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 and register values
- 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 $Cs05 and $18 at $Cs07 (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 $Cs00$ address space (s being the slot number where the peripheral card is installed). This space was chosen instead of the $Cb00$ space, since under BASIC protocol the $Cs00$ space is required, while the $Cb00$ 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>$Cs0D</td>
<td>initialization routine offset (required)</td>
</tr>
<tr>
<td>$Cs0E</td>
<td>read routine offset (required)</td>
</tr>
<tr>
<td>$Cs0F</td>
<td>write routine offset (required)</td>
</tr>
<tr>
<td>$Cs10</td>
<td>status routine offset (required)</td>
</tr>
<tr>
<td>$Cs11</td>
<td>$00 if optional offsets follow; non-zero if not</td>
</tr>
<tr>
<td>$Cs12</td>
<td>control routine offset (optional)</td>
</tr>
<tr>
<td>$Cs13</td>
<td>interrupt handling routine offset (optional)</td>
</tr>
</tbody>
</table>

Notice that $Cs11$ contains $00$ 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 $Cs11$ 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 6502 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>$Cs0D</td>
<td>Initialization</td>
<td>$Cs$</td>
<td>$s0$</td>
<td>(unchanged)</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
</tr>
<tr>
<td>$Cs0E</td>
<td>Read</td>
<td>$Cs$</td>
<td>$s0$</td>
<td>character read</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
</tr>
<tr>
<td>$Cs0F</td>
<td>Write</td>
<td>$Cs$</td>
<td>$s0$</td>
<td>char. to write</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>On exit</td>
<td>error code</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
</tr>
<tr>
<td>$Cs10</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></td>
<td></td>
<td></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>
</tr>
</thead>
<tbody>
<tr>
<td>$Cs05$</td>
<td>$38$  (like the old Serial Interface Card)</td>
</tr>
<tr>
<td>$Cs07$</td>
<td>$18$  (like the old Serial Interface Card)</td>
</tr>
<tr>
<td>$Cs08$</td>
<td>$01$  (the Generic Signature of new FW cards)</td>
</tr>
<tr>
<td>$Cs0C$</td>
<td>$ci$  (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–$CFFF</td>
<td>Expansion ROM</td>
<td>Eight 256-byte pages reserved for a 2K ROM or PROM; SSC maps its FW onto $C800–$CFFF (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# * 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>ZTEMPP</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>CSHW</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>Φ - 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>Φ - 7</td>
<td>Buffer count for handshake (P8A Mode)</td>
</tr>
<tr>
<td>PARAMETER</td>
<td></td>
<td></td>
<td>Accumulator for FW's command processor</td>
</tr>
<tr>
<td>$0578+s</td>
<td>STATEFLG</td>
<td>Φ - 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>7</td>
<td>Enable &lt;CR&gt; gen. when 1 (other 3 modes)</td>
</tr>
<tr>
<td>$05F8+s</td>
<td>CMDBYTE</td>
<td>Φ - 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>$06F8+s</td>
<td>CHNBYTE</td>
<td>Φ - 2</td>
<td>Current Apple screen slot (Comm Mode); when slot = Ø, chaining is enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 - 7</td>
<td>$CS/Ø space entry point (Comm Mode)</td>
</tr>
<tr>
<td>PWDBYTE</td>
<td></td>
<td>Φ - 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>Φ - 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>COLBYTE</td>
<td></td>
<td>Φ - 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 $C080 + (slot \times 16)$; 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>0</td>
<td>SW1-6 is OFF when 1, ON when $0$</td>
</tr>
<tr>
<td>(SW1-x)</td>
<td></td>
<td>1</td>
<td>SW1-5 is OFF when 1, ON when $0$</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>0</td>
<td>Clear To Send (CTS) is true (−) when $0$</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>5-7</td>
<td>same as above for SW2-2 &amp; SW2-1</td>
</tr>
<tr>
<td>$C088+s0$</td>
<td>TDREG</td>
<td>$0-7$</td>
<td>ACIA Transmit Register (write)</td>
</tr>
<tr>
<td></td>
<td>RDREG</td>
<td>$0-7$</td>
<td>ACIA Receive Register (read)</td>
</tr>
<tr>
<td>$C089+s0$</td>
<td>STATUS</td>
<td>$0$</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 $0$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Data Set Ready (DSR) true when $0$</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>$0$</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 ($0$) 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 $0$, normal mode for receiver; when 1, echo mode (but bits 2 and 3 must be $0$)</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>$0-3$</td>
<td>ACIA Control Register (read/write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>Baud rate: $$0 = 16$ times external clock; $$1 - $$P = \text{decimal in Table 2-5}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5-6</td>
<td>When 1, use baud rate generator; when 0, use external clock (not supported)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>Number of data bits: 8 (bit 5 and 6 = $0$)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>7 (5 = 1, 6 = $0$), 6 (5 = $0$, 6 = 1) or 5 (bit 5 and 6 both = 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Number of stop bits: 1 (bit 7 = $0$); 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 points 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>$FF6</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>(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 $Cs00.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Cs05</td>
<td>Input</td>
<td>anything</td>
<td>anything</td>
<td>anything</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
<td>character in</td>
</tr>
<tr>
<td>Notes:</td>
<td>Character in is from ACIA or keyboard.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$Cs07</td>
<td>Output</td>
<td>anything</td>
<td>anything</td>
<td>character out</td>
</tr>
<tr>
<td></td>
<td>On entry</td>
<td>(unchanged)</td>
<td>(unchanged)</td>
<td>(changed)</td>
</tr>
<tr>
<td>Notes:</td>
<td>Character out is transmitted through the ACIA.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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>$s0</td>
<td>anything</td>
</tr>
<tr>
<td>On entry</td>
<td>$Cs</td>
<td>$s0</td>
<td>(unchanged)</td>
<td></td>
</tr>
<tr>
<td>On exit</td>
<td>$Cs</td>
<td>$s0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notes:</td>
<td>$C800 space is enabled. Firmware initializes SSC to default values plus SW1 and SW2 selections.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| $C84D | Read | $Cs | $s0 | anything |
| On entry | $Cs | $s0 | (unchanged) |
| On exit | $Cs | $s0 |          |
| Notes: | $C800 space is enabled. Pascal returns ACIA or keyboard data in the A Register and location $678+5 with high bit cleared |

| $C9AA | Write | $Cs | $s0 | character out |
| 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.
### Table A-11. Pascal I.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>$Cs05</td>
<td>$38</td>
<td>Pascal serial/firmware card identifier (as well as BASIC input entry point)</td>
</tr>
<tr>
<td>$Cs07</td>
<td>$18</td>
<td>Pascal serial/firmware card identifier (as well as BASIC output entry point)</td>
</tr>
<tr>
<td>$Cs0B</td>
<td>$01</td>
<td>Pascal 1.1 generic signature byte ($01 = firmware card)</td>
</tr>
<tr>
<td>$Cs0C</td>
<td>$31</td>
<td>Pascal 1.1 Device Signature byte ($31 = serial or parallel I/O card #1)</td>
</tr>
<tr>
<td>$Cs11</td>
<td>$85</td>
<td>Pascal 1.1 optional routines flag (nonzero value = not supported)</td>
</tr>
<tr>
<td>$CsFF</td>
<td>$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 EQUS *
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 CHARS
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  RNDL   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  KRDST RB EQU $C010 ;KEYBOARD CLEAR
CFFF:         40  ROMSOFF EQU $CFPF ;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  TDRS2 EQU $C088 ;(+SN) TRANSMIT DATA REG (WRITE)
C08A:         47  RDRS2 EQU $C088 ;(+SN) READ DATA REG (READ)
C08B:         48  STRS2 EQU $C089 ;(+SN) STATUS REGISTER (READ)
C089:         49  RESET  EQU $C089 ;(+SN) SOFTWARE RESET (WRITE)
C08A:         50  CMDREG EQU $C08A ;(+SN) COMMAND REGISTER (R/W)
C088:         51  CTLREG 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 * CTRLREG STB << WL >> CK << BAUD RATE >>
0000: 65 *
0000: 66 * CMDFREG <<PARITY >> ECH <<XMIT>> RE DTR
0000: 67 *
0000: 68 ****************************************************************************
0000: 69 * SCREEN VARIABLES: PPC AND SIC MOCES *
0000: 70 ****************************************************************************
0538: 72 CMDFBYTE EQU $5F8-$8C0 ;HOLDS COMMAND CHARACTER (PPC & CIC)
0438: 73 HANDSHAKE EQU $4F8-$8C0 ;SIC PBA CHAR COUNTER FOR ETX/ACK
0438: 74 PARAMETER EQU $4F8-$8C0 ;ACOMMULATOR FOR CMD PARAMETER
0488: 75 STATEFLG EQU $7F8-$8C0 ;
0490: 76 * B7=CR GEN ENB FLAG B6=AFTER LC INPUT FLG
0500: 77 * B2-B0=COMMAND INTERPRETER STATES
0500: 78 * 0 0 0 0 IDLE
0500: 79 * 0 0 1 CMD CHAR RECEIVED
0500: 80 * 0 1 0 COLLECT <N> UNTIL CHAR THEN DO COMMAND
0500: 81 * 0 1 1 SKIP UNTIL SPACE, THEN GOTO STATE 4
0500: 82 * 1 0 0 E/D COMMANDS
0500: 83 * 1 0 1 UNUSED
0500: 84 * 1 1 0 WAIT UNTIL CR THEN SET STATE TO ZERO
0500: 85 * 1 1 1 WAIT UNTIL CR THEN DO PROC INDICATED BY PARM
0500: 86 *
0500: 87 * (B4-B0 DETERMINE ENQUIRE CHAR FOR PSA MODE)
0500: 88 *
0388: 89 DELAYFLG EQU $478-$8C0
0400: 90 * B7-B6=SCREEN TRANSLATION OPTIONS
0400: 91 * 0 0 LC->UC
0400: 92 * 0 1 NO TRANSLATION
0400: 93 * 1 0 LC->UC INVERSE
0400: 94 * 1 1 LC->UC, UC->UC INVERSE
0400: 95 * (1-3 WILL ALLOW LC CHARS TO PASS THRU MONITOR)
0400: 96 *
0400: 97 * B5-B4=CR DELAY 0 0 = NO DELAY
0400: 98 * B3-B2=LF DELAY 0 1 = 32 MILLISecs
0400: 99 * B1-B0=FF DELAY 1 0 = 1/4 Sec
0400: 100 * 1 1 = 2 Sec
0400: 101 *
0588: 102 STSBYTE EQU $678-$8C0 ;STATUS/IRESULT/INPUT BYTE
0638: 103 PDBYTE EQU $6F8-$8C0 ;PRINTER (FORMAT) WIDTH
0688: 104 COLBYTE EQU $778-$8C0 ;COLUMN POSITION COUNTER
0738: 105 MiscFLG EQU $7F8-$8C0 ;
0400: 106 * B7=ECHO BIT B6=TABBING OPTION ENABLE
0400: 107 * B5=LINEFEED EAT B4=PASCAL/BASIC FLAG
0400: 108 * B3=OFF ENB FLAG B2=KEYBOARD ENB
0400: 109 * B1=PP/CIC MODE B0=LF GENERATE ENB
0400: 110 *

FIRMWARE 59
**PAGE OF CODE**

0000: 112 -------------------------------------
0000: 113 * TEMP SCREEN VARS (SLOT INDEPENDENT) *
0000: 114 -------------------------------------
07F8: 115 MSLOT EQU $7F8 ;BUFFER FOR HI SLOT ADDR ($CN)
0000: 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 *
0638: 123 CHNBYTE EQU $6F8~$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 *
0000: 129 * INPUT BYTE: HIGH BIT IS SET
0000: 130 *
0000: 131 * MISCFLG: 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 -------------------------------------
FDED: 138 COUT EQU $FDED ;CHARACTER OUT (THRU CSW)
F89: 139 SETKB8 EQU $F89 ;SETS KSW TO APPLE KEYBOARD
FF58: 140 IORTS EQU $FF58 ;KNOWN "RTS" LOCATION
PCBA: 141 NXTA1 EQU $PCBA ;INCREMENT AH,L AND CMP TO A2H,L
FEB3: 142 SETSCR EQU $FEB3 ;SETS CSW TO APPLE SCREEN
FP6: 143 VIDOUT EQU $FP6 ;OUTPUT A CHAR TO APPLE SCREEN
0000: 144 CHN SSC.CN00
0000: 1 *
0000: 2 *
0000: 3 * APPLE II SSC 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 FP 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 DFB $01 ; GENERIC SIGNATURE BYTE
C70C:31 26 DFB $31 ; DEVICE SIGNATURE BYTE
C70D:8E 27 DFB >PINIT
C70E:94 28 DFB >PREAD
C70F:97 29 DFB >PWRITE
C710:9A 30 DFB >PSTATUS
C711:85 31 BENTRY STA CHARACTER
C713:86 32 STX ZPTEMP ; INPUT BUFFER INDEX
C715:8A 33 TXA ; SAVE X AND Y REGS ON STACK
C716:48 34 PHA
C717:98 35 TYA
C718:48 36 PHA
C719:08 37 PHP ; SAVE ENTRY FLAGS
C71A:78 38 SIE ; NO RUPS DURING SLOT DETERMINATION
C71B:8D FF CF 39 STA ROMSOFF ; SWITCH OUT OTHER $C800 ROMS
C718:20 58 FF 40 JSR IORTS
C721:BA 41 TSX
C722:8D 00 01 42 LDA STACK,X ; RECOVER $CN
C725:8D 38 07 43 STA MLSLOT
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:A8 50 TAY ; Y-REG WILL GENERALLY BE $NO
C730:28 51 PLP ; RESTORE RUPS
C731:50 29 52 BVC NORMIO
C733: 53 *
C733: 54 * BASIC INITIALIZATION
C733: 55 *
C733:1E 38 05 56 ASL CMDBYTE,X ; ALWAYS ENABLE COMMANDS
C736:5E 38 05 57 LSR CMDBYTE,X
C739:89 8A C0 58 LDA CMDBEG,Y ; JUST HAD A POWER-ON OR PROGRAM RESET?
C73C:29 1F 59 AND #$1F
C73E:00 05 60 BNE INIT1
C740:AF EF 61 LDA #$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 #$ENTRY
C74B:C5 36 67 CMP CSWL ; IF CSW IS ALREADY POINTING TO ENTRY,
C74D:F0 05 68 BEQ FROMIN ; THEN WE MUST HAVE COME FROM KSW
C74F:85 36 69 STA CSWL ; OTHERWISE, SET CSW TO ENTRY
C751:18 70 JMOCP UT CLC ; INDICATE WE ARE CALLED FOR OUTPUT
C752:90 08 71 BCC NORMIO ; <ALWAYS>
C754:E4 39 72 FROMIN CPX KSWH ; MAKE SURE KSW POINTS HERE
C756:00 F9 73 BNE FROMOUT ;
C758:A9 05 74 LDA #$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

FIRMWARE 61
C764:4C BF C8 83 JMP BINPUT
C767: 84 *
C767:8D B8 04 85 BOUTPUT LDA STATELG,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:46 35 89 LDX S2TEMP
C770:A5 27 90 LDA CHARACTER
C772:09 20 91 ORA #$20
C774:9D 00 02 92 STA INBUFF,X ;RESTORE LOWERCASE IN BUFFER
C777:85 27 93 STA CHARACTER ;AND FOR OUTPUT ECHO
C779:AF F8 07 94 LDX MSLOT
C77C:68 95 BOUTPUT1 PLA
C77D:29 BF 96 AND #$BF ;ZERO THE FLAG
C77F:9D B8 04 97 STA STATELG,X
C782:28 98 PLP ;RETRIEVE CIC MODE INDICATION
C783:F0 06 99 BEQ BOUTPUT2 ;BRANCH FOR PPC, SIC MODES
C785:20 63 C8 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
C78B: 104 ******************************************************
C78E: 105 *
C78E: 106 * NEW PASCAL INTERFACE ENTRIES *
C78E: 107 *
C78E: 108 ******************************************************
C78E:20 00 C8 109 FINIT JSR PASCALINIT ;
C791:22 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 ;
C798: 114 *
C799: 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 0B 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 0F 126 BCC PSTATUS2 ;CARRY CLEAR FOR NOT READY
C7AB: 127 *
C7AB:20 82 03 128 PSTATIN JSR SRIN ;SETS CARRY CORRECTLY
C7AB:3D B8 05 129 PSTATUS2 LDA STSBYTE,X ;GET ERROR FLAGS
C7AC: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
C788:E5 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
C7D0:48 159 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
C7DF:0A 170 ASL A
C7EF: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 ROMSOF
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
C7F1:88 185 DNX
C7F2:04 186 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:CC C5
C7FF:08 193 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
C900:20 9B C9 19 PASCALINIT JSR PENTRY ;PASCAL 1.0 INITIALIZATION ENTRY
C903:A9 16 20 LDA #$16 ;NO XOFF, ECHO, LF EAT, OR LF GEN
C905:48 21 INIT1 PHA ;GOES TO MISCEFLG AFTER MODIFICATION
C906:A9 00 22 LDA #0
C908:9D B8 04 23 STA STATEFLG,X
C90B:9D B8 03 24 STA DELAYFLG,X
C90E:9D 38 04 25 STA HANDSHKE,X
C911:9D B8 05 26 STA STBYTE,X
C914:9D 38 06 27 STA PWBYTE,X
C917:9D B8 06 28 STA COLBYTE,X
C91A:B9 82 C0 29 LDA DIPSW2,Y ;SET LP GEN OPTION FROM D2-S5
C91D:85 2B 30 STA ZPTMP2 ;SAVE FOR LATER
C91F:4A 31 LSR A ;S5-> CARRY
C920:4A 32 LSR A ;IF S5=ON=0 THEN LEAVE MISCEFLG ALONE
C921:90 04 33 BCC INITIA
C923:68 34 PLA ;OTHERWISE, MAKE SURE LP GEN
C924:29 FE 35 AND #$FE ;ENABLE IS RESET
C926:48 36 PHA ;
C927:B8 37 INITIA CLV ;V WILL BE CLEAR FOR CIC MODE
C928:B9 81 C0 38 LDA DIPSW1,Y
C92B:4A 39 LSR A ;SIC MODES SET CARRY
C92C:B0 07 40 BCS INIT2 ;BRANCH FOR SIC MODES
C92D:4A 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 P8A
C936:A9 03 47 LDA #$03 ;SET ETX AS DEFAULT INQUIRY CHAR
C938:B0 02 48 BCS INIT2A ;BRANCH FOR P8A
C93A:A9 80 49 LDA #$80 ;FOR P8 SET AUTO CR GEN
C93C:9D B8 04 50 INIT2A STA STATEFLG,X
C93F:2C 58 FF 51 INIT2B BIT IORTS ;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 *
C840:70 0A 57 BVS 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 PREAD O JSR PASCALREAD ;DO PASCAL 1.1 READ
C850:AE F8 07 63 LDX MSLOT ;MODIFY FOR 1.0
C853:9D B8 05 64 STA STHBYTE,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 #TMP2 ;PPC, SIC MURES USE SWITCHES
C859:4A 71 LSR A ;TO SET PWIDTH, CR DELAY
C85A:4A 72 LSR A
C85B:29 03 73 AND #$3
C85D:4A 74 TAY
C85E:00 04 75 BEQ INIT4
C860:68 76 *
C860:68 77 PLA ;RESET VIDEO ENABLE FOR PWIDTH#40
C861:29 7F 78 AND #$7F
C863:48 79 PHA
C864: 80 *
C864:A9 C9 81 INIT4 LDA PWDTBL,X
C867:9D 38 06 82 STA PWDBYTE,X
C86A:44 26 83 LDY SLOTT6
C86C: 84 *
C86C:68 85 PLA ;CLEAR CIC BIT IN FUTURE MISCEPFL
C86D:29 95 86 AND #$95 ;(AND TABbing, XOFF AND LF EAT BITS)
C86F:48 87 PHA
C870:49 09 88 LDA #$09 ;CTL-I
C872: 89 *
C872:9D 38 05 90 INIT5 STA CMDBYTE,X ;CMD ESC CHAR (IGNORED FOR SIC MURES)
C875:68 91 PLA
C876:9D 38 07 92 STA MISCEPFL,X ;SET MISCEPFL FLGS
C879: 93 *
C879: 94 * NOW FOR THE ACIA INITIALIZATION ROUTINE
C879: 95 *
C879:A5 2B 96 INITACIA LDA ZTPMP2 ;DIPSW2
C87B:48 97 PHA
C87C:29 A0 98 AND #$A0 ;DATA BIT OPTIONS FOR CIC MODE
C87E:50 02 99 BVC INITACIA1 ;BRANCH FOR CIC MODE
C880:29 80 100 AND #$80 ;8 DATA, 1 OR 2 STOP FOR SIC, PPC
C882:20 A1 CD 101 INITACIA1 JSR DATACMD1 ;SET CONTROL REG
C885:20 81 CD 102 JSR BAUDCMD1 ;SET DIPSWITCH BAUD RATE
C888:68 103 PLA
C889:29 0C 104 AND #$OC ;PARITY OPTIONS FOR CIC MODE
C88B:50 02 105 BVC INITACIA2 ;BRANCH FOR CIC MODE
C88D: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
C894:9A C0 111 STA CMDREG,Y
C897:B9 88 C0 112 LDA RDREG,Y ;THROW OUT THE STRANGE STUFF
C89A:60 113 RTS
C89B: 114 **********************
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 FB 07 120 PASEXIT LDY MSLOT
C8A6: BE B0 05 121 LDX STSBYTE,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 * TURND IN THE A-REGISTER *
C8AA: 131 ****************************************
C8AA: 20 FF CA 132 GETCHAR JSR INPUT ;ACIA DATA?
C8AD: B0 05 133 BCS GETCHAR1
C8AF: 20 2C CC 134 JSR CKKRD ;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
C8B5: 2 ****************************
C8B5: 3 *
C8B5: 4 * APPLE II SSC FIRMWARE *
C8B5: 5 *
C8B5: 6 * BY LARRY KENYON *
C8B5: 7 *
C8B5: 8 * -FEBRUARY 1981- ****************************
C8B5: 9 *
C8B5: 10 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC. *
C8B5: 11 *
C8B5: 12 ****************************
C8B5: 13 *
C8B5: 14 * CIC, SIC, PPC MODE HIGH-LEVEL *
C8B5: 15 *
C8B5: 16 ****************************
C8B5: 17 * CIC EXIT ROUTINE . . *
C8B5: 18 ****************************

C8B5:20 1E CA 19 CICEXIT JSR CHECKTERM ;SEE IF WE'VE ENTERED TERMINAL MODE
C8B8: 20 **********************
C8B8: 21 * BASIC EXIT ROUTINE *
C8B9: 22 **********************
C8BB:68 23 BASICEIXT PLA
C8BD:A8 24 TAY
C8BB:68 25 PLA
C8BD:AA 26 TAX
C8BC:A5 27 LDA CHARACTER
C8BB:60 28 RTS
C8BF: 29 **********************
C8BF: 30 * BASIC INPUT ROUTINE *
C8BF: 31 **********************
C8BF:F0 29 32 BINPUT BEQ BINACIA ;BRANCH IF NOT CIC MODE
C8C1:BD B8 06 33 LDA BUFBYTE,X ;INPUT BUFFER FULL?
C8C4:10 05 34 BPL BINKBD
C8C6:5E B8 06 35 LSR BUFBYTE,X ;RESET BUFFER FULL
C8C9:DO 24 36 BNE BINACIA1 ;<ALWAYS>
C8C8: 37 *
C8CB:20 3E CC 38 BINKBD JSR GETKBDR ;KEYBOARD DATA?
C8CC:90 1A 39 BCC BINACIA
C8DD: 40 *
C8DD:BD B8 03 41 BINEND LDA DELAYFLG,X
C8D3:29 CO 42 AND #$C0 ;TRANSLATE LOWERCASE TO UPPERCASE?
C8D5:F0 0E 43 BEQ BINEND1 ;IF SO, LET THE MONITOR DO IT
C8D7:A5 27 44 LDA CHARACTER ;IF NOT, SET FLAG IF
C8D9:C9 00 45 CMP #$80 ;THIS IS A LOWERCASE CHAR
C8DB:90 08 46 BCC BINEND1 ;FOR INPUT BUFFER CORRECTION
C8DD:BD B8 04 47 LDA STATEFLG,X ; (CIRCUITVENT APPLE MONITOR)
C8DE:09 40 48 ORA #$40
C8E2:9D B8 04 49 STA STATEFLG,X
C8E5: 50 *
C8E5:28 51 BINEND1 PLP
C8EE:F0 DO 52 BEQ BASICEIXT ;BRANCH IF NOT CIC MODE
C8ED:00 CB 53 BNE CICEXIT ;<ALWAYS> CHECK TO SEE IF WE
C8EA: 54 * ENTERED TERM MODE (VIA KYBD ESCAPE
C8EA:20 FF CA 55 BINACIA JSR INPUT ;ACIA DATA?
C8ED:90 DC 56 BCC BINKBD
C8EF:20 11 CC 57 BINACIA1 JSR RESTORE ;DO BASIC CURSED DUTY
C8F2:28 58 PLP
C8F3:08 59 PHP ;GET CIC MODE INDICATOR
CBF4:F0 DA  60  BEQ  BINEND  ;SKIP IF NOT CIC MODE
CBF6:20 D1 C9  61  JSR  CINPUT  ;LOOK FOR INPUT STREAM SPECIAL CHARS
CBF9:4C D0 C8  62  JMP  BINEND  ;
CBFC:  63  ************************************
CBF8:  64  * SIC, PPC BASIC OUTPUT ROUTINE *
CBF9:  65  ************************************
CBFC:20 1A CB  66  SEROUT  JSR  CMDSQCK  ;CHECK FOR A COMMAND SEQUENCE
CBFP:80 87  67  BCS  BASICEXIT  ;BRANCH IF WE WERE IN COMMAND MODE
C901:A5 27  68  LDA  CHARACTER  ;SAVE CHAR ON STACK
C903:48  69  PHA
C904:BD 38 07  70  LDA  MISCPLG,X  ;IF VIDEO OR TABLING ENABLED,
C907:29 C0  71  AND  #$CO  ; DON'T MESS WITH THE CURSOR
C909:00 16  72  BNE  TARCHCK
C90B:  73  *
C90B:A5 24  74  LDA  CH  ;CHECK FOR COMMA TABLING
C90D:00 42  75  BEQ  NOTAB  ;IF CH=0, THERE WAS NO TAB OR COMMA
C90F:C9 08  76  CMP  #$8  ;INTEGER BASIC COMMA?
C911:F0 04  77  BEQ  COMMA
C913:C9 10  78  CMP  #$16  ;APPLESOFT COMMA?
C915:D0 0A  79  BNE  TARCHCK
C917:09 F0  80  COMMA  ORA  #$F0
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  MISCPLG,X  ;DON'T BACKSPACE UNLESS TABBING
C92F:0A  93  BPL  NOTAB
C932:A9 B8  94  ASL  A  ; OPTION IS ENABLED
C933:85 27  95  TAB1  STA  CHARACTER
C936:2C 58 FF  96  BIT  IORTS  ;SET V=1 TO INDICATE TABBING
C939:08  97  PHP  ;SAVE TABBING INDICATOR
C93A:70 0C  98  BVS  TAB2  ;<ALWAYS> AROUND BATCH MOVE ENTRY
C93C:EA  99  NOP
C93D:  100  ************************************
C93D:  101  * SHORT BATCH MOVE:  *
C93D:  102  * LOCATE AT $C93D FOR  *
C93D:  103  * COMPATIBILITY WITH  *
C93D:  104  * SIC PB BLOCK MOVE.  *
C93D:  105  ************************************
C93D:2C 58 FF  106  BATCHIN BIT IORTS
C940:50  107  DFB  $50  ;DUMMY BVC
C941:B8  108  BATHOUT CLV  ;V=0 FOR OUTPUT ENTRY
C942:AE F8 07  109  LDX  $MSLOT
C945:4C EF C9  110  JMP  BATHTO
C948:  111  ************************************
C94B:  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 TABBING
C94E:4C 68 CB  117  JMP  FORCECR  ;SHARE SOME CODE. .

68 SUPER SERIAL CARD
C951: 118 *
C951:68 119 NOTAB PLA
C952:B8 120 CLV
C953:08 121 PHP ;SAVE 'NO TAB' INDICATION
C954:85 27 122 NOTAB1 STA CHARACTER ;(FORCE CR RETRY)
C954:48 123 PHA
C957:20 68 CB 124 JSR OUTPUT1 ;ENTER AFTER CMD SEQ CHECK
C95A:20 B5 C9 125 JSR ADJUST
C95D:68 126 PLA
C95E:49 B0 127 EOR #$8D ;WAS IT A CR?
C960:0A 128 ASL A
C961:00 05 129 BNE FORCER
C963:9D B8 06 130 STA COLBYTE,X ;IF SO, RESET COLUMN TO 0
C966:85 24 131 STA CH
C968: 132 *
C968:BD B8 04 133 FORCER LDA STATEFLG,X ;FORCE CR DISABLED?
C96B:10 0D 134 BPL SEREND
C96D:BD 38 06 135 LDA PWDBYTE,X ;FORCE CR IF LIMIT REACHED
C970:F0 08 136 BEQ SEREND ;(FOR P8 POKE COMPATIBILITY)
C972:18 137 CLC
C973:FD B8 06 138 SBC COLBYTE,X
C976:A9 8D 139 LDA #$8D
C978:90 DA 140 BCC NOTAB1 ;BRANCH TO FORCE CR
C97A: 141 *
C97A:28 142 SEREND PLP
C97B:70 A4 143 BVS TABCHECK ;BRANCH IF TABBING
C97D: 144 *
C97D:BD 38 07 145 LDA MISCFLG,X ;DON'T MESS WITH CURSOR
C980:30 16 146 BMI SEREND2 ; WHEN VIDEO IS ON
C982:BC B8 06 147 LDY COLBYTE,X
C985:0A 148 ASL A
C986:30 0E 149 BMI SETCH ;SET CH TO VALUE OF COL FOR TABBING
C988:98 150 TIA
C989:A0 00 151 LDY #0
C98B:38 152 SBC
C98C:FD 38 06 153 SBC PWDBYTE,X ;
C98F:C9 F8 154 CMP #$F8 ;WITHIN 8 CHAR OF PWIDTH?
C991:90 03 155 BCC SETCH
C993:69 27 156 ADC #$27 ;IF SO, ADJUST TO WITHIN 8 OF 40
C995:A8 157 TAY
C996:84 24 158 SETCH STY CH
C998: 159 *
C998:4C B8 C8 160 SEREND2 JMP BASICEXIT ;THAT'S ALL
C99B: 161 *
C99B: 162 ***********************
C99B: 163 * PASCAL ENTRY ROUTINE *
C99B: 164 ***********************
C99B:8E F8 07 165 PENTRY STX MSLOT
C99B:84 26 166 STY SLOT16
C9A0:A9 00 167 LDA #0
C9A2:9D B8 05 168 STA STSBYTE,X
C9A5:60 169 RTS
C9A6: 170 *
C9A6: 171 ***********************
C9A6: 172 * SIC MODE PRINTER WIDTH TABLE *
C9A6: 173 ***********************
C9A6:29 174 PWDTBL DFB $29 ;40 COLUMNS
C9A7:48 175 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 CB  186  JSR OUTPUT
C9B2:4C A3 CB  187  JMP PASEXIT ;LOAD X-REG WITH ERROR BYTE & RTS
C9B5:  188 *
C9B5:  189 ***********************
C9B5:  190  * COLUMN ADJUST ROUTINE *
C9B5:  191  * (PPI, SIC MODES ONLY) *
C9B5:  192 ***********************
C9B5:A5  27  193 ADJUST LDA CHARACTER
C9B7:49  08  194  EOR #$08 ;BACKSPACE?
C9B9:0A  95  195  ASL A
C9BA:F0  04  196  BEQ DECRCOL ;IF SO, DECREMENT COLUMN
C9BC:49 EE  197  EOR #$EE ;DELETE? ($FF, RUB)
C9BE:00  09  198  BNE CTRLTST
C9C0:DE B8 06  199  DECRCOL DEC COLBYTE,X ;DECREMENT COLUMN COUNT
C9C3:10  03  200  BPL ADJRTS
C9C5:9D 88 06  201  STA COLBYTE,X ;DON'T ALLOW TO GO BELOW 0
C9C8:60  202  ADJRTS RTS
C9C9:C9  C0  203  CTRLTST CMP #$C0 ;DON'T INCREMENT COLUMN COUNT FOR
C9CB:60 FB  204  BCS ADJRTS ; CONTROL CHARACTERS
C9CD:FE B8 06  205  INC COLBYTE,X
C9DD:60  206  RTS
C9D1:  207 ***********************
C9D1:  208  * ROUTINE TO PROCESS SPECIAL INPUT CHAR *
C9D1:  209 ***********************
C9D1:BD 38 07 210 CKINPUT LDA MISCFLG,X
C9D4:29 08  211  AND #$08 ;INPUT CTL CHAR S ENABLED?
C9D6:0F 16  212  BEQ CIEND
C9D8:  213 *
C9D8:BD 88 04  214  LDA STATEFLG,X
C9DB:A4  27  215  LDY CHARACTER
C9DD:0C  94  216  CPY #$94 ;CTL-T?
C9DF:0D  04  217  BNE CKINPUT1
C9E1:09  80  218  ORA #$80 ;SET TERMINAL MODE
C9E3:0D  06  219  BNE CKINPUT2 ;<ALWAYS>
C9E5:  220 *
C9E5:C0  92  221  CKINPUT1 CPY #$92 ;CONTROL-R?
C9E7:0D  05  222  BNE CIEND
C9E9:29  7F  223  AND #$7F ;RESET TERMINAL MODE
C9EB:9D B8 04  224  CKINPUT2 STA STATEFLG,X
C9EE:60  225  CIEND RTS
C9EF:  226 *
C9EF:  228  CHN  SSC_TERMD
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:         14  BATCHIO TMA
C9EF:         15  ASL A
C9EF:         16  ASL A
C9EF:         17  ASL A
C9EF:         18  ASL A
C9EF:         19  STA SLOT16
C9EF:         20  LDA #$0
C9EF:         21  STA STBSY, X ;ZERO ERROR INDICATION
C9EF:         22  BVS MOVIN
C9EF:         23  *
C9EF:         24  MOVOOUT LDY #$0
C9EF:         25  LDA (A1L), Y ;GET BUFFER DATA
C9EF:         26  STA CHARACTER
C9EF:         27  JSR ACIAOUT ;SEND IT OUT THE ACIA
C9EF:         28  JSR NXTA1
C9EF:         29  BCC MOVOUT
C9EF:         30  RTS
C9EF:         31  *
C9EF:         32  MOVIN JSR SRIN
C9EF:         33  BCC MOVIN
C9EF:         34  LDA RDREG, Y
C9EF:         35  LDY #$0
C9EF:         36  STA (A1L), Y ;PUT ACIA DATA INTO BUFFER
C9EF:         37  JSR NXTA1
C9EF:         38  BCC MOVIN
C9EF:         39  RTS
C9EF:         40  *
C9EF:         41  ********************************************
C9EF:         42  *
C9EF:         43  * TERMINAL MODE ROUTINES  *
C9EF:         44  *
C9EF:         45  ********************************************
C9EF:         46  CHECKTERM LDA STATEPLG, X ;HAVE WE ENTERED TERMINAL MODE?
C9EF:         47  BPL TERMINTS ;IF NOT, A SIMPLE RTS WILL DO.  ...
C9EF:         48  *
C9EF:         49  * WE ENTER THE WORLD OF TERMINAL MODE
C9EF:         50  *
C9EF:         51  TERMMODE LDA #$02 ;START IN SHIPT-LOCK STATE
C9EF:         52  PHA ;SHIFT STATE IS SAVED ON STACK
C9EF:         53  LDA #$7F
C9EF:         54  JSR KCMD1 ;RESET ECHO (DEFAULT TO FULL DUP)
C9EF:         55  *
C9EF:         56  TERMNEXT LDY CH
C9EF:         57  LDA (BASL), Y

FIRMWARE 71
CA2F:85 27 STA CHARACTER ;SAVE SCREEN CHARACTER
CA31:A9 07 TERMNEXT1 LDA #$07 ;IMPLEMENT A FLASHING UNDERLINE
CA33:25 4F AND RNDH ;FOR A CURSOR
CA35:D0 10 BNE TERMNEXT3
CA37:A4 24 LDY CH
CA39:A9 0F LDA #$DF
CA3B:D1 28 CMP (BASL),Y ;IS UNDERLINE ON THE SCREEN?
CA3D:00 02 BNE TERMNEXT2 ;IF NOT, PUT IT THERE
CA3F:A5 27 LDA CHARACTER ;OTHERWISE USE TRUE SCREEN CHAR
CA41:91 28 67 TERMNEXT2 STA (BASL),Y
CA43:E6 4F INC RNDH ;MAKE IT FLASH, BUT
CA45:E6 4F INC RNDH ;NOT TOO SLOW AND NOT TOO FAST
CA47: 70 *
CA47:BD B8 04 71 TERMNEXT3 LDA STATEFLG,X ;ARE WE STILL IN TERM MODE?
CA4A:30 09 BMI TERRACIAIN ;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 TERMRTS RTS
CA55: 79 *
CA55:20 FF CA 80 TERRACIAIN JSR INPUT ;ACIA INPUT?
CA58:90 0C 81 BCC TERMKBW ;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 CTR-L, CTR-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 TERMKBW JSR GETKBW ;KEYPRESS?
CA69:90 C6 88 BCC TERMNEXT1 ;SKIP IF NOT
CA6B:70 BE 89 BVS TERMNEXT ;BRANCH IF WE DID A KBM ESCAPE SEQ.
CA6D:BD 38 07 90 LDA MISCBFLG,X ;SHIFTING ENABLED?
CA70:0A 91 ASL A
CA71:10 22 92 BPL TERMSEND1
CA73:6B 93 PLA ;RECOVER TERMSTATE
CA74:A8 94 TAY
CA75:A5 27 95 LDA CHARACTER
CA77:CO 01 96 CPY #1 ;1 = SHIFT LETTERS, XLAPE NUMBERS
CA79:F0 20 97 BEQ TERMCAP
CA7B:B0 34 98 BCS TERMLOCK ;2 MEANS CAPS LOCK MODE
CA7D: 99 *
CA7D:C9 9B 100 TERMNORM CMP #$9B ;ESC?
CA7F:D0 06 101 BNE TERMLETTER
CA81: 102 *
CA81:C8 103 TERMINC INY ;INCREMENT STATE
CA82:98 104 TERMINC1 TYA
CA83:48 105 PHA ;PUT BACK ON STACK
CA84:4C 2B CA 106 JMP TERMNEXT
CA87: 107 *
CA87:C9 C1 108 TERMLETTER CMP #$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
CA9B: 119 *
CA9B:C9 9B 120 TERMCAP CMP #$9B ;TWO ESCAPES?
CA9D:F0 E2 121 BEQ TERMINC
CA9F:C9 0B 122 CMP #$B0 ;<0?
CAA1:90 0A 123 BCC TERMCAP1
CAA3:C9 BB 124 CMP #$BB ;>COLON?
CAA5:BO 06 125 BCS TERMCAP1
CA7: 126 *
CA7:A8 129 TAY
CA88:B9 09 CA 130 LDA TRANSLATE-$BO,Y
CA88:85 27 131 STA CHARACTER
CA98:A0 00 132 TERMCAP1 LDY #0 ;BACK TO STATE 0
CAA8:F0 E2 133 BEQ TERMSEND ;<ALWAYS>
CAB1: 134 *
CAB1:C9 9B 135 TERMLQCK CMP #$9B ;ESC?
CAB3:D0 DE 136 BNE TERMSEND
CAB5:A0 00 137 LDY #0
CAB7:F0 C9 138 BBQ 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
CABC:DC 147 DFB $DC ;LEFT SLASH
CABE:DF 148 DFB $DF ;UNDERSCORE
CABF:FB 149 DFB $FB ;LEFT ENCLOS
CAC0:FC 150 DFB $FC ;VERTICAL BAR
CAC1:FD 151 DFB $FD ;RIGHT ENCLOS
CAC2:FE 152 DFB $FE ;TILDE
CAC3:FF 153 DFB $FF ;RUB
CAC4: 154 *
CAC4: 155 CHN SSC.CORE
CAC4:  2 ********************************************
CAC4:  3 *
CAC4:  4 * APPLE II SSC FIRMWARE *
CAC4:  5 *
CAC4:  6 * BY LARRY KENYON *
CAC4:  7 *
CAC4:  8 * -JANUARY 1981- *
CAC4:  9 *
CAC4: 10 * (C) COPYRIGHT 1981 BY APPLE COMPUTER, INC.*
CAC4: 11 *
CAC4: 12 **************************************
CAC4: 13 *
CAC4: 14 * CORE SUBROUTINES *
CAC4: 15 *
CAC4: 16 **************************************
CAC4: 17 *
CAC4: 18 * GENERAL PURPOSE WAIT ROUTINE *
CAC4: 19 **************************************
CAC4: 20 *
CAC4: 21 * WAITMS WAITS FOR [A-REG] MILLISECONDS (256 IF A-REG=0)
CAC4: 22 *
CAC4:A2 CA 23 WAITMS LDX #202
CAC6:CA 24 WAITMS1 DEX ;<DON'T LET THIS LOOP CROSS A PAGE>
CANC:D0 PD 25 BNE WAITMS1 ;5 MICROSECOND LOOP
CANC9:38 26 SEC
CACA:E9 01 27 SBC #01
CACC:D0 P6 28 BNE WAIMS
CACE:A8 F8 07 29 LDX MLSLOT
CAD1:60 30 RTS
CAD2: 31 ******************************************
CAD2: 32 * ACIA STATUS REGISTER READ ROUTINES *
CAD2: 33 ******************************************
CAD2: 34 *
CAD2: 35 * SRIN USED TO CHECK ACIA INPUT STATUS
CAD2: 36 *
CAD2:A4 26 37 SRIN LDY SLOT16 ;SLOT16=$00
CAD4:B9 89 C0 38 LDA SREG, Y
CAD7:48 39 PHA
CAD8:29 20 40 AND #$20 ;DCDP?
CADA:4A 41 LSR A ;AN ERROR IF NOT
CADB:4A 42 LSR A
CADC:85 35 43 STA ZPTEMP
CADE:68 44 PLA
CADEF:29 0F 45 AND #$0F
CAEA1:C9 08 46 CMP #$08 ;SET CARRY IF RDR FULL, ELSE CLEAR
CAEA3:90 04 47 BCC SRIN1
CAEA5:29 07 48 AND #$07 ;PE, PE, OVR VALID ONLY WHEN RDR=1
CAEA7:B0 02 49 BCS SRIN2 ;<AWAYS>
CAEA9:A5 35 50 SRIN1 LDA ZPTEMP
CAEB:05 35 51 SRIN2 ORA ZPTEMP ;GET DCD ERROR BIT
CAED:F0 05 52 BEQ SRIN3 ;BRANCH IF NO ERRORS FOUND
CAEF:09 20 53 ORA #$20 ;ELSE SET BIT 5 TO OFFSET FOR PASCAL
CAF1:9D B8 05 54 STA STBYTE,X ;AND SAVE IN STATUS TEMP
CAF4:60 55 SRIN3 RTS ;CY=1 MEANS DATA IS AVAILABLE
CAF5: 56 *
CAF5: 57 * SROUT CHECKS IF TOR IS EMPTY + HARDWARE HANDSHAKE IS OK
CAF5: 58 *
CAF5:A4 26 59 SROUT LDY SLOT16

74 SUPER SERIAL CARD
CAF7:B9 89 C0 60 LDA STREG, Y
CAF8:29 70 61 AND #$70
CAF8:C9 10 62 CMP #$10 ;EQU IF TDR EMPTY, IOC, DSR, & CTS
CAF8:60 63 RTS
CAF9: 64 *
CAF9: 65 ***********************
CAF9: 66 * GENERAL INPUT ROUTINE *
CAF9: 67 ***********************
CAF9:20 D2 CA 68 INPUT JSR SRIN
CB02:90 15 69 BCC NOINPUT1
CB04: 70 *
CB04:B9 88 C0 71 LDA RREG, Y ;GET THE ACIA INPUT
CB07:09 80 72 ORA #$80 ;SET HI BIT FOR BASIC
CB09:C9 8A 73 CMP #$8A ;LINEFEED?
CB0B:00 09 74 BNE INPUT2
CB0D: 75 *
CB0D:A8 76 TAY
CB0E:BD 38 07 77 LDA MISCFLG,X ;SEE IF WE SHOULD EAT IT
CB11:29 20 78 AND #$20
CB13:D0 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
CB20:80 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 *
CB2E:AD 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:DD 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 MISCFLG,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 MISCLFLG,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 88 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 09 133 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 01 140 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 C0 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 CHAR
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:BD 22 165 BCS P8AOUT3 ;SO NO HANDSHAKE IS NEEDED YET
CB84:C9 68 166 CMP #107 ;SET CARRY IF 107 (149 SENT)
CB86:68 167 PLA
CB87:48 168 PHA
CB88:49 9B 169 EOR #$9B ;ESC?
CB8A:29 7F 170 AND #$7F ;IGNORE HI-BIT
CB8C:D0 18 171 BNE P8AOUT2 ;COUNT AS 1 OF 5 IF NOT 'ESC'
CB8E:BD 19 172 BCS P8AOUT3 ;DON'T COUNT IF 149TH CHAR IS 'ESC'
CB90: 173 *
CB90:BD B8 04 174 ETX LDA STATEPLG,X ;SEND QUERY CHAR TO PRINTER
CB93:29 1F 175 AND #$1F ;DEFAULT IS ETX

76 SUPER SERIAL CARD
CB95:09 80  176  ORA  $80
CB97:85 27  177  STA  CHARACTER
CB99:20 02 CC  178  JSR  ACLAOUT
CB9C:20 AA C8  179  ACK  JSR  GETCHAR ;GET ACIA/KBD DATA
CB9F:49 86  180  EOR  $86  ;ACK?
CBAD:DD 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
CBA9:68  185  P8AOUT3 PLA  ;GET REAL CHAR TO OUTPUT
CBAE:85 27  186  STA  CHARACTER
CBB:49 8D  187  EOR  $8D  ;IP CR AND CR DELAY MODE
CBB:0A  188  ASL  A
CBBF:D0 0A  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
CBB6:0F 03  192  BEQ  P8AOUT4
CBB8:9D 38 04  193  STA  HANDSHKE,X
CBBB:  194 *
CBBB:20 02 CC  195  P8AOUT4 JSR  ACLAOUT
CBBE:4C EA CB  196  JMP  LFGEN ;(SKIP DELAYS)
CBCE:  197 **********************
CBCE:  198 * AND BACK TO NORMAL OUTPUT *
CBCE:  199 **********************
CBCE:12 02 CC  200  OUTPUT3 JSR  ACLAOUT ;OUTPUT THE CHARACTER
CBE0:  201 *
CBE0:  202 * NOW CHECK FOR CR, LF, AND FF DELAYS
CBE0:  203 *
CBE4:0A  204  ASL  A
CBE5:AB  205  TAY
CBE6:BD B8 03  206  LDA  DELAYFLG,X ;GET DELAY FLAGS
CBE9:CO 18  207  CPY  $18  ;FORM FEED?
CBEB:F0 0C  208  BEQ  OUTDLY1
CBED:4A  209  LSR  A
CBEE:4A  210  LSR  A  ;RIGHT JUSTIFY LF DELAY
CBEF:CO 14  211  CPY  $14  ;LINE FEED?
CBF0:FO 06  212  BEQ  OUTDLY1
CBF3:4A  213  LSR  A
CBF4:4A  214  LSR  A  ;RIGHT JUSTIFY CR DELAY
CBF5:CO 1A  215  CPY  $1A  ;CARRIAGE RETURN?
CBF7:DD 25  216  BNE  OUTDYP
CBF9:29 03  217  OUTDLY1 AND  $03  ;JUST WANT LOWEST 2 BITS
CBFB:FO 0D  218  BEQ  LFGEN  ;NO DELAY INDICATED
CBFD:AB  219  TAY
CBFE:BD FE CB  220  LDA  DLYTLB-1,Y
CBF1:AB  221  TAY  ;DELAY IN 32 USEC INCREMENTS
CBF2:98 27  222  OUTDYLp LDA  $32  
CBB4:20 C4 CA  223  JSR  WAITSMS
CBB7:8B  224  DEY
CBB8:D0 F8  225  BNE  OUTDLYLP
CBEB:  226 *
CBEB:  227 * CHECK ON LF GENERATION OPTION
CBEB:  228 *
CBEB:AE 27  229  LFGEN  LDA  CHARACTER
CBEC:0A  230  ASL  A
CBED:C9 1A  231  CMP  $1A  ;CARRIAGE RETURN?
CBEF:D0 OD  232  BNE  OUTDYP
CBF1: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 F5 CA 247  ACIAOUT JSR SROUT ;READY FOR OUTPUT?
CC05:D0 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:5 A 27 252  LDA CHARACTER ; ON PAGE $8F (AVOIDING RDR READ)
CC0D:99 FF BF 253  STA $BF7F,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 24 261  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 95 266  CMP #$95 ;SCREEN PICK?
CC1B:0C 267  BNE RESTOREND
CC1D:A5 27 268  LDA CHARACTER ;IF SO, USE SCREEN CHAR
CC1F:C9 20 269  CMP #$20 ;INVERSE?
CC21:80 06 270  BCS RESTOREND
CC23:20 FF CC 271  JSR GETXDATE ;REVERSE THE TRANSLATION
CC26:59 DB CC 272  EOR REVMAK,Y
CC29:85 27 273  RESTOREND STA CHARACTER
CC2B:60 274  RTS
CC2C: 275  *
CC2C: 276  CHN SSC.UTIL

78  SUPER SERIAL CARD
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:                       19  CCKBD  CLC  ;RETURN CARRY CLEAR FOR NO DATA
CC2D:                     20  LDA  MISCEFLG,X
CC30:                     21  AND  #$04  ;ANSWER NO IF KEYBOARD IS DISABLED
CC32:                   22  BEQ  CCKBDXIT
CC34:                      23  *
CC34:                   24  CCKBD1  LDA  KBD
CC37:                  25  BPL  CCKBDXIT
CC39:                 26  STA  KBDSTRB
CC3C:               27  SBC  ;INDICATE DATA
CC3D:            28  CCKBDXIT  RTS
CC3E:                29  ***************************************
CC3E:                30  * GET A CHAR FROM KEYBOARD FOR BASIC ONLY *
CC3E:                31  ***************************************
CC38:            32  GETKBD  INC  RNDL  ;MIX UP RANDOM # SEED
CC40:            33  BNE  GETKBD1  ;FOR BASIC
CC42:            34  INC  RNDH
CC44:            35  GETKBD1  JSR  CCKBD  ;KEYBOARD FETCH ROUTINE
CC47:            36  CLV  ;INDICATE NO ESCAPE SEQUENCE
CC48:            37  SCC  CCKBDXIT  ;EXIT IF NO KEY PRESS
CC4A:            38  JSR  RESTORE  ;DO BASIC CURSED DUTY
CC4D:            39  AND  #$7F
CC4F:            40  CMP  CMDBYTE,X  ;IS IT THE START OF A COMMAND?
CC52:            41  BNE  GETKBDONE  ;IF NOT, EXIT INDICATING DATA
CC54:            42  LDY  SLOT16
CC56:            43  LDA  DIPSW1,Y  ;ONLY DO CMD ESC FOR PPC, SIC MODES
CC59:            44  LSR  A
CC5A:            45  BCS  GETKBDONE
CC5C:            46  ***************************************
CC5C:            47  * KEYBOARD ESCAPE HANDLER *
CC5C:            48  ***************************************
CC5C:            49  KBDESC  LDY  #$A  ;FIRST PRINT A PROMPT
CC5E:          50  PROMPTLOP  LDA  PROMPTBL,Y
CC61:          51  STA  CHARACTER
CC63:          52  TYA
CC64:          53  PHA
CC65:          54  JSR  SCREENOUT1  ;ALWAYS SEND TO SCREEN
CC68:          55  PLA
CC69:          56  TAY
CC6A:          57  DEY
CC6B:          58  BPL  PROMPTLOOP
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 CKKMD1 ;WAIT FOR KEYBOARD CHARACTER
CC75:10 FB 64 SPL GETCMD
CC77:C9 88 65 CMP #$88 ;BACKSPACE?
CC79:FO 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:DO E8 74 BNE GETCMD ;IF NOT, GO AGAIN
CC8A: 75 *
CC8A:99 8D 76 LDA #$8D ;FORCE BACK A CARRIAGE RETURN
CC8C:05 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 MISCLFLG,X
CCA1:10 13 90 BPL NOOUT ;IF SCREEN DISABLED
CCA3: 91 *
CCA3:BD 38 07 92 SCREENOUT1 LDA MISCLFLG,X ;ENTRY AFTER ECHO CHECK
CCA6:29 02 93 AND #$02 ;IF IT ISN'T CIC MODE,
CCA8:FO 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 #$38
CCAF:FO 06 97 BEQ ASCREEN ;SLOT 0=APPLE SCREEN
CCB1: 98 *
CCB1:8A 99 TXA ;JUMP TO CNOO SPACE
CCB2:48 100 PHA
CCB3:AF 101 LDA #$SENDCD-1 ; TO VECTOR TO THE PERIPHERAL
CCB5:48 102 PHA ; IN THE CHAIN SLOT
CCB6:60 103 NOOUT RTS
CCB7: 104 *
CCB7: 105 * APPLE 40-COL SCREEN DRIVER
CCB7: 106 *
CCB7:20 DF CC 107 ASCREEN JSR GETXLAITE ;GET THE TRANSLATE OPTIONS
CCBA:09 80 108 ORA #$80 ;SET HIGH BIT OF CHAR
CCBC:C9 E0 109 CMP #$EO ;LOWERCASE?
CCBE:90 06 110 BCC TESTLETTER
CCCD:59 D3 CC 111 EOR LCINDEX,Y ;DO LOWERCASE TRIP
CCCE:4C F6 PD 112 TOSCREEN JMP VIDOUT ;ALL REGS ARE PRESERVED
CCCF: 113 *
CCCF: 114 * IF UPPERCASE, WE ONLY MAP LETTERS
(listings continued on next page)
**CCEB: 138**

**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

19 CMDTBL DFB $42 ; BREAK

**CCEC:** 67

20 DFB $67 ; CIC PAS NS=7

**CCEED:**

21 DFB >BREAKCMD-1

**CCEEF:** 54

22 DFB $54 ; TERMINAL

**CCEEP:** 47

23 DFB $47 ; CIC NS=7

**CCEF:**

24 DFB >TERMCMDO-1

**CCF:**

25 DFB $43 ; C(R GENERATE)

**CCF:**

26 DFB $87 ; PPC NS=7

**CCFG:**

27 DFB >TERMCMDO-1

**CCFK:**

28 DFB $51 ; QUIT

**CCFL:**

29 DFB $47 ; CIC NS=7

**CCFM:**

30 DFB >QUITCMD-1

**CCFN:**

31 DFB $52 ; R(LEASE)

**CCFP:**

32 DFB $C7 ; CIC PPC NS=7

**CCFR:**

33 DFB >RESETCMD-1

**CCFS:**

34 DFB $5A ; Z COMMAND

**CCFT:**

35 DFB $E7 ; CIC PPC PAS NS=7

**CCGH:**

36 DFB >ZCMD-1

**CCGI:**

37 DFB $49 ; I COMMAND

**CCGJ:**

38 DFB $90 ; PPC NS=0

**CCGK:**

39 DFB >ICMD-1

**CCGL:**

40 DFB $4B ; K COMMAND

**CCGM:**

41 DFB $90 ; PPC NS=0

**CCGN:**

42 DFB >KCMD-1

**CCGO:**

43 *

**CCGP:**

44 DFB $45 ; E(CH0)

**CCGQ:**

45 DFB $43 ; CIC NS=3

**CCGR:**

46 DFB $80

**CCGS:**

47 DFB $46 ; F(ROMKEYBD)

**CCGT:**

48 DFB $E3 ; CIC PPC PAS NS=3

**CCGU:**

49 DFB $04

**CCGV:**

50 DFB $4C ; L(F GENERATE)

**CCGW:**

51 DFB $E3 ; CIC PPC PAS NS=3

**CCGX:**

52 DFB $01

**CCGY:**

53 DFB $58 ; X(OPF)

**CCGZ:**

54 DFB $E3 ; CIC PPC PAS NS=3

**CCH:**

55 DFB $08

**CCHI:**

56 DFB $54 ; T(ABBING)

**CCHJ:**

57 DFB $83 ; PPC NS=3

82 SUPER SERIAL CARD
CD11:40  58  DFB $40  ;S(HIFTING)
CD12:53  59  DFB $53  ;CIC
CD13:43  60  DFB $43  ;M(UNCH LF)
CD14:40  61  DFB $40  ;CIC PPC PAS NS=3
CD15:4D  62  DFB $4D  ;CIC PPC PAS NS=3
CD16:E3  63  DFB $E3  ;CIC PPC PAS NS=3
CD17:20  64  DFB $20  ;END OF FIRST PART MARKER
CD18:  65  
CD18:00  66  DFB $0C  ;END OF TABLE MARKER
CD19:  67  
CD19:42  68  CMDTBL1  DFB $42  ;B(AUD)
CD1A:F6  69  DFB $F6  ;CIC PPC PAS NS=6
CD1B:7C  70  DFB >BAUDCMD-1
CD1C:50  71  DFB $50  ;P(ARITY)
CD1D:F6  72  DFB $F6  ;CIC PPC PAS NS=6
CD1E:9A  73  DFB >PARITYCMD-1
CD1F:44  74  DFB $44  ;D(ATA)
CD20:F6  75  DFB $F6  ;CIC PPC PAS NS=6
CD21:9B  76  DFB >DATACMD-1
CD22:46  77  DFB $46  ;F(P DELAY)
CD23:F6  78  DFB $F6  ;CIC PPC PAS NS=6
CD24:46  79  DFB >FCMD-1
CD25:4C  80  DFB $4C  ;L(F DELAY)
CD26:F6  81  DFB $F6  ;CIC PPC PAS NS=6
CD27:40  82  DFB >LPCCMD-1
CD28:43  83  DFB $43  ;C(R DELAY)
CD29:F6  84  DFB $F6  ;CIC PPC PAS NS=6
CD2A:3A  85  DFB >CRCMD-1
CD2B:54  86  DFB $54  ;T(TRANSLATE)
CD2C:D6  87  DFB $D6  ;CIC PPC NS=6
CD2D:34  88  DFB >TRANCMD-1
CD2E:4E  89  DFB $4E  ;N COMMAND
CD2F:90  90  DFB $90  ;PPC NS=0
CD30:E9  91  DFB >NCMD-1
CD31:53  92  DFB $53  ;S(SCREENSHOT)
CD32:56  93  DFB $56  ;CIC NS=6
CD33:60  94  DFB >SSLOTCMD-1
CD34:  95  
CD34:00  96  DFB $00  ;END OF TABLE MARKER
CD35:  97  
CD35:  98  *********************************************
CD35:  99  * COMMAND ROUTINES  *
CD35: 100  * (CALLED BY PARSER)  *
CD35: 101  * (MUST START IN)  *
CD35: 102  * PAGE $CD . . . )  *
CD35: 103  *********************************************
CD35:A9 3F  104 TRANCMD LDA #$3F  ;SET SCREEN TRANSLATE OPTIONS
CD37:A0 07  105 LDY #$7
CD39:D0 10  106 BNE DELAYSET ;<ALWAYS>
CD3B:A9 CF  107 CRCMD LDA #$CF  ;SET CR DELAY
CD3D:0A 05  108 LDY #$5
CD3F:D0 0A  109 BNE DELAYSET ;<ALWAYS>
CD41:  110  
CD41:A9 F3  111 LPCCMD LDA #$F3  ;SET LF DELAY
CD43:A0 03  112 LDY #$3
CD45:D0 04  113 BNE DELAYSET ;<ALWAYS>
CD47:  114  
CD47:A9 FC  115 FCMD LDA #$FC  ;SET FF DELAY
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 STA ZTMP1
CD50: BD 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 STA 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 C300 ENTRY)
CD7C: 60 145 SSLOTCMD1 RTS
CD7D: 146 *
CD7D: 29 0F 147 BAUDCMD AND #$0F ;SET NEW BAUD RATE
CD7F: DD 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 #$80
CD8E: 85 2B 157 CTLREGSET STA ZTMP2
CD90: 89 8B C0 158 LDA CTLREG,Y
CD93: 25 2B 159 AND ZTMP2
CD95: 05 2A 160 ORA ZTMP1
CD97: 99 8B 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
CDA0: 0A 171 ASL A
CDA1: 85 2A 172 DATACMD1 STA ZTMP1
CDA3: A9 1F 173 LDA #$1F

84 SUPER SERIAL CARD
CDA5:DO E7  174  BNE  CTRLREGSET  ;<ALWAYS>
CDA7:  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:  179 *
CDA9:99 89 C0  180  RESETCMD  STA  RESSET,Y  ;DROP RTS, DTR
CDA9:20 93 FE  181  JSR  SETSCR  ;PR#0
CDA9:20 89 FE  182  JSR  SETKB  ;IN#0
CDA9:AE F8 07  183  LDX  MELOT
CDA9:1E B8 04  184  QUITCMD  ASL  STATEFLG,X  ;CLEAR TERMINAL MODE
CDA9:C1  185  CLC
CDA9:7E B8 04  186  QCMD1  ROR  STATEFLG,X
CDA9:00 60  187  RTS
CDA9:1C  188 *
CDA9:89 8A C0  189  BREAKCMD  LDA  CMDREG,Y  ;SEND BREAK SIGNAL
CDA9:48  190  PHA  ;FOR 233 MILISECONDS
CDA9:09 0C  191  ORA  #$0C
CDA9:79 8A C0  192  STA  CMDREG,Y
CDA9:A9 E9  193  LDA  #$233  ;DELAY  FOR 233 MICROSEC.
CDA9:20 C4 CA  194  JSR  WATMS
CDA9:68  195  PLA  ;RESTORE  OLD  COMMAND  REG  CONTENTS
CDA9:00 8A C0  196  STA  CMDREG,Y
CDA9:3D 60  197  RTS
CDA9:4D  198 *
CDA9:A9 28  199  ICM  LDA  #$28
CDA9:6D 38 06  200  STA  PWDBYTE,X  ;SET  PRINTER  WIDTH  TO  40
CDA9:6D 9A 80  201  LDA  #$80
CDA9:8D 1D 38 07  202  ORA  MISCPFLG,X  ;SET  SCREEN  ECHO
CDA9:DE 0D  203  BNE  KCMD2  ;<ALWAYS>
CDA9:60  204 *
CDA9:A9 FE  205  KCMD  LDA  #$FE  ;RESET  THE  LF  GENERATE  FLAG
CDA9:23 D8 07  206  KCMD1  AND  MISCPFLG,X
CDA9:5D 38 07  207  KCMD2  STA  MISCPFLG,X
CDA9:88 60  208  RTS
CDA9:89  209 *
CDA9:C9 28  210  NCM  CMP  #$40  ;>=407
CDA9:EB 90 08  211  BCC  ZCMDRTS  ;IF  NOT,  JUST  EXIT
CDA9:ED 38 06  212  STA  PWDBYTE,X  ;SET  NEW  PRINTER  WIDTH
CDA9:F0 A9 3F  213  LDA  #$3F  ;DISABLE  SCREEN,  SET  LISTING  MODE
CDA9:F2 0D EE  214  BNE  KCMD1  ;<ALWAYS>
CDA9:F4  215 *
CDA9:F4 1E 38 05  216  ZCMD  ASL  PWDBYTE,X  ;DISABLE  COMMAND  RECOGNITION
CDA9:F7 38  217  SEC
CDA9:F8 7E 38 05  218  ROR  PWDBYTE,X
CDA9:F8 60  219  ZCMDRTS  RTS
CDA9:F8  220 *
CDA9:F8  221 *****************************************************
CDA9:F8  222 *VECTOR  ACCORDING  TO  COMMAND  STATE*
CDA9:F8  223 *****************************************************
CDA9:F8:AB  224  CMDPROC  TAY  ;A-REG=COMMAND  STATE
CDA9:F8:A5 27  225  LDA  CHARACTER
CDA9:F8:29 7F  226  AND  #$7F
CDA9:F8:01  227 *
CDA9:F8:C9 20  228  CMP  #$20  ;SKIP  SPACES  FOR  ALL  MODES
CDA9:F8:3D 09  229  BNE  CMDPROC2
CDA9:F8:05 C0 03  230  CPY  #$3  ;EXCEPT  MODE  3
CDA9:F8:07 0F 01  231  BEQ  CMDPROC1
CE09:60 232 RTS
CE0A:09 04 233 CMDPROC1 LDA #$34
CE0C:0D 6D 234 BNE SETOSTATE ;<ALWAYS>
CE0E: 235 *
CE0E:C9 0D 236 CMDPROC2 CMP #$0D ;CARRIAGE RETURN?
CE10:1D 12 237 BNE CMDPROC4 ;
CE12:20 79 CE 238 JSR ZEROSTATE ;ABORT FOR STATES 0-5, EXIT FOR 6,7
CE15:0C 07 239 CPY #$07 ;IN STATE 7 WE VECTOR TO THE PROC
CE17:F0 01 240 BEQ CMDPROC3 ;
CE19:60 241 RTS ;OTHERWISE, JUST EXIT
CE1A: 242 *
CE1A:A9 CD 243 CMDPROC3 LDA #$CD ;ALL PROC'S MUST START IN PAGE #$CD
CE1C:48 244 PHA
CE1D:BD 38 04 245 LDA PARAMETER,X
CE20:48 246 PHA
CE21:A4 26 247 LDY SLOT16 ;NEEDED BY BREAK CMD
CE23:60 248 RTS
CE24: 249 *
CE24:85 35 250 CMDPROC4 STA ZPTEMP
CE26:A9 CE 251 LDA #$CE ;ALL ROUTINES MUST START
CE28:48 252 PHA ; IN PAGE #$CE
CE29:B9 30 CE 253 LDA STATEBL,Y
CE2C:48 254 PHA
CE2D:A5 35 255 LDA ZPTEMP
CE2F:60 256 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 263 STATEBL DF# >STATEERR-1 ;BAD STATE
CE31:37 264 DB# >CSTATE1-1 ;<CMD> SEEN
CE32:61 265 DF# >CSTATE2-1 ;ACCUMULATE PARAMETER
CE33:89 266 DF# >CDONE-1 ;SKIP UNTIL SPACE
CE34:8A 267 DF# >CSTATE4-1 ;E/D SOMETHING
CE35:A7 268 DF# >STATEERR-1 ;ILLEGAL STATE
CE36:89 269 DF# >CDONE-1 ;SKIP UNTIL CR
CE37:89 270 DF# >CDONE-1 ;SKIP UNTIL CR THEN DO CMD
CE38: 271 ***********************
CE38: 272 * COMMAND STATE 1 *
CE38: 273 ***********************
CE38:DD 38 05 274 CSTATE1 CMP CMDBYTE,X ;IS IT <CMD>?
CE3B:06 275 BNE CSTATE1A
CE3D:DE B8 04 276 DEC STATEFLG,X ;SET STATE BACK TO ZERO
CE40:4C 02 CC 277 JMP ACIAOUT ;OUTPUT <CMD> IF SO
CE43: 278 *
CE43:C9 30 279 CSTATE1A CMP #$30 ;>=0?
CE45:90 0D 280 BCC CSTATE1B
CE47:C9 3A 281 CMP #$3A ;<=9?
CE49:80 09 282 BCS CSTATE1B
CE4B:29 0F 283 AND #$0F ;IT'S A NUMBER
CE4D:9D 38 04 284 STA PARAMETER,X
CE50:A9 02 285 LDA #$2
CE52:DD 27 286 BNE SETOSTATE ;<ALWAYS> SET MODE 2 AND RETURN
CE54: 287 *
CE54:C9 20 288 CSTATE1B CMP #$20 ;IS IT A CONTROL CHAR?
CE56:80 06 289 BCS CSTATE1C
CE58:9D 38 05 290 STA CMDBYTE,X ;SET NEW COMMAND CHARACTER
CE58:9C 79 CE 291 JMP ZEROSTATE ;RESET STATE TO ZERO
CE58: 292 *
CE5E:A0 00 293 CSTATE1C LDY #0 ;USE COMMAND TABLE
CE5E:F0 4D 294 BSR CMDSEARCH ;<ALWAYS>
CE5E: 295 ********************************************
CE5E: 296 * COMMAND STATE 2: ACCUMULATE PARAMETER *
CE5E: 297 ********************************************
CE5E:24 49 30 298 CSTATE2 EOR #$30 ;CONVERT $30-$39 TO 0-9
CE5E:C9 0A 299 CMP #$A ;0-9?
CE5E:B0 0D 300 BCS CSTATE2A
CE5E:A0 0A 301 LDY #$A ;IT'S A NUMBER, SO ADD
CE5E:7D 38 04 302 ACLOOP ADC PARAMETER,X ; IT TO 10*PARAMETER
CE5E:B8 303 DEY
CE5E:D0 FA 304 BNE ACLOOP
CE5E:9D 38 04 305 STA PARAMETER,X
CE5E:F0 15 306 BEQ CDONE ;<ALWAYS>
CE5E: 307 *
CE5E:A0 2E 308 CSTATE2A LDY #CMDYPE1-CMDYPE2 ;USE COMMAND TABLE
CE5E:D0 36 309 BNE CMDSEARCH ;<ALWAYS>
CE5E: 310 ********************************************
CE5E: 311 * SET COMMAND STATE *
CE5E: 312 ********************************************
CE5E:A9 00 313 ZEROSTATE LDA #0
CE5E:B5 85 2A 314 SETOSTATE STA ZPTMP1
CE5E:DA F8 07 315 LDX MSLOT
CE5E:BD B8 04 316 LDA STATEFLG,X
CE5E:39 F8 317 AND #$F8
CE5E:55 05 2A 318 ORA ZPTMP1
CE5E:79 9D B8 04 319 STA STATEFLG,X
CE5E:60 320 CDONE RTS
CE5E: 321 ********************************************
CE5E: 322 * COMMAND STATE 4 (E/D) *
CE5E: 323 ********************************************
CE5E:85 A8 324 CSTATE4 TAY ;E/D -> Y-REG
CE5E:CD BD 38 04 325 LDA PARAMETER,X
CE5E:FP C0 44 326 CPY #$44 ;D(ISABLE)??
CE5E:F1 F0 09 327 BSR CSTATE4A
CE5E:3C 45 328 CPY #$45 ;E(NABLE)??
CE5E:5D 11 329 BNE STATERR ;IF NOT, IGNORE THIS COMMAND
CE5E:7D 38 07 330 ORA MISCFLG,X ;SET FLAG
CE5E:DA D0 05 331 BNE CSTATE4B ;<ALWAYS>
CE5E:C9 4F FF 332 CSTATE4A EOR #$FF ;INVERT FOR DISABLE
CE5E:3D 38 07 333 AND MISCFLG,X ;RESET FLAG
CE5E:D0 48 06 334 CSTATE4B STA MISCFLG,X
CE5E:A9 36 335 ********************************************
CE5E: 336 * ESCAPE TO STATE 6 *
CE5E: 337 ********************************************
CE5E:A9 06 338 SETSTATE5 LDA #6
CE5E:D0 D3 339 BNE SETSTATE5 ;<ALWAYS>
CE5E:A8 2A 340 STATERR LDA #$32 ;CODE FOR BAD COMMAND
CE5E:AD B8 05 341 STA STSBYTE,X
CE5E:D0 F5 342 BNE SETSTATE6 ;<ALWAYS>
CE5E: 343 ********************************************
CE5E: 344 * TABLE DRIVEN COMMAND PROCESSOR *
CE5E: 345 ********************************************
CE5E:B9 88 0C 346 CMDSEARCH LDA CMDTRL,Y ;GET CANDIDATE CHARACTER
CE5E:F0 F4 347 BSR STATERR ;A ZERO MARKS THE END OF A SUBTABLE

FIRMWARE 87
CEB4:C5 35 348 CMP ZPTMP1 ;MATCH?
CEB6:F0 05 349 BGE CMDMATCH
CEB8:C8 350 INY
CEB9:C8 351 CMDSEARCH1 INY ;REENTRY FOR WRONG MODES
CEBAA:C8 352 INY ;ENTRY LENGTH = 3
CEBBD:0D F2 353 BNE CMDSEARCH ;<ALWAYS>
CEBBD:CB 354 *
CEBBD:CB 355 CMDMATCH INY
CEBEE: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 DONT CHECK P-BIT
CEC7:BD 38 07 360 LDA MISCPFLG,X ;OFF SO MAKE SURE
CECA:29 10 361 AND #$10 ; THAT WE AREN'T IN PASCAL
CECC:DD EB 362 BNE CMDSEARCH1 ;BRANCH IF WE ARE
CECE: 363 *
CECE:BD 38 07 364 CMDMATCH1 LDA MISCPFLG,X ;GET CIC/PPC BIT
CED1:4A 365 LSR A ;SHIFT CIC/PPC MODE BIT TO CARRY
CED2:4A 366 LSR A "
CED3:24 2A 367 BIT ZPTMP1 ;PPC->N CIC->V
CED5:BO 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
CEDM: 372 *
CEDE:A5 2A 373 CMDEXEC LDA ZPTMP1 ;RETRIEVE TABLE MODE BYTE
CEEF:48 374 PHA
CEF0:29 07 375 AND #$07
CEF2:20 7B CE 376 JSR SETOSTATE ;SET NEXT STATE
CEF5:C8 377 INY
CEF6:68 378 PLA
CEF7:29 10 379 AND #$10 ;
CEF9:DD 07 380 BNE CMDEXEC1 ;IF BIT 4 IS SET, VECTOR TO ROUTINE
CEFB:B9 EB CC 381 LDA CMDTBL,Y
CEFB:9D 38 04 382 STA PARAMETER,X
CEF1:60 383 RTS
CEF2: 384 *
CEF2:A9 CD 385 CMDEXEC1 LDA #$SCD ;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 SLOT#6
CEFB:BD 38 04 390 LDA PARAMETER,X ;LOT OF ROUTINES NEED THIS
CEFB:60 391 RTS
CEFF: 392 *
CEFF:00 393 DPB $00

SYMBOL TABLE SORTED BY SYMBOL

3C A1L C68A ACCLOOP CC02 ACIAOUT ?CB9C ACK
C9CB A1DS CS95 ADJUST C859 ANRTS C9C7 ASCREEN
C888 ASCESS C995 ADJUST C959 ANRTS C997 ASCREEN
?C941 BACHOUT C7D7 BAUCMD CDB1 BAUCMD1 C9D8 BAUCMD2
C711 BEND C85F BIAUCMD C770 BINT C8CB BINCDB
C8BD BINFU C747 BINT1 C767 BOUTPUT C78B BOUTPUT2
C7DC BINFU C77C BOUTPUT1 C767 BOUTPUT C78B BOUTPUT2
27 CHARACTER C4E CHECKTERM 0638 CHNBYTE C885 CICEXIT
C9BE CICEND C9D1 CKINPUT C9B5 CKINPUT1 C9BE CKINPUT2

88 SUPER SERIAL CARD
CC3D CKBXDXIT  CC2C CKKBD  CC34 CKKB1D 0538 CMDBYTE
CEF2 CMDEXEC1  CEDD CMDEXEC  CEF6 CMDEXACT1 CEB0 CMDEXACT2
CED8 CMDMATCH2  CEA0 CMDPROC1  CEB8 CMDPROC2
CED4 CMDPROC4  CEDC CMDPROC  CEB0 CMDREG  CEB4 CMDEXCH
CE99 CMDSEARCH  CB1A CMDSEQCK  CD19 CMDBL1  CCB8 CMDBL
0688 COLBYTE  C917 CMMA  FEDD COUT  CD38 CRCMD
CE43 CSTATE1A  CB54 CSTATE1B  CE58 CSTATE1C  CE38 CSTATE1E
CE75 CSTATE2A  CB62 CSTATE2  CE9C CSTATE4A  CE41 CSTATE4B
CE98 CSTATE4  37 CSWL  36 CSWL  CE8B CTRLREG
CDB8 CTLGETSET  CB9C CTRLTST  CDA1 DATACMD1  CDB9 DATACMD
C9C0 DECRCOL  0388 DELAYFLG  CD48 DELAYSET  CD91 DIPSW1
C9B2 DIPSW2  CBFF DLITBL  CBE2 ESRCHECK  CB90 ETH
CD47 FCMD  CB68 FORCES  C754 FROMIN  C751 PROMOUT
CE84 GETCHAR1  COA4 GETCHAR  CC72 GETCMD  CC38 GETDIO
CC44 GETKB1D  CC91 GETKBDONE  CDMP GETXLATE  0438 HANDSHKE
CD44 ICMD  C705 ENTRY  0200 INBUFF  C305 INIT1
C827 INIT1A  CB35 INIT2  C83C INIT2A  C83F INIT2B
C857 INIT3  CB64 INIT4  C872 INIT5  C879 INIT6A
C892 INITIACIA1  CB8F INITIACIA2  CB16 INPUT2  CAPP INPUT
PF58 IORCS  CO10 KBDSTR  C000 KBD  C55C KBRDESC
C828 KCMD1  C85 KCMD2  C80 KCMD  39 KSWH
38 KSWL  C83 KLCMD  C84 LFCMD  CBB8 LGGEN
0733 MISCFLG  CAOC MOVIN  C9F9 MOVOUT  07F8 MSLOT
CE99 NCMOD  CB58 NOCMD  CB19 NOINPUT1  C818 NOINPUT
C369 NOOUT  C7C5 NORM10  C954 NOTAB1  C951 NOTAB
PCBA NZTA1  C707 ONENTRY  CB99 OUTDL1  C862 OUTDLYLP
C6B8 OUTPUT1  CB68 OUTPUT2  CB63 OUTPUT  C8C1 OUTPUT3
CBB8 OUTPUTF  CBB8 PBAOUT1  CBAG PBAOUT2  C8A9 PBAOUT3
CBB8 PBAOUT4  0438 PARAMETER  CD98 PARITYCMD  C800 PASCALINIT
?C898 PASCALREAD1  C898 PASCALREAD  C9AA PASCALWR1  C8A3 PASEXIT
C998 PENTRY  C78E PINIT  ?C84D PREAD  C794 PREAD
CC93 PROMPTBL  CC5E PROMPTLOOP  C7A8 PSTATIN  C79A PSTATUS
C7A8 PSTAT1  0638 PWORDTY  C9A6 PWDL1  C797 PWRITE
CD80 QCMD1  CDR9 QUITCMD  C088 RDRBG  C899 RESSET
C800 RESSTCMD  CC11 RESTORE  CC29 RESTOREND  C7EE RESTROOK
CD80 RESSTCMD  C864 RIM  4F RNDL  C884 RNDL  C889 RNDL
CD57 ROTATE  CB82 SAVEHOOK  C898 SCREENOUT  C8A8 SCREENOUT1
C800 SENSDO  C998 SEREND2  C979 SEREND  C8FC SERRUT
C996 SETCH  FEB9 SETKB  CB78 SETSTATE  FE93 SETSCR
CE44 SETSTATE8  26 SLOT16  CA99 SRIN12  C8ER SRIN2
CAD2 SHN  CAP4 SRIN3  CAP5 SHOUT  CD7C SLOTCMD1
CD61 SLOTCMD  0100 STACK  0488 STATEFLG  CB08 STATEFL
CE30 STATEBL  C089 STRBG  0588 STBYTE  C934 TAB1
C984 TAB2  C921 TABCHECK  ?C988 TOREG  C955 TERRMACIA
CA9D TCMP1  CA9B TCMP2  CA8D TCMP3  ?CA4C TERRHICT
CA82 TERMIN1  CA81 TERMIN  CA86 TERMKBD1  CA87 TERRLETTER
CA81 TERMIN2  CA82 TERMIN3  CA98 TERMKBD2  CA87 TERRLETTER
CA93 TERMSEND  CA95 TERMSEND1  CCA6 TERMMAC  CA50 TERMTS
CD35 TRNMOD  CA95 TRNSEQ  CCD7 UCMD  C850 TOSCREEN
C8C4 WAITMS  CA86 WAITMS1  CB7C VDOUOUT  CB5A VONNIGHT
CDFB ZCMDRTS  CDF4 ZCMD  CB79 ZEROSTATE  35 ZPTEMP

SYMBOL TABLE SORTED BY ADDRESS

24 CH  26 SLOT16  27 CHARACTER  28 BASL
2A ZPMP1  2B ZPMP2  35 ZPTEMP  36 CSWL
37 CSWL  38 KSWL  39 KSWH  3C AIL
48 RNUL  4F RNUL  0100 STACK  0200 INBUFF
0388 DELAYFLG  0438 HANDSHKE  0438 PARAMETER  0488 STATEFLG

FIRMWARE 89
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 4Ø 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  \(49291 + 3\times16; \ 3 + 24\Phi\)
POKE 49338, 1Ø7  \(4929\Phi + 3\times16; \ p = 107\)
POKE 2Ø43, 132  (plug in magic number)
POKE 1147, 64   (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 ON OFF OFF OFF
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</th>
<th>P8 Serial Card</th>
<th>Super Serial Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>P8 Mode:</td>
<td>SWL-5 ON,</td>
<td>POKE 1144+s,r</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SWL-6 OFF</td>
<td>r = b + d; b =</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(not available)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 dec/$@1 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 dec/$@2 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 dec/$@3 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 dec/$@4 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 dec/$@5 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 dec/$@6 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 dec/$@7 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 dec/$@8 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 dec/$@9 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 dec/$@A hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 dec/$@B hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>12 dec/$@C hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>13 dec/$@D hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14 dec/$@E hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15 dec/$@F hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8A Mode:</td>
<td>SWL-5 OFF,</td>
<td>POKE 49291+s*16,r</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SWL-6 OFF</td>
<td>r = b + d; b =</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(not available)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baud Rate:</td>
<td>SWL-1 to SWL-4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>same as Printer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mode</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>1 dec/$@1 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td></td>
<td>2 dec/$@2 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td></td>
<td>3 dec/$@3 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>135</td>
<td></td>
<td>4 dec/$@4 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>150</td>
<td></td>
<td>5 dec/$@5 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>300</td>
<td></td>
<td>6 dec/$@6 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>600</td>
<td></td>
<td>7 dec/$@7 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1200</td>
<td></td>
<td>8 dec/$@8 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1800</td>
<td></td>
<td>9 dec/$@9 hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2400</td>
<td></td>
<td>10 dec/$@A hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3600</td>
<td></td>
<td>11 dec/$@B hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4800</td>
<td></td>
<td>12 dec/$@C hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7200</td>
<td></td>
<td>13 dec/$@D hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9600</td>
<td></td>
<td>14 dec/$@E hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19200</td>
<td></td>
<td>15 dec/$@F hex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Format:</td>
<td>SW2-1 ON</td>
<td>POKE 1912+s,r</td>
<td>(to get r above,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>POKE 1272+s,t</td>
<td>add d to b) d =</td>
<td></td>
</tr>
<tr>
<td>8 data,1 stop</td>
<td></td>
<td>r = 9; t = 1*</td>
<td>16 dec/$@9 hex</td>
<td></td>
</tr>
<tr>
<td>7 data,1 stop</td>
<td></td>
<td>r = 8; t = 1*</td>
<td>48 dec/$@30 hex</td>
<td></td>
</tr>
<tr>
<td>6 data,1 stop</td>
<td></td>
<td>r = 7; t = 1*</td>
<td>80 dec/$@50 hex</td>
<td></td>
</tr>
<tr>
<td>5 data,1 stop</td>
<td></td>
<td>r = 6; t = 1*</td>
<td>112 dec/$@70 hex</td>
<td></td>
</tr>
<tr>
<td>8 data,2 stop</td>
<td></td>
<td>r = 9; t = 2*</td>
<td>144 dec/$@90 hex</td>
<td></td>
</tr>
<tr>
<td>7 data,2 stop</td>
<td></td>
<td>r = 8; t = 2*</td>
<td>176 dec/$@B0 hex</td>
<td></td>
</tr>
<tr>
<td>6 data,2 stop</td>
<td></td>
<td>r = 7; t = 2*</td>
<td>208 dec/$@D0 hex</td>
<td></td>
</tr>
<tr>
<td>5 data,2 stop</td>
<td></td>
<td>r = 6; t = 2*</td>
<td>240 dec/$@F0 hex</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>* add l if</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p = 1 or 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parity:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>odd</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>even</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MARK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPACE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table B-1. SIC Switch Settings, PEEKs and POKEs, Part I
<table>
<thead>
<tr>
<th>Selection</th>
<th>SSC switches and settings</th>
<th>PEEKs and POKEs to use for</th>
<th>P8 Serial Card</th>
<th>Super Serial Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line Width:</td>
<td>SW2-3 &amp; SW2-4,</td>
<td>POKE 1784+s,r</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>same as Printer Mode</td>
<td>r=1 to 255; for no &lt;CR&gt;, r=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Video/ Generate &lt;LF&gt;/</td>
<td>SW2-3 &amp; SW2-4</td>
<td>V = Video on?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Generate &lt;LF&gt;/ Translate/</td>
<td>SW2-5</td>
<td>G = Gen. &lt;LF&gt;?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;CR&gt; Delay:</td>
<td>(no switch)</td>
<td>T = LC to UC?</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SW2-2</td>
<td>D = Dly 1/4 s?</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(all switches</td>
<td>POKE 2040+s,r</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>same as in Printer Mode</td>
<td>r=</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dec hex V G T D</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 $84 Y N Y Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 $85 Y Y Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>36 $24 Y N N Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>37 $25 Y N Y Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>68 $44 Y N Y N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>69 $45 Y Y Y N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>100 $64 Y N N N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>101 $65 Y Y N N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>132 $84 N N Y Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>133 $85 N Y Y Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>164 $A4 N N N Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>165 $A5 N Y N Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>196 $C4 N N Y N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>197 $C5 N Y Y N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>228 $E4 N N N N</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>229 $E5 N Y N N</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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 1490+5 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 1490+5. For example, to change the enquire character to ENQ (ASCII 5), which is used by many RS-232 devices, use this POKE: POKE 1490+5, 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 $C900 (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>.

96  SUPER SERIAL CARD
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 1Ø-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
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>Language</th>
<th>Version</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer BASIC</td>
<td>DOS 3.2</td>
<td>Pascal 1.0</td>
</tr>
<tr>
<td>Applesoft BASIC</td>
<td>DOS 3.3</td>
<td>Pascal 1.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6502 Assembler</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>Header</th>
<th>Connector</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Frame Ground</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Transmit Data (TXD)</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>Receive Data (RXD)</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>Request To Send (RTS)</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>Clear To Send (CTS)</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>Data Set Ready (DSR)</td>
</tr>
<tr>
<td>7</td>
<td>19</td>
<td>Secondary Clear To Send (SCTS)</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>Signal Ground</td>
</tr>
<tr>
<td>9</td>
<td>20</td>
<td>Data Terminal Ready (DTR)</td>
</tr>
<tr>
<td>10</td>
<td>8</td>
<td>Data Carrier Detect (DCD)</td>
</tr>
</tbody>
</table>

Table C-1. Connector Pin Assignments
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>
</tr>
</thead>
<tbody>
<tr>
<td>BINARY</td>
<td>DEC</td>
<td>HEX</td>
<td>DEC</td>
<td>HEX</td>
<td>CHAR</td>
<td>INTERPRETATION</td>
<td>NORMAL</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-Ø&gt;</td>
</tr>
<tr>
<td>00000001</td>
<td>1</td>
<td>01</td>
<td>129</td>
<td>81</td>
<td>SOH</td>
<td>Start of Header</td>
<td>&lt;CTRL-A&gt;</td>
</tr>
<tr>
<td>00000100</td>
<td>2</td>
<td>02</td>
<td>130</td>
<td>82</td>
<td>STX</td>
<td>Start of Text</td>
<td>&lt;CTRL-B&gt;</td>
</tr>
<tr>
<td>00000110</td>
<td>3</td>
<td>03</td>
<td>131</td>
<td>83</td>
<td>ETX</td>
<td>End of Text</td>
<td>&lt;CTRL-C&gt;</td>
</tr>
<tr>
<td>00001000</td>
<td>4</td>
<td>04</td>
<td>132</td>
<td>84</td>
<td>EOT</td>
<td>End of Transm.</td>
<td>&lt;CTRL-D&gt;</td>
</tr>
<tr>
<td>00001010</td>
<td>5</td>
<td>05</td>
<td>133</td>
<td>85</td>
<td>ENQ</td>
<td>Enquiry</td>
<td>&lt;CTRL-E&gt;</td>
</tr>
<tr>
<td>00001100</td>
<td>6</td>
<td>06</td>
<td>134</td>
<td>86</td>
<td>ACK</td>
<td>Acknowledge</td>
<td>&lt;CTRL-F&gt;</td>
</tr>
<tr>
<td>00001110</td>
<td>7</td>
<td>07</td>
<td>135</td>
<td>87</td>
<td>BEL</td>
<td>Bell</td>
<td>&lt;CTRL-G&gt;</td>
</tr>
<tr>
<td>00010000</td>
<td>8</td>
<td>08</td>
<td>136</td>
<td>88</td>
<td>BS</td>
<td>Backspace</td>
<td>&lt;CTRL-H&gt;</td>
</tr>
<tr>
<td>00010100</td>
<td>9</td>
<td>09</td>
<td>137</td>
<td>89</td>
<td>HT</td>
<td>Horizontal Tab</td>
<td>&lt;CTRL-I&gt;</td>
</tr>
<tr>
<td>00010110</td>
<td>10</td>
<td>0A</td>
<td>138</td>
<td>8A</td>
<td>LF</td>
<td>Linefeed</td>
<td>&lt;CTRL-J&gt;</td>
</tr>
<tr>
<td>00011010</td>
<td>11</td>
<td>0B</td>
<td>139</td>
<td>8B</td>
<td>VT</td>
<td>Vertical Tab</td>
<td>&lt;CTRL-K&gt;</td>
</tr>
<tr>
<td>00011010</td>
<td>12</td>
<td>0C</td>
<td>140</td>
<td>8C</td>
<td>FF</td>
<td>Form Feed</td>
<td>&lt;CTRL-L&gt;</td>
</tr>
<tr>
<td>00011110</td>
<td>13</td>
<td>0D</td>
<td>141</td>
<td>8D</td>
<td>CR</td>
<td>Carriage Return</td>
<td>&lt;CTRL-M&gt;</td>
</tr>
<tr>
<td>00011110</td>
<td>14</td>
<td>0E</td>
<td>142</td>
<td>8E</td>
<td>SO</td>
<td>Shift Out</td>
<td>&lt;CTRL-N&gt;</td>
</tr>
<tr>
<td>00011110</td>
<td>15</td>
<td>0F</td>
<td>143</td>
<td>8F</td>
<td>SI</td>
<td>Shift In</td>
<td>&lt;CTRL-O&gt;</td>
</tr>
<tr>
<td>00100000</td>
<td>16</td>
<td>10</td>
<td>144</td>
<td>90</td>
<td>DLE</td>
<td>Data Link Escape</td>
<td>&lt;CTRL-P&gt;</td>
</tr>
<tr>
<td>00100100</td>
<td>17</td>
<td>11</td>
<td>145</td>
<td>91</td>
<td>DC1</td>
<td>Device Control 1</td>
<td>&lt;CTRL-Q&gt;</td>
</tr>
<tr>
<td>00100110</td>
<td>18</td>
<td>12</td>
<td>146</td>
<td>92</td>
<td>DC2</td>
<td>Device Control 2</td>
<td>&lt;CTRL-R&gt;</td>
</tr>
<tr>
<td>00100110</td>
<td>19</td>
<td>13</td>
<td>147</td>
<td>93</td>
<td>DC3</td>
<td>Device Control 3</td>
<td>&lt;CTRL-S&gt;</td>
</tr>
<tr>
<td>00101010</td>
<td>20</td>
<td>14</td>
<td>148</td>
<td>94</td>
<td>DC4</td>
<td>Device Control 4</td>
<td>&lt;CTRL-T&gt;</td>
</tr>
<tr>
<td>00101110</td>
<td>21</td>
<td>15</td>
<td>149</td>
<td>95</td>
<td>NAK</td>
<td>Neg. Acknowledge</td>
<td>&lt;CTRL-U&gt;</td>
</tr>
<tr>
<td>00110100</td>
<td>22</td>
<td>16</td>
<td>150</td>
<td>96</td>
<td>SYN</td>
<td>Synchronization</td>
<td>&lt;CTRL-V&gt;</td>
</tr>
<tr>
<td>00111110</td>
<td>23</td>
<td>17</td>
<td>151</td>
<td>97</td>
<td>ETB</td>
<td>End of Text Blk.</td>
<td>&lt;CTRL-W&gt;</td>
</tr>
<tr>
<td>01100000</td>
<td>24</td>
<td>18</td>
<td>152</td>
<td>98</td>
<td>CAN</td>
<td>Cancel</td>
<td>&lt;CTRL-X&gt;</td>
</tr>
<tr>
<td>01100100</td>
<td>25</td>
<td>19</td>
<td>153</td>
<td>99</td>
<td>EM</td>
<td>End of Medium</td>
<td>&lt;CTRL-Y&gt;</td>
</tr>
<tr>
<td>01101100</td>
<td>26</td>
<td>1A</td>
<td>154</td>
<td>9A</td>
<td>SUB</td>
<td>Substitute</td>
<td>&lt;CTRL-Z&gt;</td>
</tr>
<tr>
<td>01101110</td>
<td>27</td>
<td>1B</td>
<td>155</td>
<td>9B</td>
<td>ESC</td>
<td>Escape</td>
<td>&lt;ESC&gt;</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>Ø.
<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</td>
<td>DEC</td>
<td>HEX</td>
<td>DEC</td>
<td>HEX</td>
<td>CHAR</td>
<td>INTERPRETATION</td>
</tr>
<tr>
<td>00111000</td>
<td>28</td>
<td>1C</td>
<td>156</td>
<td>9C</td>
<td>FS</td>
<td>n/a</td>
</tr>
<tr>
<td>00111010</td>
<td>29</td>
<td>1D</td>
<td>157</td>
<td>9D</td>
<td>GS</td>
<td>(CTRL-SHIFT-M)</td>
</tr>
<tr>
<td>00111010</td>
<td>2E</td>
<td>1E</td>
<td>158</td>
<td>9E</td>
<td>RS</td>
<td>(CTRL-SHIFT-N)</td>
</tr>
<tr>
<td>00111111</td>
<td>31</td>
<td>1F</td>
<td>159</td>
<td>9F</td>
<td>US</td>
<td>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>01000001</td>
<td>33</td>
<td>21</td>
<td>161</td>
<td>A1</td>
<td></td>
<td>!</td>
</tr>
<tr>
<td>01000100</td>
<td>34</td>
<td>22</td>
<td>162</td>
<td>A2</td>
<td>&quot;</td>
<td>&quot;</td>
</tr>
<tr>
<td>01001100</td>
<td>35</td>
<td>23</td>
<td>163</td>
<td>A3</td>
<td>#</td>
<td>#</td>
</tr>
<tr>
<td>01001100</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>01010010</td>
<td>38</td>
<td>26</td>
<td>166</td>
<td>A6</td>
<td>&amp;</td>
<td>&amp;</td>
</tr>
<tr>
<td>01010111</td>
<td>39</td>
<td>27</td>
<td>167</td>
<td>A7</td>
<td>'</td>
<td>Closing Quote</td>
</tr>
<tr>
<td>01100000</td>
<td>40</td>
<td>28</td>
<td>168</td>
<td>A8</td>
<td>(</td>
<td>(</td>
</tr>
<tr>
<td>01100001</td>
<td>41</td>
<td>29</td>
<td>169</td>
<td>A9</td>
<td>)</td>
<td>)</td>
</tr>
<tr>
<td>01100110</td>
<td>42</td>
<td>2A</td>
<td>170</td>
<td>AA</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>01100111</td>
<td>43</td>
<td>2B</td>
<td>171</td>
<td>AB</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>01101000</td>
<td>44</td>
<td>2C</td>
<td>172</td>
<td>AC</td>
<td>`,</td>
<td>Comma</td>
</tr>
<tr>
<td>01101010</td>
<td>45</td>
<td>2D</td>
<td>173</td>
<td>AD</td>
<td>-</td>
<td>Hyphen</td>
</tr>
<tr>
<td>01101110</td>
<td>46</td>
<td>2E</td>
<td>174</td>
<td>AE</td>
<td>.</td>
<td>Period</td>
</tr>
<tr>
<td>01110000</td>
<td>47</td>
<td>2F</td>
<td>175</td>
<td>AF</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>01100000</td>
<td>48</td>
<td>30</td>
<td>176</td>
<td>B0</td>
<td>Ø</td>
<td>Ø</td>
</tr>
<tr>
<td>01100001</td>
<td>49</td>
<td>31</td>
<td>177</td>
<td>B1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>01100100</td>
<td>50</td>
<td>32</td>
<td>178</td>
<td>B2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>01100101</td>
<td>51</td>
<td>33</td>
<td>179</td>
<td>B3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>01101000</td>
<td>52</td>
<td>34</td>
<td>180</td>
<td>B4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>01101001</td>
<td>53</td>
<td>35</td>
<td>181</td>
<td>B5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>01101100</td>
<td>54</td>
<td>36</td>
<td>182</td>
<td>B6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>01101101</td>
<td>55</td>
<td>37</td>
<td>183</td>
<td>B7</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>01110000</td>
<td>56</td>
<td>38</td>
<td>184</td>
<td>B8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>01110001</td>
<td>57</td>
<td>39</td>
<td>185</td>
<td>B9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>01110100</td>
<td>58</td>
<td>3A</td>
<td>186</td>
<td>BA</td>
<td>;</td>
<td>;</td>
</tr>
<tr>
<td>01110101</td>
<td>59</td>
<td>3B</td>
<td>187</td>
<td>BB</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
<tr>
<td>01111000</td>
<td>60</td>
<td>3C</td>
<td>188</td>
<td>BC</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>01111001</td>
<td>61</td>
<td>3D</td>
<td>189</td>
<td>BD</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>01111010</td>
<td>62</td>
<td>3E</td>
<td>190</td>
<td>BE</td>
<td>&gt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>01111111</td>
<td>63</td>
<td>3F</td>
<td>191</td>
<td>BF</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>01000000</td>
<td>64</td>
<td>40</td>
<td>192</td>
<td>C0</td>
<td>@</td>
<td>@</td>
</tr>
<tr>
<td>01000001</td>
<td>65</td>
<td>41</td>
<td>193</td>
<td>C1</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>01000100</td>
<td>66</td>
<td>42</td>
<td>194</td>
<td>C2</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>01000101</td>
<td>67</td>
<td>43</td>
<td>195</td>
<td>C3</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>01001000</td>
<td>68</td>
<td>44</td>
<td>196</td>
<td>C4</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>01001001</td>
<td>69</td>
<td>45</td>
<td>197</td>
<td>C5</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>01001100</td>
<td>70</td>
<td>46</td>
<td>198</td>
<td>C6</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>01001101</td>
<td>71</td>
<td>47</td>
<td>199</td>
<td>C7</td>
<td>G</td>
<td>G</td>
</tr>
<tr>
<td>01010000</td>
<td>72</td>
<td>48</td>
<td>200</td>
<td>C8</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>01010001</td>
<td>73</td>
<td>49</td>
<td>201</td>
<td>C9</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>01010100</td>
<td>74</td>
<td>4A</td>
<td>202</td>
<td>CA</td>
<td>J</td>
<td>J</td>
</tr>
<tr>
<td>01010101</td>
<td>75</td>
<td>4B</td>
<td>203</td>
<td>CB</td>
<td>K</td>
<td>K</td>
</tr>
<tr>
<td>01011000</td>
<td>76</td>
<td>4C</td>
<td>204</td>
<td>CC</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>01011001</td>
<td>77</td>
<td>4D</td>
<td>205</td>
<td>CD</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>01011110</td>
<td>78</td>
<td>4E</td>
<td>206</td>
<td>CE</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>BINARY</td>
<td>LOW</td>
<td>LOW</td>
<td>HI</td>
<td>HI</td>
<td>ASCII</td>
<td>WHAT TO TYPE</td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
<td>-----</td>
<td>----</td>
<td>----</td>
<td>-------</td>
<td>--------------</td>
</tr>
<tr>
<td>1001111 79 4F</td>
<td>207 CF</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010000 80 50</td>
<td>208 D0</td>
<td>P</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010001 81 51</td>
<td>209 D1</td>
<td>Q</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010010 82 52</td>
<td>210 D2</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010011 83 53</td>
<td>211 D3</td>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010100 84 54</td>
<td>212 D4</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010101 85 55</td>
<td>213 D5</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010110 86 56</td>
<td>214 D6</td>
<td>V</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010111 87 57</td>
<td>215 D7</td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011000 88 58</td>
<td>216 D8</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011001 89 59</td>
<td>217 D9</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011010 90 5A</td>
<td>218 DA</td>
<td>Z</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011011 91 5B</td>
<td>219 DB</td>
<td>[</td>
<td>Opening Bracket</td>
<td>n/a</td>
<td>&lt;ESC&gt;3</td>
<td></td>
</tr>
<tr>
<td>1011100 92 5C</td>
<td>220 DC</td>
<td>\</td>
<td>Reverse Slant</td>
<td>n/a</td>
<td>&lt;ESC&gt;4</td>
<td></td>
</tr>
<tr>
<td>1011101 93 5D</td>
<td>221 DD</td>
<td>]</td>
<td>Closing Bracket</td>
<td>&lt;SHIFT-MO&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011110 94 5E</td>
<td>222 DE</td>
<td>^ Circumflex</td>
<td>n/a</td>
<td>&lt;ESC&gt;5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011111 95 5F</td>
<td>223 DF</td>
<td>~ Underline</td>
<td>n/a</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100000 96 60</td>
<td>224 E0</td>
<td>7 Opening Quote</td>
<td>n/a</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100001 97 61</td>
<td>225 E1</td>
<td>a</td>
<td>n/a</td>
<td>a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100010 98 62</td>
<td>226 E2</td>
<td>b</td>
<td>n/a</td>
<td>b</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100011 99 63</td>
<td>227 E3</td>
<td>c</td>
<td>n/a</td>
<td>c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100100 100 64</td>
<td>228 E4</td>
<td>d</td>
<td>n/a</td>
<td>d</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100101 101 65</td>
<td>229 E5</td>
<td>e</td>
<td>n/a</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100110 102 66</td>
<td>230 E6</td>
<td>f</td>
<td>n/a</td>
<td>f</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100111 103 67</td>
<td>231 E7</td>
<td>g</td>
<td>n/a</td>
<td>g</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101000 104 68</td>
<td>232 E8</td>
<td>h</td>
<td>n/a</td>
<td>h</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101001 105 69</td>
<td>233 E9</td>
<td>i</td>
<td>n/a</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101010 106 70</td>
<td>234 EA</td>
<td>j</td>
<td>n/a</td>
<td>j</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101011 107 71</td>
<td>235 EB</td>
<td>k</td>
<td>n/a</td>
<td>k</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101100 108 72</td>
<td>236 EC</td>
<td>l</td>
<td>n/a</td>
<td>l</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101101 109 73</td>
<td>237 ED</td>
<td>m</td>
<td>n/a</td>
<td>m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101110 110 74</td>
<td>238 EE</td>
<td>n</td>
<td>n/a</td>
<td>n</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101111 111 75</td>
<td>239 EF</td>
<td>o</td>
<td>n/a</td>
<td>o</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110000 112 76</td>
<td>240 F0</td>
<td>p</td>
<td>n/a</td>
<td>p</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110001 113 77</td>
<td>241 F1</td>
<td>q</td>
<td>n/a</td>
<td>q</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110010 114 78</td>
<td>242 F2</td>
<td>r</td>
<td>n/a</td>
<td>r</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110011 115 79</td>
<td>243 F3</td>
<td>s</td>
<td>n/a</td>
<td>s</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110100 116 80</td>
<td>244 F4</td>
<td>t</td>
<td>n/a</td>
<td>t</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110101 117 81</td>
<td>245 F5</td>
<td>u</td>
<td>n/a</td>
<td>u</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110110 118 82</td>
<td>246 F6</td>
<td>v</td>
<td>n/a</td>
<td>v</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110111 119 83</td>
<td>247 F7</td>
<td>w</td>
<td>n/a</td>
<td>w</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111000 120 84</td>
<td>248 F8</td>
<td>x</td>
<td>n/a</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111001 121 85</td>
<td>249 F9</td>
<td>y</td>
<td>n/a</td>
<td>y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111010 122 86</td>
<td>250 FA</td>
<td>z</td>
<td>n/a</td>
<td>z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111011 123 87</td>
<td>251 FB</td>
<td>{ Opening Brace</td>
<td>n/a</td>
<td>&lt;ESC&gt;6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111100 124 88</td>
<td>252 FC</td>
<td></td>
<td>Vertical Line</td>
<td>n/a</td>
<td>&lt;ESC&gt;7</td>
<td></td>
</tr>
<tr>
<td>1111101 125 89</td>
<td>253 FD</td>
<td>} Closing Brace</td>
<td>n/a</td>
<td>&lt;ESC&gt;8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111110 126 90</td>
<td>254 FE</td>
<td>~ Overline (Tilde)</td>
<td>n/a</td>
<td>&lt;ESC&gt;9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111111 127 91</td>
<td>255 FF</td>
<td>DEL delete/Rubberout</td>
<td>n/a</td>
<td>&lt;ESC&gt;:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

15. Use Closing Quote (39). For high value, use CHR$(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 Enable 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 caractrs</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 Enable 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;N) cmd</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 Ø</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; in SIC or Printer Mode, use I command or SW2-3 &amp; -4 ON</td>
</tr>
<tr>
<td>screen is seeing double</td>
<td>eevveerry ttthhiinnnngg ttwwiwiceee</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 bgnning 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 (300 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>
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:

$678+s

<table>
<thead>
<tr>
<th>Bit</th>
<th>&quot;1&quot; Means</th>
<th>&quot;0&quot; Means</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</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 Ø, 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 Ø, 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></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>(no error)</td>
<td></td>
<td>yes</td>
</tr>
<tr>
<td>33</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>34</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>35</td>
<td>no</td>
<td>no</td>
<td>yes</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>no</td>
</tr>
<tr>
<td>38</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>39</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>40</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>yes</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.
GLOSSARY

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 (Syntek 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 Ø.233 second SPACE (Ø) 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 Ø 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 <CTRL-A> or <CTRL-I> (Appendix D), that causes the SSC firmware to interpret subsequent characters as a command.

Command Register: An ACIA location (at hexadecimal address $C08Ah+sØ) 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 $C08Bh+sØ) 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 8Ø-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 19-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 $C088h$) 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 $\emptyset$ 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 $C089h$) 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 $C088h$) 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
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
9. Table 2-4 Printer Mode Commands
10. Figure 2-2 SSC in Slot #1 and Clamp Assembly in Notch
11. Table 2-5 Baud Rate Selections
12. Table 2-6 Data Format Selections
13. Table 2-7 Parity Selections
14. Table 2-8 Time Delay Selections
15. Table 2-9 Lowercase Character Displays

Chapter 3
COMMUNICATIONS MODE

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

**HOW THE SSC WORKS**

<table>
<thead>
<tr>
<th>Page</th>
<th>Figure/Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>Figure 4-1</td>
<td>Parallel Data Transfer</td>
</tr>
<tr>
<td>39</td>
<td>Figure 4-2</td>
<td>Serial Data Transfer</td>
</tr>
<tr>
<td>39</td>
<td>Figure 4-3</td>
<td>Parallel-to-Serial Data Conversion</td>
</tr>
<tr>
<td>40</td>
<td>Figure 4-4</td>
<td>RS-232-C Serial Data Format</td>
</tr>
<tr>
<td>42</td>
<td>Figure 4-5</td>
<td>An RS-232-C Setup with Modems</td>
</tr>
<tr>
<td>43</td>
<td>Figure 4-6</td>
<td>An RS-232-C Setup with a Modem Eliminator</td>
</tr>
<tr>
<td>43</td>
<td>Figure 4-7</td>
<td>SSC Operating Modes</td>
</tr>
<tr>
<td>44</td>
<td>Figure 4-8</td>
<td>SSC Configurations</td>
</tr>
<tr>
<td>45</td>
<td>Figure 4-9</td>
<td>Overall Block Diagram of the SSC</td>
</tr>
<tr>
<td>38</td>
<td>Table 4-1</td>
<td>Binary and Decimal Digits and Quantities</td>
</tr>
<tr>
<td>41</td>
<td>Table 4-2</td>
<td>RS-232-C Signals as Interpreted by the Sender</td>
</tr>
<tr>
<td>41</td>
<td>Table 4-3</td>
<td>RS-232-C Signals as Interpreted by the Receiver</td>
</tr>
<tr>
<td>46</td>
<td>Table 4-4</td>
<td>SSC Address Remapping</td>
</tr>
<tr>
<td>47</td>
<td>Table 4-5</td>
<td>Registers in SSC Peripheral I/O Space</td>
</tr>
</tbody>
</table>

**Appendix A**

**FIRMWARE**

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

**Appendix B**

**APPLE INTERFACE CARD EMULATION**

<table>
<thead>
<tr>
<th>Page</th>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>93</td>
<td>B-1</td>
<td>SIC Switch Settings, PEEKs and POKEs, Part I</td>
</tr>
<tr>
<td>94</td>
<td>B-2</td>
<td>SIC Switch Settings, PEEKs and POKEs, Part II</td>
</tr>
</tbody>
</table>
## Appendix C
**SPECIFICATIONS AND SCHEMATICS**

<table>
<thead>
<tr>
<th>98</th>
<th>Table C-1</th>
<th>Connector Pin Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>Table C-2</td>
<td>Jumper Block Wiring</td>
</tr>
</tbody>
</table>

## Appendix E
**TROUBLESHOOTING HINTS**

<table>
<thead>
<tr>
<th>106</th>
<th>Table E-1</th>
<th>Problems Detected at the Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>Table E-2</td>
<td>Problems Detected at the Apple</td>
</tr>
</tbody>
</table>

## Appendix F
**ERROR CODES**

<table>
<thead>
<tr>
<th>109</th>
<th>Table F-1</th>
<th>STSBYTE Bit Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>Table F-2</td>
<td>Error Codes and Bits</td>
</tr>
</tbody>
</table>
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
entry 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

C

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
Printer Mode 12, 102
send 12
data format 15
format 12, 27
Communications Mode 27-28
register 45
remote-control 102
reset 15, 95
set time delay 16
summary, Terminal Mode 35
translate 91
commands
CIC 96
Communications Mode 27, 29, 102
Parallel Card 96
Printer Mode 11, 14
Terminal Mode 22, 35
communications 37
mode 21-33, 91, 95, 96
Communications Card,
Apple II (see CIC)
Communications Mode 22, 26
command
caracter 28
summary 28-29
commands 27
control character 27
components SSC 45
configurations, SSC 44
connector
pin assignments 98
slots 9, 25, 98
control
character
communications mode 27
suppress 18, 33
logic 45, 46
register 45
<CR> 7, 17, 94, 95, 96, 102, 107
<CTRL-A> 27, 102
<CTRL-I> 8, 12, 102
<CTRL-R> 28, 35, 96, 102
<CTRL-S> 28, 96, 102
<CTRL-SHIFT-M> 101, 103
<CTRL-SHIFT-N> 103
<CTRL-T> 28, 35, 96, 102
CTS 41, 42, 43, 47, 48, 95, 98, 99

D

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
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
   <ESC>L 91
   <ESC>U 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
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
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
emulation 94
entry points 50, 55-57
environment 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 STBYTE)
register, ACIA 109
status/reset register 45
stop bit 7, 15, 23, 24, 30, 40, 92, 94
STBYTE 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
SW1 48, 54, 56, 57, 92, 93
SW1-1 through SW1-4 2, 7,
15, 23, 31, 92, 93, 106
SW1-5 and SW1-6 2, 5, 21,
92, 93, 102, 115
SW1-7 2, 8, 24, 48, 92, 99
SW2 6, 22, 48, 54, 56, 57,
92, 93, 94
SW2-1 2, 7, 15, 23-24, 31,
92, 93, 94, 106
SW2-2 2, 7, 16, 23-24, 31,
92, 94, 106
SW2-3 and SW2-4 2, 8, 23,
92, 94, 106, 107
128 SUPER SERIAL CARD
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, 103
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

U

unintelligent terminal 34, 35, 36
upercase 14, 18, 29, 33, 34, 91, 92, 102

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
Tuck end flap inside back cover when using manual.