Notice

Apple Computer Inc. reserves the right to make improvements in the product described in this manual at any time and without notice.

Disclaimer of All Warranties And Liability

Apple Computer Inc. makes no warranties, either express or implied, with respect to this manual or with respect to the software described in this manual, its quality, performance, merchantability, or fitness for any particular purpose. Apple Computer Inc. software is sold or licensed “as is.” The entire risk as to its quality and performance is with the buyer. Should the programs prove defective following their purchase, the buyer (and not Apple Computer Inc., its distributor, or its retailer) assumes the entire cost of all necessary servicing, repair, or correction and any incidental or consequential damages. In no event will Apple Computer Inc. be liable for direct, indirect, incidental, or consequential damages resulting from any defect in the software, even if Apple Computer Inc. has been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you.

This manual is copyrighted. 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.

© 1981 by Apple Computer Inc.
10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010

The word Apple and the Apple logo are registered trademarks of Apple Computer Inc.

Reorder Apple Product #A2L0044

WARNING: This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.
Please read this manual before attempting to install the Super Serial Card in the Apple Computer. Incorrect installation could cause permanent damage to both the Super Serial Card and the Apple.
RADIO AND TELEVISION INTERFERENCE

The equipment described in this manual generates and uses radio frequency energy. If it is not installed and used properly, that is in strict accordance with our instructions, it may cause interference to radio and television reception.

This equipment has been tested and complies with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC rules. These rules are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that the interference will not occur in a particular installation.

You can determine whether your computer is causing interference by turning it off. If the interference stops, it was probably caused by the computer. If your computer does cause interference to radio or television reception, you can try to correct the interference by using one or more of the following measures:

- Turn the TV or radio antenna until the interference stops.
- Move the computer to one side or the other of the TV or radio.
- Move the computer farther away from the TV or radio.
- Plug the computer into an outlet that is on a different circuit from the TV or radio. (That is, make certain the computer and the TV or radio are on circuits controlled by different circuit breakers or fuses.)

If necessary, you should consult your dealer or an experienced radio/television technician for additional suggestions. You may find the following booklet prepared by the Federal Communications Commission helpful:

"How to Identify and Resolve Radio-TV Interference Problems"

This booklet is available from the U.S. Government Printing Office, Washington, DC 20402, Stock number 004-000-00345-4.

Downloaded from www.Apple2Online.com
TABLE OF CONTENTS

PREFACE

Chapter 1
GETTING STARTED
1 Unpacking
2 A Close Look
2 Preparing Cable and Clamp Assembly
3 Attaching Internal Cable to SSC

Chapter 2
PRINTER MODE
5 Preparing the SSC for Printer Mode
6 Setting the Switches
6 Commonly Used Settings
6 Baud Rate
7 Data Format and Parity
7 Carriage Return Delay
8 Line Width and Video On/Off
8 Generate <LF> Out
8 Special Switches
9 Installation Procedure
10 External Cable and Connector
11 Using the SSC in Printer Mode
11 With a Printer
11 With a Terminal
11 Printer Mode Commands
12 Command Formats
12 The Command Character
13 Printer Mode Command Summary
15 Commands That Change Switch Settings
15 Baud Rate--<n>B
15 Data Format--<n>D
15 Parity--<n>P
16 Set Time Delay--<n>C, <n>L, <n>F
17 Generate <CR> On Column Overflow--C
17 Generate <LF> Out--L <E/D>
17 Mask (Suppress) <LF> In--M <E/D>
17 Reset the SSC--R
Other Commands

Translate Lowercase Characters--<n>T
Zap (Suppress) Control Characters--Z
Find Keyboard--<n>E/D
XOFF Recognition--<n>E/D
Tab in BASIC--<n>E/D

Chapter 3
COMMUNICATIONS MODE

Preparing the SSC for Communications Mode
Setting the Switches
Commonly Used Settings
Baud Rate
Data Format and Parity
Generate <LF> Out
Special Switches
Installation Procedure
External Cable and Connector
Using the SSC in Communications Mode
Communications Mode Commands
Command Formats
The Command Character
Communications Mode Command Summary
Commands That Change Switch Settings
Baud Rate--<n>B
Data Format--<n>D
Parity--<n>P
Generate <LF> Out--<n>L<E/D>
Mask (Suppress) <LF> In--<n>M<E/D>
Reset the SSC--R
Other Commands
Set Time Delays--<n>C, <n>L, <n>F
Translate Lowercase Characters--<n>T
Zap (Suppress) Control Characters--Z
Find Keyboard--<n>E/D
XOFF Recognition--<n>X<E/D>
Specify Screen Slot--<n>S
Echo Characters on the Screen--<n>E<E/D>

Terminal Mode

Terminal Mode Commands
Enter Terminal Mode--T
Transmit a Break Signal--B
Special Characters--<n>E/D
Quit (Exit from) Terminal Mode--Q
A Terminal Mode Example
Chapter 4

HOW THE SSC WORKS

37 Serial Data Communication
38 Parallel Data in the Apple II
39 Serial Data for Long Distances
39 Data Conversion
40 RS-232-C Data Formats
40 RS-232-C Signals
41 Modems
42 Modem Eliminators
43 SSC Modes and Configurations
45 Theory of Operation
46 Addressing and Control Logic
46 ROM/RAM Space
47 Registers in Peripheral I/O Space
47 The ACIA
48 Data Input and Output
48 Data Bus
48 Jumper Block

Appendix A

FIRMWARE

49 Pascal 1.1 Firmware Protocol
50 I/O Routine Entry Points
51 Device Identification
52 SSC Firmware Memory Usage
52 Zero Page Locations
53 Scratchpad RAM Locations
54 Peripheral Card I/O Space
55 SSC Entry Points
55 Monitor ROM Entry Points
55 BASIC Entry Points
56 Pascal 1.0 Entry Points
56 Pascal 1.1 Entry Points
57 Other Special Firmware Locations
58 SSC Firmware Listings

Appendix B

APPLE INTERFACE CARD EMULATION

91 Old Serial Interface Card Emulation
92 P8 Emulation POKEs
94 P8A Emulation POKEs
95 Other Emulation Mode Differences
Old Communications Card Commands
- Switch to Terminal Mode—<CTRL-T>
- Bypass Terminal Mode—<CTRL-R>
- XOFF—<CTRL-S>
- Parallel Card Commands
  - Line Width n and Video Off—<CTRL-I><n>N
  - Line Width 40 and Video On—<CTRL-I>I
  - Disable Automatic Linefeed—<CTRL-I>K

Appendix C
SPECIFICATIONS AND SCHEMATICS

- SSC Specifications
- Connector Pin Assignments
- Jumper Block Wiring
- Schematic Diagram

Appendix D
ASCII CODE TABLE

Appendix E
TROUBLESHOOTING HINTS

Appendix F
ERROR CODES

GLOSSARY

FIGURES AND TABLES

INDEX
The Super Serial Card (SSC) provides a two-way serial interface to a wide variety of devices, including printers, terminals, plotters, and other computers. All these devices can be connected to the SSC either directly or via modem.

The SSC replaces both the P8 and P8A variety of Apple II Serial Interface Card, although it does not manipulate all specific Apple II memory locations in the same way. The SSC also replaces the Apple II Communications Card, and supports Terminal Mode. Finally, the SSC supports Apple II parallel interface card software commands.

The Super Serial Card conforms to the Electronic Industries Association (EIA) interface definitions A through E. (To obtain a copy of the EIA RS-232-C Standard, write to the EIA Engineering Department, Electronics Industries Association, 2001 Eye Street, N.W., Washington, D.C. 20006.)

The SSC can be configured to the attached external device in three ways: (1) by setting switches on the card itself, (2) by typing in commands at the keyboard under the Monitor, Integer BASIC, Applesoft or DOS, or (3) by issuing commands from assembly language, BASIC or Pascal programs. The SSC can be configured and operated by programs in Integer BASIC, APPLESOFT, Pascal, and assembly language.

How you prepare, install and use the Super Serial Card depends on what you connect to it:

- Read Chapter 1 for unpacking and cable clamp preparation instructions.

- If you are going to connect a printer, terminal or some other device directly to the SSC, then read the first four sections of Chapter 2. (Many commonly used switch settings are listed in Table 2-1 for your convenience.) You only need to read the section Printer Mode Commands of Chapter 2 if you need special commands to change the SSC's characteristics.

- If you are going to connect a device to the SSC via a modem or similar communications equipment, then read the first four sections of Chapter 3. (Switch settings for many Communications Mode applications are listed in Table 3-1.) You only need to read the section Communications Mode Commands of Chapter 3 if you need special commands to change the SSC's characteristics.

- If you want to use the Apple II as an unintelligent terminal connected via a modem, read the section Terminal Mode of Chapter 3.

- Troubleshooting Hints are discussed in Appendix E.
The SSC also emulates ("imitates") the Apple II Serial Interface Card (both the P8 and P8A varieties), and supports many of the software commands used by the Apple II parallel printer interface card and the Apple II Communications Card. These are all discussed in Appendix B.

Chapter 4 explains how the SSC works, both in everyday terms (Serial Data Communication Simply Explained) and from an engineering viewpoint (Theory of Operation). The Theory of Operation section is keyed to the schematic diagram in Appendix C. Chapter 4 also contains a section on SSC modes and configurations.

Appendix A discusses SSC firmware and its entry points in the SSC ROM, as well as the Apple II memory locations the firmware uses.

Appendix C contains SSC specifications and connector pin assignments, and its schematic diagram.

Appendix D lists the ASCII codes and their equivalents. Appendix E has troubleshooting hints. Appendix F explains the SSC error codes.

A glossary explains the meaning of most important terms as they apply to the SSC.

The Reference Card summarizes the switch settings and commands for the SSC Printer Mode and Communications Mode.

There are three symbols that set off information of special importance:

- This symbol points to a paragraph that contains especially useful information.

- Watch out! This symbol precedes a paragraph that warns you to be careful.

- This symbol precedes a warning that you are about to harm hardware or destroy data.
CHAPTER 1
GETTING STARTED

This chapter takes you through the first steps of getting acquainted with your Super Serial Card (SSC). After unpacking the SSC and examining it, you will assemble the short internal cable (if it is not already assembled) that connects the 16-pin cable socket on the SSC to the 25-pin socket at the back of the Apple II case.

UNPACKING

As you unpack your Super Serial Card (Figure 1-1), check the contents against the items described on the packing list.

Fill out the pre-addressed warranty card and mail it in. If any items are missing, contact the dealer you purchased the SSC from.

You will need a shielded external cable (not provided as part of the SSC package) to connect the external device—the printer, modem, terminal, or other computer—to your Apple II. Suitable cables are available through your Apple dealer.

Figure 1-1. Photo of the Super Serial Card
A CLOSE LOOK

Let's examine the Super Serial Card for a moment. Pick up the SSC carefully and put it on a flat surface oriented as shown in Figure 1-1. Now use Figure 1-2 to help identify the chief parts of the SSC. Those that you will have to deal with as you prepare it for installation are:

- The **jumper block.** This ordinarily points toward the word TERMINAL; if you attach a modem to the SSC, you will turn this around so the arrow points toward the word MODEM (Chapter 3).

- The **switches.** The left group is numbered from SW1-1 through SW1-7; the right group is numbered from SW2-1 through SW2-7. You can see the characters "SW1" and "SW2" printed on the SSC.

- The **edge connector.** It is important not to touch the gold fingers on this connector: they must make a clean electrical contact in the Apple II connector slot when you install the SSC (Chapter 2 or Chapter 3).

- The **cable socket.** The next section of this chapter explains how to install the short internal cable between the SSC and the Apple II case.

![Figure 1-2. Line Drawing of the SSC](image)

PREPARING CABLE AND CLAMP ASSEMBLY

Before preparing and installing the SSC, you may need to prepare the clamp assembly for the internal cable that will go from the SSC to the back of the Apple II's case. The components of this clamp assembly are shown in Figure 1-3. If these components are already assembled, skip to the next section, Attaching the Internal Cable to the SSC.
Figure 1-3. Components of Internal Cable and Clamp Assembly

Lay the short cable down as shown in Figure 1-3. Pick up the clamp piece that has the word TOP stamped on one end. Hold this clamp piece with the word TOP facing away from you, and the oval cutout toward the smaller connector on the cable. Bend the cable slightly, and insert it into the oval cutout through the opening; then straighten the cable in the cutout so that it moves easily.

The other clamp piece has flanges (Figure 1-3) and a rectangular opening that is closer to one end (its top end) than to the other. Hold this clamp piece with its top end away from you and its flanges facing the 25-pin connector end of the cable. Then tilt the connector and feed it completely through the rectangular cutout.

Now slide the two clamp pieces all the way down the cable until they are right up against the 25-pin connector, and their screw holes line up with the connector's screw holes. Slide the washers onto the screws and then thread the screws a couple of turns into the lined-up holes. Don't screw them in very far.

ATTACHING INTERNAL CABLE TO SCC

This step in the preparation of your Super Serial Card is simple to do, but you must do it carefully.

⚠️ It is very important to connect the cable to the SSC correctly. Improper connection of the cable to the SSC may result in damage to the Apple and the SSC; such damage is NOT covered by your warranty.

Lay the SSC down on a flat surface, component-side up and gold fingers at the lower right. Examine the 10-pin end of the cable: the wires come out of the SIDE of the connector—-the same side as the raised "key" in the plastic (Figure 1-3). Hold the connector so
the wires are on the side away from the SSC, and insert the connector firmly into the cable socket along the right edge of the SSC. The raised "key" should slide into the groove in the cable socket (Figure 1-4).

⚠️ If the cable is now jammed between the 19-pin cable socket and the SSC board, the connector is plugged in backwards. Unplug the connector and reconnect it so that the cable is on the side AWAY from the SSC (Figure 1-5).

Figure 1-4. Sliding the "Key" into the Groove

Figure 1-5. Internal Cable Attached Correctly to SSC
CHAPTER 2
PRINTER MODE

This chapter explains how to prepare, install and use the SSC in Printer Mode, and change the SSC's activities via commands.

PREPARING THE SSC FOR PRINTER MODE

The SSC is ready to operate in Printer Mode when the jumper block and switches SW1-5 and SW1-6 are correctly positioned (Figure 2-1).

If the triangle on the jumper block is pointing down toward the word MODEM, remove the block (using an IC Extractor, if necessary) and carefully reinsert it so the triangle is pointing toward TERMINAL.

Using a pointed object, set switch SW1-5 OFF and switch SW1-6 ON as shown in Figure 2-1.

Figure 2-1. SSC Set for Printer Mode

When the jumper block is pointing toward TERMINAL, it is acting as a Modem Eliminator. Therefore, DO NOT connect a separate Modem Eliminator, or it will cancel the effect of the jumper block, and the attached device will not work.
SETTING THE SWITCHES

Use a pointed object, such as the tip of a ballpoint pen, to flip the appropriate tiny switches on the SSC. A switch is ON when the top of the switch rocker is pushed in, and OFF when the bottom is in. The following subsections explain what settings to use.

COMMONLY USED SETTINGS

Table 2-1 lists the switch settings you can use for direct connection, via the SSC, of some commonly used printers. Most printers can use any one of several setups.

<table>
<thead>
<tr>
<th>Printer</th>
<th>Switch Settings, Cable Connections, Other Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDS 560</td>
<td>SW1: OFF OFF OFF ON OFF ON SW2: ON ON ** OFF OFF ON</td>
</tr>
<tr>
<td>Paper Tiger</td>
<td>Printer Mode, HW Hndshk, 9600 baud, 1 stop bit, ** width</td>
</tr>
<tr>
<td></td>
<td>IDS SW1: - - - ON ON OFF OFF SW2: - - - OFF - -</td>
</tr>
<tr>
<td></td>
<td>SSC/IDS pins: 3/3, 7/7, 20/20; all IDS jumpers removed</td>
</tr>
<tr>
<td>NEC 5510</td>
<td>SW1: OFF ON ON OFF OFF OFF SW2: ON ON ** OFF OFF ON</td>
</tr>
<tr>
<td>Spinwriter</td>
<td>Printer Mode, ETR/ACK, 1200 baud, 1 stop bit, ** line width</td>
</tr>
<tr>
<td></td>
<td>NEC switches: OFF ON OFF OFF OFF OFF ON ON</td>
</tr>
<tr>
<td></td>
<td>SSC/NEC pins: 2/2, 3/3, 7/7, 20/68; 4&amp;5 tied on NEC end</td>
</tr>
<tr>
<td></td>
<td>May need keystroke to force first ETX after power-up.</td>
</tr>
<tr>
<td>NEC 5510</td>
<td>SW1: OFF ON ON OFF ON OFF SW2: ON ON ** OFF OFF ON</td>
</tr>
<tr>
<td>Spinwriter</td>
<td>Printer Mode, hardware handshake, rest same as above</td>
</tr>
<tr>
<td></td>
<td>NEC switches: OFF ON OFF OFF OFF ON ON</td>
</tr>
<tr>
<td></td>
<td>SSC/NEC pins: 3/3, 6/68, 7/7, 20/20; 4&amp;5 NOT tied</td>
</tr>
<tr>
<td>Qume Sprint 5</td>
<td>SW1: OFF ON ON OFF OFF ON SW2: ON OFF ** OFF OFF OFF</td>
</tr>
<tr>
<td>Sprint 5</td>
<td>Printer Mode, HW Hndshk, 1200 baud, 1 stop bit, ** width</td>
</tr>
<tr>
<td>Qume Sprint 9/35</td>
<td>Qume switches: 1200 baud, no modem; pins: 3, 4, 7, 20</td>
</tr>
<tr>
<td>Sprint 9/35</td>
<td>Qume asserts RTS and DTR only when ready to receive data</td>
</tr>
</tbody>
</table>

Table 2-1. Commonly Used Switch Settings for Printer Mode

BAUD RATE

No matter what type of printer or terminal you connect to the SSC, the SSC is going to pass information between the Apple II and the device at a certain prearranged speed, called the baud rate. Since the Apple II can usually send and receive information faster than what is connected to it, the simplest way to determine the baud rate is to consult the user manual for the device you will connect. Find out what rate is the fastest the device can handle (up to 19,200 baud). Once you know this, you are ready to set the baud rate switches on the SSC.
<table>
<thead>
<tr>
<th>Baud</th>
<th>SW1-1</th>
<th>SW1-2</th>
<th>SW1-3</th>
<th>SW1-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>75</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>110*</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>135**</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>150</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>300</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>600</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>(* 109.92)</td>
<td>(** 134.58)</td>
<td>19200</td>
<td>OFF</td>
<td>OFF</td>
</tr>
</tbody>
</table>

Table 2-2. Baud Rate Switch Settings

Make sure the printer or terminal you connect is set (with its own switches, dials or thumb wheels) to the SAME baud rate! If you don’t, the SSC will send and receive unrecognizable garbage.

DATA FORMAT AND PARITY
The SSC sends each character (such as a "3" or an "F" or a Carriage Return) as a string of zeroes and ones (bits). The way it can send a character in Printer Mode, using switch settings, is this:

- first a single **start bit** to signal to the printer or terminal that a character is coming;
- then a string of 8 **data bits** representing the character;
- no **error-checking parity bit**;
- one or two **stop bits** to signal the end of a character.

For Printer Mode, the only aspect of the data format you can change with switch settings is whether to send one stop bit or two. If you set the baud rate switches to 50, 75 or 110 baud, set switch SW2-1 OFF (two stop bits). For all other baud rates, set switch SW2-1 ON (one stop bit) unless the documentation for the device you are connecting specifies otherwise.

The SSC does not send or check parity bits in Printer Mode unless you select some parity using the <n>P command, explained later in this chapter.

CARRIAGE RETURN DELAY
If you connect a slow printer to the SSC, and it has no handshaking capability, you may need to set switch SW2-2 ON to cause the Apple II to wait 1/4 second after a Carriage Return (<CR>). This gives
the print head assembly time to reposition to the beginning of the next line. Otherwise, set switch SW2-2 OFF (no delay).

Additional delay values (32 ms and 2 s) are available via the \(<n>C\) command described later in this chapter.

**LINE WIDTH AND VIDEO ON/OFF**

Switches SW2-3 and SW2-4 determine the printer or terminal line width and also turn the Apple II video screen on or off.

If you are connecting a printer to the SSC, select the appropriate switch settings for the number of characters the printer can fit on a line. If you set the line width to \(40\) char/line, the Apple II video screen is turned on, since it too can display \(40\) characters per line, and so can display an exact replica of what is being printed.

If you plan to connect a terminal to the SSC, set the switches for the number of characters the terminal screen can display on a line—usually 72 or 80. For these line widths, the Apple II video screen is off.

<table>
<thead>
<tr>
<th>Line Width</th>
<th>Video Screen</th>
<th>SW2-3</th>
<th>SW2-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 char/line</td>
<td>on</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>72 char/line</td>
<td>off</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>80 char/line</td>
<td>off</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>132 char/line</td>
<td>off</td>
<td>OFF</td>
<td>OFF</td>
</tr>
</tbody>
</table>

Table 2-3. Line Width and Video Switch Settings

The switch settings that turn off the Apple II video screen take effect only after PRE under BASIC or DOS. \(<\text{CTRL}-I>\) commands are still recognized, and cause the message APPLE SSC: to appear on the Apple II video screen.

**GENERATE \(<\text{LF}>\) OUT**

If you are connecting a printer to the SSC, check the printer’s user manual to see if it automatically generates a linefeed \(<\text{LF}>\) after a carriage return \(<\text{CR}>\). If it does not, set switch SW2-5 ON.

If your printer does automatically generate a linefeed after a carriage return, or if you are connecting some other device that does not need automatic linefeed generation, set switch SW2-5 OFF.

**SPECIAL SWITCHES**

Switch SW2-6 controls forwarding of interrupts to the Apple II. Since the Apple II and III do not handle interrupts, set SW2-6 OFF.
Normally, switch SW1-7 is ON and switch SW2-7 is OFF. In the rare cases where the device uses pin 19, Secondary Clear To Send, in place of pin 4 or 20, Clear To Send, set SW1-7 OFF and SW2-7 ON.

Your Super Serial Card is now ready to install and use in Printer Mode.

**INSTALLATION PROCEDURE**

This section explains how to install the SSC and its internal cable in the Apple II. If the cable clamp is not already assembled, do so now, following the instructions given in Chapter 1.

- Before connecting or disconnecting anything on the Apple, turn off the power with the switch at the back left corner of the Apple case. THIS IS ABSOLUTELY NECESSARY. If you try to connect or disconnect anything from the inside of your Apple when the power is on, you are likely to damage the circuits.

Do not unplug the Apple, just turn it off. If you unplug the Apple, you will isolate it from earth ground and leave it vulnerable to static discharges.

Remove the Apple cover by pulling up on the two back corners of the cover until the two corner fasteners pop apart. Slide the cover back until it is free of the case and then lift the cover off.

Look inside the Apple and locate the power supply case—the rectangular metal box along the left inside the Apple II. To avoid damaging the SSC, touch the power supply case with one hand; this discharges any static charge that may be on your clothes or body.

Along the back inside edge of the Apple you will see eight long narrow slots called connector slots. The connector slots are numbered from 0 at the left to 7 at the right. The numbers are printed along the back edge behind the connector slots. For use with Pascal, install the SSC in slot #1 for a printer, or slot #3 for a terminal. For use with BASIC, install the SSC in any slot from #1 through #7.

- Handle the Super Serial Card as you would handle an expensive phonograph record. Grasp it only by the corners or edges, and do not touch the components or pins, especially the gold fingers on the edge connector.

There are three deep notches along the back of the Apple II case. Temporarily set the SSC down near the desired slot. Then take the clamp assembly and slide it down into the notch closest to the slot that the SSC will be in. Tighten the screws until the connector assembly can no longer be moved in the opening.
Grasp the upper corners of the SSC and insert the gold fingers of the edge connector into the slot in the back of the Apple, rear edge first. Gently push the front edge of the card down until it is level and firmly seated.

Note that the outer ends of the screws in the clamp assembly can act as nuts. They are threaded and can receive screws from the printer or terminal connector, to ensure a good connection with the Apple.

Slide the Apple case top plate in place and press down on the rear corners until the corner fasteners pop into place. The Super Serial Card is now installed.

**EXTERNAL CABLE AND CONNECTOR**

The SSC cable connector you installed in the notch is a standard DB-25 connector with 25 pins. Ten pins of the connector are connected internally to the SSC. Connector pin assignments are listed in Appendix C.

You will need a cable to connect your external device to the SSC connector on the Apple II. Shielded cables with 25-pin connectors on one end are available from your Apple dealer.

The cable must have internal shielding, with the shielding properly terminated at both ends, to prevent electromagnetic interference to nearby radios, television sets, and communication equipment. This shielding is necessary for the system to comply with Class B Federal Communications Commission limits as defined by Subpart J of Part 15 of the FCC rules. Unshielded cables are not recommended.

Make sure that all devices are connected to the same grounded AC power circuit (three-wire wall outlet) as the Apple II. Connecting ungrounded equipment to your Apple II can cause severe electrical damage.
USING THE SSC IN PRINTER MODE

Printer Mode allows you to use the SSC with a local (that is, directly connected) printer or terminal, as well as other local serial devices. After installing the SSC, you can control its operation from a BASIC, Pascal or assembly-language program, or even directly from the keyboard. The two parts of this section explain the easiest way to get the SSC up and running from the keyboard with a printer or terminal.

WITH A PRINTER
To use the SSC with a printer, do the following:

- Make sure the jumper block points toward TERMINAL.
- Under BASIC or DOS, boot the Apple II and then type in PR#s to send output to the printer (with the SSC in slot #1).
- Under Pascal, boot the Apple II and then use the F(iler T(ransfer command to send output data to #6: or PRINTER: (with the SSC in slot #1).
- If the printer doesn’t work, refer to Appendix E for troubleshooting hints, or consult your Apple dealer.

WITH A TERMINAL
To use the SSC with a terminal, do the following:

- Make sure the jumper block points toward TERMINAL.
- Under BASIC or DOS, boot the Apple II and then type in PR#s and IN#s to route both input and output through the terminal (with the SSC in slot #3).
- Under Pascal, boot the Apple II and then use the terminal as the input/output console (with the SSC in slot #3).
- If the terminal doesn’t work, refer to Appendix E for troubleshooting hints, or consult your Apple dealer.

PRINTER MODE COMMANDS

You can issue any of the commands described in this section by embedding them in a computer program. Under BASIC, DOS or the Apple Monitor, you can also enter them directly at the Apple (or terminal) keyboard.
In a BASIC program, put the control character and command in a PRINT statement. In a Pascal program, issue the command in a WRITE or WRITELN statement.

When you enter the command character (usually \(<\text{CTRL-I}\>\); see below), the prompting message \(\text{APPLE SSC:}\) appears on the display screen. Subsequent characters, up to \(<\text{RETURN}\>\), will be interpreted as an SSC command. Pressing the left arrow key before pressing \(<\text{RETURN}\>\) cancels the command and causes the \(\text{APPLE SSC:}\) prompt to reappear.

Many of these commands override the physical switch settings on the SSC. This makes it unnecessary to open the Apple II case and manually flip the SSC switches. To change the values back to the physical switch settings, reboot or reset the Apple II, or type in the Reset command described below.

**COMMAND FORMATS**

All commands are preceded by the Printer Mode command character (usually \(<\text{CTRL-I}\>, \text{see below}\) and followed by \(<\text{RETURN}\>\). The notation \(<\text{CTRL-I}\>\) means "hold down the CTRL key while pressing I." There are three types of command formats:

- a number \(<n>\) followed by an uppercase letter (for example, 4D to set Data Format 4)
- simply an uppercase letter (for example, R to Reset the SSC)
- an uppercase letter followed by a space and then either E to Enable or D to Disable a feature (for example, L D to Disable automatic generation of linefeed characters)

The allowable range of \(<n>\) is given in each command description (next section). The choice of Enable or Disable is indicated as \(<\text{E/D}\>\).

The underscore character (\(_\)) before the \(<\text{E/D}\>\) in Enable/Disable commands is merely a reminder that a space is required there.

The SSC checks only numbers and the first letters of commands and options. All such letters must be uppercase. Further letters, which you can add to assist your memory, have no effect on the SSC. For example, X(OFF E)nable is the same as X E. The SSC ignores invalid commands.

**THE COMMAND CHARACTER**

The normal command character in Printer Mode is \(<\text{CTRL-I}\>\) (decimal 9; Appendix D). You can send the command character itself through the SSC by typing it twice in a row: \(<\text{CTRL-I}\><\text{CTRL-I}\>\); no \(<\text{RETURN}\>\) is required after this command. This special command allows you to transmit the command character without affecting the operation of the SSC, and without having to change to another command character and then back again later.
If you want to change the command character from `<CTRL-I>` to `<CTRL-something else>`, type `<CTRL-I><CTRL-something else>`. For example, to change the command character to `<CTRL-W>`, type `<CTRL-I><CTRL-W>`. To change back, type `<CTRL-W><CTRL-I>`. No `<RETURN>` is required after either of these commands.

The command character `<CTRL-I>` is ASCII code 9. Here is how to generate this character in BASIC and Pascal:

- **Integer BASIC:**
  ```plaintext```
  PRINT "#command" *embedded `<CTRL-I>`
  ```plaintext```

- **Applesoft BASIC:**
  ```plaintext```
  PRINT CHR$(9): "command"
  ```plaintext```

- **Pascal:**
  ```plaintext```
  WRITELN (CHR(9),'command');
  ```plaintext```

**PRINTER MODE COMMAND SUMMARY**

Table 2-4 is a summary of the commands available in Printer Mode. Some details, explained fully in the remainder of this chapter, have been omitted from the table for the sake of brevity. Commands marked with an asterisk are not supported by Pascal.
<table>
<thead>
<tr>
<th>Format</th>
<th>Command Name</th>
<th>Values</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;n&gt;B</td>
<td>Baud Rate</td>
<td>0 - 15</td>
<td>see Table 2-5</td>
</tr>
<tr>
<td>&lt;n&gt;C</td>
<td>&lt;CR&gt; Delay</td>
<td></td>
<td>no delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>32 milliseconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>250 milliseconds (1/4 s)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>2 seconds</td>
</tr>
<tr>
<td>&lt;n&gt;D</td>
<td>Data Format</td>
<td></td>
<td>8 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>7 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>6 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>5 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>8 data bits, 2 stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>7 data bits, 2 stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>6 data bits, 2 stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>5 data bits, 2 stop bits</td>
</tr>
<tr>
<td>&lt;n&gt;F</td>
<td>&lt;FF&gt; Delay</td>
<td></td>
<td>no delay (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>32 milliseconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>250 milliseconds (1/4 s)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>2 seconds</td>
</tr>
<tr>
<td>&lt;n&gt;L</td>
<td>&lt;LF&gt; Delay</td>
<td></td>
<td>no delay (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>32 milliseconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>250 milliseconds (1/4 s)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>2 seconds</td>
</tr>
<tr>
<td>&lt;n&gt;P</td>
<td>Parity</td>
<td>0,2,4,6</td>
<td>no parity (default = 0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>odd parity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>even parity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>MARK (parity bit always 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>SPACE (parity bit always 0)</td>
</tr>
<tr>
<td>&lt;n&gt;T</td>
<td>Translate</td>
<td>0</td>
<td>change LC to UC (default)</td>
</tr>
<tr>
<td></td>
<td>Lowercase (LC)</td>
<td>1</td>
<td>leave LC (possible garbage)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>LC to UC inverse; leave UC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>LC to UC; UC to inverse</td>
</tr>
<tr>
<td>*</td>
<td>C</td>
<td></td>
<td>auto-&lt;CR&gt; at column’s end</td>
</tr>
<tr>
<td>*</td>
<td>R</td>
<td></td>
<td>reset SSC + PR# and IN#</td>
</tr>
<tr>
<td>*</td>
<td>Z</td>
<td></td>
<td>ignore all &lt;CTRL&gt; commands</td>
</tr>
<tr>
<td>F &lt;E/D&gt;</td>
<td>Find Keyboard</td>
<td>E or D</td>
<td>accept keyboard entries</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>E or D send &lt;LF&gt; out after &lt;CR&gt;</td>
</tr>
<tr>
<td>L &lt;E/D&gt;</td>
<td>Generate &lt;LF&gt; Out</td>
<td>E or D</td>
<td>E or D send &lt;LF&gt; out in after &lt;CR&gt;</td>
</tr>
<tr>
<td>M &lt;E/D&gt;</td>
<td>Mask &lt;LF&gt; In</td>
<td>E or D</td>
<td>recognize BASIC tabs</td>
</tr>
<tr>
<td>*</td>
<td>T &lt;E/D&gt;</td>
<td>E or D</td>
<td>detect XOFF; await XON</td>
</tr>
<tr>
<td>*</td>
<td>X &lt;E/D&gt;</td>
<td>E or D</td>
<td></td>
</tr>
</tbody>
</table>

Table 2-4. Printer Mode Commands
COMMANDS THAT CHANGE SWITCH SETTINGS

The group of commands discussed in this section either directly override the SSC switch settings, or affect related behavior of the SSC. The Reset command restores the switch selections.

Baud Rate—<n>B

This command overrides the physical settings of switches SW1-1 through SW1-4 on the SSC. For example, to change the baud rate to 135 baud, type in <CTRL-I>4B<RETURN>.

<table>
<thead>
<tr>
<th>&lt;n&gt;</th>
<th>SSC Baud Rate</th>
<th>&lt;n&gt;</th>
<th>SSC Baud Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>use SW1-1 to SW1-4</td>
<td>8</td>
<td>1200</td>
</tr>
<tr>
<td>1</td>
<td>50</td>
<td>9</td>
<td>1800</td>
</tr>
<tr>
<td>2</td>
<td>75</td>
<td>10</td>
<td>2400</td>
</tr>
<tr>
<td>3</td>
<td>109.92 (110)</td>
<td>11</td>
<td>3600</td>
</tr>
<tr>
<td>4</td>
<td>134.58 (135)</td>
<td>12</td>
<td>4800</td>
</tr>
<tr>
<td>5</td>
<td>150</td>
<td>13</td>
<td>7200</td>
</tr>
<tr>
<td>6</td>
<td>300</td>
<td>14</td>
<td>9600</td>
</tr>
<tr>
<td>7</td>
<td>600</td>
<td>15</td>
<td>19200</td>
</tr>
</tbody>
</table>

Table 2-5. Baud Rate Selections

Data Format—<n>D

With this command you can override the settings of switch SW2-1. The table below shows how many data and stop bits correspond to each value of <n>. For example, <CTRL-I>2D<RETURN> causes the SSC to transmit each character in the form: one start bit (always transmitted), six data bits, and one stop bit.

<table>
<thead>
<tr>
<th>&lt;n&gt;</th>
<th>Data Bits</th>
<th>Stop Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>2 (1 with Parity options 4 through 7)</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>2 (1-1/2 with Parity options Ø through 3)</td>
</tr>
</tbody>
</table>

Table 2-6. Data Format Selections

Parity—<n>P

You can use this command to determine the kind of parity the SSC is to generate when sending data and check for when receiving data. In general, parity checking is not needed in Printer Mode. However, there are five parity options available (Table 2-4).
<table>
<thead>
<tr>
<th>Parity to Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$, 2, 4 or 6</td>
</tr>
<tr>
<td>none (default value)</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>odd parity (odd total number of ones)</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>even parity (even total number of ones)</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>MARK parity (parity bit always 1)</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>SPACE parity (parity bit always $\emptyset$)</td>
</tr>
</tbody>
</table>

Table 2-7. Parity Selections

For example, type `<CTRL>-I>1P<RETURN>` to cause the SSC to transmit and check for odd parity. Odd parity means that the high bit of every character is $\emptyset$ if there is already an odd number of 1 bits in that character, or 1 if there is otherwise an even number of 1 bits in the character, making the total always odd. This is an easy (but not foolproof) way to check data for transmission errors. Parity errors are recorded in a status byte (Appendix F).

Set Time Delay - `<n>C`, `<n>L`, `<n>F`

Some printers are slow and do not provide a "printer busy" or handshake signal to the Apple II. The `<n>C` command causes the Apple II to delay a specified amount of time, after sending a carriage return character, before sending another group (usually another line) to it. This gives the print head enough time to return to the left side of the page so it is ready to continue printing.

The `<n>C` command overrides the setting of switch SW2-2 on the SSC. That switch provides only two choices: no delay or a 250 millisecond delay.

The `<n>L` command allows time after a linefeed character for a printer platen to turn so the paper is vertically positioned to receive the next line.

The `<n>F` command allows time after a form feed character for the printer platen to move the paper form to the top of the next page (typically a longer time than a linefeed).

<table>
<thead>
<tr>
<th>Time Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset$</td>
</tr>
<tr>
<td>none</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>32 milliseconds</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>250 $\emptyset$ milliseconds (1/4 second)</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>2 seconds</td>
</tr>
</tbody>
</table>

Table 2-8. Time Delay Selections
Consult the user manual for the printer to find out how much time it takes to move its print head and platen, and so to determine an appropriate set of values for these three delays. The idea is to have at least enough time for the printer parts to move the required distance, but not so much time that overall printing speed is slowed down drastically. A typical set for a very slow printer would be \(<\text{CTRL-1}>2\text{C}<\text{RETURN}>\), \(<\text{CTRL-1}>2\text{L}<\text{RETURN}>\), \(<\text{CTRL-1}>3\text{F}<\text{RETURN}>\); that is, the SSC waits 250 milliseconds after transmitting carriage returns, 250 milliseconds after transmitting linefeeds, and 2 seconds after transmitting form feed characters.

**Generate \langle CR \rangle On Column Overflow-C**

Typing \(<\text{CTRL-1}>C<\text{RETURN}>\) causes the SSC to generate a carriage return character automatically any time the column count exceeds the printer line width.

Once this is on, only clearing the high-order bit at location $578+s$ (where $s$ is the slot the SSC is in) can turn this option back off. This option is normally off.

**Generate \langle LF \rangle Out-L_\langle E/D \rangle**

You can use this command to have the SSC automatically generate and transmit a linefeed character after each carriage return character. This overrides the setting of switch SW2-5. For example, you can type \(<\text{CTRL-1}>L\text{E}<\text{RETURN}>\) to cause your printer to print listings or double-spaced manuscripts for editing.

**Mask (Suppress) \langle LF \rangle In-M_\langle E/D \rangle**

If you type \(<\text{CTRL-1}>M\text{E}<\text{RETURN}>\), the SSC will suppress any incoming linefeed character that immediately follows a carriage return character.

**Reset the SSC-R**

Typing \(<\text{CTRL-1}>R<\text{RETURN}>\) has the same effect as sending a PR#Ø and an INSØ to a BASIC program and then resetting the SSC. This keyboard command cancels all previous commands to the SSC and puts the physical switch settings back into force.

**OTHER COMMANDS**

The commands described here affect the handling of characters and tabs. The Translate command determines how characters will appear on the video screen. The Z and F commands prevent the SSC from responding to control characters or ALL characters coming from the keyboard, respectively. The X command causes the SSC to respond to the XON/XOFF software protocol. Finally, the T command implements the tabbing feature of BASIC.
Translate Lowercase Characters—\(<n\>T\)

The Apple II Monitor "translates" all incoming lowercase characters into uppercase ones before sending them to the video screen or to a BASIC program. The SSC offers four translation options:

<table>
<thead>
<tr>
<th>(&lt;n)=</th>
<th>What to Do with Lowercase Characters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>Change all lowercase characters to uppercase ones before passing them to a BASIC program or to the video screen. This is the way the Apple II monitor handles lowercase.</td>
</tr>
<tr>
<td>1</td>
<td>Pass along all lowercase characters unchanged. The appearance of the lowercase characters on the Apple II screen is undefined (garbage).</td>
</tr>
<tr>
<td>2</td>
<td>Display lowercase characters as uppercase inverse characters (that is, as black characters on a white background).</td>
</tr>
<tr>
<td>3</td>
<td>Pass lowercase characters to programs unchanged, but display lowercase as uppercase, and uppercase as inverse uppercase (that is, as black characters on a white background).</td>
</tr>
</tbody>
</table>

Table 2-9. Lowercase Character Displays

Zap (Suppress) Control Characters—Z

Typing \(<\text{CTRL}-D\>Z<\text{RETURN}>\) prevents the SSC from recognizing any further control characters (and hence commands) whether coming from the keyboard or contained in a stream of characters moving through the SSC.

If you issue the Z command described here, all further commands are ignored; this is useful if the data you are transmitting contains bit patterns that the SSC can mistake for control characters.

The only way to reinstate command recognition after the Z command is to reinitialize the SSC, or clear the high-order bit at location \(5F8+s\) (where \(s\) is the slot in which the SSC is installed).

Find Keyboard—F_ <E/D>

You can protect incoming data from disruption by keystrokes with this command. For example, you can include an F D command in a program, followed by a routine that retrieves data coming in through the SSC, followed by F E later in the program. Default is F E.

XOFF Recognition—X_ <E/D>

Typing \(<\text{CTRL}-I>X E<\text{RETURN}>\) causes the SSC to look for any XOFF (decimal 19; Appendix D) character coming from a device attached to the SSC, and to respond to it by halting transmission of characters
until the SSC receives an XON (decimal 17; Appendix D) from the
device, signalling the SSC to continue transmission. In Printer
Mode, the default value of this command is X D.

In Printer Mode, full duplex communication may not work with
XOFF recognition turned on, so be careful.

Tab in BASIC-T_<E/D>
If you type in <CTRL-I>T E<RETURN>, the BASIC horizontal position
counter is left equal to the column count. All TABs work, including
back-tabs. TABs beyond column 40 require a POKE to location 36, as
usual. Commas only work as far as column 40, and BASIC programs
will be listed in 40-column format.
This chapter explains how to prepare, install and use the SSC in Communications Mode, and change the SSC's activities via commands.

PREPARING THE SSC FOR COMMUNICATIONS MODE

The SSC is ready to operate in Communications Mode when the jumper block and switches SW1-5 and SW1-6 are correctly positioned.

If the triangle on the jumper block is pointing up toward the word MODEM, remove the block (using an IC Extractor, if necessary) and reinsert it with the triangle pointing toward MODEM (Figure 3-1).

Using a pointed object, set switches SW1-5 and SW1-6 both ON as shown in Figure 3-1. This puts the SSC in Communications Mode.

Figure 3-1. SSC Set for Communications Mode
SETTING THE SWITCHES

Use the tip of a ballpoint pen or some other sharp object to flip the appropriate tiny switches on the SSC. A switch is ON when the top of the switch rocker is pushed in. The following subsections explain what settings to use.

COMMONLY USED SETTINGS

Table 3-1 lists the switch settings you can use for connection to various devices and services via the SSC and a modem.

<table>
<thead>
<tr>
<th>Application</th>
<th>Switch Settings</th>
<th>Cable Connections</th>
<th>Other Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple II via modem</td>
<td>SW1: ON</td>
<td>OFF ON ON ON ON ON SW2: ON ON * * OFF OFF OFF Comm Mode, 300 baud, 8 data, 1 stop, * * parity If using SSC in each Apple, set both the same; for local connection, second jumper block points toward TERMINAL.</td>
<td></td>
</tr>
<tr>
<td>Printer via modem</td>
<td>SW1: ON</td>
<td>OFF ON ON ON ON ON SW2: ON ON * * OFF OFF OFF Comm Mode, 300 baud, 7 data, 1 stop, * * parity Baud rate is limited by modem and transmission lines; some modems can also use 1200 baud; SW1-7 is always ON, and SW2-7 is always OFF; SCTS hookup is at remote modem.</td>
<td></td>
</tr>
<tr>
<td>Dow Jones News and Quotes Reporter</td>
<td>SW1: ON</td>
<td>OFF OFF ON ON ON ON SW2: ON OFF - ON OFF OFF Comm Mode, 300 baud, 7 data, 1 stop, no parity Sample program at end of this chapter sets same traits. Use T command for Terminal Mode operation.</td>
<td></td>
</tr>
</tbody>
</table>

Table 3-1. Commonly Used Switch Settings for Communications Mode

Make sure that the settings on the SSC, modem and remote device are all compatible. Successful operation using a modem depends on this.

After setting the switches on the SSC, you can go on to the next major section of this chapter, Installation Procedure.

BAUD RATE

No matter what kind of modem and remote device you connect to the SSC, the SSC is going to pass information between the Apple II and the device at a certain prearranged speed, called the baud rate. Since the Apple II can usually send and receive information faster than what is connected to it, the simplest way to determine the maximum baud rate you can use is to consult the user manual for the modem and remote device you will connect. Find out what rate is the fastest they both can handle. Once you know this, you are ready to
set the baud rate switches on the SSC. The following table shows the correct switch positions.

\[
\begin{array}{ccccc}
\text{Baud} & \text{SW1-1} & \text{SW1-2} & \text{SW1-3} & \text{SW1-4} \\
50 & ON & ON & ON & OFF \\
75 & ON & ON & OFF & ON \\
110* & ON & ON & OFF & OFF \\
135** & OFF & ON & ON & OFF \\
150 & ON & OFF & ON & OFF \\
300 & ON & OFF & OFF & ON \\
600 & ON & OFF & OFF & OFF \\
(* 109.92) & (** 134.58) & \\
\end{array}
\]

\[
\begin{array}{ccccc}
\text{Baud} & \text{SW1-1} & \text{SW1-2} & \text{SW1-3} & \text{SW1-4} \\
1200 & OFF & ON & ON & ON \\
1800 & OFF & ON & ON & OFF \\
2400 & OFF & ON & OFF & ON \\
3600 & OFF & ON & OFF & OFF \\
4800 & OFF & OFF & OFF & ON \\
7200 & OFF & OFF & OFF & OFF \\
9600 & OFF & OFF & OFF & OFF \\
19200 & OFF & OFF & OFF & OFF \\
\end{array}
\]

Table 3-2. Baud Rate Switch Settings

If you are connecting a printer or terminal at the other end of the modem, make sure that it is set (with its own switches, dials or thumb wheels) to the SAME baud rate! If you don't, the SSC will send and receive unrecognizable garbage.

**DATA FORMAT AND PARITY**

The SSC sends each character (such as a "7" or an "H" or a "?") as a string of zeroes and ones (bits). The way it can send a character in Communications Mode, using switch settings, is this:

- first a single start bit to signal to the printer or terminal that a character is coming;
- then a string of 7 or 8 data bits representing the character;
- possibly a parity bit for error checking;
- lastly one or two stop bits that signal the end of a character.

For Communications Mode, you can use switch settings to change three aspects of the data format: the number of data bits, the number of stop bits, and the kind (if any) of parity bit to send. Switches SW2-1 through SW2-4 determine the data format as shown in this table.

\[
\begin{array}{cccccc}
\text{Stop} & \text{Data} & \text{Parity} \\
\text{Bits} & \text{Bits} & \text{Bits} & \text{SW2-1} & \text{SW2-2} & \text{SW2-3} & \text{SW2-4} \\
1 & ON & 8 & ON & none & -- & ON \\
2 & OFF & 7 & OFF & odd & ON & OFF \\
& & & & even & OFF & OFF \\
\end{array}
\]

Table 3-3. Data Format Selections
If SW2-1 is OFF, the number of stop bits will be 1 instead of 2 if both 8 data bits (SW2-2 ON) and a parity bit (SW2-4 OFF) have been selected.

To determine the correct combination of switch settings, consult the literature describing the device or timesharing service you plan to connect to the SSC in this mode.

The most commonly used format for ASCII data is: 7 data bits, 1 stop bit, and no parity bit (SW2-1 and SW2-4 ON; SW2-2 OFF).

If you set the data rate switches to 50, 75 or 110 baud, choose a switch combination that specifies 2 stop bits; for all data rates 135 baud or higher, use 1 stop bit (switch SW2-1 ON), unless device or timesharing service literature specifies otherwise.

\[\text{To set the SSC for a data format different from those shown in this table, or to change the data format temporarily, use the SSC commands described later in this chapter.}\]

**GENERATE \langle LF\rangle OUT**

If the remote device (for example, a faraway printer) does not automatically generate linefeeds after carriage returns, and it desperately needs them, then set switch SW2-5 ON. Otherwise set SW2-5 OFF.

In Communications Mode, the SSC automatically discards incoming linefeeds that immediately follow carriage returns, unless you use the M D command as described later in this chapter.

**SPECIAL SWITCHES**

Switch SW2-6 controls forwarding of interrupts to the Apple II. Since the Apple II and II+ do not handle interrupts, set SW2-6 OFF.

For Communications Mode, set SW1-7 ON and SW2-7 OFF.

Your Super Serial Card is now ready to install and use in Communications Mode.

**INSTALLATION PROCEDURE**

This section explains how to install the SSC and its internal cable in the Apple II. If the cable clamp is not already assembled, do so now, following the instructions given in Chapter 1.
Before connecting or disconnecting anything on the Apple, turn off the power with the switch at the back left corner of the Apple case. THIS IS ABSOLUTELY NECESSARY. If you try to connect or disconnect anything from the inside of your Apple when the power is on, you are likely to damage the circuits.

Do not unplug the Apple, just turn it off. If you unplug the Apple, you will isolate it from earth ground and leave it vulnerable to static discharges.

Remove the Apple cover by pulling up on the two back corners of the cover until the two corner fasteners pop apart. Slide the cover back until it is free of the case and then lift the cover off.

Look inside the Apple and locate the power supply case—the rectangular metal box along the left inside the Apple II. To avoid damaging the SSC, touch the power supply case with one hand; this discharges any static charge that may be on your clothes or body.

Along the back inside edge of the Apple you will see eight long narrow slots called connector slots. The connector slots are numbered from 0 at the left to 7 at the right. The numbers are printed along the back edge behind the connector slots. For use with Pascal and a modem, install the SSC in slot #2. For use with BASIC, install the SSC in any slot from #1 through #7.

Handle the Super Serial Card as you would handle an expensive phonograph record. Grasp it only by the corners or edges, and do not touch the components or pins, especially the gold fingers on the edge connector.

There are three deep notches along the back of the Apple II case. Temporarily set the SSC down near the desired slot. Then take the clamp assembly and slide it down into the notch closest to the slot that the SSC will be in. Tighten the screws until the connector assembly can no longer be moved in the opening.

Grasp the upper corners of the SSC and insert the gold fingers of the edge connector into the slot in the back of the Apple, rear edge first. Gently push the front edge of the card down until it is level and firmly seated. Figure 3-2 shows how the SSC looks when installed in slot #2.

Note that the outer ends of the screws in the clamp assembly can act as nuts. They are threaded and can receive screws from the printer or terminal connector, to ensure a good connection with the Apple.
Slide the Apple case top plate in place and press down on the rear corners until the corner fasteners pop into place. The Super Serial Card is now installed.

**EXTERNAL CABLE AND CONNECTOR**

The SSC cable connector you installed in the notch is a standard DB-25 connector with 25 pins. Ten pins of the connector are connected internally to the SSC.

You will need a cable to connect the modem or other device to the SSC connector on the Apple II. Cables with 25-pin connectors on one end are available from your Apple dealer.

The cable must have internal shielding, with the shielding properly terminated at both ends, to prevent electromagnetic interference to nearby radios, television sets, and communication equipment. This shielding is necessary for the system to comply with Class B Federal Communications Commission limits as defined by Subpart J of Part 15 of the FCC rules. Unshielded cables are not recommended.

⚠️ Make sure that all devices are connected to the same grounded AC power circuit (three-wire wall outlet) as the Apple II. Connecting ungrounded equipment to your Apple II can cause severe electrical damage.

**USING SSC IN COMMUNICATIONS MODE**

Communications Mode allows you to use the SSC with a modem, connected to a remote device (such as a remote printer, terminal, or other computer). After installing the SSC, you can control its operation...
from a BASIC, Pascal or assembly-language program, or even directly from the keyboard. To use the SSC in Communications Mode, do the following:

- Make sure the jumper block points toward MODEM.

- Under BASIC or DOS, boot the Apple II, and then type in PR#s and IN#s to route input and output, respectively, to and from the remote device. (The SSC is in slot s.)

- Under Pascal, boot the Apple II and then use #7: or REMIN: for input, and #8: or REMOUT: for output. (The SSC is in slot #2.)

- If the modem and remote device don’t work, refer to Appendix E for troubleshooting hints, or consult your Apple dealer.

**COMMUNICATIONS MODE COMMANDS**

You can issue any of the commands described in this section by embedding them in a computer program. Under BASIC or DOS, you can also enter them directly at the Apple (or remote terminal) keyboard.

In a BASIC program, put the control character and command in a PRINT statement. In a Pascal program, embed the command in a WRITE or WRITELN statement.

Before keyboard entry of these commands has any effect on the SSC, you must first issue an IN#s command (with the SSC in slot s). When you then enter the command character (usually <CTRL-A>, see below), the prompt APPLE SSC: appears on the display screen. Subsequent characters up to <RETURN> will be interpreted as an SSC command. Pressing the left arrow key before pressing <RETURN> cancels the command and causes the APPLE SSC: prompt to reappear.

Many of these commands override the physical switch settings on the SSC. This makes it unnecessary to open the Apple II case and manually change the SSC switch settings. To change the values back to the physical switch settings, reboot or reset the Apple II, or type in the Reset command described below.

**COMMAND FORMATS**

All commands are preceded by the Communications Mode command character (usually <CTRL-A>, see below) and followed by <RETURN>. The notation <CTRL-A> means "hold down the CTRL key while pressing A." There are three types of command formats:
- a number \(<n>\) followed by an uppercase letter (for example, 4D to set Data Format 4)

- simply an uppercase letter (for example, R to Reset the SSC)

- an uppercase letter followed by a space and then either E to Enable or D to Disable a feature (for example, LD to Disable automatic generation of linefeed characters)

The allowable range of \(<n>\) is given in each command description below. The choice of Enable or Disable is written as \(<E/D>\).

The underscore character (\(\_\)) before the \(<E/D>\) in Enable/Disable commands is merely a reminder that a space is required there.

The SSC checks only numbers and the first letters of commands and options. All such letters must be uppercase. Further letters, which you can add to assist your memory, have no effect on the SSC. For example, E(cho E(nable) is the same as E E. The SSC ignores invalid commands.

THE COMMAND CHARACTER

The normal command character in Communications Mode is \(<CTRL-A>\).

You can send the command character itself through the SSC by typing it twice in a row: \(<CTRL-A><CTRL-A>\) (no \(<RETURN>\) necessary). This special command allows you to transmit the command character without affecting the operation of the SSC, and without having to change to another command character and then back again later.

If you want to change the command character from \(<CTRL-A>\) to \(<CTRL-something else>\)—for example, \(<CTRL-W>\)—type \(<CTRL-A><CTRL-W>\). To change back, type \(<CTRL-W><CTRL-A>\). No \(<RETURN>\) is required after either of these commands.

Do not change the control character to \(<CTRL-S>\), \(<CTRL-T>\) or \(<CTRL-R>\), since in Communications Mode the SSC interprets these as special control commands from a remote device.

The command character \(<CTRL-A>\) is ASCII code 1. Here is how to generate this character in BASIC and Pascal:

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer BASIC:</td>
<td>PRINT &quot;*command&quot; *embedded &lt;CTRL-A&gt;</td>
</tr>
<tr>
<td>AppleSoft BASIC:</td>
<td>PRINT CHR$(2): &quot;command&quot;</td>
</tr>
<tr>
<td>Pascal:</td>
<td>WRITELN (CHR(2),'command');</td>
</tr>
</tbody>
</table>

COMMUNICATIONS MODE COMMAND SUMMARY

Table 3-4 is a summary of the commands available in Communications Mode. Some details, explained fully in the remainder of this chapter, have been omitted from the table for the sake of brevity. Commands marked with an asterisk are not supported by Pascal.
<table>
<thead>
<tr>
<th>Format</th>
<th>Command Name</th>
<th>Values</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;n&gt;B</td>
<td>Baud Rate</td>
<td>0 - 15</td>
<td>see Table 3-5</td>
</tr>
<tr>
<td>&lt;n&gt;C</td>
<td>&lt;CR&gt; Delay</td>
<td>0 – 3</td>
<td>no delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>32 milliseconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>250 milliseconds (1/4 s)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>2 seconds</td>
</tr>
<tr>
<td>&lt;n&gt;D</td>
<td>Data Format</td>
<td>0 – 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>8 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>7 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>6 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>5 data bits, 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>8 data bits, 2 stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>7 data bits, 2 stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>6 data bits, 2 stop bits</td>
</tr>
<tr>
<td>&lt;n&gt;F</td>
<td>&lt;FF&gt; Delay</td>
<td>0 – 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>no delay (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>32 milliseconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>250 milliseconds (1/4 s)</td>
</tr>
<tr>
<td>&lt;n&gt;L</td>
<td>&lt;LF&gt; Delay</td>
<td>0 – 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>no delay (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>32 milliseconds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>250 milliseconds (1/4 s)</td>
</tr>
<tr>
<td>&lt;n&gt;P</td>
<td>Parity</td>
<td>0, 1, 2, 3, 4, 5, 6</td>
<td>no parity (default = 0F)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>odd parity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>even parity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>MARK (parity bit always 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>SPACE (parity bit always 0)</td>
</tr>
<tr>
<td>*</td>
<td>&lt;n&gt;S</td>
<td>0 – 7</td>
<td>chain SSC output to slot n</td>
</tr>
<tr>
<td>*</td>
<td>&lt;n&gt;T</td>
<td>0 – 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Translate</td>
<td>1</td>
<td>change all LC to UC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>leave LC (possible garbage)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>LC to UC inverse; leave UC</td>
</tr>
<tr>
<td>B</td>
<td>Break</td>
<td></td>
<td>transmit 233 ms BREAK</td>
</tr>
<tr>
<td>R</td>
<td>Reset the SSC</td>
<td></td>
<td>SW reset + PR#0 and IN#0</td>
</tr>
<tr>
<td>T</td>
<td>Terminal Mode</td>
<td></td>
<td>(see Terminal Mode section)</td>
</tr>
<tr>
<td>Z</td>
<td>Zap &lt;CTRL&gt;</td>
<td></td>
<td>ignore all &lt;CTRL&gt; commands</td>
</tr>
<tr>
<td>*</td>
<td>E &lt;E/D&gt;</td>
<td>E or D</td>
<td>echo input on the screen</td>
</tr>
<tr>
<td></td>
<td>Echo</td>
<td>E or D</td>
<td>accept keyboard entries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E or D</td>
<td>send &lt;LF&gt; out after &lt;CR&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E or D</td>
<td>drop &lt;LF&gt; in after &lt;CR&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E or D</td>
<td>detect XOFF; await XON</td>
</tr>
<tr>
<td>F &lt;E/D&gt;</td>
<td>Find Keyboard</td>
<td>E or D</td>
<td>echo input on the screen</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E or D</td>
<td>accept keyboard entries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E or D</td>
<td>send &lt;LF&gt; out after &lt;CR&gt;</td>
</tr>
<tr>
<td>L &lt;E/D&gt;</td>
<td>Generate &lt;LF&gt; Out</td>
<td>E or D</td>
<td>drop &lt;LF&gt; in after &lt;CR&gt;</td>
</tr>
<tr>
<td>M &lt;E/D&gt;</td>
<td>Mask &lt;LF&gt; In</td>
<td>E or D</td>
<td>detect XOFF; await XON</td>
</tr>
<tr>
<td>X &lt;E/D&gt;</td>
<td>XOFF Recognition</td>
<td>E or D</td>
<td>echo input on the screen</td>
</tr>
</tbody>
</table>

Table 3-4. Summary of Communications Mode Commands
COMMANDS THAT CHANGE SWITCH SETTINGS

The commands discussed in this section either override the SSC switch settings, or affect related behavior of the SSC. The Reset command restores the switch selections.

Baud Rate—\( \langle n \rangle B \)

This command overrides the physical settings of switches SW1-1 to SW1-4 on the SSC. For example, to change the rate to 9600 baud, type <CTRL-A>14B<RETURN>.

<table>
<thead>
<tr>
<th>( \langle n \rangle )</th>
<th>SSC Baud Rate</th>
<th>( \langle n \rangle )</th>
<th>SSC Baud Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>use SW1-1 to SW1-4</td>
<td>8</td>
<td>12000</td>
</tr>
<tr>
<td>1</td>
<td>5Ø</td>
<td>9</td>
<td>18000</td>
</tr>
<tr>
<td>2</td>
<td>75</td>
<td>10</td>
<td>2400</td>
</tr>
<tr>
<td>3</td>
<td>109.92 (110)</td>
<td>11</td>
<td>3600</td>
</tr>
<tr>
<td>4</td>
<td>134.58 (135)</td>
<td>12</td>
<td>4800</td>
</tr>
<tr>
<td>5</td>
<td>15Ø</td>
<td>13</td>
<td>7200</td>
</tr>
<tr>
<td>6</td>
<td>3Ø</td>
<td>14</td>
<td>9600</td>
</tr>
<tr>
<td>7</td>
<td>6Ø</td>
<td>15</td>
<td>19200</td>
</tr>
</tbody>
</table>

Table 3-5. Baud Rate Selections

Data Format—\( \langle n \rangle D \)

With this command you can override the settings of switches SW2-1 and SW2-2. The table below shows how many data and stop bits correspond to each value of \( \langle n \rangle \). For example, typing <CTRL-A>3D<RETURN> causes the SSC to transmit each character in the form: one start bit (always transmitted), five data bits, and one stop bit.

<table>
<thead>
<tr>
<th>( \langle n \rangle )</th>
<th>Data Bits</th>
<th>Stop Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>2 (1 with ( \langle n \rangle P ) options 4 through 7)</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>2 (1-1/2 with ( \langle n \rangle P ) options Ø through 3)</td>
</tr>
</tbody>
</table>

Table 3-6. Data Format Selections

Parity—\( \langle n \rangle P \)

You can use this command to determine the kind of parity the SSC is to generate when sending data and check for when receiving data. There are five parity options available:
<table>
<thead>
<tr>
<th>n</th>
<th>Parity to Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, 2, 4 or 6</td>
<td>none</td>
</tr>
<tr>
<td>1</td>
<td>odd parity (odd number of 1's)</td>
</tr>
<tr>
<td>3</td>
<td>even parity (even number of 1's)</td>
</tr>
<tr>
<td>5</td>
<td>MARK parity (parity bit always 1)</td>
</tr>
<tr>
<td>7</td>
<td>SPACE parity (parity bit always 0)</td>
</tr>
</tbody>
</table>

Table 3-7. Parity Selections

For example, type <CTRL-A>L<RETURN> to cause the SSC to transmit and check for odd parity. Odd parity means that the high bit of every character is 0 if there is already an odd number of 1 bits in that character, or 1 if there is otherwise an even number of 1 bits, making the total always odd. This is an easy (but not foolproof) way to check data for transmission errors. (See Appendix F.)

Generate <LF> Out-L_<E/D>
You can use this command to have the SSC automatically generate and transmit a linefeed (<LF>) character after each carriage return (<CR>) character. This overrides the setting of switch SW2-5. For example, you can type <CTRL-A>L<E<RETURN> to cause your printer to produce double-spaced listings or manuscripts for editing.

Mask (Suppress) <LF> In-M_<E/D>
If you type <CTRL-A>M D<RETURN>, the SSC will not remove incoming linefeed (<LF>) characters that immediately follow carriage return (<CR>) characters.

Reset the SSC—R
Typing <CTRL-A>R<RETURN> has the same effect as sending a PR#0 and an IN#0 to a BASIC program and then resetting the SSC. This keyboard command cancels all previous commands to the SSC and puts the physical switch settings back into force.

OTHER COMMANDS
The commands described in this subsection control the handling of characters and of the video screen. Three commands control timed delays following transmission of <CR>, <LF> and <FF> characters. The Translate command controls the display of lowercase and uppercase characters. The Z and F commands suppress control characters and characters entered at the keyboard, respectively. The X command causes the SSC to check the character stream for XOFF, as part of the XON/XOFF protocol. Finally, the <n>S command routes video output to a selected slot, and the E command suppresses display (echo) of characters on the screen.
Set Time Delays—\(<n>C, \(<n\)L, \(<n\)F
Some printers are slow and do not provide a "printer busy" or handshake signal to the Apple II. If such a printer is connected to the SSC via a modem, you may want to use these three delay commands.

The \(<n>C\) command causes the Apple II to delay a specified amount of time, after sending a carriage return character, before sending another group (usually another line) to it. This gives the print head enough time to return to the left side of the page so it is ready to continue printing.

The \(<n>L\) command allows time after a linefeed character for a printer platen to turn so the paper is vertically positioned to receive the next line.

The \(<n>F\) command allows time after a form feed character for the printer platen to move the paper form to the top of the next page (typically a longer time than a linefeed).

\[
\begin{array}{ll}
\text{\(<n>\)} & \text{Time Delay} \\
\emptyset & \text{none} \\
1 & 32 \text{ milliseconds} \\
2 & 25\emptyset \text{ milliseconds (1/4 second)} \\
3 & 2 \text{ seconds} \\
\end{array}
\]

Table 3-8. Time Delay Selections

Consult the user manual for the printer to find out how much time it takes to move its print head and platen, and so to determine an appropriate set of values for these three delays if a printer is used as the remote device. The idea is to have at least enough time for the printer parts to move the required distance, but not so much time that overall printing speed is slowed down drastically.

Translate Lowercase Characters—\(<n>T\)
The Apple II monitor "translates" all incoming lowercase characters into uppercase ones before sending them to the video screen or to a BASIC program. With the \(<n>T\) command, four options are available:
**What to Do with Lowercase Characters**

<table>
<thead>
<tr>
<th>0</th>
<th>Change all lowercase characters to uppercase before passing them to a BASIC program or to the video screen. This is what the Apple II monitor does to lowercase.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Pass along all lowercase characters unchanged. The appearance of the lowercase characters on the Apple II screen is undefined (garbage).</td>
</tr>
<tr>
<td>2</td>
<td>Display lowercase characters as uppercase inverse characters (that is, as black characters on a white background).</td>
</tr>
<tr>
<td>3</td>
<td>Pass lowercase characters to programs unchanged, but display lowercase as uppercase, and uppercase as inverse uppercase (that is, as black characters on a white background).</td>
</tr>
</tbody>
</table>

Table 3-9. Lowercase Character Displays

**Zap ( Suppress) Control Characters—Z**

Typing `<CTRL-A>z<RETURN>` prevents the SSC from recognizing any further control characters (and hence commands) in the stream of characters moving through the SSC.

If you issue the Z command, all further commands are ignored; this is useful if the data you are transmitting contains bit patterns that the SSC can mistake for control characters.

The only way to reinstate command recognition after invoking the Z command is to reset the SSC, or clear the high-order bit at location $5F8+s$ (with the SSC in slot s).

**Find Keyboard—F_—<E/D>**

You can protect incoming data from disruption by keystrokes with this command. For example, you can include `<CTRL-A>F D in a program, followed by a routine that retrieves data coming in through the SSC, followed by `<CTRL-A>F E later in the program.

**XOFF Recognition—X_—<E/D>**

In Communications Mode, the SSC automatically recognizes any XOFF (decimal 19; Appendix D) character coming from a device attached to it, and responds to it by halting transmission of characters. The SSC resumes transmission as soon as it receives an XON character (decimal 17; Appendix D) from the device. To disable XOFF recognition, use `<CTRL-A>X D<RETURN>`.
Specify Screen Slot—(n)S

With this command you can specify the slot number of the device where you want text or listings displayed. (Normally this is slot #0, the Apple II video screen.) This allows "chaining" of the SSC to another card slot, such as an 80-column-display peripheral card. For the firmware in the SSC to pass on information to the firmware in the other card, the other card must have an output entry point within its Cs00 space; this is the case for all currently available 80-column-display cards for the Apple II.

For example, let's say you have the SSC in slot #2 with a remote terminal connected to it, and an 80-column-display card in slot #3. Type <CTRL-A>3S<RETURN> to cause the data from the remote terminal to be chained through the card in slot #3, so that it is displayed on the Apple II in 80-column format. (Not available in Pascal.)

Echo Characters on the Screen—E_<E/D>

For the Apple II, as for most computers, displaying (echoing) a character on the video screen is a separate step from receiving it from the keyboard, though we tend to think of these as one step, as on a typewriter. For example, if you type in <CTRL-A>E D<RETURN>, the SSC does not forward incoming characters to the Apple II screen. This can be used to hide someone's password entered at a terminal, or to avoid double-display of characters.

TERMINAL MODE

Under Communication Mode, the SSC can enter Terminal Mode and make the Apple II act like an unintelligent terminal. This is useful for connecting the Apple II to a computer timesharing service, or for conversing with another Apple II.

Terminal Mode makes it possible to generate lowercase characters, plus the ten ASCII characters not provided on the Apple II keyboard (plus ESC, since <ESC> is used for this feature).

To generate lowercase characters, press <ESC> (the "ESCAPE" key near the upper left corner of the Apple II keyboard) once, and then type alphabetic characters as you would normally do. After that, to capitalize a single letter, press <ESC> again before typing the letter. To lock the keyboard in uppercase, press <ESC> twice in succession. To get back to lowercase, press <ESC> once, as before.

To generate one of the special ASCII characters listed in Table 3-10, first press <ESC> once (if necessary) to place the keyboard in lowercase mode. Then press <ESC> a second time, followed by one of the top-row keys as shown in Table 3-10. For example, to send a tilde, make sure the keyboard is in lowercase mode, then type <ESC> followed by 9.
<table>
<thead>
<tr>
<th>Format</th>
<th>Command Name</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>Enter Terminal Mode</td>
<td>Go into Terminal Mode.</td>
</tr>
<tr>
<td>B</td>
<td>Transmit a Break Signal</td>
<td>Send a 233-millisecond BREAK (signoff) signal.</td>
</tr>
<tr>
<td>* E &lt;E/D&gt;</td>
<td>Echo Enable/Disable</td>
<td>Default E D (full-duplex); use E E for half-duplex.</td>
</tr>
<tr>
<td>S &lt;E/D&gt;</td>
<td>Special Characters Enable/Disable</td>
<td>Default S E; allows/defeats generation of lowercase and special characters (Table 3-10).</td>
</tr>
<tr>
<td>* &lt;n&gt;T</td>
<td>Translate Lowercase Characters</td>
<td>Determine treatment of incoming lowercase characters.</td>
</tr>
<tr>
<td>* X &lt;E/D&gt;</td>
<td>XOFF Recognition Enable/Disable</td>
<td>Default X E; in Terminal Mode, X E makes SSC detect &lt;CTRL-R&gt; and &lt;CTRL-T&gt; (remote-control OFF &amp; ON, respectively), but not &lt;CTRL-S&gt;.</td>
</tr>
<tr>
<td>Q</td>
<td>Quit (Exit from Terminal Mode)</td>
<td>Return to normal Communications Mode operation.</td>
</tr>
</tbody>
</table>

* Fully described earlier in this chapter.

Table 3-11. Terminal Mode Commands

**Enter Terminal Mode-T**
This causes the Apple II to function as a full-duplex unintelligent terminal. You can use this command in conjunction with the ECHO command to simulate the half-duplex terminal mode of the old Apple II Communications Card. Type <CTRL-A>T<RETURN> to enter this mode.

If you enter Terminal Mode and don’t see what you type echoed on the Apple video screen, probably the modem link has not yet been established, or you need to use the E(cho E(nable command.
Transmit a Break Signal—B
Typing <CTRL-A>B<RETURN> causes the SSC to transmit a 233-millisecond break signal, recognized by most time-sharing systems as a signoff.

Special Characters—S (E/D)
Typing <CTRL-A>S E<RETURN> causes the SSC to interpret <ESC><n> pairs as special characters, allowing a keyboard in this way to generate all possible ASCII characters. If you type <CTRL-A>S D<RETURN>, the SSC will treat the <ESC> key like any other key.

Quit (Exit from) Terminal Mode—Q
Type <CTRL-A>Q<RETURN> to exit from terminal mode.

A TERMINAL MODE EXAMPLE
You can use the sample program below to change the SSC temporarily from the characteristics you ordinarily use, to the characteristics needed to make the Apple II into a dumb terminal connected to the Dow Jones News & Quotes Reporter. This program assumes that the SSC is set for Communications Mode and that the jumper block is pointing toward MODEM. Neither of these conditions can be changed by software. This program also assumes that the SSC is in slot #1 and that you want to chain I/O to an 80-column card in slot #3; these conditions you can change via software. To change this Integer BASIC program to an Applesoft program, substitute CHR$(5) for D$ and CHR$(2) for A$, and leave out program lines 40 and 42.

10 REM************************************************************************
20 REM * THIS PROGRAM SETS UP THE SSC FOR DOW JONES *
30 REM************************************************************************
40 D$="": REM TYPE <CTRL-D> ESCAPE CHARACTER BETWEEN QUOTES
42 A$="": REM TYPE <CTRL-A> COMMAND CHARACTER BETWEEN QUOTES
50 PRINT D$; "PRf": REM SSC IS IN SLOT #1;
52 PRINT A$; "6 BAUD": REM SET BAUD RATE TO 300;
54 PRINT A$; "1 DATA": REM DATA FORMAT OF 7 DATA, 1 STOP
56 PRINT A$; "Ø PARITY": REM AND NO PARITY;
58 PRINT A$; "LF DISABLE": REM NO <LF> GENERATION AFTER <CR>.
60 PRINT A$; "3 SLOUCHN": REM CHAIN TO CARD IN SLOT #3
62 PRINT A$; "TERM MODE": REM AND ENTER TERMINAL MODE.
70 REM************************************************************************
72 REM * NOW YOU SHOULD BE IN TERMINAL MODE, GETTING THE *
74 REM * INFO YOU NEED FROM THE DOW JONES SERVICE. WHEN *
76 REM * FINISHED, EXIT WITH THE <CTRL-A>Q(EXIT COMMAND). *
78 REM************************************************************************
100 REM Q(EXIT COMMAND SENDS CONTROL BACK TO THIS PROGRAM:
110 PRINT A$; "RESET": REM RESET SWITCH-SELECTED OPTIONS
120 END
CHAPTER 4
HOW THE SCC WORKS

This chapter is divided into three major sections. The first explains what the SSC does, using everyday terms wherever possible. Those of you already familiar with serial data communication can skip this section.

The second section is for anyone who wants an overview of the SSC’s operating modes and configuration possibilities.

The third section is a dyed-in-the-wool hardware theory of operation for both the expert and the adventuresome layperson.

SERIAL DATA COMMUNICATION

The SSC is a device that performs serial data communication. Let’s consider communication first, then data, and then serial data and data transfer.

Communication is easy enough: getting information from here to there or from there to here. In this discussion, the Apple II is "here." "There" can be nearby (local) or far enough away (remote) that some intermediate device, like a telephone, is needed. Information moving from here to there (out of the Apple) is called output; information moving from there to here (into the Apple) is called input.

Data denotes information in its many forms. For successful data communication, it is essential that both the sender and receiver agree on their interpretation of the data transferred.

Inside the Apple II, data can be numbers and letters and symbols, or program instructions for the computer to carry out, or pointers to storage locations, or error message numbers, or codes for generating pictures or sounds (or lots of other things).

In the Apple II, as in all other computers, data is represented in codes made up of ones and zeros, the only two digits allowed in the binary (two-element) system. Each one or zero is called a Binary digit or bit. In the binary system, as in our ordinary decimal...
system, you can count to as high a number as you want—it just takes more digits to get there than in the decimal system—and use each number as a code to represent that number of different items. Table 4-1 gives some examples of how many items you can represent with various quantities of digits.

<table>
<thead>
<tr>
<th>System</th>
<th>Digits</th>
<th>Using</th>
<th>You can represent</th>
</tr>
</thead>
<tbody>
<tr>
<td>decimal</td>
<td>Ø - 9</td>
<td>1</td>
<td>ten items (Ø through 9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>one hundred (Ø through 99)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>one thousand (Ø through 999)</td>
</tr>
<tr>
<td>binary</td>
<td>Ø and 1</td>
<td>1</td>
<td>two items (Ø or 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>four (Ø, 1, 1Ø or 11)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>eight (Ø through 111)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>sixteen (Ø through 1111)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>thirty-two (Ø through 11111)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>sixty-four (Ø through 111111)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>one hundred twenty-eight</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>two hundred fifty-six, etc.</td>
</tr>
</tbody>
</table>

Table 4-1. Binary and Decimal Digits and Quantities

For printers, plotters, terminals, and many other devices, 128 codes are enough to distinguish all the necessary characters: 52 for the upper and lowercase alphabet, 10 for the decimal digits, and dozens of others for punctuation marks and special symbols. As a result, the 128-character American Standard Code for Information Interchange (ASCII) is widely used. (This 7-bit code is listed in Appendix D.)

Throughout the world, post, telegraph, telex and wire services use 5-bit and 6-bit code sets, even though so few bits cannot represent a very large selection of items. Meanwhile, computers have a penchant for sending each other streams of 8-bit codes with obscure meanings. As long as sender and receiver agree on interpretation, any set of codes will do. The SSC can send all of them.

**PARALLEL DATA IN THE APPLE II**

The Apple II is called an eight-bit processor because the basic unit of data it uses and moves around internally is an eight-bit byte. The Apple II has sets of eight lines interconnecting its various internal parts, so it can move around all eight bits at the same time. Since the bits travel together like eight cars side by side on an eight-lane highway, data in the Apple II is called parallel data, and data movements within the Apple II are called parallel data transfers (Figure 4-1).
SERIAL DATA FOR LONG DISTANCES

Just as it would be extremely costly to build highways with eight lanes in each direction over great distances, so it is costly to connect two widely separated pieces of equipment using eight lines in each direction. So, many manufacturers produce computers, printers, plotters, terminals and so forth that send and receive information along one line in each direction, one bit after another. Such a setup, with bits moving from one place to another like a string of cars in a single lane, is called a serial data transfer (Figure 4-2).

DATA CONVERSION

Changing parallel data to serial data or vice versa is called data conversion (Figure 4-3). By convention (see the later subsection describing RS-232-C), whenever parallel data is converted to serial data, the right-hand bit is sent first. It is as though there were a traffic law that when a multi-lane highway narrows to a single lane, the car in the right lane goes first, then the car from the next lane to the left, etc.
RS-232-C DATA FORMATS

Serial data communication became popular so quickly that a group of manufacturers and the telephone company formed the Electronic Industries Association (EIA) to agree upon standard ways of sending and receiving data. What has become the most widely used standard in the world is called Revision C of standard RS-232, or RS-232-C. The SSC sends and receives data in accordance with this standard. The serial data has the form shown in Figure 4-3, plus a start bit at the beginning, an optional parity bit after the five to eight data bits, and finally one or two stop bits at the end (Figure 4-4). This is the data format that most RS-232-C devices use.

![Figure 4-4. RS-232-C Serial Data Format](image)

What is this mysterious parity bit all about? It is an optional extra bit set to 0 or 1 to make the total number of data and stop bits set to 1 an odd number (odd parity) or an even number (even parity); or this extra bit can always be set to 0 (called SPACE parity) or to 1 (MARK parity).

The combined total of data and parity bits set to 1 in Figure 4-4 is 5, an odd number (and the parity bit is 1), so it qualifies as a correct character if odd parity (or MARK parity) has been agreed upon by sender and receiver. However, if that same character were received under even parity (or SPACE parity), the receiving device would signal that a transmission error had occurred. If one bit in a character changes during transmission, parity checking will detect the error. If two bits change, the error will go undetected.

RS-232-C SIGNALS

Since the RS-232-C standard stems from the early days of telephone and telegraph, the names given to its signals may sound quaint to our "modern" ears. However, the signals correspond to familiar conditions that we take for granted when using a telephone. Table 4-2 lists the basic signals required by the RS-232-C standard, and what conditions they correspond to in a telephone call that you originate. Think of yourself as the Data Terminal (a terminus or end point of the conversation), and the phone as the Data Set (the communication device). Note: not is indicated by a bar above a signal name.
### RS-232-C Signals

<table>
<thead>
<tr>
<th>Signal Description</th>
<th>Abbreviation</th>
<th>Similar to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Terminal Ready</td>
<td>DTR</td>
<td>you pick up the phone</td>
</tr>
<tr>
<td>Data Set Ready</td>
<td>DSR</td>
<td>the phone is working</td>
</tr>
<tr>
<td>Request To Send</td>
<td>RTS</td>
<td>you want to talk</td>
</tr>
<tr>
<td>Clear To Send</td>
<td>CTS</td>
<td>the phone has established a connection and the person at the other end is ready to listen</td>
</tr>
<tr>
<td>Transmit Data</td>
<td>TxD</td>
<td>you speak into the phone</td>
</tr>
<tr>
<td>not Request To Send</td>
<td>RTS</td>
<td>you’ve finished talking and are ready to listen or to hang up</td>
</tr>
<tr>
<td>not Clear To Send</td>
<td>CTS</td>
<td>the phone has sent your words and is ready for your next request to send a message</td>
</tr>
<tr>
<td>not Data Terminal Rdy</td>
<td>DTR</td>
<td>you hang up</td>
</tr>
</tbody>
</table>

Table 4-2. RS-232-C Signals As Interpreted by the Sender

Here are the RS-232-C signals and how you would interpret them if you were to answer a telephone call (Table 4-3).

### RS-232-C Signals

<table>
<thead>
<tr>
<th>Signal Description</th>
<th>Abbreviation</th>
<th>Similar to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring Indicator</td>
<td>RI</td>
<td>the phone rings (optional)</td>
</tr>
<tr>
<td>Data Set Ready</td>
<td>DSR</td>
<td>you pick up the phone</td>
</tr>
<tr>
<td>Data Carrier Detect</td>
<td>DCD</td>
<td>you hear background noise</td>
</tr>
<tr>
<td>Receive Data</td>
<td>RXD</td>
<td>you hear what is said</td>
</tr>
<tr>
<td>not Data Set Ready</td>
<td>DSR</td>
<td>the other party has hung up</td>
</tr>
</tbody>
</table>

Table 4-3. RS-232-C Signals As Interpreted by the Receiver

### Modems

All of the above signals refer to the interaction between what RS-232-C calls Data Terminal Equipment (DTE—end points of data transfers, such as the Apple II or a printer) and what it calls Data Communication Equipment (DCE—transmitting or receiving devices, such as modems).

What is a **modem**? The name is short for MOdulator/DEModulator. As a modulator it takes electrical signals from a computer or printer (or other device) that it is connected to, and turns them into musical tones over a telephone line. As a demodulator it takes the musical tones it detects on a telephone line and turns them back into electrical signals for use by the printer or computer (or other device) that it is connected to. It also handles the RS-232-C control signals to and from that device (Figure 4-5).
By convention, the calling (originate) modem produces a fairly high tone (let's say LA) as the background or carrier signal that it sends; it then modulates (changes) that tone to SO to mean 0 and TI to mean 1. Meanwhile, the called (answer) modem plays a lower tone, MI, as a carrier signal, and modulates that tone to RE to indicate 0 or FA to indicate 1. In this way, both modems can send and receive information along the same wires without interpreting what they send as received messages and vice versa. (All their voices sound alike.)

Modem Eliminators

RS-232 signals are designed for the interactions of two DTE's, two DCE's, and telephone lines, as shown in Figure 4-5. What if you just want to connect two DTE's together in the same room, directly (for example, an Apple II and a printer)? You can use what is called a null modem or modem eliminator. The jumper block on the SSC does just that when it is connected with its triangle pointing toward the word TERMINAL.

By using different tones to send and receive information, modems can make sure that what comes from the "mouthpiece" (transmit register) of one DTE gets routed to the "earpiece" (receive register) of the other. A null modem simply crosses those two wires (Figure 4-6).

To simulate the other signal exchanges that modems would perform, the null modem interconnects the signal wires as shown in Figure 4-6. Thus RTS gets turned back to the sender as CTS as though the phone had instantly established a connection; RTS is also connected to DCD on the other side to pretend that a carrier signal has been detected. Finally, connecting DTR (willing to transfer data) from one side to both RI and DSR (a call arriving) on the other side completes the simulated telephone connection. (RI is optional.) The jumper block does it all!
SSC MODES AND CONFIGURATIONS

Figure 4-7 outlines the possible operating modes of the Super Serial Card and their relationships to each other.

Printer Mode
SW1-5 OFF
SW1-6 ON
Jumper Block: TERMINAL
(usually)
default: HW HS

Communications Mode
SW1-5 ON
SW1-6 ON
Jumper Block: MODEM
(usually)
default: XON/XOFF

SIC P8 Emulation Mode
SW1-5 OFF
SW1-6 OFF
Jumper Block: TERMINAL
no commands
SWX-Y = ptr Mode

SIC P8A Emulation Mode
SW1-5 OFF
SW1-6 OFF
Jumper Block: TERMINAL
no commands
SWX-Y = ptr Mode
ETX/ACK

Recognition of Parallel Card commands:
I:K:<n>N

full-duplex (default)

half-duplex (IEE cmd)

Unintelligent Terminal Mode
T command to enter
Q command to exit

Figure 4-7. SSC Operating Modes
Figure 4-8 illustrates the chief configurations possible with the Super Serial Card and how to set them up.

Figure 4-8. SSC Configurations
THEORY OF OPERATION

This section explains the SSC's overall theory of operation, but not the internal workings of each IC chip. If you would like such information, it is best to obtain specifications from the IC manufacturers. The most complex component is the ACIA, which is a Synertek 6551 or equivalent.

While reading through this section, you may find it useful to refer to Figure 4-9, a block diagram of the SSC, or to the schematic diagram in Appendix C. All references in the form 1A, 3C, etc., pertain to coordinates on the printed circuit board itself. Here is an inventory of the main components of the SSC:

- 5Ø-pin connection to the Apple II peripheral connector slot
- a 12-line address bus
- addressing and control logic (1B, 1C, 2C, 3C)
- a 2K-by-8-bit ROM (4B-5C)
- jumpers and bow ties for optional substitution of RAM (3-4A)
- two blocks of 7 switches each (1A, 2A)
- two registers for reading the switch settings (2B, 3B)
- an Asynchronous Communications Interface Adapter (ACIA; 4-5A) with its internal registers:
  - status/reset register
  - control register
  - transmit/receive data register
  - command register
- a 1.8432 MHz oscillator (3A) for the ACIA
- a transmit interface (6A) and a receive interface (7A)
- an 8-line data bus
- a buffer for the data bus (6C)
- a jumper block (6B) that can function as a modem eliminator
- a 1Ø-pin header (7B) to connect the SSC to a DB-25 jack via a short internal cable (discussed in Appendix C)

![Figure 4-9. Overall Block Diagram of the SSC](image-url)
ADDRESSING AND CONTROL LOGIC

The twelve address lines (A₀ - A₁₁) from the Apple II provide all the necessary $C000$ addressing on the SSC. Control logic at IB, IC, 2C and 3C, plus the signals RESET, DEVICE SELECT, I/O SELECT, and I/O STROBE, ensure the routing of signals to the appropriate addresses.

The SSC follows the Apple II protocol in its use of the $C800$ address space. An LS279 (1B) serves as a NAND gate, a pair of inverters, and a set-reset latch. The latch is set by an access to the $Csxx$ space, and is reset by access to the $CFxx$ space or by a reset. When this set-reset latch is set, the Apple II can access the $C800$ space on the SSC. A small RC filter prevents the latch from being reset by spurious noise.

ROM/RAM Space

The 2K ROM (4B-5C) containing the SSC driver firmware resides in the $C800$-$CFFF$ address space. However, an LS90 (2C) and an LS32 (3C) remap the addresses from the range $Cs00$-$CsFF$ to the range $CF00$-$CFFF$, since the $CFxx$ addresses are unusable. (Access to them disables use of the $C800$ address space.) As a result of this remapping, only one ROM is required, and none of the ROM space is wasted.

The SSC can use a 2K-by-8-bit RAM in place of the ROM. Between columns 3 and 4 and rows A and B on the SSC, there are three jumper pads and three bow ties. If you solder the jumper pads and cut the bow ties, pins 18, 20 and 21 will be, respectively, chip enable, output enable and read-write control (instead of ROM enables).

The ROM (or RAM) addresses are mapped as follows (Table 4-4). The first 256-byte block is the Peripheral Card ROM Space, selected when I/O SELECT from the Apple II drops to 0 volts. The remaining seven blocks are in the I/O Expansion ROM Space, selected when I/O STROBE from the Apple II drops to 0 volts.

<table>
<thead>
<tr>
<th>SSC ROM/RAM Addresses</th>
<th>Become Apple II Addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0700$ - $07FF$</td>
<td>$Cs00$ - $CsFF$</td>
</tr>
<tr>
<td>$0000$ - $00FF$</td>
<td>$C800$ - $C8FF$</td>
</tr>
<tr>
<td>$0100$ - $01FF$</td>
<td>$C900$ - $C9FF$</td>
</tr>
<tr>
<td>$0200$ - $02FF$</td>
<td>$CA00$ - $CAFF$</td>
</tr>
<tr>
<td>$0300$ - $03FF$</td>
<td>$CB00$ - $CBFF$</td>
</tr>
<tr>
<td>$0400$ - $04FF$</td>
<td>$CC00$ - $CCFF$</td>
</tr>
<tr>
<td>$0500$ - $05FF$</td>
<td>$CD00$ - $Ccff$</td>
</tr>
<tr>
<td>$0600$ - $06FF$</td>
<td>$CE00$ - $CEFF$</td>
</tr>
</tbody>
</table>

Table 4-4. SSC Address Remapping

46 SUPER SERIAL CARD
Registers in Peripheral I/O Space

Whenever DEVICE SELECT drops to 0 volts, the Apple II is addressing the SSC's Peripheral I/O Space (the sixteen bytes starting at $C080 + 0$). This signal is combined logically with address lines A0 through A3 to select one of the six registers that reside in that space (Table 4-5).

<table>
<thead>
<tr>
<th>Chip selected</th>
<th>Address(+sΦ)</th>
<th>Purpose of register</th>
</tr>
</thead>
<tbody>
<tr>
<td>LS365 (2B)</td>
<td>$C081</td>
<td>store state of SW1 (1A) (read)</td>
</tr>
<tr>
<td>LS365 (3B)</td>
<td>$C082</td>
<td>store state of SW2 (2A) and state of CTS (read)</td>
</tr>
<tr>
<td>ACIA (4-5A)</td>
<td>$C088</td>
<td>receive (read), transmit (write)</td>
</tr>
<tr>
<td>ACIA (4-5A)</td>
<td>$C089</td>
<td>status (read), reset (write)</td>
</tr>
<tr>
<td>ACIA (4-5A)</td>
<td>$C08A</td>
<td>command (read and write)</td>
</tr>
<tr>
<td>ACIA (4-5A)</td>
<td>$C08B</td>
<td>control (read and write)</td>
</tr>
</tbody>
</table>

Table 4-5. Registers in SSC Peripheral I/O Space

The two LS365 chips act as buffers so that the state of eleven of the fourteen available switches, plus the state of RS-232-C signal Clear To Send (CTS), can be read. There are 3.3K ohm pullup resistors at the switch inputs of the LS365 chips. A closed switch pulls down an input, and it is read as zero.

Three switches are not connected to the LS365s. Switch SW2-6, when ON, passes interrupt requests from the ACIA to the Apple II. (The Apple II, however, currently does not support interrupts.) Setting switches SW1-7 ON and SW2-7 OFF connects DB-25 pin 8 (DCD) to the DCD input of the ACIA. Setting SW1-7 OFF and SW2-7 ON splices pin 19, Secondary Clear To Send (SCTS), onto the DCD input of the ACIA when the jumper block is in the TERMINAL position.

The ACIA has two pins used to select one of its four registers. While address lines A2 and A3 select the chip, A0 and A1 select the actual register. The SSC firmware reads and writes ACIA register contents; these registers are discussed in detail in Appendix A.

THE ACIA

The Asynchronous Communications Interface Adapter (ACIA) is the central and most complex element of the SSC. It and the crystal at 3A form a 1.8432 MHz oscillator. The ACIA divides this frequency down to one of the fifteen baud rates it supports. The ACIA also handles all incoming and outgoing primary RS-232-C signals. The ACIA registers (discussed fully in Appendix A) control hardware handshaking and select the baud rate, data format and parity. Finally, the ACIA performs parallel/serial and serial/parallel data conversion, and single-buffers data transfers.
DATA INPUT AND OUTPUT
The MC1489 at 7A converts the incoming serial data from RS-232-C to TTL voltage levels. The MC1488 at 6A converts the outgoing serial data from TTL to RS-232-C voltage levels, and in conjunction with three capacitors limits the output slew rate. Three of the received handshake lines (Clear To Send, Data Carrier Detect, and Data Set Ready) have 15K ohm pullup resistors so the SSC will work with devices that do not assert those signals.

DATA BUS
The 8-bit data bus on the SSC is, of course, a parallel bus. The ACIA takes output from it and gives input to it in parallel form. Also connected to the bus are the two switch detection registers (2B and 3B) and the ROM or RAM chip.

An LS245 (6C) buffers the output to the data bus, and minimizes input loading. The data bus has a 3.3K ohm pullup resistor on each line so the data inputs on the LS245 are not floating when it turns on in output mode.

JUMPER BLOCK
The jumper block has two positions: when its arrow points toward MODEM, the SSC looks like Data Terminal Equipment (DTE); that is, the SSC is prepared to talk to Data Communication Equipment (DCE), such as a modem. When installed with its arrow pointing toward TERMINAL, the jumper block acts as a modem eliminator (null modem); that is, the SSC looks like the DCE on the other device's side of a serial communication connection. In this position, the SSC can talk directly to a printer or any other DTE. Figure 4-6 shows the signal swapping that the jumper block in the TERMINAL position performs.
APPENDIX A
FIRMWARE

This appendix contains the following information:

- an explanation of the Pascal 1.1 firmware card protocol
- a firmware memory map
- a description of the SSC's use of its peripheral slot
  scratchpad RAM addresses
- a description of the ACLA registers and switch detection
  registers in the SSC's peripheral I/O space
- a list of firmware entry points
- the actual SSC firmware listings

PASCAL 1.1 FIRMWARE PROTOCOL

The old Apple II Serial Interface Card (SIC) ran under Pascal 1.0 with three direct firmware entry points, one for each of the three I/O functions it supported:

<table>
<thead>
<tr>
<th>Address</th>
<th>Contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C800</td>
<td>initialization routine entry point</td>
</tr>
<tr>
<td>$C84D</td>
<td>read routine entry point</td>
</tr>
<tr>
<td>$C9AA</td>
<td>write routine entry point</td>
</tr>
</tbody>
</table>

New peripheral cards can be "accepted" into the Pascal 1.0 system by appearing to be a SIC; that is, with these same three entry points and with $38 at $C805 and $18 at $C907 (see Device ID section below).

Pascal 1.1, on the other hand, has a more flexible setup, and also supports more I/O functions. It can make indirect calls to the firmware in a (new) peripheral card through addresses in a branch table in the card's firmware. It also has facilities for uniquely identifying new peripheral I/O devices.
I/O ROUTINE ENTRY POINTS
The I/O routine entry point branch table is located near the beginning of the $CS\times0$ address space (s being the slot number where the peripheral card is installed). This space was chosen instead of the $CB\times0$ space, since under BASIC protocol the $CS\times0$ space is required, while the $CB\times0$ space is optional.

The branch table locations that Pascal 1.1 uses are:

<table>
<thead>
<tr>
<th>Address</th>
<th>Contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CS\times D$</td>
<td>initialization routine offset (required)</td>
</tr>
<tr>
<td>$CS\times E$</td>
<td>read routine offset (required)</td>
</tr>
<tr>
<td>$CS\times F$</td>
<td>write routine offset (required)</td>
</tr>
<tr>
<td>$CS\times O$</td>
<td>status routine offset (required)</td>
</tr>
<tr>
<td>$CS\times L$</td>
<td>$0$ if optional offsets follow; non-zero if not</td>
</tr>
<tr>
<td>$CS\times 2$</td>
<td>control routine offset (optional)</td>
</tr>
<tr>
<td>$CS\times 3$</td>
<td>interrupt handling routine offset (optional)</td>
</tr>
</tbody>
</table>

Notice that $CS\times L$ contains $0$ only if the control and interrupt handling routines are supported by the firmware. (For example, the SSC does not support these two routines, and so location $CS\times L$ contains a (non-zero) firmware instruction.) Apple II Pascal 1.0 and 1.1 do not support control and interrupt requests, but such requests may be implemented in future versions of the Pascal BIOS and other future Apple II operating systems.

Here are the entry point addresses, and the contents of the 652 registers on entry to and on exit from Pascal 1.1 I/O routines:

<table>
<thead>
<tr>
<th>Addr.</th>
<th>Offset for</th>
<th>X Register</th>
<th>Y Register</th>
<th>A Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CS\times D$</td>
<td>Initialization</td>
<td>$CS$</td>
<td>$S0$</td>
<td>(unchanged)</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>$CS$</td>
<td>$S0$</td>
<td>(unchanged)</td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
</tr>
<tr>
<td>$CS\times E$</td>
<td>Read</td>
<td>$CS$</td>
<td>$S0$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>$CS$</td>
<td>$S0$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(unchanged)</td>
<td>character read</td>
</tr>
<tr>
<td>$CS\times F$</td>
<td>Write</td>
<td>$CS$</td>
<td>$S0$</td>
<td>char. to write</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>$CS$</td>
<td>$S0$</td>
<td>char. to write</td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
</tr>
<tr>
<td>$CS\times O$</td>
<td>Status</td>
<td>$CS$</td>
<td>$S0$</td>
<td>request (0 or 1)</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>$CS$</td>
<td>$S0$</td>
<td>request (0 or 1)</td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(changed)</td>
<td>(unchanged)</td>
</tr>
</tbody>
</table>

Notes: Request code 0 means, "Are you ready to accept output?" Request code 1 means, "Do you have input ready?" On exit, the reply to the status request is in the carry bit: carry clear means "No"; carry set means "Yes."

Table A-1. I/O Routine Offsets and Registers under Pascal 1.1
**DEVICE IDENTIFICATION**

Pascal 1.1 uses four firmware bytes to identify the peripheral card. Both the identifying bytes and the branch table are near the beginning of the $Cs00$ ROM space. The identifiers are listed in Table A-2.

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Cs05$</td>
<td>$38$</td>
<td>(like the old Serial Interface Card)</td>
</tr>
<tr>
<td>$Cs07$</td>
<td>$18$</td>
<td>(like the old Serial Interface Card)</td>
</tr>
<tr>
<td>$Cs08$</td>
<td>$01$</td>
<td>(the Generic Signature of new FW cards)</td>
</tr>
<tr>
<td>$Cs0C$</td>
<td>$c1$</td>
<td>(the Device Signature; see below)</td>
</tr>
</tbody>
</table>

Table A-2. Bytes Used for Device Identification

The first digit, c, of the Device Signature byte identifies the device class as listed in Table A-3.

<table>
<thead>
<tr>
<th>Digit</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0$</td>
<td>reserved</td>
</tr>
<tr>
<td>$1$</td>
<td>printer</td>
</tr>
<tr>
<td>$2$</td>
<td>joystick or other X-Y input device</td>
</tr>
<tr>
<td>$3$</td>
<td>serial or parallel I/O card</td>
</tr>
<tr>
<td>$4$</td>
<td>modem</td>
</tr>
<tr>
<td>$5$</td>
<td>sound or speech device</td>
</tr>
<tr>
<td>$6$</td>
<td>clock</td>
</tr>
<tr>
<td>$7$</td>
<td>mass storage device</td>
</tr>
<tr>
<td>$8$</td>
<td>80-column card</td>
</tr>
<tr>
<td>$9$</td>
<td>network or bus interface</td>
</tr>
<tr>
<td>$A$</td>
<td>special purpose (none of the above)</td>
</tr>
<tr>
<td>$B-F$</td>
<td>reserved for future expansion</td>
</tr>
</tbody>
</table>

Table A-3. Device Class Digit

The second digit, i, of the Device Signature byte is a unique identifier for the card, assigned by Apple Technical Support. For example, the SSC has a Device Signature of $31$: the 3 signifies that it is a serial or parallel I/O card, and the 1 is the low-order digit supplied by Apple Technical Support.

Although version 1.1 of Pascal ignores the Device Signature, applications programs can use them to identify specific devices.
## SSC Firmware Memory Usage

Table A-4 is an overall map of the locations that the SSC uses, both in the Apple II and in the SSC’s own firmware address space.

<table>
<thead>
<tr>
<th>Addresses</th>
<th>Name of area</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0000-$00FF</td>
<td>Page Zero</td>
<td>Monitor pointers, I/O hooks, and temporary storage (Table A-5)</td>
</tr>
<tr>
<td>$04xx-$07xx</td>
<td>Peripheral Slot</td>
<td>Locations (8 per slot) in Apple’s pages $04 through $07. SSC uses all eight of them (Table A-6)</td>
</tr>
<tr>
<td>(selected</td>
<td>Scratchpad RAM</td>
<td></td>
</tr>
<tr>
<td>locations)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C0(8+s)0</td>
<td>Peripheral Card</td>
<td>Locations (16 per slot) for general I/O; SSC uses 6 bytes (Table A-7)</td>
</tr>
<tr>
<td>$C0(8+s)F</td>
<td>I/O Space</td>
<td></td>
</tr>
<tr>
<td>$Cs00-$CsFF</td>
<td>Peripheral Card</td>
<td>One 256-byte page reserved for card in slot s; first page of SSC FW</td>
</tr>
<tr>
<td></td>
<td>ROM Space</td>
<td></td>
</tr>
<tr>
<td>$C800-$CFFFF</td>
<td>Expansion ROM</td>
<td>Eight 256-byte pages reserved for a 2K ROM or PROM; SSC maps its FW onto $C800-$CFFFF (Table A-4)</td>
</tr>
</tbody>
</table>

Table A-4. Memory Usage Map

## Zero Page Locations

The SSC makes use of these zero-page locations (Table A-5):

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* $24</td>
<td>CH</td>
<td>Monitor pointer to current position of cursor on screen</td>
</tr>
<tr>
<td>$26</td>
<td>SLOT16</td>
<td>Usually (slot# x 16); that is, $s0</td>
</tr>
<tr>
<td>$27</td>
<td>CHARACTER</td>
<td>Input or output character</td>
</tr>
<tr>
<td>* $28</td>
<td>BASL</td>
<td>Monitor pointer to current screen line</td>
</tr>
<tr>
<td>$2A</td>
<td>ZTMP1</td>
<td>Temporary storage (various uses)</td>
</tr>
<tr>
<td>$2B</td>
<td>ZTMP2</td>
<td>Temporary storage (various uses)</td>
</tr>
<tr>
<td>$35</td>
<td>ZTMP3</td>
<td>Temporary storage (various uses)</td>
</tr>
<tr>
<td>* $36</td>
<td>CSWL</td>
<td>BASIC output hook (not for Pascal)</td>
</tr>
<tr>
<td>* $37</td>
<td>CSWH</td>
<td>(high byte of CSW)</td>
</tr>
<tr>
<td>* $38</td>
<td>KSWL</td>
<td>BASIC input hook (not for Pascal)</td>
</tr>
<tr>
<td>* $39</td>
<td>KSWH</td>
<td>(high byte of KSW)</td>
</tr>
<tr>
<td>* $4E</td>
<td>RNDL</td>
<td>random number location, updated when looking for a keypress (not used when initialized by Pascal)</td>
</tr>
</tbody>
</table>

* Not used when Pascal initializes SSC.

Table A-5. Zero-Page Locations Used by SSC
## Scratchpad RAM Locations

The SSC uses the Scratchpad RAM locations as listed in Table A-6.

<table>
<thead>
<tr>
<th>Address</th>
<th>Field name</th>
<th>Bit(s)</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0478+s$</td>
<td>DELAYFLG</td>
<td>0 - 1</td>
<td>&lt;FF&gt; delay selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - 3</td>
<td>&lt;LF&gt; delay selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - 5</td>
<td>&lt;CR&gt; delay selection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 - 7</td>
<td>Translate option</td>
</tr>
<tr>
<td>$04F8+s$</td>
<td>HANDSHKE</td>
<td>0 - 7</td>
<td>Buffer count for handshake (P8A Mode)</td>
</tr>
<tr>
<td></td>
<td>PARAMETER</td>
<td></td>
<td>Accumulator for FW's command processor</td>
</tr>
<tr>
<td>$0578+s$</td>
<td>STATEFLG</td>
<td>0 - 2</td>
<td>Command mode when not Ø (Printer and Communications Modes only)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 - 4</td>
<td>Enquire character (P8A Mode); dflt ETX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - 5</td>
<td>Slot to chain to (Communications Mode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>Set to 1 after lowercase input character</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Terminal Mode when 1 (Comm Mode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Enable &lt;CR&gt; gen. when 1 (other 3 modes)</td>
</tr>
<tr>
<td>$05F8+s$</td>
<td>CMDBYTE</td>
<td>0 - 6</td>
<td>Printer Mode default is &lt;CTRL-I&gt;; Comm Mode default is &lt;CTRL-A&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Set to 1 to Zap control commands</td>
</tr>
<tr>
<td>$0678+s$</td>
<td>STSBYTE</td>
<td></td>
<td>Status and IORESULT byte (Appendix F)</td>
</tr>
<tr>
<td>$06F8+s$</td>
<td>CHNBYTE</td>
<td>0 - 2</td>
<td>Current Apple screen slot (Comm Mode); when slot = Ø, chaining is enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - 7</td>
<td>$\text{CS}$Ø space entry point (Comm Mode)</td>
</tr>
<tr>
<td></td>
<td>PWDBYTE</td>
<td>0 - 7</td>
<td>Current printer width (other modes); for listing compensation, auto-&lt;CR&gt;</td>
</tr>
<tr>
<td>$0778+s$</td>
<td>BUFBYTE</td>
<td>0 - 6</td>
<td>One-byte input buffer (Comm Mode); used in conjunction with XOFF recognition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Set to 1 when buffer full (Comm Mode)</td>
</tr>
<tr>
<td></td>
<td>COLBYTE</td>
<td>0 - 7</td>
<td>Current-column counter for tabbing, etc. (other 3 modes)</td>
</tr>
<tr>
<td>$07F8+s$</td>
<td>MISCFLG</td>
<td>Ø</td>
<td>Generate &lt;LF&gt; after &lt;CR&gt; when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Printer Mode when Ø; Comm Mode when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Keyboard input enabled when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>&lt;CTRL-S&gt; (XOFF), &lt;CTRL-R&gt; and &lt;CTRL-T&gt; input checking when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>Pascal Op Sys when 1; BASIC when Ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>Discard &lt;LF&gt; input when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>Enable lowercase and special character generation when 1 (Comm Mode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>Tabbing option on when 1 (Printer Mode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Echo output to Apple screen when 1</td>
</tr>
</tbody>
</table>

Table A-6. Scratchpad RAM Locations Used by SSC
PERIPHERAL CARD I/O SPACE

There are 16 bytes of I/O space allocated to each slot in the Apple II. Each set begins at address $C0B0 + (slot \times 16)$; for example, if the SSC is in slot 3, its group of bytes extends from $C0B0$ to $C0BF$. Table A-7 interprets the 6 bytes the SSC uses.

<table>
<thead>
<tr>
<th>Address</th>
<th>Register</th>
<th>Bit(s)</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C081+s0$</td>
<td>DIPSW1</td>
<td>ø</td>
<td>SW1-6 is OFF when 1, ON when ø</td>
</tr>
<tr>
<td>(SW1-x)</td>
<td></td>
<td>1</td>
<td>SW1-5 is OFF when 1, ON when ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 - 7</td>
<td>same as above for SW1-4 through SW1-1</td>
</tr>
<tr>
<td>$C082+s0$</td>
<td>DIPSW2</td>
<td>ø</td>
<td>Clear To Send (CTS) is true (-) when ø</td>
</tr>
<tr>
<td>(SW2-x)</td>
<td></td>
<td>1 - 3</td>
<td>same as above for SW2-5 through SW2-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 &amp; 7</td>
<td>same as above for SW2-2 &amp; SW2-1</td>
</tr>
<tr>
<td>$C088+s0$</td>
<td>TDREG</td>
<td>ø - 7</td>
<td>ACIA Transmit Register (write)</td>
</tr>
<tr>
<td></td>
<td>RDREG</td>
<td>ø - 7</td>
<td>ACIA Receive Register (read)</td>
</tr>
<tr>
<td>$C089+s0$</td>
<td>STATUS</td>
<td>ø</td>
<td>ACIA Status/Reset Register</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Parity error detected when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Framing error detected when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>Overrun detected when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>ACIA Receive Register full when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>ACIA Transmit Register empty when 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>Data Carrier Detect (DCD) true when ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Data Set Ready (DSR) true when ø</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Interrupt (IRQ) has occurred when 1</td>
</tr>
<tr>
<td>$C08A+s0$</td>
<td>COMMAND</td>
<td>ø</td>
<td>ACIA Command Register (read/write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Data Terminal Ready (DTR): enable (1) or disable (ø) receiver and all interrupts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 - 3</td>
<td>When 1, allow STATUS bit 3 to cause IRQ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>Control transmit interrupt, Request To Send (RTS) level, and transmitter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - 7</td>
<td>When ø, normal mode for receiver; when 1, echo mode (but bits 2 and 3 must be ø)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Control parity (values: Table 2-7)</td>
</tr>
<tr>
<td>$C08B+s0$</td>
<td>CONTROL</td>
<td>ø - 3</td>
<td>ACIA Control Register (read/write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Baud rate: $Ø = 16$ times external clock; $1 - 8F = \text{decimal in Table 2-5}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>When 1, use baud rate generator; when ø, use external clock (not supported)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 - 6</td>
<td>Number of data bits: 8 (bit 5 and 6 = ø)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7 (5 = 1, 6 = ø), 6 (5 = ø, 6 = 1) or 5 (bit 5 and 6 both = 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Number of stop bits: 1 (bit 7 = ø); if bit 7 = 1, then 1-1/2 (with 5 data bits, no parity), 1 (8 data plus parity) or 2</td>
</tr>
</tbody>
</table>

Table A-7. SSC Registers in Peripheral Card I/O Space
SSC ENTRY POINTS

This section contains the SSC firmware entry points for the Apple II Monitor, BASIC, Pascal 1.0 and Pascal 1.1. The Pascal 1.1 entry point offsets conform to the Firmware card protocol outlined in the first section of this appendix.

MONITOR ROM ENTRY POINTS

The SSC uses these entry points in the Monitor ROM, unless Pascal initializes the SSC.

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$FDED</td>
<td>COUT</td>
<td>sends a character to output hook (chaining) used for chaining</td>
</tr>
<tr>
<td>$FE89</td>
<td>SETKBD</td>
<td>sets KSW to point to keyboard (reset)</td>
</tr>
<tr>
<td>$FE93</td>
<td>SETSCR</td>
<td>sets CSW to point to Apple screen (reset)</td>
</tr>
<tr>
<td>$FF58</td>
<td>IORTS</td>
<td>known position of an RTS instruction</td>
</tr>
<tr>
<td>$FDF6</td>
<td>VIDOUT</td>
<td>sends a character to the Apple screen</td>
</tr>
</tbody>
</table>

Table A-8. Monitor ROM Entry Points Used by SSC

BASIC ENTRY POINTS

Here are the entry point addresses, and the contents of the 6502 registers on entry to and on exit from BASIC I/O routines:

<table>
<thead>
<tr>
<th>Addr.</th>
<th>Routine</th>
<th>X Register</th>
<th>Y Register</th>
<th>A Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Cs00</td>
<td>Initialization</td>
<td>anything</td>
<td>anything</td>
<td>anything</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
<td>character</td>
</tr>
<tr>
<td>Notes:</td>
<td>CSW and/or KSW points to $Cs00. The character in the A register is output unless KSW points to $Cs00 and CSW does not point to $Cs00.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| $Cs05 | Input | anything | anything | anything |
|       | On entry | (unchanged) | (unchanged) | character in |
| Notes: | Character in is from ACIA or keyboard. |

| $Cs07 | Output | anything | anything | character out |
|       | On entry | (unchanged) | (changed) | |
| Notes: | Character out is transmitted through the ACIA. |

Table A-9. BASIC Entry Points Used by SSC
PASCAL 1.0 ENTRY POINTS

There are three Pascal 1.0 entry points: one for initialization, one for read operations, and one for write operations. These entry points are direct addresses.

<table>
<thead>
<tr>
<th>Addr.</th>
<th>Routine</th>
<th>X Register</th>
<th>Y Register</th>
<th>A Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C800</td>
<td>Initialization</td>
<td>$Cs</td>
<td>$S0</td>
<td>anything</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>$Cs</td>
<td>$S0</td>
<td>(unchanged)</td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>$Cs</td>
<td>$S0</td>
<td></td>
</tr>
</tbody>
</table>

Notes: $C800 space is enabled. Firmware initializes SSC to default values plus SW1 and SW2 selections.

$C84D  Read

| On entry | $Cs | $S0 | anything |
| On exit  | $Cs | $S0 | character in |

Notes: $C800 space is enabled. Pascal returns ACIA or keyboard data in the A Register and location $678+s with high bit cleared.

$C9AA  Write

| On entry    | $Cs | $S0 | character out |
| On exit     | Error code | $Cs | (changed) |

Notes: $C800 space is enabled. Output character is transmitted through the ACIA. Pascal posts error code to IORESULT.

Table A-10. Pascal 1.0 Entry Points Used by SSC

PASCAL 1.1 ENTRY POINTS

The Pascal 1.1 entry point protocol is outlined in the first section of this appendix. The values given here are the addresses of the routines. Unlike Pascal 1.0, Pascal 1.1 enters these routines using indirect addressing.
Addr. Offset for Value X Register Y Register A Register

$\text{CsO}D$ Initialization $(\text{Cs})\text{BE}$
On entry $\text{Cs}$ $\text{sO}$ anything
On exit $\text{EO}$ $\text{sO}$ (changed)
Notes: $\text{CBO0}$ space is enabled. Firmware initializes SSC to default values plus SW1 and SW2 selections.

$\text{CsO}E$ Read $(\text{Cs})\text{94}$
On entry $\text{Cs}$ $\text{sO}$ anything
On exit error code $\text{Cs}$ char. in
Notes: $\text{CBO0}$ space is enabled. Character in from ACIA or keyboard is returned in the A Register.

$\text{CsO}F$ Write $(\text{Cn})\text{97}$
On entry $\text{Cs}$ $\text{sO}$ char. out
On exit error code $\text{Cs}$ (changed)
Notes: $\text{CBO0}$ space is enabled. The byte in the A Register is sent out through the ACIA.

$\text{CsI}0$ Status $(\text{Cs})\text{9A}$
On entry $\text{Cs}$ $\text{sO}$ request (0 or 1)
On exit error code $\text{sO}$ error code
Notes: $\text{CBO0}$ space is enabled. Request = 0 asks ACIA whether it is ready to transmit another byte; request = 1 asks ACIA whether it has an input character available. On exit, carry bit = 0 for Yes or 1 for No.

Table A-11. Pascal 1.1 Offsets Used by SSC

OTHER SPECIAL FIRMWARE LOCATIONS

The SSC firmware uses several other addresses for predefined purposes. Table A-12 lists these locations.

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{CsO}5$</td>
<td>$38$</td>
<td>Pascal serial/firmware card identifier (as well as BASIC input entry point)</td>
</tr>
<tr>
<td>$\text{CsO}7$</td>
<td>$18$</td>
<td>Pascal serial/firmware card identifier (as well as BASIC output entry point)</td>
</tr>
<tr>
<td>$\text{CsO}B$</td>
<td>$\text{01}$</td>
<td>Pascal 1.1 generic signature byte ($\text{01} = $firmware card)</td>
</tr>
<tr>
<td>$\text{CsO}C$</td>
<td>$31$</td>
<td>Pascal 1.1 Device Signature byte ($\text{31} = $serial or parallel I/O card #1)</td>
</tr>
<tr>
<td>$\text{CSI1}$</td>
<td>$85$</td>
<td>Pascal 1.1 optional routines flag (nonzero value = not supported)</td>
</tr>
<tr>
<td>$\text{CsFF}$</td>
<td>$\text{08}$</td>
<td>Firmware revision level</td>
</tr>
</tbody>
</table>

Table A-12. SSC Special Firmware Locations
0000:  2 ******************************
0000:  3 *
0000:  4 * APPLE II SSC FIRMWARE *
0000:  5 *
0000:  6 * BY LARRY KENYON *
0000:  7 * --JANUARY 1981-- *
0000:  8 * *
0000:  9 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
0000: 10 *
0000: 11 **********************************
0000: 12 *
0000: 13 * VARIABLE DEFINITIONS *
0000: 14 *
0000: 15 **********************************
0000: 16 *
0000: 17 * ZERO PAGE EQU *
0000: 18 **********************************
0024: 19 CH EQU $24 ;CURSOR HORIZONTAL POSITION
0026: 20 SLOT16 EQU $26 ;SAVE $NO TO FREE UP Y-REG
0027: 21 CHARACTER EQU $27 ;OUTPUT, SCREEN AND INPUT CHAR
0028: 22 BASL EQU $28 ;BASE SCREEN ADDRESS POINTER
0035: 23 ZPTEMP EQU $35 ;WORKHORSE TEMPORARY
002A: 24 ZPTMP1 EQU $2A ;WHEN ZPTEMP ISN'T ENOUGH
002B: 25 ZPTMP2 EQU $2B ;TEMPORARIES, TEMPORARIES!
0036: 26 CSWL EQU $36 ;CHAR OUT VECTOR
0037: 27 CSWH EQU $37
0038: 28 KSWL EQU $38 ;CHAR IN VECTOR
0039: 29 KSWH EQU $39
003C: 30 AIL EQU $3C ;BATCH MOVE POINTER
0045: 31 RNDEL EQU $4E ;RANDOM NUMBER SEED
004F: 32 RNDH EQU $4F
0000: 33 **********************************
0000: 34 * GENERAL EQUATES *
0000: 35 **********************************
0100: 36 STACK EQU $100 ;SYSTEM STACK BLOCK
0200: 37 INBUFF EQU $200 ;SYSTEM INPUT BUFFER
C000: 38 KBD EQU $C000 ;KEYBOARD INPUT
C010: 39 KDSTRB EQU $C010 ;KEYBOARD CLEAR
CFFF: 40 ROMSOFF EQU $CFFF ;DISABLES CO-RES. $C800 ROMS
0000: 41 **********************************
0000: 42 * SSC CARD ADDRESSES *
0000: 43 **********************************
C081: 44 DIPSW1 EQU $C081 ;(+SN) DIPSWITCH BLOCK 1
C082: 45 DIPSW2 EQU $C082 ;(+SN) DIPSWITCH BLOCK 2
C088: 46 TRDREG EQU $C088 ;(+SN) TRANSMIT DATA REG (WRITE)
C088: 47 RDREG EQU $C088 ;(+SN) READ DATA REG (READ)
C089: 48 STREG EQU $C089 ;(+SN) STATUS REGISTER (READ)
C090: 49 RDRC EQU $C089 ;(+SN) SOFTWARE RESET (WRITE)
C08A: 50 CMDDREG EQU $C08A ;(+SN) COMMAND REGISTER (R/W)
C08B: 51 CTLRREG EQU $C08B ;(+SN) CONTROL REGISTER (R/W)
0000:  53  *****************************************************
0000:  54  * BIT-> B7 B6 B5 B4 B3 B2 B1 B0
0000:  55  ***********************************************+
0000:  56  * DIPSW1 S1 S2 S3 S4 Z Z S5 S6 (LEFT DIPSWITCH)
0000:  57  *
0000:  58  * (S1-S4 USED FOR BAUD RATE, S5-S6 FOR FIRMWARE MODE)
0000:  59  *
0000:  60  * DIPSW2 S1 Z S2 Z S3 S4 S5 CTS (RIGHT DIPSWITCH)
0000:  61  *
0000:  62  * STREG INT DSR DCD TDR RDR OVR FE PE
0000:  63  *
0000:  64  * CTLREG STB << WL >> CK << BAUD RATE >>
0000:  65  *
0000:  66  * CMDREG <<PARITY >> ECH <<XMIT>> RE DTR
0000:  67  *
0000:  68  *****************************************************
0000:  69  *****************************************************
0000:  70  * SCREEN VARIABLES: PPC AND SIC MODES *
0000:  71  *****************************************************
0538:  72  CMDBYTE EQU $5PB-$SC0 ;HOLDS COMMAND CHARACTER (PPC & CIC)
0438:  73  HANDSHAKE EQU $4P8-$SC0 ;SIC PBA CHAR COUNTER FOR ETX/ACK
0438:  74  PARAMETER EQU $4F8-$SC0 ;ACCMULATOR FOR CMD PARAMETER
0488:  75  STATEFLG EQU $786-$SC0 ;
0000:  76  * B7=CR GEN ENB FLAG B6=APATER LC INPUT FLG
0000:  77  * B2=BO=COMMAND INTERPRETER STATES
0000:  78  * 0 0 0 0 IDLE
0000:  79  * 0 0 1 CMD CHAR RECEIVED
0000:  80  * 0 1 0 COLLECT <N> UNTIL CHAR THEN DO COMMAND
0000:  81  * 0 1 1 SKIP UNTIL SPACE, THEN GOTO STATE 4
0000:  82  * 1 0 0 E/D COMMANDS
0000:  83  * 1 0 1 UNUSED
0000:  84  * 1 1 0 WAIT UNTIL CR THEN SET STATE TO ZERO
0000:  85  * 1 1 1 WAIT UNTIL CR THEN DO PROC INDICATED BY PARM
0000:  86  *
0000:  87  * (B4=B0 DETERMINE ENQUIRE CHAR FOR PSA MODE)
0000:  88  *
0388:  89  DELAYFLG EQU $478-$SC0
0000:  90  * B7-B6=SCREEN TRANSLATION OPTIONS
0000:  91  * 0 0 LC->UC
0000:  92  * 0 1 NO TRANSLATION
0000:  93  * 1 0 LC->UC INVERSE
0000:  94  * 1 1 LC->UC, UC->UC INVERSE
0000:  95  * (1-3 WILL ALLOW LC CHARs TO PASS THRU MONITOR)
0000:  96  *
0000:  97  * B5=B4=CR DELAY 0 0 = NO DELAY
0000:  98  * B3=B2=LF DELAY 0 1 = 32 MILLISECONDS
0000:  99  * B1=BO=FF DELAY 1 0 = 1/4 SEC
0000:  100  * 1 1 = 2 SEC
0000:  101  *
0588:  102  STSBYTE EQU $678-$SC0 ;STATUS/IRESULT/INPUT BYTE
0638:  103  PWBYTE EQU $6P8-$SC0 ;PRINTER (FORMAT) WIDTH
0688:  104  COLDBYTE EQU $778-$SC0 ;COLUMN POSITION COUNTER
0738:  105  MISCFLG EQU $7F8-$SC0 ;
0000:  106  * B7=ECHO BIT
0000:  107  * B6=TABLING OPTION ENABLE
0000:  108  * B3=PP/CIC MODE B2=KEYBOARD ENB
0000:  109  * B1=B4=LINEFEED EAT B3=PASCAL/BASIC FLAG
0000:  110  *
112 ***************************************************************************
113 * TEMP SCREEN VARS (SLOT INDEPENDENT) *
114 ***************************************************************************
07F8: 115 MSLOT EQU $7F8 ;BUFFER FOR HI SLOT ADDR ($CN)
116 ***************************************************************************
0000: 117 * SCREEN VARIABLES: CIC MODE *
0000: 118 ***************************************************************************
0000: 119 *
0000: 120 * STATEFLG: B7=TERMINAL MODE FLAG
0000: 121 * B3-B5=CHAIN SLOT
0000: 122 *
0038: 123 CHNBYTE EQU $6P8-$CO ;CURRENT OUTPUT SCREEN ($CN00 ENTRY)
0000: 124 *
0000: 125 * B0-B7=CN00 ENTRY
0000: 126 *
0688: 127 BUFBYTE EQU $778-$CO ;BUFFER FOR ONE
0000: 128 * INPUT BYTE: HIGH BIT IS SET
0000: 129 * WHEN BUFFER IS FULL
0000: 130 *
0000: 131 * MISCLFLG: B6=TERM MODE SHIFT ENB
0000: 132 *
0000: 133 * OTHER SLOT VARIABLES AS DEFINED FOR PPC AND SIC MODES
0000: 134 *
0000: 135 ***************************************************************************
0000: 136 * MONITOR SUBROUTINES *
0000: 137 ***************************************************************************
FD8D: 138 COUT EQU $FD8D ;CHARACTER OUT (THRU CSW)
FE89: 139 SETKB0 EQU $FE89 ;SETS KSW TO APPLE KEYBOARD
FF58: 140 IORTS EQU $FF58 ;KNOWN "RTS" LOCATION
FCBA: 141 NXTA1 EQU $FCBA ;INCREMENT AH,L AND CMP TO A2H,L
FE93: 142 SETSCR EQU $FE93 ;SETS CSW TO APPLE SCREEN
FDP6: 143 VIDOUT EQU $FDP6 ;OUTPUT A CHAR TO APPLE SCREEN
0000: 144 CHN SSC.CN00
0000: 1 ***************************************************************************
0000: 2 *
0000: 3 * APPLE II SIC Firmware *
0000: 4 *
0000: 5 * BY LARRY KENYON *
0000: 6 *
0000: 7 * -JANUARY 1981- *******************
0000: 8 *
0000: 9 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
0000: 10 *
0000: 11 ***************************************************************************
0000: 12 *
0000: 13 * CN00 SPACE CODE *
0000: 14 *
0000: 15 ***************************************************************************
****** NEXT OBJECT FILE NAME IS SSC.DCLS.OBJ0
C700: 16 ORG $C700
C700: 17 *
C700:2C 58 FF 18 BINIT BIT IORTS ;SET THE V-FLAG
C703:70 0C 19 BVS BENTRY ;<ALWAYS>
C705:38 20 IENTRY SEC ;BASIC INPUT ENTRY
C706:90 21 DF8 $90 ;OPCODE FOR BCC
C707:18 22 OENTRY CLC ;BASIC OUTPUT ENTRY
C708:B8 23 CLV
C709:50 06 24 BVC BENTRY ;<ALWAYS> SKIP AROUND PASCAL 1.1 ENTRY

60 SUPER SERIAL CARD
C70B:01  25  DB $01  ;GENERIC SIGNATURE BYTE
C70C:31  26  DB $31  ;DEVICE SIGNATURE BYTE
C70D:8E  27  DB >PINIT
C70E:94  28  DB >PREAD
C70F:97  29  DB >PWRITE
C710:9A  30  DB >PSTATUS
C711:85  27  BENTRY STA CHARACTER
C713:86  35  STA ZPTEMP  ;INPUT BUFFER INDEX
C715:8A  33  STA TXA  ;SAVE X AND Y REGS ON STACK
C716:48  34  PHA
C717:98  35  STA TYA
C718:48  36  STA PHA
C719:08  37  STA PHP  ;SAVE ENTRY FLAGS
C71A:78  38  STA SEI  ;NO RUPTS DURING SLOT DETERMINATION
C71B:8D FF CF  39  STA ROMSOFF  ;SWITCH OUT OTHER $C800 ROMS
C71B:20  58 FF  40  JSR IORTS
C721:BA  41  TSX
C722:8D 00 01  42  LDA STACK,X  ;RECOVER $CN
C725:8D 58 07  43  STA MSLOT
C728:AA  44  TAX  ;X-REG WILL GENERALLY BE $CN
C729:0A  45  ASL A
C72A:0A  46  ASL A  ;DETERMINE $NO
C72B:0A  47  ASL A
C72C:0A  48  ASL A
C72D:85 26  49  STA SLOT16
C72F:88  50  TAY  ;Y-REG WILL GENERALLY BE $NO
C730:28  51  PLP  ;RESTORE RUPTS
C731:50 29  52  BVC NORMIO
C733:  53 *
C734:  54 * BASIC INITIALIZATION
C735:  55 *
C733:E8 38 05  56  ASL CMDBYTE,X  ;ALWAYS ENABLE COMMANDS
C736:5E 38 05  57  LSR CMDBYTE,X
C739:89 6A C0  58  LDA CMDREG,Y  ;JUST HAD A POWER-ON OR PROGRAM RESET?
C73C:29 1F  59  AND #$1F
C73D:00  60  BNE RINIT1
C740:A9 EF  61  LDA #$EF  ;IF SO, GO JOIN INIT IN PROGRESS
C742:20 05 C8  62  JSR INIT1
C745:  63 *
C745:E4 37  64  BINIT1 CPX CSWH
C747:1D 08  65  BNE FROMIN
C749:A9 07  66  LDA >ENTRY
C74A:C5 36  67  CMP CSWL  ;IF CSW IS ALREADY POINTING TO ENTRY,
C74D:F0 05  68  BNE FROMIN  ;THEN WE MUST HAVE COME FROM KSW
C74F:85 36  69  STA CSWL  ;OTHERWISE, SET CSW TO ENTRY
C751:18  70  JMP OUTM CLC  ;INDICATE WE ARE CALLED FOR OUTPUT
C752:90 08  71  BCC NORMIO  ;<ALWAYS>
C754:E4 39  72  FROMIN CPX KSWH  ;MAKE SURE KSW POINTS HERE
C756:0D F9  73  BNE FROMOUT  ;
C758:A9 05  74  LDA >ENTRY
C75A:85 38  75  STA KSWL  ;SET UP KSW (NOTE CARRY SET FROM CPX)
C75C:  76 *
C75C:  77 * BRANCH TO APPROPRIATE BASIC I/O ROUTINE
C75C:  78 *
C75C:BD 38 07  79  NORMIO LDA MISCPFLG,X  ;SEPARATE CIC MODE FROM OTHERS
C75F:29 02  80  AND #$02  ;NOT ZERO FOR CIC MODE
C761:08  81  PHP  ;SAVE CIC MODE INDICATION
C762:90 03  82  BCC BOUTPUT
C764:4C BF C8  83  JMP  BINPUT
C767:  84 *
C767:8D B8 04  85  BOUTPUT  LDA  STATEFLG,X ;CHECK FOR AFTER LOWERCASE INPUT
C76A:48  86  PHA
C76B:0A  87  ASL  A
C76C:10 0E  88  BPL  BOUTPUT1 ;SKIP IF NOT
C76E:A6  35  89  LDX  #ZTMP
C770:A5  27  90  LDA  CHARACTER
C772:09  20  91  ORA  #$20
C774:9D  00 02  92  STA  INBUFF,X ;RESTORE LOWERCASE IN BUFFER
C777:85  27  93  STA  CHARACTER ;AND FOR OUTPUT ECHO
C779:AE F8 07  94  LDX  MSLOT
C77C:68  95  BOUTPUT1 PLA
C77D:29 BF  96  AND  #$BF ;ZERO THE FLAG
C77F:9D B8 04  97  STA  STATEFLG,X
C782:28  98  PLP ;RETRIEVE CIC MODE INDICATION
C783:F0  06  99  BEQ  BOUTPUT2 ;BRANCH FOR PPC, SIC MODES
C785:20  63 CB  100  JSR  OUTPUT ;CIC MODE OUTPUT
C788:4C B5 C8  101  JMP  CICEXIT ;FINISH BY CHECKING FOR TERM MODE
C78B:  102 *
C78B:4C FC C8  103  BOUTPUT2 JMP SEROUT
C78E:  104 ************************************
C78E:  105 *
C78E:  106 * NEW PASCAL INTERFACE ENTRIES *
C78E:  107 *
C78E:  108 ************************************
C78E:20  00 C8  109  PINIT  JSR  PASCALINIT ;
C791:2A  00  110  LDX  #0 ;NO ERROR POSSIBLE
C793:60  111  RTS
C794:4C 9B C8  112  PREAD  JMP  PASCALREAD ;
C797:4C AA C9  113  PWRITE  JMP  PASCALWRITE ;
C79A:  114 *
C79A:  115 * NEW PASCAL STATUS REQUEST
C79A:  116 *
C79A:  117 * A-REG=0 -> READY FOR OUTPUT?
C79A:  118 * A-REG=1 -> HAS INPUT BEEN RECEIVED?
C79A:  119 *
C79A:4A  120  PSTATUS  LSR  A ;SAVE REQUEST TYPE IN CARRY
C79B:20 9B C9  121  JSR  PENTRY ;(PRESERVES CARRY)
C79B:80  08  122  BCS  PSTATIN
C7A0:20 F5 CA  123  JSR  SROUT ;READY FOR OUTPUT?
C7A3:F0  06  124  BEQ  PSTATUS2
C7A5:18  125  CLC
C7A6:90  03  126  BCC  PSTATUS2 ;CARRY CLEAR FOR NOT READY
C7AB:  127 *
C7AB:20 D2 CA  128  PSTATIN  JSR  SRIN ;SETS CARRY CORRECTLY
C7AB:3D B8 05  129  PSTATUS2  LDA  STBYTE,X ;GET ERROR FLAGS
C7AB:AA  130  TAX
C7AF:60  131  RTS
C7B0:  132 ************************************
C7B0:  133 * ROUTINE TO SEND A CHARACTER TO ANOTHER CARD *
C7B0:  134 ************************************
C7B0:A2  03  135  SENDCD  LDX  #3
C7B2:B5  36  136  SAVEHOOK  LDA  CSWL,X
C7B4:48  137  PHA
C7B5:CA  138  DEX
C7B6:10 FA  139  BPL  SAVEHOOK
C7BB:  140 *
C788:  141 * NOW PUT CARD ADDRESS IN HOOK
C788:  142 *
C788: AE F8 07 143 LDX MSLOT
C788: BD 38 06 144 LDA CHNBYTE,X
C789: 85 36 145 STA CSWL
C7C0: BD B8 04 146 LDA STATEFLG,X ;GET SLOT #
C7C3: 29 38 147 AND #$38
C7C5: 4A 148 LSR A
C7C6: 4A 149 LSR A
C7C7: 4A 150 LSR A
C7C8: 09 C0 151 ORA #$C0 ;FORM $CN
C7CA: 85 37 152 STA CSWH
C7CC:  153 *
C7CC:  154 * OUTPUT TO THE PERIPHERAL
C7CC:  155 *
C7CC:8A 156 TXA ;SAVE $CN
C7CD: 48 157 PHA
C7CE: A5 27 158 LDA CHARACTER
C7DD: 48 159 PHA
C7D1: 09 80 160 ORA #$80 ;80 COL BOARDS WANT HI-BIT ON
C7D3: 20 ED FD 161 JSR COUT
C7D6:  162 *
C7D6:  163 * NOW RESTORE EVERYTHING THE OTHER CARD MAY HAVE CLOBBERED
C7D6:  164 *
C7D6:68 165 PLA
C7D7: 85 27 166 STA CHARACTER
C7D9: 68 167 PLA
C7DA: 8D F8 07 168 STA MSLOT
C7DD: AA 169 TAX
C7DE: 0A 170 ASL A
C7DF: 0A 171 ASL A
C7E0: 0A 172 ASL A
C7E1: 0A 173 ASL A
C7E2: 85 26 174 STA SLOT16
C7E4: 8D PF CF 175 STA ROMSOFF
C7E7:  176 *
C7E7:  177 * PUT BACK CSWL INTO CHNBYTE
C7E7:  178 *
C7E7: A5 36 179 LDA CSWL
C7E9: 9D 38 06 180 STA CHNBYTE,X
C7EC:  181 *
C7EC: A2 00 182 LDX #0
C7EE: 68 183 RESTORHOOK PLA
C7EF: 95 36 184 STA CSWL,X
C7F1: E8 185 DNX
C7F2: E0 04 186 CPX #4
C7F4: 90 F8 187 BCC RESTORHOOK
C7F6:  188 *
C7F6: AE F8 07 189 LDX MSLOT
C7F9: 60 190 RTS
C7FA:  191 *
C7FA: C1 D0 D0 192 ASC "APPLE"
C7FD: C5 CC 193 DFB $8
C800:  194 *
C800: 196 CHN SSC.C800
C800: 1 ******************************************
C800: 2 *
C800: 3 * APPLE II SSC Firmware *
C800: 4 *
C800: 5 * by Larry Kenyon *
C800: 6 *
C800: 7 * -jANUARY 1981 - ******************************************
C800: 8 *
C800: 9 * (C) COPYRIGHT 1981 by Apple Computer, INC. *
C800: 10 *
C800: 11 *******************************************************
C800: 12 *
C800: 13 * c800 space: high level stuff *
C800: 14 *
C800: 15 *******************************************************
C800: 16 * pascal 1.0 init entry *
C800: 17 *******************************************************

----- next object file name is SSC.DCLS.OBJ1
C900: 18 ORG $C800
C900: 19 PASCALINIT JSR PENTRY ;PASCAL 1.0 INITIALIZATION ENTRY
C903:A9 16 20 LDA #$16 ;NO XOFF, ECHO, LF EAT, OR LF GEN
C905:48 21 INIT1 PHA ;GOES TO MISCFLG AFTER MODIFICATION
C906:A9 00 22 LDA #0
C908:9D B8 04 23 STA STATEFLG,X
C908:9D B8 03 24 STA DELAYFLG,X
C908:9D 3B 04 25 STA HANDSHCR,X
C911:9D B8 05 26 STA STBYTE,X
C914:9D 3B 06 27 STA PWBYTE,X
C917:9D B8 06 28 STA COLBYTE,X
C91A:B9 82 C0 29 LDA DIPS2,Y ;SET LF GEN OPTION FROM D2-S5
C91D:B5 2B 30 STA ZPTMP2 ;SAVE FOR LATER
C91F:4A 31 LSR A ;S5-> CARRY
C920:4A 32 LSR A ;IF S5=ON=0 THEN LEAVE MISCFLG ALONE
C921:90 04 33 BCC INIT1A
C923:68 34 PLA ;OTHERWISE, MAKE SURE LF GEN
C924:29 FE 35 AND #$FE ;ENABLE IS RESET
C926:48 36 PHA ;
C927:B8 37 INIT1A CLV ;V WILL BE CLEAR FOR CIC MODE
C928:B9 81 C0 38 LDA DIPS1,Y
C928:4A 39 LSR A ;SIC MODES SET CARRY
C92C:B0 07 40 BCS INIT2 ;BRANCH FOR SIC MODES
C92C:B4 41 LSR A
C92F:B0 0E 42 BCS INIT2B ;PPC MODE BRANCH
C931:A9 01 43 LDA #$01 ;CTRL-A
C933:D0 3D 44 BNE INIT5 ;<ALWAYS> CIC MODE BRANCH
C935: 45 *
C935:4A 46 INIT2 LSR A ;SET CARRY FOR PBA
C936:A9 03 47 LDA #$03 ;SET ETX AS DEFAULT INQUIRY CHAR
C936:B0 02 48 BCS INIT2A ;BRANCH FOR PBA
C938:A9 80 49 LDA #$80 ;FOR PB SET AUTO CR GEN
C93C:9D B8 04 50 INIT2A STA STATEFLG,X
C93F:2C 58 4F 51 INIT2B BIT IORT5 ;SET V-FLAG FOR PPC, SIC MODES
C942:A5 2B 52 LDA ZPTMP2
C944:29 20 53 AND #$20 ;SET CR DELAY
C946:49 20 54 EOR #$20 ;SO 1=ENB, 0=DISABLE
C948:9D B8 03 55 STA DELAYFLG,X ; FROM D2-S2
C94B: 56 *
C84D:70 0A 57 BV S INIT3 <ALWAYS> BRANCH AROUND PASCAL
C84D: 58 ****************************
C84D: 59 * PASCAL 1.0 READ ENTRY *
C84D: 60 * (MUST BE AT $C84D) *
C84D: 61 ****************************
C84D:20 9B C8 62 PREADO JSR PASCALREAD ;DO PASCAL 1.1 READ
C850:AE F8 07 63 LDX MSLOT ;MODIFY FOR 1.0
C853:9D B8 05 64 STA STBYTE,X ;CHARACTER READ
C856:60 65 RTS
C857: 66 ****************************
C857: 67 * NOW WHERE WERE WE?? *
C857: 68 ****************************
C857: 69 *
C857:A5 2B 70 INIT3 LDA ZPTMP2 ;PPC, SIC MIMES USE SWITCHES
C859:4A 71 LSR A ;TO SET WIDTH, CR DELAY
C85A:4A 72 LSR A
C85B:29 03 73 AND #$03
C85D:A8 74 TAY
C85E:F0 04 75 BEQ INIT4
C860:76 * 77 PLA ;RESET VIDEO ENABLE FOR WIDTH#40
C861:29 7F 78 AND #$7F
C863:48 79 PHA
C864: 80 *
C864:B9 A6 C9 81 INIT4 LDA FWDTHL,Y
C867:9D 38 06 82 STA FWDBYTE,X
C86A:A4 26 83 LDY SLOT16
C86C: 84 *
C86C:68 85 PLA ;CLEAR CIC BIT IN FUTURE MISCELL
C86D:29 95 86 AND #$95 ;(AND TABING, XOFF AND LF EAT BITS)
C86F:48 87 PHA
C870:A9 09 88 LDA #$09 ;CTRL-I
C872: 89 *
C872:9D 38 05 90 INIT5 STA CMDBYTE,X ;CMD ESC CHAR (IGNORED FOR SIC MIMES)
C875:68 91 PLA
C876:9D 38 07 92 STA MISCELL,X ;SET MISCELL FLAGS
C879: 93 *
C879: 94 * NOW FOR THE ACIA INITIALIZATION ROUTINE
C879: 95 *
C879:A5 2B 96 INITACIA LDA ZPTMP2 ;DIPSW2
C87B:48 97 PHA
C87C:29 A0 98 AND #$A0 ;DATA BIT OPTIONS FOR CIC MODE
C87E:50 02 99 BVC INITACIA1 ;BRANCH FOR CIC MODE
C880:29 80 100 AND #$80 ;8 DATA, 1 OR 2 STOP FOR SIC, PPC
C882:20 A1 CD 101 INITACIA1 JSR DATACMD1 ;SET CONTROL REG
C885:20 81 CD 102 JSR BAUDCMD1 ;SET DIPSWITCH BAUD RATE
C888:68 103 PLA
C889:29 0C 104 AND #$0C ;PARITY OPTIONS FOR CIC MODE
C88B:50 02 105 BVC INITACIA2 ;BRANCH FOR CIC MODE
C88D:A9 00 106 LDA #$0 ;DISABLE PARITY FOR SIC, PPC MODES
C88F:0A 107 INITACIA2 ASL A
C890:0A 108 ASL A
C891:0A 109 ASL A
C892:09 0B 110 ORA #$0B
C894:99 8A C0 111 STA CMDREG,Y
C897:B9 88 C0 112 LDA RDREG,Y ;THROW OUT THE STRANGE STUFF
C89A:60 113 RTS
C89B: 114 ****************************

FIRMWARE 65
C89B: 115 * PASCAL READ ROUTINE *
C89B: 116 **************************
C89B:20 9B C9 117 PASCALREAD JSR PENTRY ;SHARED BY BOTH PASCAL VERSIONS
C89B:20 AA C8 118 PASCALREAD1 JSR GETCHAR ;GET ACIA/KBD DATA
C8A1:29 7F 119 AND #$7F ;CLEAR HIGH BIT FOR PASCAL
C8A3:AC F8 07 120 PASEXIT LDY MLSLOT
C8A6:BE B8 05 121 LDX STBYTE,Y ;ERROR STATUS-> X-REG
C8A9:60 122 RTS
C8AA: 123 **************************
C8AA: 124 * GETCHAR ROUTINE WAITS FOR *
C8AA: 125 * THE NEXT CHAR FROM EITHER *
C8AA: 126 * THE ACIA OR KEYBOARD (IF *
C8AA: 127 * ENABLED). USED BY PASCAL *
C8AA: 128 * READ ROUTINE, XON WAIT, *
C8AA: 129 * AND ACK WAIT. DATA IS RE- *
C8AA: 130 * TURNED IN THE A-REGISTER *
C8AA: 131 **************************
C8AA:20 FF CA 132 GETCHAR JSR INPUT ;ACIA DATA?
C8AD:B0 05 133 BCS GETCHAR
C8AE:20 2C CC 134 JSR CKKBD ;KEYBOARD INPUT?
C8B2:90 F6 135 BCC GETCHAR
C8B4:60 136 GETCHAR1 RTS ;EXIT WHEN WE HAVE SOMETHING
C8B5: 137 *
C8B5: 138 CHN SSC.HILEV

66 SUPER SERIAL CARD
CBF4: F0 DA 60 BEQ BINEND ;SKIP IF NOT CIC MODE
CBF6: 20 D1 C9 61 JSR CKINPUT ;LOOK FOR INPUT STREAM SPECIAL CHAR
CBF9: 4C D0 CB 62 JMP BINEND ;
CBFC: 63 *****************************************************
C8FC: 64 * SIC, PPC BASIC OUTPUT ROUTINE *
C8FC: 65 *****************************************************
C8FC: 20 1A CB 66 SEROUT JSR CMDSQCK ;CHECK FOR A COMMAND SEQUENCE
C8PP: B0 B7 67 BCS BASEXIT ;BRANCH IF WE WERE IN COMMAND MODE
C901: A5 27 68 LDA CHARACTER ;SAVE CHAR ON STACK
C903: 48 69 PHA
C904: BD 38 07 70 LDA MISCPFL, X ;IF VIDEO OR TABLING ENABLED,
C907: 29 C0 71 AND #$C0 ; DON'T MESS WITH THE CURSOR
C909: D0 16 72 BNE TABCHECK
C90B: 73 *
C90B: A5 24 74 LDA CH ;CHECK FOR COMMA TABLING
C90D: PO 42 75 BEQ NOTAB ; IF CH=0, THERE WAS NO TAB OR COMMA
C90F: C9 08 76 CMP #$B ; INTEGER BASIC COMMA?
C911: FO 04 77 BEQ COMMA
C913: C9 10 78 CMP #$E ;APPLESOFT COMMA?
C915: D0 0A 79 BNE TABCHECK
C917: 09 PO 80 COMMA ORA #$FO
C919: 3D B8 06 81 AND COLBYTE, X ; SET COL TO PREVIOUS TAB
C91C: 18 82 CLC
C91D: 65 24 83 ADC CH ; THEN INCREMENT TO NEXT TAB
C91F: 85 24 84 STA CH
C921: 85 *
C921: 86 *
C921: BD B8 06 87 TABCHECK LDA COLBYTE, X
C924: C5 24 88 CMP CH ; IS TABLING NEEDED?
C926: F0 29 89 BEQ NOTAB ; IF EQUAL THEN NO TAB NEEDED
C928: A9 A0 90 LDA #$A0 ; SPACE FOR FORWARD TAB
C92A: 90 08 91 BCC TAB1
C92C: BD 38 07 92 LDA MISCPFL, X ; DON'T BACKSPACE UNLESS TABLING
C92F: 0A 93 BPL NOTAB
C932: A9 88 95 LDA #$88 ; BACKSPACE FOR BACKTAB
C934: 85 27 96 TAB1 STA CHARACTER
C936: 2C 58 FF 97 BIT IORTS ; SET V=1 TO INDICATE TABBLING
C939: 08 98 PHP ; SAVE TABBLING INDICATOR
C93A: 70 0C 99 BVS TAB2 ; <ALWAYS> AROUND BATCH MOVE ENTRY
C93C: EA 100 NOP
C93D: 101 *****************************************************
C93D: 102 * SHORT BATCH MOVE; *
C93D: 103 * LOCATE AT #$C93D FOR *
C93D: 104 * COMPATIBILITY WITH *
C93D: 105 * SIC P8 BLOCK MOVE. *
C93D: 106 *****************************************************
C93D: 2C 58 FF 107 BATCHIN BIT IORTS
C940: 50 108 DFB #$50 ; DUMMY BVC
C941: B8 109 BATCHOUT CLV ; V=0 FOR OUTPUT ENTRY
C942: AE F8 07 110 LDX MSLOT
C945: 4C EF C9 111 JMP BATCHIO
C948: 112 *****************************************************
C94B: 113 * BURP . . *
C94B: 114 *****************************************************
C94B: 20 B5 C9 115 TAB2 JSR ADJUST ; ADJUST COLUMN COUNT
C94B: 20 6B CB 116 JSR OUTPUT2 ; DON'T GO TO SCREEN WHEN TABBLING
C94E: 4C 68 CB 117 JMP FORCER ; SHARE SOME CODE. ..
C951: 118 *
C951: 119 NOTAB PLA
C952: 120
C953: 121 PHP ;SAVE 'NO TAB' INDICATION
C954: 122 NOTAB1 STA CHARACTER ;(FORCE CR REENTRY)
C954: 123 PHA
C957: 124 JSR OUTPUT1 ;ENTER AFTER CMD SEQ CHECK
C95A: 125 JSR ADJUST
C95D: 126 PLA
C95E: 127 EOR #$8D ;WAS IT A CR?
C960: 128 ASL A
C961: 129 BNE FORCECR
C963: 130 STA COLBYTE,X ;IF SO, RESET COLUMN TO 0
C966: 131 STA CH
C968: 132 *
C968: 133 FORCECR LDA STATEFLG,X ;FORCE CR DISABLED?
C968: 134 BPL SEREND
C96D: 135 LDA PWDBYTE,X ;FORCE CR IF LIMIT REACHED
C970: 136 BEQ SEREND ;(FOR P8 POKE COMPATIBILITY)
C972: 137 CLC
C973: 138 SBC COLBYTE,X
C976: 139 LDA #$8D
C978: 140 BCC NOTAB1 ;BRANCH TO FORCE CR
C97A: 141 *
C97A: 142 SEREND PLP
C97B: 143 BVS TABCHECK ;BRANCH IF TABBING
C97D: 144 *
C97D: 145 LDA MISCFLG,X ;DON'T MESS WITH CURSOR
C980: 146 BMI SEREND2 ; WHEN VIDEO IS ON
C982: 147 LDY COLBYTE,X
C985: 148 ASL A
C986: 149 BMI SETCH ;SET CH TO VALUE OF COL FOR TABBING
C988: 150 TIA
C989: 151 LDY #0
C98B: 152 SEC
C98C: 153 SBC PWDBYTE,X;
C98F: 154 CMP #$F8 ;WITHIN 8 CHAR OF PWIDTH?
C991: 155 BCC SETCH
C993: 156 ADC #$27 ;IF SO, ADJUST TO WITHIN 8 OF 40
C995: 157 TAY
C996: 158 SETCH STY CH
C998: 159 *
C998: 160 SEREND2 JMP BASICEXIT ;THAT'S ALL
C998: 161 *
C998: 162 ***************
C998: 163 * PASCAL ENTRY ROUTINE *
C998: 164 ***************
C99B: 165 PENTRY STX MSLOT
C99B: 166 STY SLOT16
C9AD: 167 LDA #0
C9A2: 168 STA STSBYTE,X
C9A5: 169 RTS
C9A6: 170 *
C9A6: 171 *******************
C9A6: 172 * SIC MODE PRINTER WIDTH TABLE *
C9A6: 173 *******************
C9A6: 174 PWDBYL DFB $29 ;40 COLUMNS
C9A7: 175 DFB $48 ;72 COLUMNS
C9A8:50  176  DFB  $50 ; 80 COLUMNS
C9A9:84  177  DFB  $84 ; 132 COLUMNS
C9AA:  178  ***************
C9AA:  179  * PASCAL WRITE ROUTINE *
C9AA:  180  * (DOUBLES AS PASCAL *)
C9AA:  181  * 1.0 ENTRY POINT *)
C9AA:  182  * -MUST BE AT $C9AA-*
C9AA:  183  ***************
C9AA:B5 27  184  PASCALWRITE STA CHARACTER
C9AC:20 9B C9  185  JSR  PENTRY
C9AF:20 63 C8  186  JSR  OUTPUT
C9B2:4C A3 C8  187  JMP  PASEXIT ; LOAD X-REG WITH ERROR BYTE & RTS
C9B5:  188 *
C9B5:  189  ***************
C9B5:  190  * COLUMN ADJUST ROUTINE *
C9B5:  191  * (PC, SIC MOWES ONLY) *
C9B5:  192  ********************
C9B5:A5 27  193  ADJUST LDA CHARACTER
C9B7:49 08  194  EOR  #$08 ; BACKSPACE?
C9B9:0A  195  ASL  A
C9BA:00 04  196  BEQ  DECRCOL ; IF SO, DECREMENT COLUMN
C9BC:49 EE  197  EOR  #$EE ; DELETE? ($FF, RUB)
C9BE:D0 09  198  BNE  CTRLTST
C9C0:DE B8 06  199  DECRCOL DEC COLBYTE,X ; DECREMENT COLUMN COUNT
C9C3:10 03  200  BPL  ADJRST
C9C5:9D B8 06  201  STA  COLBYTE,X ; DON'T ALLOW TO GO BELOW 0
C9C8:60  202  ADJRST  RTS
C9CA:00 00  203  CTRLTST CMP  #$C0 ; DON'T INCREMENT COLUMN COUNT FOR
C9CB:00 00  204  BCS  ADJRST ; CONTROL CHARACTERS
C9CD:FE B8 06  205  INC  COLBYTE,X
C9D0:60  206  RTS
C9D1:  207  ********************
C9D1:  208  * ROUTINE TO PROCESS SPECIAL INPUT CHARSES *
C9D1:  209  ********************
C9D1:BD 38 07  210  CKINPUT LDA  MISCLG,X
C9D4:29 08  211  AND  #$08 ; INPUT CTL CHARSES ENABLED?
C9D6:B0 16  212  BEQ  CIEND
C9DB:88 04  214  LDA  STATEFLAG,X
C9DB:A4 27  215  LDY  CHARACTER
C9DD:C0 94  216  CPY  #$94 ; CTI-,T?
C9DF:D0 04  217  BNE  CKINPUT1
C9E1:09 80  218  ORA  #$80 ; SET TERMINAL MODE
C9E3:D0 06  219  BNE  CKINPUT2 ; <ALWAYS>
C9E5:  220 *
C9E5:C0 92  221  CKINPUT1 CPY  #$92 ; CONTROL-R?
C9E7:D0 05  222  BNE  CIEND
C9E9:29 7F  223  AND  #$7F ; RESET TERMINAL MODE
C9EB:BD B8 04  224  CKINPUT2 STA  STATEFLAG,X
C9EE:60  225  CIEND  RTS
C9EF:  226 *
C9EF: 228  CHN  SSC_TERM
C9EF:  1 **********************
C9EF:  2 *
C9EF:  3 * APPLE II SSC FIRMWARE *
C9EF:  4 *
C9EF:  5 * BY LARRY KENYON *
C9EF:  6 *
C9EF:  7 * -APRIL 1981- ****************
C9EF:  8 *
C9EF:  9 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
C9EF: 10 *
C9EF: 11 **********************
C9EF: 12 * SHORT BLOCK MOVE *
C9EF: 13 **********************
C9EF:  8A 14 BATCHIO TIA
C9F0:0A 15 ASL A
C9F1:0A 16 ASL A
C9F2:0A 17 ASL A
C9F3:0A 18 ASL A
C9F4:85 26 19 STA SLOT16
C9F6:99 00 20 LDA #0
C9F7:99 05 21 STA STSBYTE,X ;ZERO ERROR INDICATION
C9FB:70 0F 22 BVS MOVIN
C9FD:  23 *
C9FE:0A  00 24 MOVOU LDY #0
C9F:1B  3C 25 LDA (AIL),Y ;GET BUFFER DATA
C9A0:1B  27 26 STA CHARACTER
CA03:20  02 CC 27 JSR ACIAOUT ;SEND IT OUT THE ACIA
CA06:20  BA FC 28 JSR NXTA1
CA09:9E  F2 29 BCC MOVOU
CA0B:60  30 30 RTS
CA0C:  31 *
CA0C:20  D2 CA 32 MOVIN JSR SRIN
CA0F:9E  90 PB 33 BCC MOVIN
CA11:88  00 C0 34 LDA RDR Reg,Y
CA14:0A  00 35 LDY #0
CA16:30  3C 36 STA (AIL),Y ;PUT ACIA DATA INTO BUFFER
CA18:20  BA FC 37 JSR NXTA1
CA1B:9E  EF 38 BCC MOVIN
CA1D:60  39 39 RTS
CA1E:  40 *
CA1E:  41 **********************
CA1E:  42 *
CA1E:  43 * TERMINAL MODE ROUTINES *
CA1E:  44 *
CA1E:  45 **********************
CA1E:BD  B8 04 46 CHECKTERM LDA STATEPLG,X ;HAVE WE ENTERED TERMINAL MODE?
CA21:10  31 47 BPL TERMINTS ;IF NOT, A SIMPLE RTS WILL 00...
CA23:  48 *
CA23:  49 * WE ENTER THE WORLD OF TERMINAL MODE
CA23:  50 *
CA23:A9  02 51 TERRMODE LDA #$02 ;START IN SHIFT-LOCK STATE
CA25:48  52 PHA ;SHIFT STATE IS SAVED ON STACK
CA26:A9  7F 53 LDA #$7F
CA28:20  E2 CD 54 JSR KCMD1 ;RESET ECHO (DEFAULT TO FULL DUP)
CA2B:  55 *
CA2B:44  24 56 TERNEXT LDY CH
CA2D:B1  28 57 LDA (BASL),Y
CA2F:85 27    58    STA CHARACTER ;SAVE SCREEN CHARACTER
CA31:A9 07    59    TERMEXT1 LDA #$S07 ;IMPLEMENT A FLASHING UNDERLINE
CA33:25 4F    60    AND RNDH ; FOR A CURSOR
CA35:DD 10    61    BNE TERMNEXT3
CA37:4A 24    62    LDY CH
CA39:A9 DF    63    LDA #$0F
CA3B:D1 28    64    CMP (BASL),Y ;IS UNDERLINE ON THE SCREEN?
CA3D:DD 02    65    BNE TERMNEXT2 ;IF NOT, PUT IT THERE
CA3F:A5 27    66    LDA CHARACTER ;OTHERWISE USE TRUE SCREEN CHAR
CA41:91 28    67    TERMEXT2 STA (BASL),Y
CA43:E6 4F    68    INC RNDH ;MAKE IT FLASH, BUT
CA45:E6 4F    69    INC RNDH ;NOT TOO SLOW AND NOT TOO FAST
CA47:    70
CA47:BD B8 04    71    TERMEXT3 LDA STATEFLG,X ;ARE WE STILL IN TERM MODE?
CA4A:30 09    72    BMI TERMACIAIN ;IF SO, GO CHECK ACIA
CA4C:    73
CA4C:20 11 CC    74    TERMEXIT JSR RESTORE ;ALWAYS REPLACE OUR CURSOR
CA4F:68    75    PLA ;CLEAN UP THE STACK
CA50:A9 8D    76    LDA #$8D ;RETURN A <CR> TO COVER UP
CA52:85 27    77    STA CHARACTER
CA54:60    78    TERMITS RTS
CA55:    79
CA55:20 FF CA    80    TERMACIAIN JSR INPUT ;ACIA INPUT?
CA58:90 0C    81    BCC TERMKBDMIN ;IF NOT, GO CHECK KEYBOARD
CA5A:20 11 CC    82    JSR RESTORE ;RESTORE CURSOR, INPUT->CHARACTER
CA5D:20 D1 C9    83    JSR CKINPUT ;CHECK FOR CTL-T, CTL-R
CA60:20 A3 CC    84    JSR SCREENOUT1 ;INPUT->SCREEN ALWAYS
CA63:4C 2B CA    85    JMP TERMNEXT ;
CA66:    86
CA66:20 3E CC    87    TERMKBDMIN JSR GETKBD ;KEYPRESS?
CA69:90 C6    88    BCC TERMNEXT1 ;SKIP IF NOT
CA6B:70 BE    89    BVS TERMNEXT ;BRANCH IF WE DID A KBD ESCAPE SEQ.
CA6D:BD 38 07    90    LDA MISCPLG,X ;SHIFTING ENABLED?
CA70:0A    91    ASL A
CA71:10 22    92    BPL TERMSEND1
CA73:68    93    PLA ;RECOVER TERMSSTATE
CA74:4B    94    TAY
CA75:A5 27    95    LDA CHARACTER
CA77:CO 01    96    CPY #$1 ;1 = SHIFT LETTERS, XLATED NUMBERS
CA79:F0 20    97    BEQ TERMCAP
CA7B:BD 34    98    BCS TERMCLOCK ;2 MEANS CAPS LOCK MODE
CA7D:    99
CA7D:C9 9B    100    TERMNORM CMP #$9B ;ESC?
CA7F:DD 06    101    BNE TERMLETTER
CA81:    102
CA81:C8    103    TERMINC INY ;INCREMENT STATE
CA82:98    104    TERMINC TYA
CA83:48    105    PHA ;PUT BACK ON STACK
CA84:4C 2B CA    106    JMP TERMNEXT
CA87:    107
CA87:C9 C1    108    TERMLETTER CMP #$C1 ;<A?
CA89:90 08    109    BCC TERMSEND
CA8B:C9 DB    110    CMP #$DB ;>Z?
CA8D:BD 04    111    BCS TERMSEND
CA8F:09 20    112    ORA #$20 ;IT'S A LETTER SO TRANSLATE TO LC
CA91:85 27    113    STA CHARACTER
CA93:    114
CA93:98    115    TERMSEND TYA
CA94:48  116  PHA ;PUT STATE BACK ON STACK
CA95:20  68  CB  117  TERMSEND1 JSR OUTPUT1 ;GO OUTPUT
CA98:4C  2B  CA  118  JMP TERMINEXT

CA98:  119 *
CA9B:C9  9B  120  TERMCAP CMP #$9B ;TWO ESCAPES?
CA9D:F0  E2  121  BEQ TERMINC
CA9F:C9  B0  122  CMP #$B0 ;<0?
CAA1:90  0A  123  BCC TERMCAP1
CAA3:C9  BB  124  CMP #$BB ;>COLON?
CAA5:B0  06  125  BCS TERMCAP1

CAA7:  126 *
CAA7:  127 * ESC <NUMBER> SO TRANSLATE INTO MISSING ASCII CHAR
CAA7:  128 *

CAA7:A8  129  TAY
CAAB:B9  09  CA  130  LDA TRANSLATE-$BO,Y
CAAB:85  27  131  STA CHARACTER
CAAD:A0  00  132  TERMCAP1 LDY #0 ;BACK TO STATE 0
CAAE:F0  E2  133  BEQ TERMSEND ;<ALWAYS>

CAB1:  134 *
CAB1:C9  9B  135  TERMLQCK CMP #$9B ;ESC?
CAB3:DO  DE  136  BNE TERMSEND
CAB5:A0  00  137  LDY #0
CAB7:F0  C9  138  BBEQ TERMINC1;<ALWAYS>

CAB9:  139 *
CAB9:  140 ***********************
CAB9:  141 * TRANSLATE TABLE *
CAB9:  142 ***********************
CAB9:9B  143 TRANSLATE DFB $9B ;ESC
CABB:9C  144  DFB $9C ;FS
CABB:9F  145  DFB $9F ;US
CABC:DB  146  DFB $DB ;LEFT BRACKET
CABD:DC  147  DFB $DC ;LEFT SLASH
CABE:DF  148  DFB $DF ;UNDERSCORE
CABF:FB  149  DFB $FB ;LEFT ENCLOSE
CAC0:FC  150  DFB $FC ;VERTICAL BAR
CAC1:FD  151  DFB $FD ;RIGHT ENCLOSE
CAC2:FE  152  DFB $FE ;TILDE
CAC3:FF  153  DFB $FF ;RUB
CAC4:  154 *
CAC4:  155  CHN SSC.CORE
C4C4: 2 **************************************************************
C4C4: 3 *
C4C4: 4 * APPLE II SSC FIRMWARE *
C4C4: 5 *
C4C4: 6 * BY LARRY KENYON *
C4C4: 7 *
C4C4: 8 * -JANUARY 1981- *
C4C4: 9 *
C4C4: 10 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
C4C4: 11 *
C4C4: 12 **************************************************************
C4C4: 13 *
C4C4: 14 * CORE SUBROUTINES *
C4C4: 15 *
C4C4: 16 **************************************************************
C4C4: 17 *
C4C4: 18 * GENERAL PURPOSE WAIT ROUTINE *
C4C4: 19 **************************************************************
C4C4: 20 *
C4C4: 21 * WAITMS WAITS FOR [A-REG] MILLISECONDS (256 IF A-REG=0)
C4C4: 22 *
C4C4:A2 CA 23 WAITMS LDX #202
C4C4:CA6:CA 24 WAITMS1 DEX ;<DON'T LET THIS LOOP CROSS A PAGE>
C4C4:CD7:D0 PD 25 BNE WAITMS1 ;5 MICROSECOND LOOP
C4C4:CA9:38 26 SEC
C4C4:CA:A9 01 27 SBC #01
C4C4:CD8:D0 P6 28 BNE WAITMS
C4C4:CAE:AE 08 29 LDX MLSLOT
C4C4:D1:60 30 RTS
C4C4:D2: 31 **************************************************************
C4C4:D2: 32 * ACIA STATUS REGISTER READ ROUTINES *
C4C4:D2: 33 **************************************************************
C4C4:D2: 34 *
C4C4:D2: 35 * SRSN USED TO CHECK ACIA INPUT STATUS
C4C4:D2: 36 *
C4C4:D2:A4 26 37 SRSN LDY SLOT16 ;SLOT16=$NO
C4C4:D4:B9 04 38 LDA STREG,Y
C4C4:D7:48 39 PHA
C4C4:D8:29 A 20 40 AND #$20 ;DCDP?
C4C4:D4:4A 41 LSR A ;AN ERROR IF NOT
C4C4:D4:4A 42 LSR A
C4C4:D8:85 35 43 STA ZPTEMP
C4C4:D8:68 44 PLA
C4C4:D8:29 OP 45 AND #$0F
C4C4:E1:C9 08 46 CMP #$08 ;SET CARRY IF RDR FULL, ELSE CLEAR
C4C4:E3:00 47 BCC SRIN1
C4C4:E5:29 07 48 AND #$07 ;PE, PE, OVR VALID ONLY WHEN RDR=1
C4C4:E7:00 02 49 BCS SRIN2 ;<ALWAYS>
C4C4:E9:A5 35 50 SRIN1 LDA ZPTEMP
C4C4:E9:05 35 51 SRIN2 ORA ZPTEMP ;GET DCD ERROR BIT
C4C4:EF:00 05 52 BEQ SRIN3 ;BRANCH IF NO ERRORS FOUND
C4C4:EF:20 20 53 ORA #$20 ;ELSE SET BIT 5 TO OFFSET FOR PASCAL
C4C4:F1:9D 88 05 54 STA STBYTE,X ;AND SAVE IN STATUS TEMP
C4C4:F4:60 55 SRIN3 RTS ;CY=1 MEANS DATA IS AVAILABLE
C4C4:F5: 56 *
C4C4:F5: 57 * SROUT CHECKS IF TOR IS EMPTY + HARDWARE HANDSHAKE IS OK
C4C4:F5: 58 *
C4C4:A2 A4 26 59 SROUT LDY SLOT16
CAF7:B9 89 C0 60  LDA  STREG,X
CAF7:A29 70 61  AND  #$70
CAF7:C9 10 62  CMP  #$10 ;EQU IF TDR EMPTY, ICD, DSR, & CTS
CAF7:60 63  RTS
CAF7:
CAF7:FF
CAF7: 64 *
CAF7: 65 ***********************
CAF7:
CAF7: 66 * GENERAL INPUT ROUTINE *
CAF7:
CAF7: 67 ***********************
CAF7:20 D2 CA 68  INPUT  JSR  SRIN
CB02:90 15 69  BCC  NOINPUT1

CB04: 70 *
CB04:B9 88 C0 71  LDA  RDRREG,Y ;GET THE ACIA INPUT
CB07:09 80 72  ORA  #$80 ;SET HI BIT FOR BASIC
CB09:C9 8A 73  CMP  #$8A ;LINEFEED?
CB0B:D0 09 74  BNE  INPUT2
CB0D: 75 *
CB0D:A8 76  TAY
CB0E:BD 38 07 77  LDA  MISCRFLG,X ;SEE IF WE SHOULD EAT IT
CB11:29 20 78  AND  #$20
CB13:00 03 79  BNE  NOINPUT ;IF SO, JUST KEEP IT A SECRET
CB15:98 80  TYA
CB16: 81 *
CB16:38 82  INPUT2  SEC ;INDICATE DATA
CB17:60 83  RTS
CB18: 84 *
CB18:18 85  NOINPUT  CLC ;CARRY CLEAR FOR NO INPUT
CB19:60 86  NOINPUT1  RTS
CB1A: 87 *
CB1A: 88 ***********************
CB1A: 89 * GENERAL OUTPUT ROUTINE *
CB1A: 90 ***********************
CB1A: 91 *
CB1A: 92 * START OF COMMAND CHECK ROUTINE
CB1A: 93 *
CB1A:A4 26 94  CMDSEQCK  LDY  SLOT16
CB1C:B9 81 C0 95  LDA  DIPSW1,Y
CB1F:4A 96  LSR  A
CB20:B0 36 97  BCS  NOCMD ;DON'T WORRY ABOUT CMD SEQ FOR SIC
CB22:BD B8 04 98  LDA  STATEFLG,X
CB25:29 07 99  AND  #$07 ;ARE WE IN A COMMAND SEQUENCE?
CB27:F0 05 100  BEQ  ESCCHECK
CB29:20 FC CD 101  JSR  CMDPROC ;IF SO, GOTO COMMAND CENTRAL
CB2C:38 102  SEC ;INDICATE COMMAND
CB2D:60 103  RTS
CB2E: 104 *
CB2F:A5 27 105  ESCCHECK  LDA  CHARACTER
CB30:29 7F 106  AND  #$7F ;IGNORE HIGH BIT
CB32:DD 38 05 107  CMP  CMDBYTE,X ;IS THIS BEGINNING OF A CMD SEQ?
CB35:D0 05 108  BNE  XOFFCK
CB37:FE B8 04 109  INC  STATEFLG,X ;START UP COMMAND MODES
CB3A:38 110  SEC ;INDICATE COMMAND
CB3B:60 111  RTS
CB3C: 112 *
CB3C:BD 38 07 113  XOFFCK  LDA  MISCRFLG,X ;IS XON ENABLED?
CB3F:29 08 114  AND  #$08
CB41:F0 15 115  BEQ  NOCMD ;SKIP THIS IF NOT
CB43: 116 *
CB43:20 FF CA 117  JSR  INPUT ;ANY INPUT?
CB46:90 10  118  BCC  NOCMD  ;IF NOT, GO OUTPUT
CB48:C9 93  119  CMP  #$93  ;IS IT AN XOFF?
CB4A:F0 0E  120  BEQ  XONWAIT  ;IF SO, GO WAIT FOR ANOTHER INPUT
CB4C:48  121  PHA
CB4D:BD 38 07  122  LDA  MISCLG,X  ;CIC MODE?
CB50:4A  123  LSR  A
CB51:4A  124  LSR  A
CB52:68  125  PLA
CB53:90 04  126  BCC  ANRTS
CB55:90 B8 06  127  STA  BUFBYTE,X  ;IF SO, WE HAVE A BUFFER
CB58:18  128  NOCMD  CLC  ;INDICATE NOT A CMD SEQ
CB59:60  129  ANRTS  RTS
CB5A:  130 *
CB5A:20 AA CB  131  XONWAIT  JSR  GETCHAR  ;GET ACIA/KBD DATA
CB5D:C9 91  132  CMP  #$91  ;IS IT AN XON?
CB5F:D0 9F  133  BNE  XONWAIT  ;IF NOT, WAIT
CB61:18  134  CLC  ;OTHERWISE, INDICATE NOT A CMD SEQ
CB62:60  135  RTS  ;AND RETURN
CB63:  136 ******************************************************************************
CB63:  137 * NOW THE OUTPUT ROUTINE YOU'VE BEEN WAITING FOR *
CB63:  138 ******************************************************************************
CB63:20 1A CB  139  OUTPUT  JSR  CMDSEQCK
CB66:BO 0F  140  BCS  ANRTS  ;DON'T OUTPUT COMMAND SEQUENCES
CB68:  141 *
CB68:20 9E CC  142  OUTPUT1  JSR  SCREENOUT
CB6B:  143 *
CB6B:A4 26  144  OUTPUT2  LDY  SLOT16
CB6D:B9 81 CO  145  LDA  DIPSW1,Y
CB70:4A  146  LSR  A
CB71:90 4E  147  BCC  OUTPUT3  ;SKIP ETX/ACK FOR NATIVE MODES
CB73:4A  148  LSR  A
CB74:90 4B  149  BCC  OUTPUT3  ;BRANCH IF NOT P8A EMULATION
CB76:  150 *
CB76:  151 ******************************************************************************
CB76:  152 * P8A ETX/ACK STUFF*
CB76:  153 ******************************************************************************
CB76:  154 * AFTER 148 CHARACTERS BUT NOT WITHIN AN ESCAPE SEQUENCE
CB76:  155 * OF UP TO 5 CHARACTERS, THE HANDSHAKE IS PERFORMED
CB76:  156 * (WILL DELAY UNTIL 'NOT ESC' AND THEN 4 MORE CHARS
CB76:  157 * OR UNTIL AN 'ESC')
CB76:  158 *
CB76:A5 27  159  P8AOUT1  LDA  CHARACTER  ;SAVE CHAR ON STACK
CB78:48  160  PHA
CB79:BD 38 04  161  LDA  HANDSHAKE,X  ;CHAR COUNT FOR BUFFER PULL
CB7C:C9 67  162  CMP  #$103  ;IF <103 THEN 153 CHAR IN BUFFER
CB7E:90 10  163  BCC  ETX
CB80:C9 6C  164  CMP  #$108  ;IF >=108 THEN LESS THAN 149 CHAR
CB82:B0 22  165  BCS  P8AOUT2  ;SO NO HANDSHAKE IS NEEDED YET
CB84:C9 6B  166  CMP  #$107  ;SETS CARRY IF 107 (149 SENT)
CB86:68  167  PLA
CB87:48  168  PHA
CB8B:49 9B  169  EOR  #$9B  ;ESC?
CB8A:29 7F  170  AND  #$7F  ;IGNORE HI-BIT
CB8C:0D 18  171  BNE  P8AOUT2  ;COUNT AS 1 OF 5 IF NOT 'ESC'
CB8E:B0 19  172  BCS  P8AOUT3  ;DON'T COUNT IF 149TH CHAR IS 'ESC'
CB90:  173 *
CB90:BD B8 04  174  ETX  LDA  STATELG,X  ;SEND QUERY CHAR TO PRINTER
CB93:29 1F  175  AND  #$1F  ;(DEFAULT IS ETX)
CB95:09 80 176 ORA $80
CB97:85 27 177 STA CHARACTER
CB99:20 02 CC 178 JSR ACTOUT
CB9C:20 AA C8 179 ACK JSR GETCHAR ;GET ACIA/KBD DATA
CB9F:49 86 180 EOR $86 ;ACK?
CHA1:DD 00 ED 181 BNE ETX ;IF NOT ACK, REPEAT HANDSHAKE
CBA3:9D 38 04 182 STA HANDSHKE,X ;INIT CHAR COUNT TO 255
CBA6: 183 *
CBA6:DE 38 04 184 P8AOUT2 DEC HANDSHKE,X
CBA9:6B 185 P8AOUT3 PLA ;GET REAL CHAR TO OUTPUT
CBAE:85 27 186 STA CHARACTER
CBAC:49 8D 187 EOR $8D ;IP CR AND CR DELAY MODE
CBAB:0A 188 ASL A
CBAD:D0 00 A 189 BNE P8AOUT4 ;THEN FAKE CHAR COUNT TO LESS THAN
CBB1:BD B8 03 190 LDA DELAYFLG,X ; 48 TO FORCE HANDSHAKE ON NEXT
CBB4:29 30 191 AND $30 ;CHARACTER OUT
CBB6:F0 03 192 BEQ P8AOUT4
CBB8:9D 38 04 193 STA HANDSHKE,X
CBBB: 194 *
CBBB:20 02 CC 195 P8AOUT4 JSR ACTOUT
CBBE:4C EA CB 196 JMP LFGEN ;(SKIP DELAYS)
CBC1: 197 ****************************************
CBC1: 198 * AND BACK TO NORMAL OUTPUT *
CBC1: 199 ****************************************
CBC1:20 02 CC 200 OUTPUT3 JSR ACTOUT ;OUTPUT THE CHARACTER
CBC4: 201 *
CBC4: 202 * NOW CHECK FOR CR, LF, AND FF DELAYS
CBC4: 203 *
CBC4:0A 204 ASL A
CBC5:A8 205 TAY
CBC6:BD B8 03 206 LDA DELAYFLG,X ;GET DELAY FLAGS
CBC9:C0 18 207 CPY $18 ;FORM FEED?
CBCB:F0 0C 208 BEQ OUTDLY1
CBCD:4A 209 LSR A
CBCE:4A 210 LSR A ;RIGHT JUSTIFY LF DELAY
CBCF:C0 14 211 CPY $14 ;LINE FEED?
CBD1:F0 06 212 BEQ OUTDLY1
CBD3:4A 213 LSR A
CBD4:4A 214 LSR A ;RIGHT JUSTIFY CR DELAY
CDB5:C0 1A 215 CPY $1A ;CARRIAGE RETURN?
CDB7:DO 25 216 BNE OUTPUTEND
CDB9:29 03 217 OUTDLY1 AND $03 ;JUST WANT LOWEST 2 BITS
CDBB:F0 0D 218 BEQ LFGEN ;NO DELAY INDICATED
CDBD:AB 219 TAY
CDBE:B9 FE CB 220 LDA DLTYBL-1,Y
CDBE:1A 221 TAY ;DELAY IN 32 MSEC INCREMENTS
CDBE:2A 222 OUTDLYLP LDA #32 ;
CDBF:20 C4 CA 223 JSR WAITMS
CDBF:88 224 DEY
CDBF:DO F8 225 BNE OUTDLYLP
CBEA: 226 *
CBEA: 227 * CHECK ON LF GENERATION OPTION
CBEA: 228 *
CBEA:A5 27 229 LFGEN LDA CHARACTER
CBEA:0A 230 ASL A
CBEA:C9 1A 231 CMP $1A ;CARRIAGE RETURN?
CBEF:DO OD 232 BNE OUTPUTEND
CBEF:BD 38 07 233 LDA MISCFLG,X ;IS LF GENERATE ENABLED?
CBF4:6A 234  ROR A
CBF5:90 07  235  BCC OUTPUTEND
CBF7:A9 8A  236  LDA #$8A
CBF9:85 27  237  STA CHARACTER ;LINE FEED
CBFB:4C 6B CB  238  JMP OUTPUT2 ;(DON'T ECHO IT)
CBFE:60  239  OUTPUTEND RTS
CBFF:  240 *
CBFF:01  241  DLYTBL  DFB $01 ;32 MSEC
CC00:08  242  DFB $08 ;1/4 SEC
CC01:40  243  DFB $40 ;2 SEC
CC02:  244  ***********************
CC02:  245  * ACIA OUTPUT ROUTINE *
CC02:  246  ***********************
CC02:20 P5 CA  247  ACIAOUT JSR SROUT ;READY FOR OUTPUT?
CC05:00 FB  248  BNE ACIAOUT
CC07:98  249  TYA
CC08:09 89  250  ORA #$89 ;prepare to address ACIA,
CC0A:A8  251  TAY ;causing 6502 false read to occur
CC0B:57 27  252  LDA CHARACTER ;ON PAGE $BF (AVOIDING RDR READ)
CC0D:9F FF BF  253  STA $BFFF,Y ;HERE YOU ARE ACIA
CC10:60  254  RTS
CC11:  255 *
CC11:  256  ***********************
CC11:  257  * RESTORE CURSOR (NOT FOR PASCAL) *
CC11:  258  * (A-REG SHOULD CONTAIN NEW CHAR) *
CC11:  259  ***********************
CC11:48  260  RESTORE PHA ;SAVE NEW CHARACTER
CC12:A4  261  LDY CH
CC14:A5 27  262  LDA CHARACTER ;OLD CHARACTER
CC16:91 28  263  STA (BASL),Y
CC18:68  264  PLA
CC19:  265 *
CC19:C9  266  CMP #$95 ;SCREEN PICK?
CC1B:D0  267  BNE RESTOREND
CC1D:A5  268  LDA CHARACTER ;IF SO, USE SCREEN CHAR
CC1F:C9  269  CMP #$20 ;INVERSE?
CC21:80  270  BCS RESTOREND
CC23:20 DF CC  271  JSR GETXLATE ;REVERSE THE TRANSLATION
CC26:59 DB CC  272  EOR REVMAK,Y
CC29:85  273  RESTORE END STA CHARACTER
CC2B:60  274  RTS
CC2C:  275 *
CC2C:  276  CHN SSC,UTIL
CC2C: 2 *********************************************
CC2C: 3 *
CC2C: 4 * APPLE II SSC FIRMWARE *
CC2C: 5 *
CC2C: 6 * BY LARRY KENYON *
CC2C: 7 *
CC2C: 8 * -JANUARY 1981- *
CC2C: 9 *
CC2C: 10 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
CC2C: 11 *
CC2C: 12 *********************************************
CC2C: 13 *
CC2C: 14 * UTILITY ROUTINES *
CC2C: 15 *
CC2C: 16 *********************************************
CC2C: 17 * PASCAL-BASIC KEYBOARD FETCH *
CC2C: 18 *********************************************
CC2C: 18 19 CKKB0D CLC ;RETURN CARRY CLEAR FOR NO DATA
CC2D:BD 38 07 20 LDA MISCRE, X
CC30:29 04 21 AND $504 ;ANSWER NO IF KEYBOARD IS DISABLED
CC32:F0 09 22 BEQ CKKB0DXIT
CC34: 23 *
CC34:AD 00 CO 24 CKKB0DXIT LDA KBD
CC37:10 04 25 BPL CKKB0DXIT
CC39:BD 10 CO 26 STA KDSTB
CC3C:38 27 SBC ;INDICATE DATA
CC3D:60 28 CKKB0DXIT RTS
CC3E: 29 *********************************************
CC3E: 29 30 * GET A CHAR FROM KEYSBOARD FOR BASIC ONLY *
CC3E: 31 *********************************************
CC3E: 31 32 GETKBD INC RNDL ;MIX UP RANDOM # SEED
CC3F:D0 02 33 BNE GETKBD1 ; FOR BASIC
CC42:E6 4F 34 INC RNDH
CC44:20 2C CC 35 GETKBD1 JSR CKKB0D ;KEYBOARD FETCH ROUTINE
CC47:B8 36 CLV ;INDICATE NO ESCAPE SEQUENCE
CC48:90 F3 37 SCC CKKB0DXIT ;EXIT IF NO KEY PRESS
CC4A:20 11 CC 38 JSR RESTORE ;DO BASIC CURSED DUTY
CC4D:29 7F 39 AND $57F
CC4F:DD 38 05 40 CMP CMDBYTE,X ;IS IT THE START OF A COMMAND?
CC52:D0 3D 41 BNE GETKB0DNE ;IF NOT, EXIT INDICATING DATA
CC54:A4 26 42 LDY SLOT16
CC56:B9 81 CO 43 LDA DPI5W1,Y ;ONLY DO CMD ESC FOR PPC, SIC MODES
CC59:4A 44 LSR A
CC5A:80 35 45 BCS GETKB0DNE
CC5C: 46 *********************************************
CC5C: 46 47 * KEYBOARD ESCAPE HANDLER *
CC5C: 48 *********************************************
CC5C: 48 49 GETKB0DNE 49 KBDSC LDA $5A ;FIRST PRINT A PROMPT
CC5E:B9 93 CC 50 PROMPT LOOP LDA PROMPTBL,Y
CC61:85 27 51 STA CHARACTER
CC63:98 52 TYA
CC64:48 53 PHA
CC65:20 A3 CC 54 JSR SCREENOUT1 ;ALWAYS SEND TO SCREEN
CC68:68 55 PLA
CC69:A8 56 TAY
CC6A:88 57 DEY
CC6B:10 F1 58 BPL PROMPT LOOP
CC6D: 59 *
CC6D:A9 01 60 LDA #1 ; START OUT IN COMMAND STATE 1
CC6F:20 7B CE 61 JSR SETOSTATE
CC72: 62 *
CC72:20 34 CC 63 GETCMD JSR CKKND1 ; WAIT FOR KEYBOARD CHARACTER
CC75:10 FB 64 SPL GETCMD
CC77:C9 88 65 CMP #$88 ; BACKSPACE?
CC79:F0 E1 66 BEQ KBDESC ; IF SO, THEN START OVER
CC7B:05 27 67 STA CHARACTER
CC7D: 68 *
CC7D:20 A3 CC 69 JSR SCREENOUT1
CC80:20 1A CB 70 JSR CMDSEQCK ; PUMP THRU CMD INTERPRETER
CC83: 71 *
CC83:BD B8 04 72 LDA STATEFLG.X ; ARE WE DONE?
CC86:29 07 73 AND #$07
CC88:D0 E8 74 SNE GETCMD ; IF NOT, GO AGAIN
CC8A: 75 *
CC8A:90 8D 76 LDA #$8D ; FORCE BACK A CARRIAGE RETURN
CC8C:85 27 77 STA CHARACTER
CC8E:2C 58 FF 78 BIT IORTS ; INDICATE THAT A CMD SEQ HAS OCCURRED
CC91:38 79 GETKBDONE SEC ; INDICATE SUCCESS
CC92:60 80 RTS
CC93: 81 *
CC93: 82 *
CC93:BA C3 D3 83 PROMPTBL ASC ":CSS ELPPA"
CC96:D3 A0 C5
CC99:CC D0 D0
CC9C:C1
CC9D:8D 84 DFB #$8D
CC9E: 85 *
CC9E: 86 ************************************************************
CC9E: 87 * ROUTINE TO PRINT A CHARACTER ON THE CURRENT DISPLAY *
CC9E: 88 ************************************************************
CC9E:BD 38 07 89 SCREENOUT LDA MISCFLG.X
CCA1:10 13 90 BPL NOUT ; IF SCREEN DISABLED
CCA3: 91 *
CCA3:BD 38 07 92 SCREENOUT1 LDA MISCFLG.X ; ENTRY AFTER ECHO CHECK
CCA6:29 02 93 AND #$02 ; IF IT ISN'T CIC MODE,
CCA8:F0 0D 94 BEQ ASCREEN ; ALWAYS USE THE APPLE SCREEN
CCA9:BD B8 04 95 LDA STATEFLG.X ; CURRENT SCREEN = APPLE SCREEN?
CCAD:29 38 96 AND #$38
CCAF:F0 06 97 BEQ ASCREEN ; SLOT 0 = APPLE SCREEN
CCB1: 98 *
CCB1:8A 99 TXA ; JUMP TO CN00 SPACE
CCB2:48 100 PHA
CCB3:A9 AF 101 LDA #$SENDCD-1 ; TO VECTOR TO THE PERIPHERAL
CCB5:48 102 PHA ; IN THE CHAIN SLOT
CCB6:60 103 NOUT RTS
CCB7: 104 *
CCB7: 105 * APPLE 40-COL SCREEN DRIVER
CCB7: 106 *
CCB7:20 DF CC 107 ASCREEN JSR GETXLATE ; GET THE TRANSLATE OPTIONS
CCBA:09 80 108 ORA #$80 ; SET HIGH BIT OF CHAR
CCBC:C9 E0 109 CMP #$E0 ; LOWERCASE?
CCBE:90 06 110 BCC TESTLETTER
CCCD:59 D3 CC 111 EOR LCMAKSY ; DO LOWERCASE TRIP
CCCE:4C F6 FD 112 TOSCREEN JMP VDOUT ; ALL REGS ARE PRESERVED
CCCF: 113 *
CCCF: 114 * IF UPPERCASE, WE ONLY MAP LETTERS
CCC6: 115 *
CCC6:C9 C1 116 TESTLETTER CMP #$C1 ;<A?
CCC8:90 F9 117 BCC TOSCREEN
CCC8:C9 DB 118 CMP #$DB ;>Z?
CCC8:B0 F5 119 BCS TOSCREEN
CCC8:S9 D7 CC 120 EOR UCMASK,Y
CCD1:90 F0 121 BCC TOSCREEN ;<ALWAYS>
CCD3: 122 *
CCD3: 123 * MASKS FOR CASE TRANSLATION
CCD3:20 00 E0 124 LCMASK DFB $20,$00,$00,$20
CCD6:20
CCD7:00 00 00 125 UCMASK DFB $00,$00,$00,$00
CCDA:C0
CCDB:00 00 E0 126 RVMASK DFB $00,$00,$00,$00
CCDE:C0
CCDF: 127 *
CCDF:BD B8 03 128 GETXLATE LDA DELAYFLG,X ;TRANSLATE OPTIONS IN B6-B7
CCE2:2A 129 ROL A
CCE3:2A 130 ROL A
CCE4:2A 131 ROL A
CCE5:29 03 132 AND #$03
CCE7:48 133 TAY
CCE8:45 27 134 LDA CHARACTER
CCEA:60 135 RTS
CCRB: 136 *

(listings continued on next page)
CCEB: 138 CHN SSC.CMD
CCEB: 1 **************************
CCEB: 2 *
CCEB: 3 * APPLE II SSC FIRMWARE *
CCEB: 4 *
CCEB: 5 * BY LARRY KENYON *
CCEB: 6 *
CCEB: 7 * -JANUARY 1981- *************
CCEB: 8 *
CCEB: 9 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
CCEB: 10 *
CCEB: 11 *********************************
CCEB: 12 *
CCEB: 13 * SSC COMMAND PROCESSOR *
CCEB: 14 *
CCEB: 15 *****************************************
CCEB: 16 ****************************************
CCEB: 17 * COMMAND TABLE (USED BY COMMAND PROCESSOR ROUTINE *)
CCEB: 18 ***************************************

CCEB: 42 CMDTBBL DFB $42 ;(REAK)
CCEC: 67 DFB $67 ;CIC PAS NS=7
CCED: 0 CDFB $4B >BREAKCMD-1
CCEE: 54 DFB $54 ;(TERMINAL)
CCEF: 47 DFB $47 ;CIC NS=7
CCFO: A6 DFB >TERMCMD-1
CCP1: 43 DFB $43 ;(C GENERATE)
CCP2: 87 DFB $87 ; PPC NS=7
CCP3: A6 DFB >TERMCMD-1
CCP4: 51 DFB $51 ;(QUIT)
CCP5: 47 DFB $47 ;CIC NS=7
CCP6: B8 DFB >QUITCMD-1
CCP7: F2 DFB $52 ;(RESET)
CCP8: C7 DFB $C7 ;CIC PPC NS=7
CCP9: AC DFB >RESETCMD-1
CCPA: 5A DFB $5A ;Z COMMAND
CCFB: E7 DFB $87 ;CIC PPC PAS NS=7
CCFC: F3 DFB >ZCMD-1
CCFD: 49 DFB $49 ;I COMMAND
CCFE: 90 DFB $90 ; PPC NS=0
CCFF: D3 DFB >ICMD-1
CD00: 4B DFB $4B ;K COMMAND
CD01: 90 DFB $90 ; PPC NS=0
CD02: DF DFB >KCMD-1
CD03: 43 *
CD03: 45 DFB $45 ;E(CHG)
CD04: 43 DFB $43 ;CIC NS=3
CD05: 80 DFB $80
CD06: 46 DFB $46 ;F(ROMKBND)
CD07: E3 DFB $E3 ;CIC PPC PAS NS=3
CD08: 04 DFB $04
CD09: 4C DFB $4C ;(F GENERATE)
CD0A: E3 DFB $E3 ;CIC PPC PAS NS=3
CD0B: 01 DFB $01
CD0C: 58 DFB $58 ;X(ONP)
CD0D: E3 DFB $E3 ;CIC PPC PAS NS=3
CD0E: 08 DFB $08
CD0F: 54 DFB $54 ;T(ABBING)
CD10: 83 DFB $83 ; PPC NS=3
CD49: A0 01 116  LDY #$1
CD4B: 3D B8 03 117  DELAYS AND DELAYFLG,X ;DON'T DISTURB THE OTHER FLAGS
CD4E: E5 2A 118  STAX ZTMP1
CD50: ED 38 04 119  LDA PARAMETER,X
CD53: 29 03 120  AND #$03 ;JUST USE TWO BITS
CD55: 18 121  CLC
CD56: 6A 122  ROR A ;ONCE FOR FUN
CD57: 2A 123  ROTATE ROL A ;CHANGE DIRECTIONS
CD58: 88 124  DEY
CD59: DD FC 125  BNE ROTATE ;PREPARE IT TO OR INTO THE FLAGS
CD5B: 126 *
CD5B: 05 2A 127  ORA ZTMP1
CD5D: 9D B8 03 128  STAX DELAYFLG,X
CD60: 60 129  RTS
CD61: 130 *
CD61: 29 07 131  SSLOTCMD AND #$7 ;SET SLOT COMMAND
CD63: 0A 132  ASL A
CD64: 0A 133  ASL A
CD65: 0A 134  ASL A
CD66: 85 2A 135  STA ZTMP1
CD68: 0A 136  ASL A
CD69: C5 26 137  CMP SLOT16 ;MAKE SURE WE DON'T SET IT
CD6B: F0 0F 138  BEQ SSLOTCMD1 ; TO OUR OWN SLOT
CD6D: BD B8 04 139  LDA STATEFLG,X
CD70: 29 C7 140  AND #$C7 ;PUT NEW SLOT NUMBER IN BITS 3-5
CD72: 05 2A 141  ORA ZTMP1 ; OF CMDBYTE,X
CD74: 9D B8 04 142  STA STATEFLG,X
CD77: A9 00 143  LDA #0 ;STORE ZERO INTO
CD79: 9D 38 06 144  STA CMDBYTE,X ;SLOT OFFSET (SET TO CNOO ENTRY)
CD7C: 60 145  SSLOTCMD1 RTS
CD7D: 146 *
CD7D: 29 0F 147  BAUDCMD AND #$OF ;SET NEW BAUD RATE
CD7F: D0 07 148  BNE BAUDCMD2
CD81: B9 81 C0 149  BAUDCMD1 LDA DIPSW1,Y ;ZERO PARM = RELOAD FROM SWITCHES
CD84: 4A 150  LSR A
CD85: 4A 151  LSR A
CD86: 4A 152  LSR A
CD87: 4A 153  LSR A
CD88: 09 10 154  BAUDCMD2 ORA #$10 ;SET INT. BAUD RATE GENERATOR
CD8A: 85 2A 155  STA ZTMP1
CD8C: A9 E0 156  LDA #$E0
CD8E: 85 2B 157  CTLREGSET STA ZTMP2
CD90: 89 BB C0 158  LDA CTLREG,Y
CD93: 25 2B 159  AND ZTMP2
CD95: 05 2A 160  ORA ZTMP1
CD97: 99 BB C0 161  STA CTLREG,Y
CD9A: 60 162  RTS
CD9B: 163 *
CD9B: 88 164  PARITYCMD DEY ;TRICK: SO CTLREG,Y ACTUALLY
CD9C: 165 * ADDRESSES THE COMMAND REG
CD9C: 166 *
CD9C: 0A 167  DATACMD ASL A ;SET NEW # OF DATA BITS
CD9D: 0A 168  ASL A
CD9E: 0A 169  ASL A
CD9F: 0A 170  ASL A
CDAD: 0A 171  ASL A
CDAE: 85 2A 172  DATACMD1 STA ZTMP1
CDA3: A9 1F 173  LDA #$1F

84 SUPER SERIAL CARD
CDA5:D0 E7 174 BNE CTRLREGSET ;<ALWAYS>
CDA7: E7 175 *
CDA7:1E B8 04 176 TERMCMD ASL STATEFLG,X ;SET TERMINAL MODE
CDA8:38 177 SEC
CDA8:B0 10 178 BCS QCMD1 ;<ALWAYS>
CDA9:1E 179 *
CDA9:99 89 C0 180 RESETCMD STA RESET,Y ;DROP RTS, DTR
CDB0:20 93 FE 181 JSR SJTSCR ;PR#0
CDB3:20 89 FE 182 JSR SETKB ;IN#0
CDB6:AE F8 07 183 LDX MELOT
CDB9:1E B8 04 184 QUITCMD ASL STATEFLG,X ;CLEAR TERMINAL MODE
CDBC:18 185 CLC
CDBD:7E B8 04 186 QCMD1 ROR STATEFLG,X
CDD0:60 187 RTS
CDD1: 188 *
CDD1:89 8A C0 189 BREAKCMD LDA CMDREG,Y ;SEND BREAK SIGNAL
CDD4:4B 190 PHA ;FOR 233 MILLISECONDS
CDD5:09 0C 191 ORA #$0C
CDD7:99 8A C0 192 STA CMDREG,Y
CDDA:A9 E9 193 LDA #$233 ;DELAY FOR 233 MICROSEC.
CDDC:2A C4 CA 194 JSR WAITMS
CDDF:68 195 PLA ;RESTORE OLD COMMAND REG CONTENTS
CDD0:99 8A C0 196 STA CMDREG,Y
CDD3:60 197 RTS
CDD4: 198 *
CDD4:A9 28 199 ICMCMD LDA #$28
CDD6:9D 38 06 200 STA FWDBYTE,X ;SET PRINTER WIDTH TO 40
CDD9:A9 80 201 LDA #$80
CDDB:1D 38 07 202 ORA MISCPFLG,X ;SET SCREEN ECHO
CDE0:D0 05 203 BNE KCMD2 ;<ALWAYS>
CDE0: 204 *
CDE0:A9 28 205 KCMD LDA #$FE ;RESET THE LF GENERATE FLAG
CDE2:3D 38 07 206 KCMD1 AND MISCPFLG,X
CDE5:9D 38 07 207 KCMD2 STA MISCPFLG,X
CDE8:60 208 RTS
CDE9: 209 *
CDE9:C9 28 210 NCMCMD CMP #40 ;>=40?,
CDEB:90 08 211 BCC ZCMDRTS ;IF NOT, JUST EXIT
CDED:9D 38 06 212 STA FWDBYTE,X ;SET NEW PRINTER WIDTH
CDF0:A9 3F 213 LDA #$3F ;DISABLE SCREEN, SET LISTING MODE
CDF2:D0 DF 214 BNE KCMD1 ;<ALWAYS>
CDF4: 215 *
CDF4:1E 38 05 216 ZCMDM ASL CMDBYTE,X ;DISABLE COMMAND RECOGNITION
CDF7:38 217 SEC
CDF8:7E 38 05 218 ROR CMDBYTE,X
CDFB:60 219 ZCMDRTS RTS
CDFC: 220 *
CDFC: 221 ***********************************************
CDFC: 222 *VECTOR ACCORDING TO COMMAND STATE*
CDFC: 223 ***********************************************
CDFC:A8 224 CMDPROC TAY ;A-REG=COMMAND STATE
CDFD:A5 27 225 LDA CHARACTER
CDFF:29 7F 226 AND #$7F
CE01: 227 *
CE01:C9 20 228 CMP #$20 ;SKIP SPACES FOR ALL MODES
CE03:D0 09 229 BNE CMDPROC2
CE05:C0 03 230 CPY #$3 ;EXCEPT MODE 3
CE07:F0 01 231 BEQ CMPROC1
CE09:60    RTS
CE0A:A9 04    CMDPROC1 LDA #S4
CE0C:D0 6D    BNE SETOSTATE ;<ALWAYS>
CE0E:CE0F 00    CMDPROC2 CMP #$0D ;CARRIAGE RETURN?
CE10:10 12    BNE CMDPROC4 ;
CE11:20 79 CE    JSR ZEROSTATE ;ABORT FOR STATUES 0-5, EXIT FOR 6,7
CE15:07 07    CPY #$07 ;IN STATE 7 WE VECTOR TO THE PROC
CE17:0A 01    BEQ CMDPROC3 ;
CE19:60    RTS ;OTHERWISE, JUST EXIT
CE1A:    42 *
CE1A:A9 CD    CMDPROC3 LDA #$CD ;ALL PROCES MUST START IN PAGE #$CD
CE1C:48    PHA
CE1D:BD 38 04    LDA PARAMETER,X
CE20:48    PHA
CE21:A4 26    LDY SLOT16 ;NEEDED BY BREAK CMD
CE23:60    RTS
CE24:    49 *
CE24:85 35    CMDPROC4 STA ZPTMP
CE26:A9 CE    LDA #$CE ;ALL ROUTINES MUST START
CE28:48    PHA ;IN PAGE #$CE
CE29:B9 30 CE    LDA STATETBL,Y
CE2C:48    PHA
CE2D:A5 35    LDA ZPTMP
CE2F:60    RTS ;RTS TO COMMAND PROCEDURE
CE30:    257 *
CE30:    258 * NOW THE STATE ROUTINES
CE30:    259 *
CE30:    260 **********************
CE30:    261 * STATE BRANCH TABLE *
CE30:    262 **********************
CE30:A7    STATETBL DFB >STATERR-1 ;BAD STATE
CE31:37    DBF >CSTATE1-1 ;<CMD> SEEN
CE32:61    DBF >CSTATE2-1 ;ACCUMULATE PARAMETER
CE33:89    DBF >CDONE-1 ;SKIDTILL SPACE
CE34:8A    DBF >CSTATE4-1 ;E/D SOMETHING
CE35:A7    DBF >STATERR-1 ;ILLEGAL STATE
CE36:89    DBF >CDONE-1 ;SKIDTILL CR
CE37:89    DBF >CDONE-1 ;SKIDTILL CR THEN DO CMD
CE38:    271 **********************
CE38:    272 * COMMAND STATE 1 *
CE38:    273 **********************
CE38:DD 38 05    CSTATE1 CMP CMDBYTE,X ;IS IT <CMD>?
CE3B:06    BNE CSTATE1A
CE3D:DE B8 04    DEC STATEFLG,X ;SET STATE BACK TO ZERO
CE40:4C 02 CC    JMP ACIAOUT ;OUTPUT <CMD> IF SO
CE43:    278 *
CE43:C9 30    CSTATE1A CMP #$30 ;>=0?
CE45:90 0D    BCC CSTATE1B
CE47:C9 3A    CMP #$3A ;<=9?
CE49:B0 00    BCS CSTATE1B
CE4B:29 0F    AND #$0F ;IT'S A NUMBER
CE4D:9D 38 04    STA PARAMETER,X
CE50:A9 02    LDA #2
CE52:DD 27    BNE SETOSTATE ;<ALWAYS> SET MODE 2 AND RETURN
CE54:    287 *
CE54:C9 20    CSTATE1B CMP #$20 ;IS IT A CONTROL CHAR?
CE56:BO 06    BCS CSTATE1C
CE58: 9D 38 05 290  STA CMDBYTE,X ;SET NEW COMMAND CHARACTER
CE58: 4C 79 CE 291  JMP ZEROSTATE ;RESET STATE TO ZERO
CE58: 292 *
CE58: A0 00 293  CSTATE1C LDY #0 ;USE COMMAND TABLE
CE58: F0 4D 294  BRC CMDSEARCH ;<ALWAYS>
CE58: 295  **************************************
CE58: 296  * COMMAND STATE 2: ACCUMULATE PARAMETER *
CE58: 297  **************************************
CE58: 49 30 298  CSTATE2 EOR $30 ;CONVERT $30-$39 TO 0-9
CE58: C9 0A 299  CMP #$A ;0-9?
CE58: 80 0D 300  BCS CSTATE2A
CE58: A0 0A 301  LDY #$A ;IT'S A NUMBER, SO ADD
CE58: 7D 38 04 302  ACCLOOP ADC PARAMETER,X ; IT TO 10*PARAMETER
CE58: 8B 33 303  DEY
CE58: D0 FA 304  BNE ACCLoop
CE58: 9D 38 04 305  STA PARAMETER,X
CE58: F0 15 306  BEQ CDONE ;<ALWAYS>
CE58: 307 *
CE58: A0 2E 308  CSTATE2A LDY #CMDBTL1-CMDBTYL ;USE COMMAND TABLE
CE58: D0 36 309  BNE CMDSEARCH ;<ALWAYS>
CE58: 310  **************************************
CE58: 311  * SET COMMAND STATE *
CE58: 312  **************************************
CE58: A9 00 313  ZEROSTATE LDA #0
CE58: 85 2A 314  SETOSTATE STA ZPTMP1
CE58: AE F8 07 315  LDX MSLLOT
CE58: BD B8 04 316  LDA STATEFLG,X
CE58: 29 F8 317  AND #$F8
CE58: 05 2A 318 ORA ZPTMP1
CE58: 7D B8 04 319  STA STATEFLG,X
CE58: 60 320  CDONE RTS
CE58: BB: 321  **************************************
CE58: BB: 322  * COMMAND STATE 4 (E/D) *
CE58: BB: 323  **************************************
CE58: A8 324  CSTATE4 TAY ;E/D -> Y-REG
CE58: B0 325  LDA PARAMETER,X
CE58: C0 44 326  CPY #$44 ;(ISABLE)?
CE58: F0 09 327  BRC CSTATE4A
CE58: C0 45 328  CPY #$45 ;(ENABLE)?
CE58: D0 11 329  BNE STATEBR ;IF NOT, IGNORE THIS COMMAND
CE58: 1D 38 07 330  ORA MISCFLG,X ;SET FLAG
CE58: DA 0D 331  BNE CSTATE4B ;<ALWAYS>
CE58: 49 FF 332  CSTATE4A EOR #$F ;INVERT FOR DISABLE
CE58: E3 38 07 333  AND MISCFLG,X ;RESET FLAG
CE58: 9D 38 07 334  CSTATE4B STA MISCFLG,X
CE58: A4: 335  **************************************
CE58: A4: 336  * ESCAPE TO STATE 6 *
CE58: A4: 337  **************************************
CE58: A9 06 338  SETSTATES LDA #6
CE58: D0 D3 339  BNE SETOSTATE ;<ALWAYS>
CE58: A9 20 340  STATEBR LDA #$32 ;CODE FOR BAD COMMAND
CE58: AD B8 05 341  STA STSBYTE,X
CE58: D0 F5 342  BNE SETSTATE6 ;<ALWAYS>
CE58: AF: 343  **************************************
CE58: AF: 344  * TABLE DRIVEN COMMAND PROCESSOR *
CE58: AF: 345  **************************************
CE58: B9 FB CC 346  CMDSEARCH LDA CMDTL,Y ;GET CANDIDATE CHARACTER
CE58: F0 F4 347  BEQ STATEBR ;A ZERO MARKS THE END OF A SUBTABLE
CEB4: C5 35 348 CMP ZPTMP1 ;MATCH?
CEB6: F0 05 349 BEQ CMDMATCH
CEB8: C8 350 INY
CEB9: C8 351 CMDSEARCH1 INY ;REENTRY FOR WRONG MODES
CEBA: C8 352 INY ;ENTRY LENGTH = 3
CEBB: D0 F2 353 BNE CMDSEARCH ;<ALWAYS>
CEBD: 354 *
CEBD: C8 355 CMDMATCH INY
CEBE: B9 EB CC 356 LDA CMDTBL,Y
CEC1: 85 2A 357 STA ZPTMP1
CEC3: 29 20 358 AND #$20 ;CHECK PASCAL ENABLE
CEC5: D0 07 359 BNE CMDMATCH1 ;IT'S ON SO DON'T CHECK P-BIT
CEC7: BD 38 07 360 LDA MISClgl,X ;OFF SO MAKE SURE
CECA: 29 10 361 AND #$10 ; THAT WE AREN'T IN PASCAL
CECC: D0 EB 362 BNE CMDSEARCH1 ;BRANCH IF WE ARE
CECE: 363 *
CECE: BD 38 07 364 CMDMATCH1 LDA MISClgl,X ;GET CIC/PPC BIT
CEDI: 4A 365 LSR A ;SHIFT CIC/PPC MODE BIT TO CARRY
CEDI: 4A 366 LSR A *
CED3: 24 2A 367 BIT ZPTMP1 ;PPC->N CIC->V
CED5: B0 04 368 BCS CMDMATCH2 ;BRANCH IF CIC MODE
CED7: 10 E0 369 BPL CMDSEARCH1 ;NOT OK FOR PPC
CED9: 30 02 370 BMI CMDEXEC ;AND OK
CEDB: 50 DC 371 CMDMATCH2 BVC CMDSEARCH1 ;NOT OK FOR CIC
CEDB: 372 *
CEDD: A5 2A 373 CMDEXEC LDA ZPTMP1 ;RETRIEVE TABLE MODE BYTE
CEDF: 48 374 PHA
CEEO: 29 07 375 AND #$07
CEE2: 20 7B CE 376 JSR SETSTATE ;SET NEXT STATE
CEE5: C8 377 INY
CEE6: 68 378 PLA
CEE7: 29 10 379 AND #$10 ;
CEE9: D0 07 380 BNE CMDEXEC1 ;IF BIT 4 IS SET, VECTOR TO ROUTINE
CEEB: B9 EB CC 381 LDA CMDTBL,Y
CEEC: 9D 38 04 382 STA PARAMETER,X
CEF1: 60 383 RTS
CEF2: 384 *
CEF2: A9 CD 385 CMDEXEC1 LDA #$3D ;ROUTINES MUST BE IN PAGE $CD
CEF4: 48 386 PHA
CEF5: B9 EB CC 387 LDA CMDTBL,Y
CEF8: 48 388 PHA
CEF9: A4 26 389 LDY SLOT16
CEFB: BD 38 04 390 LDA PARAMETER,X ;LOT OF ROUTINES NEED THIS
CEF8: 60 391 RTS
CEF9: 392 *
CEF9: 00 393 DFB $00

SYMBOL TABLE SORTED BY SYMBOL

3C A1L CEB6 ACCLoop CCO2 ACIAOUT ?CB9C ACK
C9C8 ADJRTS C985 ADJUST C859 ANRTS CCB7 ASCREEn
CB8 BASICEXIT 28 BASL ?C93D BATHIN C9EF BATHIO
?C941 BATHOUT CD7D BAUDCMD CD81 BAUDCMD1 CD88 BAUDCMD2
C711 BENTRY C8EF BINAcl C8EA BINAcl C8EB BINECl
C9D0 BINEND C74S BINIT ?C700 BINIT C8CB BINKBD
C8BF BINPUT C77C BOUTPUT C767 BOUTPUI C78B BOUTPUT2
CDB1 BRCMKMD 068B BUPBYTE C88A CDONE 24 CH
27 CHARACTER CA1E CHECKTERM 0638 CHNBYTE C885 CICEXIT
C9EB CIEND C9D1 CKINPUT C985 CKINPUT1 C9EB CKINPUT2
<table>
<thead>
<tr>
<th>CC3D</th>
<th>CKKBDIT</th>
<th>CC2C</th>
<th>CKKBD</th>
<th>CC34</th>
<th>CKKBD</th>
<th>0538</th>
<th>CMDBYTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEF2</td>
<td>CMDXEC1</td>
<td>CED0</td>
<td>CMDXEC</td>
<td>CEF5</td>
<td>CMDMATCH1</td>
<td>CEB0</td>
<td>CMDBATCH</td>
</tr>
<tr>
<td>CED8</td>
<td>CMDMATCH2</td>
<td>CEA0</td>
<td>CMDPRO1C</td>
<td>CER8</td>
<td>CMDPRO2</td>
<td>CRE1</td>
<td>CMDPRO3C</td>
</tr>
<tr>
<td>CED4</td>
<td>CMDPROC4</td>
<td>CDFC</td>
<td>CMDPRO</td>
<td>CEB4</td>
<td>CMDSEARCH</td>
<td>CEBF</td>
<td>CMDBSEARCH</td>
</tr>
<tr>
<td>CEB9</td>
<td>CMDSEARCH1</td>
<td>CB1A</td>
<td>CMDSEQCK</td>
<td>CD19</td>
<td>CMDBL1</td>
<td>CEBB</td>
<td>CMDBL</td>
</tr>
<tr>
<td>06B8</td>
<td>COLBYTE</td>
<td>C917</td>
<td>COMMA</td>
<td>FDED</td>
<td>COUT</td>
<td>CD3B</td>
<td>CRCMD</td>
</tr>
<tr>
<td>CEB3</td>
<td>CSTATE1A</td>
<td>C854</td>
<td>CSTATE1B</td>
<td>CE5E</td>
<td>CSTATE1C</td>
<td>CE38</td>
<td>CSTATE1A</td>
</tr>
<tr>
<td>CE75</td>
<td>CSTATE2A</td>
<td>C862</td>
<td>CSTATE2B</td>
<td>CE9C</td>
<td>CSTATE4A</td>
<td>CE41</td>
<td>CSTATE4B</td>
</tr>
<tr>
<td>CEB8</td>
<td>CSTATE4</td>
<td>37</td>
<td>CSWW</td>
<td>36</td>
<td>CSWL</td>
<td>C088</td>
<td>CTLEG</td>
</tr>
<tr>
<td>CED8</td>
<td>CTLAEGSET</td>
<td>C9C9</td>
<td>CTRTLST</td>
<td>CDA1</td>
<td>DATACMD1</td>
<td>C69C</td>
<td>DATACMD</td>
</tr>
<tr>
<td>C9C0</td>
<td>DECRCOL</td>
<td>03BB</td>
<td>DELAYFLG</td>
<td>CD48</td>
<td>DELAYST</td>
<td>C681</td>
<td>DIPSW1</td>
</tr>
<tr>
<td>C082</td>
<td>DIPSW2</td>
<td>CBFF</td>
<td>DLITBL</td>
<td>CB29</td>
<td>ESCHECK</td>
<td>CB90</td>
<td>ETX</td>
</tr>
<tr>
<td>C647</td>
<td>FCCMD</td>
<td>C968</td>
<td>FORCSCR</td>
<td>C754</td>
<td>FROMIN</td>
<td>C751</td>
<td>FROMOUT</td>
</tr>
<tr>
<td>CEB4</td>
<td>GETCHARG</td>
<td>C08A</td>
<td>GETCHAR</td>
<td>C672</td>
<td>GETCMD</td>
<td>CE38</td>
<td>GETCMD</td>
</tr>
<tr>
<td>CC44</td>
<td>GETKB1D</td>
<td>CC91</td>
<td>GETKBDONE</td>
<td>C6DF</td>
<td>GETXTATE</td>
<td>0438</td>
<td>HANDSHKE</td>
</tr>
<tr>
<td>CD04</td>
<td>ICMD</td>
<td>C705</td>
<td>INTENTRY</td>
<td>0200</td>
<td>INBUF</td>
<td>C828</td>
<td>INIT1</td>
</tr>
<tr>
<td>C987</td>
<td>INICIA1</td>
<td>C835</td>
<td>INTT2</td>
<td>C83C</td>
<td>INIT2A</td>
<td>C83F</td>
<td>INIT2B</td>
</tr>
<tr>
<td>08C7</td>
<td>INIT3</td>
<td>C964</td>
<td>INTT4</td>
<td>C972</td>
<td>INIT5</td>
<td>C979</td>
<td>INITIA1</td>
</tr>
<tr>
<td>C982</td>
<td>INICIA1</td>
<td>C88P</td>
<td>INITIACIA2</td>
<td>CB16</td>
<td>INPUT2</td>
<td>C9FF</td>
<td>INPUT</td>
</tr>
<tr>
<td>0F58</td>
<td>IOSRS</td>
<td>C010</td>
<td>KBDSTB</td>
<td>C600</td>
<td>KBD</td>
<td>CCSC</td>
<td>KBDESC</td>
</tr>
<tr>
<td>C082</td>
<td>KCMD1</td>
<td>C95</td>
<td>KCMD2</td>
<td>C80</td>
<td>KCMD</td>
<td>39</td>
<td>KSWH</td>
</tr>
<tr>
<td>38</td>
<td>KSWL</td>
<td>CCD3</td>
<td>LCMAK</td>
<td>CD41</td>
<td>LCMD</td>
<td>CBEA</td>
<td>LFCEN</td>
</tr>
<tr>
<td>0738</td>
<td>MISCEFL</td>
<td>CAOC</td>
<td>MOVIN</td>
<td>C9FD</td>
<td>MOVOUT</td>
<td>07F8</td>
<td>MSLOT</td>
</tr>
<tr>
<td>C6E9</td>
<td>NCMD</td>
<td>CB58</td>
<td>NOCMD</td>
<td>CB19</td>
<td>NOINPUT1</td>
<td>C618</td>
<td>NOINPUT</td>
</tr>
<tr>
<td>CC96</td>
<td>NOOUT</td>
<td>C75C</td>
<td>NORMIO</td>
<td>C954</td>
<td>NOTAB1</td>
<td>C951</td>
<td>NOTAB</td>
</tr>
<tr>
<td>PCB0</td>
<td>NZTA1</td>
<td>C707</td>
<td>ONENTRY</td>
<td>CB9D</td>
<td>OUTDL1</td>
<td>CB62</td>
<td>OUTDLYP</td>
</tr>
<tr>
<td>C6B8</td>
<td>OUTPUT1</td>
<td>CB6B</td>
<td>OUTPUT2</td>
<td>C663</td>
<td>OUTPUT</td>
<td>C8C1</td>
<td>OUTPUT3</td>
</tr>
<tr>
<td>CB8E</td>
<td>OUTPUTEND</td>
<td>5C76</td>
<td>PBAOUT1</td>
<td>CB86</td>
<td>PBAOUT2</td>
<td>CB9A</td>
<td>PBAOUT3</td>
</tr>
<tr>
<td>C8BB</td>
<td>PBAOUT4</td>
<td>0438</td>
<td>PARAMETER</td>
<td>CD9B</td>
<td>PARITYCMD</td>
<td>C800</td>
<td>PASCALINIT</td>
</tr>
<tr>
<td>?C9B8</td>
<td>PASCALREAD1</td>
<td>C89B</td>
<td>PASCALREAD</td>
<td>C99A</td>
<td>PASCALWRITE</td>
<td>C83A</td>
<td>PASEXIT</td>
</tr>
<tr>
<td>C99B</td>
<td>PENTRY</td>
<td>C7B7</td>
<td>PINT</td>
<td>?C8AD</td>
<td>PREAD</td>
<td>C794</td>
<td>READ</td>
</tr>
<tr>
<td>C9C3</td>
<td>PROMPTEL</td>
<td>CC5E</td>
<td>PROMLOOP</td>
<td>C7A8</td>
<td>PSTATIN</td>
<td>C79A</td>
<td>PSTATIS</td>
</tr>
<tr>
<td>C7A8</td>
<td>PSTATUS2</td>
<td>0638</td>
<td>PBYTE</td>
<td>C9AE</td>
<td>PDMBL</td>
<td>C797</td>
<td>PWRITE</td>
</tr>
<tr>
<td>CD80</td>
<td>QCMD1</td>
<td>CDR9</td>
<td>QICTBD</td>
<td>C608</td>
<td>Rabbit</td>
<td>C089</td>
<td>RESIST</td>
</tr>
<tr>
<td>CDAD</td>
<td>RESICTCMD</td>
<td>CC11</td>
<td>RESTORE</td>
<td>CC92</td>
<td>RESTOREN</td>
<td>C7EE</td>
<td>RESTROHOOK</td>
</tr>
<tr>
<td>CD08</td>
<td>RESTORERES</td>
<td>4F</td>
<td>RNDH</td>
<td>4F</td>
<td>RNDL</td>
<td>C6FF</td>
<td>RMOFF</td>
</tr>
<tr>
<td>CD57</td>
<td>ROTATE</td>
<td>C7B2</td>
<td>SAVEMAK</td>
<td>C9CE</td>
<td>SCREENOUT</td>
<td>C0A3</td>
<td>SCREENOUT1</td>
</tr>
<tr>
<td>C780</td>
<td>SENDC</td>
<td>C998</td>
<td>SEREND2</td>
<td>C97A</td>
<td>SEREND</td>
<td>C6FC</td>
<td>SEROUT</td>
</tr>
<tr>
<td>C996</td>
<td>SETCH</td>
<td>FE89</td>
<td>SETBD</td>
<td>C7EB</td>
<td>SETSTATE</td>
<td>FE93</td>
<td>SETSCR</td>
</tr>
<tr>
<td>CE44</td>
<td>SETSTATE6</td>
<td>26</td>
<td>SLOT16</td>
<td>CA99</td>
<td>SIGNAL3</td>
<td>CA99</td>
<td>SIGNR</td>
</tr>
<tr>
<td>08A2</td>
<td>SRIN</td>
<td>0438</td>
<td>TRANS</td>
<td>?C088</td>
<td>TOREG</td>
<td>CA55</td>
<td>TERMCIAIN</td>
</tr>
<tr>
<td>08A1</td>
<td>SRLD</td>
<td>0100</td>
<td>STACK</td>
<td>0408</td>
<td>STATEFL</td>
<td>CA8B</td>
<td>STATER</td>
</tr>
<tr>
<td>CE30</td>
<td>STATEBL</td>
<td>C089</td>
<td>STREG</td>
<td>058B</td>
<td>STBYTE</td>
<td>C934</td>
<td>TAB1</td>
</tr>
<tr>
<td>C948</td>
<td>TAB2</td>
<td>C921</td>
<td>TABCHECK</td>
<td>?C088</td>
<td>TOREG</td>
<td>CA55</td>
<td>TERRMCIAIN</td>
</tr>
<tr>
<td>CA6D</td>
<td>TERMPCAP</td>
<td>CAA8</td>
<td>TERMPCAP</td>
<td>CDA7</td>
<td>TERMCMD</td>
<td>CA9C</td>
<td>TERMEXIT</td>
</tr>
<tr>
<td>CA82</td>
<td>TERMIN1</td>
<td>CAA1</td>
<td>TERMIN</td>
<td>CA66</td>
<td>TERMBDI</td>
<td>CA97</td>
<td>TERRLETTER</td>
</tr>
<tr>
<td>CA84</td>
<td>TERMIN2</td>
<td>CA23</td>
<td>TERMODE</td>
<td>CA28</td>
<td>TERMINEXT</td>
<td>CA31</td>
<td>TEMERNEXT1</td>
</tr>
<tr>
<td>CA41</td>
<td>TERMEXT2</td>
<td>CA47</td>
<td>TERMINEXT3</td>
<td>?C7AD</td>
<td>TERMINORM</td>
<td>CA54</td>
<td>TERRMTRS</td>
</tr>
<tr>
<td>CA93</td>
<td>TERNSEND</td>
<td>CA9S</td>
<td>TERNSEND1</td>
<td>C66C</td>
<td>TESTLETTER</td>
<td>CC3</td>
<td>TOSCREEN</td>
</tr>
<tr>
<td>CD35</td>
<td>TRANCMND</td>
<td>CA97</td>
<td>TRANSLATE</td>
<td>C67D</td>
<td>UCMAK</td>
<td>FDF6</td>
<td>VIDOUT</td>
</tr>
<tr>
<td>CAC4</td>
<td>WAITMS</td>
<td>CA6C</td>
<td>WAITS1</td>
<td>C83C</td>
<td>XOFFCK</td>
<td>CB5A</td>
<td>XONNAT</td>
</tr>
<tr>
<td>CDF8</td>
<td>ZCMDSRTS</td>
<td>CDF4</td>
<td>ZCMD</td>
<td>CB79</td>
<td>ZEROSTATE</td>
<td>35</td>
<td>ZTEMP</td>
</tr>
</tbody>
</table>

**SYMBOL TABLE**

| 2A ZPTMP1 | 2B ZPTMP2 |

**SORTED BY ADDRESS**

<table>
<thead>
<tr>
<th>24 CH</th>
<th>26 SLOT16</th>
<th>27 CHARACTER</th>
<th>28 BASL</th>
</tr>
</thead>
<tbody>
<tr>
<td>2A ZPTMP1</td>
<td>2B ZPTMP2</td>
<td>35 ZTEMP</td>
<td>36 CSWL</td>
</tr>
<tr>
<td>37 CSWH</td>
<td>38 KSWL</td>
<td>39 KSWH</td>
<td>3C A1L</td>
</tr>
<tr>
<td>4E RNLD</td>
<td>4F RNHD</td>
<td>0100 STACK</td>
<td>0200 INBUF</td>
</tr>
<tr>
<td>0388 DELAYFLG</td>
<td>0438 HANDSHKE</td>
<td>0438 PARAMETER</td>
<td>0488 STATEFLG</td>
</tr>
</tbody>
</table>
APPENDIX B
APPLE INTERFACE CARD EMULATION

The SSC emulates both the P8 and the P8A versions of the Apple II Serial Interface Card (SIC), although the SSC is not completely POKE-compatible with either. In addition, the SSC supports several Apple II Communications Card and Parallel Card software commands.

OLD SERIAL INTERFACE CARD EMULATION

The SSC replaces the P8 and P8A versions of the Apple II Serial Interface Card (SIC) and it has two switch-selectable modes to emulate them, as explained below. However, because of firmware space limitations, the SSC does not support all functions of the older interface cards, and various POKE locations are different. This section explains these functional differences.

It is best to use Printer Mode rather than one of the emulation modes, except under these circumstances:

- if you have extensive existent applications that use PEEKs and POKEs to modify SIC operating characteristics

- if you need SIC P8A mode’s ETX/ACK (or other—character/ACK) handshaking capabilities

What the SSC does NOT support that the old SIC does:

- P8 SIC block moves

- baud rates other than the 15 listed in the various baud rate tables in this manual (ACIA hardware generates only those 15)

- data formats other than 5 - 8 data bits and 1, 1-1/2 or 2 stop bits (ACIA characteristic; other formats rarely used anyway)

- <ESC>U and <ESC>L commands for upper and lowercase (but SSC’s Translate command offers more options; POKEs also available)

- current-loop operation
To run the SSC in emulation of the old Apple II Serial Interface Card (SIC), prepare and install the SSC the same way as for Printer Mode (Chapters 1 and 2), with the following exceptions:

- Set mode switches SW1-5 ON and SW1-6 OFF to emulate the old SIC with a P8 ROM.
- Set mode switches SW1-5 OFF and SW1-6 OFF to emulate the old SIC with a P8A ROM.
- Install the SSC in whatever slot the old SIC was installed in for the application involved.
- Follow the instructions given in the next sections if the application program did PEEKs and POKEs.

**P8 EMULATION POKEs**

Changing SIC parameters was done either by setting the seven switches located on the card, or by POKEing the SIC slot RAM locations where this configuration data was stored. BASIC programs that talked through the old SIC may be used with the new SSC; however, if the program POKEs at these slot RAM locations, those POKEs must be changed to be compatible with the SSC's use of the RAM. The P8 and P8A ROMs differ slightly in their use of these RAM locations. Tables B-1 and B-2 show the transformation for P8 mode; additional differences for P8A mode are noted in the following section. Other POKE possibilities are described in Appendix A.

In the tables, the letter s stands for the slot number (1-7) in which the SSC is installed; the other letters are used as variables whose values are noted in the table (sometimes further down).

There is no claim that making these changes is simple. In fact, whenever possible it is best to use Printer Mode and its software commands to change SSC operating variables.

Here is an example of how to use the tables: let's say that the SSC is in slot #3. You want: a baud rate of 110; data format of 5 data bits and 2 stop bits, even parity; line width of 40 with video on, no automatic <LF> after <CR>; no translation of lowercase to uppercase; and no 1/4-second delay after <CR>. The PEEKs and POKEs:

POKE 49339, 243 \((49291 + 3*16; 3 + 240)\)
POKE 49338, 107 \((49290 + 3*16; p = 107)\)
POKE 2043, 132 \(\text{plug in magic number}\)
POKE 1147, 64 \(\text{plug in magic number}\)

The same thing in Printer Mode with appropriate switch settings is:

SW1-1 to SW1-7: ON ON OFF OFF OFF ON ON
SW2-1 to SW2-7: -- OFF ON OFF OFF OFF

92 SUPER SERIAL CARD
Then to set 5 data and 2 stop bits, use `<CTRL-I>7D<RETURN>`; for even parity, use `<CTRL-I>3P<RETURN>`; to leave lowercase alone, use `<CTRL-I>1T<RETURN>`. You can use commands to change baud rate, etc.

<table>
<thead>
<tr>
<th>Selection</th>
<th>SSC switches and settings</th>
<th>PEEKs and POKEs to use for P8 Serial Card</th>
<th>Super Serial Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>P8 Mode:</td>
<td>SW1-5 ON,</td>
<td>POKE 1144+s,r</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SW1-6 OFF</td>
<td>r = (not available)</td>
<td></td>
</tr>
<tr>
<td>P8A Mode:</td>
<td>SW1-5 OFF,</td>
<td>0 dec/$00 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SW1-6 OFF</td>
<td>176 dec/$80 hex</td>
<td></td>
</tr>
<tr>
<td>Baud Rate:</td>
<td>SW1-1 to SW1-4</td>
<td>144 dec/$90 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td>same as Printer Mode</td>
<td>128 dec/$80 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>64 dec/$40 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>32 dec/$20 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 dec/$10 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 dec/$8 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 dec/$4 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 dec/$2 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 dec/$1 hex</td>
<td></td>
</tr>
<tr>
<td>Data Format:</td>
<td>SW2-1 ON</td>
<td>POKE 1912+s,r</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(to get r above, add d to b) d =</td>
<td></td>
</tr>
<tr>
<td>8 data, 1 stop</td>
<td></td>
<td>8 dec/$80 hex</td>
<td></td>
</tr>
<tr>
<td>7 data, 1 stop</td>
<td></td>
<td>6 dec/$60 hex</td>
<td></td>
</tr>
<tr>
<td>6 data, 1 stop</td>
<td></td>
<td>4 dec/$40 hex</td>
<td></td>
</tr>
<tr>
<td>5 data, 1 stop</td>
<td></td>
<td>(not available)</td>
<td></td>
</tr>
<tr>
<td>8 data, 2 stop</td>
<td>SW2-1 OFF</td>
<td>POKE 1272+s,t</td>
<td></td>
</tr>
<tr>
<td>7 data, 2 stop</td>
<td></td>
<td>r = 9; t = 1*</td>
<td></td>
</tr>
<tr>
<td>6 data, 2 stop</td>
<td></td>
<td>r = 8; t = 1*</td>
<td></td>
</tr>
<tr>
<td>5 data, 2 stop</td>
<td></td>
<td>r = 7; t = 1*</td>
<td></td>
</tr>
<tr>
<td>Parity:</td>
<td></td>
<td>r = 6; t = 1*</td>
<td></td>
</tr>
<tr>
<td>none</td>
<td></td>
<td>r = 9; t = 2*</td>
<td></td>
</tr>
<tr>
<td>odd</td>
<td></td>
<td>r = 8; t = 2*</td>
<td></td>
</tr>
<tr>
<td>even</td>
<td></td>
<td>r = 7; t = 2*</td>
<td></td>
</tr>
<tr>
<td>MARK</td>
<td></td>
<td>r = 6; t = 2*</td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td></td>
<td>* add l if</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p = 1 or 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>POKE 49291+s*16,r</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>r = b + d; b =</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 dec/$61 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 dec/$62 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 dec/$63 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 dec/$64 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 dec/$65 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 dec/$66 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 dec/$67 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 dec/$68 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 dec/$69 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 dec/$6A hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 dec/$6B hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>12 dec/$6C hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>13 dec/$6D hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14 dec/$6E hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15 dec/$6F hex</td>
<td></td>
</tr>
</tbody>
</table>

Table B-1. SIC Switch Settings, PEEKs and POKEs, Part I
Table B-2. SIC Switch Settings, PEEKs and POKEs, Part II

### P8A EMULATION POKEs

The P8A ROM differs from the P8 ROM in several ways:

1) The <CR> delay switch now determines whether an ETX/ACK handshake is performed after each <CR> that is transmitted. The corresponding RAM bit was not the same as the P8 <CR> delay bit, but was kept in bit 2 of location 1400+s. For SSC emulation, the control is the same as the <CR> delay bit as noted above (in location 1144+s).

2) The number of stop bits was always 2; for SSC P8A mode this is configured via switch SW2-1 and can also be set via software by POKEing location 4929 as noted above.

3) The printer width information was kept in the same location that the P8 ROM kept the number of stop bits; the P8 printer width byte was zeroed to avoid automatic generation of carriage returns. The SSC P8A emulation code keeps the printer width information in the
same place as for P8 emulation and uses the high-order bit at location $1400+s to control automatic generation of carriage returns.

4) Lowercase input is enabled by default for the P8A ROM; in P8A emulation, however, it is enabled by the POKE shown in Table B-2.

5) In contrast to the P8 ROM, the P8A ROM and the SSC do not support batch moves.

6) The enquire character for the SIC P8A ROM was ETX (ASCII 3); for SSC P8A mode, this can be changed to another control character by a POKE to location $1400+s. For example, to change the enquire character to ENQ (ASCII 5), which is used by many RS-232 devices, use this POKE: POKE $1400+s, 5. Note that this also disables the automatic generation of carriage returns. Actually, any character between $0 and $31 can be used, although only 3 and 5 are used much.

OTHER EMULATION MODE DIFFERENCES

If your old programs, written to control one of the old Serial Interface Card ROMs, still don’t work after you’ve followed all this handy advice, then read on.

The SSC always monitors the RS-232-C handshake lines to determine whether or not the device is ready to accept data. If your device fails to assert one of these lines, the SSC will wait patiently forever.

When the arrow on the jumper block is pointing toward TERMINAL, your device sees DCD and DSR asserted as soon as the SSC is initialized, and the SSC sees CTS whenever the device sends RTS. If the device does not assert both RTS and DTR, the SSC will assume it is not ready to receive data. This can be used as a hardware handshake to prevent buffer overflow at the device (e.g., when your printer runs out of paper it can stop asserting one of these lines and the SSC will wait while you put in more paper). If you do not connect these lines, the SSC will always treat them as if they were asserted.

The Serial Interface Card tied RTS to CTS, and DTR to DCD and DSR; if your RS-232 device depended upon this, you may want to make a special connector which does this.

Your device may have depended upon the half-duplex nature of the SIC. The ACIA on the SSC is able to send and receive at the same time and is always configured to do so.

The SIC was initialized each time it was called at location $CS00 (for example, by a PR#s or IN#s). The SSC is only reinitialized after the ACIA has been reset (either by resetting the Apple or by exiting from Printer or Communication Mode via a Reset command).
OLD COMMUNICATIONS CARD COMMANDS

The SSC supports all the functions supported by the old Apple II Communications Interface Card (CIC), although the two ACIAs' registers are not the same on a bit-by-bit level. The SSC also supports the CIC commands: <CTRL-T>, <CTRL-R>, and <CTRL-S>.

SWITCH TO TERMINAL MODE—<CTRL-T>

In Communication Mode, the SSC is initialized to recognize the remote-control command <CTRL-T> arriving in the stream of incoming data. This character causes the SSC to enter Terminal Mode (the same as the T erminal command (Chapter 3). You can disable <CTRL-T> recognition by issuing an X(OFF D(isable command.

BYPASS TERMINAL MODE—<CTRL-R>

When the SSC is in Terminal Mode and X(OFF E(nable (the default in this mode) is in effect, the SSC recognizes the remote control command <CTRL-R> arriving in the input data stream, and responds by bypassing (exiting from) Terminal Mode. This is the same as the Q uit Terminal Mode command (Chapter 3).

XOFF—<CTRL-S>

The SSC interprets <CTRL-S> as the ASCII XOFF character. When it receives <CTRL-S> from a remote device, it stops transmitting data until it receives an XON character from that device.

PARALLEL CARD COMMANDS

The SSC is not hardware compatible with the Apple II Parallel Cards. However, for the sake of compatibility with software written for parallel interface applications, the SSC supports the following commands. You do not need to follow these commands with <RETURN>.

LINE WIDTH n AND VIDEO OFF—<CTRL-I><n><N>

This command turns off the Apple II video screen and generates a <CR> after n characters (if automatic <CR> generation is enabled via the C command (Chapter 2); n can be any value from 40 through 255.

LINE WIDTH 40 AND VIDEO ON—<CTRL-I>1

This command turns on the Apple II video screen and sets the line width to 40.

DISABLE AUTOMATIC LINEFEED—<CTRL-I>K

This command has the same effect as L(inefeed D(isable (Chapter 2): it turns off automatic generation of <LF> after <CR>.
APPENDIX C

SPECIFICATIONS
AND SCHEMATICS

This appendix contains the SSC specifications, connector pin assignments, jumper block wiring, and a schematic diagram. Use the schematic diagram with the Theory of Operation section in Chapter 4.

SSC SPECIFICATIONS

PHYSICAL CHARACTERISTICS

Dimensions 2-3/4" x 7" (68.8 mm x 177.8 mm)
Weight 3 oz. (90 gm), approximately
Cables required internal cable from 10-pin header on SSC to DB-25 connector on case of Apple II (supplied); shielded RS-232-C cable to external device (not supplied)
Controls 2 blocks of 7 switches each, set by user before installation
Special Tools none required

ENVIRONMENT

Operating temperature 40°F to 95°F (5°C to 35°C)
Storage temperature -40°F to 122°F (-40°C to 50°C)
Operating relative humidity 5% to 95% (noncondensing)
Storage relative humidity 5% to 95% (noncondensing)

SPECIAL CIRCUITS

SY6551 Asynchronous Communications Interface Adapter
2316 Read Only Memory (2,048 by 8 bits) with SSC firmware
The SSC has the usual power supply bypassing capacitors

SPECIFICATIONS AND SCHEMATICS 97
APPLE II SLOT LOCATION

BASIC programs
APPLESOFT programs
PASCAL programs

any slot except slot #0
any slot except slot #0
slot #1 for use with printer, etc.
slot #2 for use with modem
slot #3 for use with terminal

SOFTWARE COMPATIBILITY

The SSC is compatible with the following languages and operating systems:

<table>
<thead>
<tr>
<th>Integer BASIC</th>
<th>DOS 3.2</th>
<th>Pascal 1.0</th>
<th>6502 Assembler</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applesoft BASIC</td>
<td>DOS 3.3</td>
<td>Pascal 1.1</td>
<td></td>
</tr>
</tbody>
</table>

Under BASIC, input sent to the SSC at high baud rates may be lost, since the SSC can only buffer two characters at a time and BASIC may not be fast enough to read characters before they are overlaid.

In any software environment, characters may be lost when sent to the video screen in scrolling mode at greater than 300 baud. There are at least three solutions to this problem: lower the baud rate to 300 baud; reduce the scrolling window size (using 2 fewer lines already makes 1200 baud possible), or use an 80-column card with automatic hardware scrolling.

CONNECTOR PIN ASSIGNMENTS

Table C-1 lists the signals assigned to the connector pins on the 10-pin header at location 78 on the SSC, and the corresponding pins on the DB-25 connector that you attach to the back of the Apple II case.

<table>
<thead>
<tr>
<th>10-pin</th>
<th>DB-25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>Connector</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>19</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>

Table C-1. Connector Pin Assignments

98 SUPER SERIAL CARD
**JUMPER BLOCK WIRING**

Table C-2 lists the signals that the jumper block connects to the SSC when the arrow points toward the word MODEM and when it points toward the word TERMINAL. In the latter case, the jumper block acts as a modem eliminator.

Note that all RS-232-C signals on the SSC use negative-true logic; that is, they are true (asserted) at 0 volts and false at +5 volts.

<table>
<thead>
<tr>
<th>Signal at SSC</th>
<th>MODEM position (pin)</th>
<th>TERMINAL position (pin)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmit Data</td>
<td>Transmit Data (2)</td>
<td>Receive Data (3)</td>
</tr>
<tr>
<td>Receive Data</td>
<td>Receive Data (3)</td>
<td>Transmit Data (2)</td>
</tr>
<tr>
<td>Request To Send</td>
<td>Request To Send (4)</td>
<td>Data Carrier Detect (8)</td>
</tr>
<tr>
<td>Clear To Send</td>
<td>Clear To Send (5)</td>
<td>Data Carrier Detect (8)</td>
</tr>
<tr>
<td>Data Set Ready</td>
<td>Data Set Ready (6)</td>
<td>Data Terminal Ready (2Ø)</td>
</tr>
<tr>
<td>Data Terminal Ready</td>
<td>Data Term. Ready (2Ø)</td>
<td>Data Set Ready (6)</td>
</tr>
<tr>
<td>Data Carrier Detect</td>
<td>Data Carrier Detect (8)</td>
<td>Request To Send (4)</td>
</tr>
<tr>
<td>Data Carrier Detect</td>
<td>Data Carrier Detect (8)</td>
<td>Clear To Send (5)*</td>
</tr>
</tbody>
</table>

*When SW1-7 is OFF and SW2-7 is ON, the jumper block in the TERMINAL position connects Data Carrier Detect on the SSC to Secondary Clear To Send on the DB-25 connector.

Table C-2. Jumper Block Wiring
APPENDIX D
ASCII CODE TABLE

The table below shows the entire ASCII character set, and how to generate each character. Not all characters are available directly from the Apple II keyboard. However, in Terminal Mode (Chapter 3) you can generate all of the lowercase and special ASCII characters not accessible directly from the Apple II keyboard.

Here is how to interpret this table:

- The BINARY column has the 7-bit code for each ASCII character.
- The LOW DEC column gives the decimal equivalent of the 7-bit binary value. This value is the same if the binary code has 8 bits and the high-order bit is 0 (SPACE parity; Pascal).
- The LOW HEX column gives the corresponding hexadecimal value.
- The HI DEC column gives the decimal equivalent of the 7-bit binary value if a high-order bit equal to 1 is appended to it (MARK parity; BASIC); for example, 11001000 for the letter H.
- The HI HEX column gives the corresponding hexadecimal value.
- The ASCII CHAR column gives the ASCII character name.
- The INTERPRETATION column spells out the meaning of special symbols and abbreviations where necessary.
- The WHAT TO TYPE column indicates what keystrokes generate the ASCII character from the NORMAL (unaided) Apple II keyboard, and from the TERMINAL Mode (firmware assisted) keyboard. Characters not accessible are labeled "n/a."

The numbers between columns refer to footnotes.

- Angle brackets enclose the names of single keys (like <ESC> for the ESC key), or enclose keystrokes involving more than one key (like <CTRL-SHIFT-M>, which means "hold down CTRL and SHIFT while pressing M." But <ESC>9 means "type ESC, THEN type 9" because the 9 is outside the angle brackets.
To put the SSC in Terminal Mode, set SW1-5 and SW1-6 both ON; then use the T command or the remote-control <CTRL-T> command. When the SSC first enters Terminal Mode, the keyboard is locked in uppercase. Press <ESC> once for lowercase. This also prepares the SSC for the special <ESC>-plus-number keystrokes. Press <ESC> twice in a row to lock the keyboard in uppercase again.

<table>
<thead>
<tr>
<th>7-BIT</th>
<th>LOW</th>
<th>LOW</th>
<th>HI</th>
<th>HI</th>
<th>ASCII</th>
<th>INTERPRETATION</th>
<th>WHAT TO TYPE</th>
<th>NORMAL</th>
<th>TERMINAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>BINARY</td>
<td>DEC</td>
<td>HEX</td>
<td>DEC</td>
<td>HEX</td>
<td>CHAR</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000000</td>
<td>0</td>
<td>0</td>
<td>128</td>
<td>80</td>
<td>NUL</td>
<td>Blank (null)</td>
<td>&lt;CTRL-0&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000001</td>
<td>1</td>
<td>1</td>
<td>129</td>
<td>81</td>
<td>SOH</td>
<td>Start of Header</td>
<td>&lt;CTRL-A&gt; 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000100</td>
<td>2</td>
<td>2</td>
<td>130</td>
<td>82</td>
<td>STX</td>
<td>Start of Text</td>
<td>&lt;CTRL-B&gt; 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00000101</td>
<td>3</td>
<td>3</td>
<td>131</td>
<td>83</td>
<td>ETX</td>
<td>End of Text</td>
<td>&lt;CTRL-C&gt; 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001000</td>
<td>4</td>
<td>4</td>
<td>132</td>
<td>84</td>
<td>EOT</td>
<td>End of Transm.</td>
<td>&lt;CTRL-D&gt; 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001001</td>
<td>5</td>
<td>5</td>
<td>133</td>
<td>85</td>
<td>ENQ</td>
<td>Enquiry</td>
<td>&lt;CTRL-E&gt; 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001100</td>
<td>6</td>
<td>6</td>
<td>134</td>
<td>86</td>
<td>ACK</td>
<td>Acknowledge</td>
<td>&lt;CTRL-F&gt; 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00001101</td>
<td>7</td>
<td>7</td>
<td>135</td>
<td>87</td>
<td>BEL</td>
<td>Bell</td>
<td>&lt;CTRL-G&gt; 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00010000</td>
<td>8</td>
<td>8</td>
<td>136</td>
<td>88</td>
<td>BS</td>
<td>Backspace</td>
<td>&lt;CTRL-H&gt; 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00010001</td>
<td>9</td>
<td>9</td>
<td>137</td>
<td>89</td>
<td>HT</td>
<td>Horizontal Tab</td>
<td>&lt;CTRL-I&gt; 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00010100</td>
<td>10</td>
<td>A</td>
<td>138</td>
<td>90</td>
<td>LF</td>
<td>Linefeed</td>
<td>&lt;CTRL-J&gt; 10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00010101</td>
<td>11</td>
<td>B</td>
<td>139</td>
<td>91</td>
<td>VT</td>
<td>Vertical Tab</td>
<td>&lt;CTRL-K&gt; 11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00011000</td>
<td>12</td>
<td>C</td>
<td>140</td>
<td>92</td>
<td>FF</td>
<td>Form Feed</td>
<td>&lt;CTRL-L&gt; 12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00011001</td>
<td>13</td>
<td>D</td>
<td>141</td>
<td>93</td>
<td>CR</td>
<td>Carriage Return</td>
<td>&lt;CTRL-M&gt; 13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00011100</td>
<td>14</td>
<td>E</td>
<td>142</td>
<td>94</td>
<td>SO</td>
<td>Shift Out</td>
<td>&lt;CTRL-N&gt; 14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00011101</td>
<td>15</td>
<td>F</td>
<td>143</td>
<td>95</td>
<td>SI</td>
<td>Shift In</td>
<td>&lt;CTRL-O&gt; 15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00100000</td>
<td>16</td>
<td>10</td>
<td>144</td>
<td>96</td>
<td>DLE</td>
<td>Data Link Escape</td>
<td>&lt;CTRL-P&gt; 16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00100001</td>
<td>17</td>
<td>11</td>
<td>145</td>
<td>97</td>
<td>DC1</td>
<td>Device Control 1</td>
<td>&lt;CTRL-Q&gt; 17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00100100</td>
<td>18</td>
<td>12</td>
<td>146</td>
<td>98</td>
<td>DC2</td>
<td>Device Control 2</td>
<td>&lt;CTRL-R&gt; 18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00100101</td>
<td>19</td>
<td>13</td>
<td>147</td>
<td>99</td>
<td>DC3</td>
<td>Device Control 3</td>
<td>&lt;CTRL-S&gt; 19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101000</td>
<td>20</td>
<td>14</td>
<td>148</td>
<td>100</td>
<td>DC4</td>
<td>Device Control 4</td>
<td>&lt;CTRL-T&gt; 20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101001</td>
<td>21</td>
<td>15</td>
<td>149</td>
<td>101</td>
<td>NAK</td>
<td>Neg. Acknowledge</td>
<td>&lt;CTRL-U&gt; 21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101100</td>
<td>22</td>
<td>16</td>
<td>150</td>
<td>102</td>
<td>SYN</td>
<td>Synchronization</td>
<td>&lt;CTRL-V&gt; 22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00101101</td>
<td>23</td>
<td>17</td>
<td>151</td>
<td>103</td>
<td>ETB</td>
<td>End of Text Blk.</td>
<td>&lt;CTRL-W&gt; 23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00110000</td>
<td>24</td>
<td>18</td>
<td>152</td>
<td>104</td>
<td>CAN</td>
<td>Cancel</td>
<td>&lt;CTRL-X&gt; 24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00110001</td>
<td>25</td>
<td>19</td>
<td>153</td>
<td>105</td>
<td>EM</td>
<td>End of Medium</td>
<td>&lt;CTRL-Y&gt; 25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00110100</td>
<td>26</td>
<td>1A</td>
<td>154</td>
<td>106</td>
<td>SUB</td>
<td>Substitute</td>
<td>&lt;CTRL-Z&gt; 26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00110101</td>
<td>27</td>
<td>1B</td>
<td>155</td>
<td>107</td>
<td>ESC</td>
<td>Escape</td>
<td>&lt;ESC&gt; 13</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. Normal command character in Communication Mode.
2. Used in ETX/ACK protocol (SIC P8A Emulation Mode).
4. Used in ETX/ACK or ENQ/ACK protocol (SIC P8A Emulation Mode).
5. Or use ← key.
7. Or use <RETURN> key.
8. XON in XON/XOFF protocol (usually in Communication Mode).
9. Remote-control command to Exit from Terminal Mode.
10. XOFF in XON/XOFF protocol (usually in Communication Mode).
11. Remote-control command to Enter Terminal Mode.
12. Or use → key.
13. Use the ESC key to generate the Escape character with the normal Apple II keyboard. In Terminal Mode, use <ESC>⇧.
<table>
<thead>
<tr>
<th>7-BIT</th>
<th>LOW</th>
<th>LOW</th>
<th>HI</th>
<th>HI</th>
<th>ASCII</th>
<th>WHAT TO TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BINARY DEC</td>
<td>HEX</td>
<td>DEC</td>
<td>HEX</td>
<td>CHAR</td>
<td>INTERPRETATION</td>
<td>NORMAL</td>
</tr>
<tr>
<td>00111000</td>
<td>28</td>
<td>1C</td>
<td>156</td>
<td>9C</td>
<td>FS</td>
<td>File Separator n/a</td>
</tr>
<tr>
<td>00111101</td>
<td>29</td>
<td>1D</td>
<td>157</td>
<td>9D</td>
<td>GS</td>
<td>Group Separator &lt;CTRL-SHIFT-N&gt;</td>
</tr>
<tr>
<td>00111110</td>
<td>30</td>
<td>1E</td>
<td>158</td>
<td>9E</td>
<td>RS</td>
<td>Record Separator &lt;CTRL-SHIFT-N&gt;</td>
</tr>
<tr>
<td>00111111</td>
<td>31</td>
<td>1F</td>
<td>159</td>
<td>9F</td>
<td>US</td>
<td>Unit Separator n/a</td>
</tr>
<tr>
<td>01000000</td>
<td>32</td>
<td>20</td>
<td>160</td>
<td>A0</td>
<td>SP</td>
<td>Space</td>
</tr>
<tr>
<td>01000011</td>
<td>33</td>
<td>21</td>
<td>161</td>
<td>A1</td>
<td>&quot;</td>
<td>Closing Quote</td>
</tr>
<tr>
<td>01000110</td>
<td>34</td>
<td>22</td>
<td>162</td>
<td>A2</td>
<td>&quot;</td>
<td></td>
</tr>
<tr>
<td>01000111</td>
<td>35</td>
<td>23</td>
<td>163</td>
<td>A3</td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>01001000</td>
<td>36</td>
<td>24</td>
<td>164</td>
<td>A4</td>
<td>$</td>
<td></td>
</tr>
<tr>
<td>01001101</td>
<td>37</td>
<td>25</td>
<td>165</td>
<td>A5</td>
<td>%</td>
<td></td>
</tr>
<tr>
<td>01001110</td>
<td>38</td>
<td>26</td>
<td>166</td>
<td>A6</td>
<td>&amp;</td>
<td></td>
</tr>
<tr>
<td>01010011</td>
<td>39</td>
<td>27</td>
<td>167</td>
<td>A7</td>
<td>'</td>
<td>Closing Quote</td>
</tr>
<tr>
<td>01010100</td>
<td>40</td>
<td>28</td>
<td>168</td>
<td>A8</td>
<td>(</td>
<td></td>
</tr>
<tr>
<td>01010101</td>
<td>41</td>
<td>29</td>
<td>169</td>
<td>A9</td>
<td>)</td>
<td></td>
</tr>
<tr>
<td>01010110</td>
<td>42</td>
<td>2A</td>
<td>170</td>
<td>AA</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td>01010111</td>
<td>43</td>
<td>2B</td>
<td>171</td>
<td>AB</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>01011000</td>
<td>44</td>
<td>2C</td>
<td>172</td>
<td>AC</td>
<td>,</td>
<td>Comma</td>
</tr>
<tr>
<td>01011001</td>
<td>45</td>
<td>2D</td>
<td>173</td>
<td>AD</td>
<td>-</td>
<td>Hyphen</td>
</tr>
<tr>
<td>01011010</td>
<td>46</td>
<td>2E</td>
<td>174</td>
<td>AE</td>
<td>.</td>
<td>Period</td>
</tr>
<tr>
<td>01011011</td>
<td>47</td>
<td>2F</td>
<td>175</td>
<td>AF</td>
<td>/</td>
<td></td>
</tr>
<tr>
<td>01011000</td>
<td>48</td>
<td>30</td>
<td>176</td>
<td>B0</td>
<td>Ø</td>
<td>Ø</td>
</tr>
<tr>
<td>01011001</td>
<td>49</td>
<td>31</td>
<td>177</td>
<td>B1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>01011010</td>
<td>50</td>
<td>32</td>
<td>178</td>
<td>B2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>01011011</td>
<td>51</td>
<td>33</td>
<td>179</td>
<td>B3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>01011100</td>
<td>52</td>
<td>34</td>
<td>180</td>
<td>B4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>01011101</td>
<td>53</td>
<td>35</td>
<td>181</td>
<td>B5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>01011110</td>
<td>54</td>
<td>36</td>
<td>182</td>
<td>B6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>01011111</td>
<td>55</td>
<td>37</td>
<td>183</td>
<td>B7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>01100000</td>
<td>56</td>
<td>38</td>
<td>184</td>
<td>B8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>01100001</td>
<td>57</td>
<td>39</td>
<td>185</td>
<td>B9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>01100010</td>
<td>58</td>
<td>3A</td>
<td>186</td>
<td>BA</td>
<td>;</td>
<td>;</td>
</tr>
<tr>
<td>01100011</td>
<td>59</td>
<td>3B</td>
<td>187</td>
<td>BB</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
<tr>
<td>01100100</td>
<td>60</td>
<td>3C</td>
<td>188</td>
<td>BC</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>01100101</td>
<td>61</td>
<td>3D</td>
<td>189</td>
<td>BD</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>01100110</td>
<td>62</td>
<td>3E</td>
<td>190</td>
<td>BE</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>01100111</td>
<td>63</td>
<td>3F</td>
<td>191</td>
<td>BF</td>
<td>@</td>
<td>@</td>
</tr>
<tr>
<td>01101000</td>
<td>64</td>
<td>40</td>
<td>192</td>
<td>C0</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>01101001</td>
<td>65</td>
<td>41</td>
<td>193</td>
<td>C1</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>01101010</td>
<td>66</td>
<td>42</td>
<td>194</td>
<td>C2</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>01101011</td>
<td>67</td>
<td>43</td>
<td>195</td>
<td>C3</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>01101100</td>
<td>68</td>
<td>44</td>
<td>196</td>
<td>C4</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>01101101</td>
<td>69</td>
<td>45</td>
<td>197</td>
<td>C5</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>01101110</td>
<td>70</td>
<td>46</td>
<td>198</td>
<td>C6</td>
<td>G</td>
<td>G</td>
</tr>
<tr>
<td>01101111</td>
<td>71</td>
<td>47</td>
<td>199</td>
<td>C7</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>01110000</td>
<td>72</td>
<td>48</td>
<td>200</td>
<td>C8</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>01110001</td>
<td>73</td>
<td>49</td>
<td>201</td>
<td>C9</td>
<td>J</td>
<td>J</td>
</tr>
<tr>
<td>01110010</td>
<td>74</td>
<td>4A</td>
<td>202</td>
<td>CA</td>
<td>K</td>
<td>K</td>
</tr>
<tr>
<td>01110011</td>
<td>75</td>
<td>4B</td>
<td>203</td>
<td>CB</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>01110100</td>
<td>76</td>
<td>4C</td>
<td>204</td>
<td>CC</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>01110101</td>
<td>77</td>
<td>4D</td>
<td>205</td>
<td>CD</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>7-BIT BINARY</td>
<td>LOW DEC</td>
<td>LOW HEX</td>
<td>HI DEC</td>
<td>HI HEX</td>
<td>ASCII CHAR</td>
<td>INTERPRETATION</td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>------------</td>
<td>----------------</td>
</tr>
<tr>
<td>10011111</td>
<td>79</td>
<td>4F</td>
<td>207</td>
<td>CF</td>
<td>0</td>
<td>Opening Bracket</td>
</tr>
<tr>
<td>10100000</td>
<td>80</td>
<td>50</td>
<td>208</td>
<td>D0</td>
<td>P</td>
<td>Reverse Slant</td>
</tr>
<tr>
<td>10100001</td>
<td>81</td>
<td>51</td>
<td>209</td>
<td>D1</td>
<td>Q</td>
<td>Closing Bracket</td>
</tr>
<tr>
<td>10100101</td>
<td>82</td>
<td>52</td>
<td>210</td>
<td>D2</td>
<td>R</td>
<td>Circumflex</td>
</tr>
<tr>
<td>10100111</td>
<td>83</td>
<td>53</td>
<td>211</td>
<td>D3</td>
<td>S</td>
<td>Underline</td>
</tr>
<tr>
<td>10101000</td>
<td>84</td>
<td>54</td>
<td>212</td>
<td>D4</td>
<td>T</td>
<td>Opening Quote</td>
</tr>
<tr>
<td>10101001</td>
<td>85</td>
<td>55</td>
<td>213</td>
<td>D5</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>10101110</td>
<td>86</td>
<td>56</td>
<td>214</td>
<td>D6</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>10101111</td>
<td>87</td>
<td>57</td>
<td>215</td>
<td>D7</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>10110000</td>
<td>88</td>
<td>58</td>
<td>216</td>
<td>D8</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>10110001</td>
<td>89</td>
<td>59</td>
<td>217</td>
<td>D9</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>10110100</td>
<td>90</td>
<td>5A</td>
<td>218</td>
<td>DA</td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>10110101</td>
<td>91</td>
<td>5B</td>
<td>219</td>
<td>DB</td>
<td></td>
<td>n/a</td>
</tr>
<tr>
<td>10111000</td>
<td>92</td>
<td>5C</td>
<td>220</td>
<td>DC</td>
<td>\</td>
<td>Reverse Slant</td>
</tr>
<tr>
<td>10111001</td>
<td>93</td>
<td>5D</td>
<td>221</td>
<td>DD</td>
<td></td>
<td>Closing Bracket</td>
</tr>
<tr>
<td>10111100</td>
<td>94</td>
<td>5E</td>
<td>222</td>
<td>DE</td>
<td>^</td>
<td>Circumflex</td>
</tr>
<tr>
<td>10111101</td>
<td>95</td>
<td>5F</td>
<td>223</td>
<td>DF</td>
<td></td>
<td>Underline</td>
</tr>
<tr>
<td>11000000</td>
<td>96</td>
<td>60</td>
<td>224</td>
<td>E0</td>
<td></td>
<td>Opening Quote</td>
</tr>
<tr>
<td>11000001</td>
<td>97</td>
<td>61</td>
<td>225</td>
<td>E1</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td>11000010</td>
<td>98</td>
<td>62</td>
<td>226</td>
<td>E2</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>11000011</td>
<td>99</td>
<td>63</td>
<td>227</td>
<td>E3</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>11000100</td>
<td>100</td>
<td>64</td>
<td>228</td>
<td>E4</td>
<td>d</td>
<td></td>
</tr>
<tr>
<td>11000101</td>
<td>101</td>
<td>65</td>
<td>229</td>
<td>E5</td>
<td>e</td>
<td></td>
</tr>
<tr>
<td>11000110</td>
<td>102</td>
<td>66</td>
<td>230</td>
<td>E6</td>
<td>f</td>
<td></td>
</tr>
<tr>
<td>11001111</td>
<td>103</td>
<td>67</td>
<td>231</td>
<td>E7</td>
<td>g</td>
<td></td>
</tr>
<tr>
<td>11010000</td>
<td>104</td>
<td>68</td>
<td>232</td>
<td>E8</td>
<td>h</td>
<td></td>
</tr>
<tr>
<td>11010001</td>
<td>105</td>
<td>69</td>
<td>233</td>
<td>E9</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>11010010</td>
<td>106</td>
<td>6A</td>
<td>234</td>
<td>EA</td>
<td>j</td>
<td></td>
</tr>
<tr>
<td>11010011</td>
<td>107</td>
<td>6B</td>
<td>235</td>
<td>EB</td>
<td>k</td>
<td></td>
</tr>
<tr>
<td>11010100</td>
<td>108</td>
<td>6C</td>
<td>236</td>
<td>EC</td>
<td>l</td>
<td></td>
</tr>
<tr>
<td>11010101</td>
<td>109</td>
<td>6D</td>
<td>237</td>
<td>ED</td>
<td>m</td>
<td></td>
</tr>
<tr>
<td>11011010</td>
<td>110</td>
<td>6E</td>
<td>238</td>
<td>EE</td>
<td>n</td>
<td></td>
</tr>
<tr>
<td>11011011</td>
<td>111</td>
<td>6F</td>
<td>239</td>
<td>EF</td>
<td>o</td>
<td></td>
</tr>
<tr>
<td>11011100</td>
<td>112</td>
<td>70</td>
<td>240</td>
<td>F0</td>
<td>p</td>
<td></td>
</tr>
<tr>
<td>11011101</td>
<td>113</td>
<td>71</td>
<td>241</td>
<td>F1</td>
<td>q</td>
<td></td>
</tr>
<tr>
<td>11011110</td>
<td>114</td>
<td>72</td>
<td>242</td>
<td>F2</td>
<td>r</td>
<td></td>
</tr>
<tr>
<td>11011111</td>
<td>115</td>
<td>73</td>
<td>243</td>
<td>F3</td>
<td>s</td>
<td></td>
</tr>
<tr>
<td>11100000</td>
<td>116</td>
<td>74</td>
<td>244</td>
<td>F4</td>
<td>t</td>
<td></td>
</tr>
<tr>
<td>11100001</td>
<td>117</td>
<td>75</td>
<td>245</td>
<td>F5</td>
<td>u</td>
<td></td>
</tr>
<tr>
<td>11100010</td>
<td>118</td>
<td>76</td>
<td>246</td>
<td>F6</td>
<td>v</td>
<td></td>
</tr>
<tr>
<td>11100011</td>
<td>119</td>
<td>77</td>
<td>247</td>
<td>F7</td>
<td>w</td>
<td></td>
</tr>
<tr>
<td>11100100</td>
<td>120</td>
<td>78</td>
<td>248</td>
<td>F8</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>11100101</td>
<td>121</td>
<td>79</td>
<td>249</td>
<td>F9</td>
<td>y</td>
<td></td>
</tr>
<tr>
<td>11101010</td>
<td>122</td>
<td>7A</td>
<td>250</td>
<td>FA</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>11101011</td>
<td>123</td>
<td>7B</td>
<td>251</td>
<td>FB</td>
<td></td>
<td>Opening Brace</td>
</tr>
<tr>
<td>11110000</td>
<td>124</td>
<td>7C</td>
<td>252</td>
<td>FC</td>
<td>Vertical Line</td>
<td>n/a</td>
</tr>
<tr>
<td>11110001</td>
<td>125</td>
<td>7D</td>
<td>253</td>
<td>FD</td>
<td>Closing Brace</td>
<td>n/a</td>
</tr>
<tr>
<td>11110110</td>
<td>126</td>
<td>7E</td>
<td>254</td>
<td>FE</td>
<td>Overline (Tilde)</td>
<td>n/a</td>
</tr>
<tr>
<td>11111111</td>
<td>127</td>
<td>7F</td>
<td>255</td>
<td>FF</td>
<td>DEL</td>
<td>Delete/Rubout</td>
</tr>
</tbody>
</table>

15. Use Closing Quote (39). For high value, use CHRS$(96), etc.
This appendix contains two tables designed to help you diagnose problems that can occur when using the SSC to communicate with an RS-232-C device. The device can be a printer, or a plotter, or terminal, or another computer, or some other Data Terminal Equipment (DTE), and it can be connected either directly, or via a modem or some other Data Communication Equipment (DCE). Whenever two DTEs are connected together, there must be TWO modems (DCEs) or ONE modem eliminator (such as the jumper block when it points toward the word TERMINAL) between them.

When diagnosing problems, remember that there are many variables involved in the communications connection:

- the Apple II and its keyboard, screen, and software
- the SSC, the slot it is in, its switch settings (especially mode selection), its jumper block, cable, and software commands
- the external cable, with some number of wires (enough wires?) connected to pins (all the correct pins?) at each end
- possibly two modems connected by low-grade telephone lines, plus another cable from the remote modem to the remote device
- an RS-232-C device at the other end, with its own switch settings and needs (such as paper, ribbon, AC power...)

As you can see, making all these components work together correctly is no mean feat. If there are problems, the easiest way to resolve them is to start with very simple, sure communication between the Apple and the device. Once you have established basic communication (even if the characters are garbled), further troubleshooting becomes much easier. Be patient and methodical.

Trouble usually has characteristics visible on the Apple II screen (Table E-1), or at the device (Table E-2). If your troubleshooting efforts fail, consult your Apple dealer—but first record all the variables (as outlined above) and the symptoms you observed.
<table>
<thead>
<tr>
<th>Problem</th>
<th>Symptom</th>
<th>Possible Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>no data transfer</td>
<td>no sign of any communication at all</td>
<td>cable wires not connected OK; jumper block facing wrong way</td>
<td>check all cable connections, then pin assignments; try reversing jumper block</td>
</tr>
<tr>
<td>characters garbled</td>
<td>jh2 3g%$Q</td>
<td>wrong baud rate</td>
<td>change SW1-1 TO SW1-4 or use (&lt;n&gt;B` command</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wrong data format</td>
<td>change SW2-1 (and SW2-2 in Comm Mode) or use (&lt;n&gt;D` command to change format</td>
</tr>
<tr>
<td></td>
<td></td>
<td>other device is off, out of paper, etc., off-line</td>
<td>turn on device, remedy its problems, put it on-line</td>
</tr>
<tr>
<td>paper not advancing</td>
<td>one line of smudge</td>
<td>printer needs line feeds from SSC</td>
<td>turn SW2-5 ON or use L(inefeed E(nable command</td>
</tr>
<tr>
<td>printer is skipping lines</td>
<td>lines look like this</td>
<td>printer and SSC both generating &lt;LF&gt; after &lt;CR&gt;</td>
<td>turn off SW2-5 in Printer Mode, or use L(inefeed D(isable command</td>
</tr>
<tr>
<td>missing characters</td>
<td>mssig caracrts</td>
<td>device buffer is overflowing</td>
<td>if device supports full RS-232-C handshaking, ensure all required cable wires are connected</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>if device supports only ETX/ACK, set SIC P8A Mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>if device supports XON/ XOFF, set Printer Mode and use X(OFF E(nable cmd or set Comm Mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>if device supports none of these, set delays with (&lt;n&gt;C, (&lt;n&gt;L and (&lt;n&gt;$` cmds</td>
</tr>
<tr>
<td>device sticks at line’s end going nuts</td>
<td>one long OK line, smudge at right end</td>
<td>device doesn’t generate own &lt;CR&gt;, and isn’t getting enough from Apple</td>
<td>use SIC P8 Mode and (&lt;n&gt;N` command, or Printer Mode and C command plus appropriate SW2-3 and SW2-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>have software send &lt;CR&gt; before right margin</td>
</tr>
</tbody>
</table>

Table E-1. Problems Detected at the Device
<table>
<thead>
<tr>
<th>Problem</th>
<th>Symptom</th>
<th>Possible Cause</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple has occasional bad times</td>
<td>it works one minute &amp; not next</td>
<td>ACIA interrupting the Apple when DCD or DSR changes</td>
<td>make sure that interrupt switch SW2-6 is OFF</td>
</tr>
<tr>
<td>Apple not working</td>
<td>dead kybd and screen</td>
<td>SSC in slot #3 under Pascal</td>
<td>Pascal expects external terminal to run the show</td>
</tr>
<tr>
<td>Apple kybd seems off</td>
<td>keystrokes all lost</td>
<td>echo off; keyboard zapped; IN# not 0</td>
<td>use E(cho E(nable cmd; unzap with POKE; IN##)</td>
</tr>
<tr>
<td>screen seems off</td>
<td>nothing typed is displayed</td>
<td>device not echoing (half duplex) or ACIA not sending to screen</td>
<td>in Comm or Terminal Mode, use E(cho E(nable cmd; in SIC or Printer Mode, use I command or SW2-3 &amp; -4 ON</td>
</tr>
<tr>
<td>screen is seeing double</td>
<td>eevveerryy tthhiinnngg ttwwiiccee</td>
<td>device &amp; SSC both echoing to Apple (full duplex)</td>
<td>use E(cho D(isable cmd in Comm Mode or use &lt;n&gt;N cmd in Printer Mode</td>
</tr>
<tr>
<td>screen is spacing double</td>
<td>lines look like this</td>
<td>device generating and sending &lt;LF&gt; after &lt;CR&gt;</td>
<td>use M(ask E(nable command to remove extra linefeeds</td>
</tr>
<tr>
<td>forced uppercase display</td>
<td>lowercase becomes UPPERCASE</td>
<td>Apple monitor changing letters in GETLINE routine</td>
<td>use &lt;n&gt;T command to allow lowercase to pass through (not possible in Pascal)</td>
</tr>
<tr>
<td>Apple misses some characters at the beginning of lines</td>
<td>pple sses ome racters t the bgmning lines</td>
<td>screen scrolling too slowly, or BASIC or Pascal program running too slowly, and so ACIA overruns</td>
<td>turn off screen (&lt;n&gt;N or SW2-3 &amp; -4 in Prtr Mode); reduce scroll window; use assembly language or faster program routines; use lower baud rate (9600 vs. 1200); use &lt;n&gt;C, &lt;n&gt;L or &lt;n&gt;F commands; in Comm Mode, chain (&lt;n&gt;S cmd) to 80-column card with its own scrolling hardware</td>
</tr>
</tbody>
</table>

Table E-2. Problems Detected at the Apple
APPENDIX F
ERROR CODES

The SSC uses I/O scratchpad address $678+s (s is the number of the slot that the SSC is in) to record status after a read operation. The firmware calls this byte STSBYTE. Table F-1 lists the bit definitions of this byte:

```
<table>
<thead>
<tr>
<th>Bit</th>
<th>&quot;1&quot; Means</th>
<th>&quot;0&quot; Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Parity Error occurred</td>
<td>No Parity Error occurred</td>
</tr>
<tr>
<td>1</td>
<td>Framing Error occurred</td>
<td>No Framing Error occurred</td>
</tr>
<tr>
<td>2</td>
<td>Overrun occurred</td>
<td>No Overrun occurred</td>
</tr>
<tr>
<td>3</td>
<td>Carrier lost</td>
<td>Carrier present</td>
</tr>
<tr>
<td>5</td>
<td>Error occurred</td>
<td>No error occurred</td>
</tr>
</tbody>
</table>
```

Table F-1. STSBYTE Bit Definitions

The terms Parity Error, Framing Error and Overrun are defined in the Glossary.

Bits 0, 1, and 2 are the same as the corresponding three bits of the ACIA Status Register (Appendix A). Bit 3 indicates whether or not the Data Carrier Detect (DCD; Chapter 4) signal went false at any time during the receive operation. Bit 5 is set if any of the other bits are set, as an overall error indicator. If bit 5 is the only bit set, an unrecognized command was detected. If all bits are 0, no error occurred.

In BASIC, you can check this status byte via a PEEK $678+s (s is the SSC slot), and reset it with a POKE command at the same location.

In Pascal, the IORESULT function returns the error code value.
Any character—including the carriage return at the end of a WRITELN statement—will cause posting of a new value in IORESULT.

Table F-2 shows the possible combinations of error bits correspond to these decimal error codes.

<table>
<thead>
<tr>
<th>BASIC PEEK $678+s</th>
<th>Carrier</th>
<th>Framing</th>
<th>Parity</th>
</tr>
</thead>
<tbody>
<tr>
<td>or Pascal IORESULT</td>
<td>Lost</td>
<td>Overrun</td>
<td>Error</td>
</tr>
<tr>
<td>Ø</td>
<td>(no error)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>33</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>34</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>35</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>36</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>37</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>38</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>39</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>4Ø</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>41</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>42</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>43</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>44</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>45</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>46</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>47</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table F-2. Error Codes and Bits

These error codes begin with the number 32 to avoid conflicting with previously defined and documented system error codes.
To avoid lengthy or repetitive definitions, many terms used in one
definition are themselves defined elsewhere in this glossary. Also
for the sake of brevity, terms and expressions are spelled out, with
their abbreviations immediately after them. In a glossary of this
size, the reader will have little difficulty locating abbreviations.

**ACK:** An ASCII character (decimal 6; Appendix D) sent from a device
to the Apple II in response to an ETX or ENQ character in SIC
P8A Emulation Mode.

**American Standard Code for Information Interchange (ASCII):** A
standard defining the codes to represent a 128-element
character set (Appendix D) in a fixed way for devices of
different manufacturers. It is the standard for digital
communication over telephone lines.

**Asserted:** Made true (positive in positive-true logic; negative in
negative-true logic). Usually refers to electrical signals,
like the RS-232-C signal Clear To Send, etc.

**Asynchronous:** Having a variable time interval between characters.

**Asynchronous Communications Interface Adapter (ACIA):** In the SSC, a
single chip (Synertek 6551 or equivalent) that converts data
from parallel to serial form and vice versa, and handles serial
transmission and reception and RS-232-C signals, under the
control of internal registers set and changed by SSC firmware.

**Baud:** A unit of signalling speed equal to the number of discrete
conditions or signal events per second. With the SSC, for
example, using a data format of 1 start bit, 7 data bits, 1
parity bit and 1 stop bit (10 bits in all), 300 baud is
approximately equal to 30 characters per second.

**Binary:** A number system with two digits, "$0$" and "$1$," with each
digit position moving from right to left representing a
successive power of two. For example, 1 represents decimal 1;
10 represents 2; 100 represents 4; 1000 represents 8, etc.

**Bit:** A BInary digiT, either a $0$ or a $1$. 
BREAK: A \( \Phi.233 \) second SPACE (\( \Phi \)) signal sent over a communication line to interrupt the sender. This signal is often used to end a session with a timesharing service.

Carriage Return (CR): An ASCII character (decimal 13; Appendix D) that ordinarily causes a printer or display screen to place the subsequent character on the left margin. On a manual typewriter, this movement is combined with linefeed (the advancement of the paper to the next line). With computers, carriage return and linefeed are separate, causing hair-raising problems for the user.

Carrier: The background signal on a communication channel that is modified to "carry" the information. Under RS-232-C, the carrier signal is equivalent to a continuous MARK or 1; a transition to \( \Phi \) then represents a start bit.

Character: Any symbol that has a widely understood meaning. In the ASCII code, letters, numbers, punctuation marks, and so on, are all characters (Appendix D).

Chip: A tiny wafer of silicon, with conductive metallic impurities, that has layers of microscopic circuits etched on it.

Clear To Send (CTS): An RS-232-C signal from a DCE to a DTE that the SSC keeps false until the DCE makes it true, indicating that all circuits are ready to transfer data.

Command Character: An ASCII character, usually \(<\text{CTRL-A}\>\) or \(<\text{CTRL-I}\>\) (Appendix D), that causes the SSC firmware to interpret subsequent characters as a command.

Command Register: An ACIA location (at hexadecimal address \$C08A+s\$) that stores parity type and RS-232-C signal characteristics.

Communications Interface Card (CIC): An Apple II interface card designed to connect the Apple II to a device via a DCE.

Communications Mode: An operating state in which the SSC is prepared to exchange data and signals with a DCE.

Control Character: Any character generated by holding down the key marked CTRL while pressing some other key.

Control Register: An ACIA location (at hexadecimal address \$C08B+s\$) that stores data format and baud rate selections.

Daisy Chaining: A method of passing incoming signals and data from one peripheral connector slot to another, such as from the SSC slot to a slot containing an 80-column-display card.

Data Bit: With the SSC, one of 5 to 8 bits representing a character.
Data Carrier Detect (DCD): An RS-232-C signal from a DCE to a DTE (such as the Apple II) indicating that a communication connection has been established. The SSC's internal circuits hold DCD false until the external device sets DCD true.

Data Communication Equipment (DCE): As defined by the RS-232-C standard, any device that transmits or receives information. Usually this is a modem. However, when a Modem Eliminator is used, the Apple II looks like a DCE to the other device, and the other device looks like a DCE to the Apple.

Data Conversion: Changing of data from parallel to serial form or from serial to parallel form.

Data Format: The form in which data is stored, manipulated or transferred. Serial data transmitted and received by the SSC has a data format of: one start bit, 5 to 8 data bits, an optional parity bit, and one, one and a half, or two stop bits.

Data Set Ready (DSR): An RS-232-C signal from a DCE to a DTE indicating that the DCE has established a connection.

Data Terminal Equipment (DTE): As defined by the RS-232-C standard, any device that generates or absorbs information, thus acting as a terminus of a communication connection.

Data Terminal Ready (DTR): An RS-232-C signal from a DTE to a DCE indicating a readiness to transmit or receive data.

Default Value: A value that is assumed or set in the absence of explicit instructions otherwise.

Device: A piece of equipment; usually a printer, plotter, terminal or computer. When the jumper block is in the MODEM position, the SSC expects the device to be a DCE (such as a modem).

Echo: To send an input character to a video screen, printer, or other output device. On a typewriter, what we strike on the keyboard appears on the page in the same step. With a computer, these two steps are controlled separately.

Electromagnetic Interference (EMI): Electrical or magnetic signals or noise that disturbs the operation of radio or television receivers. For example, a hair dryer often creates EMI that fuzzes up the picture on a nearby television set.

Emulation Mode: A manner of operating in which one computer or interface imitates another. For example, in SIC P8 Emulation Mode, the SSC acts very much like an Apple II Serial Interface Card with the P8 version of firmware.

ENQ: An ASCII character (decimal 5; Appendix D) used in the RNQ/ACK protocol (SIC P8A Emulation Mode).
ETX: An ASCII character (decimal 3; Appendix D) used in the ETX/ACK protocol (SIC P8A Emulation Mode).

Even Parity: Use of an extra bit set to 0 or 1 as necessary to make the total number of 1 bits an even number. For example, the 7-bit ASCII code for the letter A (1000001) has two 1 bits; for even parity, the transmitting device appends an eighth bit equal to 0 so that the total number of 1 bits remains even. The receiving device can count 1 bits as a way of checking for transmission errors.

False: Zero or negative voltage in positive-true logic; positive voltage in negative-true logic. Absence of an arbitrary signal or condition.

Firmware (FW): Software that resides in ROM and so is relatively unchangeable (firm) compared to software in RAM.

Form Feed (FF): An ASCII character (decimal 12; Appendix D) that causes a printer or other paper-handling device to advance to the top of the next page.

Framing Error (FRM): Absence of the expected stop bit(s) on a received character. The ACIA records this error by setting bit 1 (FRM) of its Status Register to 1. The ACIA checks and records each framing error separately: if the next character is OK, the FRM bit is cleared.

Full Duplex: Capable of simultaneous two-way communications.

Half Duplex: Capable of communications in one direction at a time.

Handshake: A kind of communication protocol in which the receiving device, when it has successfully gotten a character or block of characters, sends back an acknowledging signal, thereby triggering the next transmission.

Hardware: The actual physical switches, wires, chips, PC boards, and so on, of a computer system.

Header: A cable connector mounted on a PC board.

Hexadecimal: A numbering system that uses 16 digits; usually these are represented by the ten decimal digits, 0 through 9, plus the letters A through F (A representing decimal ten, F representing decimal fifteen, etc.). Each hexadecimal digit can represent a string of four binary digits.

High-order Bit: See Most Significant Bit.

Initialization: The process of setting up initial values and conditions. In the SSC, the firmware finds out the switch positions and the current operating system, and uses these
findings to initialize both the ACIA registers and the Scratchpad RAM locations for the slot the SSC is in.

Input: Data that flows from the outside world into the Apple II.

Interface: Some combination of hardware, firmware and software that makes possible the useful connection of two otherwise incompatible pieces of equipment.

Interrupt: A special control signal from an external source that diverts the Apple II from the program it is executing to a specific routine that handles the condition (such as a printer gone awry) that caused the interrupt.

Jumper Block: In the SSC, a plastic plug with pins connected in such a way that it passes RS-232-C signals between the SSC and the external device either unchanged (MODEM position) or permuted in the manner of a Modem Eliminator (TERMINAL position).

Least Significant Bit (LSB): The right-hand bit of a binary number as written down; its positional value is $0$ or $1$ (that is, $0$ or $1$ times $2$ to the $0$ power).

Linefeed (LF): An ASCII character (decimal $10$; Appendix D) that ordinarily causes a printer or video display to advance to the next line.

Local: Nearby; capable of direct connection using wires only.

Low-order Bit: See Least Significant Bit.

MARK Parity: A bit of value $1$ appended to the high-order end of a binary number for transmission. The receiving device can then check for errors by looking for this value on each character.

Mode: Manner of operating. The SSC can operate in one of four chief modes, depending on the settings of switches SW1-5 and SW1-6: Printer Mode, Communications Mode, SIC P8 Emulation Mode, and SIC P8A Emulation Mode.

Modem: MODulator/DEModulator; a DCE device that connects a DTE to communications lines. As used with the SSC, a device that exchanges RS-232-C signals with the ACIA to establish a communications connection, and then either converts data from RS-232-C voltages to RS-232-C tones for transmission, or performs the opposite conversion on received data.

Modem Eliminator: The physical crossing of wires that replaces a pair of modems for direct connection of two pieces of RS-232-C Data Terminal Equipment. In the SSC, the jumper block serves this purpose when installed in the TERMINAL position.
Most Significant Bit (MSB): The leftmost bit of a binary number as written down. This bit represents $0$ or $1$ times 2 to the power one less than the total number of bits in the binary number. For example, in the binary number $100000$, the $1$ represents $1$ times 2 to the fourth power, or sixteen.

Odd Parity: Use of an extra bit set to $0$ or $1$ as necessary to make the total number of $1$ bits an odd number. For example, the 7-bit ASCII code for the letter A (1000001) has two $1$ bits; for odd parity, the transmitting device appends an eighth bit equal to 1, making the total number of $1$ bits odd. The receiving device can check for transmission errors by counting $1$ bits.

Output: Data that flows from the Apple II to an external device.

Overrun (OVR): A condition that occurs when the Apple II processor does not retrieve a received character from the Receive Data Register before the subsequent character arrives. The ACIA automatically sets bit 2 (OVR) of its Status Register; subsequent characters are lost. The Receive Data Register contains the last valid data word received.

P8: One of two types of Programmable ROM (PROM) installed in the Apple II Serial Interface Card. This PROM performed batch moves, but had no provision for software handshaking.

P8A: One of two types of Programmable ROM (PROM) installed in the Apple II Serial Interface Card. This PROM provided the ENQ/ACK software handshaking required by several types of printers.

Parallel Interface: A connection between two devices where there is a separate wire for each bit of a character, so that an entire character can be transferred in a single instant.

Parity: Maintenance of a sameness of level or count, usually the count of $1$ bits in each character, for error checking. In the SSC, the ACIA has a register that stores the type of parity selected (none, odd, even, MARK or SPACE). It automatically generates the parity bit when transmitting, and both checks and discards parity bits appended to received characters.

Parity Error (PAR): Absence of the correct parity bit value in a received character. The ACIA records this error by setting bit $0$ (PAR) of its Status Register to $1$.

Peripheral Connector Slot: One of eight $50$-pin slots inside the Apple II case near the back. Within certain restrictions, each slot can contain add-on memory, an adapter for $80$-column display, or an interface to an external device.

Polarized Header: On the SSC, a $10$-pin female connector for the internal cable; this connector has a slot on one side that receives a "key" on the cable's male connector.
Printed Circuit (PC) Board: A sheet of stiff nonconductive material with one or more thin layers of metal bonded to it. Unwanted areas of this metal are etched away, leaving the paths of the desired circuits. Electronic components can then be soldered to the board. Small PC boards are also called cards.

Printer Mode: An operating state in which the SSC is prepared to exchange data and signals with another DTE (such as a printer).

Protocol: A predefined exchange of control signals between devices enabling them to prepare for coordinated data transfer.

Radio Frequency Interference (RFI): Electromagnetic interference occurring at frequencies used for radio communications.

Random Access Memory (RAM): A series of storage locations that can be accessed directly (by means of horizontal and vertical coordinates) for both reading and writing.

Read Only Memory (ROM): A series of storage locations that can be read but cannot be written to; this protects the programs and data in the ROM from alteration or destruction.

Receive Data Register: A read-only register in the ACIA (at hexadecimal location $C\text{888}+$s$\text{0}$) that stores the most recent character successfully received.

Remote: Too distant for direct connection via wires or cables only.

Request To Send (RTS): An RS-232-C signal from a DTE to a DCE to prepare the DCE for data transmission.

Ring Indicator (RI): An optional RS-232-C signal from a DCE to a DTE that indicates the arrival of a call.

RS-232-C: A standard created by the Electronic Industries Association (EIA) to allow devices of different manufacturers to exchange serial data—particularly via telephone lines. The ACIA in the SSC implements all the required primary RS-232-C signals. These signals are true when at $\text{0}$ volts.

Scratchpad RAM: Eight locations in the Apple's memory reserved for each of the 8 peripheral connector slots (64 bytes in all).


Serial Interface: A connection in which all the bits of a character are sent along a single wire one after the other.

Serial Interface Card (SIC): An Apple II product designed to connect an RS-232-C device directly to the Apple II.
SIC Emulation Mode: A state of operation in which the SSC imitates an Apple II Serial Interface Card.

SPACE Parity: A bit of value 0 appended to a binary number for transmission. The receiving device can look for this value on each character as a means of error checking.

Start Bit: A transition from a MARK signal to a SPACE signal for one bit-time, indicating that the next string of bits represents a character.

Status Register: An ACIA register (hexadecimal location $C089+0$) that stores the state of two of the RS-232-C signals and of the Transmit and Receive Data Registers, as well as the outcome of the most recent character transfer.

Stop Bit: A MARK signal following a string of data bits to indicate the end of a character.

Super Serial Card (SSC): The interface card described in this manual. It is called "super" because it can simultaneously transmit and receive data in one of 35 formats at any of 15 speeds, honor several software protocols, communicate directly with either DTE or DCE, change operating characteristics in response to software commands, and dovetail with the chief operating environments offered with the Apple II.

Terminal: An input/output device, usually made up of a keyboard and video display and sometimes including its own printer and magnetic storage devices, that can act as a separate and even remote site for data transfer with a computer system.

Terminal Mode: An operating state of the SSC in which the firmware bypasses the Apple II's central processor, and makes the Apple act as a simple terminal capable of generating all of the ASCII characters.

Transmit Data Register: A write-only register in the ACIA (at hexadecimal location $C088+0$) that holds the current character to be transmitted.

True: Positive voltage in positive-true logic; zero or negative voltage in negative-true logic. Assertion of an arbitrary signal or condition.

XOFF: An ASCII character (decimal 19; Appendix D) sent by a receiving device to a transmitting device to halt transmission of characters.

XON: An ASCII character (decimal 17; Appendix D) used in the XON/XOFF protocol as a go-ahead character from the receiving device to the sending device after an XOFF has been sent to halt transmission.
FIGURES AND TABLES

Chapter 1
GETTING STARTED

1. Figure 1-1 Photo of the Super Serial Card
2. Figure 1-2 Line Drawing of the SSC
3. Figure 1-3 Components of the Internal Cable and Clamp Assembly
4. Figure 1-4 Sliding the "Key" into the Groove
5. Figure 1-5 Internal Cable Attached Correctly to SSC

Chapter 2
PRINTER MODE

5. Figure 2-1 SSC Set for Printer Mode
10. Figure 2-2 SSC in Slot #1 and Clamp Assembly in Notch
6. Table 2-1 Commonly Used Switch Settings for Printer Mode
7. Table 2-2 Baud Rate Switch Settings
8. Table 2-3 Line Width and Video Switch Settings
14. Table 2-4 Printer Mode Commands
15. Table 2-5 Baud Rate Selections
15. Table 2-6 Data Format Selections
16. Table 2-7 Parity Selections
16. Table 2-8 Time Delay Selections
18. Table 2-9 Lowercase Character Displays

Chapter 3
COMMUNICATIONS MODE

21. Figure 3-1 SSC Set for Communications Mode
26. Figure 3-2 SSC in Slot #2 and Clamp Assembly in Notch
22. Table 3-1 Commonly Used Switch Settings for Communications Mode
23. Table 3-2 Baud Rate Switch Settings
23. Table 3-3 Data Format Selections
29. Table 3-4 Summary of Communications Mode Commands
30. Table 3-5 Baud Rate Selections
30. Table 3-6 Data Format Selections
31. Table 3-7 Parity Selections
32. Table 3-8 Time Delay Selections
33. Table 3-9 Lowercase Character Displays
35. Table 3-10 Special ASCII Character Generation
35. Table 3-11 Terminal Mode Commands
Chapter 4

HOW THE SSC WORKS

39  Figure 4-1 Parallel Data Transfer
39  Figure 4-2 Serial Data Transfer
39  Figure 4-3 Parallel-to-Serial Data Conversion
40  Figure 4-4 RS-232-C Serial Data Format
42  Figure 4-5 An RS-232-C Setup with Modems
43  Figure 4-6 An RS-232-C Setup with a Modem Eliminator
43  Figure 4-7 SSC Operating Modes
44  Figure 4-8 SSC Configurations
45  Figure 4-9 Overall Block Diagram of the SSC

38  Table 4-1 Binary and Decimal Digits and Quantities
41  Table 4-2 RS-232-C Signals as Interpreted by the Sender
41  Table 4-3 RS-232-C Signals as Interpreted by the Receiver
46  Table 4-4 SSC Address Remapping
47  Table 4-5 Registers in SSC Peripheral I/O Space

Appendix A

FIRMWARE

50  Table A-1 I/O Routine Offsets and Registers under Pascal 1.1
51  Table A-2 Bytes Used for Device Identification
51  Table A-3 Device Class Digit
52  Table A-4 Memory Usage Map
52  Table A-5 Zero-Page Locations Used by SSC
53  Table A-6 Scratchpad RAM Locations Used by SSC
54  Table A-7 SSC Registers in Peripheral Card I/O Space
55  Table A-8 Monitor ROM Entry Points Used by SSC
55  Table A-9 BASIC Entry Points Used by SSC
56  Table A-10 Pascal 1.0 Entry Points Used by SSC
57  Table A-11 Pascal 1.1 Offsets Used by SSC
57  Table A-12 SSC Special Firmware Locations

Appendix B

APPLE INTERFACE CARD EMULATION

93  Table B-1 SIC Switch Settings, PEEKs and POKEs, Part I
94  Table B-2 SIC Switch Settings, PEEKs and POKEs, Part II
Appendix C
SPECIFICATIONS AND SCHEMATICS

98  Table C-1  Connector Pin Assignments
99  Table C-2  Jumper Block Wiring

Appendix E
TROUBLESHOOTING HINTS

106  Table E-1  Problems Detected at the Device
107  Table E-2  Problems Detected at the Apple

Appendix F
ERROR CODES

109  Table F-1  STSBYTE Bit Definitions
110  Table F-2  Error Codes and Bits
INDEX

A
A register 50
ACIA 45, 47, 48, 91, 95, 96, 97
function 47
registers 49, 54
status register 109
ACK 91, 94, 102, 106
address
bus 45
lines 46
mapping 46
space 46
addressing logic 45, 46
answer modem 42
Apple II Communications Card (see CIC)
Apple II Communications Interface Card (see CIC)
Apple II Parallel Cards 96
Apple II Serial Interface Card (see SIC)
APPLE SSC: 8, 12, 27
Applesoft 36, 98
ASCII 24, 34, 38, 95, 96
code table 101-104
special characters 35, 36, 101
assembly language 11, 27, 98
Asynchronous Communications Interface Adapter (see ACIA)

B
B command 35, 36
BASIC 8, 11, 12, 17, 18, 19, 27, 31, 33, 36, 92, 101, 104
tenary points 55
error codes 109, 110
protocol 50
batch moves 95
baud rate 6, 15, 22, 30
selection 15, 30
switch settings 7, 23, 47, 91, 92, 93, 98
binary 37, 101
data 24, 38

binary/decimal representation 38
bit 7, 23, 37
data 30, 40, 93
parity 40
start 40
stop 30, 40, 92, 94
block
diagram, SSC 45
moves 91
branch table locations, Pascal 1.1 50
break signal 36
buffer 45, 47
bus
address 45
data 45, 48
parallel 48
byte 38
device signature 51

cable
connector 10, 26
external 1, 10, 26
internal 1, 9, 24, 45
socket 2
capitalize 34
carriage return delay 7, 17, 94, 95, 96
carrier
lost 110
signal 42
chaining 34
character
displays 33
enquire 95
characters
echo on screen 34
lowercase 34, 102
uppercase 34, 102
capitalize 34
CIC 36, 91, 96
clamp assembly 2, 3
Clear To Send (see CTS)
code 38
column overflow 17
command
character
change 13, 28
Communications Mode 28, 102
data 37
ASCII 24
binary 24
bit 7, 15, 23, 24, 30, 40, 93
bus 45, 48
bus buffer 45
conversion 39, 47
format 7, 23, 28, 30, 40, 47, 91, 92, 93
command 15
RS-232-C 40
selection 15, 23, 30
input 48, 96
output 48
serial 48
terminal 40
transfer 47
Data Carrier Detect (see DCD)
Data Communication Equipment
(see DCE)
Data Set Ready (see DSR)
Data Terminal Equipment (see DTE)
Data Terminal Ready (see DTR)
DB-25 connector 10, 26, 97, 99
pin assignments 98
DCD 41, 42, 43, 47, 48, 95, 98, 99
DCE 41, 48, 105
decimal error codes 110
default parity 16
demodulator 41
device
class digit 51
identification 51
remote 19, 96
signature byte 51
DEVICE SELECT 46, 47
disable 12, 28
DOS 8, 11, 27
Dow Jones News & Quotes
Reporter 22, 36
setup program for 36
DSR 41, 42, 43, 48, 95, 98, 99
DTE 41, 42, 48, 105
DTR 41, 42, 43, 95, 98, 99
INDEX 125

E

E command 31, 34, 35
 echoing 34, 35
 edge connector 2
 EIA 40
 80-column-display 34
 Electronic Industries
 Association (see EIA)
 emulation
 modes 91-96
 old Serial Interface Card 91
 P8 92
 SIC 92, 102
 enable 12, 28
 enquire character 95
 enter Terminal Mode 35
 entry points
  BASIC 55
  hardware 49
  I/O routine 50
  monitor ROM 55
  Pascal 1.0 55, 56
  Pascal 1.1 55, 56
  SSC 55
 environment, SSC 97
 error
  codes 109-110
  BASIC 110
  Pascal 110
  framing 109, 110
  parity 109, 110
  ENQ 95, 102
  ENQ/ACK 102
  <ESC> 34, 35, 101-104
  <ESCL> 91
  <ESCU> 91
  ETX 91, 94, 95, 102, 106
  ETX/ACK 91, 94, 102, 106
  Expansion ROM, SSC 46, 52
  even parity 16, 31, 40, 92, 93
  external cable 1, 10, 26

locations, special 57
 memory usage 53
 protocol, Pascal 1.1 49
 SSC driver 46
 form feed delay 14, 16, 17, 29, 32
 format, data 92, 93
 framing error 109, 110
 full-duplex 35

G

generate
  ASCII characters 35
  <CR> 17, 95, 96
  <LF> 8, 17, 24, 31, 92, 94
  generic signature 51
  ground 10, 26, 42, 43

H

half-duplex 35, 36, 95
 handshake 94, 95
 handshaking 19
 hardware 96
 entry points 49
 handshaking 19
 hexadecimal 101

I

IN# 14
 IDS 560 Paper Tiger 6
 indirect addressing 56
 input 37
 data 48, 96
 installation, SSC 9
 Integer BASIC 36
 interface
 receive 45
 transmit 45
 internal
  cable 1, 9, 24, 45
  attachment 3
  clamp assembly 2, 3
  shielding 10, 26
 interrupt
 handling 50
 request 47
 Pascal 1.0 50

F

F command 17, 31
 FCC rules 10, 26
 <FF> 14, 16, 17, 29, 32
 firmware 34, 47, 49, 109
 entry points, Pascal 1.0
  (SIC) 49
 interrupt handling 50
 listings 57-96
I/O
  routine entry points 50
  scratchpad address 109
  SELECT 46
  space
    peripheral card 52
    registers in 54
  STROBE 46
  IORESULT 109

J
  jumper block 2, 5, 11, 21, 27, 42, 45, 48, 95, 99, 105
  wiring 99

K
  keyboard 11, 18, 27, 33, 101, 102

L
  <LF> 8, 17, 24, 31, 91, 92, 94, 102
  line width 8, 92, 94, 96
  linefeed (see <LF>)
  listings, firmware 57-96
  local 37
  lowercase 14, 18, 29, 32-33, 34, 91, 92, 93, 95, 101, 102
  character displays 18

M
  MARK parity 16, 31, 40, 101
  mask <LF> in 17, 31
  memory usage map, SSC 53
  mode
    communications 21-33, 26, 91, 96, 102
    emulation 91-96
  P8 93
  P8A 93-94, 102, 106
  printer 5-19, 91, 92, 93-94, 102
  SSC operating 43
  terminal 22, 34-36, 96, 101, 102
  modem 22, 26, 27, 41, 42, 105
  answer 42
  eliminator 5, 42, 43, 45, 48, 99, 105
  null 42, 48
  originate 42
  MODEM 5, 21, 27, 48, 99
  modulator 41
  monitor
    Apple II 11
    ROM entry points 55

N
  NEC 5510 Spinwriter 6
  no parity 31
  null modem 42, 48

O
  odd parity 16, 31, 40
  offsets, Pascal 1.1 57
  originate 40
    modem 42
  oscillator 45, 47
  output 37
    data 48
  override switch settings 27
  overrun 109, 110

P
  P8 91-95
    emulation POKEs 92
    ROM 92, 94, 95
  P8A 91
    mode 6, 102, 106
    ROM 92, 94, 95
  parallel
    bus 48
    card commands 96
    data 37, 38-39
    I/O card 52
  parity 7, 15, 23, 30, 47, 92, 93
    bit 7, 23, 24, 40
    error 109, 110
    even 40
    MARK 40, 101
    odd 40
    selection 16, 31
Q

Q command 35, 36
Qume 6
quit Terminal Mode 36, 102

R

RAM 45, 49, 92
scratchpad locations 53

receive
interface 45
register 42
Received Data (see RXD)
register
command 45, 54
control 45, 54
receive 42
status/reset 45, 54
transmit 42, 54
transmit/receive 45

registers
A 50, 55-57
ACIA 49, 54
Pascal 1.1 50
peripheral I/O space 47
switch settings 45
X 50, 55-57
Y 50, 55-57
remote 37
device 19, 22, 26, 27, 96
terminal 34
remote-control commands 102
Request To Send (see RTS)
reset 27
command 15, 95
SSC 17, 31
RESET 46
<RETURN> 12, 27, 102
Ring Indicator (RI) 41
ROM 45
SSC Expansion 46, 52
ROM/RAM
address mapping 46
space 46
RS-232-C 40, 43, 47, 48, 95, 105
signals 40-43, 49, 98-99
RTS 41, 42, 43, 95, 98, 99
RXD 41, 42, 43, 98, 99

S

schematic, SSC 45, 100
scratchpad RAM locations 53
SCTS 98, 99
Secondary Clear To Send (see SCTS)
selection
baud rate 15, 30
data format 15, 23, 30
parity 16, 31
time delay 16, 32

INDEX 127
serial data 37, 39, 40, 48
interface card 95
setting switches 6, 22, 92
settings, switch 6, 22, 93-94
shielding, internal 10, 26
SIC 91, 92
emulation mode 92, 102
firmware entry points 49
P8A mode 102, 106
switch settings 93, 94
6502 registers 49, 50, 55-57
slew rate 48
software compatibility 98
SPACE parity 16, 31, 40, 101
special characters, ASCII 101
circuits, SSC 97
specifications, SSC 97-98
specify screen slot 34
SSC 1, 91, 95, 96, 102
block diagram 45
cable connector 10
preparation 21
configurations 44
connect printer 7
terminal 8
controlling 11
driver firmware 46
terminal 8
entry points 50, 55-57
evironment 97
Expansion ROM 46, 52
firmware memory usage 53
installation 9, 92
lowercase 18
main components 45
operating modes 43
peripheral card I/O space 52
photo 1
physical characteristics 97
registers in I/O space 54
reset 17, 31
RS-232-C signals 99
schematic 45, 100
scratchpad RAM locations 53
slot location 98
special circuits 97
specifications 97-98
switch settings 93-94
theory of operation 45
unpacking 1
using Communications Mode 27
printer 11
Printer Mode 11
terminal 11
zero-page locations 52
start bit 7, 23, 40
status byte (see STSBYTE)
register, ACIA 109
status/reset register 45
stop bit 7, 15, 23, 24, 30,
40, 92, 94
STSBYTE 109
summary Communications Mode commands 28-29
Printer Mode commands 14
Terminal Mode commands 35
Super Serial Card (see SSC)
suppress characters control 33
keyboard 18, 33
<LF> in 17, 31
switch settings baud rate 7, 23
change 14, 30
Communications Mode 22
line width 8
override 12, 27
Printer Mode 6
SIC 93-94
SSC 93-94
video 8
switches 2
Communications Mode 21-22
setting 22, 92
SSC 45
state of 47
SWI 48, 54, 56, 57, 92, 93
SWI-1 through SWI-4 2, 7,
15, 23, 31, 92, 93, 106
SWI-5 and SWI-6 2, 5, 21,
92, 93, 102, 115
SWI-7 2, 8, 24, 48, 92, 99
SWZ 6, 22, 48, 54, 56, 57,
92, 93, 94
SWZ-1 2, 7, 15, 23-24, 31,
92, 93, 94, 106
SWZ-2 2, 7, 16, 23-24, 31,
92, 94, 106
SWZ-3 and SWZ-4 2, 8, 23,
92, 94, 106, 107
SW2-5  2, 8, 17, 24, 32, 92, 93, 106
SW2-6  2, 8, 24, 48, 92, 107
SW2-7  2, 8, 24, 48, 92, 99

T

T command 35
tab 19
terminal
  connecting 8
  local 11
  unintelligent 34, 35
TERMINAL  5, 11, 21, 42, 48, 95, 99, 101, 105
Terminal Mode  22, 34-36, 101-104
  bypass 96
  change to 96
  commands 35
  enter 35, 102
  example 36
  quit 36, 102
theory of operation, SSC 45
time delay
  selection 16, 32
time-sharing 34, 36
translate 94
  command 17, 31, 91
  lowercase 18
transmit
  break signal 36
  interface 45
  register 42
transmit/receive register 45
Transmitted Data (see TXD)
troubleshooting 105-107
  at Apple II 107
  at device 106
TXD  41, 42, 43, 98, 99

V

video on/off  8, 92, 94, 96

W

wiring, jumper block 99

X

X command 17, 31, 35, 106
X register  50, 55-57
XOFF 35, 102
  recognition 18, 33
XON 102
XON/XOFF  17, 19, 31, 33, 96, 102

Y

Y register  50, 55-57

Z

Z command 17, 18, 31
zero-page locations, SSC 52

U

unintelligent terminal 34, 35, 36
uppercase 14, 18, 29, 33, 34, 91, 92, 102
Tuck end flap inside back cover when using manual.