RM 65 FAMILY

RM 65 CRT CONTROLLER MODULE

USER'S MANUAL

RM 65 FAMILY

Rockwell International

RM 65 FAMILY

RM 65 CRT CONTROLLER MODULE

USER'S MANUAL

RM 65 FAMILY

Rockwell International

Document No.
29801N14
Order No. 814
November 1981

NOTICE

Rockwell International does not assume any liability arising out of the application or use of any products, circuit, or software described herein, neither does it convey any license under its patent rights nor the patent rights of others. Rockwell International further reserves the right to make changes in any products herein without notice.

TABLE OF CONTENTS

Section		Page
1	INTRODUCTION	
	1.1 Purpose/Function	1-1
	1.2 Features	1-3
	1.3 Characteristics	1-3
	1.4 Reference Documents	1-3
2	INSTALLATION	
	2.1 Unpacking	2-1
	2.2 Operating Options	2-1
	2.2.1 Base Address Select	2-4
	2.2.2 Bank Select	2-4
	2.2.3 Dot Clock Select	2-6
	2.2.4 Program ROM Select	2-6
	2.2.5 Program ROM Enable	2-7
	2.2.6 Composite Video Amplitude Adjustment	2-8
	2.3 Installing the Module	2-8
	2.4 Removing the Module	2-10
3	OPERATION	
	3.1 Initialization Program	3-1
	3.1.1 General Initialization Procedure	3-2
	3.1.2 Language Variations	3-4
	3.2 The Display Program	3-12
	3.2.1 AIM 65 Output Through DILINK	3-13
	3.2.2 AIM 65 Active Output Device = User.	3-13
	3.2.3 Direct Call to Command Data Processing	3-14
	3.2.4 Direct Store into Refresh RAM	3-16
	3.2.5 Store into Refresh RAM using the Display Program	3-17
	3.3 Display Program Commands	3-18
	3.3.1 Control Command Definition	3-18
	3.3.2 Escape Command Definition	3-24
	3.3.3 Backspace Command Definition	3-24
	3.4 Character Data ROM Formats	3-25
	3.5 AIM 65 Assembler Reformatter	3-27
	3.5.1 Using the Reformatter	3-27
	3.5.2 Reformatter Output Format	3-30
	3.5.3 Reformatter Variables	3-31
4	FUNCTIONAL AND INTERFACE DESCRIPTION	
	4.1 Functional Description	4-1
	4.2 Interface Description	4-6

APPENDIX A ASCII Character Set
APPENDIX B Preprogrammed and User Defined Screen Format Values
APPENDIX C Program ROM Variables
APPENDIX D Useful Program ROM Subroutines
APPENDIX E Program ROM Assembly Listing
APPENDIX F Character ROM Listing
ENCLOSURE CRT Controller Module Schematic

LIST OF FIGURES

Figure		Page
1-1	CRT Controller Module	1-2
1-2	CRT Controller Module Outline	1-5
2-1	CRT Controller Module Detail	2-2
3-1	CRT Controller Module Normal/Semi-Graphic Characters	3-26
3-1	Typical AIM 65 Assembler Reformatter Output	3-28
4-1	CRT Controller Module Block Diagram	4-2
4-2	CRT Controller Module Memory Map	4-3
B-1	Screen Parameters	B-2

LIST OF TABLES

Table		Page
1-1	CRT Controller Module Physical and Electrical Characteristics	1-4
2-1	CRT Controller Module Switches, Jumpers and Connectors	2-3
2-2	Base Address Select Switch Positions	2-5
2-3	Bank Select Switch Positions	2-5
2-4	Dot Clock Select Jumper Positions	2-6
2-5	Program ROM Select Jumper Positions	2-7
2-6	Program ROM Enable Jumper Positions	2-8
3-1	Display Control Commands	3-19
3-2	Display Escape Commands	3-24
3-3	AIM 65 Assembler Reformatter Output Format	3-31
4-1	CRT Controller Module I/O Addresses	4-4
4-2	Connector P1 (RM 65 Bus) Pin Assignments	4-7
4-3	Connector P3 (Separate Video and Sync) Pin Assignments	4-9
4-4	Connector P1 (RM 65 Bus) Signal Descriptions	4-10
B-1	Firmware Screen Format Tables	B-3
B-2	Firmware Screen Parameter Table	B-4
C-1	Program ROM Page Zero Variables	C-1
C-2	Program ROM Page Three Variables	C-2
C-3	Variable TABLE Options	C-5
C-4	Variable AIM 65 Options	C-7
D-1	Program ROM Subroutines	D-1

SECTION 1

INTRODUCTION

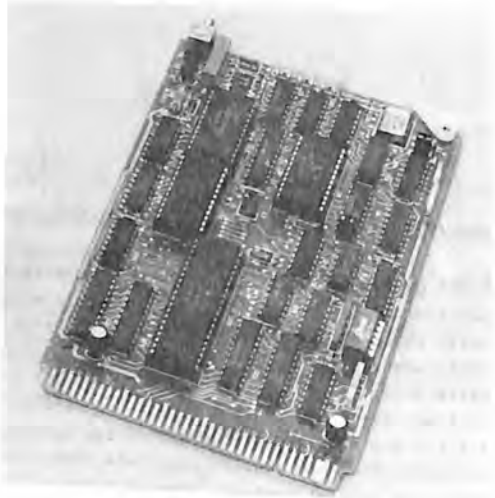
1.1 PURPOSE/FUNCTION

The RM 65 CRT Controller (CRTC) module interfaces an AIM 65 microcomputer or RM 65 SBC module based system to a CRT monitor or television receiver. The CRTC module outputs HSYNC, VSYNC, and raw video signals for direct connection to a CRT monitor, and composite video for connection to a CRT monitor or to a TV receiver through an RF modulator. A socketed on-board ROM generates 5 x 7 dot matrix characters with two descenders in a 7 x 10 dot matrix field to provide upper and lower case alphanumerics and special symbols. The 2K bytes of on-board display RAM are memory-mapped on the RM 65 Bus.

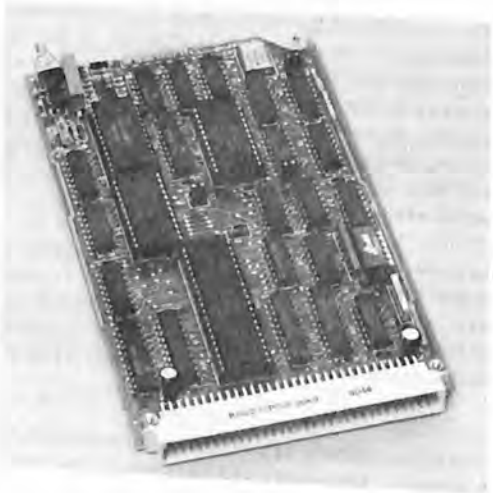
A 2K-byte program ROM provides firmware to configure the display to user-defined formats. Four user-selectable formats are preprogrammed -- two at 60 Hz and two at 50 Hz; one format at each scan rate can be used with a wide bandwidth CRT monitor while the other format can be used with a standard TV receiver. Cursor control, screen editing and utility functions are also provided. In addition, the ROM includes an AIM 65 assembler reformatter that outputs the AIM 65 assembly listing in a 60 column format with addresses on each line.

Bank Select and Bank Select Enable switches dedicate the module to one of two 65K-byte memory banks, or assign it common to both banks. A jumper allows the on-board ROM to be enabled or disabled. Base address select switches allow the module I/O address to be assigned to any page (256 bytes) if the ROM is disabled.

The CRT Controller module is available in a 72-pin Edge Connector version (RM65-5102) and in a 64-pin Eurocard version (RM65-5102E). Both versions are shown in Figure 1-1. The pin assignments of the two versions are identical except that the edge connector version has four additional pins connected to +5 Vdc, and four unused pins (see Table 4-2 for pin assignments).



a. Edge Connector Version



b. Eurocard Version

Figure 1-1. CRT Controller Module

1.2 FEATURES

- RM 65 Bus compatible
- Compact size - about 4" x 6-1/4" (100 mm x 160 mm)
- 4K-byte Character ROM with:
  - Upper and lower case alphabetics
  - Special, math and semi-graphic characters
  - Numbers including subscripts and superscripts
- 2K-byte Program ROM supports:
  - Scrolling
  - Screen editing
  - Full cursor movement control
  - Full screen standard or inverse video
  - Predefined formats for
    - 25 rows by 80 columns (50 Hz)
    - 22 rows by 72 columns (60 Hz)
    - 25 rows by 40 columns (50 Hz)
    - 16 rows by 40 columns (60 Hz)
  - Selectable format from 1 to 80 columns by 1 to 25 rows
  - NTSC (60 Hz, 525 lines per frame) and European (50 Hz, 625 lines per frame) raster format
  - CRT display driver for AIM 65 microcomputer
  - AIM 65 Assembler reformatter
- Single +5 volt operation
- Fully assembled, tested and warranted

1.3 CHARACTERISTICS

The physical and electrical characteristics of the CRTC module are listed in Table 1-1. Figure 1-2 shows the module outline.

1.4 REFERENCE DOCUMENTS

Rockwell Document No.	
29650N30	R6500 Programming Manual
29650N31	R6500 Hardware Manual
29650N36	AIM 65 Microcomputer User's Manual
29000D67	R6545-1 CRT Controller Data Sheet

Table 1-1. CRT Controller Module Physical and Electrical Characteristics

Characteristics	Value
Dimensions (See Figure 1-2)	
Edge Connector Version	
Width	3.9 in. (100 mm)
Length (see 2 below)	6.5 in. (164 mm)
Height (see 1 below)	0.56 in. (14 mm)
Weight	4.6 oz. (130 g)
Eurocard Version	
Width	3.94 in. (100 mm)
Length (see 2 below)	6.30 in. (160 mm)
Height (see 1 below)	0.56 in. (14 mm)
Weight	5.0 oz. (140 g)
Environment	
Operating Temperature	0°C to 70°C
Storage Temperature	-40°C to 85°C
Relative Humidity	0% to 85% (without condensation)
Power Requirements	+5 Vdc ± 5% 0.94A (4.7 W) - typical 1.3 A (6.8 W) - maximum
Edge Connector Version	72-pin edge connector (0.100 in. centers)
Eurocard Version	64-pin plug (0.100 in. centers per DIN 41612 (Row b not installed)
Module I/O Interface	
Composite Video	Mini-coax connector (50 ohm SMC type). Mates to Sealectro Part No. 050-024-0000-220 or equivalent
Raw Video and Sync	6-pin connector. Mates to AMP No. 87159-6 or equivalent

NOTES

1. The height includes the maximum values for component height above the board surface (0.4 in. for populated modules), printed circuit board thickness (0.062 in.), and pin extension through the module (0.1 in.).
2. The length does not include the module ejector.
3. The Eurocard dimensions conform to DIN 41612.

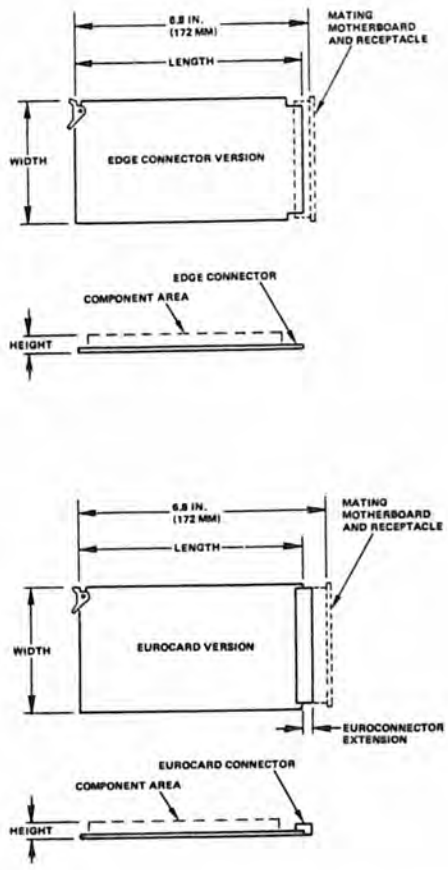


Figure 1-2. CRT Controller Module Outline

SECTION 2

INSTALLATION AND INITIALIZATION

2.1 UNPACKING

Unpack the CRT Controller module from its shipping carton and refer to the packing sheet to verify that all of the parts are included. Save the packing material for storing the module.

> **CAUTION**
>
> This module contains voltage-sensitive items. The module should be stored in an anti-static container when not in use and anyone handling the unit should observe anti-static precautions. Damage to the unit may result if this protection is not maintained.

2.2 OPERATING OPTIONS

Four operating options are switch or jumper selectable:

- Base address selection
- Bank selection
- Program ROM selection
- Dot clock selection

Figure 2-1 identifies the detail on the CRTC module. The function of each switch, jumper, and connector is identified in Table 2-1 along with reference to the section and table that describes its use.

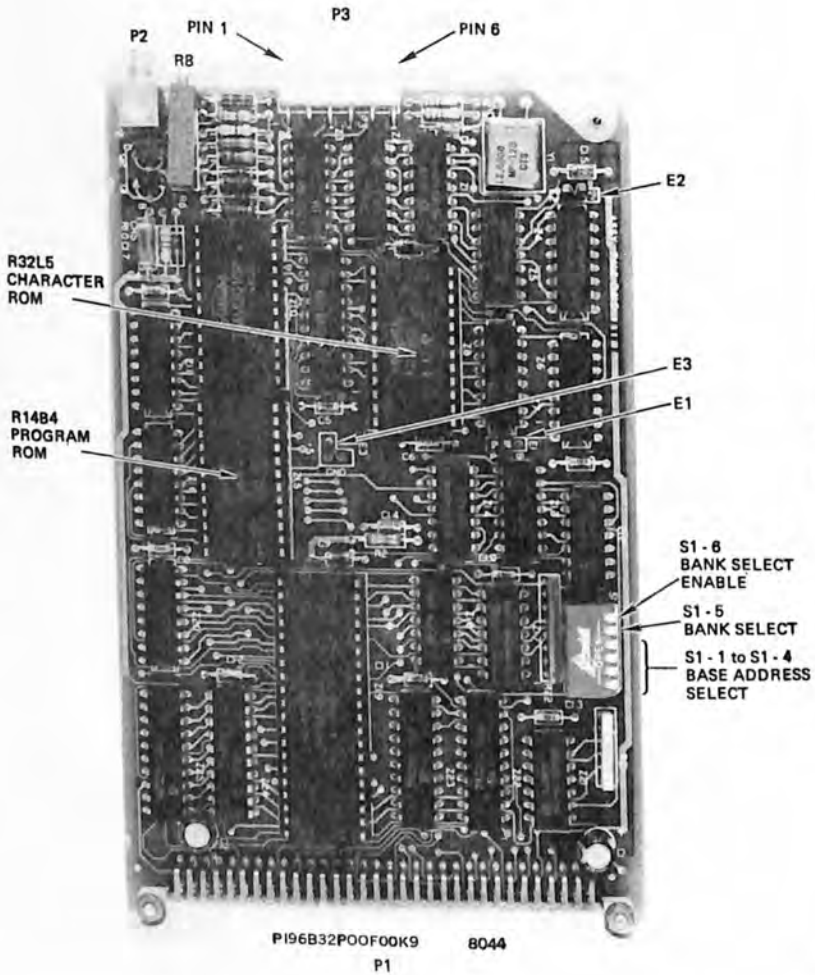


Figure 2-1. CRT Controller Module Detail

Table 2-1. CRT Controller Module Switches, Jumpers, and Connectors

Category	Item	Description	Reference
Switches	S1-1 to S1-4	Base Address Select	Section 2.2.1 Table 2-2
	S1-5	Bank Select	Section 2.2.2 Table 2-3
	S1-6	Bank Select Enable	
Jumpers	E1	Program ROM Select	Section 2.2.3 Table 2-4
	E2	Dot Clock Select	Section 2.2.4 Table 2-5
	E3	Program ROM Enable	2.2.5 Table 2-6
Connectors	P1	RM 65 Bus Connector	Tables 4-2 and 4-4
	P2	Composite Video	
	P3	Separate Video	Table 4-3

2.2.1 Base Address Select

The CRTC module Program ROM, Display RAM and I/O can be assigned to any one of sixteen 4K byte blocks in either 65K bank by setting the four Base Address Select switches (S1-1 to S1-4). The Display RAM is assigned the lower eight 256-byte pages ($X000-$X7FF) within the selected 4K blocks. The I/O logic is assigned to the ninth page ($X800-$X8FF) within the selected 4K block. The Program ROM (when selected) is assigned the upper seven pages ($9400-$9FFF) within the selected 4K block. The selected base address must be $9000 if the on-board ROM is to be selected. The switch positions are described in Table 2-2.

2.2.2 Bank Select

The Bank Select Enable switch (S1-6), in conjunction with the Bank Select switch (S1-5), allows the module to be assigned common to both memory banks (Bank 0 and Bank 1) or to be dedicated to a selected memory bank (either Bank 0 or Bank 1). When OPEN, the Bank Select Enable switch assigns the I/O logic and Program ROM to both banks regardless of the position of the Bank Select switch. When the Bank Select Enable switch is CLOSED, the assigned bank is determined by the position of the Bank Select switch. See Table 2-3 for the switch positions.

In applications where the module is to be addressed by a microcomputer that does not have bank addressing capabilities, the module must be assigned common to Bank 0 and Bank 1, or dedicated to Bank 0.

Table 2-2. Base Address Select Switch Positions

4K Base Address (Hexadecimal)	Switch Position			
	S1-4	S1-3	S1-2	S1-1
0000	OPEN	OPEN	OPEN	OPEN
1000	OPEN	OPEN	OPEN	CLOSED
2000	OPEN	OPEN	CLOSED	OPEN
3000	OPEN	OPEN	CLOSED	CLOSED
4000	OPEN	CLOSED	OPEN	OPEN
5000	OPEN	CLOSED	OPEN	CLOSED
6000	OPEN	CLOSED	CLOSED	OPEN
7000	OPEN	CLOSED	CLOSED	CLOSED
8000	CLOSED	OPEN	OPEN	OPEN
9000	CLOSED	OPEN	OPEN	CLOSED
A000	CLOSED	OPEN	CLOSED	OPEN
B000	CLOSED	OPEN	CLOSED	CLOSED
C000	CLOSED	CLOSED	OPEN	OPEN
D000	CLOSED	CLOSED	OPEN	CLOSED
E000	CLOSED	CLOSED	CLOSED	OPEN
F000	CLOSED	CLOSED	CLOSED	CLOSED

Table 2-3. Bank Select Switch Positions

Memory Bank Selected	Switch Position	
	Bank Select Enable Switch S1-6	Bank Select Switch S1-5
Bank 0 and 1	OPEN	EITHER
Bank 0 (Lower 65K)	CLOSED	OPEN
Bank 1 (Upper 65K)	CLOSED	CLOSED

2.2.3 Dot Clock Select

Jumper E2 selects either 6 MHz or 12 MHz frequency for the dot clock (see Table 2-4).

When jumper E2 is in the A position, the 6 MHz clock is selected. With the 6 MHz clock selected, up to 40 characters per line can be displayed on a CRT monitor or standard television set using an RF modulator.

When jumper E2 is in the B position, the 12 MHz clock is selected. With the 12 MHz clock selected, up to 80 characters per line may be displayed on a high bandwidth monitor.

Table 2-4. Dot Clock Select Jumper Positions

Dot Clock Frequency	Jumper E2 Position
6 MHz	A
12 MHz	B

2.2.4 Program ROM Select

Jumper E1 allows the module to operate with the Program ROM, RAM and I/O active, or with the RAM and I/O only (see Table 2-5).

When jumper E1 is in position B, only the RAM and I/O are selected, and the module is active in the lower nine 256-byte pages assigned by the four Base Address select switches. The RAM is the lower eight pages (2048 bytes) and the I/O is the ninth page (256 bytes).

When jumper E1 is in position A, the Program ROM, RAM and I/O are all selected. The module is active in the entire 4K block with the Program ROM active in the seven remaining pages.

To use the Program ROM firmware supplied with the module, the Program ROM must be selected and the module Base Address assigned to $9000.

The module may be assigned to any Base Address with user firmware, with the Program ROM selected (user device installed) or deselected.

Table 2-5. Program ROM Select Jumper Positions

Program Source	Program ROM	Jumper E1 Position
Supplied Firmware (on-board)	Selected	E1 = A
User Firmware (on-board)	Selected	E1 = A
User Firmware (off-board)	Deselected	E1 = B

2.2.5 Program ROM Enable

Jumper E3 enables the Program ROM to operate either from the lower or upper half of the 4K-bytes available in the program ROM socket (see Table 2-6). This allows two different display programs to be installed in the Program ROM but only one selected for use at any given time. The object code in each half of the ROM must be programmed in the address range $X000 - $XFFF (see Table 2-6). Note that I/O is assigned from $X800 - $X8FF (see Figure 4-2).

When jumper E3 is in the GND position, the CRTC module will enable the lower half of the 4K ROM. If program ROM R14B4 is installed, jumper E3 must be installed in this position and the base address set to $9000 (see Section 2.2.1).

When jumper E3 is in the +5 position, the CRTC module will enable the upper half of the 4K ROM.

Table 2-6. Program ROM Enable Jumper Positions

Program ROM Half Enabled	ROM Address	Program Address	Jumper E3 Position
Lower Half	$X100-$X7FF	$X900-$XFFF	GND
Upper Half	$X900-$XFFF	$X900-$XFFF	+5

2.2.6 Composite Video Amplitude Adjustment

The setting of potentiometer R8 determines the composite video signal output level. R8 is adjusted fully counter-clockwise at the factory. If the composite video is too high for your television receiver or CRT monitor, adjust R8 clockwise until the desired brightness level is attained. Note that too high a level may cause slight "ghosting" of un-displayed characters beyond the display field.

2.3 INSTALLING THE MODULE

Before installing the module, ensure that it is not damaged and is free of grease, dirt, liquid or other foreign matter.

> **CAUTION**
>
> Prior to module installation, turn off power to the RM 65 bus.
>
> Also, turn-off power to the RM 65 Bus when the CRT Controller module is installed prior to changing switch or jumper positions.

a. Based on the RM 65 system memory map and requirements (refer to Section 4), select the proper module operating options relating to the RM 65 bus interface:

1. Select the module base address by positioning switches S1-1 to S1-4 (refer to Section 2.2.1 and Table 2-2). If the Program ROM routines will be used, the address must be $9000.

2. Select common or dedicated bank operation for the module by positioning switches S1-5 and S1-6 (refer to Section 2.2.2 and Table 2-3).

3. Select or deselect the Program ROM by positioning jumper E1 (refer to Section 2.2.3 and Table 2.4). If the Program ROM routines will be used, the ROM must be installed and selected.

4. Select the Dot Clock frequency by positioning jumper E2 (refer to Section 2.2.4 and Table 2-5).

b. Align pin Wa (for Edge Connector version) or pin 1a (for Eurocard version) of the module with the identical pin on the mating RM 65 Bus receptacle.

> **CAUTION**
>
> RM 65 connectors are keyed to prevent improper module connection. If the module does not insert into the receptacle with moderate pressure applied, check the orientation and the connector alignment of the module. Forcing the module improperly into the receptacle will damage the receptacle and/or the module.

c. Insert the module into the desired card slot (if a card cage is used) and position it in front of the mating receptacle.

d. Press in firmly on the end of module until all pins are securely seated.

e. Connect the interface cable to P2 (Composite Video) or P3 (Separate Video and Sync) I/O connectors on the module (see Figure 2-1).

f. Reapply power to the RM 65 Bus.

2.4 REMOVING THE MODULE

a. Turn-off power to the RM 65 bus.

b. Disconnect the cables from the I/O connectors.

c. If the module is installed in a card cage, lift up on the module ejector tab to release the module from the mating receptacle. Pull the module straight back until it is free from the module guides.

d. If the module is installed in a single card adapter, or in a motherboard without a card cage, pull back on the module while moving it slightly from side to side until it is free from the mating receptacle.

SECTION 3

OPERATION

This section describes the operation of the CRT Controller module using the 2K-byte Program ROM (ID=R14B3). This program facilitates the use of a CRT monitor or TV set as the output medium for the AIM 65 microcomputer or an RM 65 based system compatible with the AIM 65 memory map. The module can also be operated in conjunction with user-provided software. In this case, the operation of the module is dependent upon the programming of the R6545-1 CRTC device (refer to the R6545-1 data sheet for the CRTC device operating characteristics and to Table 4-1 for the I/O addresses). The programming of the R6545-1 as described in Appendix B for use with the R14B3 Program ROM may also be helpful.

The Program ROM firmware is divided into three main programs: the Initialization program, the Display program and the AIM 65 Assembler Reformatter.

3.1 INITIALIZATION PROGRAM

The Initialization Program must be called before any information can be displayed on the video screen. This program, depending upon the parameters passed to it, will initialize the module to varying screen sizes and scan frequencies. One of the four pre-defined screen sizes or a completely user-defined screen format can be chosen. This gives the flexibility needed for most video requirements in the U.S. or Europe.

There are two different entry points for the Initialization program:

- Entry at $9900 initializes the CRTC program for AIM 65 keyboard operation then returns control to the AIM 65 Monitor with a JMP COMIN ($E1A1).

- Entry at $9906 initializes the CRTC program for program operation then returns control to the calling program with an "RTS".

There is also a re-entry point:

- Entry at $9964 is the "warm start" entry point for keyboard or program operation. This entry point may be called if the video screen has already been initialized and the display output has been re-linked to the AIM 65 single line display (see CTRL Y command in Section 3.3.1). It will clear the screen, home the cursor and set the AIM 65 Monitor display linkage variable DILINK to point to the CRT firmware according to the value in the variable AIM65, as described in Section 3.1.1c.

3.1.1 General Initialization Procedure

a. CRT Display Entry

The general procedure to initialize the CRTC module firmware is the same whether performed under operator control or program control - only the entry points are different. The basic procedure is:

1. Set variable TABLE ($35A) to specify either a preprogrammed screen size, a user defined screen format, or the default size. The five options (explained in more detail in Appendix C) are:

TABLE Value	Rows/Columns	Screen Lines	Jumper E2
0	25 x 80	625	B
1	22 x 72	525	B
2	25 x 40	625	A
3	16 x 40	525	A
4	User Defined	525/625	A/B
>4	22 x 72	525	B

In the simplest case of initializing the CRT, that of using one of the pre-defined screen parameter tables, only the variable TABLE must be set to indicate which of the stored tables to use.

2. If the screen format is user defined, load the screen parameters and set the variable XADDR ($0363 - $0364) to point to them (see Appendix B).

3. Set variable AIM65 ($357) with a value to specify either skipping or performing AIM 65-related processing, and for the latter, either accepting or ignoring Line Feed characters (see Appendix C):

AIM65 Value	Function
0	Skip AIM 65-related processing
1	Perform AIM 65-related processing and ignore line feed characters
>1	Perform AIM 65-related processing and accept line feed characters

4. Initialize the program counter to the keyboard ($9900) or program ($9906) entry point.

5. Execute the Initialization program. The display will switch from the AIM 65 single line display (which will blank) to the CRT display.

6. If the terminal width of the support software, e.g., BASIC, is different from 59 ($3B), enter the terminal width in number of characters into variable CMAX (see Appendix C).

Once the TABLE and AIM65 variables have been initialized, re-entry from the keyboard after return to the AIM 65 display (see Step b.) can be performed easily by just

setting the program counter to $9900 and executing the Initialization program. If the TABLE or AIM65 variables are accidently altered or if the CMAX variable needs to be initialized, it may be easier to use a function key to perform the initialization under program control.

b. Return to AIM 65 Display

To switch the display back to the AIM 65 single line display, issue a CTRL Y command to the CRTC module.

c. CRT Display Re-entry

It is also possible to re-enter the Initialization program with a JSR to the warm start entry point at $9964. This entry point bypasses the screen format setup (it assumes that the screen has previously been initialized via the $9900 or $9906 entry) but still clears the screen, homes the cursor, and sets the AIM 65 Monitor display linkage variable DILINK according to the variable AIM65 value (see Section 3.1.2).

If re-entry is desired from the keyboard, performing the linkage through a function key provides convenient single keystroke operation.

For example, the following F1 linkage and instructions will re-enter the CRT program

```
010C 4C JMP 0FC0

0FC0 20 JSR 9964
0FC3 60 RTS
```

Now use the CTRL Y command to return to the AIM 65 display and the F1 key to return to the CRT.

3.1.2 Language Variations

The different languages available for the AIM 65 microcomputer each have a unique start-up procedure to enable use with the

CRTC module. The following section describes these peculiarities for each of the available languages and how to insure proper operation with the CRTC module.

a. AIM 65 Monitor/Editor

1. Enter the AIM 65 Monitor.

2. Set variable TABLE to desired screen size.

3. Set variable AIM65 >1 to accept Line Feeds.

4. Run the CRT Initialization routine ($9900).

5. Leave the variable CMAX value at 59 to correspond to the Editor line width.

6. Initialize the Editor text buffer above $3B5, e.g. $400, to prevent overwriting the CRT module variables.

   For example:

```
<M>=035A XX XX XX XX
</> 035A 05              Load TABLE
<M>=0357 XX XX XX XX
</> 0357 02              Load AIM65
<*>=9900
<G>/.                    Initialize CRTC Module
<E>                      Enter Editor
EDITOR
FROM=400  TO=FFF         Locate Text Buffer
```

NOTE

Any control characters entered into the Editor text buffer will execute their designated function if they are listed out to the CRT (See Section 3.3).

CAUTION

When using the Editor, reading or changing a line to a character length greater than 59 may cause the display to lock up or the apparent loss of data. If this occurs, return to the Monitor, with a CTRL Y, change CMAX to a value large enough to hold the offending line, edit the line to less than 59 characters, and return the CMAX value to 59 ($3B). Then return the CRT Initialization program or re-enter the CRT processing.

b. AIM 65 BASIC Entry Procedure

1. Enter BASIC using the "5" key.
2. Enter the memory size and terminal width in response to prompts.
3. Press ESC to return to the Monitor.
4. Set variable TABLE to the desired screen size.
5. Set variable AIM65 >1 to accept Line Feeds.
6. Run the CRT Initialization routine ($9900).
7. Change variable CMAX to match the entered terminal width.
8. Move the BASIC program and variables start address from $0211 to greater than $3B5, e.g. $400, to prevent overwriting of CRTC module variables. Also, load zeros into three bytes at the start of the BASIC program.

Location	Was	Change To	
0073	12	01	Program Start Address =
0074	02	04	400+1
0075	14	03	Variable Start Address =
0076	02	04	403
0400	XX	00	Load Zero into first
0401	XX	00	Three Program Bytes
0402	XX	00	

9. Re-enter BASIC with the "6" Key.

CAUTION

Do not enter BASIC with the "5" key when the display output is being sent to the CRT. The CRT variables will be overwritten and improper CRTC module operation will occur. Should this happen, you may be able to recover the AIM 65 Microcomputer display by pressing CTRL Y. If this does not work, you may have to cycle computer power off, then on, to recover.

Example 1: Total initialization from the keyboard.

```
<5>
MEMORY SIZE?
WIDTH? 72
 11758 BYTES FREE
  AIM 65 BASIC V1.1
  AIM 65 BASIC V1.1        ESC
<M>=035A XX XX XX XX
</> 035A 05
<M>=0357 XX XX XX XX       Load TABLE
</> 0357 02
<*>=9900                   Load AIM65
<G>/.
<M>=035E 3B 03 02 03       Initialize CRTC module
</> 035E 48
<M>=0073 12 02 14 02       Change CMAX to 72
</> 0073 01 04 03 04       Change BASIC Program
<M>=0400 XX XX XX XX       Starting Address
</> 0400 00 00 00           Change BASIC Initial
                            values
<6>
                           Re-enter BASIC
```

Example 2: Initialization from the keyboard using F1 key linkage and program call to CRT initialization program.

Function Key Linkage and Routine

```
010C 4C JMP 0FA0

0FA0 A9 LDA #05
0FA2 8D STA 035A           Load TABLE
0FA5 A9 LDA #02
0FA7 8D STA 0357           Load AIM65
0FAA 20 JSR 9906           Initialize CRT module
0FAD A9 LDA #48
0FAF 8D STA 035E           Load CMAX
0FB2 A9 LDA #01            Load Program Start Address
0FB4 8D STA 0073
0FB7 A9 LDA #04
0FB9 8D STA 0074
0FBC A9 LDA #03
0FBE 8D STA 0075
0FC1 A9 LDA #04
0FC3 8D STA 0076
0FC6 A9 LDA #00            Initialize Program Start Area
07C8 8D STA 0400
0FCB 8D STA 0401
0FCE 8D STA 0402
0FD1 60 RTS
                           Return to Monitor
```


Initialization Procedure

```
<5>
MEMORY SIZE? 4000          Upper Limit = $0FA0
WIDTH?                     RETURN = 72
 3470 BYTES FREE
  AIM 65 BASIC V1.1
  AIM 65 BASIC V1.1        ESC
<[>                        Press F1 key
<6>                        Re-enter BASIC
```

> **NOTE**
>
> Be sure to enter the upper memory limit to prevent BASIC from overwriting your initialization routine at $FA0-$FD1 with $AA's.

c. AIM 65 FORTH Entry Procedure

1. Set variable TABLE to the desired screen size.
2. Set variable AIM65 >1 to accept Line Feeds.
3. Run the CRT Initialization routine ($9900).
4. Enter FORTH with the "5" key.
5. Move the FORTH starting address from $300 to greater than $3B5 to prevent overwriting the CRT module variables.

> **NOTE**
>
> Any time FORTH is entered with the "5" Key, the above steps must be repeated.

6. Leave the variable CMAX value at 59 since the FORTH line length is 59.

   For example:

```
<M>=035A XX XX XX XX
</> 035A 05 XX XX XX      Load TABLE
<M>=0357 XX XX XX XX
</> 0357 02 XX XX XX      Load AIM65
<*>=9900
<G>/.                     Initialize CRTC module
<5>                       Enter FORTH
 AIM 65 FORTH V1.2
FORGET TASK
185 ALLOT
:TASK ;
```

d. AIM 65 Instant Pascal

1. Enter Instant Pascal with the "5" key.

2. Enter the memory size and terminal width in response to prompts.

3. ESC to the Monitor.

4. Change the Instant Pascal variable value at location $1A-$1B from $02FC to $03FC to move the start of the Pascal program from $300 to $400 (in order to prevent the Pascal program from overwriting the CRT firmware variables).

5. Set variable TABLE to the desired screen size.

6. Set variable AIM65 >1 to accept Line Feeds.

7. Re-enter PASCAL with the "5" key.

> **NOTE**
>
> Any time a Cold Reset (Pascal Z command) is performed, location $1A-$1B will be reset to $02FC.

8. Change variable CMAX from $3B to coincide with the terminal width entered for the Pascal output format.

For example:

```
<5>MEMORY SIZE?           Enter Pascal
WIDTH?                    Escape
<M>=001A FC 02 XX XX      Change starting address
</> 001A FC 03 XX XX
<M>=035A XX XX XX XX
</> 035A 05 XX XX XX      Load TABLE
<M>=0357 XX XX XX XX
</> 0357 02 XX XX XX      Load AIM65
<*>=9900
<G>/.                     Initialize CRTC module
<5>                       Re-enter Pascal
```

e. AIM 65 PL/65 Compiler

1. Set variable TABLE to the desired screen size.

2. Set variable AIM65 to 1 to tell the CRTC module to ignore Line Feed characters sent from PL/65 at the end of each output line to prevent double line spacing and incorrect scrolling.

3. Run the CRT Initialization routine ($9900).

4. Leave variable CMAX value at 59.

   For example:

```
<M>=035A XX XX XX XX
</> 035A 05 XX XX XX      Load TABLE
<M>=0357 XX XX XX XX
</> 0357 02 XX XX XX      Load AIM65
<*>=9900
<G>/.                     Initialize CRTC module
<5>
AIM 65 PL/65 V1.0
```

3.2 THE DISPLAY PROGRAM

The Display Program accepts data passed to it in the accumulator and interprets each byte as a command, a command parameter or a data character to be displayed.

The commands consist of 32 control commands ($00 to $1F), two escape commands ($1B $3D $YY $XX and $1B $47), and the delete character command ($7F). The individual commands are defined in Section 3.3.

All other character codes are treated as data characters (described in Section 3.4) and are stored in the refresh RAM. In the normal mode, set upon initialization or by the Enter Normal Characters command (CTRL]), codes $20-$7E are stored directly into the refresh RAM as received. In the semi-graphics mode, set by the Enter Semi-Graphics Characters Command (ESC G), bit 7 is set to "1" upon receiving data characters $20-$7F before storing them in the refresh RAM as $A0-$FF.

When using the Display Program, the only way to send $00-$1F and $80-$9F as data characters (See Section 3.4) rather than commands is to precede each byte with a CTRL P (Pass through Next Characters), otherwise these bytes will be interpreted as control commands $00-$1F (bit 7 of commands $80-$9F is masked to "0").

There are several ways to send display commands and character data from the host computer to the CRT module under operator or Program control either using the Display Program or compatible with the Display Program:

- AIM 65 Echo through DILINK (host = AIM 65)
- AIM 65 Active Output Device = User (host=AIM 65)
- Direct Call to Command/Data Processing (host=any)
- Direct Store to Refresh RAM (host=any)
- Store to Refresh RAM using Display Program (host=any)

3.2.1 AIM 65 Output Through DILINK

This will be the most common mode of operation of the CRT firmware in conjunction with the AIM 65 Monitor. The procedure discussed in Section 3.1 outputs through DILINK.

Setting variable AIM65 = 1 or >1 indicates the presence of the AIM 65 Monitor and the Initialization program (See Section 3.1) will set DILINK to point to the CRTC firmware. This sends all output normally directed to the AIM 65 display to the CRTC firmware.

When AIM65 = 1, DILINK will be set to $9973 and line feeds will be ignored.

When AIM65 >1, DILINK will be set to $9977 and line feeds will be accepted.

In both cases, the AIM 65 display will be blanked and the Monitor prompt will appear on the video screen. To return to the AIM 65 display as the output device, issue a CTRL Y ($19). Once the CRT screen has been initialized, the user may re-enter the CRTC Display program through the Warm ($9964) entry point. This will clear the screen, home the cursor and set DILINK as described above -- depending on variable AIM65 contents.

In order to send program output to the video screen, the same methods used with the AIM 65 display are employed. This means that the Monitor, BASIC, Instant Pascal, PL/65 and FORTH output routines are all compatible with the CRTC firmware, sending their output to the CRT screen instead of the AIM 65 display. The initialization for these languages are discussed in Section 3.1.2.

3.2.2 AIM 65 Active Output Device = User

Another method of outputting information is to make the CRT a user output device. A User driver routine is provided in the Program ROM to facilitate this mode of operation. This

application may be used if both the AIM 65 display and the CRT are to be used together. It also allows use of all existing I/O routines in the Monitor and the different languages. In order to send output through the User vector ($010A), three locations must be changed as follows:

Parameters	Address	Value
User vector	$10A	$70
User vector	$10B	$99
Output device code	$A413	$55

All subsequent output will be sent only through the User vector and not to the AIM 65 display, however, the AIM 65 display may be subsequently be reactivated by setting $A413 to $20. This allows sending some output to the AIM 65 display and some output to the CRT screen. Here is an example of doing this in BASIC.

```
Program                                  Comments

10  POKE 42003, 85                       Output = User ("U")
20  POKE 266, 112                        $010A = $70
30  POKE 267, 153                        $010B = $99
40  PRINT "OUTPUT SENT TO CRT"           Send Message to CRT
50  POKE 42003, 32                       Output = AIM 65 Display ("space")
60  PRINT "OUTPUT TO AIM 65"             Send Message to AIM 65
70  END
```

3.2.3 Direct Call to Command/Data Processing

A third method for outputting information to the CRT is through a direct call to the command/data processing subroutine ($9977) with the command/data byte in the accumulator. If this method is used, variable AIM65 should be set to zero to avoid the features designed specifically for keyboard input through the AIM 65 Monitor/Editor. This method can be used to keep the AIM 65 display fully active and still send output to the CRT display. A CRT output routine is given as an example.

This routine is called with the message offset (with respect to the start of the message at label M1) in the Y register and most significant bit of the last character set to one.

```
        *=$500
M1OUT   LDY #0
CRTOUT  LDA M1,Y
        PHA
        JSR $9977
        INY
        PLA
        BPL CRTOUT
        RTS

M1    .BYT 'MESSAGE #1',$A0
M2    .BYT 'MESSAGE #2',$A0
```

To send M2 to the CRT requires:

```
M2OUT   LDY #M2 - M1
        JSR CRTOUT
```

NOTE

The Display Program will always preserve the A register and X register and usually the Y register. The only instance where the Y register is not preserved is when variable AIM65 ≠0 and the number of characters since the last carriage return > CMAX. The CRT firmware must limit Y to 58 under these conditions to hold the Editor to 59 characters.

3.2.4 Direct Store Into Refresh RAM

Character data stored directly into the refresh RAM ($X000-$X7FF) will cause the corresponding characters from the character ROM to the displayed in the format compatible with the initialized CRT screen format. When used with the Display Program, the refresh RAM is formulated as row/column.

For example, if the 25 rows x 80 columns format is selected, the following addresses correspond to the initial rows and columns (i.e. before any scrolling):

Address	Row	Column
$9000-$904F	1	1-80
$9050 $909F	2	1-80
.	.	
.	.	
.	.	
$9780-$97CF	25	1-80

When using the Display Program, the CRT screen should be initialized using the Initialization Program (see Section 3.1) before storing the character data into the refresh RAM.

You can then store a character data byte directly into the refresh RAM. For example,

```
LDS #$41
STA $9000
```

will cause "A" to be displayed in row 1 column 1. This example can easily be expanded to store a string of data characters. Before doing that, however, the display/refresh status should be considered.

If the CPU tries to access the refresh RAM at the same time the CRTC device on the CRTC module does during display enable, the addressing conflict will cause an improper display, i.e., "snow", to appear.

The character data must be stored in the refresh RAM before the retrace is completed in order to prevent any access conflict from occurring.

The Display/Refresh Status (bit 7 of $X880) can be monitored to avoid storing into the refresh RAM while the CRTC is accessing it during display enable. If the refresh RAM is accessed by the CPU only during retrace, i.e, when bit 7 of $X880 = 1, the "snow" will not appear.

The following assembly language code illustrates how an "A" can be stored early during the retrace:

```
START  LDA  #$41          Get character
DLY1   LDX  $9880
       BMI  DLY1
DLY2   LDX  $9880
       BPL  DLY2
DLY3   LDX  $9880
       BMI  DLY3          Retrace just started?
       STA  $9000         Yes, store characters
```

This technique can be expanded to the general case as shown for the WRITE ($9CA0) and SDELAY ($9CAC) subroutines in the Program ROM (see Appendix E).

3.2.5 Store Into Refresh RAM Using the Display Program

The easiest way to store directly into the refresh RAM is to call the WRITE subroutine ($9CA0) in the Program ROM. This subroutine (described in Appendix D) will store the character in the accumulator into the address contained in $DD, $DE. This subroutine calls SDELAY to check the Display/Refresh status before storing in the refresh RAM.

The following segment illustrates storing of "A" into row 1 column 1 using the WRITE subroutines:

```
LDA #00          #<address
STA $DD
LDA #$90         #>address
STA $DE
LDA #$41         "A"
JSR $9CA0        Call WRITE subroutine
```

3.3 Display Program Commands

3.3.1 Control Command Definitions

The 32 control commands are summarized in Table 3-1 and defined as follows:

CTRL A - Clear Line

Clears the line which the cursor is on and places the cursor at column one on that line.

CTRL B - Clear to End of Line

Clears from (and including) the cursor position to end of line and leaves the cursor position unchanged.

CTRL C - Clear Screen

Clears the whole screen and places the cursor in the "home" position (position one, line one).

CTRL D - Clear to End of Screen

Clears from (and including) the cursor position to the end of screen and leaves the cursor position unchanged.

CTRL E - Clear Screen

Same as CTRL A.

CTRL F - Clear to End of Screen

Same as CTRL B.

Table 3-1. Display Control Commands

ASCII Code	Control Character	Description
00	CTRL @*	
01	CTRL A	Clear Line
02	CTRL B	Clear to End of Line
03	CTRL C	Clear Screen
04	CTRL D	Clear to End of Screen
05	CTRL E	Clear Screen
06	CTRL F	Clear to End of Screen
07	CTRL G*	
08	CTRL H	Backspace (Left Arrow)
09	CTRL I	Horizontal Tab (Right Arrow)
0A	CTRL J	Line Feed (Down Arrow)
0B	CTRL K	Vertical Tab (Up Arrow)
0C	CTRL L*	
0D	CTRL M	Carriage Return (Home On Line)
0E	CTRL N	Home On Screen
0F	CTRL O	Home On Screen
10	CTRL P	Pass Through Next Character
11	CTRL Q*	
12	CTRL R*	
13	CTRL S	Toggle Insert Character Mode
14	CTRL T	Delete One Character
15	CTRL U	Insert One Line
16	CTRL V	Delete One Line
17	CTRL W	Display Cursor
18	CTRL X	Blank Cursor
19	CTRL Y	Relink AIM 65 Display
1A	CTRL Z*	
1B	CTRL [	Escape Command (ESC)
1C	CTRL \	Display Blinking Cursor
1D	CTRL]	Enter Normal Characters
1E	CTRL ^	Perform Self Test
1F	CTRL __	Reverse Video (Full Screen)

NOTE
*These characters have no effect.

CTRL H - Backspace (Left Arrow)

Moves the cursor left one position. If the cursor is in first position on a line, it will be moved to last position on the line above. If the cursor is in the first position on the first line, it will be moved to the last position on the last line.

CTRL I - Horizontal Tab (Right Arrow)

Moves the cursor right one position. If the cursor is in last position on a line, it will be moved to first position on the following line. If cursor is in the last position on the last line, it will be moved to the "home" position.

CTRL J - Line Feed (Down Arrow)

Moves the cursor down one line. If the cursor is positioned on the last line, it will be moved to same position in the top line.

CTRL K - Vertical Tab (Up Arrow)

Moves the cursor up one line. If the cursor is positioned on the top line, it will be moved to same position on the bottom line.

CTRL M - Carriage Return (Home on Line)

Returns the cursor to position one on the next line. If cursor is on last line, the screen will scroll up and the cursor will be displayed in position one on the last line (now a blank line).

CTRL N - Home on Screen

Moves the cursor to "home" position in the upper left hand corner of the screen.

CTRL O - Home on Screen

Same as CTRL N.

CTRL P - Pass Through Next Character

Tells the software to treat the following character as data, regardless of which ASCII code is received. This enables characters represented the command codes $00-$1F to be displayed.

CTRL S - Toggle Insert Character Mode On/Off

Causes the Insert mode to be toggled. Characters are inserted on a line basis. The inserted character is written at the cursor position. All characters to the right are moved right one position and the last character is lost if it was at the last position on the line.

Note that this is different from the replace character mode (the normal mode of operation), where the character under the cursor is replaced by the received character then the cursor advanced one position.

CTRL T - Delete One Character

Deletes the character under the cursor and moves all characters on the line left one position. A blank is written in the right-most position on the line.

CRTL U - Insert One Line

Moves the line containing the cursor and all lines below it down one line. The last line on the screen is not displayed and a blank line is written where the cursor was. The cursor is positioned in the position one on this blank line.

CRTL V - Delete One Line

Moves every line below the cursor up one line and displays a blank line at the bottom. The line where the cursor was is written over by the line below it.

CTRL W - Display Cursor

Causes a non-blinking cursor to be displayed on the screen.

CTRL X - Blank Cursor

Blanks the cursor.

CTRL Y - Relink AIM 65 Display

Causes the AIM 65 display to become active and the CRT inactive, by changing DILINK ($A406 and $A407) to point to the AIM 65 display ($EF05). To return to the CRT, a Warm Start may be executed at $9964.

CTRL [- Escape Character (ESC)

Precedes the two escape sequences for "enter graphics" and "relative cursor positioning" (see Section 3.3.2).

CTRL \ - Display Blinking Cursor

Causes the cursor to blink at 1/32 the screen refresh rate.

CTRL] - Enter Normal Characters

Exits semi-graphics mode to display the lower 128 characters of the 256 character set.

CTRL ^ - Perform Self Test

Displays all 256 characters in a 16 x 16 character format. An increasing MSB occurs from top to bottom and an increasing LSB occurs from left to right. The top left position is ASCII 00.

CTRL _ - Reverse Video (Full Screen)

Toggles the video output from the CRT module between dark characters on a light background (reverse), and light characters on a dark background (normal).

3.3.2 Escape Command Definitions

The two escape commands are summarized in Table 3-2 and defined as:

ESC = Y X - Absolute Cursor Positioning

Moves the cursor to row YY ($20=row one) and column XX ($20=position one). The four character sequence sent in order to the CRT firmware will position the cursor. If either the row or column exceeds the screen boundary, it will be set to the maximum value for that particular screen size.

ESC G - Enter Semi-Graphics Characters

Allows the upper 128 characters of the 256 character set to be displayed. When a 7-bit code is sent to the firmware, the MSB will be set by the CRTC module firmware before the character is written into the refresh RAM. The exceptions are the control characters ($00 - $1F), $7F and $8D which must be preceded by a CTRL P ($10) in order to be displayed.

Table 3-2. Display Escape Commands

Hex Code	Character Sequence	Function
1B 3D YY XX	ESC = Y X	Absolute cursor positioning. Move cursor to row Y-$20 and column X-$20.
1B 47	ESC G	Enter Semi-Graphics mode.

3.3.3 Backspace Command Definition

The DEL character ($7F) is interpreted as a Backspace/Delete because this is the ASCII code used by the AIM 65 microcomputer. The $7F causes the cursor to move back one space and write a blank at that position.

3.4 DATA CHARACTER ROM FORMATS

The 4K-byte R2332 character ROM (ID = R32L5) contains the dot matrix patterns for the 256 standard, special and semi-graphic characters. Figure 3-1 shows the characters corresponding to 8-bit binary codes ($00-$FF). The hexadecimal byte values corresponding to the individual characters are comprised of the most significant digit (bits 4-7), shown along the left side of the table, and the least significant digit (bits 0-3), shown along the top of the table.

The character set may be categorized into three portions:

- 32 control characters ($00-$1F)
- 96 standard ASCII characters ($20-$7F)
- 128 special and semi-graphic characters ($80-$FF)

The character data may be sent to the CRTC module using one of the Display Program capabilities or by storing the data directly into the refresh RAM (see Section 3.2).

NOTE

An easy method to send data character bytes to CRTC module when using the Display Program, without switching between normal and semi-graphics character mode operation and preceding control characters with CTRL P, is to precede all data characters with CTRL P, thus allowing all 256 data bytes encoded $00-$FF to be passed directly into the refresh RAM.

Figure 3-1. CRT Controller Module Normal/Semi-Graphic Characters

3.5 AIM 65 ASSEMBLER REFORMATTER

The third main section of the CRT firmware is the AIM 65 Assembler Reformatter. This program reformats the 20 column output from the AIM 65 Assembler into a 60 column format. It also displays the hexadecimal address for every instruction on each line rather than every 16 lines (See Figure 3-2).

3.5.1 Using the Reformatter

a. User Driver Output

To use the reformatter, change the User output driver vector at $010A to point to $9E6B, then direct the assembler listing output to "U". For example:

```
<M>=010A WW XX YY ZZ
</> 010A 6B 9E YY ZZ
```

A typical assembly command sequence may then look like this:

```
<N>
ASSEMBLER
FROM=0F00 TO=0FFF
IN=M
LIST?Y
LIST-OUT=U
OBJ?Y
OBJ-OUT=X
```

b. Echoing Reformatter Output

The reformatted output which is normally output to the CRT or TV screen may be directed to an additional output device. Do this by changing the "ECHO" vector at $035C-$035D (LSB, MSB) to point to the starting address of a user routine which will send the output to a printer or other output device. The reformatter echoes its output by passing it in the 'A' register. None of the registers (A,X,Y) must be preserved in the user routine. An "RTS" instruction will return to the Reformatter to output and echo the next character. The user driver shown in Figure 3-2 directs the output to a Centronics type parallel interface through the AIM 65 Application Connector.

```
0000        PASS 1
0FCB        PASS 2
0000         AIM 65 PARALLEL PRINTER DRIVER
0000
0000
;
; THIS OUTPUT DRIVER IS FOR A PARALLEL PRINTER WITH
; DATA STROBE NOT AND DATA ACKNOWLEDGE NOT HANDSHAKING
; <<  CENTRONICS TYPE PRINTER INTERFACE   >>
; DATA  IS  SET UP ON USER VIA PORT A (BITS 0-6)
; DATA STROBE NOT IS ON PORT B (BIT 0 ONLY IS USED)
; DATA ACKNOWLEDGE NOT IS SENSED ON LINE CA2
;
;
; AIM 65 USER VIA REGISTER DEFINITIONS
0000                *=$A000
A000 PORTB          *=*+1               ;USER VIA ADDRESS
A001 PORTA          *=*+1
A002 DDRB           *=*+1
A003 DDRA           *=*+9
A00C PCR            *=*+1
A00D IFR            =*
;
; PROGRAM CONSTANTS
A00D COMIN=$E1A1
A00D
A00D IDRB=$01                           ;RETURN TO THE AIM 65 MONITOR
A00D
A00D IDRA=$FF                           ;STROBE LINE (BIT 0) IS OUTPUT
A00D
A00D IPCR=$04                           ;DATA LINES (PORT A) ARE OUTPUT
A00D
A00D STROB0=$FE                         ;POSITIVE EDGE ON ACKNOWLEDGE
A00D
A00D STROB1=$01                         ;FORCE STROBE LOW
A00D
                                        ;FORCE STROBE HIGH
; SET UP U OUT VECTOR TO POINT TO ASSEMBLER REFORMATTER
A00D                *=$10A
010A UOUT   6B9E    WOR $9E6B
; SET UP F1 KEY TO OPEN THE PRINTER AND ISSUE A CR-LF
010C        4C800F JMP OPEN
; SET UP ECHO VECTOR TO POINT TO THE PRINTER DRIVER
010F                *=$35C
035C ECHO   9A0F    WOR OUTPUT
```

Figure 3-2. Typical AIM 65 Assembler Reformatter Output


```
;
; OUTPUT HANDLER PROGRAM
035E
0F80                *=$F80
; THE USER VIA MUST BE OPENED OR SET UP BEFORE
; CALLING THE PARALLEL OUTPUT DRIVER ( F1 KEY )
0F80 OPEN   A90D    LDA #$0D            ; ISSUE A CR AT FIRST
0F82        8D01A0 STA PORTA
0F85        A901    LDA #IDRB
0F87        8D02A0 STA DDRB
0F8A        A9FF    LDA #IDRA
0F8C        8D03A0 STA DDRA
0F8F        A904    LDA #IPCR
0F91        8D0CA0 STA PCR
0F94        20A30F JSR LFEED            ; SEND OUT CHAR
0F97        4CA1E1 JMP COMIN            ;GO BACK TO MONITOR ==>
;
; OUTPUT DATA ROUTINE
0F9A OUTPUT 48      PHA
0F9B        20B50F JSR WAIT             ; WAIT UNTIL PRINTER IS READY
0F9E        68      PLA
0F9F        C90D    CMP #$0D
0FA1        D00B    BNE CHROUT          ; NOT A CR THEN ==>
0FA3 LFEED  8D01A0 STA PORTA            ; ALWAYS FOLLOW CR WITH A LF
0FA6        20BD0F JSR STROBE           ; SEND OUT THE CR
0FA9        20B50F JSR WAIT             ; WAIT UNTIL PRINTER IS READY
0FAC        A90A    LDA #$0A            ; NOW ISSUE THE LINEFEED
0FAE CHROUT 8D01A0 STA PORTA
0FB1        20BD0F JSR STROBE           ; SEND OUT THE CHARACTER
0FB4        60      RTS
; WAIT UNTIL AN ACKNOWLEDGE IS RECEIVED FROM THE PRINTER
0FB5 WAIT   AD0DA0 LDA IFR              ; ACKNOWLEDGE IS ON CA2
0FB8        4A      LSR A
0FB9        4A      LSR A               ; MOVE CA2 INTO CARRY FLAG
0FBA        90F9    BCC WAIT            ; NOT READY? ==>
0FBC        60      RTS
; HANDSHAKE OFF THE CHARACTER WITHOUT ALTERING OTHER BITS
0FBD STROBE A9FE    LDA #STROB0
0FBF        2D00A0 AND PORTB
0FC2        8D00A0 STA PORTB            ; TOGGLE STROBE LOW
0FC5        0901    ORA #STROB1
0FC7        8D00A0 STA PORTB            ; TOGGLE STROBE HIGH
0FCA        60      RTS
0FCB         END
```

Figure 3-2. Typical AIM 65 Assembler Reformatter Output (Con't)

c. Reformatter Entry Points

1. Single Entry Point

The Reformatter has one entry point at $9E6B, but the code executed depends upon the state of the 6502 CPU carry flag. If carry is clear, then the reformatter is initialized only; if the carry is set, then the actual reformatter is called. This is taken care of automatically when the Reformatter is called with the OUT=U method.

2. Separate Entry Points

If individual calls are made to the Reformatter, the following entry should be used.

Routine	Address
Reformatter initialization	$9E6D
Reformatter output (Character passed on stack)	$9E75

3.5.2 Reformatter Output Format

The format of the output from the AIM 65 Assembler Reformatter is described in Table 3-3. Note that any comments on EQUATE statements will be output on the line following the EQUATE statement in column 40. Also, when echoing the output to a printer, the "ERRORS = XXXX" line may not normally be printed since no carriage return is output after this line. Sending a carriage return to the printer after this last line will cause it to be printed.

Table 3-3. AIM 65 Assembler Reformatter Output Format

Start Column	End Column	Field Contents
1	4	Current program counter address
1	60	Stand-alone comment line
6	11	Label
13	18	Object code
20	22	Operation code (instruction mnemonic)
24	45	Operand
40*	60	Comment appended to an instruction

NOTE

* When the operand field exceeds column 38, any comments on that line will be spaced over two spaces from the end of the operand field.

3.5.3 Assembler Reformatter Variables

The variables used by the Reformatter are from $F0 to $F7, overlapping with the Editor "Find String" buffer. None of the variables are user-alterable and all may be destroyed between two assemblies.

SECTION 4

FUNCTIONAL AND INTERFACE DESCRIPTION

4.1 FUNCTIONAL DESCRIPTION

The block diagram in Figure 4-1 identifies the CRTC module functions and interface signals. A memory map reflecting the CRTC module firmware, variables and I/O is shown in Figure 4-2. Table 4-1 is a detail I/O memory map.

The Data Transceivers invert and transfer 8-bits of parallel data between the CRTC module and the RM 65 Bus, based on control signals from the Base Address Decoder and the Control Buffers. The read/write control line determines the direction, while the bus active signal enables the Data Transceivers.

The Address Buffers (Z22 and Z23) invert and transfer the 16-bit parallel address lines from the RM 65 Bus to the Base Address Decoders, the R2316 Program ROM, the CRT Controller (CRTC) device, and to the Refresh RAM device.

The Control Buffers (Z22) invert and transfer the phase 2 clock and read/write control signals from the RM 65 Bus onto the module.

The Bank Select Control circuit detects when the module's assigned memory bank is addressed, by comparing the bank address signal from the RM 65 Bus to the Bank Select and Bank Select Enable switches. The Bank Select Eanble switch allows the board to reside in common memory (both Bank Ø and Bank 1) or only in the Bank set by the Bank Select switch (either Bank Ø or Bank 1).

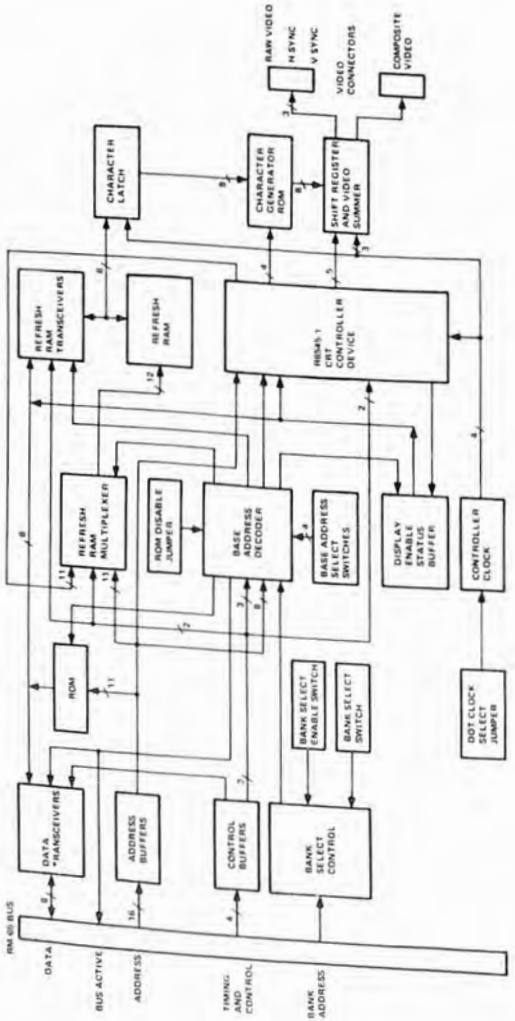


Figure 4-1. CRT Controller Module Block Diagram

Hex Addr.	Function	Hex Addr.	Function
$0000	Page 0 RAM: $0DB - $0DE(1) $0F0 - $0F7(2) Page 3 RAM: $347 - $3B5	$8000	User available
$1000	User available	$9000	RAM: $9000 - $97FF(3) I/O: $9800 - $98FF ROM: $9900 - $9FFF
$2000		$A000	User available
$3000		$B000	
$4000		$C000	
$5000		$D000	
$6000		$E000	
$7000		$F000	

NOTES

1. These locations are temporary variables used by the CRTC module Program ROM.
2. These locations are used by the Assembler Reformatter only.
3. RAM on the CRTC module is CRT refresh memory only.
4. Base Address Select switches (S1-1 to S1-4) must be set to $9000 if the CRTC module Program ROM is used.

Figure 4-2. CRT Controller Module Memory Map

Table 4-1. CRT Controller Module I/O Addresses

(1) Addr.	CRTC Reg.	Register Name	Register Units	R/$\overline{W}$
X800	-	CRTC Status Register	-	H
X801	-	CRTC Address Register (3)	-	L
X801	R0	Horiz. Total Chars	No. of Chars/Row	L
X801	R1	Horiz. Displayed Chars	No. of Chars/Row	L
X801	R2	Horiz. Sync Position	Character Position	L
X801	R3	VSYNC, HSYNC Widths	No. of Scan Lines, Char	L
X801	R4	Vertical Total Rows	No. of Char Rows	L
X801	R5	Vertical Adjust Lines	No. of Scan Lines	L
X801	R6	Vertical Displayed Rows	No. of Char Rows	L
X801	R7	Vertical Sync Position	No. of Char Rows	L
X801	R8	Mode Control	-	L
X801	R9	Scan Line	No. of Scan Lines	L
X801	R10	Cursor Start Line	Scan Line No.	L
X801	R11	Cursor End Line	Scan Line No.	L
X801	R12	Display Start Addr (H)	-	L
X801	R13	Display Start Addr (L)	-	L/H
X801	R14	Cursor Position Addr (H)	-	L/H
X801	R15	Cursor Position Addr (L)	-	H
X801	R16	Light Pen Register (H)(4)	-	H
X801	R17	Light Pen Register (L)(4)	-	H
X880	-	Display/Refresh Status Register Bit 7 = 0 Refresh Bit 7 = 1 Display		H

NOTES

1. X corresponds to the assigned Base Address of any hex value (see Section 2.2.1).
2. X802 through X87F redundantly maps X800 and X801, depending on bit 0 value.
3. X881 through X8FF redundantly maps X880.
4. Not used on the RM 65 CRTC module.

The Base Address Decoder (Z21), with the Base Address Select switches (S1-1 through S1-4), the Bank Select Control circuit (Z12-3), the ROM Disable Jumper (E1) and the read/write and phase 2 clock signals, generate device selects for the onboard ROM, RAM, and I/O (CRTC device and Display Enable Status Buffer). The Base Address Select switches allow the module to be selected to any 4K block. Within the selected 4K block, the RAM is assigned to the lower 8 pages (2K bytes), and the I/O to the first 256 bytes of the upper 2K bytes. When the ROM is disabled, only the RAM and I/O can be selected and the module is assigned 9 pages (2304 bytes) in memory. When the ROM is enabled, the module is assigned the full 4K bytes, with seven pages for ROM, in addition to the RAM and I/O.

The Controller Clock uses a 12 MHz crystal-controlled oscillator to derive a 6 MHz or 12 MHz reference for the shift register dot clock depending on the Dot Clock Select jumper (E2) position. With the 6 MHz clock, up to 40 characters per line can be displayed on any monitor or standard television using an RF modulator. Up to 80 characters per line can be achieved with the 12 MHz clock and a high bandwidth monitor. The dot clock is divided by seven (Z4) to provide a Character Clock for the CRTC device and a load character signal for the Shift Register (Z7).

The Refresh RAM (Z10) provides 2K bytes of display memory, for screen densities of up to 25 lines with as many as 80 characters each. The RAM is directly mapped into the RM 65 memory map, so the display can be updated by a block memory move or under DMA control. The Refresh RAM Multiplexers (Z11, Z16 and Z20), and RAM Transceivers (Z24) allow the RM 65 bus and the CRTC device to both access the Refresh RAM, with the RM 65 bus having priority when any conflict occurs.

The Display Enable Status Buffer (Z3-3) allows the RM 65 Bus to monitor the active display times, so that display memory transfers can be made with no visible distortion.

The Character Generator ROM (Z8) holds the fonts for the character set. These fonts are stored as 256 characters, each with 10 seven-bit rows (see Appendix E). The four CRTC device row address lines and the eight Character Latch bits, which hold the character being refreshed, create an address for the character generator ROM. The output data of the ROM, which is seven parallel bits, represents the display pattern. The Shift Register takes this data and forms the serial video data. The Video Summer combines and buffers the serial video data with CRTC device timing signals to form a composite video output at connector P2 and separate video, horizontal sync, and vertical sync outputs at connector P3.

4.2 INTERFACE DESCRIPTION

Table 4-2 lists the pin connections for the I/O signals transferred between the CRT Controller module connector P1 and the RM 65 Bus.

Table 4-3 lists the pin connections for the Separate Video and Sync connector P3.

Table 4-4 describes the RM 65 interface signals on connector P1.

Table 4-2. Connector P1 (RM 65 Bus) Pin Assignments

Pin	Signal Mnemonic	Signal Name	Input/ Output
Wa		Not Connected (See Note)	
Wc		Not Connected (See Note)	
Xa	+5V	+5 Vdc (See Note)	
Xc	+5V	+5 Vdc (See Note)	
1a	GND	Ground	
1c	+5V	+5 Vdc	
2a	BADR/	Buffered Bank Address	I
2c	BA15/	Buffered Address Bit 15	I
3a	GND	Ground	
3c	BA14/	Buffered Address Bit 14	I
4a	BA13/	Buffered Address Bit 13	I
4c	BA12/	Buffered Address Bit 12	I
5a	BA11/	Buffered Address Bit 11	I
5c	GND	Ground	
6a	BA10/	Buffered Address Bit 10	I
6c	BA9/	Buffered Address Bit 9	I
7a	BA8/	Buffered Address Bit 8	I
7c	BA7/	Buffered Address Bit 7	I
8a	GND	Ground	
8c	BA6/	Buffered Address Bit 6	I
9a	BA5/	Buffered Address Bit 5	I
9c	BA4/	Buffered Address Bit 4	I
10a	BA3/	Buffered Address Bit 3	I
10c	GND	Ground	
11a	BA2/	Buffered Address Bit 2	I
11c	BA1/	Buffered Address Bit 1	I
12a	BA0/	Buffered Address Bit 0	I
12c		Not Used	
13a	GND	Ground	
13c		Not Used	
14a		Not Used	
14c		Not Used	
15a		Not Used	

Table 4-2. Connector P1 (RM 65 Bus) Pin Assignments (Continued)

Pin	Signal Mnemonic	Signal Name	Input/ Output
15c	GND	Ground	
16a		Not Used	
16c		Not Used	
17a		Not Used	
17c		Not Used	
18a	GND	Ground	
18c		Not Used	
19a		Not Used	
19c		Not Used	
20a		Not Used	
20c	GND	Ground	
21a	BR/$\overline{W}$/	Buffered Read/Write "Not"	I
21c		Not Used	
22a		Not Used	
22c	BR/$\overline{W}$	Buffered Read/Write	I
23a	GND	Ground	
23c	BACT/	Buffered Bus Active	O
24a		Not Used	
24c		Not Used	
25a	BØ2/	Buffered Phase 2 "Not" Clock	I
25c	GND	Ground	
26a	BØ2	Buffered Phase 2 Clock	I
26c		Not Used	
27a	BD7/	Buffered Data Bit 7	I/O
27c	BD6/	Buffered Data Bit 6	I/O
28a	GND	Ground	
28c	BD5/	Buffered Data Bit 5	I/O
29a	BD4/	Buffered Data Bit 4	I/O
29c	BD3/	Buffered Data Bit 3	I/O
30a	BD2/	Buffered Data Bit 2	I/O
30c	GND	Ground	
31a	BD1/	Buffered Data Bit 1	I/O

Table 4-2. Connector P1 (RM 65 Bus) Pin Assignments (Continued)

Pin	Signal Mnemonic	Signal Name	Input/ Output
31c	BD0/	Buffered Data Bit 0	I/O
32a	+5V	+5 Vdc	
32c	GND	Ground	
Ya	+5V	+5 Vdc (See Note)	
Yc	+5V	+5 Vdc (See Note)	
Za		Not Connected (See Note)	
Zc		Not Connected (See Note)	

NOTE

1. Pins Wa, Wc, Xa, Xc, Ya, Yc, Za and Zc are available on Edge Connector version only.
2. "/" suffix denotes signal active at negative or low voltage level.

Table 4-3. Connector P3 (Separate Video) Pin Assignments

Pin	Signal Mnemonic	Signal Name	I/O	Output Drive
1	VIDEO	Discrete Video	O	Tri-state
2	GND	Ground		
3	VSYNC	Vertical Synchronization	O	Tri-state
4	GND	Ground		
5	HSYNC	Horizontal Synchronization	O	Tri-state
6	GND	Ground		

NOTE

Tri-state output is defined as TTL level:

I_{OH} = 2.6 mA

I_{OL} = 24 mA

Table 4-4. Connector P1 (RM 65 Bus) Signal Descriptions

Signal Mnemonic	Signal Name and Signal Description
+5V	+5Vdc +5 Vdc supplied to the module from the RM 65 Bus.
GND	Ground System ground.
BA0/-BA15/	Buffered Address Bits 0-15 Sixteen address lines transfer a 16-bit address from the RM 65 Bus to the module.
BADR/	Buffered Bank Address A high BADR/ addresses the lower 65K (Bank 0) memory bank; a low BADR/ addresses the upper 65K (Bank 1) memory bank.
BD0/-BD7/	Buffered Data Bits 0-7 Eight bidirectional data lines for transferring 8-bit data bytes between the Data Transceivers in the module and the RM 65 Bus.
BACT/	Buffered Bus Active A low BACT/ indicates that the module has been addressed and the Data Transceivers are enabled in either the receive (write operation) or transmit (read operation) direction. A high BACT/ indicates that the Data Transceivers are disabled.

NOTE

All signals interfaced to and from the RM 65 Bus are driven at TTL voltage levels.

Table 4-4. Connector P1 (RM 65 Bus) Signal Descriptions (Continued)

Signal Mnemonic	Signal Name and Signal Description
BR/$\bar{W}$	Buffered Read/Write This signal indicates the direction of data transfer on the bus. A high BR/$\bar{W}$ (read operation) to an addressed module enables the Data Transceivers to transfer the 8-bit data from the module onto the Bus. A low BR/$\bar{W}$ (write operation) to an addressed module enables the Data Transceivers to transfer data from the Bus onto the module.
BR/$\bar{W}$/	Buffered Read/Write "NOT" This signal indicates the direction of data transfer on the bus (the logical inverse of BR/$\bar{W}$).
B02	Buffered Phase 2 Clock The B02 signal synchronizes data transfers on the RM 65 Bus. Addresses are set up during the negative portion, and data is set up during the positive portion.
B02/	Buffered Phase 2 Clock "NOT" The B02/ signal synchronizes data transfers on the RM 65 Bus (the logical inverse of B02).

NOTE

All signals interfaced to and from the RM 65 Bus are driven at TTL voltage levels.

APPENDIX A

ASCII CHARACTER SET

HEX	DEC	ASCII	HEX	DEC	ASCII	HEX	DEC	ASCII	HEX	DEC	ASCII
00	0	NUL	20	32	SP	40	64	@	60	96	`
01	1	SOH	21	33	!	41	65	A	61	97	a
02	2	STX	22	34	"	42	66	B	62	98	b
03	3	ETX	23	35	#	43	67	C	63	99	c
04	4	EOT	24	36	$	44	68	D	64	100	d
05	5	ENQ	25	37	%	45	69	E	65	101	e
06	6	ACK	26	38	&	46	70	F	66	102	f
07	7	BEL	27	39	'	47	71	G	67	103	g
08	8	BS	28	40	(	48	72	H	68	104	h
09	9	HT	29	41	)	49	73	I	69	105	i
0A	10	LF	2A	42	*	4A	74	J	6A	106	j
0B	11	VT	2B	43	+	4B	75	K	6B	107	k
0C	12	FF	2C	44	,	4C	76	L	6C	108	l
0D	13	CR	2D	45	-	4D	77	M	6D	109	m
0E	14	SO	2E	46	.	4E	78	N	6E	110	n
0F	15	SI	2F	47	/	4F	79	O	6F	111	o
10	16	DLE	30	48	0	50	80	P	70	112	p
11	17	DC1	31	49	1	51	81	Q	71	113	q
12	18	DC2	32	50	2	52	82	R	72	114	r
13	19	DC3	33	51	3	53	83	S	73	115	s
14	20	DC4	34	52	4	54	84	T	74	116	t
15	21	NAK	35	53	5	55	85	U	75	117	u
16	22	SYN	36	54	6	56	86	V	76	118	v
17	23	ETB	37	55	7	57	87	W	77	119	w
18	24	CAN	38	56	8	58	88	X	78	120	x
19	25	EM	39	57	9	59	89	Y	79	121	y
1A	26	SUB	3A	58	:	5A	90	Z	7A	122	z
1B	27	ESC	3B	59	;	5B	91	[	7B	123	{
1C	28	FS	3C	60	<	5C	92	\	7C	124	|
1D	29	GS	3D	61	=	5D	93	]	7D	125	}
1E	30	RS	3E	62	>	5E	94	↑	7E	126	~
1F	31	VS	3F	63	?	5F	95	←	7F	127	DEL

NOTE

NUL - Null
SOH - Start of Heading
STX - Start of Text
ETX - End of Text
EOT - End of Transmission
ENQ - Enquiry
ACK - Acknowledge
BEL - Bell
BS - Backspace
HT - Horizontal Tabulation
LF - Line Feed
VT - Vertical Tabulation
FF - Form Feed
CR - Carriage Return
SO - Shift Out
SI - Shift In
DLE - Data Link Escape
DC - Device Control
NAK - Negative Acknowledge
SYN - Synchronous Idle
ETB - End of Transmission Block
CAN - Cancel
EM - End of Medium
SUB - Substitute
FSC - Escape
FS - File Separator
GS - Group Separator
RS - Record Separator
US - Unit Separator
SP - Space (Blank)
DEL - Delete

LSD \ MSD		0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0	0000	NUL	DLE	SP	0	@	P	'	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
A	1010	LF	SUB	*	:	J	Z	j	z
B	1011	VT	ESC	+	;	K	[	k	{
C	1100	FF	FS	,	<	L	\	l	|
D	1101	CR	GS	-	=	M	]	m	}
E	1110	SO	RS	.	>	N	↑	n	~
F	1111	SI	VS	/	?	O	←	o	DEL

NUL — Null
SOH — Start of Heading
STX — Start of Text
ETX — End of Text
EOT — End of Transmission
ENQ — Enquiry
ACK — Acknowledge
BEL — Bell
BS — Backspace
HT — Horizontal Tabulation
LF — Line Feed
VT — Vertical Tabulation
FF — Form Feed
CR — Carriage Return
SO — Shift Out
SI — Shift In

DLE — Data Link Escape
DC — Device Control
NAK — Negative Acknowledge
SYN — Synchronous Idle
ETB — End of Transmission Block
CAN — Cancel
EM — End of Medium
SUB — Substitute
ESC — Escape
FS — File Separator
GS — Group Separator
RS — Record Separator
US — Unit Separator
SP — Space (Blank)
DEL — Delete

APPENDIX B

PREPROGRAMMED AND USER-DEFINED SCREEN FORMAT VALUES

This appendix describes the pre-programmed screen parameters and how to compute these parameters for a user defined screen format when using the CRT module Program ROM. If the Program ROM is not being used, the procedure to compute the screen parameter values for the R6545-1 CRTC device is the same as for the Program ROM, however the addresses are I/O dependent rather than those specified for the Program ROM, Figure B-1 illustrates the screen parameters.

B.1 PREPROGRAMMED FORMATS

The contents of the four screen parameter tables stored in the firmware are shown in Table B-1. These should be a useful reference in setting up a user-defined table. The name of each byte is given here and a description of each byte is given in the following section, "User Defined Screen Format".

B.2 USER-DEFINED SCREEN FORMAT

Setting variable TABLE to 4 before calling the Initialization Program indicates a user-defined screen format is to be used. The 12-byte format table must be set up in sequence in memory. The MSB of the starting address must be stored in variable XADDR ($0363) and the LSB of the starting address in variable XADDR+1 ($0364). Either of the cold start entry points may now be called. A definition of each byte with the method for determining the value is shown in Table B-2.

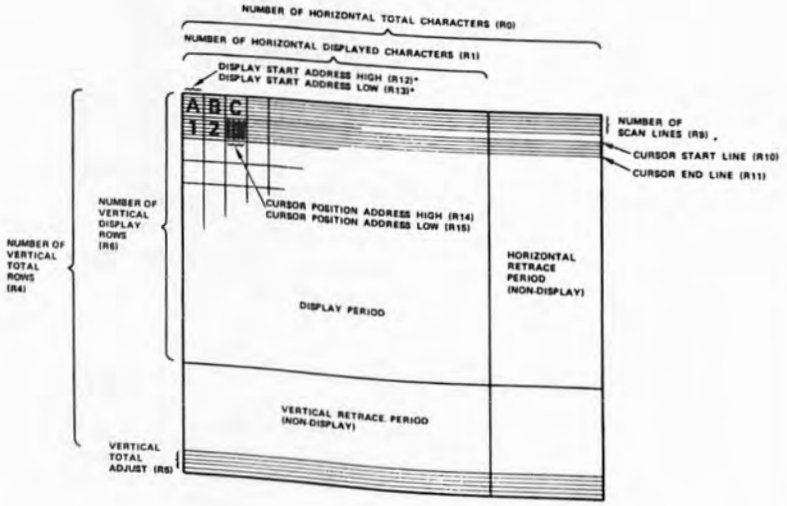


Figure B-1. Screen Parameters

Table B-1. Firmware Screen Format Tables

Byte No.	Name	Europe TABLE = 0	U.S. TABLE = 1	Europe TABLE = 2	U.S. TABLE = 3
		625 Scan (25 x 80)	525 Scan (22 x 72)	625 Scan (25 x 40)	525 Scan (16 x 40)
1	Horizontal Total	108	108	54	54
2	Horizontal Displayed	80	72	40	40
3	Horizontal Sync Position	89	85	44	45
4	Sync Widths	$59	$59	$55	$55
5	Vertical Total Rows	31	26	31	26
6	Vertical Total Adjust	2	2	2	2
7	Vertical Displayed	25	22	25	16
8	Vertical Sync Position	28	24	28	21
9	Same as byte 2	80	72	40	40
10	Same as byte 7	25	22	25	16
11	Total Characters (MSB)	$07	$06	$03	$80
12	Total Characters (LSB)	$D0	$30	$E8	$80

Table B-2. Screen Parameter Table

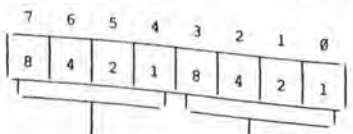
Byte No.	Parameter Definition
1	Horizontal Total Characters - The total of displayed and non-displayed characters, minus one, per horizontal line. If Jumper E2 is in position A (6 MHz dot clock), then this byte will be 54.
2	Horizontal Displayed Characters - The number of displayed characters per line.
3	Horizontal Sync Position - The character position on the horizontal line where the horizontal sync occurs. Normally for a centered screen, the same number of non-displayed characters should be on the left as on the right of the displayed characters. This is calculated by the formula: Byte 3 = INTEGER[((BYTE 1 - Hsync Pulse Width - BYTE 2)/2) + Byte 2] (Eq. 1) As the Byte 3 value increases towards the Byte 1 value, the displayed screen will move to the right and as Byte 3 decreases towards Byte 2, the displayed screen will move to the left for centering the displayed characters.
4	Horizontal and Vertical Sync Widths - The width of the horizontal and vertical sync pulses as defined below.  — HSYNC Pulse Width The width of the horizontal sync pulse (HSYNC) in the number of character clock times (CCLK). — VSYNC Pulse Width The width of the vertical sync pulse (VSYNC) in the number of scan lines. When bits 4-7 are all "Ø", VSYNC will be 16 scan lines wide. Normally, this will be $59 for the 12 MHz dot and $55 for the 6 MHz clock.

Table B-2. Screen Parameter Table (Continued)

Byte No.	Parameter Definition
5	Vertical Total Rows - A 7-bit value containing the total number of character rows, minus one, in a frame. This value may be calculated by: $\text{Byte 5} = \frac{1572}{\text{frame freq. (Hz)}}$ (Eq. 2) where frame freq. is normally 5Ø Hz or 6Ø Hz
6	Vertical Total Adjust - A 5-bit value containing the number of additional scan lines needed to complete an entire frame scan. Together, byte 4 and byte 5 determine the overall frame frequency and thus the frequency of the vertical sync pulses. Typically, this will be equal to the decimal part of the result from Eq. 2 multiplied by 1Ø and rounded to an integer number.
7	Vertical Displayed Rows - A 7-bit value defining the number of displayed character rows in each frame.
8	Vertical Sync Position - A 7-bit value defining the character row at which the vertical sync will occur. This must be greater than or equal to byte 7 and less than or equal to byte 5. It is best to have the vertical sync occur soon enough so that the beam has some non-displayed scan lines at the top of the screen to stabilize. As the byte 8 value increases, the screen moves up and as it decreases, the screen moves down.
9	This byte must be the same value as byte 2.
1Ø	This byte must be the same value as byte 7.
11	No. Characters/Screen (MSB) - This byte is the MSB of the number of characters in one screen. In an 8Ø x 25 screen, there are 8Ø x 25 = 2ØØØ characters. 2ØØØ converted to hexadecimal is $07DØ. Thus, this byte would be set to $07 (the MSB) for a 25 x 8Ø screen.
12	No. Characters/Screen (LSB) - This byte is the LSB of the number of characters in one screen. In the 8Ø x 25 example above, this would be set to $DØ, the LSB.

Using Table B-1, Table B-2 and the screen diagram Figure B-1 as references, a user-defined screen format table can be prepared. An example of this is given along with two methods for initializing the screen to these parameters.

B.3 EXAMPLE OF USER-DEFINED SCREEN FORMAT

Here is an example of the procedure and assembly language program to create a 80 column x 25 row screen for a 60 Hz screen rate on a monitor. First, determine bytes 1 to 12 for the screen parameter table.

Byte 1 = 108 (for 12 MHz dot clock)

Byte 2 = 80 (char/row)

Byte 3 = (108-9-80)/2 + 80 = 89.5 = 89 (Eq. 1)

Byte 4 = $59 (for 12 MHz dot clock)

Byte 5 = 1572/60 = 26.2 = 26

Byte 6 = (.2) x (10) = 2 (scan lines adjust)

Byte 7 = 25 (rows)

Byte 8 = 25 (vertical sync after 25th row displayed)

Byte 9 = 80 (chars/row)

Byte 10 = 25 (rows)

Byte 11 = 80 x 25 = 2000 = $07D0 MSB = $07

Byte 12 = LSB of $07D0 = $D0

Once the 12 bytes are defined, the screen may be initialized to this format through the keyboard or through a program.

From the keyboard, the procedure is as follows (all decimal numbers must be converted to hexadecimal):

(a) Enter 12 bytes of table into memory

```
<M>=0200 WW XX YY ZZ
</> 0200 6C 32 59 59
</> 0204 1A 02 19 19
</> 0208 50 19 07 D0
```


(b) Change locations $363 (LSB) and $364 (MSB) to point to the table start address, in this case, $0200.

```
<M>=0363 WW XX YY ZZ
</> 0363 00 02
```

(c) Run initialization program from entry point 1 after setting TABLE = 4

```
<M>=035A WW XX YY ZZ
</> 035A 04

<*>=9900
<G>/.
```

Here is the software to perform the same function as described above. By putting this at $B000 the CRT may be initialized to this format by typing the "5" Key.

```
TABLE  = $035A
TABLOW = $0363      ;FORMAT TABLE ADDRESS (LSB)
TABHGH = $0364      ;FORMAT TABLE ADDRESS (MSB)
INIT   = $9906      ;RETURN WITH AN RTS

START   LDA #$04
        STA TABLE
        LDA #<T2580
        STA TABLOW
        LDA #>T2580
        STA TABHGH
        JSR INIT
        JMP MAIN

T2580   .BYT $6C,$32,$59,$1A,$02,$19,$19
        .BYT $50,$19,$07,$D0

;
; USER SOFTWARE HERE

  MAIN    ...
```

APPENDIX C

PROGRAM ROM VARIABLES

The Program ROM uses RAM from $DB through $DE and $0347 through $03B5 inclusive for variables. These locations may not be overwritten (with a few exceptions) without causing improper CRTC module operation. Tables C-1 and C-2 summarize the variables, their locations and their use.

Table C-1. Program ROM Page Zero

Address (Hex)	No. Bytes	Name	Description
DB-DC	2	BSTART	Used for indirect addressing and self-test routine (LSB/MSB)
DD-DE	2	CURSOR	Used to store CURSOR address during processing. CURSOR value is read from the 6545 CRTC device upon entry to, and rewritten to the 6545 upon exit from, the Display program (LSB/MSB).
F0-F7	8	-	Assembler Reformatter Variables.

Table C-2. Program ROM Page Three Variables

Address (Hex)	No. Bytes	Name	Description
0347-0348	2	SCRMAX	Number of characters/screen
0349	1	ROWMAX	Number of rows/screen
034A	1	COLMAX	Number of columns/screen
034B-034C	2	END	End address +1 of current screen within refresh RAM.
034D	1	DLE	"Pass through next character" flag 0 = Don't pass through 1 = Pass through
034E	1	INCH	"Insert character mode" flag 0 = Not insert mode FF = Insert mode
034F	1	EFLG	"Escape Sequence" flag 0 = Not escape sequence 1 = ESC received 2 = ESC = received 3 = ESC = X received
0350	1	GRAPH	"Graphic Mode" flag 0 = Not graphics mode FF = Graphics mode
0351	1	INVERS	"Inverse Video" flag 0 = Not inverse video FF = Inverse video
0352-0353	2	DSTARL DSTARH	Display start address - the address in refresh RAM where this screen starts.
0354	1	YSAV	Saves Y position for ESC = Y,X
0355	1	XSAV	Temporary X-register save.
0356	1	ASAV	Location to save command/data passed to Display program.

Table C-2. Program ROM Page Three Variables (Continued)

Address (Hex)	No. Bytes	Name	Description
0357	1	AIM65	Flag indicating whether special AIM 65 processing should be skipped. 0 = Skip AIM 65 processing 1 = Perform AIM 65 processing but ignore line feeds ($0A) >1 = Perform AIM 65 processing with line feeds
0358	1	ROW	Current row no., where cursor is positioned. Rows are numbered 0,1,2...
0359	1	COL	Current column no., where cursor is positioned. Columns are numbered 0,1,2...
035A	1	TABLE	Parameter specifying screen size table to be used. 0 = 25 x 80 (625 scan lines) 1 = 22 x 72 (525 scan lines) 2 = 25 x 40 (625 scan lines) 3 = 16 x 40 (525 scan lines) 4 = user defined >4 = defaults to 22 x 72 table
035B	1	CURP2	Character counter taking place of AIM 65 'CURP02' ($A415),
035C-035D	2	ECHO	Address (LSB, MSB) which user must change to have Assembler Reformatter echo its output to a user program.
035E	1	CMAX	Maximum number of characters/line that can be output - display will freeze up if this is exceeded.
035F	1	PRFLG	Counter to back up cursor to overwrite printer "ON" and "OFF" messages.
0360	1	XXX	Save X on entry and restore on exit.

Table C-2. Program ROM Page Three Variables (Continued)

Address (Hex)	No. Bytes	Name	Description
0361	1	YYY	Save Y on entry and restore on exit. Y reset to 58 when CMAX = 59, YYY = 59, and AIM65 ≠0 to limit EDITOR buffer to 59 characters.
0362	1	XX2	Another temporary X save.
0363-0364	2	XADDR	LSB, MSB pointing to start address of user-defined screen-format table. Applies when Initialization program called with TABLE = 4.
0365	1	COL2	Keeps track of the number of characters since last carriage return.
0366-03B5	80	DBUFF	80 character buffer used for INSERT/DELETE LINE and INSERT/DELETE CHAR commands. Locations $0366 through $03E5 may be overwritten between calls to the Display program with no effect. These locations are not used unless the Insert/Delete Line or Insert/Delete character commands used.

a. **TABLE**

TABLE ($035A) is the parameter which determines the video format upon initialization. The five possibilities for this parameter are listed below. This should be set to the desired value before the Initialization program is called (see Table C-3).

Table C-3. Variable TABLE Options

Variable TABLE	Meaning
0	Initialize to 25 x 80 screen - 625 horizontal scan lines (Europe) (CRT)
1	Initialize to 22 x 72 screen - 525 horizontal scan lines (U.S.) (CRT)
2	Initialize to 25 x 40 screen - 625 horizontal scan lines (Europe) (CRT or TV)
3	Initialize to 16 x 40 screen - 525 horizontal scan lines (U.S.) (CRT or TV)
4	Initialize to user-defined screen format. Twelve-byte initialization table is pointed to by location $0363 (LSB) and location $0364 (MSB) (U.S. or Europe) (CRT or TV)
Greater than 4	Default to the 22 x 72 screen - 525 horizontal scan lines (U.S.) (CRT)

NOTE

1. The E2 jumper must be set correctly before the Initialization Program is called.
2. If variable TABLE is Ø,1 or >4, the jumper must be in position B for the 12 MHz dot clock.
3. If TABLE is 2 or 3, then the jumper must be set to position A (6 MHz dot clock).
4. If TABLE is 4, then the user must determine whether the 12 MHz (position B) or 6 MHz (position A) dot clock is required. Typically, the 12 MHz dot clock cannot be used with a TV set without modifying it.

b. AIM 65

AIM65 ($Ø357) tells the CRTC firmware if it should skip over the AIM 65 dependent code. It also indicates whether line feeds are to be ignored, as required when both Carriage Return ($ØD) and Line Feed ($ØA) are sent to indicate the end of a line (see Table C-4).

Table C-4. Variable AIM 65 Options

Variable AIM65	Function/Use
Ø	Skip all AIM 65-related processing 1) When using with RM 65 SBC 2) When using with AIM 65 without Monitor 3) Other user-defined instances
1	Ignore line feeds ($ØA) 1) When PL/65 is used to avoid double spacing caused by CR/LF sent from PL/65 2) When using User output on microcomputer AIM 65 microcomputer to direct characters to the CRT program.
>1	Perform AIM 65-related processing and acknowledge line feed ($ØA) 1) When using with Editor 2) When using with Assembler 3) When using with BASIC 4) When using with FORTH 5) When using with Instant Pascal

The AIM 65-related processing referred to above consists of three main functions which are performed by the CRTC firmware when variable AIM65 ≠ Ø.

(1) Set AIM65 ($Ø357) and DILINK $A406, $S407) Variables

	AIM65	DILINK
(ignore line feeds)	1	$73, $99
(acknowledge line feeds)	>1	$77, $99

(2) Clear AIM 65 Display

The AIM 65 display is blanked by calling $EFØ5 with the accumulator = $ØD. Then the PA PORT on the display's PIA is set to $ØØ every time a CR ($ØD) is received by the CRTC firmware. This keeps the AIM 65 display blank during CRT operation.

(3) Maintain AIM 65 Display Buffer

The AIM 65 display buffer ($A438 - $A473) and display pointers ($A415, $A416) are manipulated in order to recover the delete key. Limiting the Editor line length to 59 characters is necessary to do this.

When AIM65 = Ø this processing is skipped and operation with the RM 65 SBC is possible.

c. CMAX

CMAX ($Ø35E) tells the CRT firmware the maximum number of columns to allow before freezing the screen. This is set to 59 when the Cold Start initialization program is run. Whenever more than 59 characters need to be output, either CMAX must be increased (after initialization) or AIM65 must be set to zero.

APPENDIX D

USEFUL PROGRAM ROM SUBROUTINES

Some subroutines in the Program ROM are not callable through the main entry point at $9977. These may be called by a direct JSR to their fixed entry points. Table D-1 is a list of these routines, registers altered and description of operation.

Table D-1. Program ROM Subroutines

Name (Hex)	Address	Regs.	Description
ADCOLS	$9C2B	A	Adds COLMAX (Ø34A) to CURSOR ($DD, $DE) to give address of location below cursor.
LNSTRT	$9C15	A	Subtracts COL ($Ø359) from CURSOR ($DD, $DE) to give start address of current line.
RDCUR	$9CØ1	A,X	Reads cursor address from R6545 into locations $DD, $DE.
READ	$9C88	A,Y	Reads refresh RAM location pointed to by $DD, $DE into A. Increment $DD, $DE.
READLN	$9C6A	A,X,Y	Reads 'COLMAX' characters from refresh RAM starting at location pointed to by $DD, $DE and store in DBUFF to DBUFF + COLMAX.
SDELAY	$9CAC	X,Y	Delays one "display enable" to insure CPU accesses refresh RAM only during disabled period.
WRITE	$9CAØ	A,Y	Writes A into refresh RAM location pointed to by $DD, $DE. Incr. $DD, $DE.
WRITLN	$9C79	A,X,Y	Writes 'COLMAX' characters from DBUFF to refresh RAM starting at location pointed to by $DD, $DE.

Table D-1. Program ROM Subroutines (Continued)

Name (Hex)	Address	Regs.	Description
XDSTAR	$9C3C	A,X	Transfers display start address ($0352, $0353) to R6545. This is the first location in the refresh RAM that will appear on the screen.

NOTE

Locations $DD, $DE are temporary locations holding the cursor address. The only time these locations are guaranteed to have the correct address of the cursor is after a JSR $9C01. Otherwise, locations $DD, $DE may not be assumed to have the cursor address in them.

APPENDIX E

PROGRAM ROM ASSEMBLY LISTING

```
0003                    ;*****************************************************
0004                    ;                                                   *
0005                    ;          RM 65 CRT CONTROLLER MODULE              *
0006                    ;                                                   *
0007                    ;                PROGRAM ROM                        *
0008                    ;                                                   *
0009                    ;          R14B3                                    *
0010                    ;                                                   *
0011                    ;*****************************************************
0012                    ;
0013                    ; ROCKWELL INTERNATIONAL
0014                    ; ELECTRONICS DEVICES DIVISION
0015                    ; MICROSYSTEM PRODUCTS
0016                    ; 3310 MIRALOMA AVENUE
0017                    ; P. O. BOX 3669
0018                    ; ANAHEIM, CA.  U. S. A.  92803
0019                    ;
```



```
0021                    ;*****************************************************
0022                    ;
0023                    ;        CRT CONTROLLER (CPU/CRTC SHARED MEMORY)
0024                    ;
0025                    ;*****************************************************
0026                    ;
0027                    ;
0028
0029 9800               RS      =       $9800   ;6545 ADDRESS
0030 A406               DILINK  =       $A406   ;AIM DISPLAY ECHO
0031 9880               STATUS  =       $9880   ;DISPLAY ENABLE STATUS
0032 A438               DIBUFF  =       $A438   ;AIM DISPLAY BUFFER
0033 A415               CURPO2  =       $A415   ;AIM DISPLAY POINTER
0034 A416               CURPOS  =       $A416   ;AIM PRINTER POINTER
0035                    ;
0036                    ;  VARIABLES
0037                    ;
                                *=$DB
0038 0000               BSTART  *=*+2           ;TEMPORARY ZERO PAGE
0039 00DB               CURSOR  *=*+2           ;ADDRESS OF CURSOR
0040 00DD               ;
                                *=$F0           ;EDITOR 'FIND STRING'
0041                            ;  REFORMATTER
0042 00DF               COLUMN  *=*+1           ;TEMPORARY VARIABLES
0043 00F0               EQFLG   *=*+1           ;USE SAME BUFFER BECAUSE IT
0044 00F1               CRFLG   *=*+1           ;WILL NEVER CONFLICT W/ EDITOR
0045 00F2               SAVA    *=*+1
0046 00F3               SAVX    *=*+1
0047 00F4               SMFLG   *=*+1
0048 00F5               COMCOL  *=*+1
0049 00F6               COMCR   *=*+1
0050 00F7               ;
0051                    ;   CRT VARIABLES FROM $0347 TO $03B5
0052                    ;
0053                            *=$0347
0054 00FB               SCRMAX  *=*+2           ;# CHARS. /SCREEN
0055 0347               ROWMAX  *=*+1           ;# OF ROWS
0056 0349               COLMAX  *=*+1           ;# OF COLUMNS
0057 034A               END     *=*+2           ;DISPLAY END ADDRESS
0058 034B               DLE     *=*+1           ;PASS THRU NEXT CHAR. FLAG
0059 034D               INCH    *=*+1           ;INSERT MODE FLAG
0060 034E               EFLG    *=*+1           ;ESC. SEQUENCE FLAG
0061 034F               GRAPH   *=*+1           ;GRAPHICS MODE FLAG
0062 0350               INVERS  *=*+1           ;INVERSE VIDEO FLAG
0063 0351               DSTARL  *=*+1           ;DISPLAY START LSB
0064 0352               DSTARH  *=*+1           ;DISPLAY START ADDRESS
0065 0353               YSAV    *=*+1           ;LOC. TO SAVE Y-POS. FOR (Y,X)
0066 0354               XSAV    *=*+1           ;TEMPORARY X SAVE
0067 0355               ASAV    *=*+1           ;LOCATION TO SAVE COMMAND/DATA
0068 0356               AIM65   *=*+1           ;FLAG TO DISTINGUISH AIM 65/S.
0069 0357               ROW     *=*+1           ;ROW COUNTER
0070 0358               COL     *=*+1           ;COLUMN COUNTER
0071 0359               TABLE   *=*+1           ;0-3 TO CHOOSE STORED TABLE
0072 035A               CURP2   *=*+1           ;CRT DISPLAY POINTER
0073 035B               ECHO    *=*+2           ;ECHO ADDRESS FOR REFORMATTER
0074 035C               CMAX    *=*+1           ;MAX ALLOWED COLUMN ON CRT
0075 035E               PRFLG   *=*+1           ;COUNTER FOR AIM ON/OFF MSGS
0076 035F
```

```
0077 0360                XXX     *=*+1           ;SAVE/RESTORE X REG.
0078 0361                YYY     *=*+1           ;SAVE/RESTORE Y REG.
0079 0362                XX2     *=*+1           ;TEMPORARY SAVE X
0080 0363                XADDR   *=*+2           ;OWN TABLE ADDRESS
0081 0365                COL2    *=*+1           ;CURSOR POS. COUNTER
0082 0366                DBUFF   *=*+80          ;BUFFER FOR INSERT/DELETE LINE
0083                     ;
0084                     ;       COPYRIGHT MESSAGE
0085                     ;
0086 03B6                        *=$9800
0087                     ;
0088 9800  43            RKWELL  .BYT    'COPYRIGHT 1981'
0089 980E  52                    .BYT    'ROCKWELL '
0090 9817  49                    .BYT    'INTERNATIONAL'
0091                     ;
0092                     ;
```


CRT INITIALIZATION

```
0094                     ;       CRT/TV INITIALIZATION
0095                     ;       ---------------------
0096                     ;
0097 9824                        *=$9900
0098                     ;
0099                     ;  ENTRY FOR INITIALIZATION W/ A RETURN
0100                     ;  TO THE AIM 65 MONITOR (TABLE # IN 'TABLE')
0101                     ;
0102 9900  20 06 99      MONENT  JSR     INIT            ;CALL INITIALIZATION ROUTINE
0103 9903  4C A1 E1              JMP     $E1A1           ;RETURN TO MONITOR
0104                     ;
0105                     ;  ENTRY FOR INITIALIZATION W/ AN 'RTS'
0106                     ;  SCREEN PARAMETER TABLE # IN 'TABLE'
0107                     ;  DEFAULTS TO 22*72  525 LINES
0108                     ;  IF 'TABLE' = (0-3) THEN STORED TABLE
0109                     ;  IF 'TABLE' = 4 THEN USER-DEFINED TABLE
0110                     ;
0111 9906  AD 5A 03      INIT    LDA     TABLE           ;GET SCREEN TABLE #
0112 9909  C9 04                 CMP     #4              ;MUST BE < 4
0113 990B  90 0D                 BCC     OKTAB           ;IT IS, OK
0114 990D  D0 09                 BNE     NOTOWN
0115 990F  AC 63 03              LDY     XADDR           ;GET TABLE ADDRESS (LSB)
0116 9912  AD 64 03              LDA     XADDR+1         ;GET TABLE ADDRESS (MSB)
0117 9915  4C 22 99              JMP     SETUP           ;INIT. TO USER DEFINED SCREEN
0118 9918  A9 01         NOTOWN  LDA     #1              ;DEFAULT TO 22*72 SCREEN
0119 991A  0A            OKTAB   ASL     A               ;SHIFT FOR OFFSET
0120 991B  AA                    TAX                     ;PUT IN X FOR INDEX
0121 991C  BC AA 9F              LDY     SIZE,X          ;GET TABLE ADDRESS (LSB)
0122 991F  BD AB 9F              LDA     SIZE+1,X        ;GET TABLE ADDRESS (MSB)
0123 9922  85 DC         SETUP   STA     BSTART+1        ;STORE TABLE ADDRESS (MSB)
0124 9924  84 DB                 STY     BSTART          ;STORE TABLE ADDRESS (LSB)
0125                     ;
0126                     ;  CALLED WITH ADDRESS OF SCREEN PARAMETER TABLE
0127                     ;  STORED IN BSTART(LSB), BSTART+1(MSB)
0128                     ;
0129                     ;
0130                     ;  LOAD REGS. 0-7 IN 6545
0131                     ;
0132 9926  A0 07                 LDY     #7              ;LOAD INDEX TO VALUE
0133 9928  8C 00 98      TABLES  STY     RS              ;STORE REG. #
0134 992B  B1 DB                 LDA     (BSTART),Y      ;GET REG. VALUE
0135 992D  8D 01 98              STA     RS+1            ;STORE IN 6545
0136 9930  A9 00                 LDA     #0              ;INITIALIZE ALL ZERO VARAIBLES
0137 9932  99 4D 03              STA     DLE,Y           ;DLE TO DLE+7
0138 9935  88                    DEY                     ;DECR. INDEX
0139 9936  10 F0                 BPL     TABLES          ;NEXT REGISTER
0140                     ;
0141                     ;  LOAD REGS. 8-15 IN 6545
0142                     ;
0143 9938  A0 0F                 LDY     #15             ;LOAD REG. #, REGS. 8-15
0144 993A  A2 07                 LDX     #7              ;LOAD INDEX TO VALUE
0145 993C  8C 00 98      NXPRM   STY     RS              ;STORE REG. #
0146 993F  BD B2 9F              LDA     MIDB,X          ;GET VALUE
0147 9942  8D 01 98              STA     RS+1            ;STORE REG. VALUE
0148                     ;
```

```
0149 9945  88                DEY             ;DECR. REG. #
0150 9946  CA                DEX             ;DECR. INDEX
0151 9947  10 F3             BPL   NXPRM     ;NEXT REGISTER
0152                         ;
0153                         ;  SETUP SCREEN VARIABLES
0154                         ;
0155 9949  A0 08             LDY   #8        ;SET TABLE INDEX
0156 994B  A2 03             LDX   #3        ;LOAD VARIABLE INDEX
0157 994D  B1 DB     NLSTB   LDA   (BSTART),Y    ;LOAD TABLE VALUE
0158 994F  9D 47 03          STA   SCRMAX,X  ;STORE IN VARIABLE
0159 9952  C8                INY             ;INCR. INDEX
0160 9953  BD BA 9F          LDA   IVALS,X   ;GET 'ECHO,CMAX & PRFLG' VALS.
0161 9956  9D 5C 03          STA   ECHO,X    ;STORE
0162 9959  CA                DEX             ;DECR. TABLE INDEX
0163 995A  10 F1             BPL   NLSTB     ;NEXT VARIABLE
0164 995C  A9 90             LDA   #$90      ;RAM START ADDR. (MSB)
0165 995E  8D 53 03          STA   DSTARH    ;DISPLAY START (MSB)
0166 9961  20 54 9C          JSR   CALEND    ;CALCULATE END ADDR.
0167                         ;
0168                         ;   WARM START-UP FOR CRT
0169                         ;
0170 9964  AE 57 03  WARM    LDX   AIM65     ;GET AIM FLAG
0171 9967  F0 03             BEQ   NOAIM     ;NO AIM SO SKIP
0172 9969  20 43 9D          JSR   CRT       ;HAVE AIM, SET DILINK TO CRT
0173 996C  20 6E 9A  NOAIM   JSR   CLRHOM    ;CLEAR SCREEN, HOME CURSOR
0174 996F  60        SKIPIT  RTS
0175                         ;
```



```
0177                         ;
0178                         ;          CHARACTER/COMMAND PROCESSING
0179                         ;          ----------------------------
0180                         ;
0181                         ;
0182                         ;   USER OUTPUT ENTRY POINT
0183                         ;
0184 9970  90 FD     USERO   BCC   SKIPIT    ;RETURN WHEN 'U' TYPED
0185 9972  68                PLA             ;PULL CHAR. OFF STACK
0186 9973  C9 0A     SKIPOA  CMP   #$0A      ;LINE FEED ?
0187 9975  F0 FB             BEQ   SKIPIT    ;YES, IGNORE
0188                         ;
0189 9977  D8        OUTD    CLD             ;ENTRY WITH CMND./DATA IN 'A'
0190 9978  48                PHA             ;PUSH A ON STACK
0191 9979  8E 60 03          STX   XXX       ;SAVE Y REG.
0192 997C  8C 61 03          STY   YYY       ;AIM ON/OFF FLAG
0193 997F  AD 5F 03          LDA   PRFLG     ;NON-ZERO, DON'T BACKSPACE 3
0194 9982  D0 0E             BNE   SK3       ;BACKSPACE TO CLEAR 'ON ' OR '
0195 9984  20 8F 9D          JSR   BSPACE    ;2ND BACKSPACE
0196 9987  20 8F 9D          JSR   BSPACE    ;3RD BACKSPACE
0197 998A  20 8F 9D          JSR   BSPACE    ;RESTORE COUNTER/FLAG TO SHOW
0198 998D  A9 03             LDA   #3        ;NO BACKSPACES NEEDED
0199 998F  8D 5F 03          STA   PRFLG     ;GET STACK POINTER IN X
0200 9992  BA        SK3     TSX             ;GET MSB. OF CALLER
0201 9993  BC 03 01          LDY   $103,X    ;ASSEMBLER ECHO ?
0202 9996  C0 D1             CPY   #$D1      ;NO
0203 9998  D0 07             BNE   NASSM     ;GET LSB OF CALLER
0204 999A  BC 02 01          LDY   $102,X    ;ASSEMBLER ECHO ?
0205 999D  C0 09             CPY   #$09      ;YES, SKIP 1ST/2ND PASS SOURCE
0206 999F  F0 61             BEQ   DONE      ;GET PASSED 'A' VAL.
0207 99A1  68        NASSM   PLA             ;PUT BACK ON STACK
0208 99A2  48                PHA             ;SAVE PASSED 'A' VAL.
0209 99A3  8D 56 03          STA   ASAV      ;GET 'PASS THRU' FLAG
0210 99A6  AE 4D 03          LDX   DLE       ;DON'T PASS THRU
0211 99A9  F0 05             BEQ   SPECAL    ;RESET 'PASS THRU' FLAG TO ZER
0212 99AB  CE 4D 03          DEC   DLE       ;TREAT AS DATA-NOT CMND.
0213 99AE  F0 3A             BEQ   OUTIT     ;#$7F IS A DELETE
0214 99B0  C9 7F     SPECAL  CMP   #$7F      ;NOT DELETE
0215 99B2  D0 05             BNE   N2DEL     ;DELETE CHAR.
0216 99B4  20 8F 9D          JSR   BSPACE    ;RETURN TO CALLER
0217 99B7  D0 49             BNE   DONE      ;STRIP MSB
0218 99B9  29 7F     N2DEL   AND   #$7F      ;STORE A IN ASAV
0219 99BB  8D 56 03          STA   ASAV      GET ESC FLAG
0220 99BE  AE 4F 03          LDX   EFLG      NOT ESC SEQUENCE
0221 99C1  F0 23             BEQ   NESC      HAVE ESC SEQUENCE
0222 99C3  CA                DEX             EXPECT = OR T
0223 99C4  F0 08             BEQ   EFLG1     DECR. ESC. FLAG
0224 99C6  CA                DEX             EXPECT Y-COORDINATE
0225 99C7  F0 15             BEQ   EFLG2     CONVERT YSAV,A REG. TO BINARY
0226 99C9  20 F0 9C          JSR   CONVRT    BRANCH ALWAYS
0227 99CC  D0 34             BNE   DONE      IS CHAR. AN EQUAL SIGN
0228 99CE  C9 3D     EFLG1   CMP   #'=       YES, INCR. ESC. FLAG
0229 99D0  F0 0F             BEQ   INFLG
0230 99D2  C9 47             CMP   #'G       NOT GRAPHICS, RESET ESC.
0231 99D4  D0 03             BNE   NVALID
```

PGM. ENTRY POINTS

```
0232 99D6  20 BC 9C          JSR   GRPHON     ;GRAPHICS ON
0233 99D9  CE 4F 03  NVALID  DEC   EFLG       ;NO. DECR. ESC. FLAG
0234 99DC  F0 24             BEQ   DONE       ;RETURN TO CALLER
0235 99DE  8D 54 03  EFLG2   STA   YSAV       ;SAVE CURSOR POSITION
0236 99E1  EE 4F 03  INFLG   INC   EFLG       ;INCR. ESC. FLAG
0237 99E4  D0 1C             BNE   DONE       ;UNCONDITIONAL RETURN
0238                         ;
0239                         ;    NOT ESC. SEQUENCE, CHECK-CNTL. CHAR.
0240                         ;
0241 99E6  C9 20     NESC    CMP   #$20       ;IS CHAR. A CONTROL CHAR.
0242 99E8  90 06             BCC   CNTL       ;YES, BRANCH TO CNTL
0243 99EA  20 97 9D  OUTIT   JSR   CHAR       ;NO, PROCESS AS TEXT
0244 99ED  4C 02 9A          JMP   DONE       ;RETURN TO CALLER
0245                         ;
0246                         ;   CONTROL CHAR. JSR INDIRECT TO ROUTINE
0247                         ;
0248 99F0  0A        CNTL    ASL   A          ;SHIFT CONTROL CHAR. FOR INDEX
0249 99F1  AA                TAX              ;STORE JUMP TABLE INDEX
0250 99F2  BD BE 9F          LDA   JTBL,X     ;GET LSB OF ROUTINE ADDR.
0251 99F5  85 DB             STA   BSTART     ;STORE LSB
0252 99F7  BD BF 9F          LDA   JTBL+1,X   ;GET MSB
0253 99FA  85 DC             STA   BSTART+1   ;STORE MSB
0254 99FC  20 01 9C          JSR   RDCUR      ;READ CURSOR VALUE
0255 99FF  20 68 9E          JSR   INDIR      ;JSR TO CONTROL ROUTINE
0256 9A02  68        DONE    PLA              ;PULL A FROM STACK
0257 9A03  AE 60 03          LDX   XXX        ;RESTORE X REG.
0258 9A06  AC 61 03          LDY   YYY        ;RESTORE Y REG.
0259 9A09  60        NOECHO  RTS              ;RETURN TO CALLER
```


COMMAND ROUTINES

```
0261                         ;
0262                         ;*************************************
0263                         ;
0264                         ;  SELF TEST-OUTPUT CHARACTER SET
0265                         ;
0266                         ;*************************************
0267                         ;
0268 9A0A  20 6E 9A  STEST   JSR   CLRHOM     ;CLEAR SCREEN/HOME CURSOR
0269 9A0D  20 01 9C          JSR   RDCUR      ;GET HOMED VALUE
0270 9A10  A9 00             LDA   #$00       ;START WITH ASCII $00
0271 9A12  85 DB             STA   BSTART     ;SAVE ASCII
0272 9A14  20 92 9C  ROWOUT  JSR   INCUR2     ;INCREMENT CURSOR
0273 9A17  A5 DB             LDA   BSTART     ;GET ASCII
0274 9A19  20 A0 9C          JSR   WRITE      ;OUTPUT TO CRT
0275 9A1C  E6 DB             INC   BSTART     ;INCR. ASCII
0276 9A1E  A5 DB             LDA   BSTART     ;GET ASCII
0277 9A20  29 0F             AND   #$0F       ;MULTIPLE OF 16 ?
0278 9A22  D0 F0             BNE   ROWOUT     ;NO, SAME ROW
0279 9A24  A5 DB             LDA   BSTART     ;GET ASCII
0280 9A26  F0 32             BEQ   HOMECU     ;DONE WITH ALL 256
0281 9A28  20 A2 9B          JSR   CARTN      ;NEXT ROW
0282 9A2B  D0 E7             BNE   ROWOUT     ;BRANCH ALWAYS
0283                         ;
0284                         ;*************************************
0285                         ;
0286                         ;    DELETE LINE SUBROUTINE
0287                         ;
0288                         ;*************************************
0289                         ;
0290 9A2D  AD 58 03  DELINE  LDA   ROW        ;GET ROW #
0291 9A30  48                PHA              ;SAVE ROW
0292 9A31  20 15 9C          JSR   LNSTRT     ;TO 1ST POS. ON LINE
0293 9A34  EE 58 03  LOOP    INC   ROW        ;INCR ROW #
0294 9A37  AE 58 03          LDX   ROW        ;LOAD ROW IN X
0295 9A3A  EC 49 03          CPX   ROWMAX     ;COMPARE W/ LAST ROW #
0296 9A3D  D0 07             BNE   COPY       ;NOT =, COPY NEXT ROW OVER
0297 9A3F  68                PLA              ;RESTORE ROW
0298 9A40  8D 58 03          STA   ROW        ;STORE IT
0299 9A43  4C 2C 9B          JMP   BLANK      ;OUTPUT BLANK
0300 9A46  20 2B 9C  COPY    JSR   ADCOLS     ;ADD COLMAX TO CURSOR
0301 9A49  20 6A 9C          JSR   READLN     ;READ LINE
0302 9A4C  20 3D 9B          JSR   SMAX2      ;CURSOR UP 2 LINES
0303 9A4F  20 79 9C          JSR   WRITLN     ;WRITE LINE OVER 1 UP
0304 9A52  F0 E0             BEQ   LOOP       ;UNCONDITIONAL BRANCH
0305                         ;
0306                         ;*************************************
0307                         ;
0308                         ;   CLEAR LINE HOME CURSOR
0309                         ;
0310                         ;*************************************
0311                         ;
0312 9A54  20 15 9C  CLINE   JSR   LNSTRT     ;CURSOR TO START OF LINE
0313 9A57  4C 27 9B          JMP   WRBLNK     ;WRITE BLANK LINE
0314                         ;
0315                         ;*************************************
```

```
0316                            ;
0317                            ;     HOME CURSOR
0318                            ;
0319                            ;*************************************
0320                            ;
0321 9A5A  AD 52 03     HOMECU  LDA     DSTARL      ;CURSOR=DISPLAY START ADDR.
0322 9A5D  85 DD                STA     CURSOR      ;STORE LSB
0323 9A5F  AD 53 03             LDA     DSTARH      ;MSB
0324 9A62  85 DE                STA     CURSOR+1    ;STORE MSB
0325 9A64  A2 00                LDX     #0          ;ZERO ROW AND COL.
0326 9A66  8E 58 03             STX     ROW         ;ZERO ROW
0327 9A69  8E 59 03     ZERCOL  STX     COL         ;ZERO COL
0328 9A6C  F0 3B                BEQ     XFR         ;BRANCH ALWAYS
0329                            ;
0330                            ;*************************************
0331                            ;
0332                            ;     CLEAR SCREEN AND HOME CURSOR
0333                            ;
0334                            ;*************************************
0335                            ;
0336 9A6E  20 5A 9A     CLRHOM  JSR     HOMECU      ;HOME CURSOR
0337                            ;
0338                            ;*************************************
0339                            ;
0340                            ;     CLEAR FROM CURSOR TO END OF SCREEN
0341                            ;
0342                            ;*************************************
0343                            ;
0344 9A71  20 92 9D     CLREND  JSR     WRITBL      ;WRITE BLANK, INCR. CURSOR
0345 9A74  A6 DD                LDX     CURSOR      ;GET LSB
0346 9A76  EC 4B 03             CPX     END         ;COMPARE W/ END
0347 9A79  D0 F6                BNE     CLREND      ;NOT EQ. BRANCH
0348 9A7B  A5 DE                LDA     CURSOR+1    ;GET MSB
0349 9A7D  29 DF                AND     #$DF        ;MASK
0350 9A7F  CD 4C 03             CMP     END+1       ;COMPARE MSB'S
0351 9A82  D0 ED                BNE     CLREND      ;NOT EQ., CLEAR NEXT CHAR.
0352 9A84  60                   RTS
0353                            ;
0354                            ;*************************************
0355                            ;
0356                            ;    MOVE CURSOR TO THE LEFT
0357                            ;
0358                            ;*************************************
0359                            ;
0360 9A85  20 01 9C     CURBCK  JSR     RDCUR       ;READ CURSOR
0361 9A88  CE 59 03             DEC     COL         ;DECR. COLUMN
0362 9A8B  10 12                BPL     NCOL1       ;IF +, NOT COL. 1
0363 9A8D  CE 58 03             DEC     ROW         ;COL. 1, DECR. ROW
0364 9A90  10 06                BPL     COL1        ;NOT ROW 1
0365                            ;
0366                            ;     STORE "SCREEN MAX" IN CURSOR
0367                            ;
0368 9A92  20 2B 9C             JSR     ADCOLS      ;CURSOR DOWN 1 LINE
0369 9A95  20 68 9B             JSR     TOPROW      ;CURSOR TO BOTTOM LINE+1
0370 9A98  AE 4A 03     COL1    LDX     COLMAX      ;GET LAST COL. #
```



```
0371 9A9B  CA                   DEX                 ;DECR.
0372 9A9C  8E 59 03             STX     COL         ;STORE, COL.=LAST
0373                            ;
0374                            ;     DECREMENT CURSOR, X-FER TO 6545
0375                            ;
0376 9A9F  A6 DD        NCOL1   LDX     CURSOR      ;LSB OF CURSOR
0377 9AA1  D0 04                BNE     DDEC        ;<>0 BRANCH
0378 9AA3  18                   CLC                 ;CLEAR CARRY
0379 9AA4  20 1D 9C             JSR     BOUNDO      ;BOUNDS CHECK
0380 9AA7  C6 DD        DDEC    DEC     CURSOR      ;DECR. LSB
0381 9AA9  4C 78 9B     XFR     JMP     XFERCU      ;X-FER TO 6545
0382                            ;
0383                            ;*************************************
0384                            ;
0385                            ;     MOVE CURSOR RIGHT
0386                            ;
0387                            ;*************************************
0388                            ;
0389 9AAC  20 B2 9A     CURFWD  JSR     CURFW2      ;MOVE CURSOR FWD.
0390 9AAF  B0 A9                BCS     HOMECU      ;LAST POS. SO WRAP TO HOME POS
0391 9AB1  60                   RTS
0392                            ;
0393 9AB2  20 01 9C     CURFW2  JSR     RDCUR       ;READ CURSOR
0394 9AB5  EE 65 03             INC     COL2        ;INC. CURSOR COUNT
0395 9AB8  AE 59 03             LDX     COL         ;GET COLUMN
0396 9ABB  E8                   INX                 ;INCREMENT
0397 9ABC  EC 4A 03             CPX     COLMAX      ;LAST COLUMN ?
0398 9ABF  D0 13                BNE     NTLCOL      ;NO, BRANCH
0399 9AC1  AE 58 03             LDX     ROW         ;YES, LAST COL.
0400 9AC4  E8                   INX                 ;INCR. ROW
0401 9AC5  EC 49 03             CPX     ROWMAX      ;LAST ROW ?
0402 9AC8  D0 02                BNE     NTLROW      ;NO, BRANCH
0403 9ACA  38                   SEC                 ;SET CARRY
0404 9ACB  60                   RTS
0405                            ;
0406                            ;    LAST COL. NOT LAST ROW
0407                            ;
0408 9ACC  A2 FF        NTLROW  LDX     #$FF        ;COL. COUNT
0409 9ACE  8E 59 03             STX     COL         ;COL.=0
0410 9AD1  EE 58 03             INC     ROW         ;INCR. ROW COUNT
0411 9AD4  EE 59 03     NTLCOL  INC     COL         ;INCR. COL. COUNT
0412 9AD7  20 92 9C     INCUR   JSR     INCUR2      ;INCR. CURSOR
0413 9ADA  20 78 9B             JSR     XFERCU      ;X-FER TO 6545
0414 9ADD  18                   CLC                 ;CLEAR CARRY
0415 9ADE  60                   RTS
0416                            ;
0417                            ;*************************************
0418                            ;
0419                            ;     INSERT A LINE ROUTINE
0420                            ;
0421                            ;*************************************
0422                            ;
0423 9ADF  20 15 9C     INSLIN  JSR     LNSTRT      ;SUBTRACT COL. #
0424 9AE2  AC 58 03             LDY     ROW         ;GET CURRENT ROW
0425 9AE5  AE 49 03             LDX     ROWMAX      ;GET #ROWS/SCREEN
```

```
0426 9AE8  CA                 DEX             ;DECR.
0427 9AE9  EC 58 03           CPX   ROW       ;CURRENT=LAST ?
0428 9AEC  F0 39              BEQ   WRBLNK    ;YES, JUST BLANK IT
0429 9AEE  CA                 DEX             ;ROWMAX-2 NOW
0430 9AEF  8E 58 03           STX   ROW       ;ROW=ROWMAX-2
0431 9AF2  8C 54 03           STY   YSAV      ;SAVE ORIGINAL ROW
0432 9AF5  AD 4A 03           LDA   COLMAX    ;GET #COLS/SCREEN
0433 9AF8  0A                 ASL   A         ;MULT. BY 2
0434 9AF9  8D 55 03           STA   XSAV      ;SAVE COLMAX*2
0435 9AFC  38                 SEC             ;CLEAR BORROW
0436 9AFD  AD 4B 03           LDA   END       ;GET END SCREEN ADDR.
0437 9B00  ED 55 03           SBC   XSAV      ;SUB. 2 ROWS
0438 9B03  85 DD              STA   CURSOR    ;SAVE
0439 9B05  AD 4C 03           LDA   END+1     ;MSB OF END ADDR.
0440 9B08  20 1F 9C           JSR   BOUND5    ;SUB. BORROW AND CHECK BOUNDS
0441 9B0B  20 6A 9C  LOOP1    JSR   READLN    ;READ LINE
0442 9B0E  20 79 9C           JSR   WRITLN    ;WRITE IT OVER NEXT
0443 9B11  AE 58 03           LDX   ROW       ;GET ROW #
0444 9B14  EC 54 03           CPX   YSAV      ;EQUAL ORIG. POSITION
0445 9B17  F0 0B              BEQ   WRBL2     ;YES, BRANCH
0446 9B19  20 3D 9B           JSR   SMAX2     ;SUBTR. ROW
0447 9B1C  20 40 9B           JSR   CONTIN    ;SUBTR ROW
0448 9B1F  CE 58 03           DEC   ROW       ;DECR. ROW COUNT
0449 9B22  10 E7              BPL   LOOP1     ;NEXT ROW-UNCONDITIONAL BR.
0450                        ;
0451                        ;    WRITE BLANK LINE AT INSERT LINE
0452                        ;
0453 9B24  20 3D 9B  WRBL2    JSR   SMAX2     ;SUBTR. ROW TWICE
0454 9B27  A2 00     WRBLNK   LDX   #0        ;SET COL.=0
0455 9B29  20 69 9A           JSR   ZERCOL    ;ZERO COL., X-FER CURSOR
0456                        ;
0457                        ;   WRITE A BLANK LINE
0458                        ;
0459 9B2C  A2 00     BLANK    LDX   #0        ;INIT. X
0460 9B2E  20 92 9D  BLANK1   JSR   WRITBL    ;OUTPUT
0461 9B31  E8                 INX             ;INCR. CHAR. COUNT
0462 9B32  EC 4A 03           CPX   COLMAX    ;COMPARE W/ # CHARS./ROW
0463 9B35  D0 F7              BNE   BLANK1    ;NEXT BLANK
0464 9B37  60                 RTS
0465                        ;
0466                        ;*****************************************
0467                        ;
0468                        ;    CLEAR TO END OF LINE
0469                        ;
0470                        ;*****************************************
0471                        ;
0472 9B3B  AE 59 03  CLREOL   LDX   COL       ;GET COLUMN
0473 9B3B  10 F1              BPL   BLANK1    ;OUTPUT BLANKS
0474                        ;
0475                        ;   SUBTRACT COLMAX TWICE FROM CURSOR
0476                        ;
0477 9B3D  20 40 9B  SMAX2    JSR   CONTIN    ;SUB. ROW LENGTH FROM CURSOR
0478                        ;
0479                        ;   SUBTRACT COLMAX FROM CURSOR
0480                        ;
```



```
0481 9B40  A5 DD     CONTIN   LDA   CURSOR    ;CURSOR LSB
0482 9B42  38                 SEC             ;CLEAR BORROW
0483 9B43  ED 4A 03           SBC   COLMAX    ;SUBTR. 1 ROW
0484 9B46  4C 1B 9C           JMP   BOUND1    ;BOUNDS CHECK
0485                        ;
0486                        ;*****************************************
0487                        ;
0488                        ;       MOVE CURSOR DOWN
0489                        ;
0490                        ;*****************************************
0491                        ;
0492 9B49  EE 58 03  CURDWN   INC   ROW       ;INCR. ROW COUNT
0493 9B4C  AE 58 03           LDX   ROW       ;GET NEW ROW COUNT
0494 9B4F  EC 49 03           CPX   ROWMAX    ;LAST ROW+1 ?
0495 9B52  F0 08              BEQ   BOTROW    ;YES, WRAP TO TOP ROW
0496 9B54  20 2B 9C           JSR   ADCOLS    ;ADD ROW TO CURSOR
0497 9B57  D0 1F              BNE   XFERCU    ;BRANCH ALWAYS
0498 9B59  20 40 9B  TOTOP    JSR   CONTIN    ;SUB. ROW FROM CURSOR
0499 9B5C  CE 58 03  BOTROW   DEC   ROW       ;DECR. ROW COUNT
0500 9B5F  D0 F8              BNE   TOTOP     ;UNTIL TOP ROW
0501 9B61  F0 15              BEQ   XFERCU    ;BRANCH ALWAYS
0502                        ;
0503                        ;*****************************************
0504                        ;
0505                        ;       MOVE CURSOR UP
0506                        ;
0507                        ;*****************************************
0508                        ;
0509 9B63  CE 58 03  CURUP    DEC   ROW       ;DECR ROW
0510 9B66  10 0D              BPL   UP2       ;WASN'T TOP ROW
0511 9B68  AE 49 03  TOPROW   LDX   ROWMAX    ;GET # OF ROWS+1
0512 9B6B  CA                 DEX             ;DECR.
0513 9B6C  8E 58 03           STX   ROW       ;STORE
0514 9B6F  20 2B 9C  TOBOT    JSR   ADCOLS    ;ADD ROW LENGTH
0515 9B72  CA                 DEX             ;UNTIL BOTTOM
0516 9B73  10 FA              BPL   TOBOT     ;AGAIN
0517 9B75  20 40 9B  UP2      JSR   CONTIN
0518                        ;
0519                        ;    X-FER "CURSOR" TO 6545 CURSOR REG
0520                        ;
0521 9B78  38        XFERCU   SEC             ;CLEAR BORROW
0522 9B79  A5 DD              LDA   CURSOR    ;GET LSB
0523 9B7B  ED 52 03           SBC   DSTARL    ;SUBTR. START ADDR.
0524 9B7E  A5 DE              LDA   CURSOR+1  ;GET MSB
0525 9B80  AA                 TAX             ;SAVE CURSOR MSB
0526 9B81  ED 53 03           SBC   DSTARH    ;SUBTR. MSB OF START
0527 9B84  10 05              BPL   NOBIT     ;DON'T SET BIT 11
0528 9B86  8A                 TXA             ;RESTORE CURSOR MSB
0529 9B87  09 08              ORA   #$08      ;SET BIT 11, CURSOR < DSTART
0530 9B89  D0 01              BNE   NOBIT2    ;BRANCH ALWAYS
0531 9B8B  8A        NOBIT    TXA             ;RESTORE CURSOR MSB
0532 9B8C  AE 51 03  NOBIT2   LDX   INVERS    ;GET VIDEO FLAG
0533 9B8F  F0 02              BEQ   NOTSET    ;INVERSE ?
0534 9B91  09 20              ORA   #$20      ;YES, SET BIT 13
0535 9B93  A2 0E     NOTSET   LDX   #14       ;REG. #14
```

```
0536 9B95  20 9B 9B        JSR   S6545     ;X-FER TO 6545
0537 9B98  A5 DD           LDA   CURSOR    ;GET LSB
0538 9B9A  E8              INX             ;REG #15
0539 9B9B  8E 00 98  S6545 STX   RS        ;STORE IT
0540 9B9E  8D 01 98        STA   RS+1      ;STORE IN 6545
0541 9BA1  60              RTS
0542                       ;
0543                       ;*************************************
0544                       ;
0545                       ;     CARRIAGE RETURN
0546                       ;
0547                       ;*************************************
0548                       ;
0549 9BA2  AD 57 03  CARTN LDA   AIM65     ;GET AIM 65/S.B.C. FLAG
0550 9BA5  F0 0A           BEQ   CLOOP     ;NO AIM SO DON'T USE MONITOR
0551 9BA7  A9 0D           LDA   #$0D      ;CLEAR DISPLAY
0552 9BA9  20 05 EF        JSR   $EF05     ;MONITOR ROUTINE
0553 9BAC  A9 00           LDA   #0        ;DISABLE AIM DISPLAY
0554 9BAE  8D 01 AC        STA   $AC01     ;BY REVERSING DATA DIR.
0555 9BB1  8D 5B 03  CLOOP STA   CURP2     ;CLEAR CRT CHAR. COUNT
0556 9BB4  8D 65 03        STA   COL2      ;CLEAR CURSOR COUNT
0557 9BB7  EE 58 03        INC   ROW       ;INCR. ROW COUNT
0558 9BBA  20 01 9C        JSR   RDCUR     ;READ CURSOR
0559 9BBD  20 2B 9C        JSR   ADCOLS    ;ADD ROW LENGTH
0560 9BC0  20 15 9C        JSR   LNSTRT    ;SUBTR. COLUMN
0561 9BC3  AE 58 03        LDX   ROW       ;GET ROW
0562 9BC6  EC 49 03        CPX   ROWMAX    ;LAST ROW ?
0563 9BC9  D0 20           BNE   ZCOL      ;NO, BRANCH
0564 9BCB  CE 58 03        DEC   ROW       ;DECR. ROW
0565 9BCE  18        SCROLL CLC           ;SCROLL ROUTINE
0566 9BCF  AD 52 03        LDA   DSTARL    ;GET LSB OF START ADDR.
0567 9BD2  6D 4A 03        ADC   COLMAX    ;ADD ROW LENGTH
0568 9BD5  8D 52 03        STA   DSTARL    ;STORE LSB
0569 9BD8  AD 53 03        LDA   DSTARH    ;GET MSB
0570 9BDB  69 00           ADC   #0        ;ADD CARRY
0571 9BDD  29 97           AND   #$97      ;MASK
0572 9BDF  8D 53 03        STA   DSTARH    ;STORE MSB
0573 9BE2  20 2C 9B        JSR   BLANK     ;WRITE BLANK LINE
0574 9BE5  20 3C 9C        JSR   XDSTAR    ;STORE NEW DISPLAY ST. ADDR.
0575 9BE8  20 40 9B        JSR   CONTIN    ;SUBTR. ROW LENGTH
0576 9BEB  A2 00     ZCOL  LDX   #0        ;ZERO COLUMN
0577 9BED  4C 69 9A        JMP   ZERCOL    ;ZERO COL. COUNT
0578                       ;
0579                       ;*************************************
0580                       ;
0581                       ;   ROUTINE TO TOGGLE VIDEO MODE
0582                       ;
0583                       ;*************************************
0584                       ;
0585 9BF0  AD 51 03  VIDEO LDA   INVERS    ;GET FLAG
0586 9BF3  49 FF           EOR   #$FF      ;COMPLEMENT
0587 9BF5  8D 51 03        STA   INVERS    ;STORE
0588 9BF8  20 3C 9C        JSR   XDSTAR    ;X-FER TO 6545
0589 9BFB  20 01 9C        JSR   RDCUR     ;READ CURSOR
0590 9BFE  4C 78 9B        JMP   XFERCU    ;X-FER TO 6545
```



```
0591                       ;
0592                       ;
0593                       ;     READ 6545 CURSOR INTO "CURSOR"
0594                       ;
0595 9C01  A2 0E     RDCUR LDX   #14       ;MSB REG. #
0596 9C03  20 0C 9C        JSR   R6545     ;READ 6545 CURSOR (MSB)
0597 9C06  09 90           ORA   #$90      ;SET MSB
0598 9C08  20 37 9C        JSR   MASK      ;MASK IT
0599 9C0B  E8              INX             ;INCR REG. #
0600 9C0C  8E 00 98  R6545 STX   RS        ;STORE IN 6545
0601 9C0F  AD 01 98        LDA   RS+1      ;GET LSB
0602 9C12  85 DD           STA   CURSOR    ;STORE LSB
0603 9C14  60              RTS
0604                       ;
0605                       ;     SUBTRACT "COL" FROM "CURSOR"
0606                       ;
0607 9C15  38        LNSTRT SEC           ;SET CARRY
0608 9C16  A5 DD           LDA   CURSOR    ;GET LSB
0609 9C18  ED 59 03        SBC   COL       ;SUBTR. COL
0610 9C1B  85 DD     BOUND1 STA   CURSOR    ;STORE
0611                       ;
0612                       ;   BOUNDS CHECKING ON CURSOR
0613                       ;
0614 9C1D  A5 DE     BOUND0 LDA   CURSOR+1  ;GET MSB
0615 9C1F  E9 00     BOUND5 SBC   #0        ;SUBTR. BORROW
0616 9C21  85 DE           STA   CURSOR+1  ;STORE MSB
0617 9C23  C9 90           CMP   #$90      ;MASK MSB
0618 9C25  B0 14           BCS   SKCAR     ;NON-ZERO, BRANCH
0619 9C27  A9 97           LDA   #$97      ;GET NEW MSB
0620 9C29  D0 0E           BNE   MM2       ;UNCONDITIONAL BRANCH
0621                       ;
0622                       ;     ADD ROW LENGTH TO CURSOR
0623                       ;
0624 9C2B  18        ADCOLS CLC           ;CLEAR CARRY
0625 9C2C  A5 DD           LDA   CURSOR    ;GET LSB
0626 9C2E  6D 4A 03        ADC   COLMAX    ;ADD ROW LENGTH
0627 9C31  85 DD     STCUR STA   CURSOR    ;STORE LSB
0628 9C33  A5 DE     INCMSB LDA   CURSOR+1  ;GET DSB
0629 9C35  69 00           ADC   #0        ;ADD CARRY
0630 9C37  29 97     MASK  AND   #$97      ;MASK
0631 9C39  85 DE     MM2   STA   CURSOR+1  ;STORE MSB
0632 9C3B  60        SKCAR RTS
0633                       ;
0634                       ;  X-FERS DSTART TO 6545
0635                       ;
0636 9C3C  AD 53 03  XDSTAR LDA   DSTARH    ;GET MSB OF START
0637 9C3F  09 20           ORA   #$20      ;SET INVERSE BIT
0638 9C41  AE 51 03        LDX   INVERS    ;GET VIDEO FLAG
0639 9C44  D0 02           BNE   STORE     ;BRANCH IF INVERSE
0640 9C46  29 D7           AND   #$D7      ;CLEAR INVERSE BIT
0641 9C48  A2 0C     STORE LDX   #12       ;REG. #12
0642 9C4A  20 9B 9B        JSR   S6545     ;STORE IN 6545
0643 9C4D  E8              INX             ;INCR. REG. #
0644 9C4E  AD 52 03        LDA   DSTARL    ;GET LSB
0645 9C51  20 9B 9B        JSR   S6545     ;STORE IN 6545
```

```
0646                    ;
0647                    ;   CALCULATE END ADDRESS
0648                    ;
0649 9C54  18   CALEND  CLC             ;CLEAR CARRY
0650 9C55  AD 52 03     LDA   DSTARL    ;LSB OF START ADDR.
0651 9C58  6D 47 03     ADC   SCRMAX    ;ADD #CHAR/SCREEN LSB
0652 9C5B  8D 4B 03     STA   END       ;STORE IN END
0653 9C5E  AD 53 03     LDA   DSTARH    ;GET MSB OF START
0654 9C61  6D 48 03     ADC   SCRMAX+1  ;ADD MSB OF # CHARS
0655 9C64  29 D7        AND   #$D7      ;MASK
0656 9C66  8D 4C 03     STA   END+1     ;STORE MSB
0657 9C69  60           RTS
0658                    ;
0659                    ;     READS 1 LINE INTO DBUFF(80)
0660                    ;
0661 9C6A  A2 00    READLN LDX   #0         ;CHAR. INDEX
0662 9C6C  20 88 9C READ1  JSR   READ       ;READ CHAR
0663 9C6F  9D 66 03        STA   DBUFF,X    ;STORE IN BUFFER
0664 9C72  E8              INX              ;INCR. INDEX
0665 9C73  EC 4A 03        CPX   COLMAX     ;LAST CHAR. ?
0666 9C76  D0 F4           BNE   READ1      ;NO, REPEAT
0667 9C78  60              RTS
0668                    ;
0669                    ;     WRITES 1 LINE FROM DBUFF(80)
0670                    ;
0671 9C79  A2 00    WRITLN LDX   #0         ;CHAR. INDEX
0672 9C7B  BD 66 03 WRIT1  LDA   DBUFF,X    ;GET CHAR.
0673 9C7E  20 A0 9C        JSR   WRITE      ;OUTPUT TO CRT
0674 9C81  E8              INX              ;INCR. INDEX
0675 9C82  EC 4A 03 WRIT2  CPX   COLMAX     ;LAST CHAR. ?
0676 9C85  D0 F4           BNE   WRIT1      ;NO, REPEAT
0677 9C87  60              RTS
0678                    ;
0679                    ;     READS 1 CHARACTER FROM REFRESH RAM
0680                    ;
0681 9C88  20 AC 9C READ   JSR   SDELAY     ;DELAY FIRST DISPLAY ENABLE
0682 9C8B  AE 80 98 ENAB7  LDX   STATUS     ;GET STATUS AGAIN
0683 9C8E  30 FB           BMI   ENAB7      ;ENABLED, REPEAT
0684 9C90  B1 DD           LDA   (CURSOR),Y     ;READ CHAR. FROM REF. RAM
0685 9C92  48       INCUR2 PHA               ;SAVE A
0686 9C93  E6 DD           INC   CURSOR     ;LSB
0687 9C95  D0 04           BNE   NOCA       ;NON-ZERO, BRANCH
0688 9C97  38              SEC              ;INCR. CURSOR MSB
0689 9C98  20 33 9C        JSR   INCMSB     ;INCR. AND MASK
0690 9C9B  68       NOCA   PLA              ;RESTORE A
0691 9C9C  AE 62 03        LDX   XX2        ;RESTORE X
0692 9C9F  60              RTS
0693                    ;
0694                    ;
0695                    ;     WRITES 1 CHARACTER TO REFRESH RAM
0696                    ;
0697 9CA0  20 AC 9C WRITE  JSR   SDELAY     ;DELAY FIRST DISPLAY ENABLE
0698 9CA3  AE 80 98 ENAB2  LDX   STATUS     ;GET STATUS AGAIN
0699 9CA6  30 FB           BMI   ENAB2      ;ENABLED, REPEAT
0700 9CA8  91 DD           STA   (CURSOR),Y     ;WRITE TO REF. RAM
```



```
0701 9CAA  10 E6            BPL   INCUR2    ;RESTORE X, RETURN
0702                    ;
0703                    ;  DELAY 1 DISPLAY ENABLE
0704                    ;
0705 9CAC  8E 62 03 SDELAY  STX   XX2        ;SAVE X
0706 9CAF  A0 00            LDY   #0         ;INDIRECT INDEX
0707 9CB1  AE 80 98 ENAB5   LDX   STATUS     ;CHECK DISPLAY ENABLE
0708 9CB4  30 FB            BMI   ENAB5      ;BRANCH IF ENABLED
0709 9CB6  AE 80 98 DLY5    LDX   STATUS
0710 9CB9  10 FB            BPL   DLY5       ;DELAY LOOP
0711 9CBB  60               RTS              ;RETURN TO READ OR WRITE
0712                    ;*****************************************
0713                    ;
0714                    ;       TURN GRAPHICS "ON"
0715                    ;
0716                    ;*****************************************
0717                    ;
0718                    GRPHON JSR   B32CUR     ;BLINK CURSOR 1/32 FIELD RATE
0719 9CBC  20 C8 9C         BNE   GRTN       ;UNCOND. BRANCH
0720 9CBF  D0 03            ;
0721                    ;*****************************************
0722                    ;
0723                    ;       TURN GRAPHICS "OFF"
0724                    ;
0725                    ;*****************************************
0726                    ;
0727                    GRPHOF JSR   DCURS      ;DISPLAY CURSOR CONTINUOUSLY
0728 9CC1  20 CC 9C GRTN   STA   GRAPH      ;FLAG ON
0729 9CC4  8D 50 03        RTS
0730 9CC7  60               ;
0731                    ;*****************************************
0732                    ;
0733                    ;  BLINK CURSOR AT 1/32 SCREEN RATE
0734                    ;
0735                    ;*****************************************
0736                    ;
0737                    B32CUR LDA   #$60       ;$60 = BLINK
0738 9CC8  A9 60            BNE   CMODE      ;BRANCH ALWAYS
0739 9CCA  D0 06            ;
0740                    ;*****************************************
0741                    ;
0742                    ;    DISPLAY CURSOR CONTINUOUSLY
0743                    ;
0744                    ;*****************************************
0745                    ;
0746                    DCURS  LDA   #0         ;0 = DISPLAY CURSOR
0747 9CCC  A9 00            BEQ   CMODE      ;BRANCH ALWAYS
0748 9CCE  F0 02            ;
0749                    ;*****************************************
0750                    ;
0751                    ;  BLANK CURSOR OUT
0752                    ;
0753                    ;*****************************************
0754                    ;
0755
```

```
0756 9CD0  A9 20     BLKCUR LDA   #$20      ;BLANK CURSOR
0757 9CD2  A2 0A     CMODE  LDX   #$0A      ;$0A = CURSOR MODE REG. (6545)
0758 9CD4  4C 9B 9B         JMP   S6545     ;SET CURSOR MODE
0759                        ;
0760                        ;*************************************
0761                        ;
0762                        ;     TOGGLE "INSERT" MODE
0763                        ;
0764                        ;*************************************
0765                        ;
0766 9CD7  AD 4E 03  INSTOG LDA   INCH      ;GET INSERT MODE FLAG
0767 9CDA  49 FF            EOR   #$FF      ;COMPLEMENT
0768 9CDC  8D 4E 03         STA   INCH      ;STORE COMPLEMENTED FLAG
0769 9CDF  60               RTS
0770                        ;
0771                        ;*************************************
0772                        ;
0773                        ;     ESCAPE SEQUENCE
0774                        ;
0775                        ;*************************************
0776                        ;
0777 9CE0  A2 01     ESC    LDX   #1         ;ESC. FLAG
0778 9CE2  8E 4F 03         STX   EFLG      ;STORE
0779 9CE5  60               RTS
0780                        ;
0781                        ;*************************************
0782                        ;
0783                        ;    DELETE CHARACTER SUBROUTINE
0784                        ;
0785                        ;*************************************
0786                        ;
0787 9CE6  20 36 9D  DELCHR JSR   RDEOL     ;READ TO END OF LINE
0788 9CE9  E8               INX             ;INCR. COL
0789 9CEA  20 7B 9C         JSR   WRIT1     ;REWRITE ALL BUT 1ST
0790 9CED  4C 92 9D         JMP   WRITBL    ;BLANK AT END OF LINE
0791                        ;
0792                        ;*************************************
0793                        ;
0794                        ;     CONVERT (YSAV,ASAV) TO CURSOR ADDRESS
0795                        ;
0796                        ;*************************************
0797                        ;
0798 9CF0  20 5A 9A  CONVRT JSR   HOMECU    ;HOME CURSOR
0799 9CF3  AD 54 03         LDA   YSAV      ;GET ROW #
0800 9CF6  38               SEC             ;CLEAR BORROW
0801 9CF7  E9 20            SBC   #$20      ;SUBTRACT OFFSET
0802 9CF9  A8               TAY             ;PUT ROW IN Y
0803 9CFA  F0 12            BEQ   ROWZ      ;1ST ROW
0804 9CFC  CC 49 03         CPY   ROWMAX    ;BOUNDS CHECK
0805 9CFF  90 04            BCC   ADDCOL    ;IN BOUNDS
0806 9D01  AC 49 03         LDY   ROWMAX    ;OUT, SET TO LAST ROW
0807 9D04  88               DEY             ;Y=LAST ROW
0808 9D05  20 2B 9C  ADDCOL JSR   ADCOLS    ;ADD ROW LENGTH
0809 9D08  EE 58 03         INC   ROW       ;INCR. ROW COUNT
0810 9D0B  88               DEY             ;DECR. PARAMETER
```



```
0811 9D0C  D0 F7            BNE   ADDCOL    ;NEXT
0812 9D0E  8C 4F 03  ROWZ   STY   EFLG      ;RESET ESC. FLAG
0813 9D11  AD 56 03         LDA   ASAV      ;GET COL. #
0814 9D14  38               SEC             ;CLEAR BORROW
0815 9D15  E9 20            SBC   #$20      ;SUBTRACT OFFSET
0816 9D17  AA               TAX             ;PUT COL. IN X
0817 9D18  EC 4A 03         CPX   COLMAX    ;BOUNDS CHECK
0818 9D1B  90 04            BCC   OKCOL     ;IN BOUNDS
0819 9D1D  AE 4A 03         LDX   COLMAX    ;OUT, SET TO LAST COL.
0820 9D20  CA               DEX             ;X=LAST COLUMN
0821 9D21  8E 59 03  OKCOL  STX   COL       ;STORE IT
0822 9D24  A5 DD            LDA   CURSOR    ;LSB
0823 9D26  18               CLC             ;CLEAR CARRY
0824 9D27  6D 59 03         ADC   COL       ;ADD COLUMN
0825 9D2A  20 31 9C         JSR   STCUR
0826 9D2D  4C 78 9B         JMP   XFERCU    ;X-FER TO 6545
0827                        ;
0828                        ;*************************************
0829                        ;
0830                        ;   SET 'LOSE SPECIAL MEANING FLAG'
0831                        ;
0832                        ;*************************************
0833                        ;
0834 9D30  A9 01     DLESC  LDA   #1         ;PASS THRU=1
0835 9D32  8D 4D 03         STA   DLE       ;STORE IT
0836 9D35  60               RTS
0837                        ;
0838                        ; READ TO EOL
0839                        ;
0840 9D36  AE 59 03  RDEOL  LDX   COL       ;GET COL. COUNT
0841 9D39  20 6C 9C         JSR   READ1     ;READ TO END OF LINE
0842 9D3C  20 01 9C         JSR   RDCUR     ;READ ORIG. POSITION
0843 9D3F  AE 59 03         LDX   COL       ;RESTORE ORIG. COL.
0844 9D42  60               RTS
0845                        ;
0846                        ;*************************************
0847                        ;
0848                        ;   TOGGLE DILINK BETWEEN AIM AND CRT
0849                        ;
0850                        ;*************************************
0851                        ;
0852 9D43  CA        CRT    DEX             ;DECR. AIM65
0853 9D44  D0 0F            BNE   CRT2      ;DON'T SKIP $0A'S (LINE FEED)
0854 9D46  A9 73            LDA   #<SKIPOA  ;LOW BYTE '$0A FILTER' ENTRY
0855 9D48  D0 0D            BNE   PL65      ;SKIP $0A (LINE FEEDS)
0856 9D4A  A9 04     AIMD   LDA   #$04      ;RE-ENABLE AIM DISPLAY
0857 9D4C  8D 01 AC         STA   $AC01     ;SET DATA DIR.=OUTPUT
0858 9D4F  A9 05            LDA   #$05      ;SETUP FOR AIM (LSB)
0859 9D51  A2 EF            LDX   #$EF      ;SETUP FOR AIM (MSB)
0860 9D53  D0 04            BNE   STDEST    ;NO, SET TO AIM
0861 9D55  A9 77     CRT2   LDA   #<OUTD    ;SET DILINK=CRT
0862 9D57  A2 99     PL65   LDX   #>OUTD    ;SET DILINK=CRT
0863 9D59  8D 06 A4  STDEST STA   DILINK    ;STORE LSB
0864 9D5C  8E 07 A4         STX   DILINK+1  ;STORE MSB
0865 9D5F  60        DONE2  RTS
```

```
0866                        ;
0867                        ;
0868                        ;*************************************
0869                        ;
0870                        ;  DELETE KEY PROCESSED
0871                        ;
0872                        ;*************************************
0873                        ;
0874 9D60  AD 5B 03   DEELET LDA   CURP2     ;GET DISPLAY COUNTER
0875 9D63  C9 14            CMP   #20       ;MORE THAN 19 ?
0876 9D65  90 02            BCC   LTN20     ;NO, OK
0877 9D67  A9 13            LDA   #19       ;RESET TO 19 SO DELETE ECHOED
0878 9D69  8D 15 A4   LTN20 STA   CURP02    ;SET AIM DISPLAY POINTER
0879 9D6C  AD 5B 03         LDA   CURP2     ;GET CRT CHAR. COUNT
0880 9D6F  F0 0B            BEQ   DEL2      ;DON'T DECR. IF 0
0881 9D71  AD 65 03         LDA   COL2      ;GET CURSOR POS.
0882 9D74  CD 5E 03         CMP   CMAX      ;EQUAL TO CMAX ?
0883 9D77  F0 03            BEQ   DEL2      ;YES, DON'T DEC. CHAR COUNT
0884 9D79  CE 5B 03         DEC   CURP2     ;NO, DECR. CRT CHAR. COUNT
0885 9D7C  A9 E8      DEL2  LDA   #$E8      ;RETURN ADDR. MSB
0886 9D7E  9D 05 01         STA   $105,X    ;PUT ON STACK
0887 9D81  CE 65 03         DEC   COL2      ;DECR. CURSOR POS.
0888 9D84  A9 04            LDA   #04       ;RETURN ADDR. LSB
0889 9D86  9D 04 01         STA   $104,X    ;PUT ON STACK
0890 9D89  AD 5B 03         LDA   CURP2     ;GET POINTER < 20
0891 9D8C  9D 03 01         STA   $103,X    ;PUT ON STACK
0892 9D8F  20 85 9A   BSPACE JSR  CURBCK    ;BACK SPACE
0893 9D92  A9 20      WRITBL LDA   #$20     ;WRITE BLANK
0894 9D94  4C A0 9C         JMP   WRITE     ;OUTPUT
```



```
0896                        ;*************************************
0897                        ;
0898                        ;     ENTER CHARACTER ON SCREEN
0899                        ;
0900                        ;*************************************
0901                        ;
0902 9D97  AE 50 03   CHAR  LDX   GRAPH     ;GET GRAPHICS FLAG
0903 9D9A  F0 08            BEQ   NOGRPH    ;NOT GRAPHICS
0904 9D9C  AD 56 03         LDA   ASAV      ;GET CHAR.
0905 9D9F  09 80            ORA   #$80      ;SET MSB FOR GRAPHICS
0906 9DA1  8D 56 03         STA   ASAV      ;SAVE GRAPHICS CHAR.
0907 9DA4  AE 57 03   NOGRPH LDX  AIM65     ;GET AIM 65/S.B.C. FLAG
0908 9DA7  F0 68            BEQ   STCH00    ;NO AIM, SKIP AIM CHECKS
0909 9DA9  20 01 9C         JSR   RDCUR     ;READ CURSOR (6545)
0910 9DAC  BA               TSX             ;GET STACK POINTER
0911 9DAD  BD 08 01   ONOFF LDA   $108,X    ;CHECK 'ON ' OR 'OFF' MSG.
0912 9DB0  C9 EC            CMP   #$EC      ;MSB OF THAT CALLER
0913 9DB2  D0 0F            BNE   TAPECK    ;NOT 'ON/OFF', CHECK TAPE INPU
0914 9DB4  BD 07 01         LDA   $107,X    ;GET LSB OF 'ON/OFF' CALLER
0915 9DB7  C9 CF            CMP   #$CF      ;LSB OF THAT CALLER
0916 9DB9  D0 08            BNE   TAPECK    ;NOT 'ON/OFF', CHECK TAPE INPU
0917 9DBB  EE 15 A4         INC   CURP02    ;'ON/OFF'-INCR. DISPLAY POINTE
0918 9DBE  CE 5F 03         DEC   PRFLG     ;DECR. ON/OFF COUNTER
0919 9DC1  10 6A            BPL   NOINS     ;BRANCH ALWAYS
0920 9DC3  AD 12 A4   TAPECK LDA  $A412     ;CHECK INPUT SOURCE
0921 9DC6  C9 54            CMP   #'T       ;TAPE INPUT ?
0922 9DC8  D0 03            BNE   DELCK     ;NO, CHECK DELETE KEY
0923 9DCA  4C 56 9E         JMP   SCR0      ;YES, SKIP EDITOR STORAGE
0924 9DCD  BD 05 01   DELCK LDA   $105,X    ;CHECK FOR DELETE
0925 9DD0  BC 04 01         LDY   $104,X    ;GET MSB OF THAT CALLER
0926 9DD3  C9 E7            CMP   #$E7      ;LSB OF THAT CALLER
0927 9DD5  D0 1D            BNE   STORE2    ;NO, STORE IN EDITOR
0928 9DD7  C0 F2            CPY   #$F2      ;MSB OF THAT CALLER
0929 9DD9  F0 85            BEQ   DEELET    ;HAVE DELETE
0930 9DDB  C0 B7      CKMSG CPY   #$B7      ;CHECK 'PRINTER DOWN/END' MSGS
0931 9DDD  D0 15            BNE   STORE2    ;NOT 'KEP' MSG
0932 9DDF  AC 61 03         LDY   YYY       ;RETRIEVE Y
0933 9DE2  C0 72            CPY   #$72      ;'END' MSG. ?
0934 9DE4  F0 09            BEQ   EMSG      ;YES, GIVE CR
0935 9DE6  C0 3B            CPY   #$3B      ;'PRINTER DOWN' ?
0936 9DE8  D0 6C            BNE   SCR0      ;NO, JUST OUTPUT
0937 9DEA  A0 03            LDY   #3        ;RESET ON/OFF COUNTER
0938 9DEC  8C 5F 03         STY   PRFLG     ;DO IT
0939 9DEF  20 A2 9B   EMSG  JSR   CARTN     ;GIVE CARRIAGE RETURN
0940 9DF2  D0 62            BNE   SCR0      ;OUTPUT CHAR.
0941 9DF4  AD 56 03   STORE2 LDA  ASAV      ;GET CHAR
0942 9DF7  AC 15 A4         LDY   CURP02    ;GET DISPLAY POINTER
0943 9DFA  C8               INY             ;INC POINTER
0944 9DFB  C0 14            CPY   #20       ;> OR EQUAL TO 20
0945 9DFD  90 02            BCC   LT20      ;< 20 SO OK
0946 9DFF  A0 13            LDY   #19       ;RESET TO 19
0947 9E01  8C 15 A4   LT20  STY   CURP02    ;STORE AIM DISPLAY POINTER
0948 9E04  AC 5B 03         LDY   CURP2     ;TOTAL CHARS.
0949 9E07  C0 3B            CPY   #59       ;COMPARE W/ 60
0950 9E09  B0 06            BCS   STCH00    ;GREATER, BRANCH
```

```
0951 9E08  99 38 A4          STA   DIBUFF,Y  ;STORE IN EDIT BUFFER
0952 9E0E  EE 5B 03          INC   CURP2     ;INCR TOTAL
0953                         ;
0954                         ;*************************************
0955                         ;
0956                         ;   INSERT A CHARACTER
0957                         ;
0958                         ;*************************************
0959                         ;
0960 9E11  20 01 9C  STCHOO  JSR   RDCUR     ;READ CURSOR
0961 9E14  AE 4E 03          LDX   INCH      ;GET INSERT FLAG
0962 9E17  F0 14             BEQ   NOINS     ;NOPE, BRANCH
0963 9E19  20 36 9D          JSR   RDEOL     ;READ TO END OF LINE
0964 9E1C  AD 56 03          LDA   ASAV      ;GET CHAR.
0965 9E1F  20 A0 9C          JSR   WRITE     ;INSERT IT
0966 9E22  CE 4A 03          DEC   COLMAX    ;DECR. CHARS/ROW
0967 9E25  20 82 9C          JSR   WRIT2     ;WRITE ROW
0968 9E28  EE 4A 03          INC   COLMAX    ;RESTORE CHARS/ROW
0969 9E2B  D0 2F             BNE   SCR2      ;INC CUR. ,SCROLL IF NECESS.
0970 9E2D  AE 57 03  NOINS   LDX   AIM65     ;GET AIM 65/S.B.C. FLAG
0971 9E30  F0 24             BEQ   SCRO      ;NO AIM, SKIP DISPLAY KLUDGE
0972 9E32  AE 5B 03          LDX   CURP2     ;GET CHAR. COUNT
0973 9E35  EC 5E 03          CPX   CMAX      ;EQUAL TO CMAX ?
0974 9E38  D0 1C             BNE   SCRO      ;NO, OUTPUT
0975 9E3A  A2 3A             LDX   #$3A      ;YES, RESET CHAR. COUNT
0976 9E3C  8E 61 03          STX   YYY       ;PASS BACK TO AIM IN Y
0977 9E3F  8E 5B 03          STX   CURP2     ;RESET CRT CHAR. COUNT
0978 9E42  A2 12             LDX   #18       ;RESET PRINTER CHAR. COUNT
0979 9E44  8E 16 A4          STX   CURPOS    ;PRINTER COUNT
0980 9E47  AE 65 03          LDX   COL2      ;GET COL. COUNT
0981 9E4A  E8                INX
0982 9E4B  EC 5E 03          CPX   CMAX      ;EQUAL TO CMAX ?
0983 9E4E  F0 06             BEQ   SCRO      ;YES, DON'T BACKSPACE
0984 9E50  CE 65 03          DEC   COL2      ;DECR. CURSOR POS.
0985 9E53  20 B5 9A          JSR   CURBCK    ;BACKSPACE, THEN OUTPUT
0986 9E56  AD 56 03  SCRO    LDA   ASAV      ;GET CHAR
0987 9E59  20 A0 9C          JSR   WRITE     ;OUTPUT TO CRT
0988                         ;
0989                         ;    INCREMENT CURSOR, SCROLL IF NECESSARY
0990                         ;
0991 9E5C  20 B2 9A  SCR2    JSR   CURFW2    ;MOVE CURSOR FWD.
0992 9E5F  90 06             BCC   NOTSCR    ;DON'T SCROLL
0993 9E61  20 D7 9A          JSR   INCUR     ;INCR. CURSOR
0994 9E64  4C CE 9B          JMP   SCROLL    ;SCROLL
0995 9E67  60        NOTSCR  RTS             ;NO SCROLL, RETURN
0996                         ;
0997 9E68  6C DB 00  INDIR   JMP   (BSTART)  ;INDIRECT JUMP FOR CTRL. CHARS
```



```
0999                         ;
1000                         ;*************************************
1001                         ;
1002                         ;  ASSEMBLER REFORMATTER
1003                         ;
1004                         ;*************************************
1005                         ;
1006                         ;  INITIALIZATION
1007                         ;
1008 9E6B  B0 08     ENTRY   BCS   ASOUT
1009 9E6D  A2 00             LDX   #0
1010 9E6F  86 F5             STX   SMFLG
1011 9E71  E8                INX
1012 9E72  86 F7             STX   COMCR
1013 9E74  60                RTS
1014                         ;
1015                         ;  REFORMATTING ROUTINE
1016                         ;
1017 9E75  D8        ASOUT   CLD             ;CLEAR DEC. MODE
1018 9E76  86 F4             STX   SAVX      ;SAVE X REG.
1019 9E78  A2 28             LDX   #40
1020 9E7A  86 F6             STX   COMCOL    ;SET COMMENT COLUMN
1021 9E7C  68                PLA             ;GET A (CHAR.)
1022 9E7D  85 F3             STA   SAVA      ;SAVE CHAR.
1023 9E7F  C9 20             CMP   #$20
1024 9E81  90 04             BCC   NO        ;BRANCH IF CHAR.< #$20
1025 9E83  C9 5E             CMP   #$5E
1026 9E85  90 08             BCC   N1        ;BRANCH IF CHAR.< #$5E
1027 9E87  A2 01     NO      LDX   #1
1028 9E89  86 F2             STX   CRFLG     ;SET CARRIAGE RETN. FLAG
1029 9E8B  A6 F5             LDX   SMFLG
1030 9E8D  10 02             BPL   SKIP
1031 9E8F  06 F5             ASL   SMFLG     ;CLEAR SEMI-COLON FLAG
1032 9E91  60        SKIP    RTS             ;RETURN
1033                         ;
1034 9E92  A6 F5     N1      LDX   SMFLG     ;GET ; FLAG
1035 9E94  10 03             BPL   N1A       ;IF +, PROCESS
1036 9E96  4C 28 9F          JMP   NEXT      ;COL. 0 COMMENT,  OUTPUT CHAR.
1037 9E99  C9 3D     N1A     CMP   #'=       ;IS CHAR. AN =
1038 9E9B  D0 1B             BNE   N6        ;NO, BRANCH
1039 9E9D  A6 F2             LDX   CRFLG     ;YES, GET CARRIAGE RTN. FLAG
1040 9E9F  F0 0B             BEQ   N2        ;IF =0, BRANCH
1041 9EA1  20 3B 9F          JSR   CRLF      ;OUTPUT CARRIAGE RETURN
1042 9EA4  86 F7             STX   COMCR     ;ZERO FLAG
1043 9EA6  E8                INX             ;SET X=1
1044 9EA7  86 F1             STX   EQFLG     ;EQFLG=1
1045 9EA9  4C 38 9F          JMP   R1        ;RETURN
1046 9EAC  A6 F1     N2      LDX   EQFLG     ;GET EQFLAG
1047 9EAE  D0 03             BNE   N3        ;IF <>0 CLEAR IT
1048 9EB0  4C 28 9F          JMP   NEXT      ;IF =0 OUTPUT CHAR
1049 9EB3  46 F1     CRSP    LSR   EQFLG     ;EQFLG=0
1050 9EB5  4C 38 9F  N3      JMP   R1        ;RETURN
1051 9EB8  A6 F2             LDX   CRFLG     ;GET C.R. FLAG
1052 9EBA  F0 57             BEQ   N10       ;IF =0, BRANCH
1053 9EBC  C9 3B     N6      CMP   #';       ;CHAR. A ; ?
```

```
1054 9EBE  D0 36        BNE   N8      ;NO, BRANCH
1055 9EC0  A5 29        LDA   $29     ;CHECK ASSMBLER COL. #
1056 9EC2  D0 0B        BNE   NOTZ    ;IF <>0, BRANCH
1057 9EC4  20 3B 9F     JSR   CRLF    ;OUTPUT CARRIAGE RETURN
1058 9EC7  A9 80        LDA   #$80    ;SET FLAG
1059 9EC9  85 F5        STA   SMFLG   ;TO COL. 0 COMMENT
1060 9ECB  85 F7        STA   COMCR
1061 9ECD  D0 53        BNE   SEMI    ;OUTPUT ;
1062 9ECF  46 F2        LSR   CRFLG   ;CLEAR C.R FLAG
1063 9ED1  A2 01        LDX   #1
1064 9ED3  86 F5        STX   SMFLG   ;SET ; FLAG
1065 9ED5  D0 61        BNE   R1      ;RETURN
1066 9ED7  A6 F0        LDX   COLUMN  ;GET COL. #
1067 9ED9  E0 22        CPX   #34
1068 9EDB  90 05        BCC   SP1     ;BRANCH IF X<34
1069 9EDD  8A           TXA           ;A=COLUMN
1070 9EDE  69 02        ADC   #2      ;INCR. (CARRY IS SET-ADD 2)
1071 9EE0  85 F6        STA   COMCOL  ;STORE AS COMMENT COL. #
1072 9EE2  A6 F0        LDX   COLUMN  ;GET COL. #
1073 9EE4  E4 F6        CPX   COMCOL
1074 9EE6  F0 06        BEQ   N7      ;BRANCH IF =COMMENT COL.#
1075 9EE8  20 26 9F     JSR   SPACE   ;OUTPUT SPACE
1076 9EEB  4C E2 9E     JMP   SP1     ;LOOP
1077 9EEE  A9 3B        LDA   #';
1078 9EF0  20 28 9F     JSR   NEXT    ;OUTPUT SEMI-COLON
1079 9EF3  4C 0B 9F     JMP   NN      ;OUTPUT CHAR.
1080 9EF6  A6 F0        LDX   COLUMN  ;GET COL. #
1081 9EF8  E0 0C        CPX   #12
1082 9EFA  B0 14        BCS   N9      ;BRANCH IF >=12
1083 9EFC  A5 F7        LDA   COMCR   ;GET COMMENT FLAG
1084 9EFE  C9 80        CMP   #$80
1085 9F00  D0 04        BNE   SKCR    ;SKIP C.R. IF <>#$80
1086 9F02  06 F7        ASL   COMCR   ;CLEAR COMMENT C.R. FLAG
1087 9F04  F0 0A        BEQ   N9      ;UNCONDITIONAL BRANCH
1088 9F06  46 F2        LSR   CRFLG   ;CLEAR C.R. FLAG
1089 9F08  20 57 9F     JSR   COUNT   ;SPACE TO COL. 12
1090 9F0B  A5 F3        LDA   SAVA    ;OUTPUT CHAR.
1091 9F0D  4C 28 9F     JMP   NEXT    ;OUTPUT
1092 9F10  20 4A 9F     JSR   RESET   ;OUTPUT LOCATION COUNTER
1093 9F13  A5 F3        LDA   SAVA    ;GET CHAR.
1094 9F15  A6 F5        LDX   SMFLG   ;GET ; FLAG
1095 9F17  F0 0F        BEQ   NEXT    ;OUTPUT IF 0
1096 9F19  46 F5        LSR   SMFLG   ;CLEAR ; FLAG
1097 9F1B  C9 2E        CMP   #'.     ;CHAR.=. ?
1098 9F1D  D0 B8        BNE   NC      ;NO, BRANCH
1099 9F1F  20 4A 9F     JSR   RESET   ;OUTPUT LOC. COUNTER
1100 9F22  A9 3B        LDA   #';
1101 9F24  D0 02        BNE   NEXT    ;OUTPUT SEMI-COLON
1102 9F26  A9 20        LDA   #$20    ;OUTPUT SPACE
1103 9F28  A6 F0        LDX   COLUMN  ;GET COLUMN #
1104 9F2A  E0 47        CPX   #71     ;EQUAL TO 71
1105 9F2C  B0 08        BCS   IN1     ;YES, BRANCH
1106 9F2E  20 77 99     JSR   OUTD    ;OUTPUT TO CRT
1107 9F31  48           PHA           ;SAVE A
1108 9F32  20 77 9F     JSR   INDR2   ;ECHO THRU VECTOR
```



```
1109 9F35  68           PLA           ;RESTORE A
1110 9F36  E6 F0        INC   COLUMN  ;INCR. COL. #
1111 9F38  A6 F4        LDX   SAVX    ;RESTORE X
1112 9F3A  60           RTS           ;RETURN
1113                    ;
1114 9F3B  A9 0D        LDA   #$0D
1115 9F3D  20 77 99     JSR   OUTD    ;C.R. TO C.R.T.
1116 9F40  20 77 9F     JSR   INDR2   ;ECHO THRU VECTOR
1117 9F43  A2 00        LDX   #0
1118 9F45  86 F0        STX   COLUMN  ;CLEAR COL. COUNT
1119 9F47  86 F2        STX   CRFLG   ;CLEAR C.R. FLAG
1120 9F49  60           RTS
1121                    ;
1122                    ;
1123 9F4A  20 3B 9F     JSR   CRLF    ;OUTPUT C.R.
1124 9F4D  A5 33        LDA   $33     ;CONVERT LOC. 33 TO HEX
1125 9F4F  20 61 9F     JSR   NUMA
1126 9F52  A5 32        LDA   $32     ;CONVERT LOC. 32 TO HEX
1127 9F54  20 61 9F     JSR   NUMA
1128 9F57  20 26 9F     JSR   SPACE   ;OUTPUT SPACE
1129 9F5A  A6 F0        LDX   COLUMN  ;GET COL. #
1130 9F5C  E0 0C        CPX   #12     ;COL. 12 ?
1131 9F5E  D0 F7        BNE   COUNT   ;NO, OUTPUT SPACE
1132 9F60  60           RTS
1133                    ;
1134 9F61  48           PHA           ;SAVE A
1135 9F62  4A           LSR   A       ;SHIFT OUT LSB
1136 9F63  4A           LSR   A
1137 9F64  4A           LSR   A
1138 9F65  4A           LSR   A
1139 9F66  20 6C 9F     JSR   NOUT    ;CONVERT MSB
1140 9F69  68           PLA           ;RESTORE A
1141 9F6A  29 0F        AND   #$F     ;GET LSB
1142 9F6C  18           CLC           ;CONVERT NUMBER TO HEX
1143 9F6D  69 30        ADC   #$30
1144 9F6F  C9 3A        CMP   #$3A
1145 9F71  90 02        BCC   LT10
1146 9F73  69 06        ADC   #6
1147 9F75  D0 B1        BNE   NEXT    ;OUTPUT HEX
1148 9F77  6C 5C 03     JMP   (ECHO)  ;ECHO REFORMATTED OUTPUT
```

```
1150                            ;
1151                            ;   SCREEN VARIATION TABLES
1152                            ;
1153                            ;
1154 9F7A   6C                  T25806  .BYT    108,80,89,$59,31,2,25,28,80,25
1155 9F84   07                          .BYT    $07,$D0
1156                            ;
1157 9F86   6C                  T22725  .BYT    108,72,85,$59,26,2,22,24,72,22
1158 9F90   06                          .BYT    $06,$30
1159                            ;
1160                            ;
1161 9F92   36                  T16405  .BYT    54,40,45,$55,26,2,16,21,40,16
1162 9F9C   02                          .BYT    $02,$80
1163 9F9E   36                  T25406  .BYT    54,40,44,$55,31,2,25,28,40,25
1164 9FA8   03                          .BYT    $03,$E8
1165                            ;
1166                            ;
1167 9FAA   7A 9F               SIZE    .WORD   T25806,T22725,T25406,T16405
1168 9FB2   30                  MID8    .BYT    $30,9,0,10,$90,0,$90,0
1169 9FBA   09                  IVALS   .BYT    <NOECHO,>NOECHO,59,3
1170                            ;
1171                            ;*************************************
1172                            ;
1173                            ;   JUMP TABLE FOR CONTROL CHARACTERS
1174                            ;
1175                            ;*************************************
1176                            ;
1177 9FBE   5F 9D               JTBL    .WORD   DONE2,CLINE,CLREOL,CLRHOM,CLREND
1178 9FC8   6E 9A                       .WORD   CLRHOM,CLREND,DONE2,CURBCK,CURFWD
1179 9FD2   49 9B                       .WORD   CURDWN,CURUP,DONE2,CARTN,HOMECU
1180 9FDC   5A 9A                       .WORD   HOMECU,DLESC,DONE2,DONE2,INSTOG,DELCHR
1181 9FE8   DF 9A                       .WORD   INSLIN,DELINE,DCURS,BLKCUR,AIMD
1182 9FF2   5F 9D                       .WORD   DONE2,ESC,B32CUR,GRPHOF,STEST
1183 9FFC   F0 9B               BLAST   .WORD   VIDEO
1184                                    .END

ERRORS = 0000 <0000>
```


SYMBOL TABLE

```
ADCOLS 9C2B    0300  0368  0496  0514  0559 #0624  0808
ADDCOL 9D05    0805 #0808  0811
AIMD   9D4A   #0856  1181
AIM65  0357   #0069  0170  0549  0907  0970
ASAV   0356   #0068  0209  0219  0813  0904  0906  0941  0964  0986
ASOUT  9E75    1008 #1017
BLANK  9B2C    0299 #0459  0573
BLANK1 9B2E   #0460  0463  0473
BLAST  9FFC   #1183
BLKCUR 9CD0   #0756  1181
BOTROW 9B5C    0495 #0499
BOUNDO 9C1D    0379 #0614
BOUND1 9C1B    0484 #0610
BOUND5 9C1F    0440 #0615
BSPACE 9D8F    0195  0196  0197  0216 #0892
BSTART 00DB   #0039  0124  0125  0135  0157  0251  0253  0271  0273  0275
               0276  0279  0997
B32CUR 9CC8    0719 #0738  1182
CALEND 9C54    0166 #0649
CARTN  9BA2    0281 #0549  0939  1179
CHAR   9D97    0243 #0902
CKMSG  9DDB   #0930
CLINE  9A54   #0312  1177
CLOOP  9BB1    0550 #0555
CLREND 9A71   #0344  0347  0351  1177  1178
CLREOL 9B38   #0472  1177
CLRHOM 9A6E    0173  0268 #0336  1177  1178
CMAX   035E   #0075  0882  0973  0982
CMODE  9CD2    0739  0748 #0757
CNTL   99F0    0242 #0248
COL    0359   #0071  0327  0361  0372  0395  0409  0411  0472  0609  0821
               0824  0840  0843
COLMAX 034A   #0057  0370  0397  0432  0462  0483  0567  0626  0665  0675
               0817  0819  0966  0968
COLUMN 00F0   #0043  1066  1072  1080  1103  1110  1118  1129
COL1   9A98    0364 #0370
COL2   0365   #0081  0394  0556  0881  0887  0980  0984
COMCOL 00F6   #0049  1020  1071  1073
COMCR  00F7   #0050  1012  1042  1060  1083  1086
CONTIN 9B40    0447  0477 #0481  0498  0517  0575
CONVRT 9CF0    0226 #0798
COPY   9A46    0296 #0300
COUNT  9F57    1089 #1128  1131
CRFLG  00F2   #0045  1028  1039  1051  1062  1088  1119
CRLF   9F3B    1041  1057 #1114  1123
CRSP   9EB0   #1048
CRT    9D43    0172 #0852
CRT2   9D55    0853 #0861
CURBCK 9A85   #0360  0892  0985  1178
CURDWN 9B49   #0492  1179
CURFWD 9AAC   #0389  1178
CURFW2 9AB2    0389 #0393  0991
CURPOS A416   #0034  0979
CURPO2 A415   #0033  0878  0917  0942  0947
CURP2  035B   #0073  0555  0874  0879  0884  0890  0948  0952  0972  0977
```

SYMBOL TABLE

```
CURSOR 00DD    #0040  0322  0324  0345  0348  0376  0380  0438  0481  0522
                0524  0537  0602  0608  0610  0614  0616  0625  0627  0628
                0631  0684  0686  0700  0822
CURUP  9B63    #0509  1179
DBUFF  0366    #0082  0663  0672
DCURS  9CCC     0728 #0747  1181
DDEC   9AA7     0377 #0380
DEELET 9D60    #0874  0929
DELCHR 9CE6    #0787  1180
DELCK  9DCD     0922 #0924
DELINE 9A2D    #0290  1181
DEL2   9D7C     0880  0883 #0885
DIBUFF A438    #0032  0951
DILINK A406    #0030  0863  0864
DLE    034D    #0059  0138  0210  0212  0835
DLESC  9D30    #0834  1180
DLY5   9CB6    #0709  0710
DONE   9A02     0206  0217  0227  0234  0237  0244 #0256
DONE2  9D5F    #0865  1177  1178  1179  1180  1182
DSTARH 0353    #0065  0165  0323  0526  0569  0572  0636  0653
DSTARL 0352    #0064  0321  0523  0566  0568  0644  0650
ECHO   035C    #0074  0161  1148
EFLG   034F    #0061  0220  0233  0236  0778  0812
EFLG1  99CE     0223 #0228
EFLG2  99DE     0225 #0235
EMSG   9DEF     0934 #0939
ENAB2  9CA3    #0698  0699
ENAB5  9CB1    #0707  0708
ENAB7  9C8B    #0682  0683
END    034B    #0058  0346  0350  0436  0439  0652  0656
ENTRY  9E6B    #1008
EOFLG  00F1    #0044  1044  1046  1049
ESC    9CE0    #0777  1182
GRAPH  0350    #0062  0729  0902
GRPHOF 9CC1    #0728  1182
GRPHON 9CBC     0232 #0719
GRTN   9CC4     0720 #0729
HOMECU 9A5A     0280 #0321  0336  0390  0798  1179  1180
INCH   034E    #0060  0766  0768  0961
INCMSB 9C33    #0628  0689
INCUR  9AD7    #0412  0993
INCUR2 9C92     0272  0412 #0685  0701
INDIR  9E68     0255 #0997
INDR2  9F77     1108  1116 #1148
INFLG  99E1     0229 #0236
INIT   9906     0102 #0112
INSLIN 9ADF    #0423  1181
INSTOG 9CD7    #0766  1180
INVERS 0351    #0063  0532  0585  0587  0638
IN1    9F36     1105 #1110
IVALS  9FBA     0160 #1169
JTBL   9FBE     0250  0252 #1177
LNSTRT 9C15     0292  0312  0423  0560 #0607
LOOP   9A34    #0293  0304
LOOP1  9B0B    #0441  0449
```


SYMBOL TABLE

```
LTN20  9D69     0876 #0878
LT10   9F75     1145 #1147
LT20   9E01     0945 #0947
MASK   9C37     0598 #0630
MID8   9FB2     0147 #1168
MM2    9C39     0620 #0631
MONENT 9900    #0102
NASSM  99A1     0203 #0207
NC     9ED7    #1066  1098
NCOL1  9A9F     0362 #0376
NESC   99E6     0221 #0241
NEXT   9F28     1036  1048  1078  1091  1095  1101 #1103  1147
NLST8  994D    #0157  0163
NN     9F0B     1079 #1090
NOAIM  996C     0171 #0173
NOBIT  9B8B     0527 #0531
NOBIT2 9B8C     0530 #0532
NOCA   9C9B     0687 #0690
NOECHO 9A09    #0259  1169
NOGRPH 9DA4     0903 #0907
NOINS  9E2D     0919  0962 #0970
NOTOWN 9918     0115 #0119
NOTSCR 9E67     0992 #0995
NOTSET 9B93     0533 #0535
NOTZ   9ECF     1056 #1062
NOUT   9F6C     1139 #1142
NTLCOL 9AD4     0398 #0411
NTLROW 9ACC     0402 #0408
NUMA   9F61     1125  1127 #1134
NVALID 99D9     0231 #0233
NXPRM  993C    #0146  0151
NO     9E87     1024 #1027
N1     9E92     1026 #1034
N1A    9E99     1035 #1037
N10    9F13     1052 #1093
N2     9EAC     1040 #1046
N2DEL  99B9     0215 #0218
N3     9EB3     1047 #1049
N6     9EB8     1038 #1051
N7     9EEE     1074 #1077
N8     9EF6     1054 #1080
N9     9F10     1082  1087 #1092
OKCOL  9D21     0818 #0821
OKTAB  991A     0114 #0120
ONOFF  9DAD    #0911
OUTD   9977    #0189  0861  0862  1106  1115
OUTIT  99EA     0213 #0243
PL65   9D57     0855 #0862
PRFLG  035F    #0076  0193  0199  0918  0938
RDCUR  9C01     0254  0269  0360  0393  0558  0589 #0595  0842  0909  0960
RDEOL  9D36     0787 #0840  0963
READ   9C88     0662 #0681
READLN 9C6A     0301  0441 #0661
READ1  9C6C    #0662  0666  0841
RESET  9F4A     1092  1099 #1123
```

SYMBOL TABLE

```
RKWELL 9800  #0088
ROW    0358  #0070  0290  0293  0294  0298  0326  0363  0399  0410  0424
             0427  0430  0443  0448  0492  0493  0499  0509  0513  0557
             0561  0564  0809
ROWMAX 0349  #0056  0295  0401  0425  0494  0511  0562  0804  0806
ROWOUT 9A14  #0272  0278  0282
ROWZ   9D0E   0803 #0812
RS     9800  #0029  0134  0136  0146  0148  0539  0540  0600  0601
R1     9F38   1045  1050  1065 #1111
R6545  9C0C   0596 #0600
SAVA   00F3  #0046  1022  1090  1093
SAVX   00F4  #0047  1018  1111
SCRMAX 0347  #0055  0158  0651  0654
SCROLL 9BCE  #0565  0994
SCR0   9E56   0923  0936  0940  0971  0974  0983 #0986
SCR2   9E5C   0969 #0991
SDELAY 9CAC   0681  0697 #0705
SEMI   9F22   1061 #1100
SETUP  9922   0118 #0124
SIZE   9FAA   0122  0123 #1167
SKCAR  9C3B   0618 #0632
SKCR   9F06   1085 #1088
SKIP   9E91   1030 #1032
SKIPIT 996F  #0174  0184  0187
SKIPOA 9973  #0186  0854
SK3    9992   0194 #0200
SMAX2  9B3D   0302  0446  0453 #0477
SMFLG  00F5  #0048  1010  1029  1031  1034  1059  1064  1094  1096
SPACE  9F26   1075 #1102  1128
SPECAL 99B0   0211 #0214
SP1    9EE2   1068 #1072  1076
STATUS 9880  #0031  0682  0698  0707  0709
STCHOO 9E11   0908  0950 #0960
STCUR  9C31  #0627  0825
STDEST 9D59   0860 #0863
STEST  9A0A  #0268  1182
STORE  9C48   0639 #0641
STORE2 9DF4   0927  0931 #0941
S6545  9B9B   0536 #0539  0642  0645  0758
TABLE  035A  #0072  0112
TABLE8 9928  #0134  0140
TAPECK 9DC3   0913  0916 #0920
TOBOT  9B6F  #0514  0516
TOPROW 9B68   0369 #0511
TOTOP  9B59  #0498  0500
T16405 9F92  #1161  1167
T22725 9F86  #1157  1167
T25406 9F9E  #1163  1167
T25806 9F7A  #1154  1167
UP2    9B75   0510 #0517
USER0  9970  #0184
VIDEO  9BF0  #0585  1183
WARM   9964  #0170
WRBLNK 9B27   0313  0428 #0454
WRBL2  9B24   0445 #0453
```


SYMBOL TABLE

```
WRITBL 9D92   0344  0460  0790 #0893
WRITE  9CA0   0274  0673 #0697  0894  0965  0987
WRITLN 9C79   0303  0442 #0671
WRIT1  9C7B  #0672  0676  0789
WRIT2  9C82  #0675  0967
XADDR  0363  #0080  0116  0117
XDSTAR 9C3C   0574  0588 #0636
XFERCU 9B7B   0381  0413  0497  0501 #0521  0590  0826
XFR    9AA9   0328 #0381
XSAV   0355  #0067  0434  0437
XXX    0360  #0077  0191  0257
XX2    0362  #0079  0691  0705
YSAV   0354  #0066  0235  0431  0444  0799
YYY    0361  #0078  0192  0258  0932  0976
ZCOL   9BEB   0563 #0576
ZERCOL 9A69  #0327  0455  0577
.NARG  ****
```

APPENDIX F

CHARACTER ROM LISTING

The R3265 ROM contains the dot matrix patterns for the 256 printable characters (see Table 3-1). The ROM is formatted as follows:

a. The bit pattern for each character is composed of 16 bytes:

   Byte 1 = top row (row 1)
   Byte 9 = bottom row (row 10)
   Bytes A-F = zero

b. The first character (blank), hex = $00, starts at address $000. The last character (solid block), hex code = $FF, starts at address $FF0.

c. The ROM listing is formatted as follows:

```
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   -- Byte
       1  2  3  4  5  6  7  8  9 10  -  -  -  -  -  -   -- Row No.
Addr
000   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   -- Char $00
                              .
                              .
                              .
FF0   FE FE FE FE FE FE FE FE FE FE 00 00 00 00 00 00   -- Char $FF
```

```
000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
010 00 7C 82 9A A2 A2 9A 82 7C 00 00 00 00 00 00 00
020 00 7C 82 BA A6 BA AA A6 7C 00 00 00 00 00 00 00
030 00 20 40 FE 00 FE 04 08 00 00 00 00 00 00 00 00
040 00 F0 C0 A0 90 08 04 02 00 00 00 00 00 00 00 00
050 00 1E 06 0A 12 20 40 80 00 00 00 00 00 00 00 00
060 00 80 40 20 12 0A 06 1E 00 00 00 00 00 00 00 00
070 00 02 04 08 90 A0 C0 F0 00 00 00 00 00 00 00 00
080 00 00 20 40 FE 40 20 00 00 00 00 00 00 00 00 00
090 00 00 08 04 FE 04 08 00 00 00 00 00 00 00 00 00
0A0 00 10 10 10 10 54 38 10 00 00 00 00 00 00 00 00
0B0 00 10 38 54 10 10 10 10 00 00 00 00 00 00 00 00
0C0 00 10 38 54 10 10 54 38 10 00 00 00 00 00 00 00
0D0 00 70 80 80 9C 72 1C 14 12 00 00 00 00 00 00 00
0E0 00 70 80 60 1C F2 1C 12 1C 00 00 00 00 00 00 00
0F0 00 E2 84 E8 10 2C 52 8C 00 00 00 00 00 00 00 00
100 00 0C 12 12 12 0C 00 00 00 00 00 00 00 00 00 00
110 00 04 0C 04 04 0E 00 00 00 00 00 00 00 00 00 00
120 00 0C 12 04 08 1E 00 00 00 00 00 00 00 00 00 00
130 00 1C 02 04 02 1C 00 00 00 00 00 00 00 00 00 00
140 00 14 14 1E 04 04 00 00 00 00 00 00 00 00 00 00
150 00 1E 10 1C 02 1C 00 00 00 00 00 00 00 00 00 00
160 00 0E 10 1C 12 0C 00 00 00 00 00 00 00 00 00 00
170 00 1E 02 04 08 08 00 00 00 00 00 00 00 00 00 00
180 00 0C 12 0C 12 0C 00 00 00 00 00 00 00 00 00 00
190 00 0C 12 0E 02 0C 00 00 00 00 00 00 00 00 00 00
1A0 00 FE 7C 38 10 00 00 00 00 00 00 00 00 00 00 00
1B0 00 00 00 00 00 10 38 7C FE 00 00 00 00 00 00 00
1C0 00 80 C0 E0 F0 E0 C0 80 00 00 00 00 00 00 00 00
1D0 00 02 06 0E 1E 0E 06 02 00 00 00 00 00 00 00 00
1E0 00 08 08 08 08 78 48 78 00 00 00 00 00 00 00 00
1F0 00 00 00 00 00 78 48 78 00 00 00 00 00 00 00 00
200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
210 00 10 10 10 10 10 00 00 10 00 00 00 00 00 00 00
220 00 28 28 28 00 00 00 00 00 00 00 00 00 00 00 00
230 00 28 28 7C 28 7C 28 28 00 00 00 00 00 00 00 00
240 00 10 3C 50 38 14 78 10 00 00 00 00 00 00 00 00
250 00 64 64 08 10 20 4C 4C 00 00 00 00 00 00 00 00
260 00 20 50 50 20 54 48 34 00 00 00 00 00 00 00 00
270 00 08 10 20 00 00 00 00 00 00 00 00 00 00 00 00
280 00 08 10 20 20 20 10 08 00 00 00 00 00 00 00 00
290 00 20 10 08 08 08 10 20 00 00 00 00 00 00 00 00
2A0 00 00 10 54 38 54 10 00 00 00 00 00 00 00 00 00
2B0 00 00 10 10 7C 10 10 00 00 00 00 00 00 00 00 00
2C0 00 00 00 00 00 00 20 20 40 00 00 00 00 00 00 00
2D0 00 00 00 00 7C 00 00 00 00 00 00 00 00 00 00 00
2E0 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 00
2F0 00 04 04 08 10 20 40 40 00 00 00 00 00 00 00 00
```



```
300 00 38 44 4C 54 64 44 38 00 00 00 00 00 00 00 00
310 00 10 30 50 10 10 10 7C 00 00 00 00 00 00 00 00
320 00 38 44 04 08 30 40 7C 00 00 00 00 00 00 00 00
330 00 78 04 08 10 08 04 78 00 00 00 00 00 00 00 00
340 00 08 18 28 48 7C 08 08 00 00 00 00 00 00 00 00
350 00 7C 40 78 04 04 44 38 00 00 00 00 00 00 00 00
360 00 1C 20 40 78 44 44 38 00 00 00 00 00 00 00 00
370 00 7C 04 08 10 20 20 20 00 00 00 00 00 00 00 00
380 00 38 44 44 38 44 44 38 00 00 00 00 00 00 00 00
390 00 38 44 44 3C 04 08 70 00 00 00 00 00 00 00 00
3A0 00 00 00 10 00 00 00 10 00 00 00 00 00 00 00 00
3B0 00 00 00 20 00 00 20 20 40 00 00 00 00 00 00 00
3C0 00 0C 10 20 40 20 10 0C 00 00 00 00 00 00 00 00
3D0 00 00 00 7C 00 7C 00 00 00 00 00 00 00 00 00 00
3E0 00 60 10 08 04 08 10 60 00 00 00 00 00 00 00 00
3F0 00 38 44 04 08 10 00 10 00 00 00 00 00 00 00 00
400 00 38 44 5C 54 5C 40 38 00 00 00 00 00 00 00 00
410 00 38 44 44 7C 44 44 44 00 00 00 00 00 00 00 00
420 00 78 44 44 78 44 44 78 00 00 00 00 00 00 00 00
430 00 38 44 40 40 40 44 38 00 00 00 00 00 00 00 00
440 00 70 48 44 44 44 48 70 00 00 00 00 00 00 00 00
450 00 7C 40 40 78 40 40 7C 00 00 00 00 00 00 00 00
460 00 7C 40 40 78 40 40 40 00 00 00 00 00 00 00 00
470 00 38 44 40 4C 44 44 3C 00 00 00 00 00 00 00 00
480 00 44 44 44 7C 44 44 44 00 00 00 00 00 00 00 00
490 00 38 10 10 10 10 10 38 00 00 00 00 00 00 00 00
4A0 00 1C 08 08 08 08 48 30 00 00 00 00 00 00 00 00
4B0 00 44 48 50 60 50 48 44 00 00 00 00 00 00 00 00
4C0 00 40 40 40 40 40 40 7C 00 00 00 00 00 00 00 00
4D0 00 44 6C 54 54 44 44 44 00 00 00 00 00 00 00 00
4E0 00 44 64 64 54 54 4C 44 00 00 00 00 00 00 00 00
4F0 00 38 44 44 44 44 44 38 00 00 00 00 00 00 00 00
500 00 78 44 44 78 40 40 40 00 00 00 00 00 00 00 00
510 00 38 44 44 44 44 54 48 34 00 00 00 00 00 00 00
520 00 78 44 44 44 78 50 48 44 00 00 00 00 00 00 00
530 00 3C 40 40 38 04 04 78 00 00 00 00 00 00 00 00
540 00 7C 10 10 10 10 10 10 00 00 00 00 00 00 00 00
550 00 44 44 44 44 44 44 38 00 00 00 00 00 00 00 00
560 00 44 44 44 28 28 10 10 00 00 00 00 00 00 00 00
570 00 44 44 44 44 54 54 6C 44 00 00 00 00 00 00 00
580 00 44 44 28 10 28 44 44 00 00 00 00 00 00 00 00
590 00 44 44 44 38 10 10 10 00 00 00 00 00 00 00 00
5A0 00 7C 04 08 10 20 40 7C 00 00 00 00 00 00 00 00
5B0 00 78 60 60 60 60 60 78 00 00 00 00 00 00 00 00
5C0 00 40 40 20 10 08 04 04 00 00 00 00 00 00 00 00
5D0 00 3C 0C 0C 0C 0C 0C 3C 00 00 00 00 00 00 00 00
5E0 00 00 00 10 28 44 00 00 00 00 00 00 00 00 00 00
5F0 00 00 00 00 00 00 00 7C 00 00 00 00 00 00 00 00
```

```
600 00 00 20 10 08 00 00 00 00 00 00 00 00 00 00 00
610 00 00 00 38 04 3C 44 3C 00 00 00 00 00 00 00 00
620 00 40 40 40 78 44 44 78 00 00 00 00 00 00 00 00
630 00 00 00 3C 40 40 40 3C 00 00 00 00 00 00 00 00
640 00 04 04 04 3C 44 44 3C 00 00 00 00 00 00 00 00
650 00 00 00 38 44 78 40 38 00 00 00 00 00 00 00 00
660 00 08 14 10 38 10 10 10 00 00 00 00 00 00 00 00
670 00 00 00 38 44 44 3C 04 04 38 00 00 00 00 00 00
680 00 40 40 40 78 44 44 44 00 00 00 00 00 00 00 00
690 00 00 10 00 10 10 10 10 00 00 00 00 00 00 00 00
6A0 00 00 08 00 08 08 08 08 48 30 00 00 00 00 00 00
6B0 00 20 20 24 28 30 28 24 00 00 00 00 00 00 00 00
6C0 00 10 10 10 10 10 10 10 00 00 00 00 00 00 00 00
6D0 00 00 00 EC 92 92 92 82 00 00 00 00 00 00 00 00
6E0 00 00 00 58 64 44 44 44 00 00 00 00 00 00 00 00
6F0 00 00 00 38 44 44 44 38 00 00 00 00 00 00 00 00
700 00 00 00 78 44 44 78 40 40 40 00 00 00 00 00 00
710 00 00 00 3C 44 44 3C 04 04 04 00 00 00 00 00 00
720 00 00 00 58 64 40 40 40 00 00 00 00 00 00 00 00
730 00 00 00 3C 40 38 04 78 00 00 00 00 00 00 00 00
740 00 10 10 7C 10 10 14 08 00 00 00 00 00 00 00 00
750 00 00 00 44 44 44 44 38 00 00 00 00 00 00 00 00
760 00 00 00 44 44 44 28 10 00 00 00 00 00 00 00 00
770 00 00 00 82 82 92 92 6C 00 00 00 00 00 00 00 00
780 00 00 00 44 28 10 28 44 00 00 00 00 00 00 00 00
790 00 00 00 44 44 44 3C 04 04 38 00 00 00 00 00 00
7A0 00 00 00 7C 08 10 20 7C 00 00 00 00 00 00 00 00
7B0 00 18 20 20 40 20 20 18 00 00 00 00 00 00 00 00
7C0 00 10 10 10 00 10 10 10 00 00 00 00 00 00 00 00
7D0 00 30 08 08 04 08 08 30 00 00 00 00 00 00 00 00
7E0 00 00 00 24 54 48 00 00 00 00 00 00 00 00 00 00
7F0 00 7C 7C 7C 7C 7C 7C 7C 00 00 00 00 00 00 00 00
800 00 00 00 32 4C 4C 32 00 00 00 00 00 00 00 00 00
810 00 38 44 44 78 44 44 78 40 40 00 00 00 00 00 00
820 00 00 44 28 10 28 28 10 00 00 00 00 00 00 00 00
830 00 00 00 10 28 44 82 FE 00 00 00 00 00 00 00 00
840 00 00 00 3C 40 7C 40 3C 00 00 00 00 00 00 00 00
850 00 18 24 24 3C 24 24 18 00 00 00 00 00 00 00 00
860 00 00 C0 20 10 18 24 42 00 00 00 00 00 00 00 00
870 00 00 44 44 44 44 64 5A 40 40 00 00 00 00 00 00
880 00 00 02 7C A8 28 28 28 00 00 00 00 00 00 00 00
890 00 FE 28 28 28 28 28 28 00 00 00 00 00 00 00 00
8A0 00 00 00 3E 48 48 48 30 00 00 00 00 00 00 00 00
8B0 00 7C 20 10 08 10 20 7C 00 00 00 00 00 00 00 00
8C0 00 00 02 7C 90 10 10 10 00 00 00 00 00 00 00 00
8D0 00 10 10 38 54 54 38 10 10 00 00 00 00 00 00 00
8E0 00 D6 54 54 38 10 10 10 00 00 00 00 00 00 00 00
8F0 00 00 00 44 82 92 92 6C 00 00 00 00 00 00 00 00
```



```
900 00 7C 82 82 82 44 44 C6 00 00 00 00 00 00 00 00
910 00 0C 12 12 0C 00 00 00 00 00 00 00 00 00 00 00
920 00 00 00 6C 92 6C 00 00 00 00 00 00 00 00 00 00
930 00 10 10 7C 10 10 00 7C 00 00 00 00 00 00 00 00
940 00 7C 00 10 10 7C 10 10 00 00 00 00 00 00 00 00
950 00 00 10 00 7C 00 10 00 00 00 00 00 00 00 00 00
960 00 00 64 98 00 64 98 00 00 00 00 00 00 00 00 00
970 00 00 10 00 7C 00 7C 00 00 00 00 00 00 00 00 00
980 00 04 08 7C 10 7C 20 40 00 00 00 00 00 00 00 00
990 00 00 7C 00 7C 00 7C 00 00 00 00 00 00 00 00 00
9A0 00 04 08 10 20 7C 00 7C 00 00 00 00 00 00 00 00
9B0 00 40 20 10 08 7C 00 7C 00 00 00 00 00 00 00 00
9C0 00 44 44 44 44 44 44 38 00 00 00 00 00 00 00 00
9D0 00 38 44 44 44 44 44 44 00 00 00 00 00 00 00 00
9E0 00 3E 20 20 20 20 A0 60 00 00 00 00 00 00 00 00
9F0 00 00 02 04 08 90 A0 C0 80 00 00 00 00 00 00 00
A00 00 08 14 10 10 10 10 50 20 00 00 00 00 00 00 00
A10 00 0C 10 38 54 54 38 10 60 00 00 00 00 00 00 00
A20 00 1C 20 18 24 18 04 38 00 00 00 00 00 00 00 00
A30 00 00 10 3C 50 50 3C 10 00 00 00 00 00 00 00 00
A40 00 18 24 20 70 20 20 7C 00 00 00 00 00 00 00 00
A50 00 44 28 10 38 10 38 10 00 00 00 00 00 00 00 00
A60 00 18 30 60 E0 60 30 18 00 00 00 00 00 00 00 00
A70 00 30 18 0C 0E 0C 18 30 00 00 00 00 00 00 00 00
A80 00 10 38 7C FE 7C 10 10 7C 00 00 00 00 00 00 00
A90 00 6C EE FE FE 7C 38 10 00 00 00 00 00 00 00 00
AA0 00 10 38 10 54 FE 54 10 7C 00 00 00 00 00 00 00
AB0 00 10 38 7C FE 7C 38 10 00 00 00 00 00 00 00 00
AC0 00 38 7C FE FE FE 7C 38 00 00 00 00 00 00 00 00
AD0 00 82 7C 44 44 44 7C 82 00 00 00 00 00 00 00 00
AE0 00 38 7C 38 10 7C 10 28 44 00 00 00 00 00 00 00
AF0 00 38 44 28 10 7C 10 28 44 00 00 00 00 00 00 00
B00 00 00 00 00 0C 12 12 12 0C 00 00 00 00 00 00 00
B10 00 00 00 00 04 0C 04 04 0E 00 00 00 00 00 00 00
B20 00 00 00 00 0C 12 04 08 1E 00 00 00 00 00 00 00
B30 00 00 00 00 1C 02 0C 02 1C 00 00 00 00 00 00 00
B40 00 00 00 00 14 14 1E 04 04 00 00 00 00 00 00 00
B50 00 00 00 00 1E 10 1C 02 1C 00 00 00 00 00 00 00
B60 00 00 00 00 0E 10 1C 12 0C 00 00 00 00 00 00 00
B70 00 00 00 00 1E 02 04 08 08 00 00 00 00 00 00 00
B80 00 00 00 00 0C 12 0C 12 0C 00 00 00 00 00 00 00
B90 00 00 00 00 0C 12 0E 02 1C 00 00 00 00 00 00 00
BA0 00 10 00 10 20 40 44 38 00 00 00 00 00 00 00 00
BB0 00 10 00 38 04 3C 44 3C 00 00 00 00 00 00 00 00
BC0 00 44 00 38 04 3C 44 3C 00 00 00 00 00 00 00 00
BD0 00 08 10 38 04 3C 44 3C 00 00 00 00 00 00 00 00
BE0 00 20 10 38 04 3C 44 3C 00 00 00 00 00 00 00 00
BF0 00 10 28 38 04 3C 44 3C 00 00 00 00 00 00 00 00
```

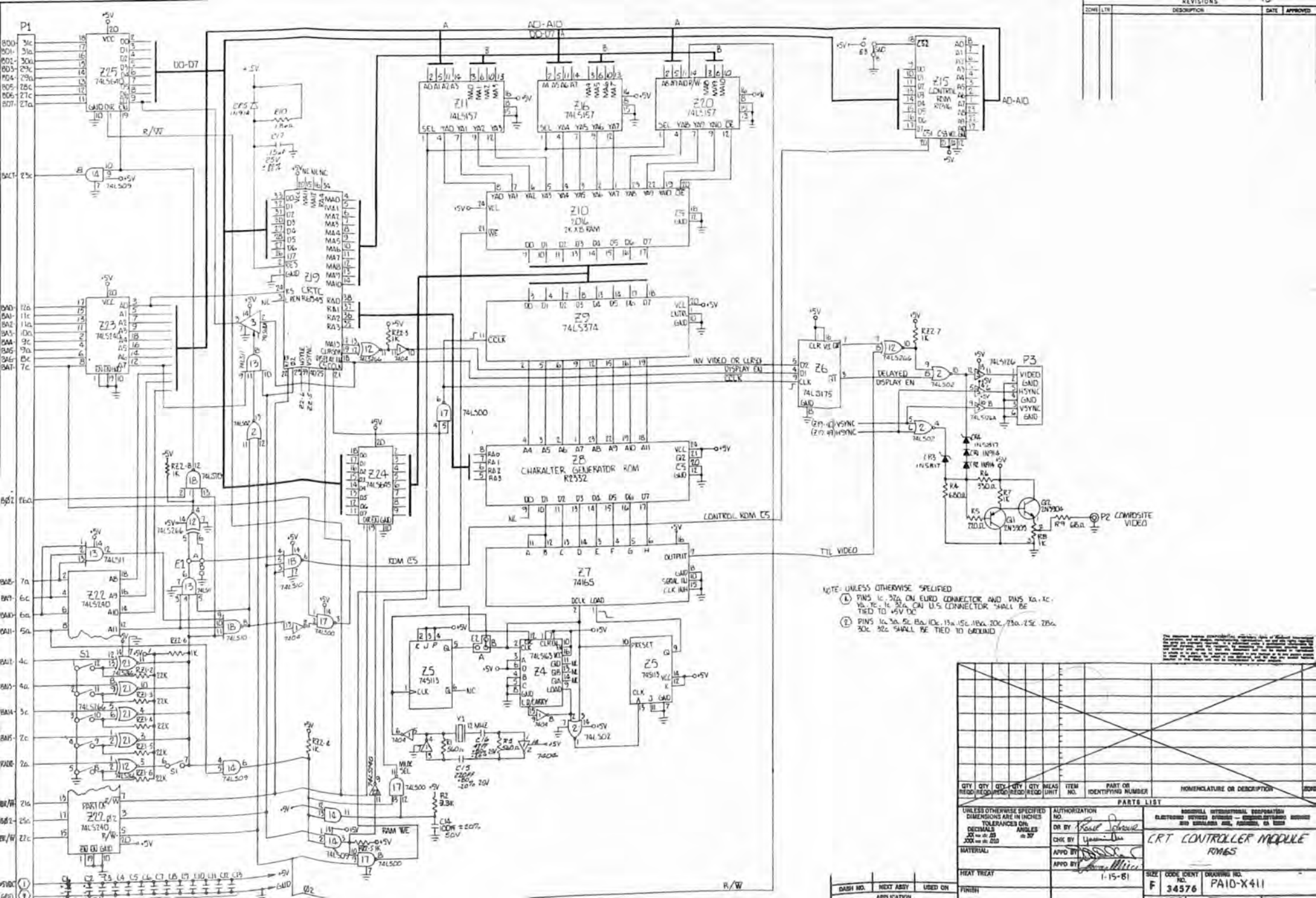
```
C00 00 80 80 80 80 80 80 80 80 00 00 00 00 00 00 00 00
C10 00 C0 40 40 40 40 40 40 C0 00 00 00 00 00 00 00 00
C20 00 E0 20 20 20 20 20 20 E0 00 00 00 00 00 00 00 00
C30 00 F8 10 10 10 10 10 10 F0 00 00 00 00 00 00 00 00
C40 00 F8 08 08 08 08 08 08 F8 00 00 00 00 00 00 00 00
C50 00 FC 04 04 04 04 04 04 FC 00 00 00 00 00 00 00 00
C60 00 FE 02 02 02 02 02 02 FE 00 00 00 00 00 00 00 00
C70 00 FE 00 00 00 00 00 00 FE 00 00 00 00 00 00 00 00
C80 00 10 00 38 44 7C 44 44 00 00 00 00 00 00 00 00
C90 00 44 00 38 44 7C 44 44 00 00 00 00 00 00 00 00
CA0 00 08 10 38 44 78 40 38 00 00 00 00 00 00 00 00
CB0 00 20 10 38 44 78 40 38 00 00 00 00 00 00 00 00
CC0 00 10 28 38 44 78 40 38 00 00 00 00 00 00 00 00
CD0 00 7C 00 58 64 44 44 44 00 00 00 00 00 00 00 00
CE0 00 10 00 38 44 44 44 38 00 00 00 00 00 00 00 00
CF0 00 44 00 38 44 44 44 38 00 00 00 00 00 00 00 00
D00 00 80 80 80 80 80 80 80 80 00 00 00 00 00 00 00
D10 00 C0 40 C0 40 C0 40 C0 C0 00 00 00 00 00 00 00
D20 00 E0 60 A0 60 A0 60 A0 E0 00 00 00 00 00 00 00
D30 00 F0 50 B0 50 B0 50 B0 F0 00 00 00 00 00 00 00
D40 00 F8 58 A8 58 A8 58 A8 F8 00 00 00 00 00 00 00
D50 00 FC 54 AC 54 AC 54 AC FC 00 00 00 00 00 00 00
D60 00 FE 56 AA 56 AA 56 AA FE 00 00 00 00 00 00 00
D70 00 FE 54 AA 54 AA 54 AA FE 00 00 00 00 00 00 00
D80 00 FE AA 54 AA 54 AA 54 FE 00 00 00 00 00 00 00
D90 00 7C 00 44 28 10 28 44 00 00 00 00 00 00 00 00
DA0 00 44 00 44 44 44 44 38 00 00 00 00 00 00 00 00
DB0 00 10 28 00 44 44 44 38 00 00 00 00 00 00 00 00
DC0 00 00 00 00 FE 00 00 00 00 FE 00 00 00 00 00 00
DD0 00 00 00 00 FE 00 00 00 00 00 00 00 00 00 00 00
DE0 10 10 10 10 10 10 10 10 10 10 00 00 00 00 00 00
DF0 10 10 10 10 FE 10 10 10 10 10 00 00 00 00 00 00
E00 00 00 80 80 80 80 80 80 80 00 00 00 00 00 00 00
E10 00 C0 C0 C0 C0 C0 C0 C0 C0 00 00 00 00 00 00 00
E20 00 E0 E0 E0 E0 E0 E0 E0 E0 00 00 00 00 00 00 00
E30 00 F0 F0 F0 F0 F0 F0 F0 F0 00 00 00 00 00 00 00
E40 00 F8 F8 F8 F8 F8 F8 F8 F8 00 00 00 00 00 00 00
E50 00 FC FC FC FC FC FC FC FC 00 00 00 00 00 00 00
E60 00 FE FE FE FE FE FE FE FE 00 00 00 00 00 00 00
E70 00 FE FE FE FE FE FE FE FE 00 00 00 00 00 00 00
E80 10 10 10 10 1E 00 00 00 00 00 00 00 00 00 00 00
E90 10 10 10 10 F0 00 00 00 00 00 00 00 00 00 00 00
EA0 00 00 00 00 1E 10 10 10 10 10 00 00 00 00 00 00
EB0 00 00 00 00 F0 10 10 10 10 10 00 00 00 00 00 00
EC0 10 10 10 10 FE 00 00 00 00 00 00 00 00 00 00 00
ED0 10 10 10 10 1E 10 10 10 10 10 00 00 00 00 00 00
EE0 10 10 10 10 F0 10 10 10 10 10 00 00 00 00 00 00
EF0 00 00 00 00 FE 10 10 10 10 10 00 00 00 00 00 00
```



```
F00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
F10 E0 E0 E0 E0 E0 00 00 00 00 00 00 00 00 00 00 00
F20 1E 1E 1E 1E 1E 00 00 00 00 00 00 00 00 00 00 00
F30 FE FE FE FE FE 00 00 00 00 00 00 00 00 00 00 00
F40 00 00 00 00 00 E0 E0 E0 E0 E0 00 00 00 00 00 00
F50 E0 E0 E0 E0 E0 E0 E0 E0 E0 E0 00 00 00 00 00 00
F60 1E 1E 1E 1E 1E E0 E0 E0 E0 E0 00 00 00 00 00 00
F70 FE FE FE FE FE E0 E0 E0 E0 E0 00 00 00 00 00 00
F80 00 00 00 00 00 1E 1E 1E 1E 1E 00 00 00 00 00 00
F90 E0 E0 E0 E0 E0 1E 1E 1E 1E 1E 00 00 00 00 00 00
FA0 1E 1E 1E 1E 1E 1E 1E 1E 1E 1E 00 00 00 00 00 00
FB0 FE FE FE FE FE 1E 1E 1E 1E 1E 00 00 00 00 00 00
FC0 00 00 00 00 00 FE FE FE FE FE 00 00 00 00 00 00
FD0 E0 E0 E0 E0 E0 E0 E0 FE FE FE 00 00 00 00 00 00
FE0 1E 1E 1E 1E 1E FE FE FE FE FE 00 00 00 00 00 00
FF0 FE FE FE FE FE FE FE FE FE FE 00 00 00 00 00 00
```

NOTES

NOTES


P1
Z15 74LS640
Z11 74LS157
Z16 74LS157
Z20 74LS157
Z15 CONTROL ROM 82S129
Z10 2016 2K X 8 RAM
Z19 6545 CRTC
Z9 74LS374
Z23 74LS244
Z24 74LS645
Z8 CHARACTER GENERATOR ROM 8232
Z6 74LS175
P3
Z22 74LS240
Z7 74165
Z5 74S113
Z4 74LS163
Z27 74LS240
Q1 2N3906
Q2 2N3904
Y1 12 MHz
S1
E1
CONTROL ROM CS
TTL VIDEO
DELAYED DISPLAY EN
P2 COMPOSITE VIDEO
NOTE: UNLESS OTHERWISE SPECIFIED
(1) PINS 1C, 32A ON EURO CONNECTOR AND PINS 2A, 2C, 28A, 28C, 1A, 30A, ON U.S. CONNECTOR SHALL BE TIED TO +5V DC
(2) PINS 2A, 2C, 5C, 8A, 10C, 13A, 15C, 18A, 20C, 23A, 25C, 28A, 30C, 32C SHALL BE TIED TO GROUND
PARTS LIST
CRT CONTROLLER MODULE RM65
PA10-X411
34576
1-15-81

ELECTRONIC DEVICES DIVISION REGIONAL ROCKWELL SALES OFFICES

HOME OFFICE

Electronic Devices Division
Rockwell International
4311 Jamboree Road
Newport Beach, California 92660

Mailing Address:
P.O. Box C
Newport Beach, California 92660
Mail Code 501-300
Tel: 714-833-4700
TWX: 910 591-1698

UNITED STATES

Electronic Devices Division
Rockwell International
1842 Reynolds
Irvine, California 92714
(714) 833-4655
ESL 62108710
TWX: 910 595-2518

Electronic Devices Division
Rockwell International
921 Bowser Road
Richardson, Texas 75080
(214) 996-6500
Telex: 73-307

Electronic Devices Division
Rockwell International
10700 West Higgins Rd., Suite 102
Rosemont, Illinois 60018
(312) 297-8862
TWX: 910 233-0179 (RI MED ROSM)

Electronic Devices Division
Rockwell International
5001B Greentree
Executive Campus, Rt. 73
Marlton, New Jersey 08053
(609) 596-0090
TWX: 710 940-1377

FAR EAST

Electronic Devices Division
Rockwell International Overseas Corp.
Itohpia Hirakawa-cho Bldg.
7-6, 2-chome, Hirakawa-cho
Chiyoda-ku, Tokyo 102, Japan
(03) 265-8806
Telex: J22198

EUROPE

Electronic Devices Division
Rockwell International GmbH
Fraunhoferstrasse 11
D-8033 Munchen-Martinsried
West Germany
(089) 857-6016
Telex: 0521/2650 rimd d

Electronic Devices Division
Rockwell International
Heathrow House, Bath Rd.
Cranford, Hounslow,
Middlesex, England
(01) 759-9911
Telex: 851-25463

Electronic Devices
Rockwell Collins
Via Boccaccio, 23
20123 Milano, Italy
498.74.79
Telex: 202/82

YOUR LOCAL REPRESENTATIVE

10/82