>\$B</b> 6	\$A6	\$A6	X	\$CD	\$8D	\$DD	\$9D
7,	<b>\$B7</b>	<b>\$B7</b>	\$A7	\$A7	ζ	\$CE	\$8E	\$DE	\$9E
)8	<b>\$B8</b>	<b>\$B8</b>	\$A8	\$A8	0	\$CF	\$8F	\$CF	\$8F
6	<b>\$B</b> 6	\$B9	\$A9	\$A9	P@	8D0	068	\$CØ	880
*.	<b>\$BA</b>	\$BA	\$AA	\$AA	0	\$D1	\$91	<b>\$</b> D1	\$91
+	\$BB	\$BB	\$AB	\$AB	8	<b>\$D2</b>	\$92	<b>\$</b> D2	\$92
V,	\$AC	\$AC	\$BC	\$BC	S	<b>\$D3</b>	\$93	<b>\$D3</b>	\$93
	\$AD	\$AD	\$BD	\$BD		<b>\$D4</b>	894	<b>\$D4</b>	894
<u>\</u>	\$AE	\$AE	\$BE	\$BE	n	<b>\$D</b> 5	\$68	<b>\$D</b> 5	\$6\$
¿/	\$AF	\$AF	<b>\$BF</b>	\$BF	>	\$D6	96\$	<b>\$D</b> 6	96\$
A	\$C1	\$81	\$C1	\$81	A	<b>\$D</b> 7	268	<b>\$D7</b>	26\$
В	\$C2	\$82	\$C2	\$82	×	<b>\$D8</b>	86\$	<b>\$D8</b>	86\$
ن ک	\$C3	\$83	\$C3	\$83	Y	\$D6	866		899
Ω	\$C4	\$84	\$C4	\$84	Z	<b>\$DA</b>	89A	\$DA	\$9A
Щ	\$C5	\$85	\$C5	\$85	1	88 <del>\$</del>	888	$\infty$	888
Щ	\$C6	98\$	\$C6	98\$	1	\$68	\$6\$	\$6\$	\$6\$
					しび日	SOR.	40B		000

Apple 2 Technical Manual • Apple ][ Reference Manual • 1979

"A2M 030-0004-01 8-t02.PICT" 249 KB 2001-07-23 dpi: 600h x 600v pix: 2195h x 2872v

Author: Apple Computer, Inc. • Document # 030-0004-01

Page 0228 of 0275

		Apple	e 2 To	echn	ical I	Manı	ıal	• /	Apple	:][R	efere	ence	Man	ual	• ′	1979		
	240	\$F0	d	ď	<b>L</b>	S	<b>+</b>	n	>	*	×	>	2	-	,		₹	rub
	224	\$E0		ಡ	þ	ပ	p	Φ	Ŧ	5.0	h	•—	.—	<b>×</b>		ш	п	0
Set	208	\$D0	Ь	0	2	S	Η	n	>	≽	×	X	7	_	_		<b>&lt;</b>	i
Character	192	\$C0	<b>®</b>	Y	В	Ŋ	Ω	田	Ľ	Ů	Н	_	т	×	J	$\mathbf{Z}$	Z	0
	176	\$B0	9		2	က	4	2	9	7	<b>∞</b>	6	••	• •	٧		٨	٠.
The ASCI	160	\$A0		<del>(**</del> -1	<b>=</b>	#	<del>⇔</del>	%	ઝ		<u> </u>		*	+	^	I	•	_
Table 3:	144	06\$	dle	dc1	dc2	dc3	dc4	nak	syn	etb	can	em	qns	esc	fs	gs	rs	sn
Ta	128	880	nul	soh	stx	etx	eot	end	ack	bel	ps	ht	If	vt	Ħ	cr	SO	Si
	ecimal:	Hex:	80	\$1	\$2	\$3	\$4	\$5	98	24	& <del>\$</del>	6\$	<b>\$</b>	\$B	&C	\$D	<b>\$</b> E	<b>\$</b> F
	Deci		0		7	m	4	2	9	7	<b>∞</b>	6	10	11	12	13	14	15

"A2M 030-0004-01 8-t03.PICT" 117 KB 2001-07-23 dpi: 600h x 600v pix: 1620h x 2158v

2 Technical Manual • Apple   Referen	nce Manual	•	1979
--------------------------------------	------------	---	------

Apple

T	Table 4: V	ideo [	Display	: Video Display Memory Ranges	ınges	
	Dece		Begins at:	ıt:	Ends at:	
Scienti	rage		Hex	Decimal		
Text/Lo-Res	Primary		\$400	1024	\$7FF	2047
	Secondary	ary	8800	2048	<b>\$BFF</b>	3071
Hi-Res	Primary	<b>&gt;</b>	\$2000	8192	<b>\$3FFF</b>	16383
	Secondary	ary	\$4000	16384	\$5FFF	24575

"A2M 030-0004-01 8-t04.PICT" 89 KB 2001-07-23 dpi: 600h x 600v pix: 685h x 2104v

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979
--------------------------	---	---------------------------	---	------

		Table 5: S	Table 5: Screen Soft Switches
Location			
Hex	Decimal		Description:
\$C050	49232	-16304	Display a GRAPHICS mode.
\$CØ51	49233	-16303	Display TEXT mode.
\$CØ52	49234	-16302	Display all TEXT or GRAPHICS.
\$CØ53	49235	-16301	Mix TEXT and a GRAPHICS mode.*
\$C054	49236	-16300	Display the Primary page (Page 1).
\$CØ55	49237	-16299	Display the Secondary page (Page 2).
\$CØ56	49238	-16298	Display LO-RES GRAPHICS mode.*
\$C057	49239	-16297	Display HI-RES GRAPHICS mode.*

"A2M 030-0004-01 8-t05.PICT" 151 KB 2001-07-23 dpi: 600h x 600v pix: 1038h x 2174v

al	•	Apple	[ Reference Manual	•
aı	-		i i vererence manuar	-

	Table 6:	Screen N	Table 6: Screen Mode Combinations	ions	
Prin	Primary Page		Secor	Secondary Page	
Screen	Switches		Screen	Switches	
All Text	\$C054	\$CØ51	All Text	\$CØ55	\$CØ51
All Lo-Res	\$CØ54	\$CØ56	All Lo-Res	\$CØ55	\$CØ56
Graphics	\$CØ52	\$C050	Graphics	\$CØ52	\$C050
All Hi-Res	\$CØ54	\$C057	All Hi-Res	\$C055	\$C057
Graphics	\$CØ52	\$C050	Graphics	\$CØ52	\$C050
Mixed Text	\$CØ54	\$CØ56	Mixed Text	\$C055	\$C056
and Lo-Res	\$CØ53	\$C050	and Lo-Res	\$CØ53	\$C050
Mixed Text	\$CØ54	\$CØ57	Mixed Text	\$CØ55	\$C057
and Hi-Res	\$CØ53	\$CØ50	and Hi-Res	\$CØ53	\$C050

"A2M 030-0004-01 8-t06.PICT" 173 KB 2001-07-23 dpi: 600h x 600v pix: 1128h x 2075v

Ap	ple 2 T	ech	nica	al M	anu	ıal	•	Ap	ple	; ][ l	Ref	erei	nce	Ma	ากนส	al	•	19	79
	rcase)	240	8F0	0	_	7	3	4	2	9	7	∞	6		٠,	V 1	H	٨	6.
	(Lowercase)	224	\$E0			=	#	<del>∽</del>	%	ઝ		$\smile$		*	+	•	1		_
		208	\$DØ	Ь	0	~	S	Н	Ω	>	≱	×	>	Z		_		•	١
	nal	192	8C0	(9)	∢	В	C	Ω	П	江	Ö	H	l	_	×	_	Σ	Z	0
	Normal	176	\$B0	0		7	3	4	2	9	7	<b>∞</b>	6			<b>V</b>	11	^	ć
cters		160	SAØ			=	#	<b>∽</b>	%	ઝ		$\smile$	_	*	+	•	ı		_
Characters	(lo.	144	06\$	Ь	0	~	S	Г	n	>	*	×	<b>&gt;</b>	7	_		_	•	-
	(Control)	128	880	(9)	V	В	၁	D	ш	Щ	Ŋ	Н	_	_	×	_	Σ	Z	0
Screen		112	870	0		7	<del>ر</del>	4	~	9	7	∞	6	•••		<b>V</b>	11	^	6
	gu	96	860			=	#	<del>69</del>	%	ઝ		$\smile$	_	*	+	•	I		
ASCI	Flashing	98	850	Ы	0	~	S	Н	Ω	>	≱	×	<b>&gt;</b>	7		_	_	•	I
le 7:		49	840	@	∢	В	C	D	丑	ц	Ŋ	Н	I	_	*	Γ	Σ	Z	0
Table		48	\$30	0	_	2	8	4	2	9	7	∞	6		. •	<b>V</b>	11	^	٠
	se	32	\$20			=	#	S	%	ઝ		<u> </u>		*	+	•	ŀ		
	Inverse	16	810	Ь	0	~	S	⊣	n	>	≱	×	<b>&gt;</b>	7	_	_	_	•	1
		0	800	@	<b>4</b>	В	C	Q	ш	ĹŢ,	Ð	Н	1	_	×	7	Σ	Z	0
			Hex	98	\$1	\$2	\$3		\$5	9\$	\$7		- 6\$	-SA	SB	SC SC	\$D	SE	\$F

"A2M 030-0004-01 8-t07.PICT" 201 KB 2001-07-23 dpi: 600h x 600v pix: 2235h x 3679v

Author: Apple Computer, Inc. • Document # 030-0004-01

Decimal

Page 0233 of 0275

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979
--------------------------	---	---------------------------	---	------

	Table 8:	8: Low-Resolution Graphics Colors	tion Graphi	cs Colo	rs
Decimal	Hex	Color	Decimal	Hex	Color
0	0\$	Black	∞	88	Brown
	\$1	Magenta	6	6\$	Orange
2	\$2	Dark Blue	10	<b>\$</b>	Grey 2
8	\$3	Purple	11	\$B	Pink
4	\$4	Dark Green	12	&C	Light Green
~	\$5	Grey 1	13	\$D	Yellow
9	9\$	Medium Blue	14	\$E	Aquamarine
7	\$7	Light Blue	15	\$ H	White

"A2M 030-0004-01 8-t08.PICT" 89 KB 2001-07-23 dpi: 600h x 600v pix: 896h x 2125v

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979

Table 9	: AI	unciator	nnunciator Special Locations	ocations
	Q+0+0	Address:		
Ann.	State	Dec	Decimal	Hex
0	off	49240	-16296	\$CØ58
	on	49241	-16295	\$CØ59
	Off	49242	-16294	\$C05A
	on	49243	-16293	\$C05B
2	Off	49244	-16292	\$C05C
	on	49245	-16291	\$C05D
3	off	49246	-16290	\$C05E
	on	49247	-16289	\$CØ5F

"A2M 030-0004-01 8-t09.PICT" 87 KB 2001-07-23 dpi: 600h x 600v pix: 1002h x 1394v

Table 10	10: Input/	: Input/Output Special Locations	ecial Locat	ions
Function:	Address: Dec	ess: Decimal	Hex	Read/Write
Speaker	49200	-16336	\$CØ3Ø	R
Cassette Out	49184	-16352	\$C020	R
Cassette In	49256	-16288	\$C060	×
Annunciators*	49240	-16296	\$CØ58	R/W
	through	through	through	
	49247	-16289	\$C05F	
Flag inputs	49249	-16287	\$CØ61	R
	49250	-16286	\$CØ62	8
	49251	-16285	\$CØ63	~
Analog Inputs	49252	-16284	\$C064	<b>8</b>
	49253	-16283	\$CØ65	
	49254	-16282	\$CØ66	
	49255	-16281	\$C067	
Analog Clear	49264	-16272	\$C070	R/W
Utility Strobe	49216	-16320	\$C040	R

"A2M 030-0004-01 8-t10.PICT" 170 KB 2001-07-23 dpi: 600h x 600v pix: 1620h x 1968v

Apple 2 recillical Maridal * Apple   Neleterice Marida	Apple 2 Technical Manual	•	Apple   Reference Manual	•	1979
--------------------------------------------------------	--------------------------	---	--------------------------	---	------

L	Table 11: Te	ext Win	1: Text Window Special Locations	Locations
	Location:		Minimum/	Minimum/Normal/Maximum Value
runction:	Decimal	Нех	Decimal	Hex
Left Edge	32	\$20	0/0/39	\$0/\$0/\$17
Width	33	\$21	0/40/40	\$0/\$28/\$28
Top Edge	34	\$22	0/0/24	\$0/\$0/\$18
Bottom Edge	35	\$23	0/24/24	\$0/\$18/\$18

"A2M 030-0004-01 8-t11.PICT" 112 KB 2001-07-23 dpi: 600h x 600v pix: 690h x 2220v

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979
--------------------------	---	---------------------------	---	------

	Table	le 12: Normal/Inverse Control Values
Value:		Effect:
Decimal	Hex	
255	\$FF	COUT will display characters in Normal mode.
63	\$3F	COUT will display characters in Inverse mode.
127	\$7F	COUT will display letters in Flashing mode, all
		other characters in Inverse mode.

"A2M 030-0004-01 8-t12.PICT" 103 KB 2001-07-23 dpi: 600h x 600v pix: 697h x 2199v

Apple 2 Technic	cal Manua	al • Apple]	[ Reference	ce Manual
3: Autostart ROM Special Locations	Contents:	Soft Entry Vector. These two locations contain the address of the reentry point for whatever language is in use. Normally contains \$E003.	Power-Up Byte. Normally contains \$45. See below.	This is the beginning of a machine language subroutine which sets up the power-up location.
Table 13: Au	Hex	\$3F2 \$3F3	\$3F4	\$FB6F
	Location: Decimal	1010	1012	64367 (-1169)

"A2M 030-0004-01 8-t13.PICT" 123 KB 2001-07-23 dpi: 600h x 600v pix: 907h x 2187v

1979

L	Table 14:	Page Three Monitor Locations	r Locations
Address:		Use:	
Decimal	Hex	Monitor ROM	Autostart ROM
1008	\$3FØ		Holds the address
1009	\$3F1		of the subroutine
	,	N C	which handles
		IAOIIG.	machine language
\$ · ·			"BRK" requests
			(normally \$FA59).
1010	\$3F2		C. F. T. V. T. 47.
1011	\$3F3	None.	Soft Entry vector.
1012	\$3F4	None.	Power-up Byte.
1013	\$3F5	Holds a "JuMP"	instruction to the
1014	\$3F6	subroutine which	subroutine which handles Applesoft II
1015	\$3F7	"%" commands.*	Normally \$4C \$58
		SFF.	
1016	\$3F8	Holds a "JuMP"	instruction to the
1017	\$3F9	subroutine which	handles "USER"
1018	\$3FA	(CTRL Y) commands	ands.
1019	\$3FB	Holds a "JuMP"	instruction to the
1020	\$3FC	subroutine which	h handles Non-
1021	\$3FD	Maskable Interrupts	S.
1022	\$3FE	Holds the address	Holds the address of the subroutine
1023	\$3FF	which handles Interrupt ReQuests.	rrupt ReQuests.

"A2M 030-0004-01 8-t14.PICT" 202 KB 2001-07-23 dpi: 600h x 600v pix: 2108h x 1858v

Table 15: Mini-Ass	Table 15: Mini-Assembler Address Formats
Mode:	Format:
Accumulator	None.
Immediate	#\${value}
Absolute	<b>\$</b> {address}
Zero Page	\${address}
Indexed Zero Page	\${address},X \${address},Y
Indexed Absolute	\${address},X \${address},Y
Implied	None.
Relative	\${address}
Indexed Indirect	(\${address},X)
Indirect Indexed	(\${address}),Y
Absolute Indirect	(\${address})

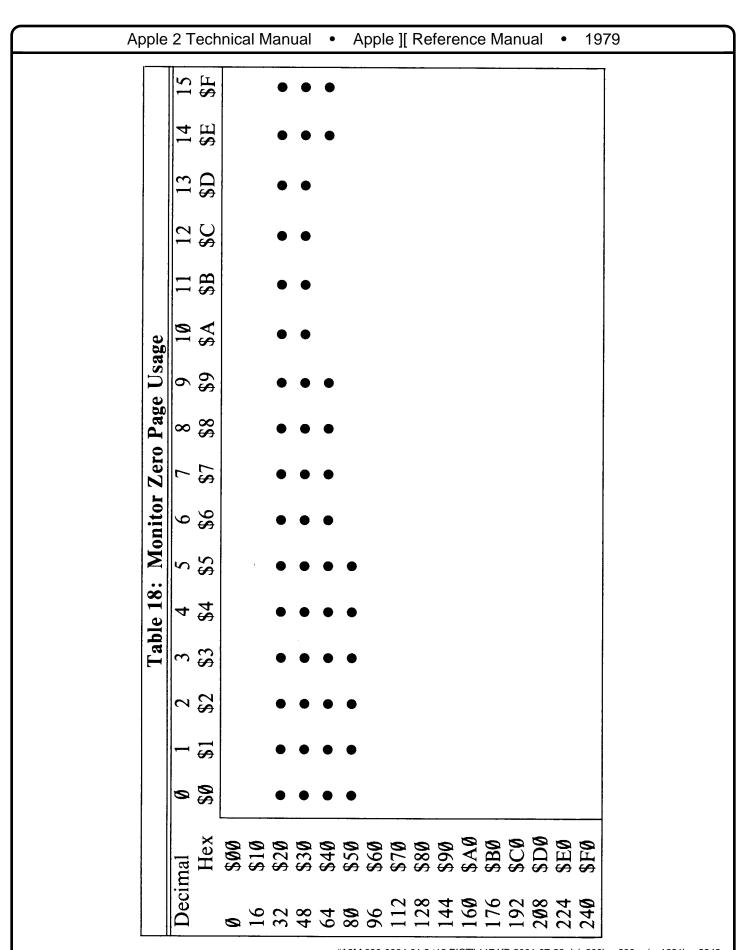
"A2M 030-0004-01 8-t15.PICT" 137 KB 2001-07-23 dpi: 600h x 600v pix: 1433h x 1458v

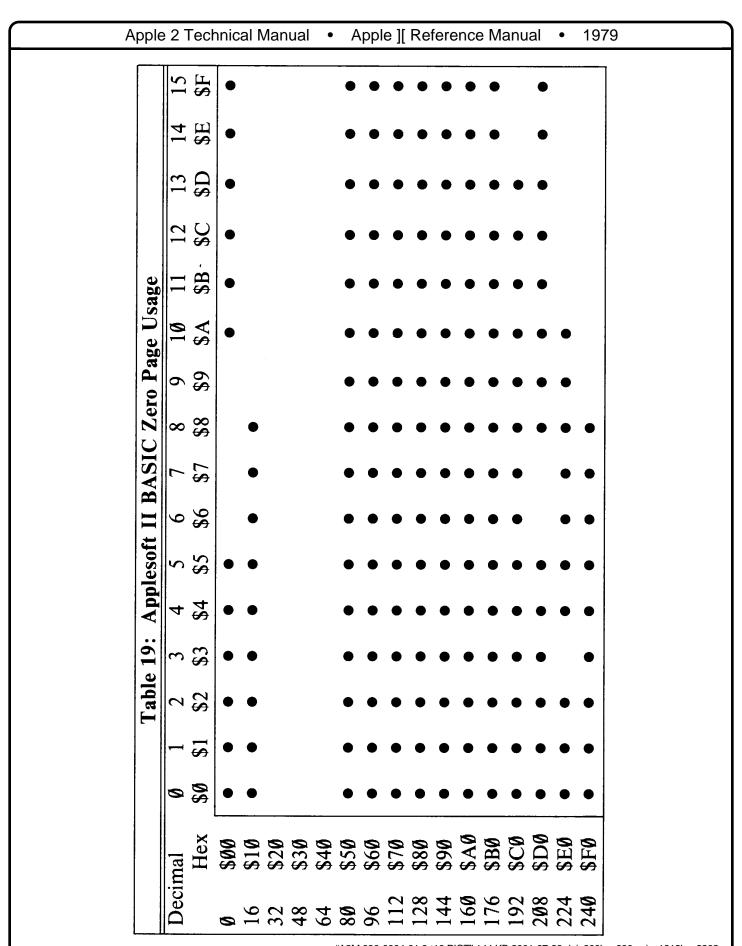
		Table 16: RAM Organization and Usage
Page Number:		Used For
Decimal	Hex	
Ø	800	System Programs
	\$01	System Stack
2	\$02	GETLN Input Buffer
3	803	Monitor Vector Locations
4	\$04	
5	805	Text and Lo-Res Graphics
9	908	Primary Page Storage
7	201	
8	808	
6	608	Text and Lo-Res Graphics
10	\$ØA	Secondary Page Storage
11	\$0B	
12	\$ØC	TKEE
through		
31	\$1F	
32	\$20	Hi-Res Graphics
through		Primary Page
63	\$3F	Storage
64	840	Hi-Res Graphics
through		Secondary Page
95	\$5F	Storage
96	09\$	
through		
191	\$BF	

"A2M 030-0004-01 8-t16.PICT" 207 KB 2001-07-23 dpi: 600h x 600v pix: 2527h x 2361v

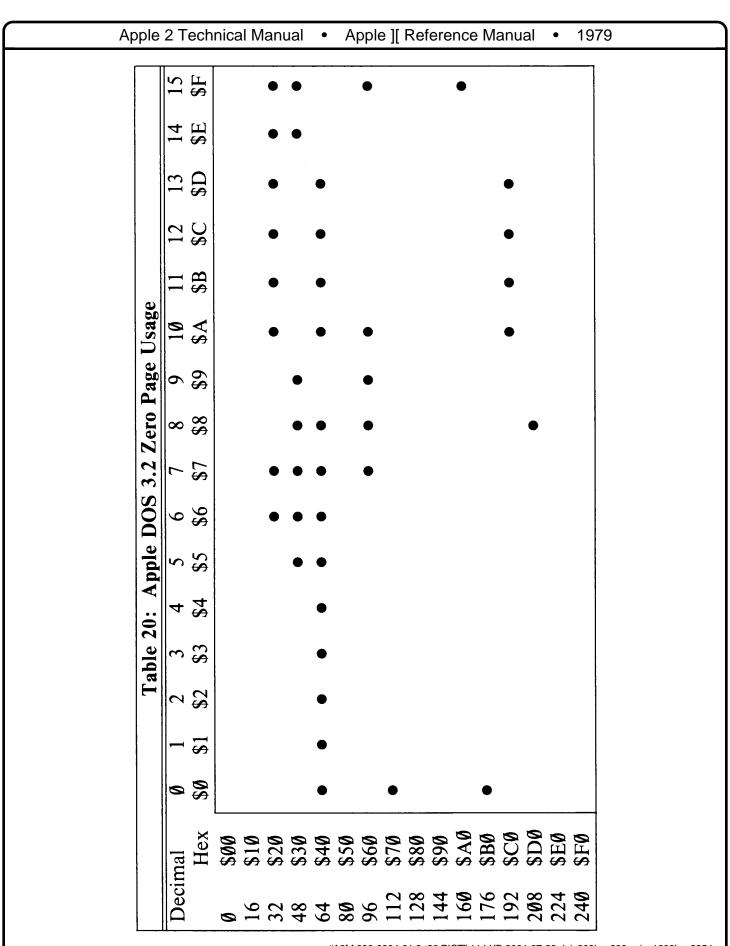
Ар	ple 2	Tech	nnica	l Mar	nual	•	Apple	e ][ R	efer	ence	Man	ual	• 1	979
and Usage						Applesoft	II	BASIC					Mod	Autostart Kolvi
17: ROM Organization and Usage	Hoad D	Osed by.	D. c. c. c. c. c. c. c. d. 4.1	rrogrammer's Aid #1					Integer BASIC			Utility Subroutines		MODITOR ROM
Table 17:	mber:	Hex	\$D0	\$D4	\$D8	\$DC	\$E0	\$E4	\$E8	\$EC	8F0	\$F4	\$F8	\$FC
	Page Number:	Decimal	208	212	216	220	224	228	232	236	240	244	248	252

"A2M 030-0004-01 8-t17.PICT" 107 KB 2001-07-23 dpi: 600h x 600v pix: 1348h x 2133v

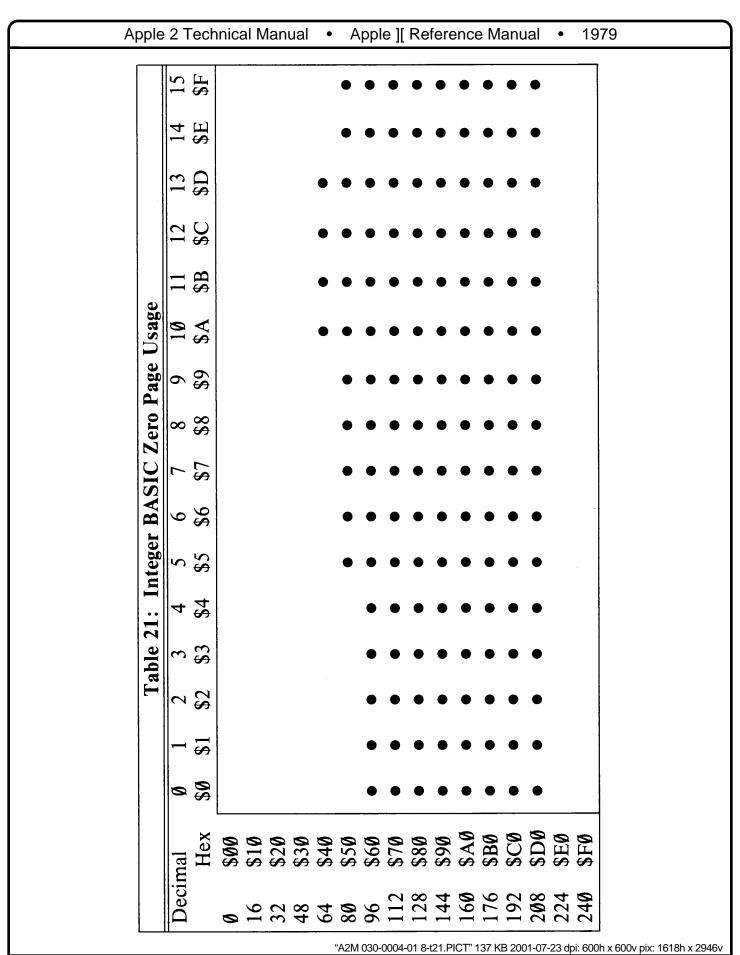




"A2M 030-0004-01 8-t19.PICT" 144 KB 2001-07-23 dpi: 600h x 600v pix: 1612h x 2902v

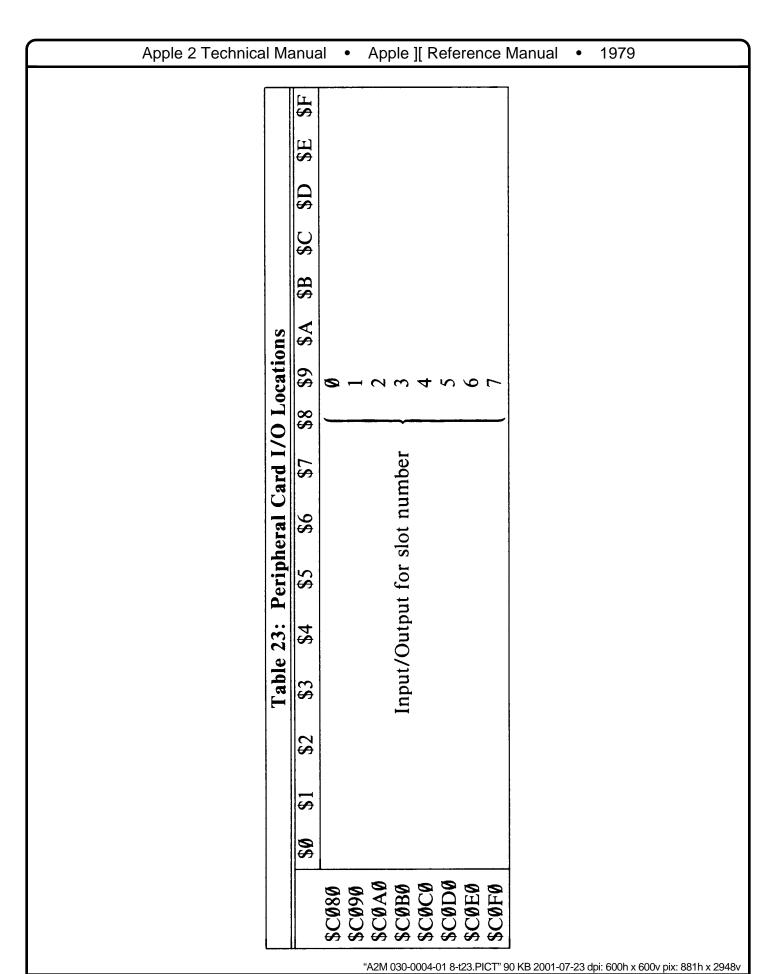


"A2M 030-0004-01 8-t20.PICT" 114 KB 2001-07-23 dpi: 600h x 600v pix: 1609h x 2954v



				L	able	22:	Built-I	Table 22: Built-In I/O Locations	Loca	tions						
	08	\$0 \$1	\$2	\$3	84	\$5	9\$	\$4 \$5 \$6 \$7 \$8 \$9 \$A \$B \$C \$D \$E \$F	88	6\$	<b>\$</b> A	\$B	\$C	\$D	\$E	\$F
\$C000 Keyboard Data Input	Key	boarc	I Data I	nput												
\$C010	Cle	ar Ke	Clear Keyboard Strobe	Strobe												
\$C020	Cas	sette	Cassette Output Toggle	Toggl	ပ											
\$C030   Speaker Toggle	Spe	aker	<b>Foggle</b>													
\$C040 Utility Strobe	Util	ity St	robe													
\$C020	gr	tx	nomix	mix	pri	sec	sec lores hires	hires	anØ	0	anl	1	an2	12	an3	8
\$C060	cin	cin pb1	pb2	pb3	gcØ	gcØ gc1	gc2	gc3			repe	eat \$C(	repeat \$C060-\$C067	190		
\$CØ7Ø   Game Controller Strobe	Gar	ne Cc	ntrolle	Strok	)e											

"A2M 030-0004-01 8-t22.PICT" 190 KB 2001-07-23 dpi: 600h x 600v pix: 997h x 2915v



Author:

Apple	e 2 T	echi	nical	Ма	nua	al	•	Ap	ole	][ R	efe	ren	ce l	Mar	nual	•	, ′	1979
		/ &C@E@	SCØF1	\$CØF2	\$CØF3	\$CØF4	\$CØF5	\$CØF6	\$CØF7	\$CØF8	\$CØF9	\$CØFA	\$CØFB	\$CØFC	\$CØFD	\$CØFE	\$CØFF	
	4	0 0 0 0 0	SCØE1	\$CØE2	\$C0E3	\$CØE4	\$CØE5	\$CØE6	\$CØE7	\$CØE8	\$CØE9	\$CØEA	\$CØEB	\$CØEC	\$CØED	\$CØEE	\$CØEF	
sses	V	SCADA	SCØD1	\$CØD2	\$CØD3	\$C004	\$CØD5	\$CØD6	\$C0D7	$\$C\emptysetD8$	\$CØD9	\$CØDA	\$CØDB	\$CØDC	\$CØDD	\$CØDE	\$CØDF	
I/O Location Base Addresses	ot 1	4	SCØC1	\$CØC2	\$CØC3	\$CØC4	\$CØC5	\$CØC6	\$CØC7	\$CØC8	\$CØC9	\$CØCA	\$CØCB	\$CØCC	\$CØCD	\$CØCE	\$CØCF	Locations
Location B	Slot	S CARA	SCØB1	\$CØB2	\$CØB3	\$CØB4	\$CØB5	\$CØB6	\$CØB7	\$CØB8	\$C0B9	\$CØBA	\$CØBB	\$CØBC	\$CØBD	\$CØBE	\$CØBF	I/O To
5:	C	2 C 0 A 0	SCØA1	\$CØA2	\$CØA3	\$CØA4	\$C0A5	\$CØA6	\$CØA7	\$CØA8	\$CØA9	\$CØAA	\$CØAB	\$CØAC	\$CØAD	\$CØAE	\$CØAF	
Table 2	-	\$C000	\$CØ91	\$CØ92	\$CØ93	\$C094	\$CØ95	\$CØ96	\$C097	\$C098	\$CØ39	\$C09A	\$C09B	\$C09C	\$C09D	\$C09E	\$C09F	
	8	\$C080	\$CØ81	\$CØ82	\$CØ83	\$CØ84	\$CØ85	\$CØ86	\$CØ87	\$CØ88	\$CØ89	\$C08A	\$CØ8B	\$C08C	\$C08D	\$C08E	\$CØ8F	
	Base	Addicss AC080	\$CØ81	\$CØ82	\$CØ83	\$CØ84	\$CØ85	\$CØ86	\$CØ87	\$CØ88	\$CØ89	\$C08A	\$C08B	\$CØ8C	\$CØ8D	\$C08E	\$C08F	

"A2M 030-0004-01 8-t25.PICT" 273 KB 2001-07-23 dpi: 600h x 600v pix: 1705h x 2941v

	Ta	ble 26: I/	O Scratcl	npad RAN	Table 26: I/O Scratchpad RAM Addresses	Se	
Base			S	Slot Number	er		
Address		2	3	4	5	9	
\$0478	\$0479	\$047A	\$047B	\$047C	\$047D	\$047E	\$047F
\$04F8	\$04F9	\$04FA	\$04FB	\$04FC	\$04FD	\$04FE	\$04FF
\$0578	80579	\$057A	\$057B	\$057C	\$057D	\$057E	\$057F
\$05F8	\$Ø5F9	\$05FA	\$05FB	<b>\$05FC</b>	\$05FD	\$05FE	\$05FF
80678	61908	\$Ø67A	\$067B	\$067C	\$Ø67D	\$067E	\$067F
\$06F8	\$Ø6F9	\$06FA	\$06FB	\$06FC	\$Ø6FD	\$06FE	\$06FF
80778	80779	\$077A	\$@77B	\$017C	\$077D	\$077E	\$077F
\$Ø7F8	\$Ø7F9	\$07FA	\$07FB	\$07FC	\$Ø7FD	\$07FE	\$07FF

"A2M 030-0004-01 8-t26.PICT" 137 KB 2001-07-23 dpi: 600h x 600v pix: 979h x 2423v

Apple 2 Technical Manual	Apple	][ Refere	ence Ma	ınual •	1979
Table 27: Timing Signal Descriptions  Master Oscillator output, 14.318 MHz. All timing signals are derived from this signal.  Intermediate timing signal, 7.159 MHz.	Color reference frequency, 3.580MHz. Used by the video generation circuitry.	Phase Ø system clock, 1.023MHz, compliment to Φ1.	Phase 1 system clock, 1.023 MHz, compliment to ΦØ.	A general-purpose timing signal, twice the frequency of the system clocks, but asymmetrical.	
14M: 7M:	COLOR REF:	ФØ (Ф2) :	Ф1:	Q3:	

Apple 2 Technical Manual	•	Apple ][ Reference Manual	•	1979
--------------------------	---	---------------------------	---	------

	Table 28: Auxil	Auxiliary Video Output Connector Signal Descriptions
Pin	Name	Description
	GROUND	System common ground; 0 volts.
7	VIDEO	NTSC compatible positive composite video. Black level is about .75 volt, white level about 2.0 volt, sync tip level is 0 volts. Output level is not adjustable. This is not protected against short circuits.
8	+12v	+12 volt power supply.
4	-5v	-5 volt line from power supply.

"A2M 030-0004-01 8-t28.PICT" 114 KB 2001-07-23 dpi: 600h x 600v pix: 1083h x 2646v

	Table 29	Table 29: Game I/O Connector Signal Descriptions	,
Pin:	Name:	Description:	Арр
1	+5v	+5 volt power supply. Total current drain on this pin must be less than 100mA.	le 2 Tech
2-4	PBØ-PB2	Single-bit (Pushbutton) inputs. These are standard 74LS series TTL inputs.	nnical Man
2	C040 STROBE	A general-purpose strobe. This line, normally high, goes low during ΦØ of a read or write cycle to any address from \$CØ4Ø through \$CØ4F. This is a standard 74LS TTL output.	ual • Appl
6,7,10,11	GCØ-GC3	Game controller inputs. These should each be connected through a 150K Ohm variable resistor to +5v.	e ][ Refere
&	Gnd	System electrical ground.	ence Ma
12-15	ANØ-AN3	Annunciator outputs. These are standard 74LS series TTL outputs and must be buffered if used to drive other than TTL inputs.	anual • 19
9,16	NC	No internal connection.	79

"A2M 030-0004-01 8-t29.PICT" 181 KB 2001-07-23 dpi: 600h x 600v pix: 1882h x 2929v

A	ople	2 Technic	cal Manual •	Apple ][	Refere	nce Ma	nual •	1979	_
Table 30: Keyboard Connector Signal Descriptions	Description:	+5 volt power supply. Total current drain on this pin must be less than 120mA.	Strobe output from keyboard. This line should be given a pulse at least $10\mu s$ long each time a key is pressed on the keyboard. The strobe can be of either polarity.	Microprocessor's RESET line. Normally high, this line should be pulled low when the RESET button is pressed.	No connection.	Seven bit ASCII keyboard data input.	System electrical ground.	-12 volt power supply. Keyboard should draw less than 50mA.	
Table	Name:	+5v	STROBE	RESET	NC	Data	Gnd	-12v	
					,16	, 10-13			

"A2M 030-0004-01 8-t30.PICT" 156 KB 2001-07-23 dpi: 600h x 600v pix: 1692h x 2781v

Apple 2	Tec	hnical Mar	nual • A	pple ][ Refe	erence Man	ual • 197	<sup>7</sup> 9
<del>- 11</del>							
Table 31: Power Connector Pin Descriptions	: Description:	nd Common electrical ground for Apple board.	$+5.0$ volts from power supply. An Apple with 48K of RAM and no peripherals draws $\sim 1.5$ amp from this supply.	+12.0 volts from power supply. An Apple with 48K of RAM and no peripherals draws ~400ma from this supply.	-12.0 volts from power supply. An Apple with 48K of RAM and no peripherals draws ~12.5ma from this supply.	-5.0 volts from power supply. An Apple with 48K of RAM and no peripherals draws ~0.0ma from this supply.	
	Name:	Ground	+5v	+12v	-12v	-5v	
	];;	- >					

"A2M 030-0004-01 8-t31.PICT" 147 KB 2001-07-23 dpi: 600h x 600v pix: 1369h x 2574v

9

2

Apple 2 Technical Man	ual • A	pple ][ Refere	ence Manual	• 19	979
	att into an 8				

		Table 32: Speaker Connector Signal Descriptions
Pin:	Name:	Description:
_	SPKR	Speaker signal. This line will deliver about .5 watt in Ohm load.
2	+5	+5 volt power supply.

"A2M 030-0004-01 8-t32.PICT" 78 KB 2001-07-23 dpi: 600h x 600v pix: 566h x 2536v

	Table 33. P.	Perinheral Connector Signal Description
Pin:	11	Description:
	I/O SELECT	This line, normally high, will become low when the microprocessor references page \$Cn, where n is the individual slot number. This signal becomes active during \$\Phi\$\$ and will drive 10 LSTTL loads*. This signal is not present on peripheral connector \$\Phi\$.
2-17	AØ-A15	The buffered address bus. The address on these lines becomes valid during Φ1 and remains valid through Φθ. These lines will each drive 5 LSTTL loads*.
18	R/₩	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 2 LSTTL loads*.
19	SYNC	On peripheral connector 7 only, this pin is connected to the video timing generator's SYNC signal.
20	I/O STROBE	This line goes low during $\Phi \theta$ when the address bus contains an address between \$C800 and \$CFFF. This line will drive 4 LSTTL loads*.
21	RDY	The 6502's RDY input. Pulling this line low during Φ1 will halt the microprocessor, with the address bus holding the address of the current location being fetched.
22	DMA	Pulling this line low disables the 6502's address bus and halts the microprocessor. This line is held high by a $3K\Omega$ resistor to $+5v$ .
23	INT OUT	Daisy-chained interrupt output to lower priority devices. This pin is usually connected to pin 28 (INT IN).
24	DMA OUT	Daisy-chained DMA output to lower priority devices. This pin is usually connected to pin 22 (DMA IN).
25	+5v	+5 volt power supply. 500mA current is available for all peripheral cards.
26	GND	System electrical ground.

"A2M 030-0004-01 8-t33.PICT" 310 KB 2001-07-23 dpi: 600h x 600v pix: 4018h x 2353v

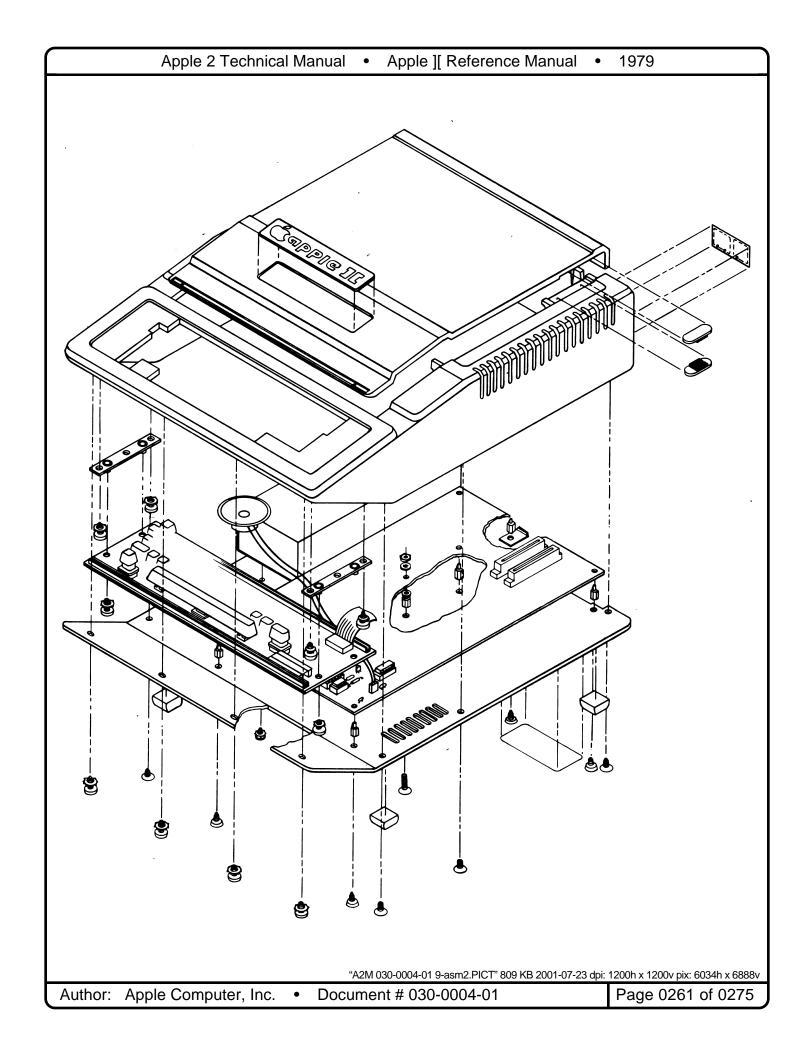


### Apple II Computer Reference Manual Information

# Apple II Case Assembly Diagram

Apple Part # 030-0004-01

"A2M 030-0004-01 9-asm1.PICT" 197 KB 2001-07-23 dpi: 600h x 600v pix: 3849h x 4429v



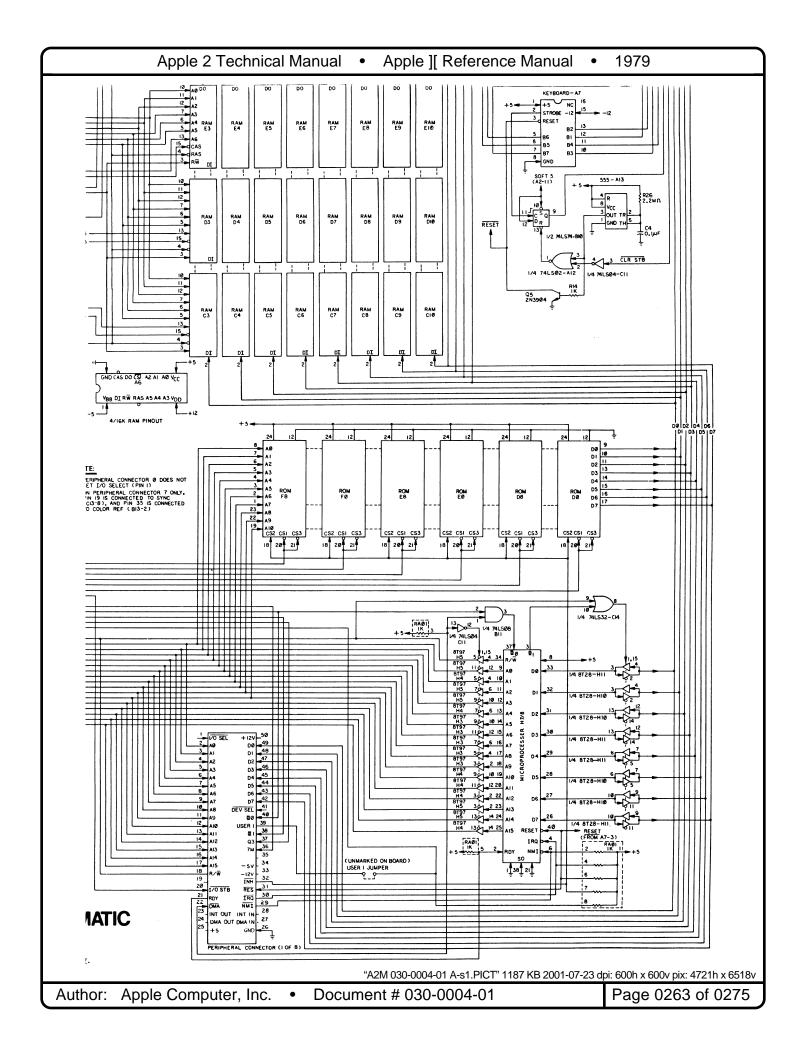


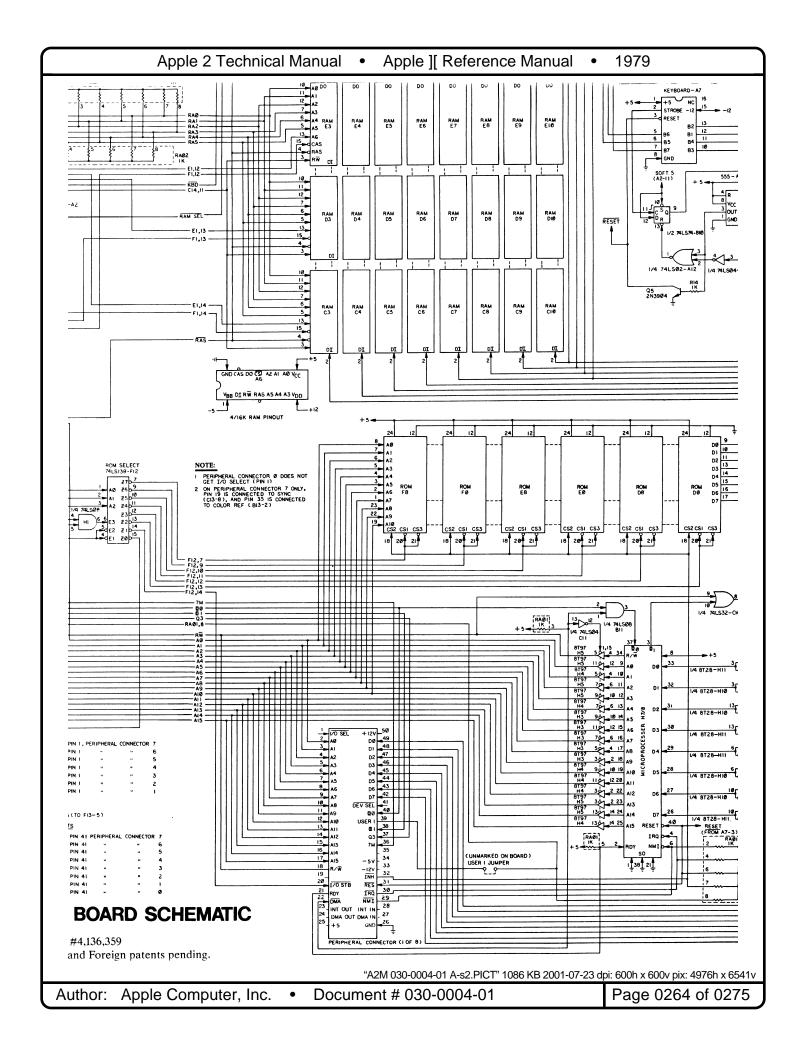
### Apple II Computer Reference Manual Information

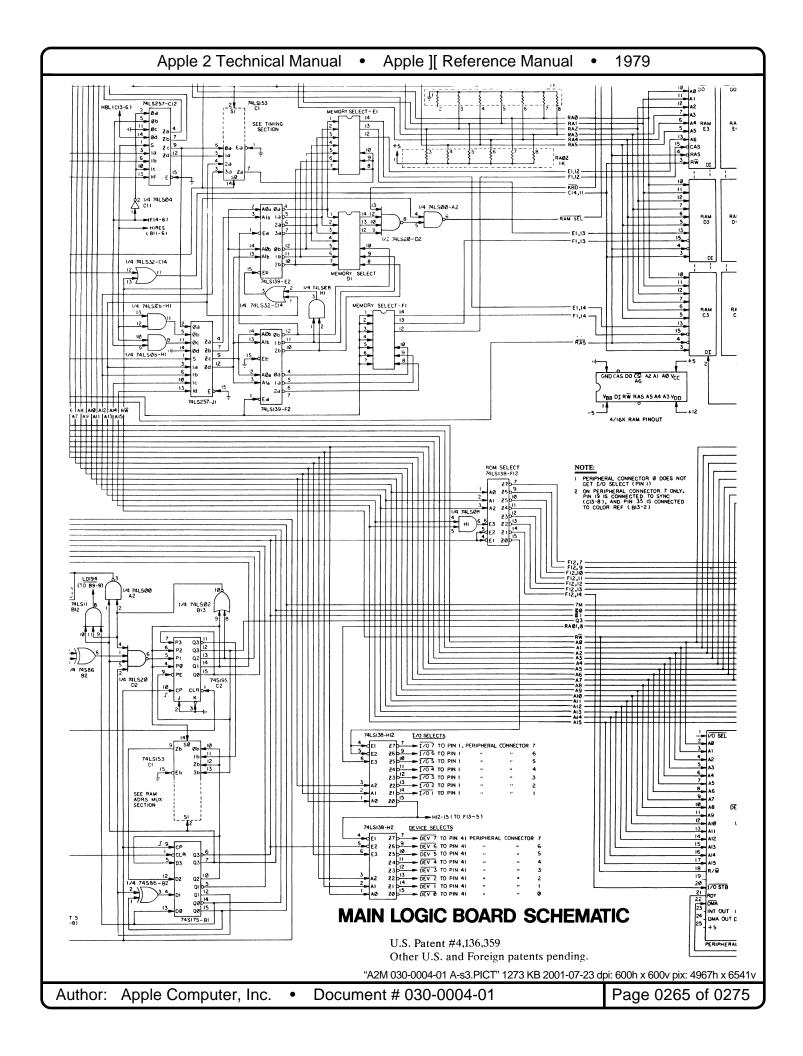
### Apple II Main Logic Board Schematic

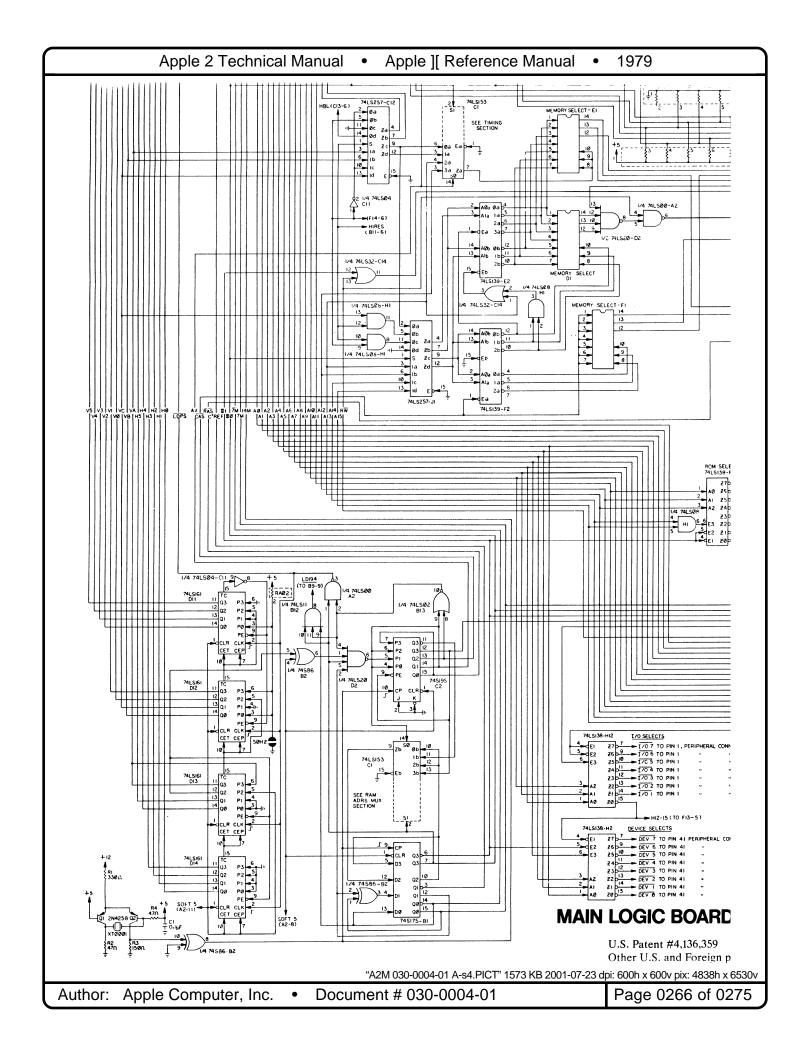
Apple Part # 030-0004-01

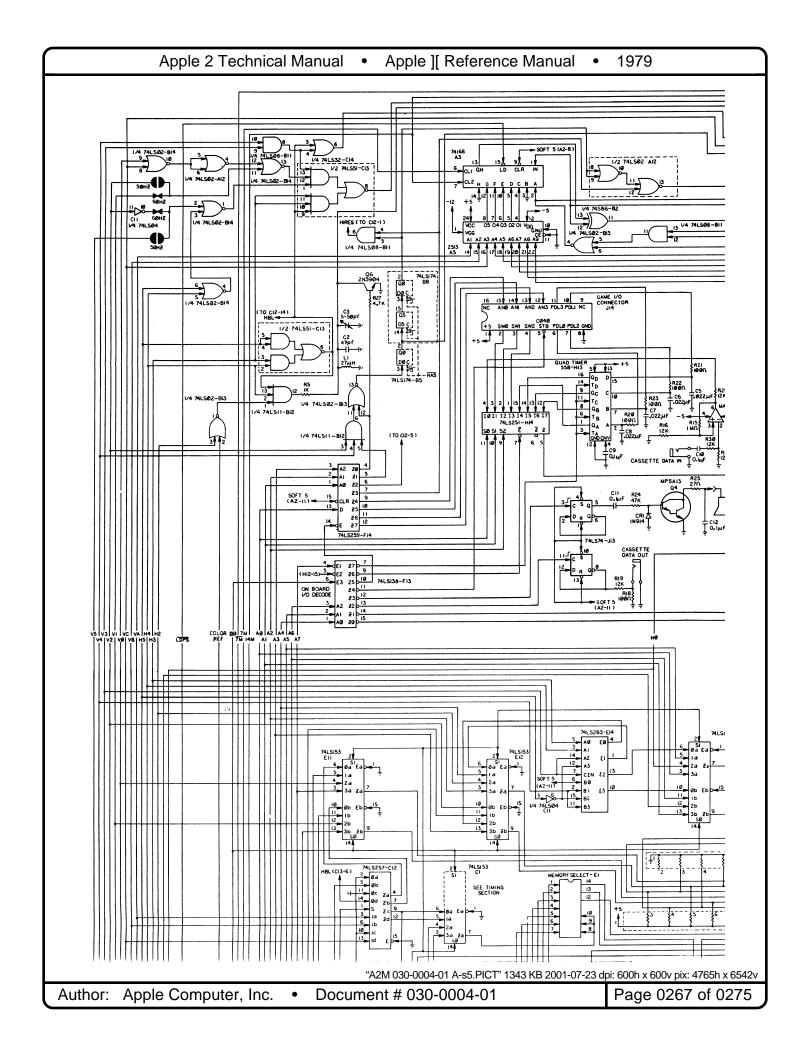
"A2M 030-0004-01 A-s0.PICT" 208 KB 2001-07-23 dpi: 600h x 600v pix: 3927h x 4406v

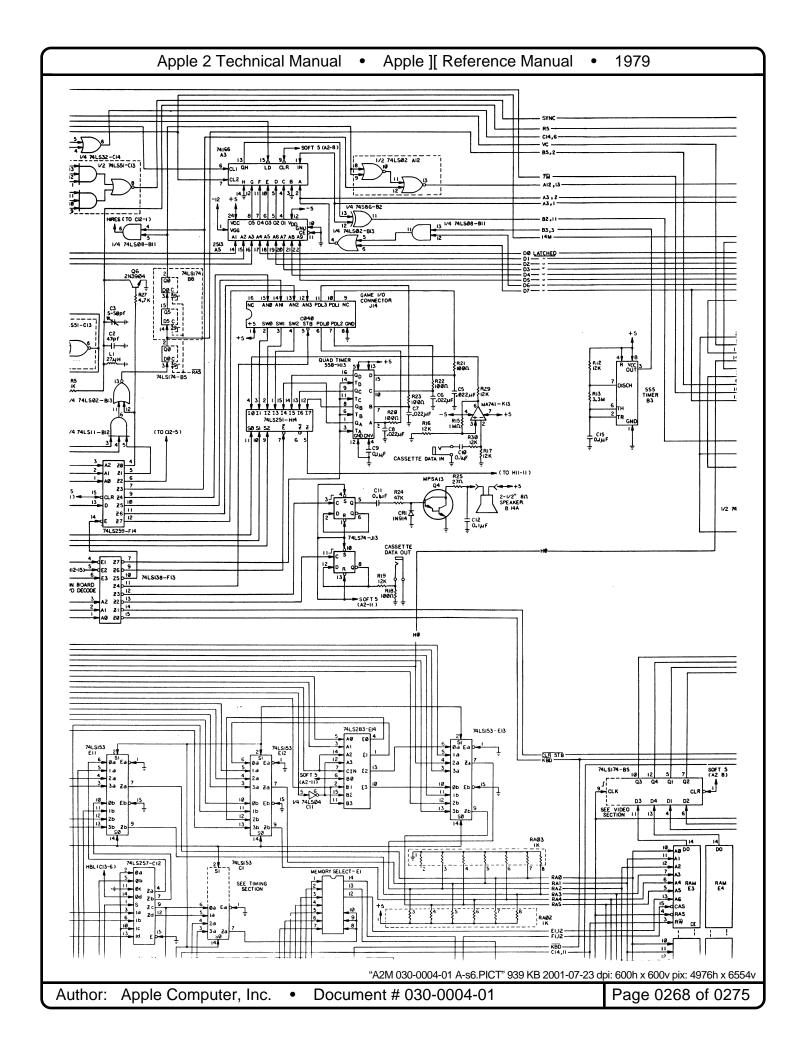


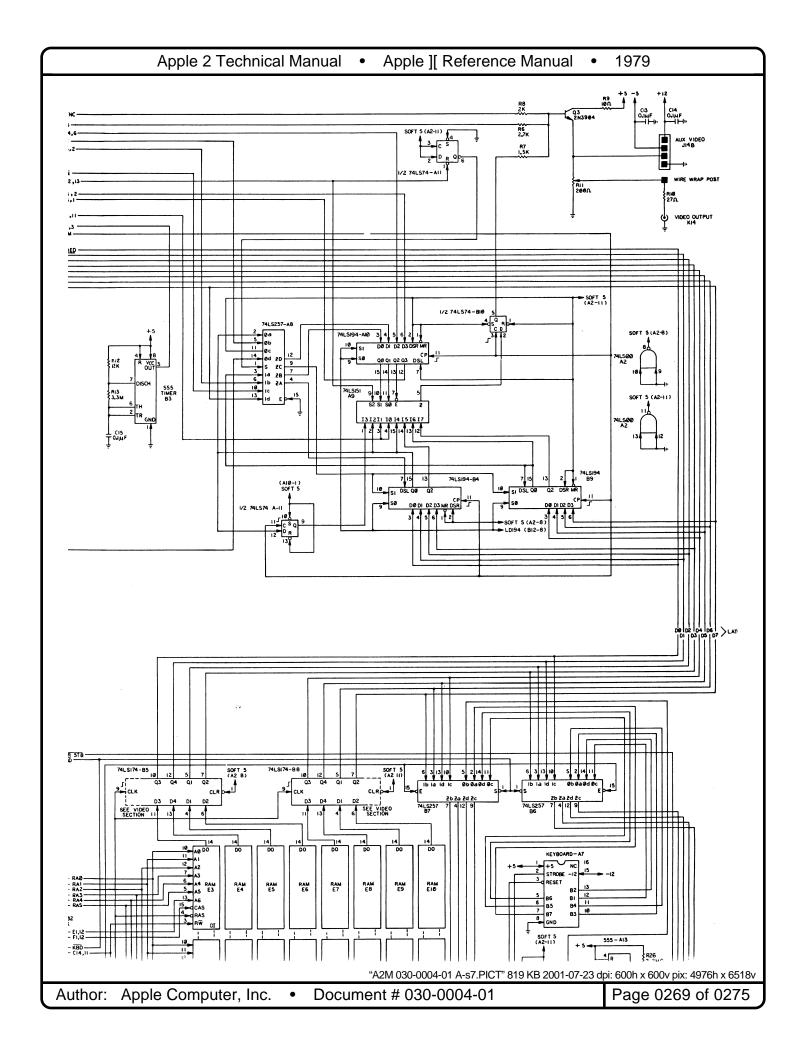














### Apple II Computer Reference Manual Information

## The Apple II Revision 07 Main Board

Apple Part # 031-0072-00

"A2M 030-0004-01 B-r70.PICT" 209 KB 2001-07-23 dpi: 600h x 600v pix: 3850h x 4428v

#### The Apple II Revision 07 Main Board

The main logic board of your Apple is a 'Revision 07' board. This means it is slightly different from the Apple boards which are described in the Apple II Reference Manual. It will not, however, behave differently in any specific way unless you have changes made to it. This Revision 07 board has greater flexibility than earlier boards.

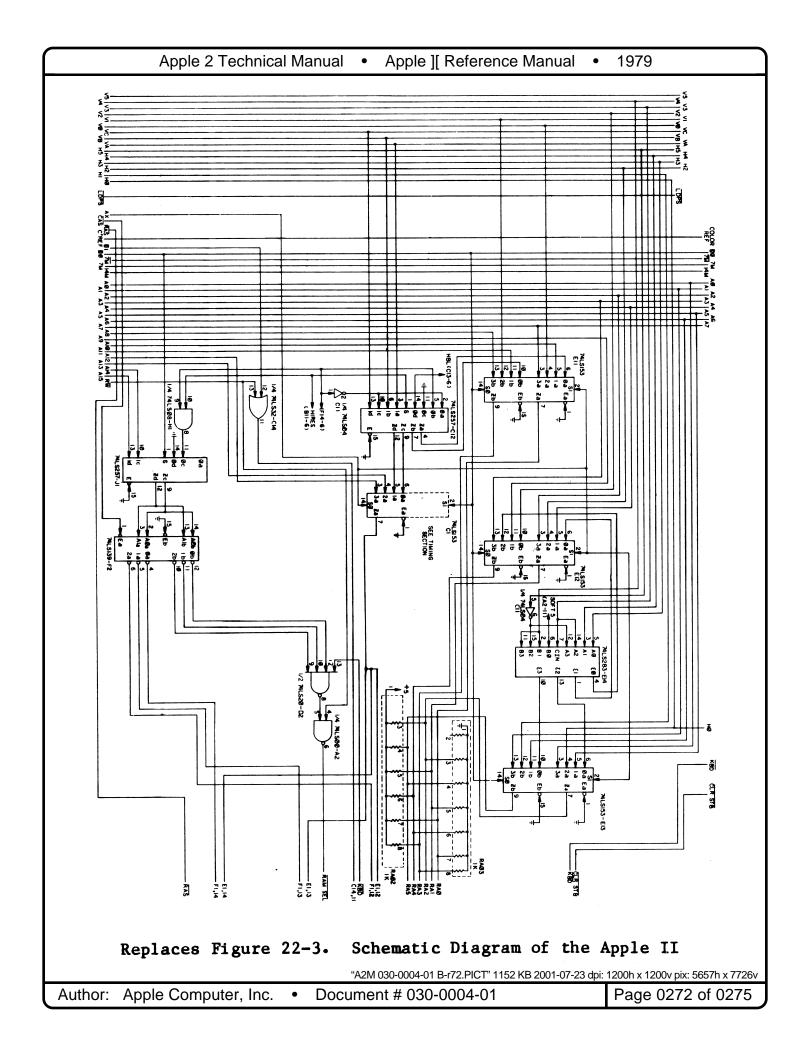
You will know you have a Revision \$7 board by looking at the white F on the far left side of the board. You'll see there a nine digit number which ends in '-07'.

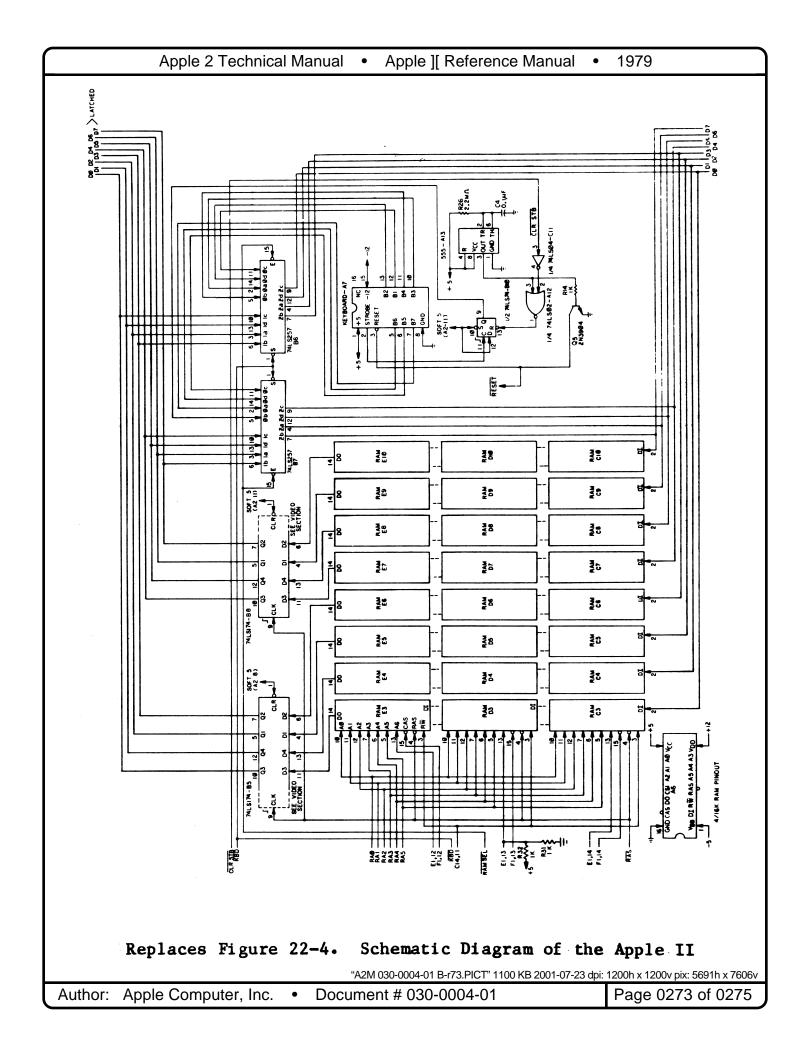
The major differences in the Revision Ø7 board are described below. Also, the attached schematics show the areas in which the Revision Ø7 board is different from earlier boards. You may wish to note these differences in your Apple II Reference manual, on the pages which correspond to the schematics here.

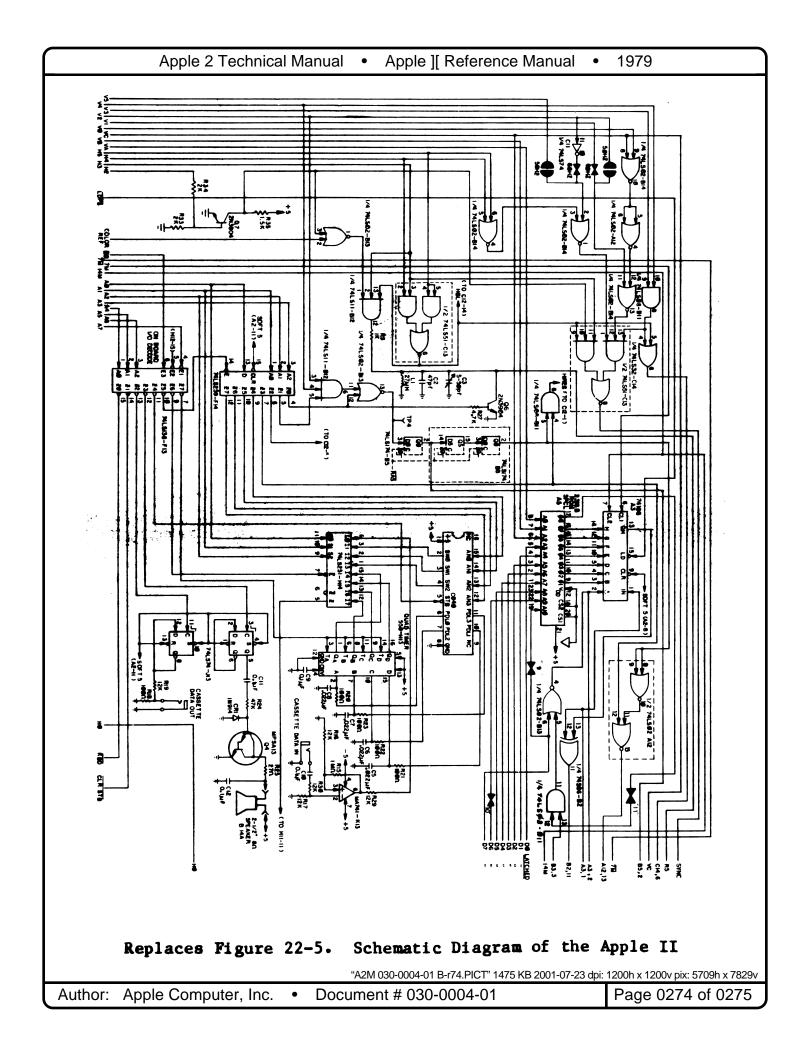
- \* The Revision Ø7 board does not have RAM configuration blocks. This means the RAM Integrated Circuits (ICs) which give your Apple its memory MUST be 16K byte ICs. All of the RAM ICs in your Apple are within the white-outlined box on the board. If you add memory to your Apple, make sure all the ICs you add in this box are 16K ICs.
- \* The IC which controlled the configuration blocks in the older versions of the Apple II board is no longer there. It was formerly in the E2 position on the Apple board (in the row labeled E, the second IC from the left of the board) and was marked 74LS139.
- \* The Revision \$7 board has a different character generator ROM IC. The character generator ROM IC determines what style of lettering, or character set, you'll see on your monitor or terminal screen. The new character generator ROM is found next to the Keyboard socket on the main board. This 2316B ROM has much more ROM (Read Only Memory) space than the former 2513 character generator ROM, so that it's possible to have more than one character set available. The 2316B ROM can also be replaced with a 2716 EPROM, which allows you to program and change your character sets.
- \* An inverter circuit has been added in the H2 line, which is a video synchronization signal. This alters the video synchronization pulse rate of the Apple so that is more compatible with modern TVs that have digital synchronization circuitry.
- \* Two lK resistors have been added on address line A6 to reduce noise in that line of RAM.

APPLE Part #031-0072-00

"A2M 030-0004-01 B-r71.PICT" 1112 KB 2001-07-23 dpi: 1200h x 1200v pix: 5743h x 8313v







Apple 2 Technical Manual • Apple ][ Reference Manual • 1979



"A2M 030-0004-01 Z-back.PICT" 3905 KB 2001-07-23 dpi: 1200h x 1200v pix: 4509h x 993v

Author: Apple Computer, Inc. • Document # 030-0004-01 Page 0275 of 0275