Framtek

Programming Manual

9000 Series
RM 9100, 9200, 9300
Graphic Display System

ramtek

RAMTEK

RM-9000

PROGRAMMING MANUAL

RAMTEK
585 North Mary Avenue
Sunnyvale, CA. 94086

March, 1977

Ramtek

NOTICE

Ramtek Corporation has prepared this manual for use by Ramtek personnel, licensees and customers. The information contained herein is the property of Ramtek and shall neither be reproduced in whole or in part without Ramtek prior written approval.

Ramtek reserves the right to make changes without notice in the specifications and materials contained herein, and shall not be responsible for any damages (including consequential) caused by reliance on the materials presented, including but not limited to typographical, arithmetic, or listing errors.

framtek

TABLE OF CONTENTS

famtek

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

Famtek

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

famtek

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

framtek

SECTION I

INTRODUCTION

1-0 INTRODUCTION TO DISPLAY CONCEPTS

Some rudimentry knowledge of basic display principals is essential to understanding the RM-9000 display system. It must be understood that the image viewed upon a cathode ray tube is produced by a maneuverable, self-contained electron gun (or guns) which emits a beam of specified intensity in order to excite phosphors painted upon the inner face of the tube. Due to the short persistence of the phosphors, the image must be continually repeated (refreshed) in order to minimize phosphor decay therefore visible flicker. The electron beam either directly traces the image being produced or indirectly scans the entire screen while being turned ON and OFF at predetermined times. This second technique is known as raster scan and is employed by the RM-9000 as well as the broadcast television industry. Although somewhat limiting the resolution of the image, the raster scan technique does not constrain the image to a determined number of vectors as does the beam steering technique, nor does it preclude the implementation of the full color spectrum. Since the screen is reduced to a finite matrix of dots, phosphors representing the three primary colors (red, green and blue) can be painting into each dot area (triad) and separate electron guns can be energized to excite each of the phosphors. The relative excitation of the phosphors determines the emitted color.

Because the CRT cannot remember its image, the RM-9000 stores this information within its self-contained, solid-state refresh memory system. The image is stored in binary and is accessed by the video generator at the television raster rate. One or more bits of information describe each picture element (dot), i.e., three bits might be used to describe the discrete excitation of each of the primary phosphors, thus giving seven colors plus black. Additional colors (or grey-scale) are achieved by way of additional memory and digital-to-analog conversion of a prescribed number of bits per picture element, i.e., two bits provides for three intensity levels plus black. Pseudo color is achieved by way of a random access table which is indexed by the binary weighted value of each element as described within the refresh memory. This feature provides for the arbitrary equation of desired color (or grey scale level) and any particular refresh memory bit pattern.

It is the function of the RM-9000 display system to interpret data from the host computer in a specified binary format and compose an image in refresh memory. The internal structure of the RM-9000 allows the refresh memory to be loaded by the internal processor at the same time that it is being accessed by the video generator hardware. Since the refresh memory is truly random-access in nature, there is no internal delay caused by the image storage mechanism, as in previous technologies such as disk-based refresh memory storage.

1-1 SYSTEM OVERVIEW

The overall display system in the normal implementation consists of four major components: the host processor (computer), the computer interface, the RM-9000 display processor and refresh memory, and the display monitor or CRT. The interaction of these four system components produces the final image which can then be interpreted and analyzed by the human observer.

The responsibility of the host processor is to construct the RM-9000 format binary display instructions. The construction of a particular set of display instructions is usually initiated and defined by interaction with some external stimulus, perhaps a human being or another computer system.

The responsibility of the computer interface is to accept RM-9000 format binary instructions and transfer this information to the RM-9000 display processor hardware. This transfer is done on a 16-bit word-per-transfer basis. When an instruction passed to the RM-9000 dictates a change in direction of data transfer (i.e., from the RM-9000 display system to the host processor), the host processor can condition the computer interface to perform this function. In general, the computer interface actually consists of two interfaces: the interface within the computer mainframe which has access to memory within the host processor, and the RAMTEK-supplied module which is resident in the RM-9000 chassis, which converts the data transfer handshaking signals into their equivalent in the specific host-processor interface.

The responsibility of the RM-9000 display processor is to interpret the binary information transferred from the host processor, through the computer interface, and perform the display setup or generation of actual display data to refresh memory. The actual storage of information into refresh memory can either be performed directly by the display processor (as in the case of graphic or text data), or the display processor can setup and initiate a high-speed DMA transfer directly from the computer interface to the refresh memory. An integral part of the display processor is the video generator module. The video generator accesses refresh memory directly and, dependent on the type of video generator, converts the binary data stored there for each pixel to a color or grey scale and sends this information to the monitor.

The responsibility of the display monitor is to accept analog information from the video generator and excite the phosphor on the screen face, thereby producing an image which visually presents information to an observer. Standard raster scan refresh techniques are used, resulting in a refresh rate of 60 or 30 Hz dependent on system line resolution.

1-2 SCREEN COORDINATE SYSTEM

The RM-9000 refresh memory is organized as a three-dimensional coordinate system with its origin at the upper-left corner of the display screen. If one thinks in terms of the classical X-Y-Z coordinate system, the X dimension represents the element coordinate value, the Y dimension represents the line coordinate value, and Z dimension represents the binary value which is stored for each pixel (or picture element), i.e., which is stored for each unique X-Y coordinate pair (See Figure 1-1).

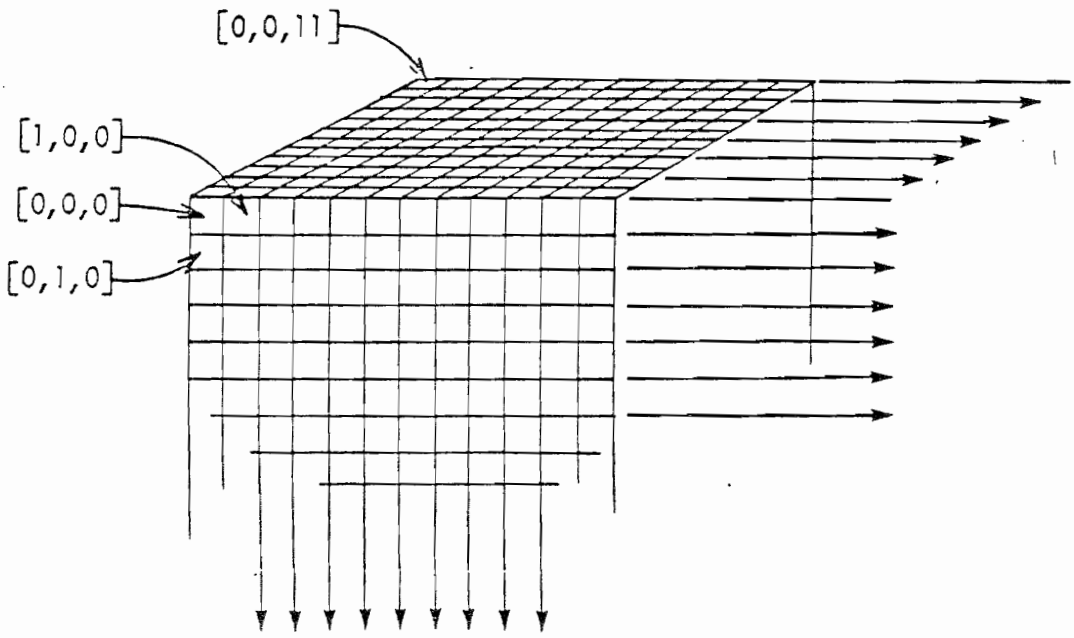


Figure 1-1 Classical X-Y-Z Coordinate System

Addressing in the Y-dimension is modulo 256_{10} or 512_{10} depending on system line resolution. Therefore, in the Y dimension, all coordinates wrap directly from the bottom of the screen back to the top of the screen.

Addressing in the X-dimension is modulo 320_{10} or 640_{10} depending upon system element resolution. Since the possible element resolution values (320_{10} and 640_{10}) are not powers of 2, if an element address is specified by the user in a display instruction is outside of the existing element address space, that element coordinate will be translated into pixels 318_{10} or 319_{10} dependent on the low-order bit of the non-existent address in a low-resolution element system, or into pixels 636_{10}, 637_{10}, 638_{10}, or 639_{10}, dependent on the low-order 2 bits of the non-existent address in a high-resolution element system.

1-3 DISPLAY DATA TYPES

Data which is transferred from the host processor to the RM-9000 display system can be classified according to the following functional types:

1) Image Data
2) Text Data
3) Raster Data
4) Graphics Data

These four (4) data types categorize all data which is transferred to the RM-9000 via a normal-format instruction and which either directly or indirectly is transferred into display refresh memory. Image data is transferred directly into refresh memory; text, raster and graphics data are transferred indirectly in the sense that the data is interpreted by the RM-9000 microprocessor firmware and data is generated to refresh memory as a result of this interpretation.

1-4 Image Data

Image data is loaded directly into refresh memory using the WRITE IMAGE normal-format instructions (See Section 3-36). The low-order 12 bits of each 16 bit image data word are loaded into the refresh memory associated with successive picture elements (or pixels) in the RM-9000 system, i.e., one image data word is written to one display pixel. Image data is always written into a rectangular region within display refresh memory. This region or "window" is defined by a parameter operand called WINDOW (See Section 3-20) which may be set in any normal-format instruction (See Section 3-4). The pixel-to-pixel updating direction for successive words of image data is defined by the parameter operand SCAN; (See Section 3-21). This parameter operand also defines the action to be performed at the window boundaries.

Image data in standard configurations defines the displayed image in one of two ways. In a Type I video standard configuration, (See Section 2-6) the image data is partitioned into three (3) 4-bit sections and used to drive three (3) 4-bit D/A converters directly to the color CRT monitor. Additionally, each subchannel (i.e., bit) of refresh memory can be displayed separately to a black and white CRT monitor. In a Type II video standard configuration (See Section 2-6), the low-order ten (10) subchannels are used as an address into a video look-up table (VLT) which contains the color definition information. Thus, any pixel which has the same data value in the low-order ten (10) subchannels will be represented by the color or greyscale value which is stored in the VLT at the address defined by the pixel value.

1-5 Text Data

Text data is transferred to the RM-9000 display system on a 2-bytes per word basis, using the WRITE TEXT normal-format instruction (See Section 3-38). Each byte of data is interpreted as an eight (8) bit ASCII code and the character font associated with the transmitted ASCII code is written into refresh memory. In a standard system (i.e., in a system without the RM-FNT option), the character font data is obtained internally from a standard PROM. The character font format is defined as a 5 pixel wide by 7 pixel high character within a 7 pixel by 9 pixel rectangle. The valid codes for a standard configuration are 20_{16} through $5F_{16}$.

Font definitions from the internal PROM are in the form of a 'ones/zeros' dot matrix. Two (2) RM-9000 internal hardware registers are used to define the color or greyscale intensity to be written into refresh memory for ones or zeros data. These are the foreground and the background registers. These values can be user-specified using the FOREGROUND and BACKGROUND parameter operands (See Sections 3-5 and 3-16). It is possible to reverse the interpretation of ones and zeros font data using the reverse-background flag in a normal-format instruction (See Section 3-9). Also, the additive write flag allows the user to specify that only ones data will be written to refresh memory (See Section 3-10).

Text data is written into refresh memory on a 'windowed' basis, i.e., characters will only be written into the rectangular region defined by the WINDOW parameter operand (See Section 3-20). The character-to-character update direction and the window margin update direction is defined by the parameter operand SCAN (See Section 3-21).

1-6 Raster Data

Raster data is transferred to the RM-9000 display system via the WRITE RASTER normal-format instruction which is part of the RM-GRA option package (See

Section 3-50). Raster data is interpreted as 'ones/zeros' data in the same manner as character font data is interpreted by the internal microprocessor. Color or grey-scale information for ones or zeros data is also defined by the foreground and background basis. Eight (8) pixels are written into by each byte of raster data, on a one (1) bit per pixel basis.

Raster data is written into refresh memory on a 'windowed' basis. The definition of this rectangular region is through the user-defined WINDOW parameter operand. Within this region, the SCAN parameter operand defines the pixel-to-pixel and the window margin update directions (See Section 3-21).

1-7 Graphics Data

Graphics data is transferred to the RM-9000 display system via the WRITE VECTOR or WRITE PLOT normal-format instructions which are part of the RM-GRA option firmware (See Sections 3-48 and 3-49). For both vector and plot generation, the graphic data specifies an endpoint coordinate which defines the vector or plot entity to be generated. For vectors, the endpoint of the previous vector is used with the current endpoint data to define the vector; for plots, the rectangular bar plot entity is defined by the previous plot entity coordinate and the current endpoint data. Windowing is not effective in graphics data mode.

1-8 DISPLAY INTERACTION

The RM-9000 display system can be configured such that interaction with the host processor through several devices is possible. Up to eight (8) keyboards or up to four (4) joysticks or trackballs are possible in a system.

1-9 Cursors/Joysticks/Trackballs

A joystick or trackball is a device which when connected to the RM-SLC serial link card can automatically change the position of a visible cursor on the display CRT monitor. This updating based on user interaction with the device is strictly a hardware function. It is possible to use the joystick or trackball to interrupt the host processor through depression of a momentary-action switch (labelled ENTER) on the device, or whenever the position of the display cursor is changed. This interrupt can be used by the host processor to signal some action to be taken perhaps based on the position of the cursor. This interpretation is completely flexible based on the needs of the host processor. Appendix 'C' defines the operational use of the joystick and trackball interactive devices.

1-10 Keyboards

The keyboard is used to transmit eight (8) bit ASCII codes from the RM-9000 display system to the host processor. The keyboards are attached to the RM-SLC serial link card and are handled by the RM-PER interactive peripheral option firmware. Each keyboard is buffered on input up to sixteen (16) characters. The entry of a character via the keyboard will generate an interrupt to the host processor if the interrupt has been enabled at the interface by the host processor. The keyboard allows user interaction via text input.

Famtek

SECTION II

FUNCTIONAL DESCRIPTION

2-0 RM-9000 FUNCTIONAL DESCRIPTION

Figure 2-1 illustrates the major functional modules of the Model RM-9100, 9200 and 9300 display systems. The following is a brief description of each system component (for a detailed description of all hardware components consult the RM-9000 Theory of Operation - Volume I):

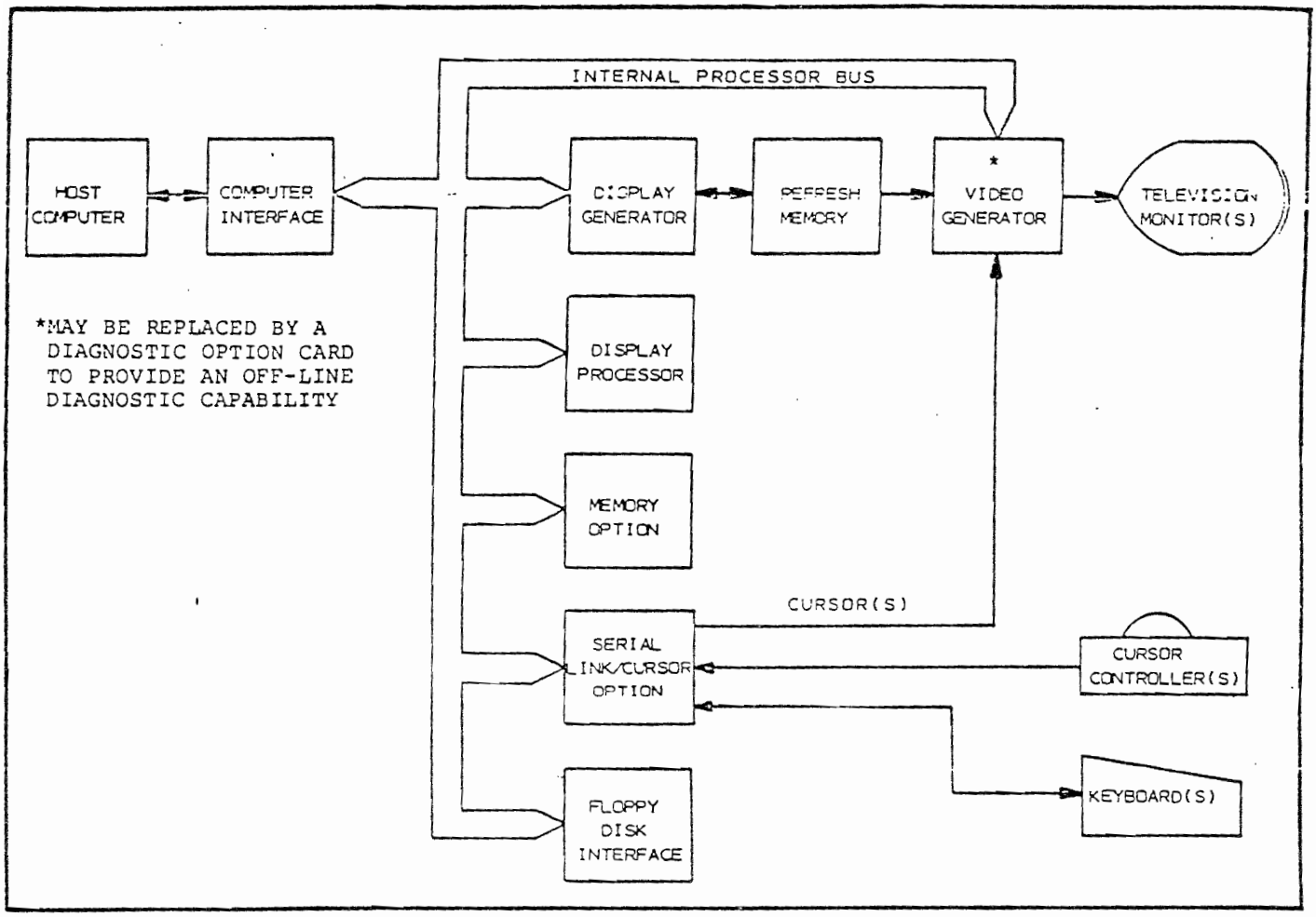


Figure 2-1 RM-9100, 9200 & 9300 Functional Block Diagram

2-1 COMPUTER INTERFACE

The RAMTEK computer interface is specific to the model of host processor used. It connects the host processor with the RM-9000 display system internal processor with the RM-9000 display system internal processor bus. A high-speed bidirectional, sixteen (16) bit parallel-bus communication path between host processor and the RM-9000 is provided. Four (4) external interrupt lines (illegal instruction, receiver, transmitter and cursor) can be generated by the RM-9000 display generator through the computer interface to the host processor. All interrupts have an enable/disable flag which is under host processor control.

Four (4) types of data transfers through the computer interface are possible:

1) Interface command word output to the RAMTEK interface
2) Interface status word input from the RM-9000 interface to the host processor
3) RM-9000 instruction/data output from the host processor to the RM-9000
4) RM-9000 data input from the RM-9000 to the host processor.

The interface command word allows the host processor to initiate such interface functions as enabling or disabling of interrupts and/or hard system resetting. This command word is strictly an interface interaction and has no RM-9000 interaction (except for hard system reset which causes an immediate reset sequence to be initiated). The interface status word defines to the host processor the status of the four (4) possible interrupts and their enable/disable functions as well as the I/O data transfer status. The instruction/data output mode is used to transfer instructions and data through the computer interface to the RM-9000 display system microprocessor. These instructions and data are in the binary formats described in Section 3 and perform the actual display generation functions. The RM-9000 data input mode is used to transfer data from the RM-9000 to the host processor after receiving one (1) of the readback initiating instructions from the host processor (RI, READ, RCS, RKB, RCM and RCMRP). This readback is always initiated by the host processor. See the Addenda to this manual for the description of computer procedures for each specific host processor interface.

2-2 CONTROL BOARD

The RM-9000 control board is the component in every RM-9000 display system which contains the 8080 microprocessor and data bus, the standard RM-9000 operating firmware, as well as the display generator hardware. The internal data exchange (IDE) bus connects the control board (and thus the 8080 microprocessor data bus) with the computer interface.

2-3 8080 Microprocessor

The 8080 microprocessor contained on the control board is the principal device used to control the generation of display data to the refresh memory. The 2048 bytes of PROM resident on the control board contain 8080 microprocessor instructions used to interact with the computer interface and refresh memory using the display registers in the display generator hardware. The 8080 microprocessor 8-bit data bus is interfaced to the 16-bit RM-9000 internal processor bus through logic on the control board. It is this internal processor bus which connects all components within the RM-9000 system to the computer interface IDE bus.

2-4 Display Generator

The display generator is that portion of logic contained on the control board which provides the mechanisms for the optimized transfer of image, graphic, raster and text data to refresh memory and for the retrieval of refresh memory data in image mode on a pixel basis.

The display generator contains registers accessible to the 8080 microprocessor (in the address range 8000_{16} through $80FF_{16}$). The display generator allows 8080 firmware to perform such internal functions as refresh memory plane selection, raster/text data pattern definition, refresh memory addressing, initiation of such DMA operations as image input/output, video lookup table input/output, and internal generation of text data, and interrupt generation to the computer interface.

2-5 REFRESH MEMORY

The display refresh memory contains storage for 1 to 12 bits of data per picture element in the display system. These memories may be loaded or read directly from or to the host processor (in imaging mode via DMA across the computer interface), or they may be loaded on a pixel-by-pixel basis via the display generator in graphics, text or cartesian mode. Access time to these memories is 1.5 μ/pixel. The element/line resolution combinations for these memories is as follows:

- RM-9100 - 320 elements X 256 lines
- RM-9200 - 640 elements X 256 lines
- RM-9300 - 640 elements X 512 lines

It is these memories which supply color or intensity information for each pixel to the various video generator modules. The pixel information is constantly refreshed to the video generator(s) either at 60 Hz for the RM-9100 and RM-9200 Systems or 30 Hz for the RM-9300. This refresh process is totally under hardware control and no display processor intervention is necessary or possible.

2-6 VIDEO GENERATOR

The video generator combines information from the refresh memory and cursor generators and produces standard RS-170 compatible composite video signals to the television monitor(s). Each video generator combines and mixes this information according to different functional algorithms (in some cases, under host processor or display processor control). The following is a description of the two basic video boards:

(a) RM-V1 - Figure 2-2 functionally illustrates the RM-V1 video board which provides for twelve (12) B/W channels, four (4) 7 color (RGB) channels, three (3) 16 level grey scale channels, one (1) 256_{10} level grey scale channel or one (1) 4096_{10} color (RGB) channel. It includes twelve (12) direct outputs plus three (3) 4-bit DAC's (or one (1) 8-bit and one (1) 4-bit DAC), four (4) cursor channels and two (2) overlay channels.

(b) RM-V2 - Figure 2-3 functionally illustrates the RM-V2 video board which provides for host programmable pseudo color or grey scale translation to any 4096_{10} colors or 256_{10} grey scale levels. It includes one (1) 1024_{10} word X 12-bit programmable function memory plus three (3) 4-bit DAC's, one (1) 8-bit DAC (Assignable before or after the function memory), two (2) cursor channels and two (2) overlay channels.

2-7 MEMORY EXPANSION BOARD (RM-MOC)

The RM-MOC memory expansion board is an optional system component which allows the installation of the RAMTEK firmware options packages. It provides addressing for 14336_{10} bytes of PROM (addresses 0800_{16} through $3FFF_{16}$) and 8192_{10} bytes of RAM (addresses 4000_{16} through $5FFF_{16}$). Hardware logic is provided supporting DMA input and output between memory expansion RAM and interface, refresh memory, or video look-up table RAM.

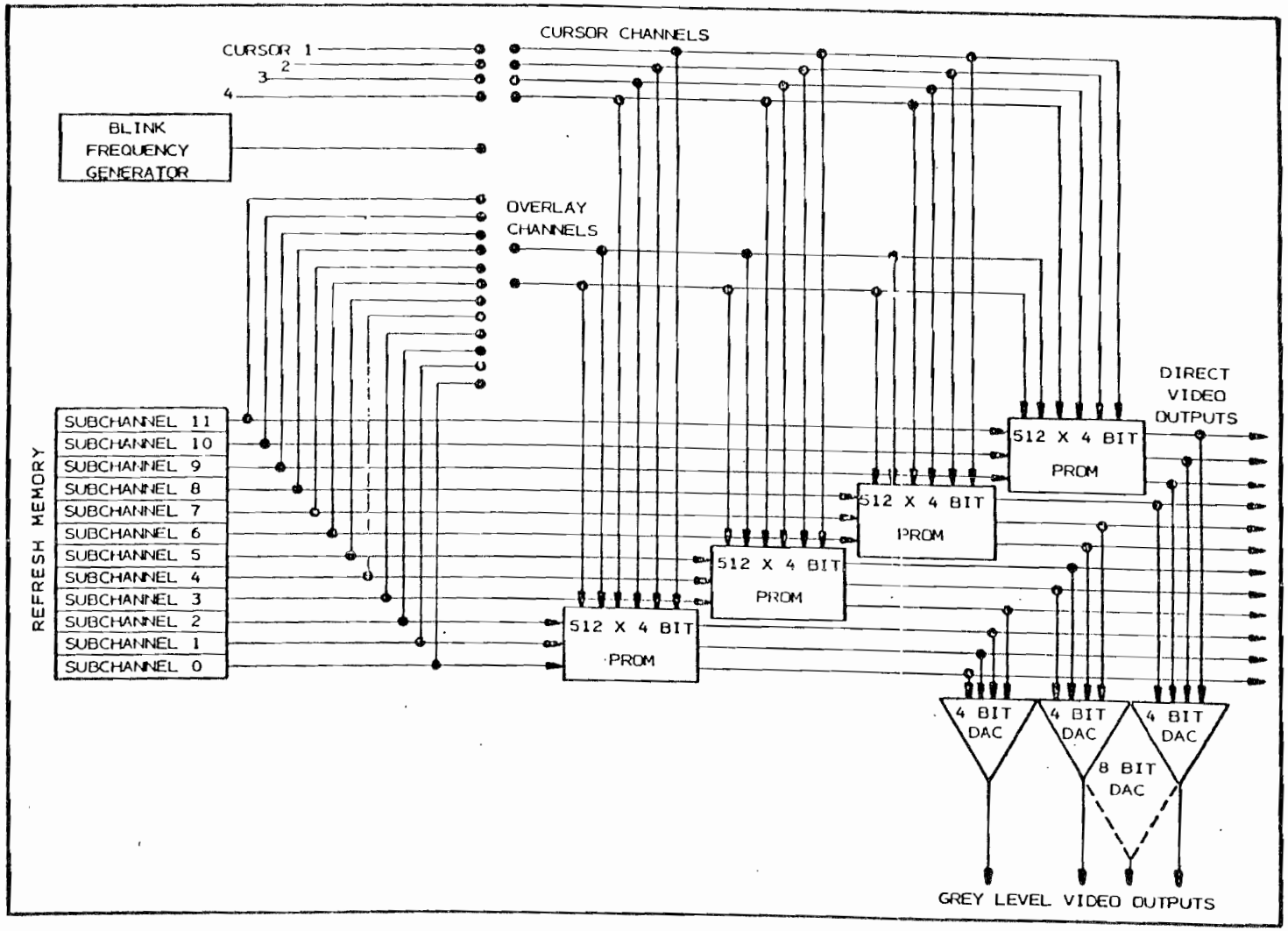


Figure 2-2 RM-V2 Functional Block Diagram

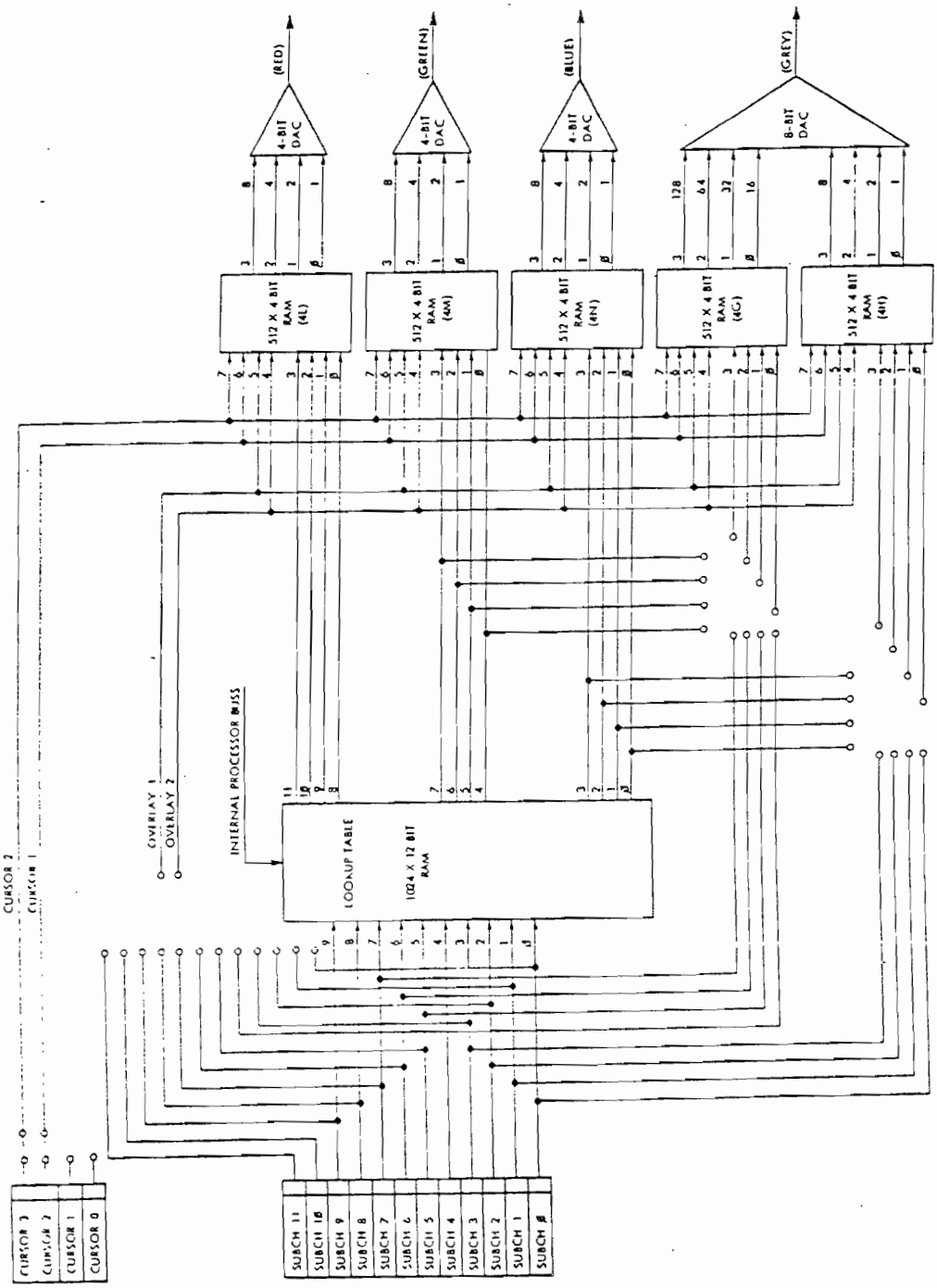


Figure 2-3 RM-V1 Functional Block Diagram

2-8 SERIAL LINK BOARD (RM-SLC)

The RM-SLC serial link board provides system communication with external interactive devices such as keyboards and joysticks or trackballs over serial communication ports. Each serial link card also can generate up to two video cursors. This cursor is mixed into the video generator card output for display. Each serial link card can accommodate either four (4) keyboard/transmitters or two (2) keyboard/transmitters and two (2) cursors. A maximum of 2 RM-SLC serial peripherals firmware packages provides the firmware mechanism for host processor communication with these interactive peripheral devices.

Famtek

SECTION III

RM-9000 INSTRUCTION SET

SECTION III

RM-9000 INSTRUCTIONS

3-0 INTRODUCTION

The RM-9000 series is a microprocessor-controlled graphic display system. The standard firmware instruction set provides a high-level mechanism for the storage and retrieval of image data and associated image-generation information at high-data rates, as well as the generation of alphanumeric text information. This instruction frees the user from the complex tasks of communication directly with the hardware display registers. The standard firmware package provided by RAMTEK reduces significantly the amount of software display processing in the host computer.

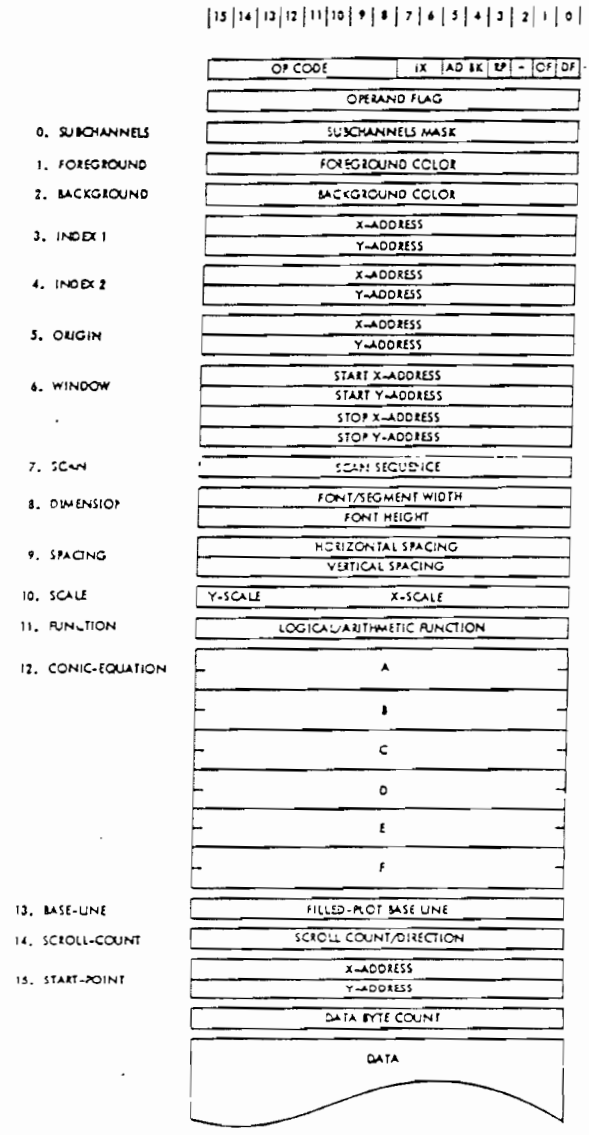
3-1 INSTRUCTIONS

The RM-9000 provides a set of standard instructions and a variety of optional instructions. This manual will cover all currently defined instructions. Since the RM-9000 is a very flexible system, the set of optional instructions will probably continue to increase. Addenda to this manual will be provided for additional instruction sets.

The RM-9000 instruction consists of two basic types of instructions, normal format and special format instructions. Normal format instructions constitute a set of high-level imaging, text and graphics functions, in a flexible, yet uniformly-defined format. The normal format instruction set reduces the user's programming effort with an attempt to optimize high-speed image and text generation. The special format instructions are used to perform functions which do not affect the display directly (i.e., do not write into or read from refresh memory). The format of the special instructions is different for each instruction.

The RM-9000 will generate an illegal instruction interrupt request whenever an optional instruction is referenced when the associated firmware option is not installed. The same interrupt request will also be generated when a truly undefined op code is referenced.

Currently defined optional instructions may be added at a later date. The standard software will sense the presence of the option PROM and reference it when



LEGEND

IX = ADDRESSING MODE (0 = ABSOLUTE, 1 = INDEX-1, 2 = INDEX-2, 3 = RELATIVE)

AD = ADDITIVE WRITE (0 = REPLACEMENT, 1 = ADDITIVE)

BK = REVERSE BACKGROUND (0 = NORMAL BACKGROUND, 1 = REVERSED BACKGROUND)

RP = REVERSE PACKING FLAG (0 = LEFT BYTE FIRST, 1 = RIGHT BYTE FIRST)

OF = OPERAND FLAG (0 = NO ARGUMENTS OR FLAG WORD EXISTS, 1 = FLAGGED ARGUMENTS EXIST)

DF = DATA FLAG (0 = NO DATA OR LENGTH WORD EXISTS, 1 = n DATA BYTES

Figure 3-1 Normal-Format Instruction Layout

it is installed. The installation of currently undefined option software will also require replacement of the standard option PROM's to enable the addition of the new op codes.

The instruction repertoire for the RM-9000 is listed in Table 3-1.

3-2 INSTRUCTION FORMAT

All instructions are 16-bit parallel and consist of one (1) or more 16-bit words. The initial 8-bits (most significant) of the initial word always describe the operation to be performed by the instruction. The remaining bits (and words) may be interpreted differently for each instruction.

The RM-9000 has a "normal" instruction format and a "special" instruction format. The normal format is specifically designed for the general requirements of imaging and graphics. The special format instructions are oriented to the individual special requirements of each instruction.

The special format may be used to reduce data flow across the interface or to handle unique data requirements. Each special format will be described with the instruction itself.

3-3 RM-9000 DATA MODES

Data is stored in the RM-9000 refresh memory in one of two data modes: image mode or raster mode. The RM-9000 display controller always writes a data value of up to twelve (12) data bits per pixel into refresh memory; the origin of this data determines the data mode. In image data mode, the low-order 12-bits of a 16-bit data word are stored in up to twelve (12) subchannels of refresh memory for a single pixel. This storage is only done in those refresh memory subchannels which are selected by the subchannel mask parameter. Therefore, in image mode, it is possible to store any 12-bit value in any pixel in refresh memory. Since the data to be loaded is externally generated, the FOREGROUND and BACKGROUND parameters which are crucial to raster data mode are not used. In raster data mode, only the FOREGROUND or BACKGROUND values are written into a pixel location. Each bit of raster data represents a separate pixel, and the value of each raster bit selects whether the FOREGROUND or BACKGROUND value will be written into the represented pixel. The reverse-background flag BK influences raster data mode in that when BK=1, the polarity of incoming raster data is reversed. The additive-write flag AD is also used in raster data mode to prevent the writing of raster data with a zero bit value. This is useful in writing text

Famtek

Table 3-1 RM-9000 Instruction Repertoire

OP-CODE	INSTRUCTION NAME	MNEM-ON	KEY		CONTROL FLAGS						PARAMETER FLAGS (NORMAL INSTRUCTIONS ONLY)														
			F	O	DF 0	OF 01	RP 03	BK 04	AD 05	IX 6,7	COP 15	SCR 14	BAS 13	CON 12	LAF 11	SCL 10	SPC 9	DIM 8	SCN 7	WIN 6	ORG 5	IX2 4	IX1 3	BGD 2	FGD 1	MSK 0
00	(UNDEFINED)	–	–	–																					
01	LOAD HARD REGISTER	LOAD	S	–																					
02	READ SOFT REGISTER	READ	S	–																					
03	LOAD AUXILIARY MEMORY	LAM	S	–																					
04	READ AUXILIARY MEMORY	RAM	S	–																					
05	RESET	RSET	S	–							I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
06	INITIALIZE	INIT	S	–							I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
07	NO OPERATION	NOP	N	–	S	S	S	S	S	S	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
08	SET PARAMETER	SET	N	–	S	X	S	S	S	X	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S
09	ERASE	ERS	N	–	S	X	S	X	S	X	S	S	S	S	S	S	S	S	S	X	X	X	X	X	X
0A	WRITE IMAGE	WI	N	–	X	X	S	S	S	X	X	S	S	S	X	X	S	S	X	X	X	X	X	S	X
0B	READ IMAGE	RI	N	–	X	X	S	S	S	X	X	S	S	S	S	S	S	S	X	X	X	X	X	S	S
0C	WRITE TEXT	WT	N	–	X	X	X	X	X	X	X	S	S	S	S	X	X	X	X	X	X	X	X	X	X
0D	WRITE RASTER	WR	N	G	X	X	X	X	X	X	X	S	S	S	S	X	S	S	X	X	X	X	X	X	X
0E	WRITE VECTOR	WV	N	G	X	X	S	X	S	X	X	S	S	S	S	S	S	S	S	S	X	X	X	X	X
0F	WRITE CONIC	WC	N	C	X	X	S	X	S	X	X	S	S	X	S	S	S	S	S	S	X	X	X	X	X
10	WRITE PLOT	WP	N	G	X	X	S	X	S	X	X	S	X	S	S	S	X	X	X	S	X	X	X	X	X
11	SCROLL X	SCRX	N	S	X	X	S	X	S	X	S	X	S	S	S	S	S	S	X	X	S	S	X	X	X
12	SCROLL Y	SCRY	N	S	X	X	S	X	S	X	S	X	S	S	S	S	S	S	X	X	S	S	X	X	X
13	SAVE ENVIRONMENT	PUSHE	S	M											?	?	↑	↑	↑	↑	↑	↑	↑	↑	↑
14	RESTORE ENVIRONMENT	POPE	S	M											?	?	↓	↓	↓	↓	↓	↓	↓	↓	↓
15	LOAD PROGRAMMABLE FONT	LPF	S	F																					
16	WRITE CURSOR STATE	WCS	S	P																					
17	READ CURSOR STATE	RCS	S	P																					
18	WRITE KEYBOARD	WKB	S	P																					
19	READ KEYBOARD	RKB	S	P																					
1A	SENSE PERIPHERAL STATUS	SPS	S	P																					
1B	LOAD CONTROL MEMORY	LCM	S	U																					
1C	READ CONTROL MEMORY	RCM	S	U																					
1D	CALL CONTROL MEMORY	CCM	S	U																					
1E	EXECUTE INSTRUCTION MEMORY	XIM	S	U																					
1F	LOAD CONTROL MEMORY (REV)	LCMRP	S	U																					
20	READ CONTROL MEMORY (REV)	RCMRP	S	U																					
21	LOAD PROGRAMMABLE FONT (REV)	LPFRP	S	F																					

KEY LEGEND

FORMAT TYPES

N NORMAL

S SPECIAL

OPTION PRE-REQUISITES

P INTERACTIVE PERIPHERALS

G GRAPHICS

S SCROLL

M STATUS MANAGEMENT

F PROGRAMMABLE FONT

U USER-SUBROUTINE

C CONICS + GRAPHICS

S => SET/DEFINE FLAG/PARAMETER

X => USED IN EXECUTING THE INSTRUCTION

– => NO EFFECT, NO OPERATION PERFORMED

↑ => PUSH DATA ON TO STACK

↓ => POP DATA OFF OF STACK

I => INITIALIZE TO DEFAULT VALUES

? => NO INTERNAL BUFFERS SAVED

on to an existing display; only the character (and not its font background) will be stored in refresh memory. Table 3-2 defines the mode used by the various complex instructions for writing data to refresh memory.

Note that these data modes define the mechanism by which data is stored in refresh memory. It is essential that the user understand the implications of each data storage mode in order to effectively use the RM-9000 instruction set.

Table 3-2 RM-9000 Instruction Set

INSTRUCTION	DATA MODE
ERASE	Raster
WRITE IMAGE	Image
READ IMAGE	Image
WRITE TEXT	Raster
WRITE RASTER	Raster
WRITE VECTOR	Raster
WRITE PLOT	Raster

3-4 NORMAL INSTRUCTION FORMAT

The normal instruction format can carry a variety of parameter and data information to the display system. The format also allows a variable amount of parameters and data information. The variable format allows the user to transmit only the information used in a particular operation. This means that a very complex function, such as window erase, can be performed via a single 16-bit word when the window parameters have been previously defined.

Figure 3-1 illustrates the normal instruction format. The first (most significant) byte of the first word defines the operation code as is done in all RM-9000 instructions.

The second (least significant) byte is called the parameter byte and defines the coordinate addressing mode, additive or replacement writing mode, reverse background mode, byte processing order and the presence of operand parameters and/or data.

3-5 Parameter Byte

The fields in the parameter byte are defined as follows:

NAME	BIT POSITION
Data Flag	0
Operand Flag	1
Undefined	2
Reverse Packing	3
Reverse Background	4
Additive Write	5
Addressing Mode	6, 7

3-6 Data Flag (Bit 0)

A Data Flag (DF) value of 1 indicates that a data length word (i.e., the numbers of bytes of data) and the specified number of data bytes will follow any complex parameters that might be set by the complex instruction. A (DF) value of 0 indicates the absence of a data length word or any data following any complex parameters. For a complete description of the data length word and the data format, see Sections 3-30 and 3-31.

3-7 Operand Flag (Bit 1)

An Operand Flag (OF) value of 1 indicates the presence of the operand flag word. An (OF) value of 0 indicates the absence of an operand flag word and correspondingly the absence of any parameter operands. The operand flag word follows the instruction word if present. (See Section 3-11 for a discussion of the operand flag word.)

3-8 Reverse Packing Flag (Bit 3)

The Reverse Packing Flag (RP) specifies the packing mode for byte oriented data. The (RP) flag effects only data (not parameter) and only byte oriented data such as text or raster. A (RP) value of 0 indicates normal packing and a (RP) value of 1 indicates reversed packing.

The normal packing mode for 8-bit bytes in the 16-bit word specifies that bytes are unpacked and processed from left-to-right, i.e., the most significant data byte is processed first. Reverse packing means the order of unpacking is right-to-left, i.e., the least significant byte is processed first.

3-9 Background Flag (Bit 4)

The Background Flag (BK) selects between normal and reverse background for text, raster and graphic commands. Normal background is selected when the (BF) bit is zero and reverse background when (BF) is one.

The RM-9000 uses either the BACKGROUND parameter or the FOREGROUND parameter when generating raster, text and graphic data. The normal mode is for the FOREGROUND value to be selected when a data bit is one (1) and the BACKGROUND value is written to refresh memory when a data bit is zero (0). The reverse background mode reverses the selection of these two parameters just as if the parameters had been exchanged or as if the data bits were reversed. (See raster data section.)

3-10 Additive Flag (Bit 5)

The Additive Write Flag (AD) when set to a one will cause raster, text and cartesian data (i.e., data generated by the ERS instruction) to be written in refresh memory in an "additive" fashion, i.e., data bits with a zero value cause nothing to be written to memory and only data bits with a one (1) value cause data to be written.

An Additive Write Flag value of zero (0) causes "replacement writing of memory, i.e., both zero and one bits will cause memory contents to change.

The user may select combinations of the additive and background flags. Table 3-3 gives the various relationships.

3-11 Addressing Mode Flags (Bit 6, 7)

The normal instruction format permits selection of one of four addressing modes for each instruction. All coordinate information such as window position is modified by the addressing mode selected.

Table 3-3 Functional Relationship of Data, Background Flag & Additive-Right Flag

		AD			
		0	0	1	1
BK	0	B	F	X	F
	1	F	B	X	B
		0	1	0	1

DATA BIT

F WRITE FOREGROUND VALUE
B WRITE BACKGROUND VALUE
X NO WRITE

Additive (AD) - Background (BK) & Data Relation

The addressing mode flags (IX) are absolute (IX = 00), index using Index 1 (IX = 01), index using Index 2 (IX = 10), and relative (IX = 11). The RM-9000 maintains two (2) internal index registers, IX1 and IX2 which are set by the INDEX parameter (See Sections 3-17 & 3-18).

Absolute Addressing

The x, y values in the parameter or data list are used directly as screen coordinates.

Index Addressing

The parameter or data value referencing screen coordinates is added to the index selected to determine the coordinate desired.

Relative Addressing

The parameter or data value referencing screen coordinates is added to the current operating point (last screen coordinate read or written).

The relative addressing mode can be used to create a sequence of vectors, each new endpoint being relative to the termination of the previous.

The index addressing modes allow the creation of a display or image independent of screen position. Note that the index register value used is the previous value defined for that index register, i.e., previous to the complex instruction which is using indexed addressing. Parameter values which refer to screen coordinates (such as WINDOW or START POINT) are computed by performing a 2's complement addition of the parameter value with the selected index register.

The absolute mode provides standard screen addressing capability.

The RM-9000 screen is addressed with a coordinate system having the upper left corner as (0, 0).

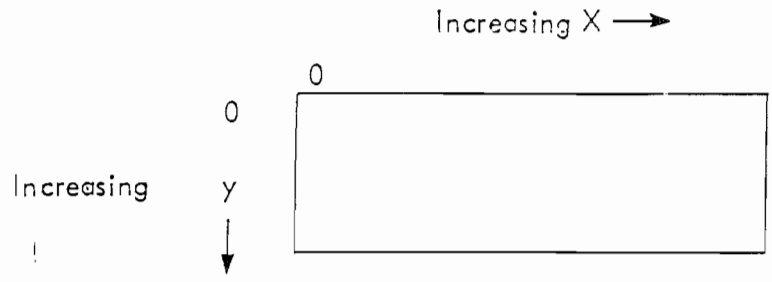


Figure 3-2 Screen Coordinate Orientation

3-12 Operand Flag Word

The Operand Flag Word indicates the presence or absence of the sixteen (16) possible parameters operands for normal instructions. Each parameter operand for a normal instruction must appear in a fixed sequence. Each bit in the Operand Flag Word corresponds to a parameter operand in the same sequence. A zero bit in the same position as a parameter operand indicates the operand is absent. Each bit is interpreted from right-to-left, i.e., Bit 0 corresponds to the first parameter operand and Bit 15 corresponds to the last parameter operand.

3-13 Parameter Operands

The parameter operands are the various internal values which define the operation and subsequent display for all normal instructions. The parameter operands may be set in any RM-9000 normal instruction and affect only the operation of normal instructions. Parameter operand values are non-volatile from normal instruction to normal instruction; once a parameter operand is set by a normal instruction, the parameter operand value remains the same until reset by the user. The presence

BIT	MNEMONIC	FUNCTION
0	MSK	SUBCHANNEL
1	FGD	FOREGROUND
2	BGD	BACKGROUND
3	IX1	INDEX 1
4	IX2	INDEX 2
5	ORG	ORIGIN
6	WIN	WINDOW
7	SCN	SCAN
8	DIM	DIMENSION
9	SPC	SPACING
10	SCL	SCALE
11	LAF	FUNCTION
12	CON	CONIC EQUATION
13	BAS	BASE LINE
14	SCR	SCROLL COUNT
15	COP	START POINT

Figure 3-3 Operand Flag Word Format

of any set or subset of parameter operands is indicated by the state of the 16-bits in the operand flag word (See Section 3-12). The order of appearance of each of the parameter operands is defined by the position of its flag bit in the operand flag word. A parameter operand with a flag-bit position of m will appear before a parameter operand with a flag-bit position of (n), if (m) is less than (n). The number of words associated with each parameter operand is fixed, but this number varies from 1 up to 12_{10} words. The parameter operands are sensitive to the addressing mode bits, but not to any of the other bits in the parameter byte.

famtek

3-14 Subchannel Parameter (No. 0)

	S_{11}	S_{10}	S_9	S_8	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0
15 14 13 12	11	10	9	8	7	6	5	4	3	2	1	0

The SUBCHANNEL parameter may be set via any normal instruction (except NOP). Its presence is flagged via Operand Flag Bit 0. The operand itself is a single 16-bit word and write-enables a prescribed combination of refresh memory bit planes (subchannels) for image generation purposes. There are twelve (12) possible subchannels and twelve (12) corresponding bits in the SUBCHANNEL argument, i.e., Bit 0 corresponds to Subchannel 0, Bit 1 to Subchannel 1, ..., and Bit 11 to Subchannel 11. When set to a "one" state, the corresponding subchannel is enabled, and visa-versa. Disabled subchannels are not affected by write operations (including Erase). The subchannel parameter has no effect during read instructions, e.g., Read Image "a" will respond with a value for all subchannels in the system. Although the RM-9000 is normally configured as a single channel system, the SUBCHANNEL parameter provides for multi-channel partitioning of the refresh memory. For example, the refresh memory might be partitioned into four (4) channels of three (3) subchannels each, with each channel producing a seven (7) color (plus black) image.

SUBCHANNEL default value = OFFF (Hex).

3-15 Foreground Parameter (No. 1)

	S_{11}	S_{10}	S_9	S_8	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0
15 14 13 12	11	10	9	8	7	6	5	4	3	2	1	0

The FOREGROUND parameter may be set via any complex format instruction (except NOP). It's presence is flagged via Operand Flag Bit 2^1. The operand itself is a single 16-bit word in length and is identical in format to the SUBCHANNELS argument. It establishes foreground color or intensity for normal foreground ("one" bits) font, raster and graphics data by assigning a "one" or "zero" for each of the twelve (12) possible subchannels. When writing reversed background character font and raster data, FOREGROUND establishes color or intensity for "zero" state data.

FOREGROUND default value = OFFF (Hex).

3-16 Background Parameter (No. 2)

				S 11	S 10	S 9	S 8	S 7	S 6	S 5	S 4	S 3	S 2	S 1	S 0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The BACKGROUND parameter may be set via any complex format instruction (except NOP). Its presence is flagged via Operand Flag Bit 2. The operand itself is a single 16-bit word in length and is identical in format to the SUB-CHANNELS argument. Interpretation is identical to the FOREGROUND argument except that background color or intensity is specified, i.e., the color of normal background, "zero" state font or raster data. When background is reversed, data interpretation is likewise reversed. That is, BACKGROUND then specifies the color or intensity of "one" state character font, raster or graphics data.

BACKGROUND default value = 0000 (Hex).

3-17 Index 1 Parameter (No. 3)

X - Address															
Y - Address															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The INDEX 1 parameter may be set via any complex format instruction (except NOP). Its presence is flagged via Operand Flag Bit 3. The operand itself is two 16-bit words in length and specifies a local addressing origin (or displacement). The first word specifies X address or horizontal displacement from element 0, whether positive or negative. The second word likewise specifies Y address or vertical displacement from line 0. Received coordinate values (X/Y) in subsequent normal instructions and in subsequent parameters in the current normal instruction are conditionally summed with the current INDEX 1 specified values in order to derive the effective (refresh memory) address, i.e., provided that Index Register 1 addressing was specified in the first word of the received command (IX = 01). If IX $\neq$ 0, the values loaded into the X and Y components of IX1 are the sum of current X and Y components of the specified registers and the X and Y

components of the operand. Therefore, IX = 01 causes IX1 to be the sum of the old contents and the new operand values.

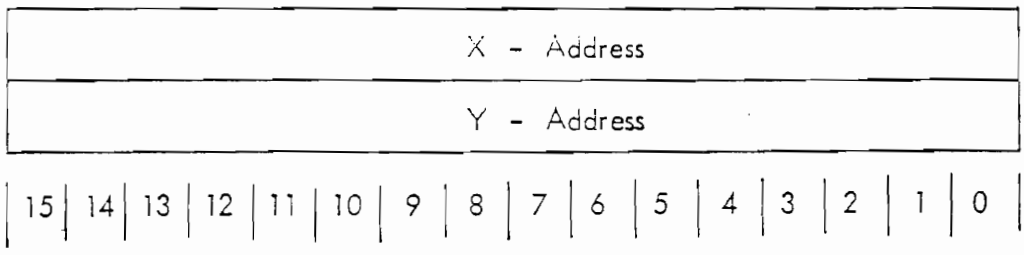
The actual values used for IX1 (and IX2) are computed as follows:

- If IX = 0 (absolute addressing mode), the actual received argument values for IX1 (both X and Y) are used as absolute addresses.
- If IX = 1 (Index 1 addressing mode), the received argument values for IX1 (both X and Y) are summed with the current values of IX1 to form a new set of IX1 values.
- If IX = 2 (Index 2 addressing mode), the actual received arguments values for IX1 (both X and Y) are summed with the current values of IX2 to form a new set of IX1 values.
- If IX = 3 (Relative addressing mode), the actual received argument values for IX1 (both X and Y) are summed with the current values of the XCOP and YCOP to form a new set of IX1 values.

INDEX 1 X - address default value = 0000 (Hex).

INDEX 1 Y - address default value = 0000 (Hex).

3-18 Index 2 Parameter (No. 4)



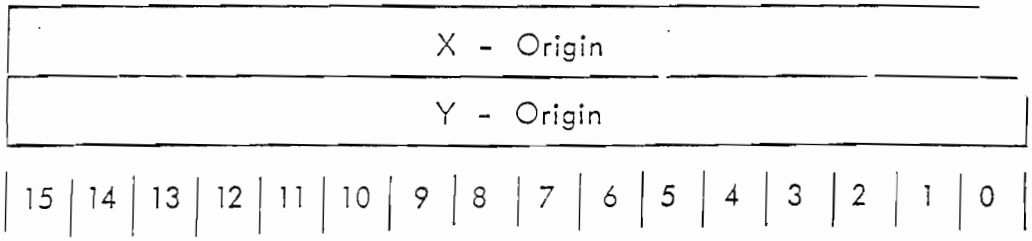
The INDEX 2 parameter may be set via any complex format instruction (except NOP). Its presence is flagged via Operand Flag Bit 2^4. The operand itself is two (2) 16-bit words in length and is identical to the INDEX 1 argument in both format and treatment. That is, received coordinate values (X/Y) in subsequent arguments are summed with the current INDEX 2 specified values in order to derive the effective (refresh memory) address, i.e., provided that Index Register 2

addressing was or is specified in the first word of the receive
Calculation of the absolute IX2 register values is computed in
to IX1.

INDEX 2 X - Address default value = 0000 (Hex).

INDEX 2 Y - Address default value = 0000 (Hex).

3-19 Origin Parameter (No. 5)



The ORIGIN parameter may be set via any normal instruction (except NOP). Its presence is flagged via Operand Flag Bit 5. The operand itself is two (2) 16-bit words and defines an address in refresh memory which becomes the origin for the video output to the CRT monitor (i.e., the upper, left-hand corner of the screen). Due to the RM-9000 memory system architecture, the ORIGIN values which place absolute refresh memory location (0,0) at the upper, left-hand corner of the video display are non-zero and different for each system type. These are the default value settings; Table 3-4 defines the default origin values for each RM-9000 series system type. The legal range of values for both the element and line origins is from zero to the element or line resolution value minus one. Therefore, for an RM-9300 display system, the legal range of element origin values is from 0 through 639_{10} and the legal range of line origin values is from 0 through 511. When altering the origin values, all data remains visible since wrapping will take place in both dimensions.

ORIGIN default values: See Table 3-4.

famtek

Table 3-4 Systems Resolution Definitions & Origin Default Values

SYSTEM	ELEMENT (X) RESOLUTION	LINE (Y) RESOLUTION	ELEMENT (X) DEFAULT ORIGIN	LINE (Y) DEFAULT ORIGIN
RM-9100	320_{10} 140_{16}	256_{10} 100_{16}	310_{10} 136_{16}	255_{10} FF_{16}
RM-9200	640_{10} 280_{16}	256_{10} 100_{16}	620_{10} $26C_{16}$	255_{10} FF_{16}
RM-9300	640_{10} 280_{16}	512_{10} 200_{16}	620_{10} $26C_{16}$	510_{10} $1FE_{16}$

3-20 Window Parameter (No. 6)

Start X - Address (X_L)
Start Y - Address (Y_T)
Stop X - Address (X_R)
Stop Y - Address (Y_B)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The WINDOW parameter may be set by any normal instruction (except NOP). Its presence is flagged by Operand Flag Bit 6. The operand itself is four (4) 16-bit words and specifies a rectangular region used in conjunction with the ERS, WI, RI, WT, WR, SCRX and SCRY instructions. The WINDOW values are read in the following order: X_L, Y_T, X_R, Y_B (where these values correspond to coordinates in Figure 3-4 . It is necessary that the WINDOW parameters conform to the following conditions: $X_L \leq X_R$ and $Y_T \leq Y_B$. Whenever the WINDOW parameter is specified, the Current Operating Point (COP) is set to the coordinate defined by Table 3-5 determined by the value of the SCAN parameter prior to this instruction. The default values for WINDOW are such that the entire refresh memory is within the window.

WINDOW START X (X_L) default value = 000
WINDOW START Y (Y_T) default value = 000
WINDOW STOP X (X_R) default value = XRES-1
WINDOW STOP Y (Y_B) default value = YRES-1

where XRES = system element resolution

YRES = system line resolution

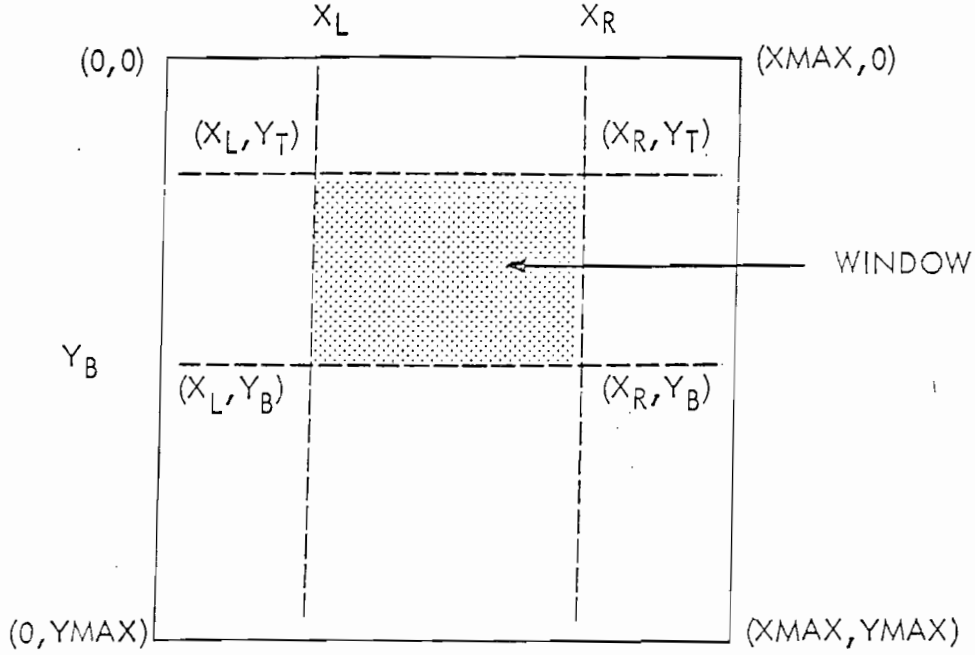


Figure 3-4 Window Definition

3-21 Scan Parameter (No. 7)

Scan Sequence															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The SCAN parameter may be set via any normal instruction (Except NOP). Its presence is flagged via Operand Flag Bit 7. The operand itself is a single 16-bit word in length and specifies one (1) of eight (8) possible scan sequences for the

Table 3-5 COP Placement After Window Setting

SCAN	XCOP	YCOP
0	X_L	Y_T
1	X_R	Y_T
2	X_L	Y_B
3	X_R	Y_B
4	X_L	Y_T
5	X_L	Y_B
6	X_R	Y_T
7	X_R	Y_B

WI, RI, WR, WT and WP instructions. For the WI, RI, WR instructions, SCAN is defined by Table 3-6.

Primary scan is the direction of consecutive pixels. Secondary scan is the wrap-around direction upon reaching a window boundary. That is, when the primary scan completes a line of pixels and is ready for wrap-around, the secondary scan will determine whether the second line of pixels is above, below, to the right or to the left of the first line.

For write text, scan direction is defined by both the SCAN and the SPACING parameters. The SCAN mode will determine the character orientation and whether the primary and secondary updates are horizontal or vertical. The SPACING parameter determines the direction of the update, i.e., to the left, to the right, up or down. The primary update is the update between successive characters. The secondary update is the update between successive character lines, i.e., the secondary update determines whether the second line of characters is above, below, to the right or to the left of the first line of characters. See Table 3-7.

A new line of characters is started whenever the last character reaches or passes the window boundary or a carriage return or line feed is encountered. The edge

Table 3-6 Image & Raster Mode Scan Directions

SCAN	DIRECTION OF WRITING PROCESS	
	PRIMARY	SECONDARY
0	Left-to-Right	Top-to-Bottom
1	Right-to-Left	Top-to-Bottom
2	Left-to-Right	Bottom-to-Top
3	Right-to-Left	Bottom-to-Top
4	Top-to-Bottom	Left-to-Right
5	Bottom-to-Top	Left-to-Right
6	Top-to-Bottom	Right-to-Left
7	Bottom-to-Top	Right-to-Left

of a character may exceed the window boundary. Wrap around is by complete characters only. A carriage return will start the next character line at the opposite window boundary.

Table 3-7 Write Text Scan Direction

SCAN	PRIMARY UPDATE	SECONDARY UPDATE
0	Horizontal	Vertical
1	Horizontal	Vertical
2	Horizontal	Vertical
3	Horizontal	Vertical
4	Vertical	Horizontal
5	Vertical	Horizontal
6	Vertical	Horizontal
7	Vertical	Horizontal

Table 3-8 Window Origin

SCAN	WINDOW ORIGIN
0	Upper Left-Hand Corner
1	Upper Right-Hand Corner
2	Lower Left-Hand Corner
3	Lower Right-Hand Corner
4	Upper Left-Hand Corner
5	Lower Left-Hand Corner
6	Upper Right-Hand Corner
7	Lower Right-Hand Corner

A line-feed will start a new character line at the current character position, i.e., no return to the opposite window boundary is made.

The positive direction for both the primary and secondary updates is to the right and down. If it is desired for either the primary or the secondary update to move to the left or up, then this update must be expressed as a two's complement negative number. See Figure 3-5 for the normal spacing values for each SCAN mode.

Normally the absolute values of the spacing parameters are equal to or greater than the character dimension parameters. If the spacing parameters are less than the dimension parameters, then the characters will overlap.

SCAN default value = 0

3-22 Dimension Parameter (No. 8)

WIDTH															
HEIGHT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

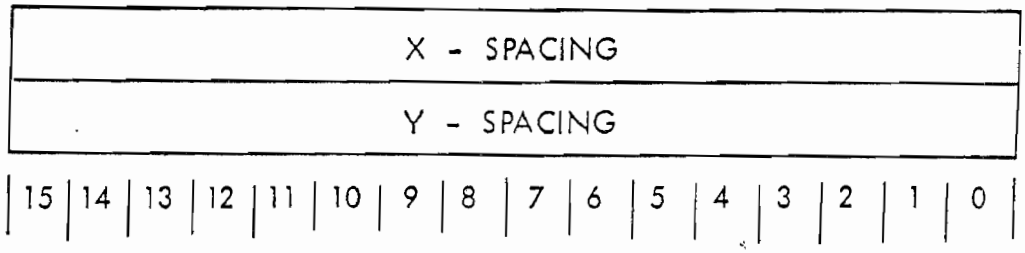
The DIMENSION parameter may be set via any normal instruction (except NOP). Its presence is flagged via Operand Flag Bit 8. The operand itself is two (2) 16-bit words in length and specifies the dimensions of the alphanumeric font in terms of height and width, and the height or width of individual plot segments in terms of lines or elements. The first word specifies character width, or plot segment width for horizontal plots, i.e., a curve being plotted from left-to-right, or visa-versa. The second word specifies character height, or plot segment height for vertical plots.

When this parameter is used to specify character width and height, it is independent of character orientation. It is possible to use the DIMENSION parameter with the programmable font option to generate character fonts of a smaller size than eight (8) elements by twelve (12) lines.

DIMENSION width default value = 7.

DIMENSION height default value = 9.

3-23 Spacing Parameter (No. 9)



The SPACING parameter may be set via any normal instruction (except NOP). Its presence is flagged via Operand Flag Bit 9. The operand itself is two (2) 16-bit words in length and negative spacing may be expressed in 2's complement form.

For write text, spacing determines the distance between successive characters and the distance between successive lines. The X-displacement is always in the horizontal direction and the Y-displacement is always in the vertical direction. The scan operand (Operand Parameter 8) will determine which displacement, X or Y, is between characters and which is between lines. See Figure 3-5.

The positive direction for the X-displacement is to the right and the positive direction for the Y-displacement is down. If it is desired for either displacement to be in the opposite direction, then that displacement must be expressed as a

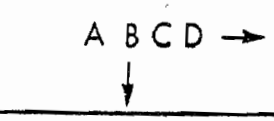
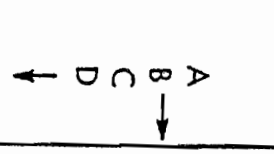
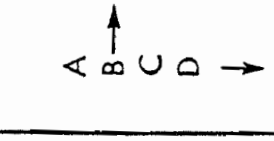
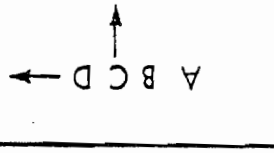
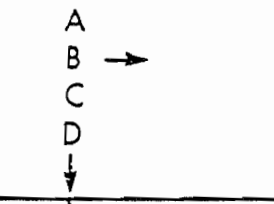
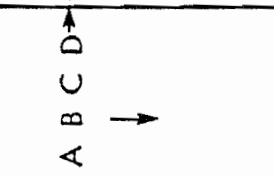
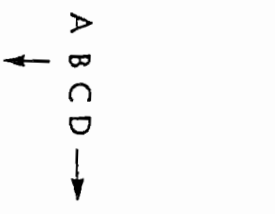
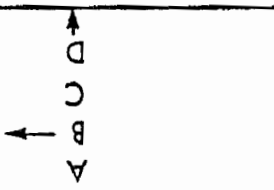
SCAN MODE	EXAMPLE	X SPACING	Y SPACING
0		Char Width	Char Height
1		- Char Height	Char Width
2		Char Height	- Char Width
3		- Char Width	- Char Height
4		Char Width	Char Height
5		Char Height	- Char Width
6		- Char Height	Char Width
7		- Char Width	- Char Height

Figure 3-5 Normal Spacing Values

two's complement negative number. See Figure 3-5 for the normal spacing values for each scan mode.

For write plot, spacing defines the increment from plot entity to plot entity, along the plot axis. For horizontal plots, X-spacing is used to define the plot axis increment; for vertical plots, Y-spacing is used to define the plot axis increment.

X - SPACING default value = 7

Y - SPACING default value = 9

3-24 Scale Parameter (No. 10)

Y - SCALE								X - SCALE							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The SCALE parameter may be set via any normal instruction (Except NOP). Its presence is flagged via Operand Flag Bit 10. The operand itself is a single 16-bit word in length and specifies a scaling factor or ratio of received or generated picture elements to displayed picture elements for the WI, WR and WT instructions. When scaling text, X-Scale always refers to character width and Y-Scale always refers to character height. When writing a negatively scaled image where fewer picture elements will be displayed than will be received or generated, each displayed pixel will represent the arithmetic average of the corresponding received or generated pixels. When raster or text data is negatively scaled, the reduction process will just ignore N-1 of every N pixels where N is the scale factor.

The scale process is window oriented; the scaled results are stored in an internal buffer within the RM-9000 until a completely composed scan line(s) has been created. When the composed scan line is completed, it is written to refresh memory. Thus, if only a partial line of scaled image or raster data is output to the RM-9000, this data will be lost if:

- SCAN, WINDOW, or SCALE parameters are set in an ensuing normal-format instruction, or
- If a valid POPE instruction is issued.

Each scale factor is represented as an 8-bit 2's complement number. The received picture element ratio is defined in Table 3-9.

SCALE element ratio default value = 0.

SCALE line ratio default value = 0.

Table 3-9 Scaled Picture Ratio

SCALE	RATIO	PICTURE ELEMENTS	
		RECEIVED	DISPLAYED
-2	4:1	4	1
-1	2:1	2	1
0	1:1	1	1
1	1:2	1	2
2	1:4	1	4

NOTE

EITHER OR BOTH THE X AND Y AXIS MAY BE SCALED EITHER UP OR DOWN, HOWEVER, ONE (1) AXIS MAY NOT BE SCALED UP WHILE THE OPPOSITE AXIS IS SCALED DOWN.

3-25 Function Parameter (No. 11)

LOGICAL/ARITHMETIC FUNCTION															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The FUNCTION parameter may be set by any normal format instruction (except NOP). Its presence is flagged by Operand Flag Bit 11. The parameter itself is a single 16-bit word in length and specifies a logical or arithmetic function (if any) to be performed in conjunction with the WRITE IMAGE instruction. By setting FUNCTION to a non-zero value, the WRITE IMAGE function is changed

from a write operation to a read-modify-write operation. Table 3-10 defines the legal function codes and their respective logical or arithmetic functions to be performed. All functions (except replacement) are performed using the old pixel value OP (i.e., the current value stored in refresh memory at the current pixel location) and the new pixel value NP (i.e., the pixel value loaded from the host processor). Before any function processing is performed, both the OP and NP are AND'ed with the current value of SUBCHANNEL; thus, only the subchannels of interest are subject to FUNCTION processing. The value resulting from FUNCTION processing RP is also AND'ed with SUBCHANNEL in the process of storing this pixel value in refresh memory.

The following is a description of each of the implemented logical/arithmetic processing functions. The following symbols are used in the ensuing section to describe the arithmetic/logical operations supported under the FUNCTION parameter:

RP	Pixel value resulting after processing specified by FUNCTION.
OP	Old pixel value (stored in refresh memory)
NP	New pixel value (input from the host computer via a WI instruction)
SC	Current value of the SUBCHANNEL parameter
$\leftarrow$	' is replaced by '
$\vee$	Logical inclusive OR
$\not\vee$	Logical Exclusive OR
$\wedge$	Logical AND
+	Addition
-	2's complement subtraction
>	' is greater than '
$\leq$	' is less than or equal to '
/	Division

Table 3-10 Logical/Arithmetic Function Codes

$CODE_{16}$	FUNCTION
0	NONE (DATA REPLACEMENT)
1	LOGICAL OR
2	LOGICAL XOR
3	LOGICAL AND
4	ARITHMETIC SUM
5	ARITHMETIC DIFFERENCE
6	GREATEST VALUE
7	LEAST VALUE
8	AVERAGE COMPUTATION
9	INVERSE ARITHMETIC DIFFERENCE
A	SIGN MAGNITUDE SUM
B	SIGN MAGNITUDE DIFFERENCE
C	SIGN MAGNITUDE INVERSE ARITHMETIC DIFFERENCE

FUNCTION 0 - REPLACEMENT

$RP \leftarrow NP \wedge SC$

The resulting pixel value is merely set equal to the new pixel value (i.e., normal WI processing is performed). The old pixel (OP) value is lost.

FUNCTION 1 - LOGICAL INCLUSIVE OR

$RP \leftarrow (NP \vee OP) \wedge SC$

The resulting pixel value is the logical inclusive OR of the old pixel (OP) value with the new pixel (NP) value.

FUNCTION 2 - LOGICAL EXCLUSIVE OR

$$RP \leftarrow (NP \wedge SC) \vee (OP \wedge SC)$$

The resulting pixel value is the logical exclusive OR of the old pixel (OP) value with the new pixel (NP) value.

FUNCTION 3 - LOGICAL AND

$$RP \leftarrow (NP \wedge OP) \wedge SC$$

The resulting value is the logical AND of OP and NP values.

FUNCTION 4 - ARITHMETIC SUM

$$RP \leftarrow ((NP \wedge SC) + (OP \wedge SC)) \wedge SC$$

The resulting pixel RP value is the arithmetic sum of NP and OP.

FUNCTION 5 - ARITHMETIC DIFFERENCE

$$RP \leftarrow ((OP \wedge SC) - (NP \wedge SC)) \wedge SC$$

The resulting pixel RP value is the 2's complement arithmetic difference of the old pixel OP value minus the new pixel NP value.

FUNCTION 6 - GREATEST VALUE

$RP \leftarrow OP \wedge SC$ if $(OP \wedge SC) - (NP \wedge SC) \wedge 0$

$NP \wedge SC$ if $(OP \wedge SC) - (NP \wedge SC) \wedge 0$

The resulting pixel RP value is set to either NP or OP whichever is greater. Since each value on input is masked by the subchannel mask value SUBCHANNEL (default value of $OFFF_{16}$), the comparison becomes unsigned when Bit 15 of SC is zero.

FUNCTION 7 - LEAST VALUE

RP $OP \wedge SC$ if $(OP \wedge SC) - (NP \wedge SC) \leq 0$

$NP \wedge SC$ if $(OP \wedge SC) - (NP \wedge SC) \rightarrow 0$

The resulting pixel RP value is set to either NP or OP whichever is lesser in value. Since each value is masked by the subchannel mask value SUBCHANNEL (default value $OFFF_{16}$) on input, the comparison becomes unsigned when Bit 15 of SC is zero.

FUNCTION 8 - AVERAGE VALUE

$$RP \leftarrow ((((OP \wedge SC) + (NP + SC)) /2) \wedge SC)$$

The resulting pixel RP value is the average value of OP and NP. The mechanism used to perform the division by 2 is such that rounding-up of the result does not occur, e.g., the average value of 0010_{16} and $000F_{16}$ is $000F_{16}$.

FUNCTION 9 - INVERSE ARITHMETIC DIFFERENCE

$$RP \leftarrow ((NP \wedge SC) - (OP \wedge SC)) \wedge SC$$

The resulting pixel RP value is the 2's complement arithmetic difference of the new pixel NP value minus the old pixel OP value.

In functions A_{16}, B_{16}, and C_{16}, sign-magnitude arithmetic is used. The sign-magnitude functions use the highest-order bit which is set to 1 in SUBCHANNEL (SC) as the sign bit. Negative numbers, therefore, have the same representation as positive numbers but with the sign bit set to 1. Since subchannels which do not exist read back as 1, it is crucial that the SUBCHANNEL parameter be set up to specifically include only those subchannels necessary for processing. In sign-magnitude arithmetic, if the sum of 2 positive numbers produces a carry into the sign bit, then the carry will be lost, i.e., the result will remain positive. If the sum of two (2) negative numbers produces a carry into the sign bit, then the result will remain negative.

EXAMPLE - Subchannel Mask = $0000001111110000 = 03F0_{16}$

$0020_{16} + 0210_{16} = 0010_{16}$ $\quad 2 + (-1) = 1$

$0220_{16} + 0210_{16} = 0230_{16}$ $\quad -2 + (-1) = 3$

$0020_{16} + 0240_{16} = 0220_{16}$ $\quad 2 + (-4) = 2$

Subchannel Mask = $0000000100001111 = 010F_{16}$

$0002_{16} + 0101_{16} = 0001 \quad 2 + (-1) = 1$

$0102_{16} + 0101_{16} = 0103 \quad -2 + (-1) = -3$

$0002_{16} + 0104_{16} = 0102 \quad 2 + (-4) = -2$

FUNCTION A_{16} - ARITHMETIC SUM (SIGN MAGNITUDE)

RP ← ((OP ^ SC) + (NP ^ SC)) ^ SC (Sign-Magnitude Sum)

The resulting pixel (RP) value is the sum of the old pixel (OP) value and the new pixel (NP) value. Both (NP) and (OP) are evaluated as sign-magnitude numbers using the SUBCHANNEL parameter to define the sign bit.

FUNCTION B_{16} - ARITHMETIC DIFFERENCE (SIGN MAGNITUDE)

RP ← ((OP ^ SC) - (NP ^ SC)) ^ SC (Sign-Magnitude Difference)

The resulting pixel (RP value is the difference of the old pixel (OP) value minus the new pixel (NP) value. Both (NP) and (OP) are evaluated as sign-magnitude numbers using the SUBCHANNEL parameter to define the sign bit.

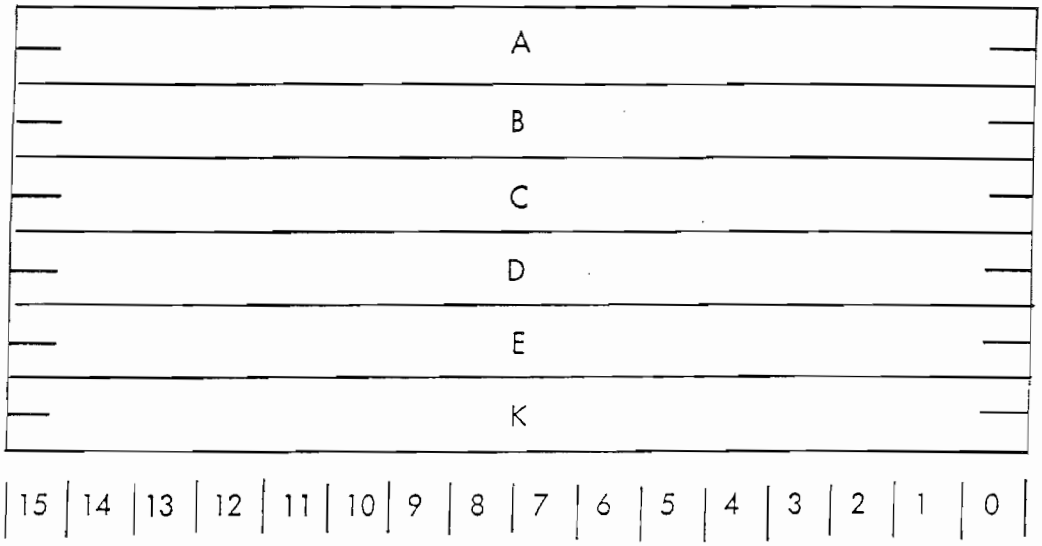
FUNCTION C_{16} - INVERSE ARITHMETIC DIFFERENCE (SIGN MAGNITUDE)

RP ← ((NP ^ SC) - (OP ^ SC)) ^ SC (Sign-Magnitude Difference)

The resulting pixel (RP) value is the difference of the new pixel (NP) value minus the old pixel (OP) value. Both (NP) and (OP) are evaluated as sign-magnitude numbers using the SUBCHANNEL parameter to define the sign bit.

FUNCTION default value = 0 (Replacement Mode)

3-26 Conic Equation (Parameter (No. 12)



The CONIC EQUATION parameter operands may be set by any normal format instruction (except NOP). Its presence is flagged by Operand Flag Bit 12. This parameter consists of six double-word arguments A, B, C, D, E and K. The first word of each double-word pair is the high-order word. The second word is the low-order word. In the current implementation of RM-CON, the high-order word is not used, and was included to allow for possible future expansion to higher resolution display systems.

The arguments A, B, C, D and E are 16-bit 2's complement integers which are used as the generalized conic equation coefficients. The generalized form of the conic equation is:

$$Ax^2 + By^2 + Cxy + Dx + Ey = 0.$$

The argument K represents the total number of pixels to be generated for the conic described by A, B, C, D and E. For example, a circle of radius 100_{10} pixels would have a value of 400_{10} for K.

For a detailed description of the use of the CONIC-EQUATION parameter operand in conjunction with the WC instruction, see Section 3-51 and Appendix B.

CONIC-EQUATION default values:

A = 0	D = 0
B = 0	E = 0
C = 0	K = 1280_{10}

3-27 Baseline Parameter (No. 13)

FILLED PLOT BASELINE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The BASELINE parameter may be set via any normal format instruction (except NOP). Its presence is flagged by Operand Flag Bit 13. The operand itself is a 16-bit word in length and specifies whether a filled plot or a line plot is to be drawn. When BASELINE is zero, a line plot (i.e., a plot in which each endpoint along the curve becomes the start point for the succeeding plot segment) is drawn. When BASELINE is non-zero, the BASELINE defines the start point for each plot segment. If SCAN is between 0 and 3, then BASELINE defines the horizontal axis to which the filled-plot segments will be drawn, i.e., BASELINE defines a Y-address. Similarly, if SCAN is between 4 and 7, then BASELINE defines the vertical axis to which the filled-plot segments will be drawn, i.e., BASELINE defines an X-address.

DEFAULT BASELINE VALUE : 0

3-28 Scroll-Count Parameter (No. 14)

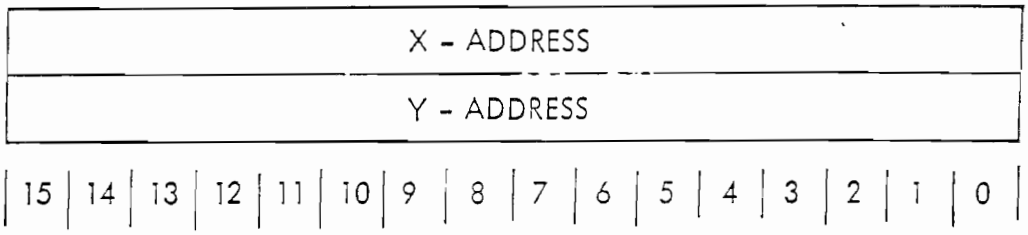
SCROLL COUNT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The SCROLL-COUNT parameter may be set by any complex format instruction (except NOP). Its presence is flagged by Operand Flag Bit 14. The operand itself is a single 16-bit word in length and specifies scroll count and direction

i.e., left or right for SCRX, or up or down for SCRY. When set to a negative value, scroll will occur in the negative direction. That is, the image will be scrolled left for SCRX and up for SCRY. When set to a positive value, the image will be scrolled right for SCRX and down for SCRY. The scroll count is a 2's complement 16-bit number.

DEFAULT SCROLL-COUNT VALUE : 0

3-29 Start-Point Argument (No. 15)



The START-POINT argument may be set by any complex format instruction (except NOP). Its presence is flagged by Operand Flag Bit 2^{15}. The operand itself is two (2) 16-bit words in length and specifies a start-point for the WI, RI, WR, WT, WV, WC and WP instructions. Note that when WINDOW or SCAN are specified, the appropriate start-point is automatically calculated for the WI, RI, WR and WT instructions and, therefore, need not be specified unless the writing process is to begin in other than the appropriate corner of the WINDOW. Table 3-8 defines the default values for START-POINT based on the setting of WINDOW and SCAN. Because WINDOW is not pertinent to the WV, WC and WP instructions, START-POINT must be specified. Otherwise, the last end-point (current COP) will be used as the new start point. The START-POINT explicitly sets the Current Operating Point (COP) according to the addressing mode specified by the instruction containing the parameter. For window oriented commands, the value specified must be within the area specified by the WINDOW parameter for proper operation.

DEFAULT X START POINT VALUE : 0

DEFAULT Y START POINT VALUE : 0

3-30 Data Length Word

The Data Length Word defines the number of bytes of data for any normal format instruction. It defines either the number of bytes to be read by the RM-9000 for the NOP, SET, ERS, WI, WT, WR, WV, WC, WP, SCRX, SCRY instructions or the number of bytes to be read by the host computer for the RI instruction. The Data Length Word will be present immediately after the opcode word or any normal instruction parameters which may be present if Bit 0 of the Parameter Byte (i.e., the DATA FLAG) is set to one (1). The data length word may take on a maximum value of 65535_{10} Bytes represented by an unsigned 16-bit number.

3-31 Normal-Format Instruction Data

If the Data Flag (Bit 0 of the parameter byte) is 1, the n bytes of data as specified in the Data Length Word will be transferred immediately following the Data Length Word in any normal format instruction. The format of this data varies from instruction to instruction. Five data types are possible for the normal instruction set:

- Image data for the WI and RI instructions
- Text data for the WT instruction
- Plot data for the WP instruction
- Endpoint data for the WV and WC instructions
- Raster data for the WR instruction.

All other normal format instructions will read in the indicated number of bytes of data and subsequently discard them.

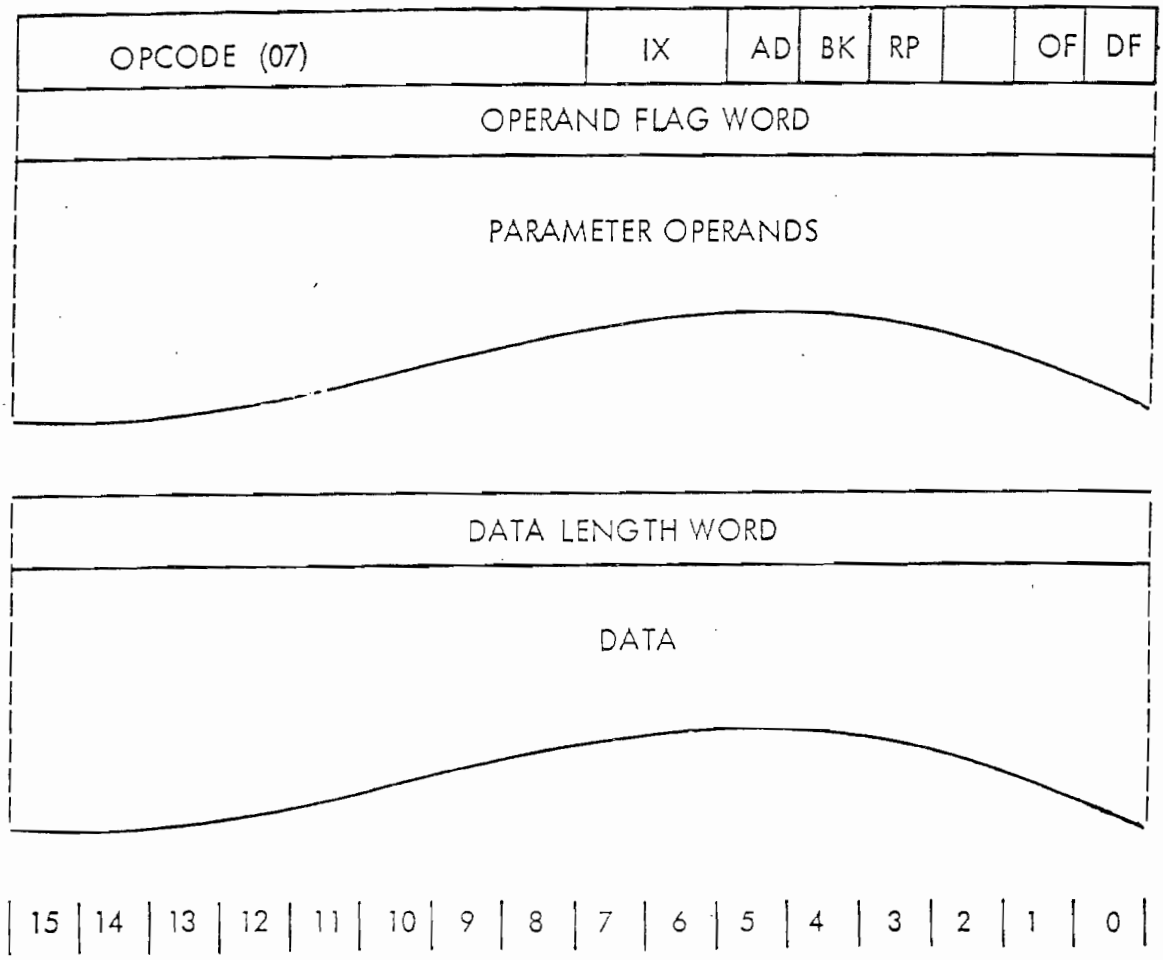
3-32 Normal-Format Standard Instructions

Sections 3-33 through 3-38 define the formats and functions of the normal-format instructions which are supported by the RM-9000 standard firmware package. The instructions in this set are:

INOP	-	No Operation Instruction
SET	-	Set Parameter Instruction
ERS	-	Erase Instruction
WI	-	Write Image Instruction
RI	-	Read Image Instruction
WT	-	Write Text Instruction

These instructions are standard in all RM-9000 systems.

3-33 No-Operation Instruction (INOP)



The INOP instruction is a normal-format instruction which is included in the RM-9000 standard firmware. The INOP instruction performs no internal functions whatsoever; any parameter operands or data which are present in the instruction stream are discarded. The INOP instruction is useful in facilitating the debugging of instruction streams passed from the host processor to the RM-9000. By simply changing the opcode byte of any normal-format instruction or of any single word special-format instruction, the integrity of the instruction stream can be maintained while selectively eliminating the effects of one or more instructions.

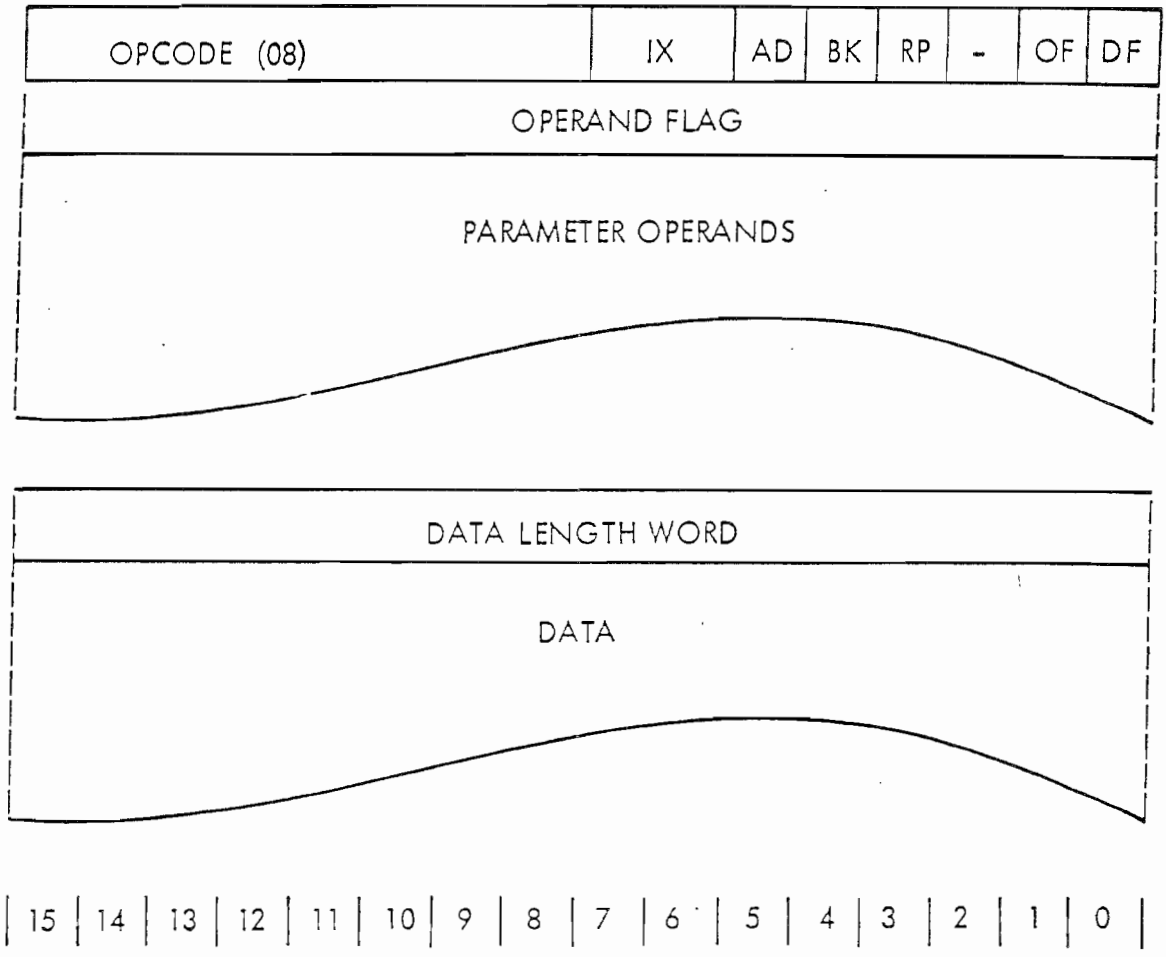
PERTINENT ARGUMENTS

None

POSSIBLE ERROR CONDITIONS

None

3-34 Set Parameter Instruction (SET)



The SET instruction is a normal-format instruction which is included in the RM-9000 standard firmware. The SET instruction allows all parameter operands to be defined (based in some cases on the addressing mode) as in any normal-format instruction, but any data which is present in the instruction stream will be ignored. This instruction is included in the standard firmware in order to facilitate debugging of a display instruction stream. It allows a user to perform parameter operand processing whose internal modifications could carry over to the subsequent instructions in the instruction stream, while ignoring the received data.

PERTINENT CONTROL BITS

IX Defines the address mode in which the INDEX 1, INDEX 2, ORIGIN, WINDOW, BASELINE and STARTPOINT will be evaluated.

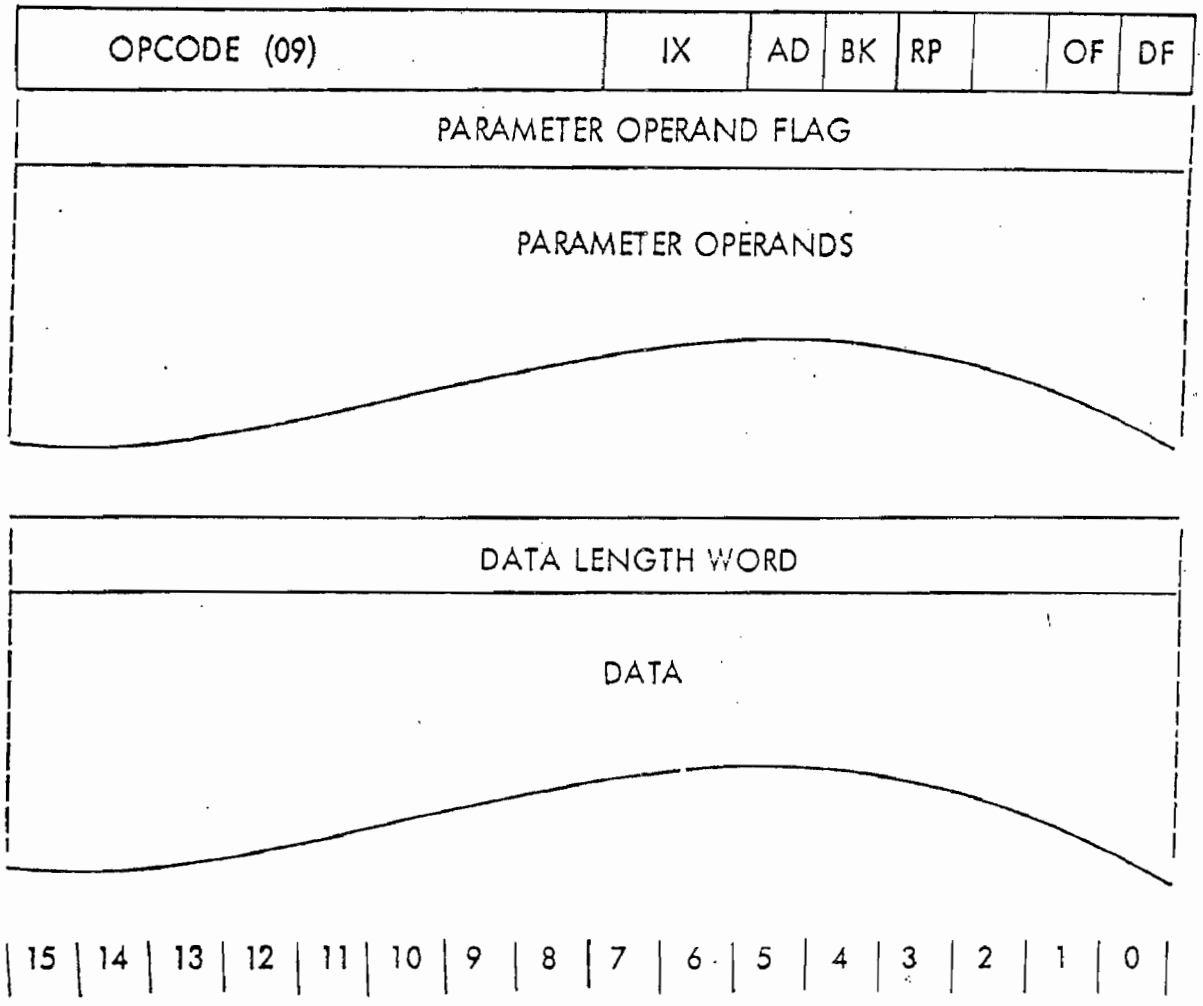
PERTINENT PARAMETER OPERANDS

All parameter operands are pertinent to the SET instruction in the sense that all parameter operands may be re-defined; however, since SET performs no display functions, none of the arguments are pertinent to the display of data in an immediate sense.

DATA FORMAT

Since all data is discarded, data format is irrelevant.

3-35 Erase Instruction (ERS)



The ERS instruction is a normal-format instruction which is included in the RM-9000 standard firmware. The ERS instruction sets the rectangular area in refresh memory defined by the WINDOW parameters to either the FOREGROUND or BACKGROUND value based on the value of BK. If BK = 0, the BACKGROUND value will be used; otherwise if BK = 1, the FOREGROUND value will be used. Regardless of FOREGROUND/BACKGROUND usage, the selected value will be write-masked by the SUBCHANNEL parameter operands. The Cartesian data generated by the ERS instruction uses the internal mode also used by the text and the raster processors, the AD flag can affect ERS processing. If AD = 1, then no Cartesian data will be written into refresh (See Table 3-3 of Section 3-10) with a data value of 0. All data associated with an ERS instruction will be discarded.

PERTINENT CONTROL BITS

IX — Defines the address mode in which the WINDOW parameter operands will be evaluated, and as such, affects the rectangular area to be used.

AD — Affects the generation of Cartesian data such that if AD = 1, no data will be written to refresh memory.

BK — Defines the color or intensity value to be used, i.e., if BK = 0, the BACKGROUND value is used; otherwise, the FOREGROUND value is used.

PERTINENT PARAMETER OPERANDS

SUBCHANNEL — Defines the subchannel write-enable mask; only those subchannels whose corresponding bit in the SUBCHANNEL mask will be written with data from the ERS instruction.

FOREGROUND — Defines the color or intensity value to be stored in refresh memory when BK = 1.

BACKGROUND — Defines the color or intensity value to be stored in refresh memory when BK = Ø.

INDEX 1 — Displaces the WINDOW parameters when WINDOW is set in the ERS instruction and IX = 1.

INDEX 2 — Displaces the WINDOW parameters when WINDOW is set in the ERS instruction and IX = 2.

WINDOW — Defines the rectangular area to be erased.

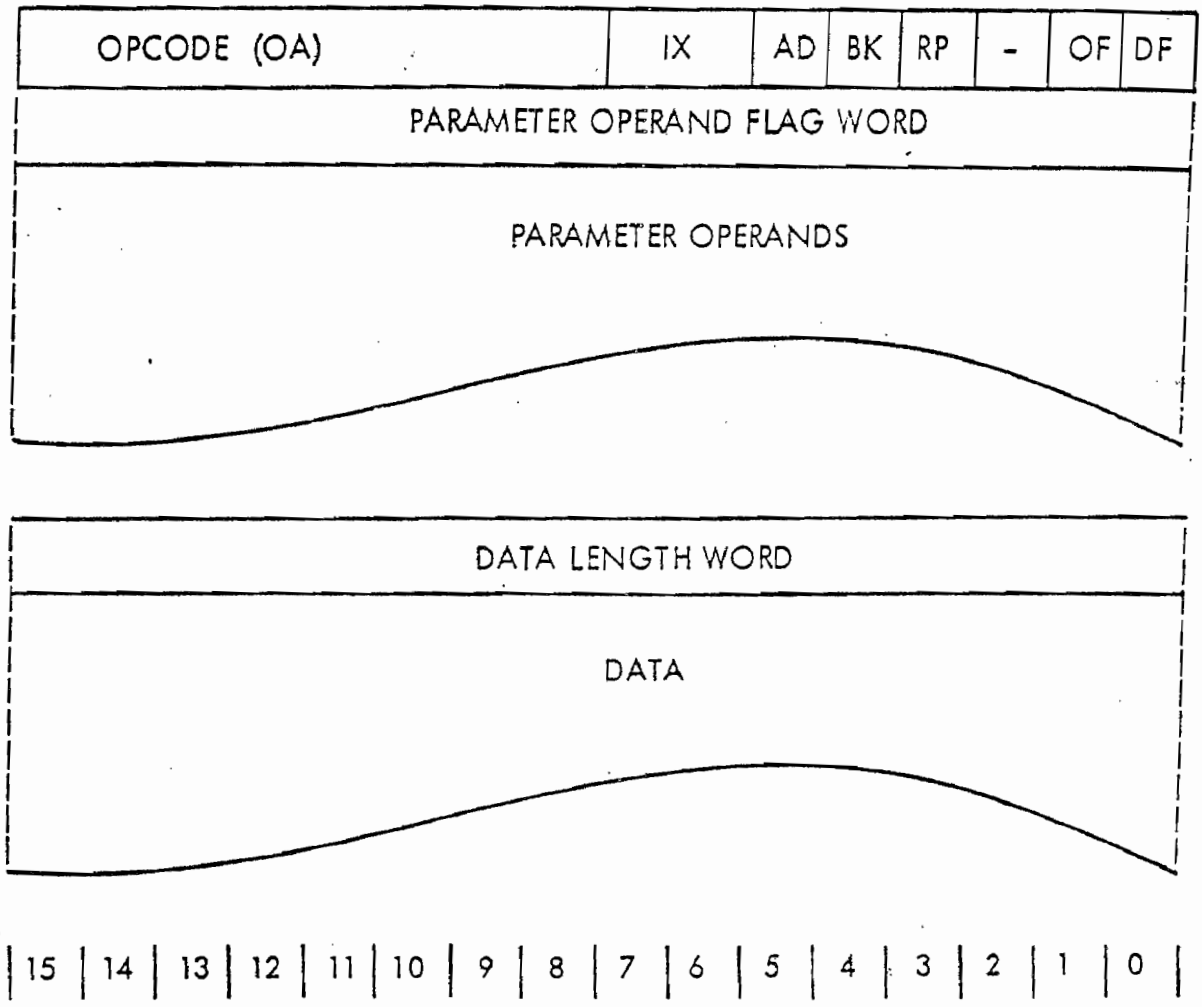
DATA FORMAT

Since all data is discarded, data format is irrelevant.

POSSIBLE ERROR CONDITIONS

If the WINDOW parameter values are outside of the system resolution, the resulting Cartesian data will be indeterminate.

3-36 Write Image Instruction (WI)



The WI instruction is a normal-format instruction which is included in the RM-9000 standard firmware. The WI instruction stores up to 32767_{10} words of data in refresh memory on a word-per-pixel basis within the rectangular area defined by WINDOW. The first 16-bit word of image data received from the host processor is stored in refresh memory at the pixel defined by the START-POINT parameter operand or by the current operating point which was the result of the previous instructions if no START-POINT value is specified in the WI instruction. The successive words of image data are stored in refresh memory based on the primary update mode associated with SCAN (See Section 3-21).

When a window boundary is encountered while storing image data words in successive pixels based on the primary update direction, the current operating point is set to the opposite window boundary and incremented in a direction

perpendicular to the primary scan direction (i.e., the secondary update direction defined by SCAN). When a window boundary is encountered in the process of incrementing in the secondary update direction, the current operating point is repositioned to the opposite window boundary.

Since the data to be written to the refresh memory is supplied directly from the host processor and not from the FOREGROUND or the BACKGROUND parameter operand, the AD and BK control flags are ineffective in image mode.

PERTINENT CONTROL BITS

IX	Defines the address mode in which the INDEX 1, INDEX 2, WINDOW and START-POINT parameter operands set by any given WI instruction, are evaluated.

PERTINENT PARAMETER OPERANDS

SUBCHANNEL	Specifies the subchannels (i.e., bit planes in refresh memory) which are write-enabled and which will receive image data.
INDEX 1	Displaces the values to be used for WINDOW and START-POINT parameter operands set in the WI instruction when IX = 1.
INDEX 2	Displaces the values to be used for WINDOW and START-POINT parameter operands set in the WI instruction when IX = 2.
WINDOW	Defines the rectangular area into which image data will be written. In addition, if the START-POINT parameter is not explicitly set in the WI instruction, WINDOW along with SCAN defines the starting pixel coordinates (See Section 3-21).
SCAN	Defines the primary and secondary update directions as well as the starting pixel coordinates when START-POINT is not explicitly defined in a WI instruction.
SCALE	Defines the ratio of pixels written in refresh memory to the number of image data words received from the host processor (See Section 3-24).

FUNCTION — Defines one of thirteen (13) possible image processing modes to be applied as each pixel is stored in refresh memory. A value of Ø defines a write with no processing operation; while the other twelve (12) non-zero functions represent a read-modify-write operation per pixel (See Section 3-25).

START-POINT — Specifies the coordinates of the first pixel to be written with image data; if not defined, the current operating point which was the result of the previous instruction is used as the starting pixel coordinate.

DATA LENGTH WORD

The DATA LENGTH WORD represents the number of bytes of data to be transmitted from the host processor to the RM-9000 with a WI instruction. Since image data is defined on a word basis, the DATA LENGTH WORD should always reflect an even number of bytes. If the DATA LENGTH WORD is odd, the byte count used will be one less than the actual byte count stored, e.g., if the DATA LENGTH WORD had a byte count value of 33_{10} bytes, 32_{10} bytes or 16_{10} words would be expected by the RM-9000. The range of the DATA LENGTH WORD is from 0 through 65534_{10} bytes or 32767_{10} words.

DATA FORMAT

Data in image mode is interpreted on a word basis. Of the 16-bits of data per word, only the low-order twelve (12) are actually used when no logical/arithmetic FUNCTION processing is to be performed. Assuming that all subchannels have been write-enabled via SUBCHANNEL, each bit of the incoming data word is written to its corresponding subchannel, i.e., data Bit 0 to Subchannel 0, Data Bit 1 to Subchannel 1, ..., Data Bit 11 to Subchannel 11. When FUNCTION processing is to be performed, the incoming data is and'ed with SUBCHANNEL before use, therefore all 16-bits of image data could be significant if the value of SUBCHANNEL enabled bits in the upper four (4) positions (See Section 3-25 on FUNCTION processing).

COP MOVEMENT

The resulting current operating point after the completion of a WI instruction is the coordinates of the next pixel which would have been written if (N + 1) words of data had been passed in the instruction rather than N words. Since

the primary and secondary update directions are defined by SCAN, the resulting COP position is a function of SCAN and the number of words of image data written. For further discussion of the COP movement, see Example 3-1.

POSSIBLE ERRORS

If the DATA LENGTH WORD is odd, it is possible for an unsuspecting user to get out of synchrony with the RM-9000, i.e., interpret a word of data as an instruction opcode word.

EXAMPLE 3-1

This example demonstrates the use of the WI instruction and the operation of SCAN and WINDOW in conjunction with this instruction. The following instruction stream stores data in a rectangular window in refresh memory and the associated Figures 3-6 a through h, indicate the resulting displays. The window is defined by the coordinate corners (100,100) and (103,103). Sixteen (16) words of data (designated by the symbols D1 through D16) will be written into the rectangular area defined by the WINDOW parameter operand:

HEX	MNEMONIC	DESCRIPTION
0500	RESET	; Clear Screen
0A03	WI+OF+DF	; Write Image Instruction
00C0	SCN+WIN	; Operand Flag Word
XXXX	XXXX	; Scan Value from 0 through 7
0064	100	; Element Left Margin = 100_{10}
0064	100	; Line Top Margin = 100_{10}
0067	103	; Element Right Margin = 103_{10}
0067	103	; Line Bottom Margin = 103_{10}
0020	32	; Data Length Word = 32_{10}
	D1	;
	D2	; 32 bytes (16 words) of data
	⋮	;
	D16	;

Where XXXX defines the scan mode from 0 through 7.

Notice that the current operating point (COP) after all of the image transfers are complete is identical to the starting COP, since after data value D16 was transferred both primary and secondary updates are exercised.

If the DATA LENGTH WORD value was increased to 34_{10} and a seventeenth data value D17 added, the primary and secondary updates return the COP to the starting coordinates and over-write the pixel at (100,100) with D17.

100	101	102	103	
D1	D2	D3	D4	100
D5	D6	D7	D8	101
D9	D10	D11	D12	102
D13	D14	D15	D16	103

(a) SCAN MODE = 0
START X = 100
START Y = 100

100	101	102	103	
D4	D3	D2	D1	100
D8	D7	D6	D5	101
D12	D11	D10	D9	102
D16	D15	D14	D13	103

(b) SCAN MODE = 1
START X = 103
START Y = 100

100	101	102	103	
D13	D14	D15	D16	100
D9	D10	D11	D12	101
D5	D6	D7	D8	102
D1	D2	D3	D4	103

(c) SCAN MODE = 3
START X = 100
START Y = 103

100	101	102	103	
D16	D15	D14	D13	100
D12	D11	D10	D9	101
D8	D7	D6	D5	102
D4	D3	D2	D1	103

(d) SCAN MODE = 3
START X = 103
START Y = 103

100	101	102	103	
D1	D5	D9	D13	100
D2	D6	D10	D14	101
D3	D7	D11	D15	102
D4	D8	D12	D16	103

(e) SCAN MODE = 4
START X = 100
START Y = 100

100	101	102	103	
D4	D8	D12	D16	100
D3	D7	D11	D15	101
D2	D6	D10	D14	102
D1	D5	D9	D13	103

(f) SCAN MODE = 5
START X = 100
START Y = 103

100	101	102	103	
D13	D9	D5	D1	100
D14	D10	D6	D2	101
D15	D11	D7	D3	102
D16	D12	D8	D4	103

(g) SCAN MODE = 6
START X = 103
START Y = 100

100	101	102	103	
D16	D12	D8	D4	100
D15	D11	D7	D3	101
D14	D10	D6	D2	102
D13	D9	D5	D1	103

(h) SCAN MODE = 7
START X = 103
START Y = 103

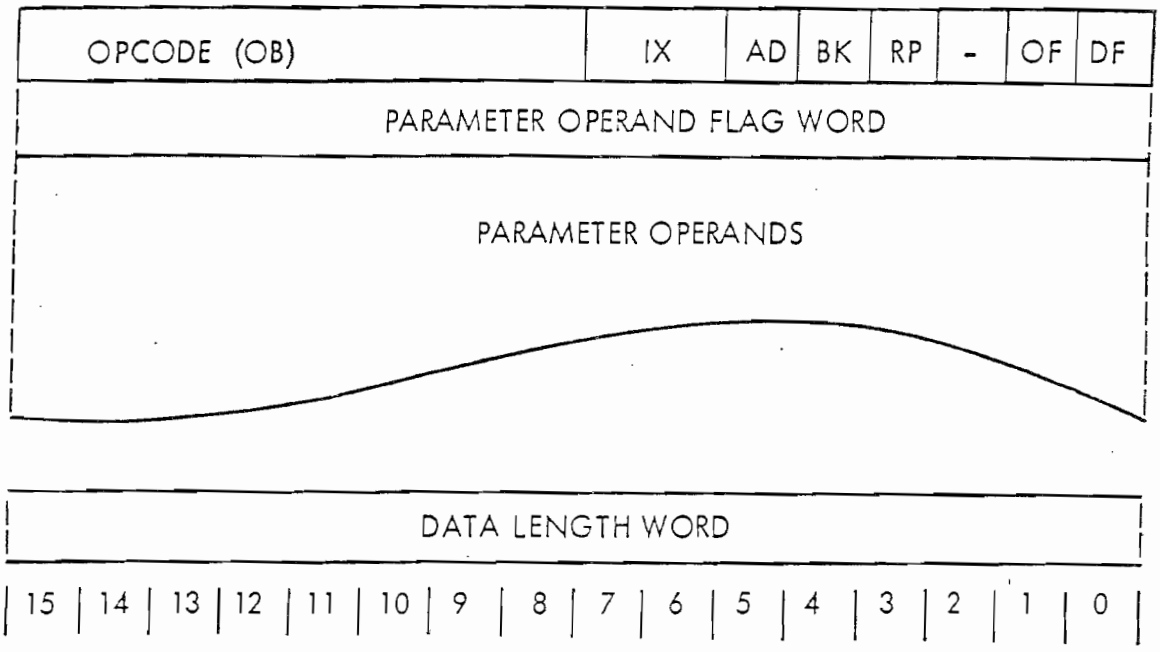
100	101	102	103	
D17	D2	D3	D4	100
D5	D6	D7	D8	101
D9	D10	D11	D12	102
D13	D14	D15	D16	103

(i) SCAN MODE = 0
START X = 100
START Y = 100

Figure 3-6 Imaging Scan Example

famtek

3-37 Read Image Instruction (RI)



The RI instruction is a normal format instruction which is included in the RM-9000 standard firmware. The RI transfers up to 32767_{10} words of refresh memory data to the host processor on a word-per-pixel basis from the rectangular area defined by WINDOW. The first 16-bit word of image data transferred from refresh memory to the host is obtained from the pixel defined by the START-POINT parameter operand or by the current operating point which was the result of the previous instruction(s) if no START-POINT value is specified in the RI instruction. The successive words of image data are read back from refresh based on the primary update mode associated with SCAN (See Section 3-21).

When a window boundary is encountered while reading image data words from successive pixels based on the primary update direction, the current operating point is set to the opposite window boundary and incremented in a direction perpendicular to the primary scan direction. When a window boundary is encountered in the process of incrementing in the secondary update direction, the current operating point is repositioned to the opposite window boundary.

Since the data to be written to the host processor is supplied directly from the refresh memory and not from the FOREGROUND or the BACKGROUND parameter operand, the AD and BK control flags are ineffective in image mode.

When data is read back from the refresh memory using the RI instruction, the SUBCHANNEL write-enable mask is not effective. All subchannels are passed back to the host processor. An RM-9000 can have any number of subchannels up to twelve (12) and the data which is read back from subchannels which are not present in the configuration is indeterminate. The high order 4 bits of the read back data value will read back to the host processor as 1. Thus, if refresh memory has been erased to zeros, all data words would read back as $F000_{16}$ in a 12-su channel system.

PERTINENT CONTROL BITS

IX	Defines the address mode in which the INDEX 1, INDEX 2, WINDOW and START-POINT parameter operands set by any given WI instruction, are evaluated.

PERTINENT PARAMETER OPERANDS

INDEX 1	Displaces the values to be used for WINDOW and START-POINT parameter operands set in the WI instruction when IX = 1.
INDEX 2	Displaces the values to be used for WINDOW and START-POINT parameter operands set in the WI instruction when IX = 2.
WINDOW	Defines the rectangular area from which image data will be read back. In addition, if the START-POINT parameter is not explicitly set in the WI instruction, WINDOW along with SCAN defines the starting pixel coordinates (See Section 3-21).
SCAN	Defines the primary and secondary update directions as well as the starting pixel coordinates when START-POINT is not explicitly defined in the RI instruction.
START-POINT	Specifies the coordinates of the first pixel to be written with image data; if not defined, the current operating point which was the result of the previous instruction is used as the starting pixel coordinate.

DATA LENGTH WORD

The DATA LENGTH WORD represents the number of bytes of data to be transmitted from the RM-9000 to the host processor with a RI instruction. Since image data is defined on a word basis, the DATA LENGTH WORD should always reflect an even number of bytes. If the DATA LENGTH WORD is odd, the byte count used will be one less than the actual byte count stored, e.g., if the DATA LENGTH WORD had a byte count value of 33_{10} bytes, 32_{10} bytes or 16_{10} words would be expected from the RM-9000. The range of the DATA LENGTH WORD is from 0 through 65534_{10} bytes or 32767_{10} words.

DATA FORMAT

Data in image mode is transmitted on a word basis. Of the 16-bits of data per word, only the low-order 12 actually carry refresh memory data. The low-order 12-bits correspond to subchannels such that Bit 2^0 represents subchannel 0, Bit 2^1 represents Subchannel 1. Subchannels which do not exist read back indeterminately. Bit 2^{12} through 2^{15} of the read back data value will always be set to 1.

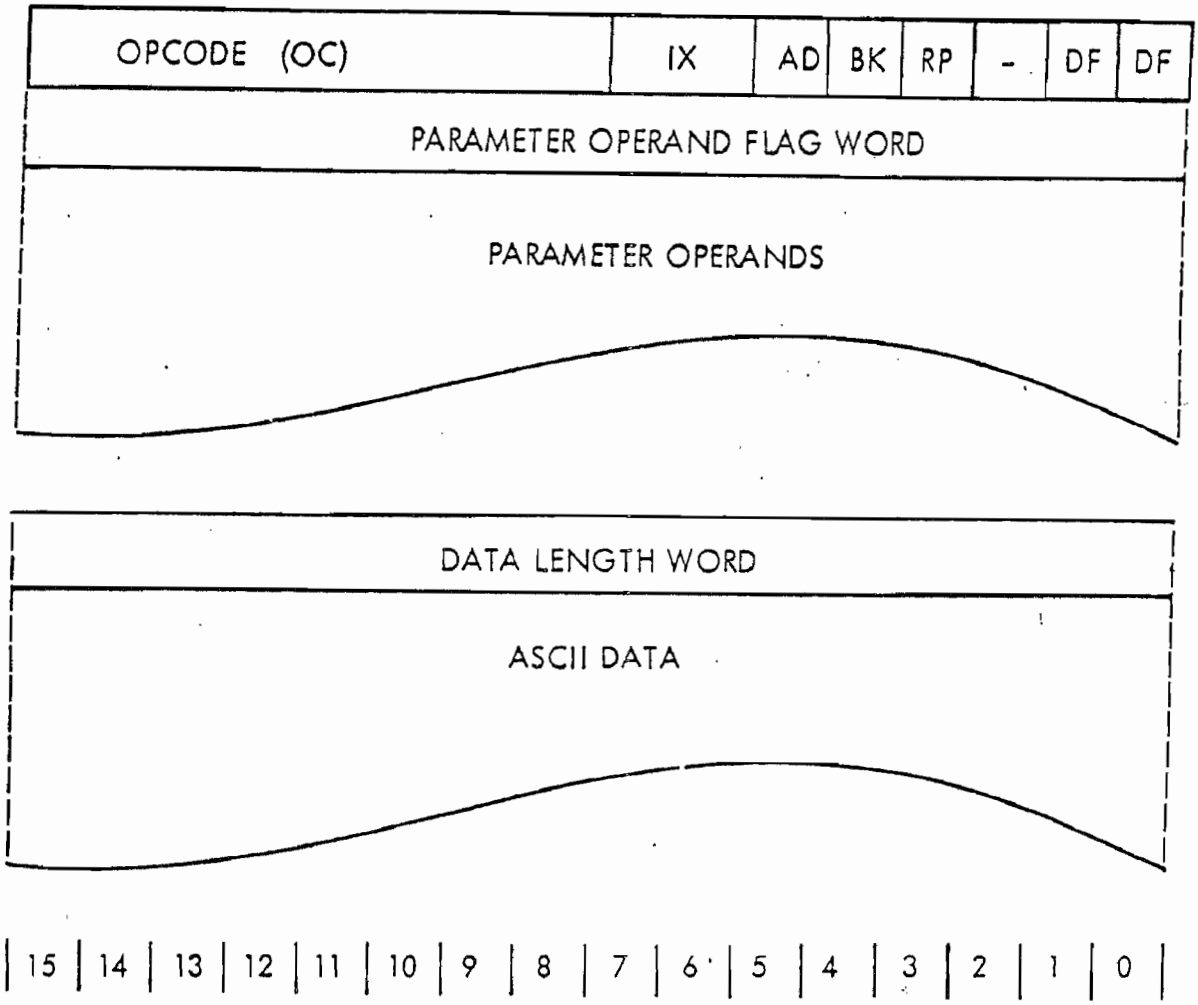
COP MOVEMENT

The resulting current operating point after the completion of an RI instruction is the coordinates of the next pixel which would have been read back if (N + 1) words of data had been passed in the instruction rather than N words. Since the primary and secondary update directions are defined by SCAN, the resulting COP position is a function of SCAN and the number of words of image data read back. For further discussion of the COP movement, see Example 3-1.

POSSIBLE ERRORS

If the DATA LENGTH WORD is odd, it is possible for the unsuspecting user to get out of synchrony with the RM-9000, i.e., both systems will be attempting to read from each other and a hard system reset will be necessary.

3-38 Write Text Instruction (WT)



The WT instruction is a normal format instruction included in the RM-9000 standard firmware. The WT instruction reads ASCII character codes from host processor and generates the corresponding text characters on to a rectangular area defined by the WINDOW parameter operand. The standard 64_{10} ASCII character codes (20_{16} through $5F_{16}$) are available in a system without the RM-FNT programmable font option (See Appendix B). If the RM-FNT programmable font option is installed, the character font descriptions for codes 20_{16} through $5F_{16}$ are copied to the programmable font option internal storage upon RESET (both hardware reset and the RM-9000 RESET instruction). Character codes 20_{16} through $9F_{16}$ are able to be loaded by the host processor via the LPF and LPFRP instructions (See Sections 3-60 & 3-61). Character codes 0 through $1F_{16}$ are illegal (except for carriage return OD_{16} and line feed OA_{16}) and will generate a character of undefined font.

The actual font size of a character is defined by the DIMENSION parameter operands. This is the size of the rectangle which will have text/font data stored in refresh memory. In a standard system without the RM-FNT option, the values of DIMENSION should never exceed its default values of 7 pixels wide and 9 pixels high. If DIMENSION is set to values less than 7 by 9, the right-most and bottommost portions of the 7 by 9 font will be lost, respectively. In a system with the RM-FNT option, the maximum values for the DIMENSION parameter operand are 8 pixels wide and 12_{10} pixels high. This is the font size which is able to be loaded by the host processor. The spacing which is performed between characters within the rectangular window is defined by the SPACING operand parameters. Since the SPACING operand parameter is defined in terms of X-spacing and Y-spacing, in order to get proper updating between characters it is necessary to change the SPACING parameter in the case where SCAN is changed (See Figure 3- 5) for the normal spacing values in text mode.

Although the WINDOW parameter operand defines the rectangular region for text generation, it is possible to generate part of a character outside of this area. If the first pixel of any character to be generated is within the WINDOW region, the entire character will be generated. Therefore, it is possible for up to (DIMENSION width - 1) pixels to be generated outside of the rectangular WINDOW region.

The carriage return character (Code OD_{16}) will cause an immediate return to the window boundary in the direction opposite to the primary update direction as well as an update in the secondary update direction as well as an update in the secondary update direction. The line feed character (Code OA_{16}) will cause only an immediate update in the secondary update direction. When the RM-FNT option is not installed, all characters in the ranges 00 through $1F_{16}$ and 60_{16} through FF_{16} are treated as legal characters in the sense that a character is generated into refresh memory, however, the font which is used for these characters is indeterminate. When the RM-FNT option is installed, all characters (other than carriage return or line feed) in the ranges 00 through $1F_{16}$ and AO_{16} through FF_{16} are generated in the same manner.

PERTINENT CONTROL BITS

IX — Defines the address mode in which the parameter operands INDEX 1, INDEX 2, ORIGIN, WINDOW, BASELINE and START-POINT will be evaluated and converted to absolute refresh memory addresses.

AD	Affects the generation of text data in conjunction with the BK flag. Characters are generated using a font consisting of one's and zeros. The AD flag inhibits the writing of pixels within the character font in a manner specified by Table 3-15 (See Section 3-10).
BK	Defines the selection of FOREGROUND and BACKGROUND colors based on the ones/zeros data bits within the character font and on the value of the AD flag (see above). If BK = 0, the FOREGROUND value is selected for 'ones' font data and the BACKGROUND value is selected for 'zeros' font data.
RP	Defines the byte-accessing order for ASCII characters within each 16-bit word. If RP = 0, the characters will be accessed high-byte first; if RP = 1, the low byte will be accessed before the high byte.

PERTINENT PARAMETER OPERANDS

SUBCHANNEL	Specifies the subchannels (i.e., bit planes in refresh memory) which are write-enabled and which will receive text data.
FOREGROUND	Defines the foreground register color or intensity value to be written to refresh memory, in combination with the BACKGROUND operand and the AD and BK flags (See Table 3-3).
BACKGROUND	Defines the background register color or intensity value to be written to refresh memory, in combination with the FOREGROUND operand and the AD and BK flags (See Table 3-3).
INDEX 1	Displaces the values to be used for INDEX 2, WINDOW, ORIGIN and START-POINT parameter operands set in the WT instruction when IX = 01.
INDEX 2	Displaces the values to be used for WINDOW and START-POINT parameter operands set in the WT instruction when IX = 10.
WINDOW	Defines the rectangular area into which text will be written. If the first pixel of any character is within this region, the

entire character will be drawn, even though part of the character may exceed the WINDOW boundaries.

SCAN	Defines the primary and secondary text update directions as well as the pixel coordinate when START-POINT is not explicitly defined in the WT instruction (See Figure 3-5).
DIMENSION	Defines the character font size independent of SCAN direction or character orientation.
SPACING	Defines the character-to-character increments for the primary and secondary text updates. (See Figure 3-5) for legal SPACING values for the various text modes.
SCALE	Defines the received-pixel-to-displayed-pixel ratio when the RM-SCA option is installed (See Table 3-9).
START-POINT	Specifies the coordinates of the first pixel of the first text character to be written into refresh memory. START-POINT must be contained within the current WINDOW region.

DATA LENGTH WORD

The DATA LENGTH WORD represents the number of ASCII characters which will be transmitted from the host processor to the RM-9000 with a WT instruction on a one (1) byte per character basis. Since text data is byte-oriented, the DATA LENGTH WORD may take on any value from 0 to 65535_{10} .

DATA FORMAT

Data in text mode is interpreted on a byte basis, i.e., two (2) characters per word. The order that the bytes are referenced is determined by the RP flag. When the RM-FNT option is not installed, the legal character set consists of Codes 20_{16} through $5F_{16}$ and OA_{16} (line feed) and OD_{16} (carriage return). When the RM-FNT option is installed, the legal character set consists of Codes 20_{16} through $9F_{16}$ as well as OA_{16} and OD_{16} .

3-39 Special-Format Standard Instructions

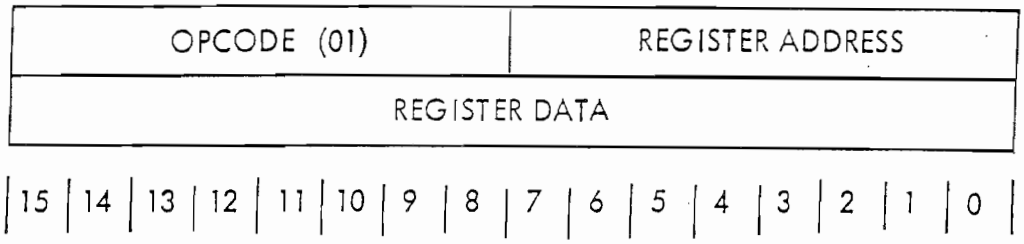
Sections 3-40 through 3-45 define the formats and functions of the special-format instructions which are supported by the RM-9000 standard firmware package. The instructions in this set are:

LOAD	-	Load Hard Register Instruction
READ	-	Read Soft Register Instruction
LAM	-	Load Auxiliary Memory Instruction
RAM	-	Read Auxiliary Memory Instruction
RSET	-	Reset Instruction
INIT	-	Initialize Instruction

These instructions are standard in all RM-9000 systems.

ramtek

3-40 Load Hard Register Instruction (LOAD)



The LOAD instruction is a special format instruction which is part of the RM-9000 standard firmware instruction set. The function of the LOAD instruction is to move the sixteen (16) bits of data, stored in the second word of the instruction, directly into the RM-9000 internal display register whose address is specified in the low-order byte of the opcode word. This instruction allows the user to bypass the overhead in execution time necessary to perform various display operations using the normal format instructions. The information which is written into the specified RM-9000 display register is not retrievable by the user. In addition, the direct storage of data into a display register using LOAD may cause the normal format instructions to malfunction. Since the normal format instruction software assumes that the state of all internal display registers can be modified only by a normal format instruction, the functioning of any normal format instruction after the execution of a LOAD can only be guaranteed if all normal format instruction parameters are reset within a normal format instruction. The use of the LOAD instruction requires extensive knowledge of the operation of the internal display registers, and as such its use is not encouraged. The function of the various RM-9000 display registers is defined in Volume I of the RAMTEK 9000 SERIES THEORY OF OPERATION.

PERTINENT OPERANDS

REGISTER ADDRESS	Defines the low-order 8 bits of the internal address of the desired display register. Table 3- defines the register address assignments.
REGISTER DATA	Defines the 16-bit value to be loaded into the display register defined by REGISTER ADDRESS. The format of the data word is specific and different for each register and may be found in VOLUME I of the RAMTEK 9000 SERIES THEORY OF OPERATION.

POSSIBLE ERRORS

If REGISTER ADDRESS is set to an undefined value (i.e., an address for which no display register exists), the display produced will be indeterminate. No register address error checking is performed.

Table 3-11 I/O Device Register Address Map (RM-9000 Series)

HEX ADDRESS	MNEMONIC	DESCRIPTION
00	SCMSKR	Memory Plane Select Register
02	ROPTRG	Interrupt Register
04	READR	Readback Register
06	XORGR	X Origin Register
08	YORGR	Y Origin Register
0A	NPRSRC	DMA Source Register
0C	NPRDST	DMA Destination Register
0F	UPDTR	COP/Write Control Register
10	BGR	Background Register
12	FGR	Foreground Register
14	XCOPR	Element COP Register
16	YCOPR	Line COP Register
18	RASREG	Raster Font Register
1A	ID	Interface Data Register
1C	SYSS	System/Interface Status Register
1E	WRDCNT	DMA Word Count Register
20	DMAADR	DMA Address Register
23	MOVWTR	Move and Write Register
25	MOVR	Move (W. No Write) Register

Table 3-11 (Continued)

HEX ADDRESS	MNEMONIC	DESCRIPTION
26	DPSW	Dip Switch Register
28		
2A		
2C		
2E		
30		
32		
34		
36		
38		
3A		
3C	SPS1	* Serial Peripherals Status Register (Board 1)
3E	SPS2	* Serial Peripherals Status Register (Board 2)
40	SPK11	* Serial Peripherals Keyboard #1 I/O Register (Board 1)
42	SPK12	* Serial Peripherals Keyboard #2 I/O Register (Board 1)
44	SPK13	* Serial Peripherals Keyboard #3 I/O Register (Board 1)
46	SPK14	* Serial Peripherals Keyboard #4 I/O Register (Board 1)

Table 3-11 (Continued)

HEX ADDRESS	MNEMONIC	DESCRIPTION
48	SPCX11	* Serial Peripherals Cursor #1 X-Position Register (Board 1)
4A	SPCY11	* Serial Peripherals Cursor #1 Y-Position Register (Board 1)
4C	SPCX2	* Serial Peripherals Cursor #2 X-Position Register (Board 1)
4E	SPCY2	* Serial Peripherals Cursor #2 Y-Position Register (Board 1)
50	SPK21	* Serial Peripherals Keyboard #1 I/O Register (Board 2)
52	SPK22	* Serial Peripherals Keyboard #2 I/O Register (Board 2)
54	SPK23	* Serial Peripherals Keyboard #3 I/O Register (Board 2)
56	SPK24	* Serial Peripherals Keyboard #4 I/O Register (Board 2)
58	SPCX21	* Serial Peripherals Cursor #1 X-Position Register (Board 2)
5A	SPCY21	* Serial Peripherals Cursor #1 Y-Position Register (Board 2)
5C	SPCX22	* Serial Peripherals Cursor #2 X-Position Register (Board 2)
5E	SPCY22	* Serial Peripherals Cursor #2 Y-Position Register (Board 2)

Table 3-11 (Continued)

HEX ADDRESS	MNEMONIC	DESCRIPTION
60	VLTAD0	* Lookup Table #0 Address Register
62	VLTD0	* Lookup Table #0 Programmed Data Register
64	VLTAD1	* Lookup Table #1 Address Register
66	VLTD1	* Lookup Table #1 Programmed Data Register
68	VLTAD2	* Lookup Table #2 Address Register
6A	VLTD2	* Lookup Table #2 Programmed Data Register
6C	VLTAD3	* Lookup Table #3 Address Register
6E	VLTD3	* Lookup Table #3 Programmed Data Register
		* May not exist, depending on system configuration.

3-41 Read Soft Register Instruction (READ)

OPCODE (02)	REGISTER ADDRESS

The READ instruction is a special format instruction which is contained in the standard RM-9000 firmware package. The function of READ is to return the current value of one of seven (7) internal registers used by the normal-format instruction firmware package. These register values reflect the state of the actual hardware registers as loaded and updated by the firmware. These values are not affected when the associated hardware display register is loaded via the LOAD instruction. The data format of the readback value is identical to that of the associated hardware register, and may be found in VOLUME I of the RAMTEK 9000 SERIES THEORY OF OPERATION manual.

Following the output of the READ instruction, the user (i.e., host computer) must read back exactly one (1) 16-bit word from the computer interface; output of succeeding instructions may then proceed. If no read back is done and an attempt is made to output instructions, the RM-9000 interface and display system will hang and can only be restored with a hard system reset. Therefore, it is necessary to maintain synchrony with the RM-9000 in terms of the direction of data transfers.

PERTINENT OPERANDS

REGISTER ADDRESS — Defines the low-order byte of the internal display register address to be read back. Table 3-12 defines the valid register addresses which may be accessed.

POSSIBLE ERRORS

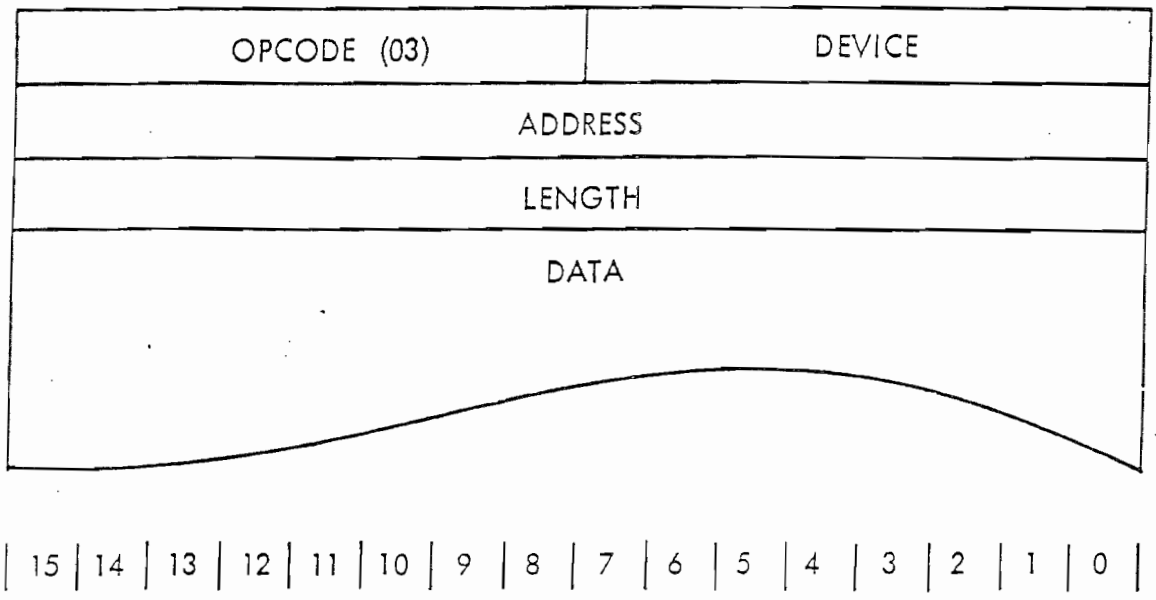
If the REGISTER ADDRESS specified in the low-order byte of the opcode word is a value not defined in Table 3-12, the read-back value will be meaningless.

If the host processor does not attempt to read back a word from the RM-9000, the system will hang and can only be restored by a hard system reset.

Table 3-12 Read Soft Register Addresses

REGISTER ADDRESS	REGISTER
00 (HEX)	SUBCHANNEL MASK
02	X (ELEMENT) COP
04	Y (LINE) COP
06	X (ELEMENT) ORIGIN
08	Y (LINE) ORIGIN
12	FOREGROUND
14	BACKGROUND

3-42 Load Auxiliary Memory Instruction (LAM)



The LAM instruction is a special-format instruction which is included in the RM-9000 standard firmware package. The function of the LAM instruction is to transfer data to auxiliary memory devices within the RM-9000 system other than refresh memory or internal 8080 memory from the host processor. In the current RM-9000 series implementation, the LAM instruction is used to write data to the video lookup tables on the Type II video boards if one of these devices exists within a given system. A description of the data format for the Type II video board is found in Appendix A. The value of DEVICE may range from 0 to 3.

PERTINENT OPERANDS

DEVICE — Defines the auxiliary memory device to which the data will be written.

ADDRESS — Specifies a 16-bit address within the auxiliary memory device specified by DEVICE. This address is device-dependent, and in the case of the Type II video cards, it represents the starting video look-up table address at which data will be written.

Table 3-13 Video Look-Up Table Addressing

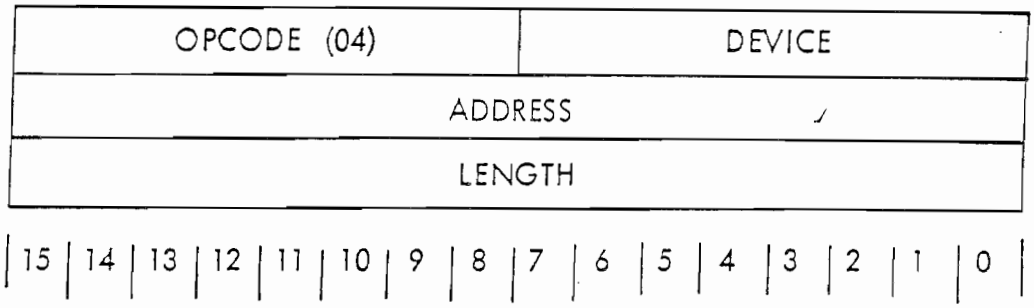
DEVICE	LOOK-UP TABLE DEFINITION
0	Type II Video Look-up Table 0 (with internal addresses 60_{16} and 62_{16})
1	Type II Video Look-up Table 1 (with internal addresses 64_{16} and 66_{16})
2	Type II Video Look-up Table 2 (with internal addresses 68_{16} and $6A_{16}$)
3	Type II Video Look-up Table 2 (with internal addresses $6C_{16}$ and $6E_{16}$)

LENGTH Defines the number of bytes of data to be written to the auxiliary memory device.

DATA FORMAT

The format of the data associated with a LAM instruction is device dependent. Appendix A defines the format for the Type II video board.

3-43 Read Auxiliary Memory Instruction (RAM)



The RAM instruction is a special-format instruction which is included in the RM-9000 standard firmware package. The function of the RAM instruction is to transfer data from an auxiliary memory device within the RM-9000 system to the host processor. In the current RM-9000 series implementation, the RAM instruction is used to read the contents of the video look-up tables on the Type II video board if one of these devices exists within a given system. A description of the data format to be read back by the host processor is found in Appendix A. The value of DEVICE may range from 0 through 3.

Following the output of the LENGTH word, it is necessary to change the sense of the computer interface board from a write mode to read mode. The number of bytes as specified by LENGTH must then be read back into the host computer. If this is not done, the handshaking sequence between the RM-9000 and the host processor will lose synchronization and it will be necessary to perform a hard system reset to restore communications.

PERTINENT OPERANDS

DEVICE — Defines the auxiliary memory device from which the data will be read back. (See Table 3-13)

ADDRESS — Specifies a 16-bit address within the auxiliary memory device specified by DEVICE. This address is device-dependent, and in the case of the Type II video cards, it represents the starting video look-up table address from which data will be read back.

LENGTH — Defines the number of bytes of data to be read back from the auxiliary memory device. This value must be an even number since data is transferred across the interface on a 16-bit basis. If LENGTH is an odd number, the value used will be LENGTH-1.

DATA FORMAT

The format of the data associated with a RAM instruction is device dependent. Appendix A defines the format for the Type II video board.

POSSIBLE ERRORS

If the host processor does not read back the number of bytes defined in LENGTH, the system will hang and can only be restored by a hard system reset.

If the value in LENGTH is odd, the value used as a count will be LENGTH minus one (1) and an I/O direction conflict may arise (See above).

famtek

3-44 Reset Instruction (RSET)

OPCODE (05)	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The RSET instruction is a special-format instruction which is included in the RM-9000 standard firmware package. This instruction checksums all PROM-resident memory, clears all interrupt lines from the processor to the interface board, clears the RM-STA status management option stack, clears the RM-PER interactive peripherals option keyboard input buffers and resets the UART devices on the RM-9000 serial link board, sets all cursors to invisible at screen location (0,0), copies the standard ASCII character fonts (hex character Codes 20-5F), and sets all normal-format instruction parameter operands to their default values (See Table 3-14 for these values). If DIP SWITCH 3 is set to Ø (down on the '+' side of the switch), the entire refresh memory will be set to the BACKGROUND parameter operand default value, i.e., a value of 0; otherwise, no refresh memory erasing is performed. The RM-9000 system is then ready to accept the next instruction. If the toggle switch mounted on the side of the control board is in the DIAG position and a RSET instruction is received, the RM-DOC diagnostic tester firmware takes control (See Figure 3-7).

PERTINENT OPERANDS None

POSSIBLE ERROR CONDITIONS None

Table 3-14 Operand Parameter Default Values

OPERAND PARAMETER	DEFAULT VALUE	DESCRIPTION
SUBCHANNEL	$\emptyset FFF_{16}$	Write-Enable all subchannels
FOREGROUND	$\emptyset FFF_{16}$	1's Data to all subchannels
BACKGROUND	ØØØØ	Ø 's Data to all subchannels
INDEX 1	Ø, Ø	INDEX 1 X = 0, Y = 0
INDEX 2	Ø, Ø	INDEX 2 X = 0, Y = 0
ORIGIN	X_{OR}, Y_{OR}	ORIGIN X and Y values defined on a system-resolution basis
WINDOW	Ø, Ø, X_{RES}, Y_{RES}	Set WINDOW boundaries to full-screen size based on the system resolution
SCAN	Ø	Left-to-right, top-to-bottom
DIMENSION	7, 9	7 Elements wide, 9 lines high character font
SPACING	7, 9	7 Elements wide, 9 lines high spacing between characters or plot entities
SCALE	Ø	No scaling
FUNCTION	Ø	No logical/arithmetic processing
CONIC-EQUATION	6 Ø, Ø	Conic parameters A, B, C, D, E, K = Ø
BASELINE	Ø	Line plotting
SCROLLCOUNT	Ø	No Scroll
START POINT	Ø, Ø	XCOP = Ø, YCOP = Ø

Table 3-15 (Continued

SYSTEM	X_{OR}	Y_{OR}	X_{RES}	Y_{RES}
9100	136_{16}	$ØFF_{16}$	$13F_{16}$	$ØFF_{16}$
9200	$26C_{16}$	$ØFF_{16}$	$27F_{16}$	$ØFF_{16}$
9300	$26C_{16}$	$1FE_{16}$	$27F_{16}$	$1FF_{16}$

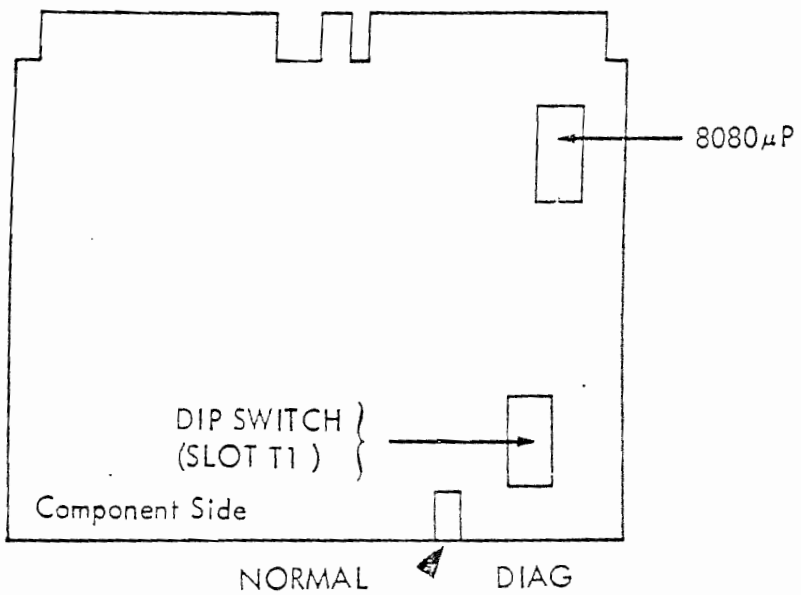


Figure 3-7 RM-9000 Control Board

3-45 Initialize Instruction (INIT)

OPCODE (06)	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The INIT instruction is a special-format instruction which is included in the RM-9000 standard firmware. The INIT instruction resets the normal-format instruction operand parameters to their default values (See Table 3-14). It is equivalent to issuing a SET instruction (See 3-42) with all parameter operand default values present in the instruction itself. The status management stack is not affected, thus, any previous PUSHE operations will still be available using the POPE instruction (See Sections 3-57 and 3-58).

PERTINENT OPERANDS

None

POSSIBLE ERROR CONDITIONS

None

3-46 OPTIONS FIRMWARE INSTRUCTION

The options firmware instruction set provides the user with additional display and display support functions. Many of the functions, such as the drawing of vectors or conics, could be performed by the host processor; however, these operations would be prohibitive in terms of host processor software overhead. By using the firmware options instructions, it is possible to distribute the display generation process more evenly between the host processor and the display system.

All of the firmware supporting the firmware options instruction set is resident on the RM-MOC memory expansion card in the form of PROM. In order to enable any of the firmware options, it is necessary that Switch 2 of the DIP SWITCH REGISTER on the control board is set to the 'on' position, i.e., up on the '+' side of the switch (See Figure 3-7). It is possible to disable all options' instruction processing by setting Switch 2 of the DIP SWITCH REGISTER to the 'OFF' position. No removal of the RM-MOC card is necessary. If, however, Switch 2 is set to the 'ON' position, it is necessary that the RM-MOC card as well as the RM-STD standard option support PROM be present in the system. If the RM-STD firmware is not present on the RM-MOC in the proper slot, the system will never leave self-test and will be hung until this condition is corrected and a hard system is issued.

The purpose of the RM-STD firmware is to provide a common interface between the RM-9000 standard and optional instruction set. The RM-STD firmware determines by inspection which options package(s) are actually installed in any given system. A section of this firmware is called during the firmware reset sequence to call any existing option power-on/reset routine to initialize all internal parameters. The RM-STD firmware allows the modularity of the various options' firmware packages. This package resides within the internal 8080 processor's address space at locations 0800_{16} through $OBFF_{16}$.

3-47 RM-GRA Graphics Option Instructions

The RM-GRA graphics option firmware implements the following instructions:

- WRITE VECTOR generating linked end-point vectors
- WRITE PLOT generating linked plots and filled bar plots with a fixed plot axis
- WRITE RASTER generating display data directly to refresh memory on a bit-per-pixel basis.

The RM-GRA option resides within the internal 8080 processor's address space at locations 2800_{16} through $2BFF_{16}$. The RM-GRA graphics option is a prerequisite for the RM-CON option (See Section 3-51).

parameters. The length value defines the positive number of bytes of vector endpoint information to be used by this instruction; therefore, length must be four (4) times the number of vectors to be drawn.

PERTINENT CONTROL BITS

IX	Defines the address mode in which the INDEX 1, INDEX 2, and START-POINT parameter operands (i.e., those operands directly affecting the absolute position of any vectors drawn) are evaluated (See Section 3-11).
BK	Causes the vector(s) to be drawn in the BACKGROUND color or intensity, rather than the FOREGROUND color or intensity (when BK = Ø).

PERTINENT ARGUMENTS

SUBCHANNELS	Defines which combination of memory planes will actually receive vector color or intensity data. The value actually written to refresh memory is the logical 'and' of SUBCHANNELS with foreground or background dependent on the state of BK.
FOREGROUND	Defines the vector color or intensity in conjunction with SUBCHANNELS when BK = Ø.
BACKGROUND	Defines the vector color or intensity in conjunction with SUBCHANNELS with BK = 1. INDEX 1 displaces all WV coordinate pairs as well as the INDEX 2 and START-POINT operands (when set in the same WV instruction) when IX = 01. INDEX 2 displaces all WV coordinate pairs as well as the START-POINT operand (when set in the same WV instruction) when IX = 10.
START-POINT	Defines the starting coordinate for the first vector to be drawn. If not specified, the starting coordinate used is the COP resulting from the previous display instruction.

DATA FORMAT

The format of the vector end-point value is as follows: Each successive pair of 16-bit words defines the X and Y coordinate for the next vector endpoint. The first 16-bit word contains the X-value, and the second 16-bit word contains the Y-value. The interpretation of this coordinate is defined by the value of IX in the opcode word (See Section 3-11 for the possible addressing modes).

POSSIBLE ERROR CONDITIONS

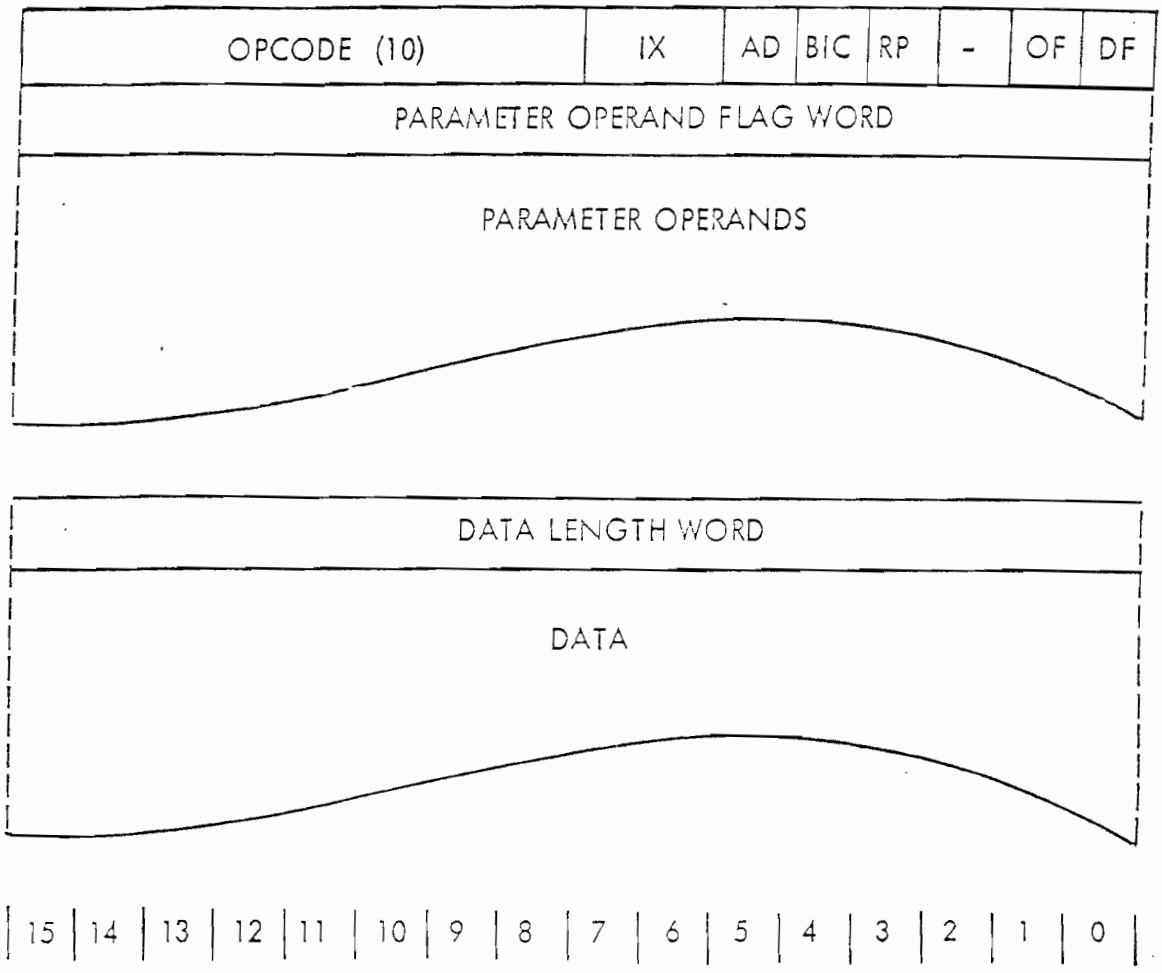
The following conditions are invalid and undefined, and will result in an unpredictable display if attempted for the WV instruction:

- If any of the specified endpoints exceeds the screen resolution, the generated vector(s) will be indeterminate.

COP MOVEMENT

After the successful execution of this instruction, the COP will be at point (X_n , Y_n) where N is the number of endpoints specified by the WV instruction.

3-49 Write Plot Instruction (WP)



The WP instruction is a normal format instruction which is part of the RM-GRA graphics firmware option. The WP instruction will automatically update one (1) COP while receiving the other. This can reduce data transfers significantly. The WP instruction generates one (1) plot segment for each 16-bit word of data present in the WP instruction using a raster-data technique. The orientation of the plot segment (i.e., differentiating between X-axis plots and Y-axis plots) is defined by SCAN. A plot segment's height (i.e., dimension perpendicular to the plot axis as defined by SCAN) is determined by the current-operating point and a 16-bit data word from the instruction. If BASELINE is zero (i.e., line plot mode), then initially the COP is either set to the START-POINT value if specified in the WP instruction or retains the COP value from the previous instruction. If BASELINE is non-zero, then initially the plot-axis COP coordinate is set to the BASELINE value. For example, if SCAN is zero (i.e., Y-axis plot), then the line COP will be set to the BASELINE value. The width of a

plot segment (i.e., dimension parallel to the plot axis as defined by SCAN) is defined by the DIMENSION parameter operand parallel to the plot axis. After each plot segment is drawn, the COP coordinate parallel to the plot axis is updated by the SPACING parameter operand parallel to the plot axis. If BASELINE is zero, the COP coordinate perpendicular to the plot axis is unchanged (i.e., plot segments in linked mode will overlap by one (1) pixel in this axis). If BASELINE is non-zero, the COP coordinate perpendicular to the plot axis is set to BASELINE after each plot segment is drawn.

The color or intensity of the plot segments is defined by either FOREGROUND (when BK = 0) or by BACKGROUND (when BK = 1) and is masked by the write-enable parameter operand SUBCHANNEL. The WRITE PLOT instruction is not affected by the WINDOW parameter operands.

PERTINENT CONTROL BITS

IX	Defines the address mode in which the INDEX 1, INDEX 2, BASELINE and START-POINT parameter operands (i.e., those operands directly affecting the absolute position of the plot axis or the plot data) are evaluated (See Section 3-11).
BK	Defines the color or intensity used for the plot segments drawn. If BK = 0, the FOREGROUND value will be used; if BK = 1, the BACKGROUND value will be used.

PERTINENT PARAMETER OPERANDS

SUBCHANNEL	Specifies the subchannels (i.e., refresh memory bit planes) which are write-enabled and which will receive plot data.
FOREGROUND	Defines the color or intensity value for all plot segments when BK = 0.
BACKGROUND	Defines the color or intensity value for all plot segments when BK = 1.
INDEX 1	Displaces all WP data values as well as INDEX 2, BASELINE and START-POINT operands (when set in the same WP instruction) when IX = 01.

INDEX 2	Displaces all WP data values as well as BASELINE and START-POINT operands (when set in the same WP instruction) when IX = 10.
SCAN	Defines the plot orientation. If SCAN = 0, the plot axis will be the Y-axis (i.e., horizontal plot); if SCAN = 4, the plot axis will be the X-axis (i.e., vertical plot).
DIMENSION	Defines the size of a plot segment parallel to the plot axis. If SCAN = 0, DIMENSION width is used; if SCAN = 4, DIMENSION height is used.
SPACING	Defines the distance between starting COP for successive plot segments along the plot axis. If SCAN = 0, X-spacing is used; if SCAN = 4, Y-spacing is used.
BASELINE	Defines the fixed-line plot axis if BASELINE is non-zero. If SCAN = 0, BASELINE represents the Y-coordinate of the plot axis; if SCAN = 4, BASELINE represents the X-coordinate of the plot axis.
START-POINT	Defines the initial COP for the first plot segment if BASELINE = 0; if BASELINE is non-zero, START-POINT defines the starting coordinate of the first plot segment along the plot axis.

DATA LENGTH WORD

The DATA LENGTH WORD represents the number of bytes of plot data to be transmitted from the host processor to the RM-9000. Since each plot segment is represented by a single 16-bit word, the DATA LENGTH WORD should always reflect an even number of bytes. If the DATA LENGTH WORD is odd, the byte count used will be one less than the actual DATA LENGTH WORD received.

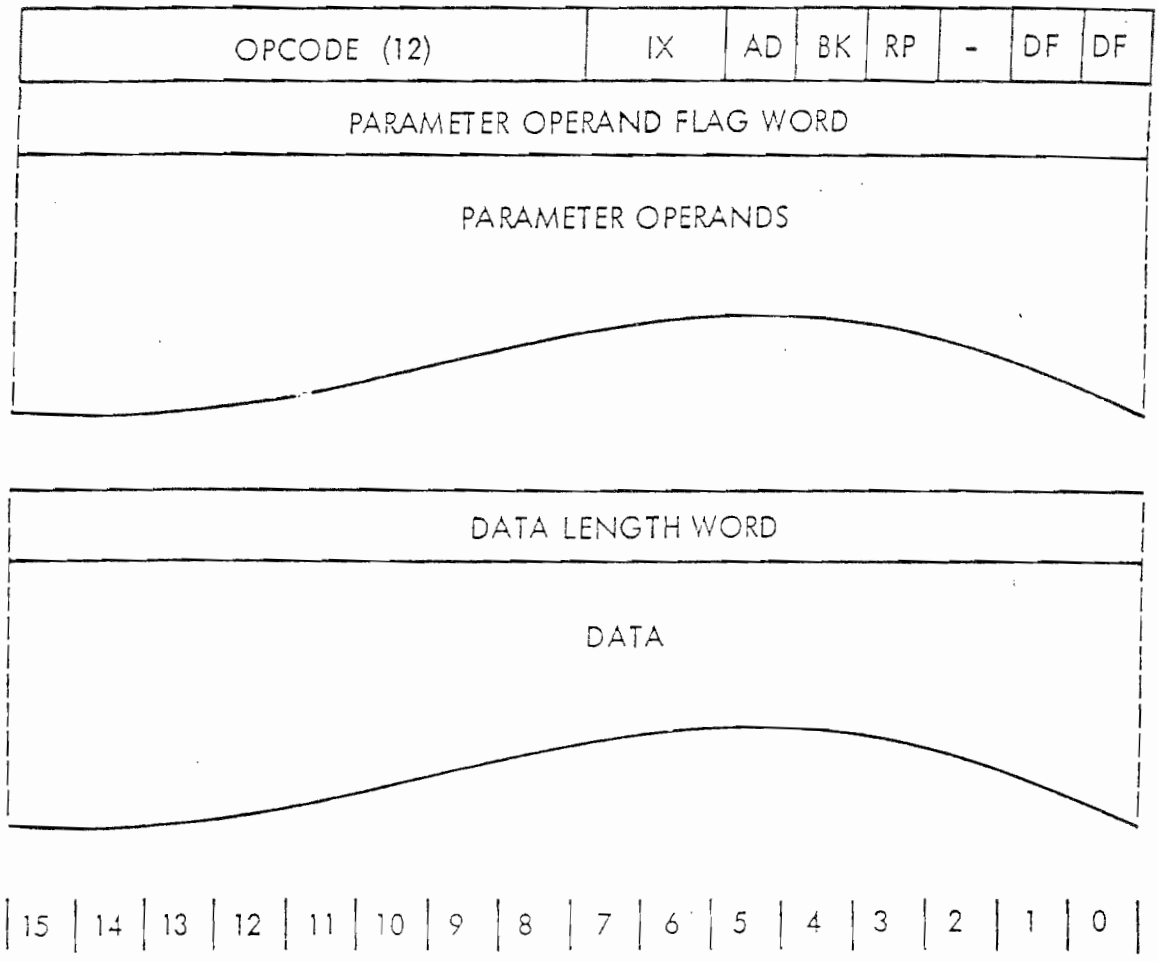
DATA FORMAT

Data in plot mode is interpreted on a word-per-plot segment basis. Each word represents the coordinate of a plot segment (as evaluated in light of the IX value) perpendicular to the plot axis.

COP MOVEMENT

After execution of the WP instruction the new COP reflects the starting coordinate of the next plot segment if one (1) additional data word has been specified in the WP instruction.

3-50 Write Raster Instruction (WR)



The WR instruction is a normal format instruction included in the RM-GRA graphics option firmware. The WR instruction interprets data from the host processor and writes either the FOREGROUND or BACKGROUND value into refresh memory as a function of the incoming data bytes (on a bit-per-pixel basis) and the AD and BK flags. The WR instruction operates in a manner analogous to the WT instruction except that the data for the character generation is obtained from RM-9000 internal memory, whereas, the WR data is obtained directly from the host processor.

When a byte of data is received from the host processor by the RM-9000, each of the eight (8) bits represents a pixel in refresh memory. The value which is actually written to refresh memory is defined by Table 3-15. Within each byte of data, the bits are evaluated from high to low order.

Table 3-15 Additive (AD), Background (BK) & Data Bit Relation

BK \ AD	0 0	0 1	1 0	1 1
0	B	F	X	F
1	F	B	X	B

F - Write Foreground Value
B - Write Background Value
X - No Write

The WR instruction operates within the rectangular region defined by the WINDOW parameter operands. The SCAN parameter defines the primary (pixel-to-pixel) update direction and the secondary (window boundary intersection) update direction in a manner identical to that of image mode (See Table 3-6). In the case of a zero data bit combined with AD = 1 (i.e., the 'no write' condition), pixel-to-pixel is still performed even though no writing into refresh memory is performed. When the window boundary is intersected, any bits remaining in the data byte current at the intersection are discarded; therefore, after any secondary update, the first pixel value written is defined by the high-order bit of the next data byte.

When the RM-SCA scaling option firmware is installed and a scaling-down operation is specified by the SCALE parameter operand in either dimension, Bits 7, 5, 3 and 1 are used for a ratio of 1:2 and Bits 7 and 3 are used for a ratio of 1:4.

PERTINENT CONTROL BITS

IX — Defines the address mode in which the INDEX 1, INDEX 2, WINDOW and START-POINT parameter operands set in the WR instruction will be evaluated.

AD — Affects the generation of raster data in conjunction with the BF flag. Input data is inhibited in the manner specified by Table 3-15.

BK — Defines the selection of FOREGROUND and BACKGROUND colors based on the one's/zeros data bits within the raster byte and on the value of the AD flag (See Table 3-15). If BK = 0, the FOREGROUND is selected for 'ones' raster

data and the BACKGROUND value is selected for 'zeros' raster data (assuming AD = 0). If BK = 1, the FOREGROUND and BACKGROUND selection is reversed.

RP — Defines the byte-accessing order for each 16-bit word of raster data from the host processor. If RP = 0, the high-byte of the data word is processed before the low-byte; if RP = 1, the low-byte is processed first.

PERTINENT PARAMETER OPERANDS

SUBCHANNEL — Specifies the subchannels (i.e., refresh memory bit planes) which are write-enabled and which will receive raster data.

FOREGROUND — Defines the foreground color or intensity value to be written to refresh memory when selected by the raster data bit value and the AD and BK flags (See Table 3-15).

BACKGROUND — Defines the background color or intensity value to be written to refresh memory when selected by the raster data bit value and the AD and BK flags (See Table 3-15).

INDEX 1 — Displaces the values to be used for INDEX 2, WINDOW and START-POINT parameter operands set in the WR instruction when IX = 01.

INDEX 2 — Displaces the values to be used for WINDOW and START-POINT parameter operands set in the WR instruction when IX = 10.

WINDOW — Defines the rectangular region into which the raster data will be written. No raster data will be written outside of the boundaries defined by this parameter operand.

SCAN — Defines the primary and secondary raster update directions as well as START-POINT when START-POINT is not specified in the WR instruction (See Table 3-6).

SCALE — Defines the received-bit-to-displayed-pixel ratio for raster data when the RM-SCA option is installed (See Section 3-24, Table 3-9).

START-POINT — Specifies the coordinates of the first pixel to be written with raster data START-POINT must be contained within the current WINDOW region.

DATA LENGTH WORD

The DATA LENGTH WORD represents the number of bytes of raster data to be transmitted from the host processor to the RM-9000 display system. Since raster data is byte oriented, the DATA LENGTH WORD may take on any value from 0 through 65535_{10} .

DATA FORMAT

Data in raster mode is interpreted on a byte basis, (i.e., two (2) bytes per data word transferred across the RM-9000 interface). The order that the bytes are referenced is determined by the RP flag. When RP = 0, the high byte will be used first; when RP = 1, the low byte will be used first. Within a data byte, the high order Bit 7 is used first, while Bit 0 will be referenced last. If a WINDOW boundary is encountered before all bits of a raster data byte are used, the remaining bits of that byte will be discarded.

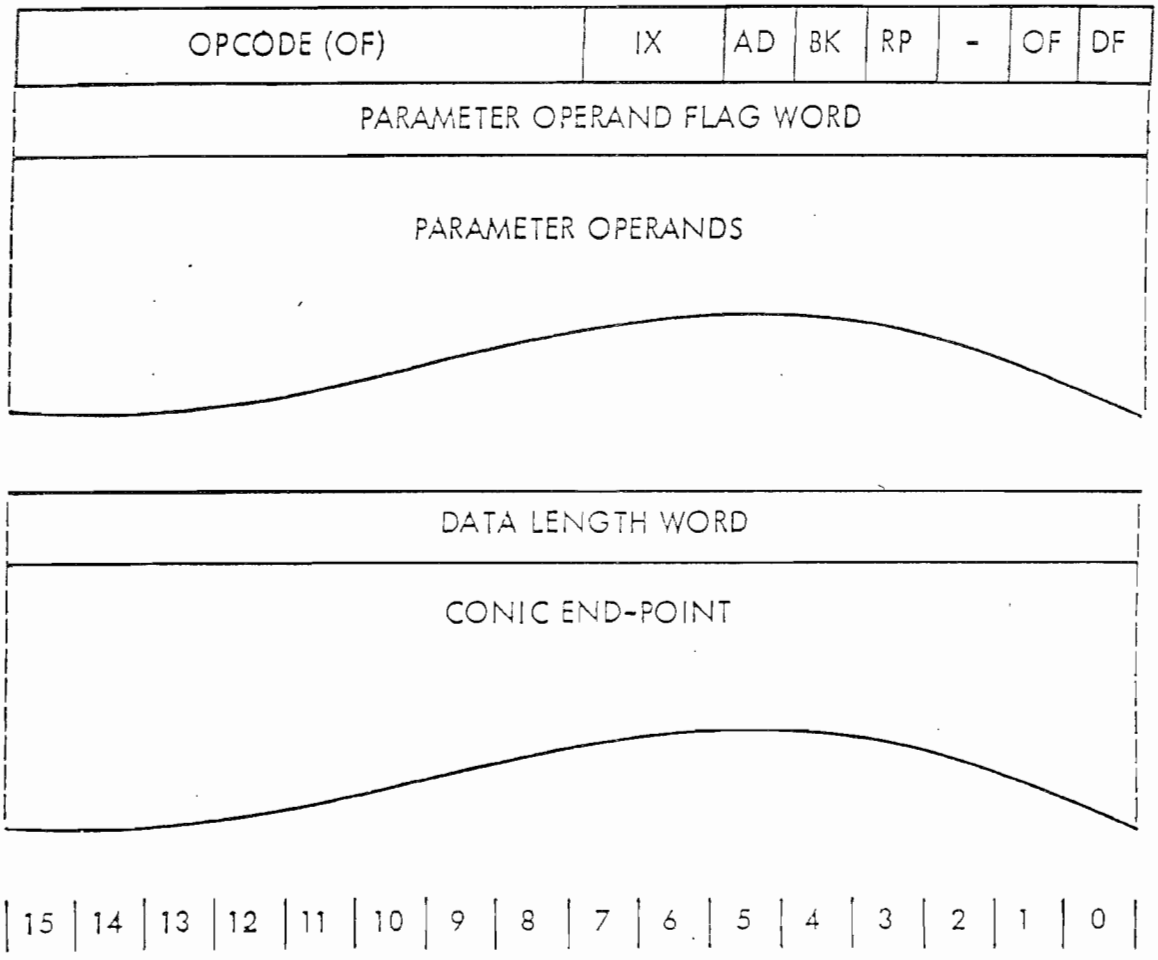
3-51 RM-CON Conics Option Instruction

The RM-CON conics option firmware implements the WRITE CONIC (WC) instruction for generation of parabolas, hyperbolas, ellipses, and circles. The generalized conic equation used in this implementation is

$$Ax^2 + By^2 + Cxy + Dx + Ey = 0.$$

The RM-GRA graphics option firmware is a prerequisite for installation of the RM-CON conics option firmware package. The RM-CON firmware resides within the internal 8080 processor's address space at locations $2C00_{16}$ through $33FF_{16}$.

3-52 WRITE CONIC Instruction (WC)



The WC instruction is a normal-format instruction which is part of the RM-CON conics firmware option. The WC instruction draws circles, parabolas, ellipses and hyperbolas or sections thereof, based on the CONIC-EQUATION parameter operands. The generalized conic equation used is

$$Ax^2 + By^2 + Cxy + Dx + Ey = 0$$

where A, B, C, D and E represent the first 5 CONIC-EQUATION parameter operands. The last CONIC-EQUATION parameter operand K represents the total number of pixels to be generated by the WC instruction for the conic specified. A default value of 1280_{10} is used for K.

The START-POINT parameter operand is used by the WC instruction to define the Cartesian origin of the conic or conic section to be drawn. All conics drawn will pass through the Cartesian origin defined by START-POINT. If a conic end-point is specified, the absolute coordinates of this pixel are compared to the coordinates of each pixel in the conic as it is being generated. If a pixel in the conic being drawn has the same coordinates as specified by the end-point, the conic will be terminated at that point.

The WC instruction is not affected by the WINDOW parameter operand.

Appendix B describes in detail the use of the WC instruction for generation of circles, ellipses, parabolas and hyperbolas.

PERTINENT CONTROL BITS

IX	defines the address mode in which the INDEX 1, INDEX 2 and START-POINT parameter operands set in the WC instruction are evaluated. Also, if a conic end-point is specified as data in the WC instruction, IX effects the address in which this coordinate will be evaluated.
BK	defines the selection of FOREGROUND or BACKGROUND value for the color or intensity value of the conic being drawn. If BK = 0, the conic will be generated with the FOREGROUND value. Otherwise, the BACKGROUND value will be used when BK = 1.

PERTINENT PARAMETER OPERANDS

SUBCHANNEL	specifies the subchannels (i.e., refresh memory bit planes) which are write-enabled to receive conic data.
FOREGROUND	defines the color or intensity value of the conic when BK = 0.
BACKGROUND	defines the color or intensity value of the conid when BK = 1.
INDEX 1	displaces the values to be used for the INDEX 2 and START-POINT parameter operands, as well as the conic

	end-point if any of these are set in the WC instruction and if IX = 01.
INDEX 2	displaces the values to be used for the START-POINT parameter operand, as well as the conic end-point if any of these are set in the WC instruction and if IX = 10.
CONIC-EQUATION	specifies the conic-defining coefficients A, B, C, D and E and the number of pixels to be drawn in the conic, K.
START-POINT	defines the origin of the conic to be generated. START-POINT is evaluated in the address mode defined by IX.

DATA LENGTH WORD

The DATA LENGTH WORD defines the total number of bytes contained in the WC following the DATA LENGTH WORD. Since only one conic can be generated per WC instruction, it is suggested that the DATA LENGTH WORD (if present due to DF = 1) always takes on a value of four (4), indicating that a single conic end-point is present in the WC instruction data stream. If more the four (4) bytes are indicated by the DATA LENGTH WORD, all additional bytes will be ignored. Since coordinates are composed of full-words, it is necessary that the DATA LENGTH WORD always reflects an even number of bytes.

DATA FORMAT

The first 16-bit word pair is interpreted as the conic end-point. All remaining data is discarded. The first 16-bit word is the element coordinate of the conic end-point. The second is the line coordinate. The conic end-point coordinate is interpreted in the address mode defined by IX.

POSSIBLE ERROR CONDITIONS

It is possible to generate a conic equation from the coefficients A, B, C, D and E which will not be drawn accurately. The conditions for this case are defined in Appendix B.

COP MOVEMENT

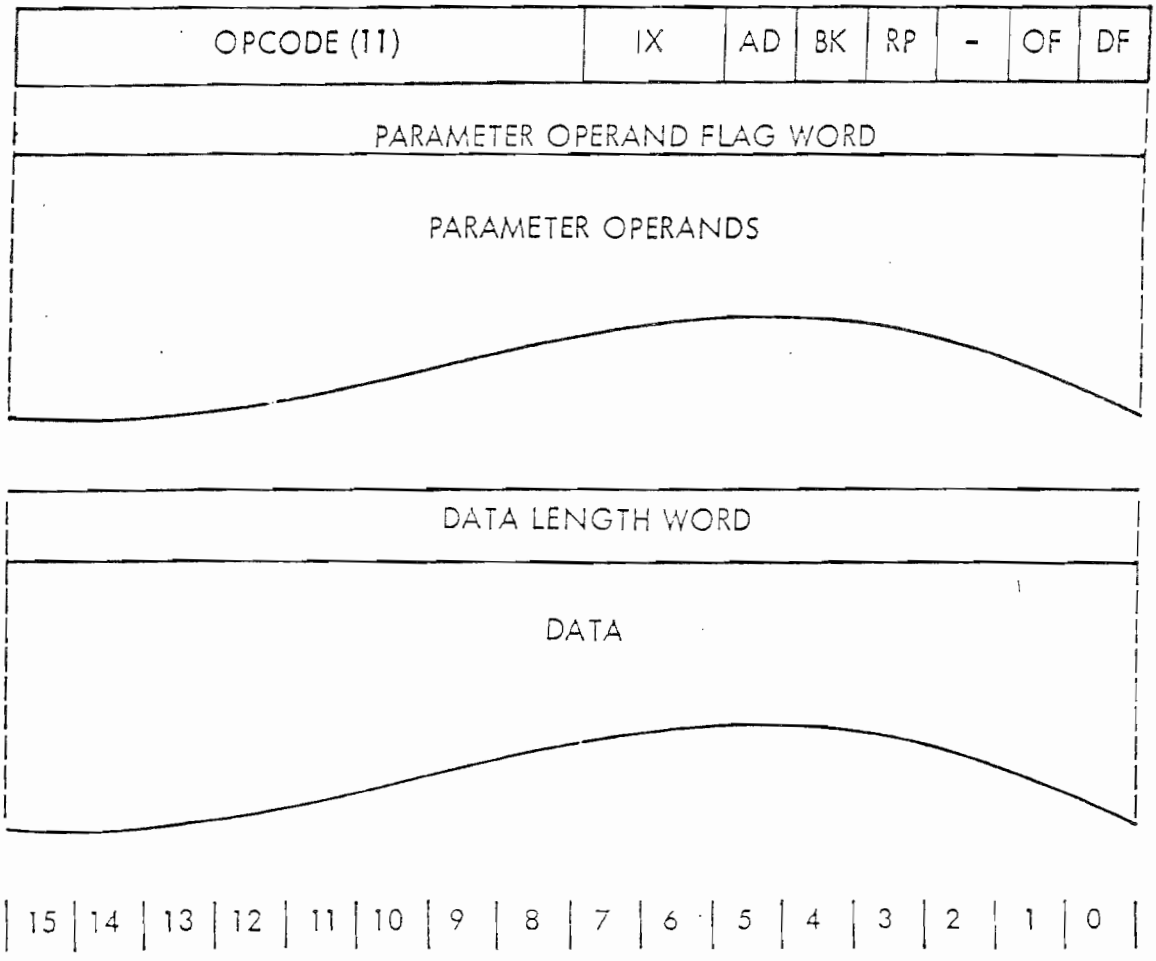
The current operating point after execution of the WC instruction is the coordinate of the last pixel of the conic generated.

3-53 RM-SCR Scroll Option Instructions

The RM-SCR scroll option firmware implements the SCROLL X and the SCROLL Y normal-format instructions. This package allows the scrolling (or moving) of sub-channel-selectable image data within any rectangular region of refresh memory. This operation can be performed in either the X or the Y axis.

The RM-SCR option resides within the internal 8080 processor's address space at locations 1400_{16} through $17FF_{16}$.

3-54 SCROLL X Instruction (SCRX)



The SCRX instruction is a normal-format instruction which is part of the RM-SCR scroll firmware option. The SCRX performs a scroll of image data within the rectangular region defined by the WINDOW parameter operand along the X-axis. The sign of the SCROLL-COUNT parameter operand defines the direction of scroll within the WINDOW region along the X-axis. If SCROLL-COUNT is positive, scrolling will be to the right, i.e., towards increasing element coordinate values; if SCROLL-COUNT is negative, scrolling will be to the left, i.e., towards decreasing element coordinate values. The absolute value of SCROLL-COUNT defines the number of elements which the image data within WINDOW region will be moved. A SCROLL-COUNT of zero will result in no scrolling operation. Only those refresh memory planes selected by the SUB-CHANNELS will be scrolled.

Data which is scrolled out of the window is discarded. If BK=0, the data which is scrolled into the WINDOW region is taken from the BACKGROUND parameter operand; if BK=1, the fill data is taken from the FOREGROUND parameter operand. If the absolute value of SCROLL-COUNT is greater than the WINDOW width, the result will be erasure of the WINDOW region to either the FOREGROUND or BACKGROUND value dependent on the value of BK.

PERTINENT CONTROL BITS

IX	Defines the address mode in which the INDEX1, INDEX2 and WINDOW parameter operands are evaluated.
BK	Defines the color or intensity value to be used to fill the sub-region of the WINDOW region from which data is scrolled. If BK=0, the BACKGROUND value is used; otherwise, if BK=1, the FOREGROUND value is used.
AD	Allows the user to inhibit filling of the WINDOW region from which image data has been scrolled. If AD=1, this erasure will not be performed; otherwise, the erasure will take place if AD=0.

PERTINENT PARAMETER OPERANDS

SUBCHANNELS	Defines the refresh memory planes which will be scrolled and which will receive fill data (when AD=0).
FOREGROUND	Defines the color or intensity value to be used as fill data when BK=1 and AD=0.
BACKGROUND	Defines the color or intensity value to be used as fill data when BK=0 and AD=0.
INDEX1	Displaces the values to be used for the INDEX2 and WINDOW parameter operands when IX=01.
INDEX2	Displaces the values to be used for the WINDOW parameter operand when IX=10.

WINDOW	Defines the rectangular region inside of which image data will be horizontally scrolled.
SCROLL-COUNT	Defines the number of elements to scroll image data within the rectangular WINDOW region. SCROLL-COUNT is defined as a 2's complement number whose absolute value is used as the number of scrolling elements. If SCROLL-COUNT is positive, scrolling is to the right, and if SCROLL-COUNT is negative, scrolling is to the left. A SCROLL-COUNT of 0 results in no scrolling operation.

DATA LENGTH WORD

The DATA LENGTH WORD defines the number of bytes of data present in the SCRX instruction stream when DF=1.

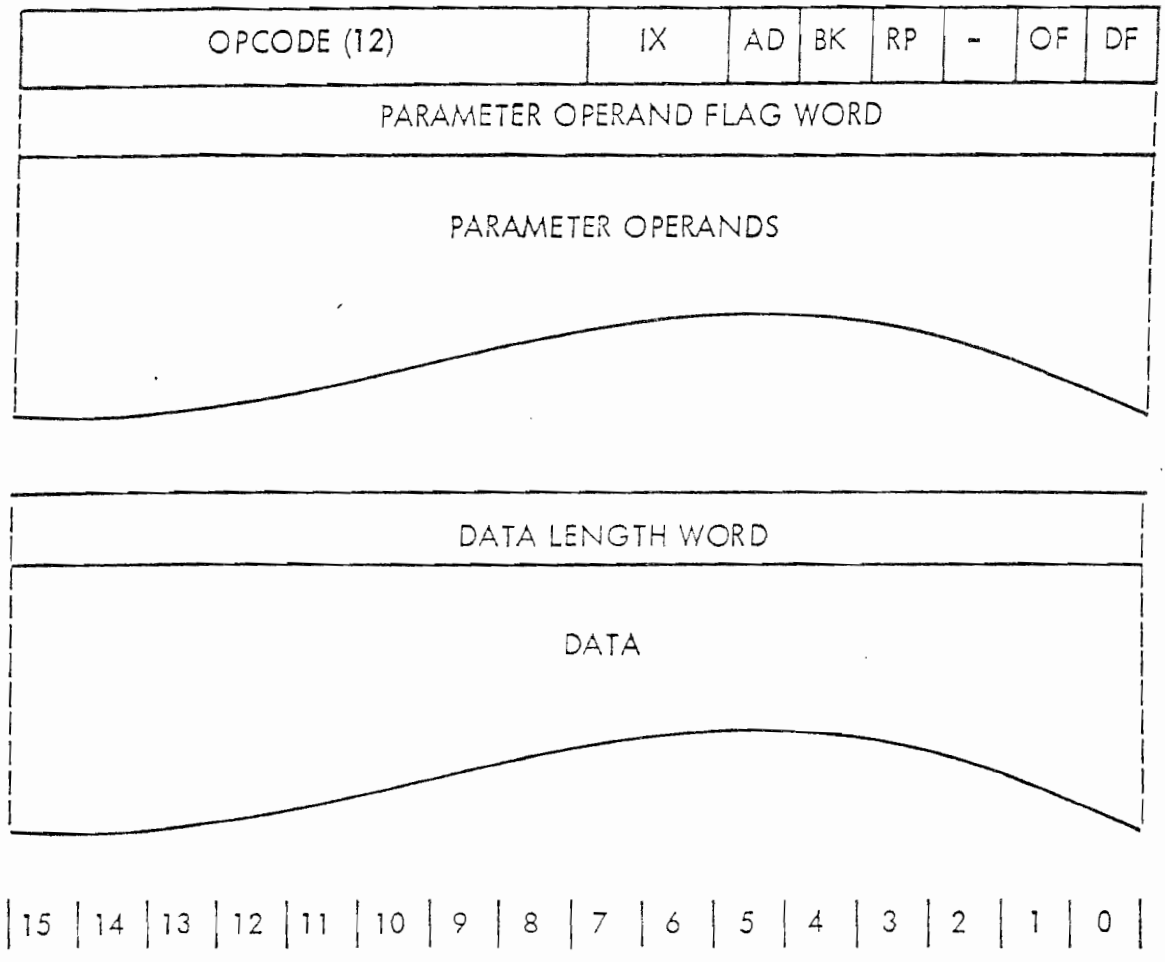
DATA FORMAT

All data present in the SCRX instruction is discarded.

COP MOVEMENT

The COP will remain unchanged during the execution of the SCRX instruction.

3-55 SCROLL Y Instruction (SCRY)



The SCRY instruction is a normal-format instruction which is part of the RM-SCR scroll firmware option. The SCRY performs a scroll of image data within the rectangular region defined by the WINDOW parameter operand along the Y-axis. The sign of the SCROLL-COUNT parameter operand defines the direction of scroll within the WINDOW region along the Y-axis. If SCROLL-COUNT is positive, scrolling will be towards the bottom of the screen, i.e., towards increasing line coordinate values. If SCROLL-COUNT is negative, scrolling will be towards the top of the screen, i.e., toward decreasing line coordinate values. The absolute value of SCROLL-COUNT defines the number of lines which the image data within WINDOW region will be moved. A SCROLL-COUNT of zero will result in no scrolling operation. Only those refresh memory planes selected by the SUBCHANNELS will be scrolled.

Data which is scrolled out of the window is discarded. If BK=0, the data which is scrolled into the WINDOW region is taken from the BACKGROUND parameter operand. If BK=1, the fill data is taken from the FOREGROUND parameter operand. If the absolute value of SCROLL-COUNT is greater than the WINDOW width, the result will be erasure of the WINDOW region to either the FOREGROUND or BACKGROUND value dependent on the value of BK.

PERTINENT CONTROL BITS

IX	Defines the address mode in which the INDEX1, INDEX2 and WINDOW parameter operands are evaluated.
BK	Defines the color or intensity value to be used to fill the sub-region of the WINDOW region from which data is scrolled. If BK=0, the BACKGROUND value is used. Otherwise, if BK=1, the FOREGROUND value is used.
AD	Allows the user to inhibit filling of the WINDOW region from which image data has been scrolled. If AD=1, this erasure will not be performed. Otherwise, the erasure will take place if AD=0.

PERTINENT PARAMETER OPERANDS

SUBCHANNELS	Defines the refresh memory bit planes which will be scrolled and which will receive fill data (when AD=0).
FOREGROUND	Defines the color or intensity value to be used as fill data when BK=1 and AD=0.
BACKGROUND	Defines the color or intensity value to be used as fill data when BK=0 and AD=0.
INDEX1	Displaces the values to be used for the INDEX2 and WINDOW parameter operands when IX=01.
INDEX2	Displaces the values to be used for the WINDOW parameter operand when IX=10.
WINDOW	Defines the rectangular region inside of which image data will be vertically scrolled.

SCROLL-COUNT — Defines the number of lines to scroll image data within the rectangular WINDOW region. SCROLL-COUNT is defined as a 2's complement number whose absolute value is used as the number of scrolling lines. If SCROLL-COUNT is positive, scrolling is to the bottom of the screen, and if SCROLL-COUNT is negative, scrolling is to the top of the screen. A SCROLL-COUNT of 0 results in no scrolling operation.

DATA LENGTH WORD

The DATA LENGTH WORD defines the number of bytes of data present in the SCRY instruction stream when DF=1.

DATA FORMAT

All data present in the SCRY instruction is discarded.

COP MOVEMENT

The COP will remain unchanged during the execution of the SCRY instruction.

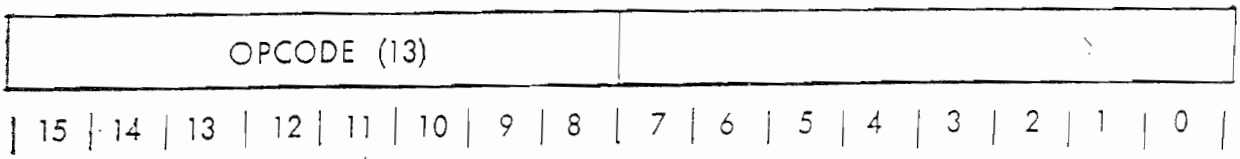
3-56 RM-STA Status Management Option Instructions

The RM-STA status management option firmware implements the SAVE ENVIRONMENT (PUSHE) and the RESTORE ENVIRONMENT (POPE) normal-format instructions. These two instructions provide for a multiple user (or level) environment by temporarily storing and subsequently retrieving display system status from an internal stack. For example, consider the case where Program A (PRGA) is writing an image to refresh memory. At a given point in time, Program B (PRGB) is initiated via keystroke input. PRGB is higher in priority than PRGA and performs the task of echoing keyboard input to the monitor. Therefore, PRGA is suspended (following completion of the current instruction) while PRGB operates. Because PRGB will write a character to a different point on the screen, display system status (environment) will be affected. Thus, PRGA's status must be stored and then restored before and after execution of PRGB. This can be accomplished with some difficulty in the host CPU, or very easily in the display system, using the status management instructions.

A constraint on the use of the status management option is that, since the use of the RM-SCA and RM-LAF options require large buffers to preserve data for each line of data being processed, partial lines of scaled or LAF-processed data will be lost when a given environment is saved via the PUSHE instruction and restored via the POPE instruction.

The RM-STA option resides within the internal 8080 processor's address space at locations 1000_{16} through $13FF_{16}$.

3-57 SAVE ENVIRONMENT Instruction (PUSHE)



The PUSHE instruction is a special-format instruction which is part of the RM-STA status management firmware option. The PUSHE instruction saves all parameter operands as well as associated internal parameters on an internal stack. This internal stack is a LIFO (last-in-first-out) structure. Up to 5 consecutive PUSHE instructions without intervening POPE instructions are allowed (i.e., up to 5 environments may be saved on the internal stack). If more than 5 are attempted by the user, an illegal instruction interrupt will be generated, and the current environment will not be saved on the internal stack.

3-58 RESTORE ENVIRONMENT Instruction (POPE)

OPCODE (14)	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The POPE instruction is a special-format instruction which is part of the RM-STA status management firmware option. The POPE instruction restores all parameter operands as well as associated internal parameters from an internal stack. This internal stack is a LIFO (last-in-first-out) structure. The display environment information which is restored from the internal stack will be the last set of parameter operands which was stored on the internal stack by a PUSHE instruction. If there is no display environment information stored on the stack, an illegal instruction interrupt will be generated and the current display environment will remain unchanged.

3-59 RM-FNT Programmable Font Option Instructions

The RM-FNT programmable font option firmware implements the LOAD PROGRAMMABLE FONT and LOAD PROGRAMMABLE FONT WITH REVERSE PACKING special-format instructions for 128_{10} characters. The RM-FNT package contains the firmware necessary to attach the programmable font character generation routines at power-on and to generate the characters from programmable font RAM in both unscaled and scaled models.

The font matrix which can be stored using the RM-FNT option is 8 bits wide by 12 font lines high. Symbols or characters larger than 8 by 12 can be generated by the user by combining one or more smaller (i.e., 8 by 12) characters.

At power-on or after receiving a RESET instruction, the character fonts from the PROM stored on the control board are copied to the RAM associated with the RM-FNT option for ASCII codes 20_{16} through $5F_{16}$. The font RAM associated with ASCII codes 60_{16} through $9F_{16}$ are not initialized and therefore contain either random data (after power-on) or contain previously loaded values. Since the DIMENSION and SPACING values remain 7 by 9, the use of RM-FNT is transparent to the user.

The RAM associated with the RM-FNT option is located from 4400_{16} through $47FF_{16}$ and used 12_{10} bytes per ASCII code. The RM-FNT option firmware resides within the internal 8080 processor's address space at locations 1800 through $1BFF_{16}$.

3-60 LOAD PROGRAMMABLE FONT Instruction (LPF)

OPCODE (15)	CHARACTER CODE
FONT LINE 1	FONT LINE 2
FONT LINE 3	FONT LINE 4
FONT LINE 5	FONT LINE 6
FONT LINE 7	FONT LINE 8
FONT LINE 9	FONT LINE 10
FONT LINE 11	FONT LINE 12

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The LPF instruction is a special-format instruction which is part of the RM-FNT programmable font option firmware. The LPF instruction loads the 8 by 12 font which will be sent to display refresh memory when the byte value defined by CHARACTER CODE is transmitted as text data in a WT instruction. Although the standard ASCII codes (20_{16} through $5F_{16}$) are copied to the programmable font storage RAM, these codes may be overwritten at any time by the user.

ARGUMENT DEFINITION

CHARACTER CODE — Specifies any of 128_{10} character codes to be loaded with font data for subsequent character or symbol generation. The legal codes which may be loaded via the LPF instruction are 20_{16} through $9F_{16}$. Character codes from 0 through $1F_{16}$ and $A0_{16}$ through FF_{16} are not legal and the font data associated with these codes will be discarded.

DATA DEFINITION

Twelve bytes of font-definition data follow immediately after the opcode/character code word. Font line 1 defines the top line of the 8 by 12 font matrix and font line 12 defines the bottom line of the font matrix. The high-order bit

of each font byte represents the left margin of the font matrix and the low-order bit represents the right margin of the font matrix.

EXAMPLE 3-2

If the symbol described by Figure 3-8 were to be generated using the RM-FNT option at (0,0), the following sequence might be used for character code 9F.

```
0500        ; RESET
159F        ; LPF + CHAR. CODE
0010        ; FONT LINE 1 & 2
3854        ; FONT LINE 3 & 4
9210        ; FONT LINE 5 & 6
1010        ; FONT LINE 7 & 8
2800        ; FONT LINE 9 & 10
0000        ; FONT LINE 11 & 12
0C01        ; WRITE TEXT WITH DATA FLAG
0001        ; NUMBER OF BYTES = 1
9F00        ; CHARACTER CODE
```

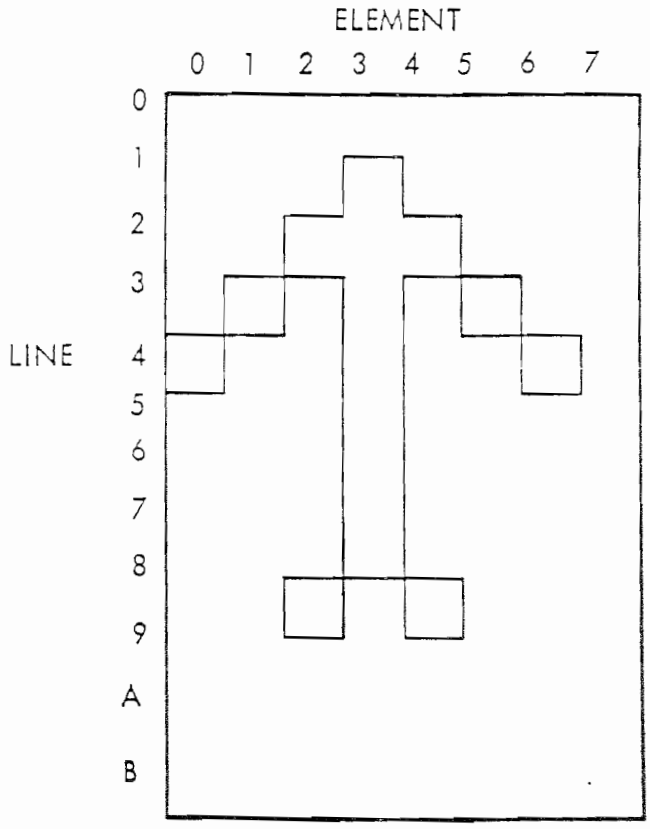


Figure 3-8 Programmable Font Example

3-61 LOAD PROGRAMMABLE FONT WITH REVERSE PACKING Instruction (LPFRP)

OPCODE (21)	CHARACTER CODE
FONT LINE 2	FONT LINE 1
FONT LINE 4	FONT LINE 3
FONT LINE 6	FONT LINE 5
FONT LINE 8	FONT LINE 7
FONT LINE 10	FONT LINE 9
FONT LINE 12	FONT LINE 11

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The LPFRP instruction is a special-format instruction which is part of the RM-FNT programmable font option firmware. The operation of the LPFRP instruction is exactly the same as the LPF instruction (See Section 3-59) except that the font line description bytes are reversed within each word of data.

EXAMPLE 3-3

The symbol described in Figure 3-8 would be generated using the following sequence with the LPFRP instruction:

```
0500        ; RESET
059F        ; LPF + CHAR. CODE = 9F16
1000        ; FONT LINES 2 & 1
5438        ; FONT LINES 4 & 3
1092        ; FONT LINES 6 & 5
1010        ; FONT LINES 8 & 7
0028        ; FONT LINES 10 & 9
0000        ; FONT LINES 12 & 11
0C01        ; WRITE TEXT + DATA FLAG
0001        ; NUMBER OF BYTES = 1
9F00        ; CHARACTER CODE
```

3-62 RM-PER Interactive Peripherals Option Instructions

The RM-PER interactive peripherals option firmware implements the following RM-9000 instructions:

- WRITE CURSOR STATE controlling the position and visible/blink state of any cursors
- READ CURSOR STATUS transmitting the position and visible/blink state of any cursors to the host processor
- WRITE KEYBOARD outputting a byte to one (1) of eight (8) possible serial output devices
- READ KEYBOARD transmitting a received character from any of eight (8) possible serial input devices to the host processor
- SENSE PERIPHERAL STATUS transmitting activity status from either RM-SLC serial link card to the host processor.

The RM-PER option handles all interactive keyboard and cursor functions, generating the appropriate interrupts to the host processor. Buffering of serial input characters up to sixteen (16) characters per device is also handled by the RM-PER option.

The maximum number of RM-SLC serial link cards which can exist in an RM-9000 display system is two (2). A single RM-SLC serial link card may be configured for either four (4) serial I/O ports or for two (2) serial I/O ports and two (2) cursors. This indicates that several different interactive configurations are possible. It is necessary for the user to know the exact RM-SLC configuration of the particular RM-9000 display system and the device addressing scheme used by the RM-PER option. This will avoid software problems caused by improper device addressing.

The RM-PER option resides within the internal 8080 processor's address space at locations $0C00_{16}$ through $0FFF_{16}$.

3-63 Write Cursor State Instruction (WCS)

OPCODE (16)															DEVICE
							X - ADDRESS								
					BL	VI	-	Y - ADDRESS							
15	14	13	12	11	10	9	8	7	5	5	4	3	2	1	0

The WCS instruction is a special-format instruction which is part of the RM-PER interactive peripherals option firmware. The WCS instruction is used to define the position of any of four (4) cursors in the RM-9000 display as well as it's visible/invisible and blink/no-blink states. No errors are possible, since the attempted addressing of non-existent cursors will take place, but will have no effect upon the display system.

PERTINENT ARGUMENTS

DEVICE — Specifies the cursor whose state is to be defined by the instruction. The DEVICE assignments are as follows:

DEVICE	CURSOR
0	Cursor #1 of SLC Board #1
1	Cursor #2 of SLC Board #2
2	Cursor #1 of SLC Board #2
3	Cursor #2 of SLC Board #2

X-ADDRESS — Defines the element coordinate at which the cursor specified by device will be positioned. Note that due to mechanism used to superimpose the cursor onto the refresh memory video, there is a region of elements in which the cursor will be off of the visible screen area. These areas are as follows:

SYSTEM TYPE	REGION (IN ELEMENTS)
RM-9000	320_{10} through 511_{10}
RM-9200	640_{10} through 1023_{10}
RM-9300	640_{10} through 1023_{10}

Y-ADDRESS — Defines the line coordinate at which the cursor specified by DEVICE will be positioned. For low line resolution systems (i.e., the RM-9100 and RM-9200), cursor line addressing is module 256_{10}, i.e., line addresses 256_{10} through 511_{10} correspond to line address 0 through 225_{10}, respectively.

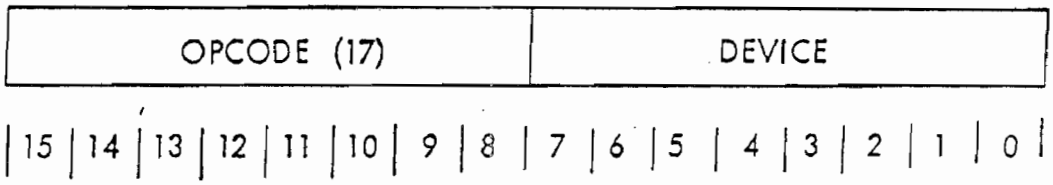
VI — Defines the visible or invisible state of the cursor specified by DEVICE. If VI = 0, the cursor will be made invisible; if VI = 1, the cursor will be made visible.

BL — Defines the blink state of the cursor specified by DEVICE. If BL = 0, the cursor will be made non-blinking; if BL = 1, the cursor will blink at a rate of one (1) Hz.

NOTE

THE CURSOR BLINK AND VISIBLE STATE CAN ALSO BE MANUALLY CONTROLLED BY A JOYSTICK OR TRACKBALL IF EITHER OF THESE DEVICES EXIST IN A GIVEN CONFIGURATION. THE VI AND BL STATE MAY BE MANUALLY CHANGED IN THIS CASE WITH NO INDICATION BEING MADE TO THE HOST PROCESSOR. (SEE APPENDIX C FOR OPERATIONAL CHARACTERISTICS OF THE JOYSTICK AND TRACKBALL IN THIS CONTEXT.

3-64 Read Cursor Status Instruction (RCS)



The RCS instruction is a special-format instruction which is part of the RM-PER interactive peripherals option firmware. The RCS instruction conditions the RM-9000 for the transfer of two (2) words of cursor status information to the host processor. After the RCS instruction has been transferred from the host processor to the RM-9000, it is the responsibility of the host processor to initiate a two (2) 16-bit word data transfer from the RM-9000. (See Addenda on the operation of specific CPU interfaces.) If these two (2) words are not read back properly by the host processor, it is possible that the inter-processor communications sequence may become hung. This condition can only be rectified using a hard system reset.

PERTINENT ARGUMENTS

DEVICE — Specifies the cursor whose status is to be read back to the host processor after the RCS instruction has been issued. The DEVICE assignment is as follows:

DEVICE	CURSOR
0	Cursor #1 of SLC Board #1
1	Cursor #2 of SLC Board #1
2	Cursor #1 of SLC Board #2
3	Cursor #2 of SLC Board #2

DATA FORMAT

The format of the two data words to be read back to the host processor is as follows:

0	0	0	0	0	0	X - ADDRESS
0	0	EN	TK	BL	VI	0	Y - ADDRESS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

X-ADDRESS — Specifies the current element coordinate of the cusor specified by DEVICE.

Y-ADDRESS — Specifies the current line coordinate of the cursor specified by DEVICE.

VI — Reflects the current visible/invisible state of the cursor selected by DEVICE. If VI = 0, the cursor is not visible; if VI = 1, the cursor is currently visible.

TK — Reflects the current state of the two-position TRACK switch on the joystick or trackball if present. If TK = 0, the TRACK function is not enabled; if TK = 1, the TRACK function is enabled.

BL — Reflects the blinking state of the cursor selected by DEVICE. If BL = 0, the cursor is non-blinking; if BL = 1, the cursor is blinking at a rate of approximately one (1) second.

EN — Reflects the state of momentary-action switch on the joystick or trackball if present. If EN = 0, the ENTER switch is not depressed; if EN = 1, the ENTER switch is currently depressed.

3-65 Write Keyboard Instruction (WKB)

OPCODE (18)	DEVICE
-	CHARACTER CODE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The WKB instruction is a special-format instruction which is part of the RM-PER interactive peripherals option firmware. The WKB instruction writes a single 8-bit character to one (1) of eight (8) possible serial output devices within the RM-9000 system. No buffering of output characters is performed; therefore, it is necessary to wait for the serial transmission of the character to complete. A transmitter interrupt will be generated by the RM-9000 to the host processor. The serial peripheral status word should then be read by the host processor using the SENSE PERIPHERAL STATUS instruction to verify that the expected transmission was completed (See Section 3-66). It is then allowable to output another data byte to the serial output device specified by DEVICE. It is not necessary to wait for completion of a transmission before outputting to a different serial output device.

PERTINENT ARGUMENTS

DEVICE — Specifies the serial output device to which the CHARACTER CODE will be output. The DEVICE code assignment is as follows:

DEVICE	OUTPUT PORT
0	Serial Output Port #1 of SLC Board #1
1	Serial Output Port #2 of SLC Board #1
2	Serial Output Port #3 of SLC Board #1
3	Serial Output Port #4 of SLC Board #1
4	Serial Output Port #1 of SLC Board #2
5	Serial Output Port #2 of SLC Board #2
6	Serial Output Port #3 of SLC Board #2
7	Serial Output Port #4 of SLC Board #2

CHARACTER CODE

Defines the 8-bit byte of data which will be output to the serial output port specified by DEVICE. All values from 0 through FF_{16} are legal.

3-66 Read Keyboard Instruction (RKB)

OPCODE (19)	DEVICE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The RKB instruction is a special-format instruction which is part of the RM-PER interactive peripherals option firmware. The RKB instruction conditions the RM-9000 for the transfer of one word of serial data from the serial port input buffer specified by DEVICE. After the RKB instruction has been transferred from the host processor to the RM-9000, it is the responsibility of the host processor to read back one (1) 16-bit data word from the RM-9000. If the word of data is not read back properly by the host processor, it is possible that the interprocessor communications sequence may become hung. This condition can only be rectified using a hard system reset.

For each of the eight (8) possible serial input ports, there exists an internal buffer which can handle up to sixteen (16) bytes of serial input data, as well as corresponding status information. A receiver interrupt will be generated to the host processor for each data byte received from a serial input port. After interrogation of the serial peripheral status words using the SPS instruction, the RKB instruction could be used to read the data byte received.

All input buffers are cleared upon the power-on or the hard reset condition. All characters received prior to this will be lost.

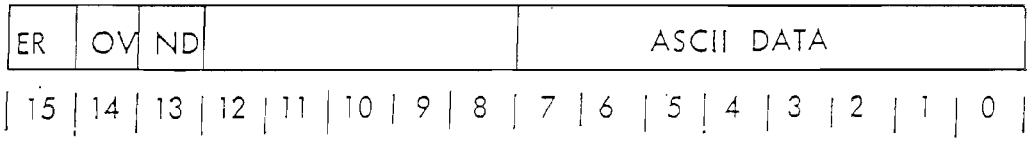
PERTINENT ARGUMENTS

DEVICE — Specifies the serial input port buffer from which data will be read along with the corresponding status bits. The DEVICE assignment is as follows:

DEVICE	INPUT PORT
0	Serial Input Port #1 of SLC Board #1
1	Serial Input Port #2 of SLC Board #1
2	Serial Input Port #3 of SLC Board #1
3	Serial Input Port #4 of SLC Board #1

4	Serial Input Port #1 of SLC Board #2
5	Serial Input Port #2 of SLC Board #2
6	Serial Input Port #3 of SLC Board #2
7	Serial Input Port #4 of SLC Board #2

DATA FORMAT — The format of the data word read back to the host processor is as follows:



DATA VALUE — Reflects the 8-bit data value input to the RM-9000 display processor from the external serial device

ND — Reflects a 'no data available' condition. If ND = 0 and ER = 0, the DATA VALUE represents a valid received data byte. If ND = 1, no data is present in the input buffer.

OV — Reflects a 'data buffer overflow' condition. If OV = 1, data was received from the serial input port but no room was available in the input buffer, i.e. data has been lost.

ER — Reflects a hardware error condition. If ER = 1, a hard error has been detected by the input UART in attempting to obtain a character from the external device.

3-67 Sense Peripheral Status Instruction (SPS)

OPCODE (1A)	DEVICE

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

The SPS instruction is a special-format instruction which is part of the RM-PER interactive peripherals option firmware. The SPS instruction conditions the RM-9000 for the transfer of one word of serial peripheral board status from the RM-SLC serial link board specified by DEVICE. After the SPS instruction has been transferred from the host processor to the RM-9000, it is the responsibility of the host processor to readback one (1) 16-bit word from the RM-9000. If the word of data is not readback properly by the host processor, it is possible that the interprocessor communication sequence may become hung. This condition can only be rectified using a hard system reset.

The data word readback from the RM-9000 reflects the status of the RM-SLC board selected by DEVICE. This status indicates which interactive device is currently requesting service or is available for output. Upon receipt of a transmitter or receiver interrupt, this data word defines the interrupting source.

PERTINENT ARGUMENTS

DEVICE — Specified the RM-SLC board whose status is to be interrogated. If DEVICE = 0, the RM-SLC Board 1 is to be interrogated. If DEVICE = 1, the RM-SLC Board 2 is to be interrogated.

DATA FORMAT

The format of the serial peripheral status word is as follows:

—	C1	C0	X_3	R_3	X_2	R_2	X_1	R_1	X_0	R_0	SLC BOARD 1
—	C3	C2	X_7	R_7	X_6	R_6	X_5	R_5	X_4	R_4	SLC BOARD 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

$C_m = 1$ Indicates that one of two events has occurred:

- The ENTER momentary action switch has been depressed with cursor m (where m = 0 through 3) selected on the joystick or trackball, or
- The coordinates of cursor m have changed with cursor m selected on the joystick or trackball and with the TRACK switch enabled. C will remain set to one (1) until the coordinates of cursor m have been read using the RCS instruction, at which time C will be reset to zero (0).

$X_n = 0$ Indicates that a serial transmission initiated by a WKB instruction has not yet completed at output port n (where n = 0 through 7). X_n will be set to one (1) when transmission is completed.

$R_n = 1$ Indicates that a data byte has been received by the serial input port n (where n = 0 through 7). R will be reset to zero (0) when the data byte along with its status has been readback after issuing an RKB instruction.

3-68 RM-USR User Subroutine Option Instructions

The RM-USR user subroutine option firmware implements the following six (6) instructions:

- LOAD CONTROL MEMORY and LOAD CONTROL MEMORY WITH REVERSE PACKING which store either executable 8080 processor instructions or RM-9000 display instructions into RM-MOC memory expansion RAM
- READ CONTROL MEMORY and READ CONTROL MEMORY WITH REVERSE PACKING which retrieve data from locations 000 through 7FFF of the RM-9000 display system internal 8080 address space
- CALL CONTROL MEMORY which executes any 8080 subroutine stored in the 8080 processor's address space
- EXECUTE INSTRUCTION MEMORY which executes one or more RM-9000 display instructions stored in the 8080 processor's address space.

The RM-USR option, in general, implement a writeable control store capability in that the host processor can load, retrieve and execute special-purpose subrountines and display instruction lists. The Call Control Memory instruction is executed within the RM-9000 display. It has direct access to all display registers and requires an intimate knowledge of the interaction between display registers for proper display generation. The Execute Instruction Memory provides local display list processing using the RM-9000 instruction. This does not require special knowledge of the RM-9000 hardware.

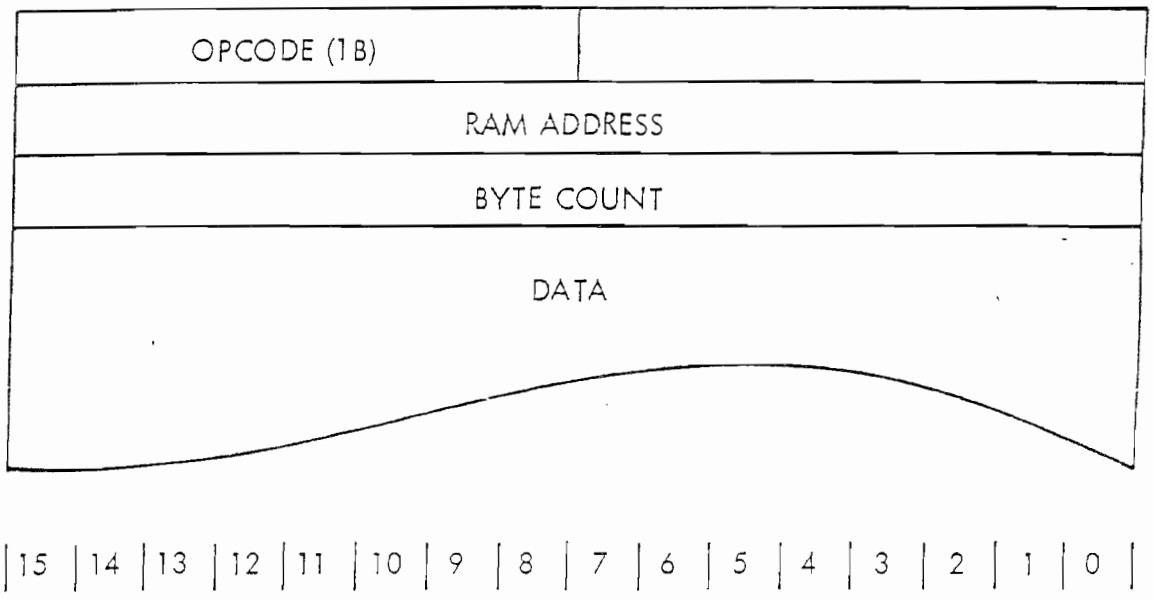
It is necessary to load either subroutine code or RM-9000 display lists into RAM within the RM-9000 display system. The memory map (Table 3-16) defines the areas of RAM and the options packages which use these specific areas. If a given option is not installed in a particular configuration, the RAM region assigned to this option is available for use with the RM-USR user subroutine option firmware. If an area used by an existing option is loaded via the RM-USR option, the operation of the option firmware will be adversely affected. No error checking is performed to restrict the use of areas of RAM.

Table 3-16 RM-9000 Internal Memory Map

PROM ADDRESS	DESCRIPTION
0000 - 07FF	RM-IMG STANDARD IMAGER FIRMWARE
0800 - 0BFF	RM-STD STANDARD OPTION SUPPORT
0C00 - 0FFF	RM-PER INTERACTIVE PERIPHERALS OPTION
1000 - 13FF	RM-STA STATUS MANAGEMENT OPTION
1400 - 17FF	RM-SCR SCROLL OPTION
1800 - 1BFF	RM-FNT PROGRAMMABLE FONT OPTION
1C00 - 23FF	RM-SCA SCALING OPTION
2400 - 27FF	RM-LAF LOGICAL/ARITHMETIC FUNCTION OPTION
2800 - 2BFF	RM-GRA GRAPHICS OPTION
2C00 - 33FF	RM-CON CONICS OPTION
3400 - 37FF	RM-USR USER SUBROUTINE OPTION
6200 - 63FF	STANDARD CHARACTER ROM
6400 - 7BFF	RM-DIA DIAGNOSTIC OPTION

RAM ADDRESS	USAGE
4000 - 414F	RM-PER INPUT BUFFERS
4150 - 415F	NOT USED
4160 - 417F	RM-GRA INTERNAL VARIABLES
4180 - 41AF	RM-CON INTERNAL VARIABLES
41B0 - 41FF	NOT USED
4200 - 438F	RM-STA INTERNAL STACK & VARIABLES
4390 - 439F	NOT USED
43A0 - 43AF	RM-PER INTERNAL VARIABLES
43B0 - 43FF	NOT USED
4400 - 487F	RM-FNT PROG FONT TABLE
4880 - 489F	RM-FNT INTERNAL VARIABLES
48A0 - 48FF	NOT USED
4900 - 491F	RM-SCR INTERNAL VARIABLES
4920 - 495F	NOT USED
4960 - 4AEF	RM-FNT PROG FONT TABLE
4AF0 - 55FF	NOT USED
5600 - 5FFF	RM-SCR AND RM-LAF INTERNAL BUFFERS
6000 - 60FF	STACK AREA
6100 - 61FF	RM-IMG INTERNAL VARIABLES
7C00 - 77FF	RM-DIAG DIAGNOSTIC CARD RAM

3-69 Load Control Memory Instruction (LCM)



The LCM instruction is a special-format instruction which is part of the RM-USR user subroutine option firmware. The LCM instruction loads either 8080 microprocessor instructions or RM-9000 display instructions into RAM memory on the RM-MOC memory expansion card. The LCM does not differentiate between 808u instructions and RM-9000 display instruction. The use of either the CCM or the XIM instruction defines the functional use of any data loaded by the LCM instruction.

PERTINENT ARGUMENTS

RAM ADDRESS	Specifies the address in internal 8080 RAM at which subroutine or display instruction data will be stored. No address validity check is made that the address specified is legal.
BYTE COUNT	Defines the number of bytes of subroutine or display instruction data to be stored in internal 8080 RAM.

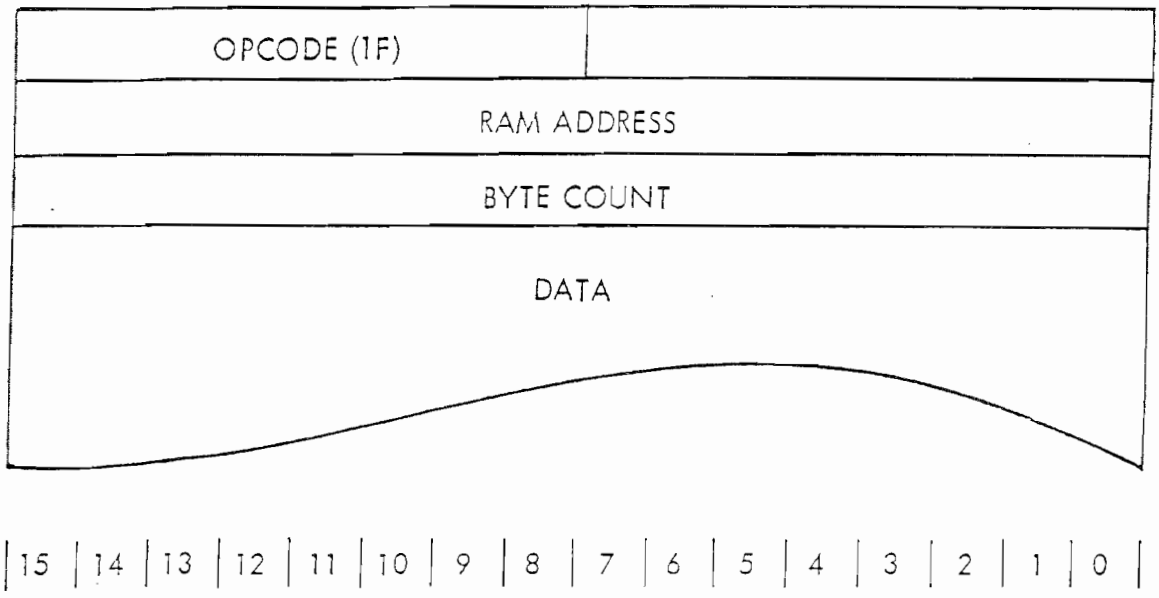
DATA FORMAT

The subroutine or display instruction data is stored into internal 8080 RAM high-order byte first, followed by the low-order byte. If the BYTE COUNT is odd, the low-order byte of the last data word will be discarded.

POSSIBLE ERRORS

If an area of RAM used by an option which exists in a given system is overwritten, the operation of that option cannot be guaranteed unless a reset sequence is initiated.

3-70 Load Control Memory With Reverse Packing Instruction (LCMRP)



The LCMRP instruction is a special-format instruction which is part of the RM-USR user subroutine option firmware. The LCMRP instruction loads either 8080 microprocessor instructions or RM-9000 display instructions into RAM memory on the RM-MOC memory expansion card. The LCMRP does not differentiate between 8080 instructions and RM-9000 display instruction the use of either the CCM or the XIM instruction defines the functional use of any data loaded by the LCMRP instruction.

PERTINENT ARGUMENTS

RAM ADDRESS — Specifies the address in internal 8080 RAM at which subroutine or display instruction data will be stored. No address validity check is made that the address specified is legal.

BYTE COUNT — Defines the number of bytes of subroutine or display instruction data to be stored in internal 8080 RAM.

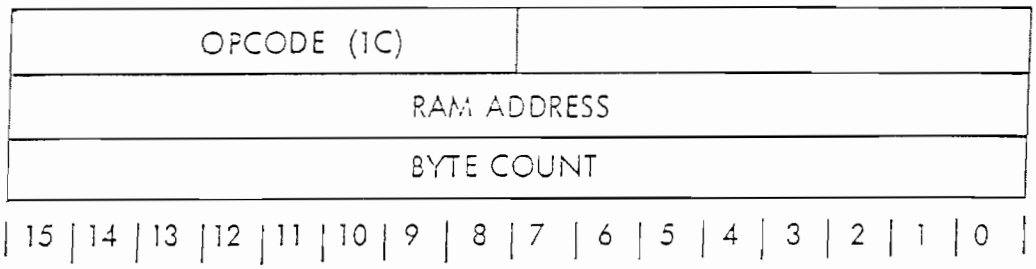
DATA FORMAT

The subroutine or display instruction data is stored into internal 8080 RAM low-order byte first, followed by the high-order byte. If the BYTE COUNT is odd, the high-order byte of the last data word will be discarded.

POSSIBLE ERRORS

If an area of RAM used by an option which exists in a given system is overwritten, the operation of that option cannot be guaranteed unless a reset sequence is initiated.

3-71 Read Control Memory Instruction (RCM)



The RCM instruction is a special-format instruction which is part of the RM-USR subroutine option firmware. The RCM instruction transfers up to 2000_{16} (8192_{10}) bytes of data from RAM on the RM-MOC memory expansion card to the host processor. This is the region within the internal 8080 microprocessor's address space which may be loaded via the LCM or LCMRP instruction. It is not valid to attempt to read back data from the region outside of the RM-MOC card RAM, and as a consequence, the system may be hung up in doing so.

PERTINENT ARGUMENTS

RAM ADDRESS — Specifies the address in internal 8080 RAM in the range 4000_{16} through $5FFF_{16}$. Addressing outside of this region is illegal and may cause the RM-9000 display system to hang up.

BYTE COUNT — Defines the number of bytes of data to be read back from RAM on the RM-MOC. Since readback to the host processor is done on a word basis, if the BYTE COUNT is odd, then one additional byte will actually be read back to the host processor. Since it is illegal to readback from outside of the RM-MOC RAM area, the sum of BYTE COUNT and RAM ADDRESS must not be greater than 6000_{16}.

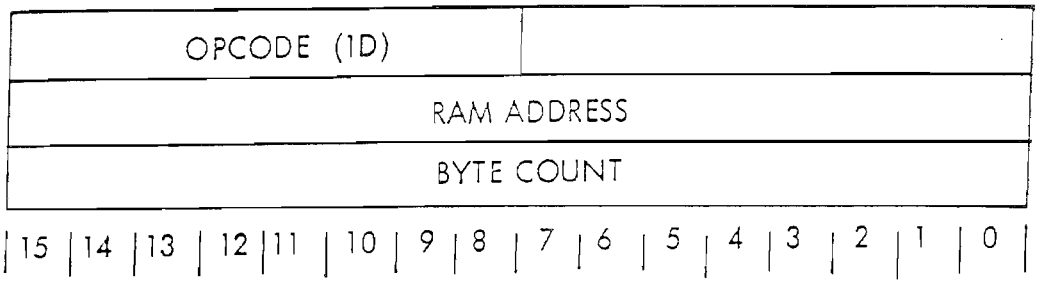
DATA FORMAT

The data which is read back to the host processor is stored in each data word as follows: The first data byte is stored in the high-order byte of the first readback word, the second data byte in the low-order byte of the first data word, the third data byte in the high-order byte of the second data byte, etc.

POSSIBLE ERRORS

If read back via the RCM is attempted outside of the address region 4000_{16} through $5FFF_{16}$, the system may hang up, and communication can only be reestablished after a hard system reset.

3-72 Read Control Memory With Reverse Packing Instruction (RCMRP)



The RCMRP instruction is a special-format instruction which is part of the RM-USR user subroutine option firmware. The RCMRP instruction transfers up to 2000 (8192_{10}) bytes of data from RAM on the RM-MOC memory expansion card to the host processor. This is the region within the internal 8080 microprocessor's address space which may be loaded via the LCM or LCMRP instructions. It is not valid to attempt to readback data from the region outside of the RM-MOC card RAM, and, as a consequence, the system may be hung up in doing so.

PERTINENT ARGUMENTS

RAM ADDRESS — Specifies the address in internal 8080 RAM in the range 4000_{16} through $5FFF_{16}$. Addressing outside of this region is illegal and may cause the RM-9000 display system to hang up.

BYTE COUNT — Defines the number of bytes of data to be read back from RAM on the RM-MOC. Since readback to the host processor is done on a word basis, if the BYTE COUNT is odd, then one additional byte will actually be read back to the host processor. Since it is illegal to read back from outside of the RM-MOC RAM area, the sum of BYTE COUNT and RAM ADDRESS must not be greater than 6000_{16}.

DATA FORMAT

The data which is read back to the host processor is stored in each data word as follows: The first data byte is stored in the low-order byte of the first readback word, the second data byte in the high-order byte of the first data word, the third data byte in the low-order byte of the second data byte, etc.

POSSIBLE ERRORS

If readback via the RCMRP is attempted outside of the address region 4000_{16} through $5FFF_{16}$, the system may hang up, and communication can only be reestablished after a hard system reset.

3-73 CALL CONTROL MEMORY INSTRUCTION (CCM)

OPCODE (1C)	
SUBROUTINE ADDRESS	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

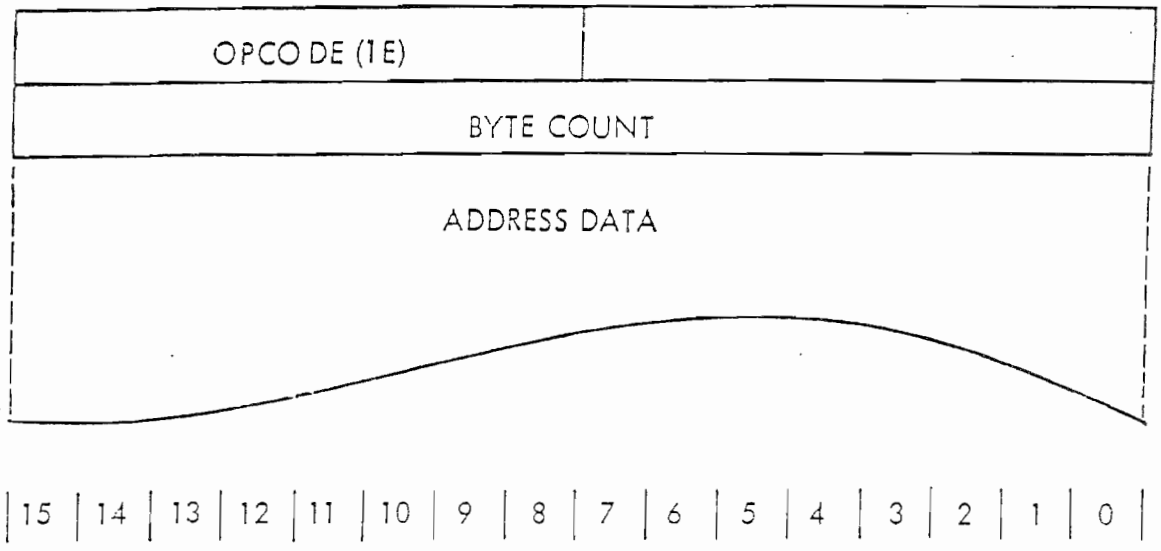
The CCM instruction is a special-format instruction which is part of the RM-USR user subroutine option package. The CCM instruction is used to transfer control from the RAMTEK-supplied firmware package to a user-written subroutine which is stored in the RAM on the RM-MOC memory expansion card via the LCM or LCMRP instructions. The user-subroutine must be executable 8080 microprocessor code with addressing compatible with the region into which the subroutine has been loaded. The user subroutine to be executed must return control to the RAMTEK-supplied firmware via the 8080 RET instruction. Since the user subroutine executes within the RM-9000 display system, it has complete access to all internal display and interfacing registers. As a result of this, it is possible to perform exotic display functions specific to a given user application. It is also possible to initiate actions which cause the system to hang up and which can only be resolved via a hard system reset. Therefore, great care should be taken in writing user subroutines, making careful reference to Volume I of the RM-9000 Theory of Operation Manual.

It is the responsibility of the user subroutine to maintain the integrity of the system stack pointer. When entry is made to the user subroutine, 127_{10} stack locations (words) are available for use. For each stack 'push' operation, there must be a corresponding stack 'pop' within the subroutine to maintain stack integrity. If this stack is not properly maintained, execution of the RET transferring control to the RAMTEK-supplied firmware system will fail and no further operations will be possible until a hard system reset is issued. All 8080 registers are available for use by the subroutine.

PERTINENT ARGUMENTS

SUBROUTINE ADDRESS Defines the internal 8080 microprocessor RAM address at which subroutine execution will begin.

3-74 Execute Instruction Memory Instruction (XIM)



The XIM instruction is a special-format instruction which is part of the RM-USR user subroutine firmware package. The XIM instruction executes one or more RM-9000 instructions from RAM memory on the RM-MOC memory expansion card. The address of each instruction to be executed via the XIM instruction is stored within the XIM instruction itself. It is possible to nest XIM instructions by pointing an address at another XIM instruction. Up to 38_{10} such nesting operations are legal. More than 38_{10} levels of nesting will result in stack overflow and may result in the system hanging. The standard system stack is used to save the nesting information. No stack overflow check is performed. Also, there is no check made for recursion, i.e., an XIM instruction referencing the XIM which referenced it previously. This will result in a recursive loop which will always lead to a stack overflow condition.

PERTINENT ARGUMENTS

BYTE COUNT — Specifies the number of bytes of address data which are present in theXIM instruction. Since the instruction addresses within the XIM instruction are 16-bit values, the BYTE COUNT must be an even value. If BYTE COUNT is odd, the value used will be one less than the stored value of BYTE COUNT.

DATA FORMAT

Each 16-bit word following the BYTE COUNT word defines the starting address of an RM-9000 display instruction. These addresses are the absolute internal 8080 RAM addresses at which data was loaded via the LCM or LCMRP instructions.

4-0 INTERRUPT OPERATIONS

The RM-9000 is capable of generating interrupts to the host processor through the RAMTEK supplied interface associated with each display system. The four (4) possible interrupt types are:

- Illegal Instruction Interrupt
- Cursor Interrupt
- Receiver Interrupt
- Transmitter Interrupt

These interrupts are generated internally using the interface interrupt register (See Section 6 of the RM-9000 Theory of Operation Manual - Volume I).

All of the RAMTEK supplied interfaces generate an interrupt to the host processor based on a low-to-high transition of the interrupt lines from the RM-9000 control board to the computer interface. The standard firmware generates this transition by loading a zero in the interface interrupt register followed by the setting of the specific interrupt line(s) in this register. Reading of the computer interface status word clears the interrupt flags in this word.

4-1 ILLEGAL INSTRUCTION INTERRUPT

The illegal instruction interrupt is generated when a word of information from the interface to be interpreted as an instruction opcode has an opcode which does not exist for the specific display system configuration. The illegal instruction interrupt is then generated. The next word will then be interpreted as an opcode word.

4-2 CURSOR INTERRUPT

The cursor interrupt is generated when the RM-PER interactive peripherals option is installed and one of the following occurs: either the ENTER momentary action switch is pressed or the cursor position is changed with the TRACK switch in the ON position. These keys (ENTER and TRACK) are present on both the joystick and the trackball.

4-3 RECEIVER INTERRUPT

The receiver interrupt is generated when the RM-PER interactive peripherals option is installed and a key on the keyboard is depressed. The 8-bit code asso-

ciated with the depressed key is stored in the FIFO (first-in-first-out) buffer associated with that particular keyboard and may be read by the host processor using the READ KEYBOARD instruction.

If any characters are present in the keyboard buffers after execution of a READ KEYBOARD instruction, a receiver interrupt will be issued.

4-4 Transmitter Interrupt

The transmitter interrupt is generated when the RM-PER interactive peripherals option is installed and the completion of a serial output transmission using the WRITE KEYBOARD instruction is indicated to the firmware by the RM-SLC serial link card. This interrupt indicates that another WRITE KEYBOARD instruction to the interrupting output device is valid.

Famtek

SECTION V

RM-9000 INSTRUCTION TIMING

5-0 RM-9000 INSTRUCTION TIMING

This section describes the execution times of the RM-9000 display instruction set. In several cases, the instruction times are approximate due to the variable nature of the parameters used as input to the instruction. In these cases (which are noted in the instruction's execution time description, the equation describing the execution speed becomes extremely complex and cumbersome, in which case a reasonable approximation to the equation has been made.

5-1 Standard Instruction Set Execution Times

The following sections describe the execution timing for the RM-9000 standard instruction set. For normal format instructions which are, by nature, of variable format, it is possible to set a variable number (up to 16) parameter operands as well as actually execute the display instruction itself. Section 5-2 defines the parameter operand processing time (POPT) for the normal format instructions.

5-2 Parameter Operand Processing Time

In any normal format instruction, it is possible to define the value of up to 16 parameter operands. The parameter operand processing time or POPT must be added to the execution time of the normal format instruction in which these parameter operands are set.

5-3 INOP Instruction Time

Execution Time = 135 µs + 65 µs * NP + 40 µs * NPW

+ 65 µs * NBD

where NP = number of parameter operands in NOP instruction to be ignored.

NPW = number of words of parameter operand values to be ignored in INOP instruction

NBD = number of bytes of data in INOP instruction to be ignored

NOTE

EVEN THOUGH INOP IS A NORMAL-FORMAT INSTRUCTION, ITS TIMING REPRESENTS A SPECIAL CASE, SINCE ALL PARAMETER OPERANDS AND DATA ARE DISCARDED.

Table 5-1 Normal-Format Parameter Operand Processing Times

i	PARAMETER OPERAND	PT_i
0	SUBCHANNEL	42 s
1	FOREGROUND	42 s
2	BACKGROUND	42 s
3	INDEX 1	140 s
4	INDEX 2	140 s
5	ORIGIN	140 s
6	WINDOW	1250 s
7	SCAN	905 s
8	DIMENSION	75 s
9	SPACING	81 s
10	SCALE	40 s
11	FUNCTION	40 s
12	CONIC-EQUATION	630 s
13	BASE-LINE	40 s
14	SCROLL-COUNT	40 s
15	START-POINT	250 s

$$POPT \cong \sum_{i=0}^{15} (65\,\mu s + PT_i)$$

where $PT_i = 0$ if the i^{th} parameter operand is not set in the instruction
otherwise the value of PT_i is the value from Table 5-1.

5-4 Set Instruction Timing

Execution Time = 125 µs + POPT + DFT +[57.5 µs * NBD]

where DFT = Ø if the data flag DF = 0

= 36 µs if the data flag DF = 1

NBD = number of bytes of data in the set instruction to be ignored

5-5 Erase Instruction Timing

Execution Time = 265 µs + POPT +[NL * [43.5 µs +[1.5 µs * NE]]

where NL = number of lines in WINDOW to be erased

NE = number of elements per line in WINDOW to be erased

For example, a full screen with no parameter operand processing in an RM-9200 system would be:

265 µs +[256 * [43.5 µs +[1.5 µs + 640]] = 257 ms

5-6 Write Image Instruction Timing

Execution Time = 490 µs + POPT

+ [[WW * DTR] + 113 µs][NL-1] + [RP * DTR]

where WW = window width in pixels (i.e., width parallel to the primary update direction)

DTR = interface data transfer rate in microseconds (minimum value of 1.5 µs due to refresh memory access rate)

NL = number of lines in image

RP = number of pixels in last line of image

For example, a full-screen image transfer on an RM-9100 system interfaced to a DR11B DMA interface (and assuming an infinitely fast image data source in the PDP-11 computer) and assuming no parameter operand processing, would be:

490 µs + [[320 * 1.5 µs] + 113 µs] * 255 + [320 * 1.5 µs] = 152 ms
using values of WW=320, DTR=1.5 µs, NL=256, RP=320

5-7 Read Image Instruction Timing

The execution time for the RI instruction is the same as that of the WI instruction (See Section 5-6).

5-8 Write Text Instruction Timing

Execution time = 480 µs + POPT
+ [263 µs + [64 µs*CH] + [1.5 µs*CH*CW]] *NPC
+ [162 µs *NCR] + [146 µs *NLF]

where CW - character width (defined by X DIMENSION)
CH = character height (defined by Y DIMENSION)
NCR = number of carriage returns or primary window boundary intersections
NLF = number of line feeds or secondary window boundary intersections
NPC = number of non-CR or non-LF characters

For example, a full-screen text write on an RM-9200 system (i.e., 91 by 28 characters of text) with no parameter processing, would be:

480 µs + [263 µs + [64 µs*9] + [1.5 µs *7*9]] *2548
+ [162 µs * 28] + [146 µs *1] = 2.38 seconds

5-9 Load Hard Register Instruction Timing

Execution time = 170 µs

5-10 Read Soft Register Instruction Timing

Execution Time = 150 µs

5-11 Load Auxiliary Memory Instruction Timing

Execution Time = 275 μs + [NW * DTR]

where NW = Number of words of auxiliary memory data to be transferred

DTR = Interface data transfer rate in microseconds (minimum value of 0.765 μs based on the Type II and V video lookup tables loading rate)

For example, to load a Type II video lookup table would take:

275 μs + [1024 * 0.765 μs] = 1.05 ms

5-12 Read Auxiliary Memory Instruction Timing

The execution time for the RAM instruction is the same as that of the LAM instruction (See Section 5-11).

5-13 Reset Instruction Timing

Execution Time = 29 ms + ERST + OPTT

where ERST = screen erase time (dependent on system resolution) when DIP SWITCH 3 = 0

STDT = 134 ms for RM-9100
= 257 ms for RM-9200
= 514 ms for RM-9300

OPTT = sum of power-on routine execution times for the options which exist in a particular configuration

$$= 63 \text{ ms} + \sum_{i=o}^{n} POT_i$$

where POT_i equals the power-on option time for the i^{th} option which exists and n is the number of existing options

RM-GRA	=	45 µs
RM-CON	=	0 µs
RM-SCR	=	0 µs
RM-SCA	=	0 µs
RM-STA	=	50 µs
RM-FNT	=	1493 µs
RM-PER	=	957 µs
RM-USR	=	0 µs

5-14 Initialize Instruction Timing

Execution Time = 4.97 ms

5-15 Options Instruction Set Execution Times

The following sections describe the execution timing for the set of RM-9000 options instructions. For normal-format options instructions, reference should be made to Section 5-2 for the parameter operand processing time POPT.

5-16 Write Vector Instruction Timing

Execution Time = 220 µs + POPT

$$+ \sum_{i=1}^{NV} \left[VST_i + \sum_{j=1}^{NPV_i} VPT_{ij} \right]$$

$$\cong 220\ \mu s + POPT + \sum_{i=1}^{NV} \left[480\ \mu s + \left[NPV_i * 36.3\ \mu s \right] \right]$$

where NV = Number of linked vectors to be generated in a given WV instruction

VST_i = Vector set-up time for the i^{th} vector (based on the slope of the vector) 480 µs

NPV_i = Number of pixels in the i^{th} vector

VPT_{ij} = Pixel calculation/write time for the j^{th} pixel of the i^{th} vector (dependent on the i^{th} vector's slope and quadrant); minimum value ~ 34.6 µs, maximum value ~ 37.9 µs, average value ~36.3 µs.

5-17 Write Plot Instruction Timings

$$\text{Execution Time} = 543\ \mu s + POPT + \sum_{i=1}^{NP} \left[223\ \mu s + \left[103\ \mu s + \left[1.5\ \mu s * EDIM\right]\right] * EH_i\right]$$

where NP = Number of plot entities to be generated in a given WP instruction

EDIM = Width in pixels of plot entities parallel to the plot axis (defined either by the x-DIMENSION or y-DIMENSION parameter operand depending on the value of SCAN). See Section 3-49.

EH_i = Size in pixels of the i^{th} plot entity perpendicular to the plot axis.

5-18 Write Raster Instruction Timing

$$\text{Execution Time} = 420\ \mu s + POPT + \left[NRB * 130\ \mu s\right] * \left[NRL * 62\ \mu s\right]$$

where NRB = Number of bytes of raster data.

NRL = Number of lines of raster data within the region specified by WINDOW (i.e., number of lines of raster data which are parallel to the primary scan direction).

5-19 Write Conic Instruction Timing

$$\text{Execution Time} = 887\ \mu s + POPT + \left[NCP * CPR\right]$$

where NCP = Number of pixels to be generated per conic.

CPR = Conic pixel generation rate; minimum value ~ 250 µs, maximum value ~ 1160 µs, typical value ~ 300 µs.

NOTE

THE CONIC PIXEL GENERATION RATE IS EXTREMELY VARIABLE, BEING DEPENDENT ON SEVERAL NON-LINEAR PARAMETERS, AND THEREFORE THE VALUE OF CPR IS DIFFERENT FOR EACH PIXEL.

5-20 Scroll X Instruction Timing

Execution Time = 470 µs + POPT

$$+ \left[238\ \mu s * WW\right] + \left[3\ \mu s * WW * WH\right] - \left[122\ \mu s * COUNT\right]$$

$$- \left[1.5\ \mu s * COUNT * WH\right]$$

where WW = Width in elements of the rectangular region defined by WINDOW.

WH = Height in lines of the rectangular region defined by WINDOW.

COUNT = Absolute value of the SCROLL-COUNT parameter operand

5-21 Scroll Y Instruction Timing

Execution Time = 456 µs + POPT

$$+ \left[238\ \mu s * WH\right] + \left[3\ \mu s * WW * WH\right] - \left[122\ \mu s * COUNT\right]$$

$$- \left[1.5\ \mu s * COUNT * WW\right]$$

where WW = Width in elements of the rectangular region defined by WINDOW

WH = Height in lines of the rectangular region defined by WINDOW

COUNT = Absolute value of the SCROLL-COUNT parameter operand

5-22 Save Environment Instruction Timing

Execution Time = 627 µs

5-23 Restore Environment Instruction Timing

Execution Time = 5.05 µs

5-24 Load Programmable Font Instruction Timing

Execution Time = 385 µs

5-25 Load Programmable Font Reverse Packing Instruction Timing

Execution Time = 385 µs

5-26 Write Cursor Instruction Timing

Execution Time = 331 µs for RM-9100
= 299 µs for RM-9200
= 271 µs for RM-9300

5-27 Read Cursor Instruction Timing

Execution Time = 268 µs for RM-9100
= 242 µs for RM-9200
= 219 µs for RM-9300

5-28 Write Keyboard Instruction Timing

Execution Time = 246 µs

5-29 Read Keyboard Instruction Timing

Execution Time = 480 µs (Avercge)

5-30 Sense Peripheral Status Instruction Timing

Execution Timing = 163 µs

5-31 Load Control Memory Instruction Timing

Execution Time = 190 µs + 64 µs * NB

where NB = Number of bytes of data to be loaded into RAM

5-32 Load Control Memory Reverse Packing Instruction Timing

Execution Time = 195 µs + 67 µs * NB

where NB = Number of bytes of data to be loaded into RAM

5-33 Read Control Memory Instruction Timing

Execution Time = 212 µs + 12.5 µs * NB

where NB = Number of bytes of data to be read from RAM

5-34 Read Control Memory Reverse Packing Instruction Timing

Execution Time = 264 µs + 0.75 µs * NB

where NB = Number of bytes of data to be read from RAM

5-35 Call Control Memory Instruction Timing

Due to the variable nature of the user-written subroutine, the timing cannot be defined in this document.

5-36 Execute Instruction Memory Instruction Timing

Due to the variable number of instructions that can be executed by the XIM instruction, the execution time is variable and approximately equal to the sum of execution times for each display instruction.

framtek

APPENDIX A

TYPE II VIDEO LOOK-UP TABLE

FUNCTIONAL DESCRIPTION

Famtek

A-0 INTRODUCTION

The RM-9000 Type II Video Board contains a 1024_{10} by 12_{10} bit video look-up table which is loadable by the host processor and which defines the functional correspondence between all possible data values stored in refresh memory and the actual color or grey scale intensity generated for each pixel. Due to chassis size, the RM-9100 and RM-9200 systems may contain up to two (2) Type II Video Boards; the RM-9300 system may contain up to four (4) of these modules.

A-1 TYPE II VIDEO LOOK-UP TABLE

The video look-up table (or VLT) on a Type II video board is comprised of 1024_{10} addressable locations; each locarion in the VLT is 12 bits in length. The VLT is loaded from the host processor using the special-format LOAD AUXILIARY MEMORY instruction (See Section 3-42). Readback from the VLT to the host processor is implemented via the special-format READ AUXILIARY MEMORY instruction. In standard configurations, the first Type II VLT is addressed as Device 0 in the LAM and RAM, the second VLT as Device 1. It is possible for a user to specify the addressing configurations for access to all VLT's in the system.

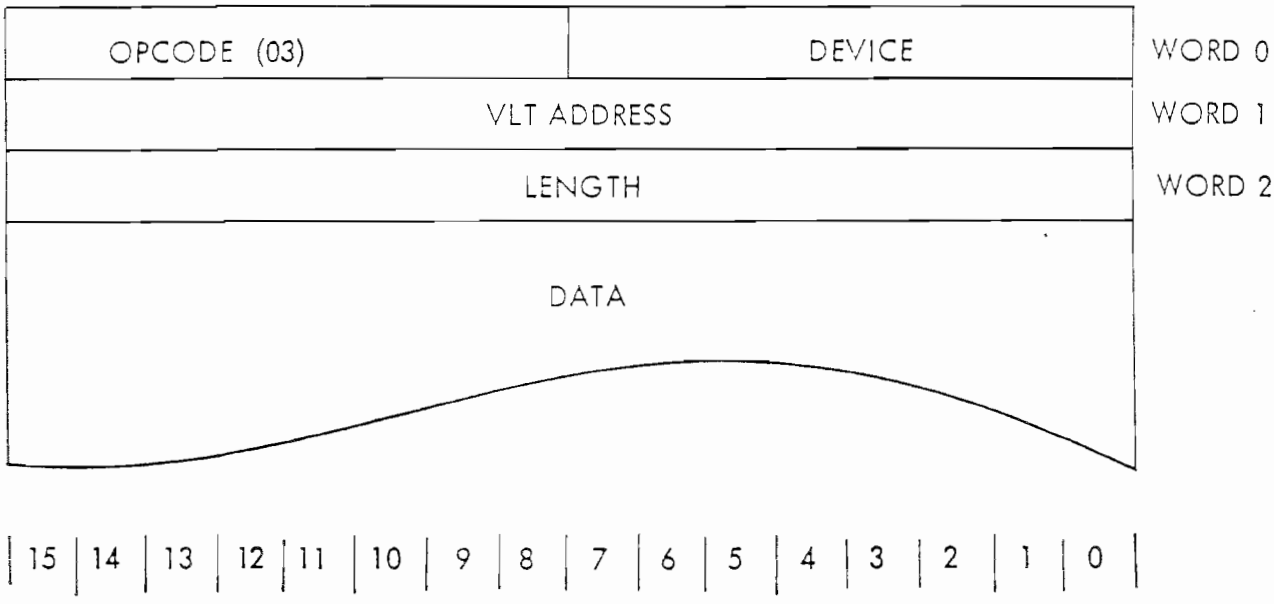


Figure A-1 Lam Instruction Format

The low-order byte of the word 0 of the LAM instruction defines the VLT to be loaded. Word 1 defines the 10_{10} - bit address at which VLT loading will begin. Bits 10-15 in the VLT ADDRESS word are ignored; only Bits 0-9 are effective. Internal addressing of the VLT is designed so that addresses greater than 1023_{10} wrap back to Address 0. The LENGTH word defines the number of bytes of VLT data to be loaded from the hostprocessor. Since loading is done on a full-word basis, the LENGTH value should always be even. The actual DATA to be loaded into the VLT RAM follows the LENGTH word. The low-order 12_{10} bits of each 16_{10}-bit word from the host processor are loaded into successive locations of the VLT. The functional use of the 12 bits of data loaded into the VLT is dependent on the Type II Video board configuration (discussed in Section A-2).

The RAM instruction is used to readback VLT data values from the Type II Video board. The definition of the DEVICE, VLT ADDRESS and LENGTH values is the same as those of the LAM instruction. The data from the VLT is stored in Bits 0-11 of the readback data word. Bits 12-15 of the readback data word are indeterminate.

Note that if a Type II VLT, which does not exist in an RM-9000, is accessed via the LAM or the RAM instructions, the RM-9000 system will hang and can only be cleared by a hardware reset.

A-2 TYPE II VIDEO CONFIGURATIONS

The function of the Type II Video board is to generate video signals to the CRT monitor based on the refresh memory data value for each pixel and the VLT RAM value stored for the corresponding location using the pixel value as an address into the VLT. For example, if pixel (0,0) contained a value of two (2) in refresh memory, the data stored in location two (2) of the VLT would define the color or grey scale intensity actually displayed for that pixel. All pixels with a refresh memory data value of two (2) would be displayed with the same color or grey scale intensity.

There are four (4) video outputs from each Type II Video board: three (3) outputs from three (3) 4-bit DAC's (digital-to-analog converters) and one (1) video output from an 8-bit DAC. The actual BNC connections for each of these four (4) video outputs are video slot-dependent and are available in Section 2 of the RM-9000 Theory of Operation Manual, Volume 1.

The three (3) 4-bit DAC outputs designated as RED, GREEN and BLUE outputs correspond to the VLT RAM values defined by Bits 8-11, Bits 4-7 and Bits 0-3, respectively. The 8-bit DAC video output can be configured such that it repre-

sents either the low-order 8 bits of output from VLT or the binary value of Subchannel 0 thru 7 from refresh memory directly.

A-3 TYPE II VLT PARTITIONING

It is possible to configure any Type II VLT such that less than 10 subchannels from refresh memory are used to generate the video outputs. Since the full 1024_{10} x 12_{10} bit RAM is always present, the Type II VLT is "partitioned" or divided up into 2^m sections of equal length where m is the number of high-order subchannels not used as address input to the VLT for video generation. Thus for example, if subchannels 8 and 9 were not used for input to the VLT, then four (4) partitions of 256_{10} locations each would be available.

All partitions are identical to each other, varying only in the address by which they are loaded and read from the host processor. In the example where subchannels 8 and 9 were not used, four (4) partitions of 256_{10} locations each are available. They are addressed as VLT locations 0 through 255_{10}, 256_{10} through 511_{10}, 512_{10} through 767_{10} and 768_{10} through 1023_{10}. Only one (1) of these partitions is used to generate the actual video outputs at any given time. The last VLT location addressed defines the partition to be used. When using the LAM or RAM instructions, the last VLT location addressed is one greater than the last VLT location actually written with data. Therefore, a LAM instruction with a length of 256_{10} words starting at VLT Address 0 would select the second partition on a VLT which did not use Subchannels 8 and 9. Correspondingly, partitioning of the Type II VLT is not possible in a system with all 10 subchannels utilized.

Famtek

APPENDIX B

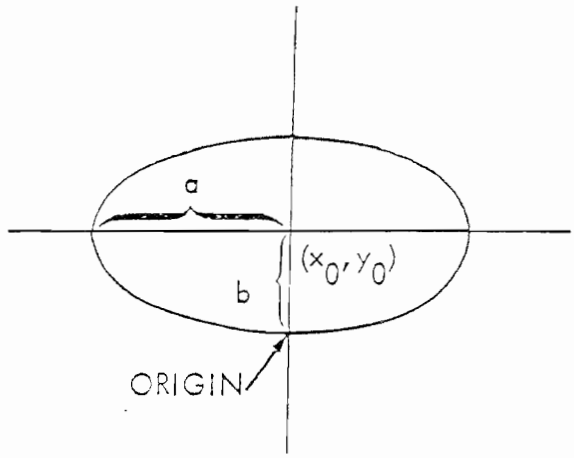
GENERATION OF CONICS

Famtek

B-1 HOW TO DRAW AN ELLIPSE

In order to draw the various conic sections, it is necessary to find their equations, and more particularly their equations which go through the origin. An example would be in order at this point.

Consider an ellipse whose semimajor axis (a) and semiminor axis (b) are parallel to the coordinate axes, with center point (x_0, y_0).



The equation for such an ellipse is

$$\frac{(x-x_0)^2}{a^2} + \frac{(y-y_0)^2}{b^2} = 1.$$

The origin must be chosen to lie on the conic locus, and in order to simplify the algebra, it is chosen so that $x_0 = 0$. In this case $y_0 = b$.

The general equation can be expanded to

$$\frac{x^2 + x_0^2 - 2xx_0}{a^2} + \frac{y^2 + y_0^2 - 2yy_0}{b^2} = 1$$

and the values for x_0 and y_0 substituted into the equation to yield

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{2y}{b} = 0.$$

This fits into our RAMTEK conic equation form

$$Ax^2 + By^2 + Cxy + Dx + Ey = 0$$

where

$$\begin{aligned} A &= 1/a^2 \\ B &= 1/b^2 \\ C &= 0 \\ D &= 0 \\ E &= -2/b. \end{aligned}$$

Thus an ellipse with a semimajor axis length of 60 pixels, and a semiminor axis length of 20 pixels would have values

$$\begin{aligned} A &= 1/60^2 = .000278 \\ B &= 1/20^2 = .0025 \\ C &= 0 \\ D &= 0 \\ E &= -2/20 = -.1\ . \end{aligned}$$

In order to make all the arguments integers, these values must be multiplied by some arbitrarily large number. If the arbitrarily large number chosen is 20,470 the values obtained would be

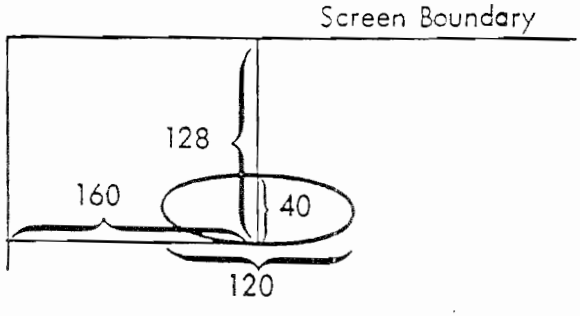
	Decimal	Hex
A =	6	$= 0006_{16}$
B =	5	$= 0029_{16}$
C =	0	= 0000
D =	0	= 0000
E =	-2047	= -03FF = FC01 (twos complement) .

It should at this time be noted that the conics arguments are each two words long. The first word of each argument is ignored by the RM-9000. Only the second word of each argument is taken to be meaningful. Negative values are taken to be twos complement, limiting the range of the arguments to integers between $7FFF_{16}$ and -8000_{16}.

With all these things in mind, the entire complex instruction for the ellipse could be formatted as follows:

0F02	Write conic instruction
9000	Write conic and start point operand flags
0000	First null word
0006	Value of A
0000	Second null word
0029	Value of B
0000	Null word
0000	Value of C
0000	Null word
0000	Value of D
0000	Null word
FC01	Value of E
0000	Null word
0000	Value of K
00A0	Value of x start point
0080	Value of y start point
0004	data flag word indicating 4 bytes to follow
00A0	Value of x end point
0080	Value of y end point

The image drawn on the screen would look as follows:



B-0 THE CONIC EQUATION

According to the method of Cartesian geometry, the generalized equation for any conic is of the form

$$\alpha x^2 + \beta y^2 + \gamma xy + \delta x + \varepsilon y + \zeta = 0.$$

Note that not every equation of this form results in real solutions that can be plotted. Some quadratic equations have complex solutions that cannot be plotted by ordinary methods, and should not be attempted on the RAMTEK system.

To fit the algorithm used in the RM-9000, the ζ term must be dropped from the general form of the equation. This does not limit the number of actual conic sections that can be drawn with the system, however, since any conic equation containing ζ can be rewritten in a form without a constant term with only a minimum of algebraic manipulation. This stipulation can be restated as: the conic section locus must pass through the Cartesian origin.

The origin on the RM-9000 can be placed anywhere on the screen. Exactly where the origin is placed is specified in raster elements and lines from the upper left hand corner. This is done using the start-point arguments.

It is possible to draw a specific section of a conic if desired. In fact it is necessary to do so in the cases of parabolas and hyperbolas, which by their nature contain an infinite number of points. The RM-9000 allows the user two methods of terminating a plot . Either a specific number of points to be drawn may be specified, or a particular end point (relative to the upper left hand corner of the screen) may be specified.

In the case of conic equations with an infinite number of point solutions, or in the case of partial ellipses, it is usual to specify a certain maximum number of pixels to be illuminated. This information is transmitted to the RM-9000 via the κ-value argument. In the case of a $\kappa = 0$ being specified, κ assumes the default value of 1280 (decimal).

In the case of conic sections known to pass through a specific point, the specific point may be specified in two (2) words following the data flag word. The first word is the element end point, the second word is the line end point, relative to the screen origin. This is the usual method for terminating complete ellipses, for which the end point is set equal to the start point.

It should be kept in mind that the ratio between the distance between picture elements along a raster scan line and the distance between raster scan lines is not always 1:1. This is dependent upon how the monitor is adjusted and the resolution of the RM-9000 model being programmed. If the ratio is not 1:1, a circle will not be round. If this is important, there are two ways to make the circle round. The first, and by far the simplest way of handling this problem is to adjust the height and/or width adjustments on the monitor until the circle becomes round. If this is impossible or undesirable, the programmer can make the circle round by finding the ratio empirically, and adjusting semimajor and semiminor axes lengths accordingly.

B-2 How To Rotate The Ellipse

Analytic geometry reveals that when an ellipse is rotated by an angle about the origin, the equation becomes

$$\left(\frac{\cos^2\theta}{a^2} + \frac{\sin^2\theta}{b^2}\right) x^2 + \left(\frac{\sin^2\theta}{a^2} + \frac{\cos^2\theta}{b^2}\right) y^2$$

$$+ 2\cos\theta\ \sin\theta\left(\frac{1}{a^2} - \frac{1}{b^2}\right)xy - \frac{2\sin\theta}{b}\,x + \frac{2\cos\theta}{b}\,y = 0\ .$$

In terms of Write Conic arguments:

$$A = \frac{\cos^2\theta}{a^2} + \frac{\sin^2\theta}{b^2}$$

$$B = \frac{\sin^2\theta}{a^2} + \frac{\cos^2\theta}{b^2}$$

$$C = 2\cos\theta\ \sin\theta\ \left(\frac{1}{a^2} - \frac{1}{b^2}\right)$$

$$D = \frac{-2\sin\theta}{b}$$

$$E = \frac{2\cos\theta}{b}$$

For example, if the previous ellipse is rotated by $\pi/6$:

$A = .000278 \cos^2 \pi/6 + .0025 \sin^2 \pi/6 = .000833 \ \alpha \ 20$

$B = .00278 \sin^2 \pi/6 + .0025 \cos^2 \pi/6 = .00194 \ \alpha \ 46$

$C = 2\,(.00278) \cos \pi/6 \sin \pi/6 - 2\,(.0025) \sin \pi/6 \cos \pi/6 = -.00192 \ \alpha \ -45$

$D = -0.1 \sin \pi/6 = -.05 \ \alpha \ -1182$

$E = -(-.1) \cos^2 \pi/6 = .0866 \ \alpha \ 2047.$

B-3 Rotating About The Center Point

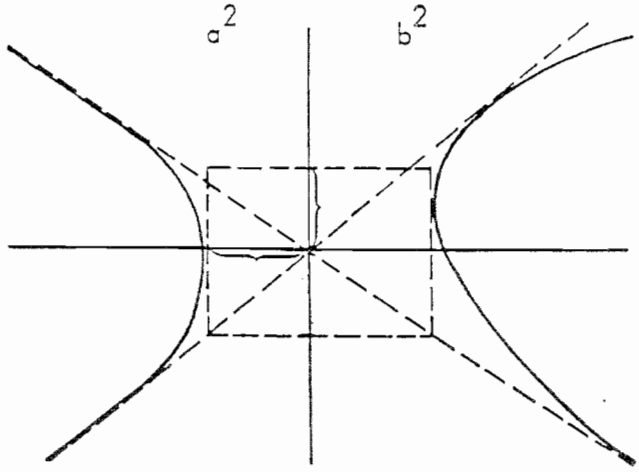
It is sometimes desirable to rotate about the center point of an ellipse rather than the origin. It is a simple trigonometry problem to show that rotating about the center point is equivalent to rotating about the origin and shifting the start point:

$$\Delta x = b \sin\theta$$
$$\Delta x = b\,(1-\cos\theta) \ .$$

B-4 HOW TO DRAW A HYPERBOLA

The same kind of analysis performed on the ellipse can be performed on a parabola of hyperbola. Recall the following equation for a hyperbola in terms of its semimajor and semiminor axes:

$$\frac{(x-x_0)^2}{a^2} - \frac{(y-y_0)^2}{b^2} = 1 \ .$$



The equation may be expanded to

$$\frac{x^2 + x_0 - 2xx_0}{a^2} - \frac{y^2 + y_0^2 - 2yy_0}{b^2} = 1.$$

To produce a hyperbola it is still necessary to choose the origin to be a point on the curve. As a first attempt we might choose the origin to be at the vertex pictured above, since at this point we have $y_0 = 0$ and $x_0 = a$. This makes it a simple task to write the equation for the hyperbola in terms of two parameters:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} - \frac{2x}{a} = 0$$

$$A = 1/a^2$$

$$B = -1/b^2$$

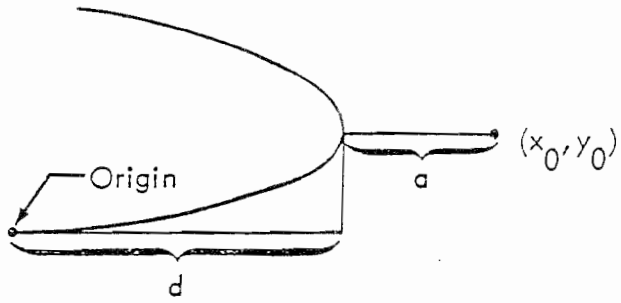
$$C = 0$$

$$D = -2/a$$

$$E = 0 .$$

If we put these exact argumental values into the WC arguments, however, the following image will be produced.

This is the hyperbola from the vertex to the assymptote. To draw from assymptote to assymptote, the origin must be moved to the assymptote.



At this point $x_0 = a + d$ and $y_0 = b\sqrt{\frac{d^2}{a^2} + \frac{2d}{a}}$

With this information, the equation for the hyperbola can be rewritten

$$\left(\frac{1}{a^2}\right)x^2 + \left(\frac{1}{b^2}\right)y^2 - \frac{2(a+d)x}{a^2} - \left(\frac{2}{b}\sqrt{\frac{d^2}{a^2} + \frac{2d}{a}}\right)y = 0$$

which gives us our coefficients in terms of the axes and the x distance from origin to vertex.

$$A = \frac{1}{a^2}$$

$$B = \frac{1}{b^2}$$

$$C = 0$$

$$D = \frac{-2(a+d)}{a^2}$$

$$E = \frac{-2}{b}\sqrt{\frac{d^2}{a^2} + \frac{2d}{a}}\,.$$

Again we can rotate the conic by θ to get:

$$A = \frac{\cos^2\theta}{a^2} - \frac{\sin^2\theta}{b^2}$$

$$B = \frac{\sin^2\theta}{a^2} - \frac{\cos^2\theta}{b}$$

$$C = 2 \cos\alpha \sin\alpha \left(\frac{1}{a^2} + \frac{1}{b^2} \right)$$

$$D = \frac{2}{b} \left(\frac{d^2}{a^2} + \frac{2d}{a} \right) 1/2 \sin\theta - \frac{2(a+d)}{a^2} \cos\theta$$

$$E = \frac{2}{b} \left(\frac{d^2}{a^2} + \frac{2d}{a} \right)^{1/2} \cos\theta - \frac{2(a-d)}{a^2} \sin\theta .$$

As an example, for a = 16, b = 7, d = 200 and θ = /6:

$$A = -.00217 \quad \alpha \quad -1$$
$$B = -.01933 \quad \alpha \quad -9$$
$$C = .02106 \quad \alpha \; 13$$
$$D = -3.385 \quad \alpha \quad -2047$$
$$E = 2.487 \quad \alpha \quad 1504 .$$

B-5 SPECIAL CASES

Degenerate conics and complete ellipses of less than 6 pixels will not be drawn accurately in the RAMTEK system. It should also be noted that the conic sections drawn by the RM-9000 series are, by the very nature of the raster scan, necessarily drawn in quantitized segments. The greater the resolution, the smoother the conic will appear. Since the RM-9000 chooses to illuminate those pixels closest to the theoretical conic section, it is possible that slight asymmetries will appear if the theoretical conic section does not correspond sell to the element and line grid of the screen.

It is useful to note that a hyperbola will be degenerate if and only if the equation can be factored into two first degree factors (the loci of two straight lines). A degenerate ellipse is but a single point.

famtek

APPENDIX C

INTERACTIVE PERIPHERALS OPERATING PROCEDURES

famtek

C-0 INTRODUCTION TO INTERACTIVE PERIPHERALS

There are three types of interactive functions which are available within the RM-9000 display system:

1) Cursor Interaction
2) Joystick/Trackball Interaction
3) Keyboard/Transmitter Interaction

In order to implement and access these interactive peripheral functions, a system must contain the RM-SLC serial link card and the RM-PER interactive peripherals option firmware package.

The purpose of these devices is to allow the human user to interact or communicate with the host processor application program in a machine-readable form, allowing visual feedback through the display system. The use of the cursor and joystick/trackball mechanism is particularly useful in this area.

C-1 CURSOR DESCRIPTION

The RM-SLC serial link card, in addition to support input and output from the keyboard and joystick/trackball devices, also supports the cursor controller/generator hardware. Cursors can be controlled by either a joystick or trackball or by the host processor using the WRITE CURSOR STATE and READ CURSOR STATUS instructions. Up to two cursors may be generated by each RM-SLC card; since up to two RM-SLC cards can be present in a given system, this means that a maximum of four cursors are possible.

C-2 Standard Cursor Pattern

The Display Controller is capable of generating up to four cursors. The cursor appears as a cross (+) on the screen with the center element missing in the configurations and sizes shown in Figure C-1. For the same size monitor, the cursor retains the same physical size regardless of the number of lines/elements or resolution of the system. Referring to Figure C-1, the cursor shown in block "A" is representative of systems with 240 or 256 lines and 320 elements. The "B" block cursor is configured in systems of 240 or 256 lines and 640 elements. The "C" block illustrates cursor for 480 or 512 lines and 640 elements.

famtek

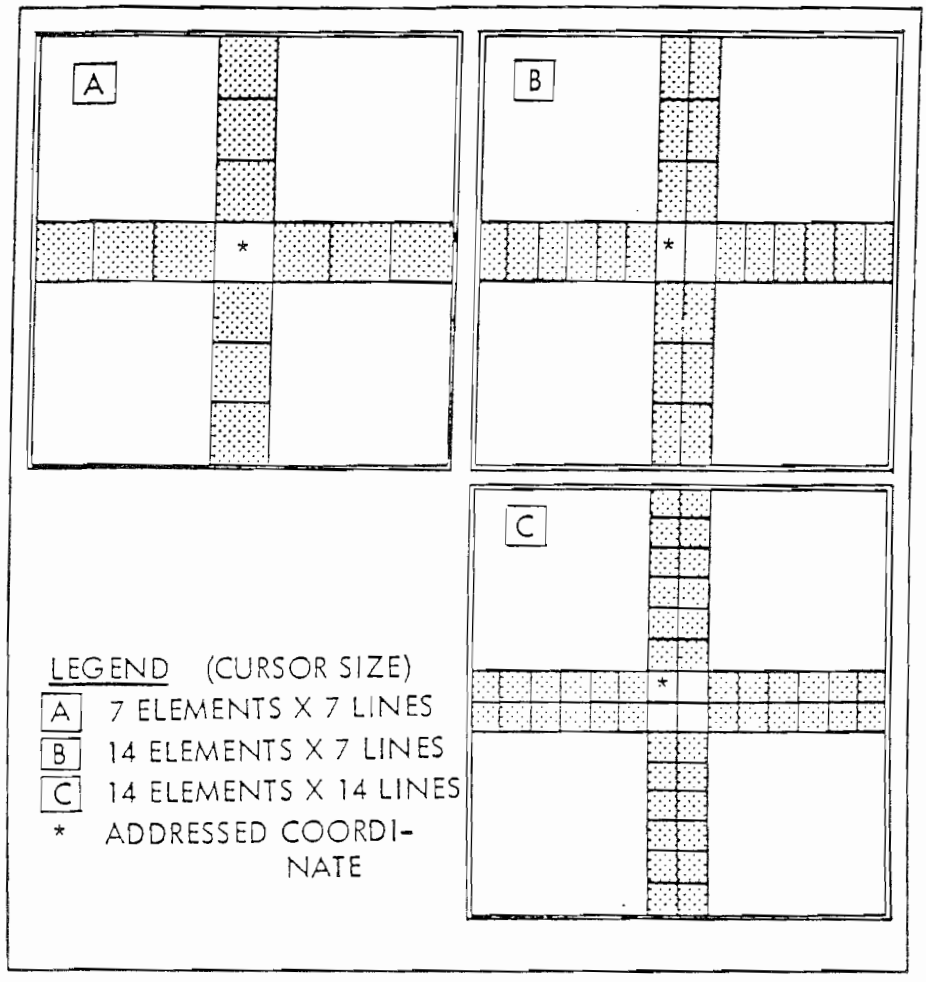


Figure C-1 Cursor Configuration

C-3 Cursor Screen Addressing

The cursor may be placed anywhere on the screen, including non-visible elements and lines in horizontal and vertical blanking. In other words, the cursor will not come to a stop at the edges of the visible portion of the screen, but will go beyond the edges and disappear into blanking. During CPU control of the cursor this is relatively unimportant as the CPU normally is programmed not to load data into non-visible elements. However, under joystick or trackball control, the operator may guide the cursor into blanking. If the same direction is maintained, "roll-over" will eventually occur and the cursor will reappear on the screen on the side opposite to that from which it disappeared, still maintaining the same direction.

C-4 JOYSTICK CURSOR CONTROLLER - GC-106

The joystick cursor controller is an interactive peripheral device used to position a cursor upon a video graphic display. The cursor controller consists of a joystick, four status switches (ENTER, TRACK, VISIBLE and BLINK), four channel select switches and a power switch. The controller interactively positions the cursor via the joystick, controls cursor status with the VISIBLE and BLINK status switches and informs the CPU of current coordinates and status by the ENTER and TRACK switches.

The cursor controller and trackball (See Section C-7) operate with the serial link option in an identical manner. Both use serial transmission lines to send data. The serial link option stores cursor coordinates and status while generating the cursor video image. The cursor controller does not store cursor coordinates, but issues increment/decrement commands to the serial link board which in turn update the cursor position on the screen. Since the amount of displacement of the joystick from the center position proportionately changes the rate of increment/decrement commands issued by the cursor controller, the further the joystick is displaced the faster the cursor moves on the screen. With a little practice, positioning of the cursor with the controller is simple, fast and more efficient than a trackball.

The cursor controller is a directional rate device and not a positional control device. That is, when the joystick is moved in any direction from the center (at rest) position, the cursor begins to move slowly in the direction the joystick was displaced. The further the joystick is displaced from center, the faster the cursor moves in that direction. When the joystick is held in a constant position, the cursor moves across the screen at a constant rate. Release of the joystick returns it to the spring loaded position and stops cursor movement.

The joystick may be displaced at any single angle even though it feels easier to move the stick directly up, down, right or left. When viewed from the front, the

position of the joystick corresponds exactly with the direction of the cursor movements as shown in Table C-1.

Table C-1 Joystick/Cursor Movement

Joystick Movement	Cursor Movement
Forward	Up
Backward	Down
Left	Left
Right	Right

The rate of cursor movement in any axis is infinitely variable from about 1 element/second when the stick is displaced ≈ 5° deflection, to a maximum of traversing the screen from one edge to the other in about 3 seconds (full deflection). This mode of operation is used to move the cursor quickly from point to point.

A second mode of joystick operation allows one element cursor movement in any direction to easily and accurately position the cursor on a single screen element. To move the cursor one element only, the joystick is slightly displaced or "bumped". This action causes the cursor to move one element or line in the direction of joystick displacement. The cursor will not move any more elements until the stick is released and "bumped" again, or displaced further to start cursor movement as defined in the above mode.

This unique feature of the cursor controller allows the operator to be assured of moving the cursor one and only one element in any direction for ease in accurate positioning. The joystick displacement versus rate of cursor movement curve is not linear but exponential.

There is a small null zone around the center position of the joystick so that minimal displacements do not cause cursor movement. This prevents the cursor from "creeping" on the screen when the joystick is centered. The null zone also allows the cursor controller to be used without requiring trim adjustments of compensation for drift effects.

C-5 JOYSTICK STATUS CONTROL SWITCHES

Four status switches determine the status of the cursor on the screen and control host processor interrupt generation. These switches are described as follows:

(a) VISIBLE - This alternate action switch turns the cursor ON or OFF. Cursor coordinates are not affected by the position of this switch.

(b) BLINK - The BLINK switch is an alternate action switch that, when ON, causes the cursor to blink at approximately a 1 Hz rate. When BLINK is OFF, the cursor remains steady on the screen. Cursor coordinates remain unaffected by the position of the BLINK switch.

(c) ENTER - ENTER is a momentary switch which causes a cursor interrupt (when enabled) to be sent to the CPU regardless of the position of any status switch or the position of the joystick. If the ENTER switch is held ON, the cursor controller ceases to function until the switch is released. As soon as the ENTER switch is released, the cursor controller resumes normal operation.

(d) TRACK - When ON, this alternate action switch causes every movement of the cursor to generate a host processor interrupt. Every movement of the cursor is defined to be a change in coordinates. When the TRACK switch is OFF, the cursor still moves on the screen, but the cursor interrupt not issued to the host processor.

C-6 Joystick Cursor Selection Switches

Using the four channel select switches, the operator can control up to four cursors simultaneously with one cursor controller unit. These alternate action switches cause the output of the controller to be distributed to the output channel(s) selected by the switches. When a switch for any channel is ON, the output of the controller appears on the serial output for that channel. When the switch is OFF, the serial output for that channel goes to an idle or no transmission mode. Any combination of switches can be ON simultaneously including all ON or all OFF.

NOTE

CHANNEL SELECT SWITCHES SHOULD NEVER BE CHANGED WHILE MOVING THE CURSOR WITH THE JOYSTICK OR WHILE SWITCHING THE STATUS SWITCHES. SINCE THE CONTROLLER OPERATES WITH A SERIAL OUTPUT LINE, CHANGING THE CHANNEL SELECT SWITCHES WHILE THE UNIT IS TRANSMITTING MAY CAUSE UNPREDICTABLE RESULTS OF CURSOR MOVEMENT OR STATUS.

As long as the joystick is centered and the status switches are stationary, the channel select switches can be changed with no effects. Power does not need to be OFF to change the channel select switches.

C-7 TRACKBALL CURSOR CONTROLLER - GC-104-2

The Model GC-104-2 trackball is a self-contained unit providing a serial output to the cursor controller for cursor updating. The trackball itself is a free-turning ball mounted in the GC-104-2 chassis. Cursor update is in the direction of rotation of the ball and the update rate is based upon the linear speed of rotation. The GC-104-2 trackball associated with the RM-9000 display system generates serial outputs at 2400 band.

A serial character is generated whenever any of the control switches changes and whenever the trackball is moved. The four changes of state, which are switch-controlled, are BLINK, VISIBLE, ENTER and TRACK. These have been described previously in the GC-106 joystick description (See Section C-5).

C-8 PROCESS CONTROL KEYBOARD - GK-120

The GK-120 Process Control Keyboard contains the following items as standard:

(a) A 61 key typewriter keyboard that generates all 128 USASC11 codes, including all alphanumeric, graphic and control characters. The keyboard features two key rollover/n key lockout operation. An auto-repeat feature of all keys is standard. Key arrangement is similar to the Model 37 teletype.

(b) Serial input and output options using a choice of EIA Standard RS-232C, TTY current loop, or short-line differential. Baud rates from 50 to 9600 baud; odd, even or no parity and one or two stop bits are selectable by the user.

(c) TTY mode operation allowing the keyboard to appear as a Model 33 teletype, generating a standard 93 character subset of USASCII. (Upper case alpha characters only).

(d) ON/OFF LINE switch allowing flexibility in the mode of operation of the keyboard.

(e) A 12 key cursor/function pad containing cursor up, down, right, left and home commands for computer controlled cursor. The remaining seven keys provide easily accessible function keys defined by the user.

(f) A convenient 12 key numeric pad containing 10 digits and 2 delimiter keys (. and ,). This pad can be modified to provide 12 additional function keys defined by the user.

Optional features include 16 special function keys defined by the user with corresponding CPU controlled status lights. Up to 40 function keys can be provided, each assigned with two codes per key. All function keys generate eight bit codes above ASCII (Octal 200-377). An attention signal activated by the reception of the USASCII defined "Bell" code.

All codes are 8 bit data. The code assignments are shown in Table C-2.

The alphanumeric section contains all functions including alphanumeric, control and graphic characters as defined by USASCII. Octal codes 000 through 177 are generated here.

If the numeric pad is selected, the codes issued are strictly the numeric and delimiter codes of the alphanumeric section. That is, when the keyboard is in the unshifted, uncontrolled mode, pressing the numeral "4" will issue the ASCII code for "4". When in the shifted mode, pressing "4" will also issue the ASCII code for "4".

If the function pad replaces the numeric pad, all functions are defined by the user. The codes issued are above ASCII, Octal 200 and higher.

The cursor pad contains the five cursor controls: cursor up, down, right, left and home. Each key has two codes associated with it providing the ability for slow and fast computer controlled cursor. The remaining 7 keys in the cursor pad are defined by the user. The codes issued are above ASCII, Octal 200 and higher.

If the 16 special function keys are included on the keyboard, all functions are defined by the user. Note that each key is assigned with two codes. The codes are above ASCII, Octal 200 and higher.

Due to MOS encoding techniques, note that all codes are preassigned by Ramtek and are unalterable. However, the user is free to assign function meaning and keytop legends to all keys.

The shift and control keys determine the coded output of all keys. For example, consider the key labeled "A" where "-" indicates key not pressed, "x" indicates key not pressed, "x" indicated key is pressed.

Shift	Control	Function	Octal Code Output
-	-	a	141
x	-	A	101
-	x	SOH	1
x	x	SOH	1

The control key allows only ASCII defined control characters (Octal 000-040 and 177) and all codes not defined by ASCII (Octal 200-377) to be serially outputted. All other codes are inhibited from appearing at the serial outputs when the control key is pressed. This is equivalent to a key being mechanically locked out on a teletype. The shift key does not lockout any codes. As an example of the control key locking out the above indicated codes consider the key labeled with the numeral "3".

Shift	Control	Function	Octal Code Output
-	-	3	63
x	-	3	43
-	x	3	Not Outputted (Lockout Operation)

Notice that both the shift and control keys determine the coded output of the keys in the alphanumeric section, but for all other sections (cursor pad, numeric/function pad and special functions), the shift key has no effect. Only the control key will determine the coded output for the keys in these sections. There are two shift keys on the keyboard with one alternate action shift lock. Only one control key is provided.

Placing the keyboard in TTY mode causes the logic to suppress lower case alpha characters. When in TTY mode, the keyboard generates a 93 character subset of ASCII. Only the keys in the alphanumeric section are affected. The codes for all other sections of the keyboard (cursor pad, numeric/function pad and special function) remain unaffected by TTY mode.

In TTY mode, the codes are altered using the following rules.

(a) ASCII

1) Octal codes 000 to 137: retain as is.
2) Octal codes 140 to 176: subtract Octal 40 to convert these codes to upper case characters. That is, convert 140 through 176 to 100 through 136.
3) Octal code 177 (delete): retain as is.

(b) NON-ASCII

1) Octal codes 200 to 377: retain as is.

This information is tabulated in Table C-2.

The following example shows how TTY mode affects keyboard behavior. In the unshift, uncontrol state, when the operator presses the key "A", normally the code for small letter "a" is issued (Octal 141). However, in TTY mode, the capital letter "A" is issued instead (Octal 101). This is shown as follows:

Shift	Control	Function	Octal Code Output
-	-	A	101
x	-	A	101
-	x	SOH	1
-	x	SOH	1

famtek

Table C-2 GK-120 TTY Mode Control

	0	1	2	3	4	5	6	7
00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL
01	BS	HT	LF	VT	FF	CR	SO	SI
02	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB
03	CAN	EM	SUB	ESC	FS	GS	RS	US
04	SP	!	"	#	$	%	&	'
05	(	)	*	+	,	-	.	/
06	0	1	2	3	4	5	6	7
07	8	9	:	;	<	=	>	?
10	@	A	B	C	D	E	F	G
11	H	I	J	K	L	M	N	O
12	P	Q	R	S	T	U	V	W
13	X	Y	Z	[	\	]	^	_
14	`	a	b	c	d	e	f	g
15	h	i	j	k	l	m	n	o
16	p	q	r	s	t	u	v	w
17	x	y	z	{	|	}	~	DEL

ASCII (octal)

Alphanumeric Section
Numeric Pad

Uncontrol

Control

	0	1	2	3	4	5	6	7
20	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
21	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
22	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}	$\overleftarrow{f_{23}}$	HOME f_{24}
23	$\overleftarrow{f_{25}}$	f_{26}	f_{27}	f_{28}	f_{29}	f_{30}	f_{31}	f_{32}
24	f_{33}	f_{34}	f_{35}	f_{36}	f_{37}	f_{38}	f_{39}	f_{40}
25								
26								
27								
30	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
31	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
32	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	$\overleftarrow{f_{22}}$	f_{23}	HOME f_{24}
33	$\overleftarrow{f_{25}}$	f_{26}	f_{27}	f_{28}	f_{29}	f_{30}	f_{31}	f_{32}
34	f_{33}	f_{34}	f_{35}	f_{36}	f_{37}	f_{38}	f_{39}	f_{40}
35								
36								
37								

NON-ASCII (octal)

Cursor Pad
Function Pad
Special Functions
Status Lights

The auto repeat feature provides that any key, if held down continuously for longer than one second, will automatically repeat until the key is released. If more than one key is held down, only the first key struck will repeat. The output will not alternate between two keys. The time delay until any key begins to repeat is normally one second. The repeat rate is 10 characters per second for all keyboards, regardless of baud rate. The cursor keys have been defined with two codes per key allowing the ability to distinguish between slow and fast cursor commands. Both the time delay and repeat rate are hard-wired and are not under CPU control.

famtek

Two key rollover/n-key lockout is provided such that after a key closure is recognized by the keyboard logic and the appropriate code for that key is issued, all further key depressions are ignored. No further codes will be issued until the first key is released.

N-Key Lockout - The following characteristics are to be expected:

(a) LOCKOUT - If a key is depressed and not released, the code for that key is issued and the keyboard scan stops, locking out any recognition of further key depressions. Any other keys depressed and released will not be recognized. As soon as the original key is released, keyboard scan resumes until another key depression is found.

(b) 2-KEY ROLLOVER - A 2-key rollover can be experienced with n-key lockout. Depress the first key and its code is issued, stopping keyboard scan. Press the second key and no code is issued. While still holding the second key down, release the first key (starting keyboard scan) and the second key will now be recognized and its code will be issued. Keyboard scan now stops again until the second key is released. This chain action can be continued indefinitely.

(c) MULTIPLE DEPRESSIONS - If a key is depressed and held, its code will be issued. While continuing to hold the first key, if additional keys are depressed and held, no further codes will be issued until the first key is released and the next code issued will be unpredictable. It depends upon which key is encountered first by the scanning mechanism and only that one code will be issued.

(d) SIMULTANEOUS DEPRESSION - The first code issued is unpredictable. It depends upon the current position of the keyboard scan mechanism. Only one code will be issued.

The CPU can sound an attention signal by using the ASCII defined code for "bell" (BEL = Octal Code 007). The signal will remain active for approximately one second after the keyboard reception of the "bell" code. If a continuous signal is required, the CPU can retrigger the attention device as many times as required. The signal can be retriggered at any point in its cycle. However, the signal will only remain active for one second after the last "bell" code received. The attention signal is Mallory's SONALERT^{C} with a fixed frequency of 2,900 Hz.

Two key rollover/n-key lockout is provided such that after a key closure is recognized by the keyboard logic and the appropriate code for that key is issued, all further key depressions are ignored. No further codes will be issued until the first key is released.

N-Key Lockout - The following characteristics are to be expected:

(a) LOCKOUT - If a key is depressed and not released, the code for that key is issued and the keyboard scan stops, locking out any recognition of further key depressions. Any other keys depressed and released will not be recognized. As soon as the original key is released, keyboard scan resumes until another key depression is found.

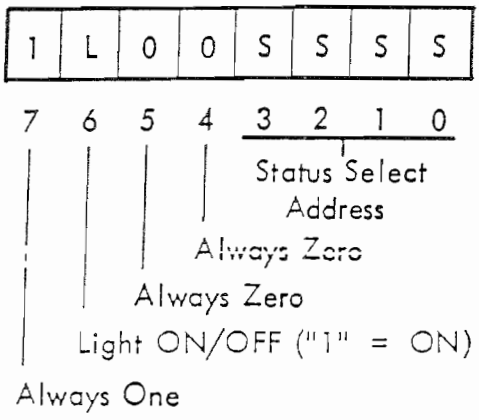
(b) 2-KEY ROLLOVER - A 2-key rollover can be experienced with n-key lockout. Depress the first key and its code is issued, stopping keyboard scan. Press the second key and no code is issued. While still holding the second key down, release the first key (starting keyboard scan) and the second key will now be recognized and its code will be issued. Keyboard scan now stops again until the second key is released. This chain action can be continued indefinitely.

(c) MULTIPLE DEPRESSIONS - If a key is depressed and held, its code will be issued. While continuing to hold the first key, if additional keys are depressed and held, no further codes will be issued until the first key is released and the next code issued will be unpredictable. It depends upon which key is encountered first by the scanning mechanism and only that one code will be issued.

(d) SIMULTANEOUS DEPRESSION - The first code issued is unpredictable. It depends upon the current position of the keyboard scan mechanism. Only one code will be issued.

The CPU can sound an attention signal by using the ASCII defined code for "bell" (BEL = Octal Code 007). The signal will remain active for approximately one second after the keyboard reception of the "bell" code. If a continuous signal is required, the CPU can retrigger the attention device as many times as required. The signal can be retriggered at any point in its cycle. However, the signal will only remain active for one second after the last "bell" code received. The attention signal is Mallory's SONALERT^{C} with a fixed frequency of 2,900 Hz.

Status lights, under host processor control, are provided by the LED's directly above the special function keys. The lights may be turned ON and OFF by the host processor using the WRITE KEYBOARD instruction. The format of the output data byte necessary to set the ON/OFF state of one of the 16 LED's on the GK-120 keyboard. The operator has no control over the operation of the status lights.



Facing the keyboard from the front, the status lights are addressed from 0 to 15 from left to right. Since the operation of the status lights is independent of the special function keys, the user may associate key functions with status lights or completely divorce the two.

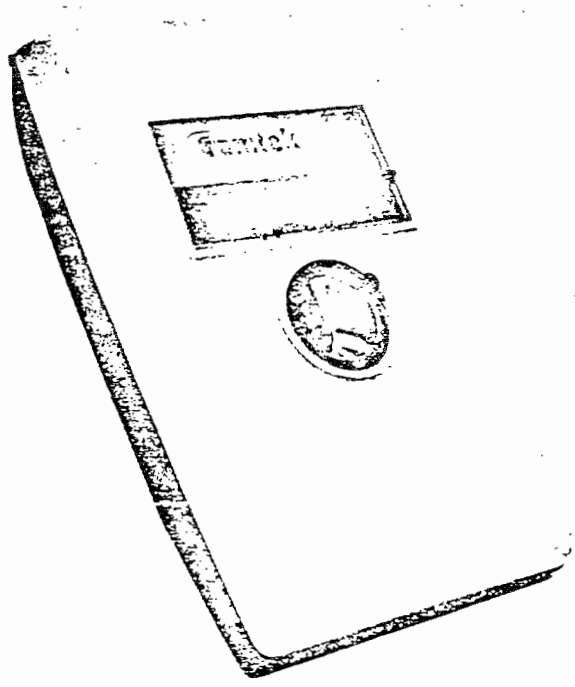


Figure C-2 Joystick Model GC-106

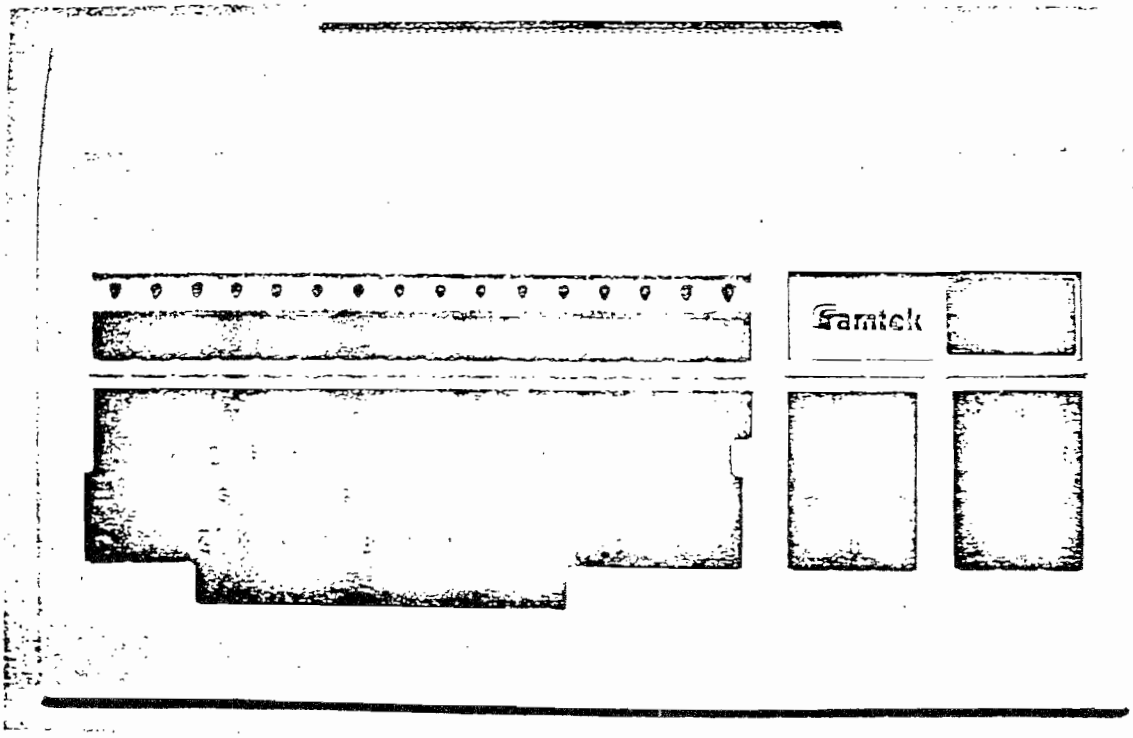


Figure C-3 Keyboard Model GK-120

framtek

APPENDIX D

RM-9000 STANDARD CHARACTER FONTS

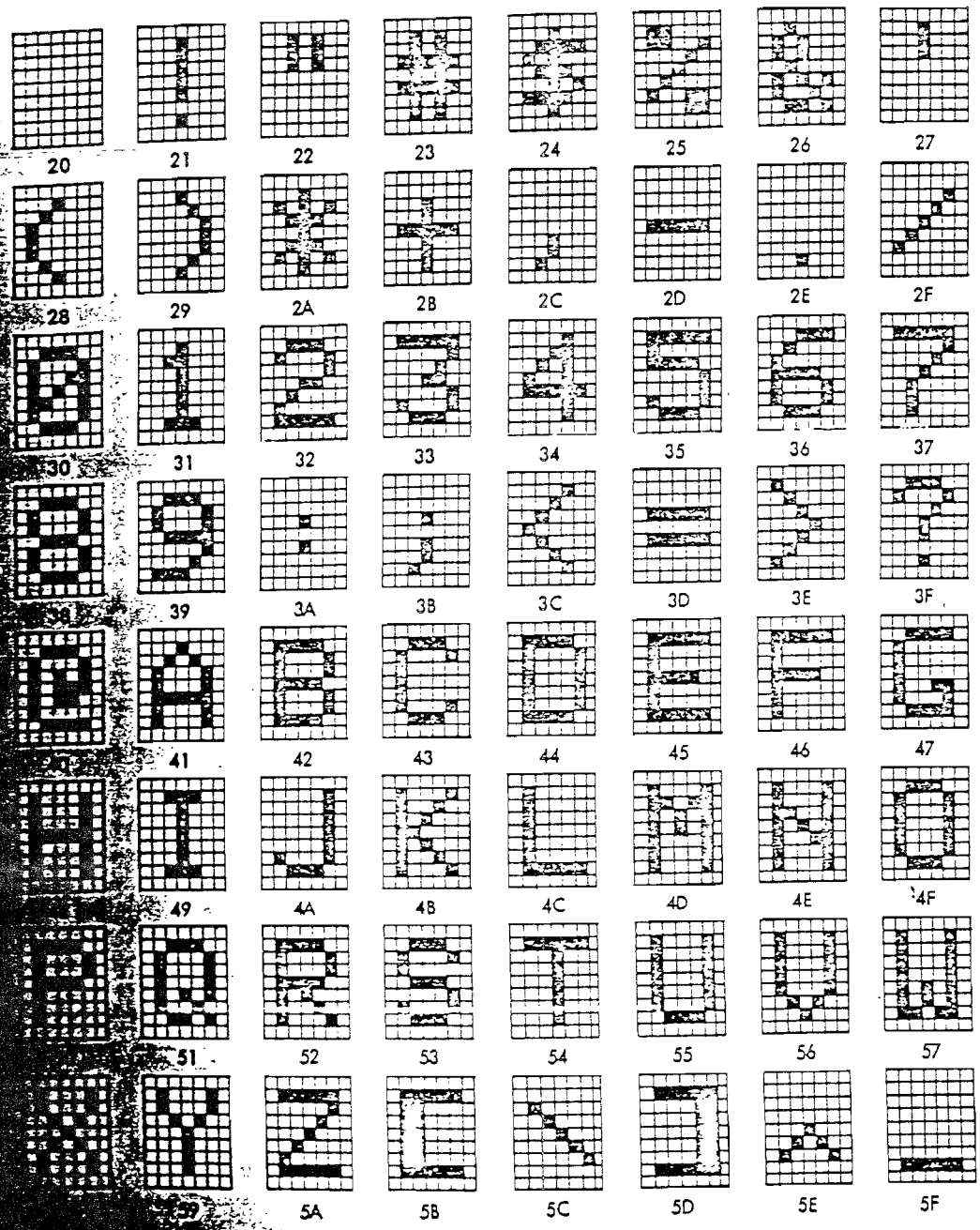


Figure D-1 Standard Text Character Fonts