RDS 3000
PROGRAMMING REFERENCE MANUAL

June 1982

Ikonas Graphics System, Inc.
a subsidiary of
Adage, Inc.

NOTICE

NOTES FOR THIS REVISION

This version of the IKONAS PROGRAMMING REFERENCE MANUAL contains the following additions and changes:

1. The "MA1024 Host Programming Guide" is new. This chapter explains the use of IKONAS-supplied routines (such as MM33) and custom microcode for rotation, clipping assist, and perspective division in the MA1024 Multiplier-Accumulator.

2. The "Video Input Module Programming Guide" is new. This chapter describes how to program the VI8/VI24 video input modules.

3. The chapter entitled "Programming the Framebuffer Memory" has been expanded to include descriptions of DR256, GM64, and GM256 memory.

4. The chapter entitled "Routines for Communicating with the IKONAS Processor" has been expanded to include information about the routines for 32-bit arithmetic available in ALU32.

5. The "MPC Programming Guide" has been substantially revised.

6. The chapters describing hardware installation and diagnostic tests have been deleted from this manual and are now in revised form in the IKONAS INSTALLATION GUIDE.

7. Other chapters contain minor revisions, noted with change bars.

TABLE OF CONTENTS

1.0 INTRODUCTION AND CONVENTIONS

This document describes the IKONAS RDS-3000 display system as seen from the host computer. Other documents describe: the principles of operation of the system (IKONAS User's Guide); how to write microcode for the BPS32 microprocessor (BPS32 Programming Guide); how to write microcode for the MA1024 3-D multiplier unit (MA1024 Programming Guide). These documents are available from IKONAS.

1.1 Conventions used in the document

Numbers appearing in this document are in decimal except as follows: numbers denoting data values within the IKONAS system (bus data values, addresses, function codes, pixel values, etc.) are in octal. Certain addresses and data in the MPC conform to Motorola's usage and are in hexadecimal.

Numbers larger than 16 bits are referred to in one of two ways as appropriate in the context:

- yy$xx for bus addresses which are normally thought of as 14-bit y values and 10-bit x values;
- hh\ll for other data items larger than 16 bits, given as upper 16 bits and lower 16 bits.

For example, 1000$30 is equivalent to 10\30 and the binary string 00000000000010000000000000011000:

```
00000000000010000000000000011000
!<-16-bit hh-->!!<-16-bit ll-->!
        !<14-bit yy->!!10-bit x!
```

2.0 IKONAS BUS ADDRESSING

2.1 Introduction

All transfers from a host computer to any portion of the IKONAS graphics system take place over the IKONAS bus. The host computer interface module can initiate a transfer to or from any device connected to the IKONAS bus. This includes microprogram memory (writeable control store), video look-up memory (color map), static memory, dynamic memory (frame buffer) and device controllers.

The host interface WRITES data TO another device by taking control of the bus (through a priority network), and then placing data, address, and function information on the bus for the required interval. The host interface then relinquishes control.

The host interface READS data FROM another device by taking control of the bus as before, placing address and function information on the bus, and then latching the returned data at the appropriate time before relinquishing control.

The IKONAS system contains a separate 24-bit address bus. The upper 4 bits are used to indicate the memory segment addressed, while the lower 20 bits perform addressing within one-megaword segments. The specific segment designations used are detailed in the following sections. In general, however, the IKONAS addressing scheme allows direct support of almost all conceivable memory organizations.

2.2 Dynamic Memory

Segment 0 through 7 are reserved for dynamic memories. These memories are typically used for frame buffer storage, but may be used as general

purpose memories if desired. For most systems, only segment 0 will be used.

2.3 Summary of IKONAS Bus Addresses

DR64, DR256, GM64, GM256	
0$0 - 17777$1777	Framebuffer Memory
MCM4	
20000$0 - 20077$1777	8K words per MCM4 (maximum of 8)
20100$0 - 20177$1777	Reserved
SR8	
20200$0 - 20277$1777	8K words per SR8 (maximum of 8)
LUVO	
20300$0 - 20300$1777	Colormap memory
20301$0	Channel crossbar switch control register
20301$1 - 20377$1777	Reserved
MA1024	
20400$0 - 20400$1777	Coefficient Memory
20401$0 - 20401$1777	Microprogram Memory (MPM)
20402$0 - 20402$3	MA1024 control registers
20402$7	MA1024 start
20403$0 - 20403$2	Result registers
20403$3	PC readback register
20403$4 - 20477$1777	Reserved
BPS	
20500$0	Status register
20500$1 - 20577$1777	Reserved
CGM4	
20600$0	Base Control Block
20600$1	Address of CGMCB
20600$2 - 20677$1777	Reserved
20700$0 - 27777$1777	Reserved

FBC

30000$0 - 30000$6	Control registers:
30000$0	Viewport location
30000$1	Viewport size
30000$2	Window location
30000$3	Zoom
30000$4	Display rate control
30000$5	Video control
30000$6	Cursor location
30000$10 - 30000$377	Reserved
30000$400 - 30000$777	Programmable cursor
30000$1000 - 30000$1777	Reserved

VIB/VI24

30100$0 - 30100$5	Control registers:
30100$0	Viewport location
30100$1	Minification factor
30100$2	Window location
30100$3	Viewport size
30100$4	Control register
30100$5	Sampling frequency
30100$6 - 30100$1777	Reserved

XBS34

30200$0 - 30200$41	Control registers
30200$42 - 30200$1777	Reserved
30300$0 - 33700$1777	Reserved

MPC

34000$0 - 34037$1777	Data RAM (lower 16 bits only)
34040$0 - 34077$1777	I/O space (lower 16 bits only)

MPC256

34000$0 - 34377$1777	Data RAM (lower 16 bits only)
34400$0 - 34477$1777	I/O space (lower 16 bits only)

3.0 HOST/IKONAS INTERFACE

UNIBUS - TM Digital Equipment Corp.

Note: all numbers in this document are octal except bit numbers, which are decimal.

3.1 PHYSICAL ARRANGEMENT

The IKONAS interface to DEC computers consists of two cards connected by two ribbon cables. One card, the IK11B, is plugged into the UNIBUS; the other card, the IKONAS IF/DMA, plugs into the IKONAS frame.

3.1.1 UNIBUS (IK11B)

The interface requires one quad size board slot in an SPC-compatible backplane. This card contains direct memory address (DMA) logic and control for data transfers to and from the IKONAS graphics system. In addition, logic is included for programmed control of IKONAS bus addressing and operation.

3.1.2 IKONAS (IF/DMA)

One card is plugged into the IKONAS bus to provide interfacing and control to the IKONAS graphics system. A microprogrammed sequencer insures proper timing and protocol for both the UNIBUS and IKONAS systems.

3.2 OPERATION

3.2.1 UNIBUS ADDRESSING AND OPERATION

Eight sequential addresses are decoded to read or load various 16 bit registers for control. Upper address bits are jumper selected for a particular machine configuration. The default board address for the IK11B is 772460; the card is shipped from the factory at this address. In the address assignments below, 'base address' refers to the address jumpered as the lowest address recognized by the card.

The IK11B provides three classes of interrupts-I/O, video field, and processor interrupts-which may be enabled separately. Each kind of interrupt has its own interrupt vector; in all, three interrupt vectors are used. The interrupt vectors occupy contiguous locations in memory and start at an address (the 'base interrupt vector address') determined by jumpers on the card. When the card is delivered from the factory, the base interrupt vector address is 400.

3.2.1.1 UNIBUS CONTROL/STATUS REGISTERS

UNIBUS WORD COUNT (address=base address) - A 16-bit read/write register which should be loaded with the two's complement of the number of 16 bit words to be transferred. Incremented after every UNIBUS DMA cycle.

UNIBUS ADDRESS REGISTER (address=base address+2) - A 16-bit read/write register which should be loaded with the starting address of memory area to be used. The starting address must be even (that is, on a word boundary).

UNIBUS STATUS/COMMAND REGISTER (address=base address+4)- A 16-bit register used to control DMA operation. See section 2.2 for bit definition.

DATA I/O REGISTER (address=base address+6) - Two separate registers, one for input and one for output, used for data transfer during DMA cycles or programmed I/O. Both registers occupy the same address.

LOWER IKONAS ADDRESS REGISTER (address=base address+10) - A 10 bit write-only register which should be loaded with the lower 10 bits of the starting IKONAS bus address. This will be X address for frame buffer organization.

UPPER IKONAS ADDRESS REGISTER (address=base address+12)- A 14 bit write-only register which should be loaded with the upper 14 bits of the starting IKONAS bus address. This will be Y address for frame buffer organization.

IKONAS STATUS/COMMAND REGISTER (address=base address+14) - A 16 bit register used to control IKONAS bus operation. See section 2.3 for bit definition.

3.2.2 UNIBUS STATUS/COMMAND REGISTER BIT DEFINITIONS

bit	function
0	GO - Set to start DMA operation. Clears READY. Reads as zero
3-1	Not used.
5-4	EXTENDED BUS ADDRESS - Determine upper two UNIBUS address bits (17-16) for DMA operation. Read/write.
6	I/O INTERRUPT ENABLE - Set to allow interrupt upon completion of transfer. When the bit is set, an interrupt at the base interrupt vector address will be taken when a DMA or programmed transfer is complete.
7	READY - Set to indicate DMA transfer or programmed I/O sequence complete. Cleared by GO. Read-only.
8	Reserved for future use.
9	PROCESSOR INTERRUPT ENABLE - Set to allow interrupts from the IKONAS BPS32 processor. This interrupt is taken to the base

interrupt vector address+10. When the bit is set to 1 from 0, an interrupt will not be taken until the next interrupt requested by the processor.

10 FIELD INTERRUPT ENABLE - Set to allow interrupts at the field rate. An interrupt to the base interrupt vector address+4 will be taken at the start of each field while this bit is on. When the bit is set to 1 from 0, an interrupt will not be taken until the next field begins.

11 Reserved fur future use.

12 INTERFACE RESET (IK11B versions AB and later) - Set to force a reset of the IK11B. Always reads as 0. When this bit is stored, all other settings in the IK11B are lost, and the interface is reset to its initial state.

13 ATTN-H - set when any interupt is pending.

14 NEX - non-existent memory error

15 ERROR - NEX or address overflow error

3.2.3 IKONAS STATUS/COMMAND REGISTER BIT DEFINITIONS

bit function

3-0 FUNCTION - Function code interpreted by particular IKONAS module addressed.

4 R/W - Set for IKONAS write (UNIBUS to IKONAS data transfer). Cleared for IKONAS read (IKONAS to UNIBUS data transfer). Read/Write.

5 HALFWORD - Set to indicate 16 bit transfers (one UNIBUS word per IKONAS word). Cleared to indicate 32 bit transfers (two UNIBUS words per IKONAS word). Read/Write. When the bit is cleared, two consecutive UNIBUS words are taken together to form one IKONAS 32-bit word. The number of IKONAS bus tranfers is one-half the number of UNIBUS transfers. The lower UNIBUS word corresponds to the lower half (bits 0-15) of the IKONAS word.

When the bit is set, each UNIBUS word corresponds to one IKONAS bus transfer. The UNIBUS word corresponds to the lower half (bits 0-15) of the IKONAS bus word; The upper half (bits 16-31) remains unchanged from previous loading for IKONAS bus write operations, and is ignored for IKONAS bus read operations.

When bit 5 is on, bit 11 (byte mode) must be off.

6 DMA ENABLE - Set to allow DMA transfers. Cleared to allow programmed I/O transfers. Read/Write.

7 INVISIBLE I/O - Set to inhibit transfers except during non-visible picture time (blanking). Cleared to allow continuous I/O. Read/Write.

8 INCREMENT - Set to force increment of IKONAS bus address after each IKONAS word transfer. Cleared to inhibit increment. Read/Write. This bit should always be set.

9 RUN PROCESSOR - Set to allow IKONAS microprocessor to run. Cleared to halt processor. Read/Write. See the "BPS Host Programming Guide" in this manual for details.

10 RESET - Set to reset programmable elements of the IKONAS system (e. g. the BPS32). When this bit is set, program counters in all devices are set to 0. This bit must be cleared to allow normal execution.

11 BYTE MODE - Set to engage byte mode (each 8-bit host byte corresponds to one 32-bit IKONAS word). Reset for word or halfword mode. When bit 11 is on, bit 5 must be off.

12 FRAME INT - This bit reads as 1 during field 0 of a video frame, 0 during field 1. When the bit changes from 0 to 1, a frame interrupt will be signaled if bit 10 of the UNIBUS status register is set. Read-only.

13 PROC INT - This bit reads as 1 when the BPS32 processor is requesting service. When the bit changes from 0 to 1, a processor interrupt will be signaled if bit 9 of the UNIBUS status register is set. Read-only.

15-14 BYTE NUMBER - When bit 11 is set, these two bits select the byte number of the IKONAS word which will be read from. 00=byte 0 (bits 0-7), 01=byte 1 (bits 8-15), 10=byte 2 (bits 16-23), 11=byte 3 (bits 24-31). Set by user for READ operations.

14-12 SENDER ID - This three-bit field specifies the IKONAS bus sender id to be used for write operations. The sender ID is used to specify which write mask is to be written or used by the DR64 and GM64; and to specify which shade register is to be used on the GM64. The sender id field is supported only on systems with a printed-circuit-board IF/IK board. Set by user for WRITE operations.

15 (write-only) Reserved for future use.

3.3 TYPICAL I/O PROGRAMMING

3.3.1 DMA TRANSFERS

```
Set IKONAS COMMAND REGISTER (include DMA bit)
Set UNIBUS WORD COUNT REGISTER
Set UNIBUS ADDRESS REGISTER
Set LOWER IKONAS ADDRESS REGISTER
Set UPPER IKONAS ADDRESS REGISTER
Set UNIBUS COMMAND REGISTER (set GO bit)
- - -  wait for interrupt or check READY bit.
```

3.3.2 PROGRAMMED I/O TRANSFERS

3.3.2.1 READ OPERATIONS

```
Set IKONAS COMMAND REGISTER
    (clear the DMA and WRITE bits)
Set LOWER IKONAS ADDRESS REGISTER
Set UPPER IKONAS ADDRESS REGISTER
Set UNIBUS COMMAND REGISTER (set GO bit)
- - -  wait for interrupt or check READY bit.
READ DATA I/O REGISTER
```

Note: if the HALFWORD bit is off, you must read from the I/O register twice to get all 32 bits.

3.3.2.2 WRITE OPERATIONS

```
Set IKONAS COMMAND REGISTER
    (clear DMA bit, set WRITE bit)
Set LOWER IKONAS ADDRESS REGISTER
Set UPPER IKONAS ADDRESS REGISTER
Set UNIBUS COMMAND REGISTER (set GO bit)
WRITE DATA I/O REGISTER
```

Note: if the HALFWORD bit is off, you must write to the I/O register twice to perform the write operation.

4.0 PROGRAMMING THE IKONAS FRAMEBUFFER MEMORY

NOTE: within this chapter, all numbers and addresses are DECIMAL unless otherwise noted.

4.1 Introduction

The heart of any IKONAS system is the framebuffer memory. There are four different types of memory boards:

- o DR64
- o DR256
- o GM64
- o GM256

The following sections describe each type of memory board in detail.

4.2 DR64

An IKONAS system consisting of DR64 memory cards can be viewed in two different ways, at the option of user programming:

- o pixel mode
- o word mode

Having these different ways of looking at the same memory gives great flexibility in the use of the IKONAS framebuffer, but it also requires the user to educate himself in the operation of the DR64. In particular, a user must know:

- o how to read and write pixels in LORES and HIRES modes;
- o how to use WORD mode;

- what the visible effects of WORD-mode operations are;
- how to use the write masks to write only desired bits.

In addition, a user must know how his IKONAS frame-buffer memory is configured, and how to use that configuration.

4.2.1 Pixel Mode

In pixel mode, all DR64's can be viewed as one continuous memory that is addressable in x and y. Each x,y memory location corresponds to a pixel. The bit depth of each memory location -- the number of bits which contribute to each pixel -- depends on the system configuration, as follows:

In low resolution (LORES) mode, each DR64 can provide 262,144 pixels of eight bits each, corresponding to a 512x512 display. Depending on the number of DR64's in the system configuration, the bit depth may be 8, 16, 24, or 32 bits. X addresses range from 0 to 511. The y address range depends on the number of DR64's and the bit depth; see Appendix 1 of this chapter, which describes standard DR64 configurations, for bit depth and y address limits.

In high resolution (HIRES) mode, each DR64 can provide 1,048,576 (1024K) pixels of 2 bits each, corresponding to a 1024 x 1024 display. Depending on the number of DR64's in the system configuration, the bit depth may be any multiple of 2 from 2-24 bits. X addresses range from 0 to 1023. The y address range depends on the number of DR64's and the bit depth; see Appendix 1.

NOTE: in LORES mode, these x and y coordinates must be multiplied by two.

4.2.1.1 Examples of Configuration

Here are some example IKONAS memory organizations:

1. (Minimum system) The user has one DR64. It has a bit depth of 8 bits, x address range 0-511, y address range 0-511 (LORES mode).

2. (Full-color 512x512 system) The user has three DR64's. In LORES: bit depth is 24 bits. Card 0 supplies bits 0-7 (red), card 1 supplies bits 8-15 (green), and card 2 supplies bits 16-23 (blue). The x address range is 0-511, y address range 0-511.

In HIRES: bit depth is 6 bits. Card 0 supplies bits 0-1, card 1 supplies bits 2-3, and card 2 supplies bits 4-5, so the user can use pseudocolor on a 6-bit HIRES image. The x address range is 0-1023; the y address range is 0-1023.

3. (Large system) The user has twelve DR64's. In LORES: the bit depth is 24 bits. The x address range is 0-511; the y address range is 0-2047.

In HIRES: the bit depth is 24 bits, with each card contributing 2 bits, for a full-color HIRES image. The x address range is 0-1023; the y address range, 0-1023.

4.2.2 Pixel-mode Addressing of the Framebuffer

In pixel mode (LORES or HIRES), each pixel is addressed independently of all the others; however, several DR64's may respond to the same pixel address, depending on the bit depth. (See the preceding section on configuration.)

When a program WRITES to a pixel address, any bits which are not configured into any DR64 are ignored.

When a program READS from a pixel address, any bits which are not configured to any DR64 are read as zero, but displayed as 1.

4.2.2.1 Pixel I/O

To write a pixel, you must do the following:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. See below for a discussion of the write mask.

2. LOAD THE PIXEL ADDRESS INTO THE IKONAS INTERFACE. The IKONAS interface contains two address registers, corresponding to x-address and y-address, so you can set the interface address registers with (x,y) coordinates.

However, for LORES mode only, you must load the x and y registers with twice the coordinate desired. For example, since the x or y address is a number between 0 and 511, you will load the interface address registers with an even number between 0 and 1022. If your y address range is from 0-1023, you will load the y-address register with an even number between 0 and 2046.

These registers correspond to a 24-bit IKONAS bus address. In HIRES, bits 10-19 contain the y address; bits 0-9, the x address. In LORES, bits 11-19 contain the y address; bits 1-9, the x address.

3. SET THE FUNCTION CODE:

22 octal for LORES
23 octal for HIRES

and the desired SENDER ID into the IKONAS interface command register. The sender id selects the write mask you will use (see below).

4. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed I/O or DMA); refer to the chapter "HOST/IKONAS INTERFACE" in this manual for details.

TO READ PIXELS, follow the same steps, with these exceptions:

1. you do not need to load the write mask;

2. use a function code of

02 octal for LORES
03 octal for HIRES

Again, see the chapter concerning the interface specification for details; in particular, for byte mode transfers, you must set the "byte mode" bit in the IKONAS status/command register.

4.2.2.2 Write Mask

The write mask is a set of eight registers, one for each sender id, which allows you to write only selected bits of a pixel when you write to the framebuffer. The write mask is used for all writes to the framebuffer, but it does not apply to any other component of the IKONAS system (for example, the LUVO). Each bit of the write mask controls a single bit of pixels written to the framebuffer. When a bit of the write mask is on (set to 1), the corresponding bit of each pixel in the framebuffer may be modified by writes to the framebuffer. When a bit if the write mask is off, however, the corresponding bit of each pixel in the framebuffer is protected, and will not be modified by writes to the framebuffer, even though other bits in the same pixel may be modified.

The sender id is part of each IKONAS bus write operation. For the DR64, it specifies which write mask is to be set or used. Having multiple write masks allows you to have different bitplanes open to modification by different processes -- for example, character graphics planes to be modified by the CGM4 character generator, image planes to be modified by the BPS32 processor, and background planes to be modified by the host. The FBC always uses sender id 0 for its display-and-erase function, so you may have different write masks for erasing and writing memory.

The write mask must have been set before you issue a write, or the results are unpredictable; but once you set the write mask, it remains unchanged until you set it again.

There is only one write mask per sender id, but how it is interpreted depends on the function code used to set it. There are two 'set-write-mask' IKONAS bus function codes, one for LORES and one for HIRES. Normally, you should use the set-write-mask function corresponding to the type of write you wish to control.

TO SET THE WRITE MASK, perform all the steps for writing a single pixel to address (0,0), but use a function code of

32 octal for LORES
33 octal for HIRES

The data written will be used as the write mask for subsequent write operations.

In LORES mode, there is a separate set of write masks for every 512 lines in memory; the y address distinguishes between these write masks. For example, if your system has 1024 lines, you would need one write mask for address (0,0) and another for address (0,512), bearing in mind that LORES mode requires you to specify twice the coordinates desired.

In HIRES mode, there is one write mask for every 1024 lines (usually the maximum in a system).

In wire-wrapped DR64's, there is only one write mask, shared by all sender ids.

4.2.3 WORD Mode

In WORD mode, each DR64 can provide 65,536 words of 32 bits each. An IKONAS system may have up to 12 DR64 cards; each memory location can be addressed with a card number and an offset. This is identical to traditional random-access memory organization.

To determine the card number for word mode addresses in your system, refer to the system configuration sheet.

4.2.3.1 WORD Mode I/O

To write a 32-bit word in WORD mode:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. You must load the write mask in LORES or HIRES mode, depending on your display mode; see above for instructions on setting the write mask in LORES or HIRES mode.

2. LOAD THE WORD ADDRESS INTO THE IKONAS INTERFACE. In WORD mode the 'x-address' and 'y-address' registers lose their physical significance as pixel coordinates and merely hold 24 address bits. The IKONAS bus address to use is a 24-bit quantity; the x-address register holds bits 0-9, and the y-address register holds bits 10-23. Bits 16-19 contain the card number, while bits 0-15 select a particular word of memory within the DR64. Bits 20-23 must be zero for word-mode accesses to a DR64.

3. SET THE FUNCTION CODE

20 octal (for word mode)

and the SENDER ID into the IKONAS interface command register. The sender id determines which write mask is used for the write operation.

4. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed or DMA); refer to the chapter "Host/IKONAS Interface" in this manual for details.

TO READ 32-BIT WORDS, follow the same steps, with these exceptions:

1. you do not need to load the write mask;

2. use a function code of 00 octal instead of 20 octal.

4.2.3.2 Effects of the Write Mask in Word Mode

When the write mask is set to protect certain bit-planes in LORES or HIRES mode, those bitplanes will be protected against WORD-mode writes also. If you want to allow modifications to all bitplanes, write all ones (37777777777 octal) to the write mask in LORES or HIRES mode.

If you ever need to modify the write mask on exactly one DR64, you may do so by writing to any WORD-mode address on the DR64 using a function code of 30 octal. The low-order four bits (bits 0-3) will be used as bits 0-3 and 4-7 of the write mask for that card. Normally this will not be necessary.

Since the WORD-mode operations refer to the same memory as the pixel-mode operations, although using a different addressing scheme, it is necessary for the programmer to understand which pixels are modified by WORD-mode writes.

4.2.3.3 The reason for using WORD mode

WORD mode gives you a quick way to modify several different pixels at the same time, even though you modify only a few bits (one in HIRES mode, four in LORES) of each pixel. WORD mode will be more efficient than pixel mode in the following cases:

1. You have fewer than 32 bits per pixel. Pixel-mode transfers are 32 bits wide; so you have unused bits in each pixel-mode transfer. WORD mode will give you higher transfer rate for loading large files.

2. Your pixels are organized as separate fields of bits, presumably organized using a colormap. In this case, you may frequently want to access portions of pixels (i. e. only the bits in a field). You can do this using the write mask; but if you have a large number of consecutive pixels in which you want to access only a portion, word mode may be faster.

The correspondence of WORD-mode access to pixel-mode access depends on whether the pixel-mode access is LORES or HIRES.

4.2.3.4 Correspondence of WORD access to HIRES access

A WORD operation on a DR64 affects 32 bits of the DR64. Those 32 bits are taken from 32 consecutive HIRES pixels. The pixels affected may be determined from the WORD address as follows:

If the WORD address is OOOOccccpyyyyyyyyyyxxxxx

where

cccc (address bits 16-19) is the WORD-mode card address, used to select a particular card;

p (address bit 15) tells which bitplane is referenced by the operation;

yyyyyyyyyy (address bits 5-14) is the y-address being referenced;

xxxxx (address bits 0-4) is the x-address being referenced;

THEN the pixels modified will be the 32 pixels at y-address yyyyyyyyyy and x-address xxxxx00000 through xxxxx11111, where the x's and y's are construed as binary numbers.

The bit numbers within each pixel which are modified by the operation depend on the configuration of the memory card. Recall that each memory card furnishes two bits of HIRES data per pixel; The bit referenced by the WORD operation is the lower of these bits if p (address bit 15) is off, or the higher bit, if p is on.

WORD-mode writes to a DR64 are controlled by the write mask; if the write-mask bit corresponding to the write is off, no modification will be made to the contents of memory. The write mask should have been set by a HIRES set-write-mask operation. The bits of the write mask used on each DR64 are the same as the pixel data bits used by that DR64 and are determined by the configuration of the DR64.

4.2.3.5 Correspondence of WORD access to LORES access

A WORD operation on a DR64 affects 32 bits of the DR64. Those 32 bits are taken from 8 consecutive LORES pixels. The pixels affected may be determined from the WORD address as follows:

If the WORD address is 0000ccccpyyyyyyyyyxxxxxx

where

cccc (address bits 16-19) is the WORD-mode card address, used to select a particular card;

p (address bit 15) determines which four-bit section is referenced by the operation;

yyyyyyyyy (address bits 6-14) are the y-address being referenced;

xxxxxx (address bits 0-5) are the x-address being referenced;

THEN the pixels accessed will be the 8 pixels at y-address yyyyyyyyy and x-address xxxxxx000 through xxxxxx111, where the x's and y's are construed as binary numbers.

The bit numbers within each pixel which are modified by the operation depend on the configuration of the memory card. Recall that each memory card furnishes eight bits of LORES data per pixel; The bits referenced by the WORD operation are the four least-significant of these bits if p (address bit 15) is off, or the four most-significant bits, if p is on.

Four bits will be accessed in each of the eight pixels addressed as described above. In particular, the first pixel corresponds to bits 0-3 of the IKONAS bus data word; the second corresponds to bits 4-7; and so on. Within each pixel, the least significant bit of the pixel corresponds to least significant of the corresponding IKONAS bus data bits.

WORD-mode writes to a DR64 are controlled by the write mask; if the write-mask bit corresponding to the write is off, no modification will be made to the contents of memory. The write mask should have been set by a HIRES set-write-mask operation. The bits of the write mask used on each DR64 are the same as the

pixel data bits used by that DR64, and are determined by the configuration of the DR64.

When the write mask is used to protect bitplanes of a DR64, the correspondence of input IKONAS bus data bits to pixels in the DR64 is unchanged. For example, suppose the write mask is set to protect all bitplanes except bitplane 1 (i. e. the write mask was set to 02 octal). Then any write will not change bitplanes 0, 2, 3, 4, 5, 6, or 7; only bitplane 1 may be affected; so only bits 1, 5, 9, 13, 17, 21, 25, and 29 of the input word will be written; these will be written into bitplane 1 of eight consecutive pixels (provided bit 15=0 in the DR64 address).

Graphically, the bits marked with an x will not be written, while the bits marked with a 1 will be written:

```
xx1xxx1xxx1xxx1xxx1xxx1xxx1xxx1xxx1x
```

If the write mask allowed modification to bitplanes 1 and 2, the bits to be written would be:

```
x11xx11xx11xx11xx11xx11xx11xx11xx11x
```

4.3 DR256

An IKONAS system consisting of DR256 memory cards can be viewed in two different ways, at the option of user programming:

- o pixel mode
- o word mode

4.3.1 Pixel Mode

In pixel mode, all DR256's can be viewed as one continuous memory that is addressable in x and y. Each x,y memory location corresponds to a pixel. The bit depth of each memory location -- the number of bits which contribute to each pixel -- depends on the system configuration, as follows:

In low resolution (LORES) mode, each DR256 can provide 1,048,576 pixels of eight bits each, corresponding to a 1024x1024 pixel display. Depending on the number of DR256's in the system configuration, the bit depth may be 8, 16, 24, or 32 bits. X addresses range from 0 to 1023. The y address range depends on the number of DR256's and the bit depth. Appendix 2 contains information about bit depth and address ranges in standard DR256 configurations.

In high resolution (HIRES) mode, each DR256 can provide 4,194,304 pixels of 2 bits each, corresponding to a 2048x2048 pixel display. Depending on the number of DR256's in the system configuration, the bit depth may be any multiple of 2 from 2-24 bits. X addresses range from 0 to 2047. The y address range depends on the number of DR256's and the bit depth; see Appendix 2.

4.3.1.1 Examples of Configuration

Here are some example IKONAS memory organizations:

1. (Minimum system) The user has one DR256. It has a bit depth of 8 bits, x address range 0-1023, y address range 0-1023 (LORES mode).

2. (Full-color 1024x1024 system) The user has three DR256's. In LORES: bit depth is 24 bits. Card 0 supplies bits 0-7 (red), card 1 supplies bits 8-15 (green), and card 2 supplies bits 16-23 (blue). The x address range is 0-1023, y address range 0-1023.

In HIRES: bit depth is 6 bits. Card 0 supplies bits 0-1, card 1 supplies bits 2-3, and card 2 supplies bits 4-5, so the user can use pseudocolor on a 6-bit HIRES image. The x address range is 0-2047; the y address range is 0-2047.

3. (Large system) The user has twelve DR256's. In LORES: the bit depth is 24 bits. The x address range is 0-1023; the y address range is 0-2047. (Note that in LORES, you cannot use all twelve cards.)

In HIRES: the bit depth is 24 bits, with each card contributing 2 bits, for a full-color HIRES image. The x address range is 0-2047; the y address range, 0-2047.

NOTE: in LORES mode, you must multiply the x and y coordinates by two.

4.3.2 Pixel-mode Addressing of the Framebuffer

In pixel mode (LORES or HIRES), each pixel is addressed independently of all the others; however, several DR256's may respond to the same pixel address, depending on the bit depth. (See the preceding section on configuration.)

When a program WRITES to a pixel address, any bits which are not configured into any DR256 are ignored.

When a program READS from a pixel address, any bits which are not configured to any DR256 are read as zero, BUT DISPLAYED AS 1.

4.3.2.1 Pixel I/O

To write a pixel, you must do the following:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. See below for a discussion of the write mask.

2. LOAD THE PIXEL ADDRESS INTO THE IKONAS INTERFACE. The IKONAS interface contains two address registers, corresponding to x-address and y-address, so you can set the interface address registers with (x,y) coordinates.

However, for LORES mode only, you must load the x and y registers with twice the coordinate desired. For example, since the x or y address is a number between 0 and 1023, you will load the interface address registers with an even number between 0 and 2046. If your y address range is from 0-2047, you will load the y-address register with an even number between 0 and 4094.

These registers correspond to a 24-bit IKONAS bus address. In HIRES, bits 10-20 contain the y address; bits 21 and 0-9, the x address. In LORES, bits 11-20 contain the y address; bits 21 and 1-9, the x address. (Bit 21 holds the high-order x address bit in each case.)

3. SET THE FUNCTION CODE:

22 octal for LORES
23 octal for HIRES

and the desired SENDER ID into the IKONAS interface command register. The sender id selects the write mask you will use (see below).

4. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed I/O or DMA); refer to the chapter "Host/IKONAS Interface" in this manual for details.

TO READ PIXELS, follow the same steps, with these exceptions:

1. you do not need to load the write mask;

2. use a function code of

02 octal for LORES
03 octal for HIRES

4.3.2.2 Write Mask

The write mask is a set of eight registers, one for each sender id, which allows you to write only selected bits of a pixel when you write to the framebuffer. The write mask is used for all writes to the framebuffer, but it does not apply to any other component of the IKONAS system (for example, the LUVO). Each bit of the write mask controls a single bit of the pixels written to the framebuffer. When a bit of the write mask is on (set to 1), the corresponding bit of each pixel in the framebuffer may be modified by writes to the framebuffer. When a bit if the write mask is off, however, the corresponding bit of each pixel in the framebuffer is protected, and will not be modified by writes to the framebuffer, even though other bits in the same pixel may be modified.

The sender id is part of each IKONAS bus write operation. For the DR256, it specifies which write mask is to be set or used. Having multiple write masks allows you to have different bitplanes open to modification by different processes, for example, character graphics planes to be modified by the CGM4 character generator, image planes to be modified by the BPS32 processor, and background planes to be modified by the host. The FBC always uses sender id 0 for its display-and-erase function, so you may have different write masks for erasing and writing memory.

The write mask must have been set before you issue a write, or the results are unpredictable; but once you set the write mask, it remains unchanged until you set it again.

There is only one write mask per sender id, but how it is interpreted depends on the function code used to set it. There are two 'set-write-mask' IKONAS bus function codes, one for LORES and one for HIRES. Normally, you should use the set-write-mask function corresponding to the type of write you wish to control.

TO SET THE WRITE MASK, perform all the steps for writing a single pixel to address (0,0), but use a function code of

32 octal for LORES

33 octal for HIRES

The data written will be used as the write mask for subsequent write operations.

In LORES mode, there is a separate set of write masks for every 1024 lines in memory; the y address distinguishes between these write masks. For example, if your system has 2048 lines, you would need one write mask for address (0,0) and another for address (0,1024), bearing in mind that LORES mode requires you to specify twice the coordinates desired.

In HIRES mode, there is one write mask for every 2048 lines (usually the maximum in a system).

In wire-wrapped DR256's, there is only one write mask, shared by all sender ids.

4.3.3 WORD Mode

In WORD mode, each DR256 can provide 262,144 words of 32 bits each. An IKONAS system may have up to 12 DR256 cards; each memory location can be addressed with a card number and an offset. This is identical to traditional random-access memory organization.

To determine the card number for word mode addresses in your system, refer to Appendix 2.

4.3.3.1 WORD Mode I/O

To write a 32-bit word in WORD mode:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. You must load the write mask in LORES or HIRES mode, depending on your display mode; see above for instructions on setting the write mask in LORES or HIRES mode.

2. LOAD THE WORD ADDRESS INTO THE IKONAS INTERFACE. In WORD mode the 'x-address' and 'y-address' registers lose their physical significance as pixel coordinates and merely hold 24 address bits. The IKONAS bus address to use is a 24-bit quantity; the x-address register holds bits 0-9, and the y-address register holds bits 10-23.

3. SET THE FUNCTION CODE

    20 octal for word mode

and the SENDER ID into the IKONAS interface command register. The sender id determines which write mask is used for the write operation.

4. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed or DMA); refer to the chapter "Host/IKONAS Interface" in this manual for details.

TO READ 32-BIT WORDS, follow the same steps, with these exceptions:

1. you do not need to load the write mask;

2. use a function code of 00 octal instead of 20 octal.

4.3.3.2 Effects of the Write Mask in Word Mode

When the write mask is set to protect certain bitplanes in LORES or HIRES mode, those bitplanes will be protected against WORD-mode writes also. If you want to allow modifications to all bitplanes, write all ones (37777777777 octal) to the write mask in LORES or HIRES mode.

If you ever need to modify the write mask on exactly one DR256, you may do so by writing to any WORD-mode address on the DR256 using a function code of 30 octal. The low-order four bits (bits 0-3) will be used as bits 0-3 and 4-7 of the write mask for that card. Normally this will not be necessary.

Since the WORD-mode operations refer to the same memory as the pixel-mode operations, although using a different addressing scheme, it is necessary for the programmer to understand which pixels are modified by WORD-mode writes.

4.3.3.3 The reason for using WORD mode

WORD mode gives you a quick way to modify several different pixels at the same time, even though you modify only a few bits (one in HIRES mode, four in LORES) of each pixel. WORD mode will be more efficient than pixel mode in the following cases:

1. You have fewer than 32 bits per pixel. Pixel-mode transfers are 32 bits wide; so you have unused bits in each pixel-mode transfer. WORD mode will give you

higher transfer rate for loading large files.

2. Your pixels are organized as separate fields of bits, presumably organized using a colormap. In this case, you may frequently want to access portions of pixels (i. e. only the bits in a field). You can do this using the write mask; but if you have a large number of consecutive pixels in which you want to access only a portion, word mode may be faster.

The correspondence of WORD-mode access to pixel-mode access depends on whether the pixel-mode access is LORES or HIRES.

4.3.3.4 Correspondence of WORD access to HIRES access

A WORD operation on a DR256 affects 32 bits of the DR256. Those 32 bits are taken from 32 consecutive HIRES pixels. The pixels affected may be determined from the WORD address as follows:

If the WORD address is OOccccxypyyyyyyyyyyxxxxx

where

cccc (address bits 18-21) is the WORD-mode card address, used to select a particular card;

p (address bit 15) tells which bitplane is referenced by the operation;

yyyyyyyyyyy (address bits 5-14 and 16) is the y-address being referenced (bit 16 is the high-order y bit);

xxxxxx (address bits 0-4 and 17) is the x-address being referenced (bit 17 is the high-order x bit);

THEN the pixels modified will be the 32 pixels at y-address yyyyyyyyyyy and x-address xxxxxx00000 through xxxxxx11111, where the x's and y's are construed as binary numbers.

The bit numbers within each pixel which are modified by the operation depend on the configuration of the memory card. Recall that each memory card furnishes two bits of HIRES data per pixel; The bit referenced by the WORD operation is the lower of these bits if p (address bit 15) is off, or the higher bit, if p is on.

WORD-mode writes to a DR256 are controlled by the write mask; if the write-mask bit corresponding to the write is off, no modification will be made to the contents of memory. The write mask should have been set by a HIRES set-write-mask operation. The bits of the write mask used on each DR256 are the same as the pixel data bits used by that DR256 and are determined by the configuration of the DR256.

4.3.3.5 Correspondence of WORD access to LORES access

A WORD operation on a DR256 affects 32 bits of the DR256. Those 32 bits are taken from 8 consecutive LORES pixels. The pixels affected may be determined from the WORD address as follows:

If the WORD address is OOccccxypyyyyyyyyyyxxxxxx

where

cccc (address bits 18-21) is the WORD-mode card address, used to select a particular card;

p (address bit 15) determines which four-bit section is referenced by the operation;

yyyyyyyyyyy (address bits 6-14 and 16) is the y-address being referenced (bit 16 is the high-order y bit);

xxxxxxx (address bits 0-5 and 17) is the x-address being referenced (bit 17 is the high-order x bit);

THEN the pixels accessed will be the 8 pixels at y-address yyyyyyyyyy and x-address xxxxxxx000 through xxxxxx111, where the x's and y's are construed as binary numbers.

The bit numbers within each pixel which are modified by the operation depend on the configuration of the memory card. Recall that each memory card furnishes eight bits of LORES data per pixel; The bits referenced by the WORD operation are the four least-significant of these bits if p (address bit 15) is off, or the four most-significant bits, if p is on.

Four bits will be accessed in each of the eight pixels addressed as described above. In particular, the first pixel corresponds to bits 0-3 of the IKONAS bus data word; the second corresponds to bits 4-7; and so on. Within each pixel, the least significant bit of the pixel corresponds to least significant of the corresponding IKONAS bus data bits.

WORD-mode writes to a DR256 are controlled by the write mask; if the write-mask bit corresponding to the write is off, no modification will be made to the contents of memory. The write mask should have been

set by a HIRES set-write-mask operation. The bits of the write mask used on each DR256 are the same as the pixel data bits used by that DR256, and are determined by the configuration of the DR256.

When the write mask is used to protect bitplanes of a DR256, the correspondence of input IKONAS bus data bits to pixels in the DR256 is unchanged. For example, suppose the write mask is set to protect all bitplanes except bitplane 1 (i. e. the write mask was set to 02 octal). Then any write will not change bitplanes 0, 2, 3, 4, 5, 6, or 7; only bitplane 1 may be affected; so only bits 1, 5, 9, 13, 17, 21, 25, and 29 of the input word will be written; these will be written into bitplane 1 of eight consecutive pixels (provided bit 15=0 in the DR256 address).

Graphically, the bits marked with an x will not be written, while the bits marked with a 1 will be written:

```
xx1xxx1xxx1xxx1xxx1xxx1xxx1xxx1xxx1x
```

If the write mask allowed modification to bitplanes 1 and 2, the bits to be written would be:

```
x11xx11xx11xx11xx11xx11xx11xx11xx11x
```

4.4 GM64

An IKONAS system consisting of GM64 memory cards allows two different kinds of write operations:

- o pixel mode
- o mask mode

This section explains:

- o how to read and write pixels in LORES and HIRES modes;
- o how to use mask mode;
- o what the visible effects of mask-mode operations are;
- o how to use the write masks to write only desired bits.

This section also provides a description of IKONAS framebuffer memory configurations.

4.4.1 Pixel Mode

In pixel mode, GM64's function in the same way as DR64's. All GM64's can be viewed as one continuous memory that is addressable in x and y. Each x,y memory location corresponds to a pixel. The bit depth of each memory location -- the number of bits which contribute to each pixel -- depends on the system configuration, as follows:

In low resolution (LORES) mode, each GM64 can provide 262,144 pixels of eight bits each, corresponding to a 512x512 display. Depending on the number of GM64's in the system configuration, the bit depth may be 8, 16, 24, or 32 bits. X addresses range from 0 to 511. The y address range depends on the number of GM64's and the bit depth. Appendix 3 contains information about address ranges and bit depth for standard GM64 configurations.

In high resolution (HIRES) mode, each GM64 can provide 1,048,576 (1024K) pixels of 2 bits each, corresponding to a 1024 x 1024 display. Depending on the number of GM64's in the system configuration, the bit depth may be any multiple of 2 from 2-24 bits. X addresses range from 0 to 1023. The y address range depends on the number of GM64's and the bit depth; see Appendix 3.

4.4.1.1 Examples of Configuration

Here are some example IKONAS memory organizations:

1. (Minimum system) The user has one GM64. It has a bit depth of 8 bits, x address range 0-511, y address range 0-511 (LORES mode).

2. (Full-color 512x512 system) The user has three GM64's. In LORES: bit depth is 24 bits. Card 0 supplies bits 0-7 (red), card 1 supplies bits 8-15 (green), and card 2 supplies bits 16-23 (blue). The x address range is 0-511, y address range 0-511.

In HIRES: bit depth is 6 bits. Card 0 supplies bits 0-1, card 1 supplies bits 2-3, and card 2 supplies bits 4-5, so the user can use pseudocolor on a 6-bit HIRES image. The x address range is 0-1023; the y address range is 0-1023.

3. (Large system) The user has twelve GM64's. In LORES: the bit depth is 24 bits. The x address range is 0-511; the y address range is 0-2047.

In HIRES: the bit depth is 24 bits, with each card contributing 2 bits, for a full-color HIRES image. The x address range is 0-1023; the y address range, 0-1023.

NOTE: in LORES mode, these x and y coordinates must be multiplied by two.

4.4.2 Pixel-mode Addressing of the Framebuffer

In pixel mode (LORES or HIRES), each pixel is addressed independently of all the others; however, several GM64's may respond to the same pixel address, depending on the bit depth. (See the preceding section on configuration.)

When a program WRITES to a pixel address, any bits which are not configured into any GM64 are ignored.

When a program READS from a pixel address, any bits which are not configured to any GM64 are read as zero, but displayed as 1.

4.4.2.1 Pixel I/O

To write a pixel, you must do the following:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. See below for a discussion of the write mask.

2. LOAD THE PIXEL ADDRESS INTO THE IKONAS INTERFACE. The IKONAS interface contains two address registers, corresponding to x-address and y-address, so you can set the interface address registers with (x,y) coordinates.

However, for LORES mode only, you must load the x and y registers with twice the coordinate desired. For example, since the x or y address is a number between 0 and 511, you will load the interface address registers with an even number between 0 and 1022. If your y address range is from 0-1023, you will load the y-address register with an even number between 0 and 2046.

These registers correspond to a 24-bit IKONAS bus address. In HIRES, bits 10-19 contain the y address; bits 0-9, the x address. In LORES, bits 11-19 contain the y address; bits 1-9, the x address.

3. SET THE FUNCTION CODE:

22 octal for LORES
23 octal for HIRES

and the desired sender id into the IKONAS interface command register. The sender id selects the write mask you will use (see below).

4. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed I/O or DMA); refer to the chapter "HOST/IKONAS INTERFACE" in this manual for details.

TO READ PIXELS, follow the same steps, with these exceptions:

1. you do not need to load the write mask;

2. use a function code of

   02 octal for LORES
   03 octal for HIRES

Again, see the chapter concerning the interface specification for details; in particular, for byte mode transfers, you must set the "byte mode" bit in the IKONAS status/command register.

4.4.2.2 Write Mask

The write mask is a set of eight registers, one for each sender id, which allows you to write only selected bits of a pixel when you write to the framebuffer. The write mask is used for all writes to the framebuffer, but it does not apply to any other component of the IKONAS system (for example, the LUVO). Each bit of the write mask controls a single bit of pixels written to the framebuffer. When a bit of the write mask is on (set to 1), the corresponding bit of each pixel in the framebuffer may be modified by writes to the framebuffer. When a bit if the write mask is off, however, the corresponding bit of each pixel in the framebuffer is protected, and will not be modified by writes to the framebuffer, even though other bits in the same pixel may be modified.

The sender id is part of each IKONAS bus write operation. For the GM64, it specifies which write mask is to be set or used. Having multiple write masks allows you to have different bitplanes open to modification by different processes, for example, character graphics planes to be modified by the CGM4 character generator, image planes to be modified by the BPS32 processor, and background planes to be modified by the host. The FBC always uses sender id 0 for its display-and-erase function, so you may have different write masks for erasing and writing memory.

The write mask must have been set before you issue a write, or the results are unpredictable; but once you set the write mask, it remains unchanged until you set it again.

There is only one write mask per sender id, but how it is interpreted depends on the function code used to set it. There are two 'set-write-mask' IKONAS bus function codes, one for LORES and one for HIRES. Normally, you should use the set-write-mask function corresponding to the type of write you wish to control.

TO SET THE WRITE MASK, perform all the steps for writing a single pixel to address (0,0), but use a function code of

32 octal for LORES
33 octal for HIRES

The data written will be used as the write mask for subsequent write operations.

In LORES mode, there is a separate set of write masks for every 512 lines in memory; the y address distinguishes between these write masks. For example, if your system has 1024 lines, you would need one write mask for address (0,0) and another for address (0,512), bearing in mind that LORES mode requires you to specify twice the coordinates desired.

In HIRES mode, there is one write mask for every 1024 lines (usually the maximum in a system).

In wire-wrapped GM64's, there is only one write mask, shared by all sender ids.

4.4.2.3 Z-buffer

The Z-buffer board, when present, allows you to do a conditional pixel write to certain bitplanes, which have been configured by wire-wrap jumpers as z-planes. When you perform a conditional write, the value to be written to these bitplanes is compared against their current value. The values are treated as unsigned integers during the comparison.

If the old value is less than the new value, the write is suppressed and the bitplanes remain unchanged.

If the old value is greater than or equal to the new value, the write is performed. Function codes to use are:

26 for LORES

27 for HIRES

4.4.3 Mask Mode

Mask mode is a feature unique to the GM64 and GM256 which allows you to write up to 32 pixels in HIRES, or up to 16 pixels in LORES, with one operation. Each of the 32 data bits (for LORES, the lower 16 bits) corresponds to a pixel. If the bit has a value of 1, the pixel remains unchanged; if it has a value of 0, the pixel is written with the shade previously set in the shade register.

While WORD-mode writes in a DR64 modify only a single card, all GM64 cards respond to a mask-mode write operation.

4.4.3.1 Shade register

The shade register holds the value to be written to each pixel selected in a mask mode write. There are eight shade registers, one for each sender id. Each mask mode write sets a sender id which selects both the shade register and the write mask for that write. Having multiple sender id's allows you to have different bitplanes open to modification by different processes, and to have different shade values from which to select.

The shade register must have been set before a write is issued, or the results are unpredictable. However, once you set the shade register, it remains unchanged until you set it again.

There is only one shade register for each sender id, but the function code to set it must correspond to the mask mode write you plan to do -- HIRES or LORES. To set the shade register, perform all the steps for writing a pixel to address (0,0), but use a function code of

36 octal for LORES

37 octal for HIRES

The data written will be used as the shade value for all subsequent mask mode write operations using this sender id.

In LORES mode, there is a separate set of shade registers for every 512 lines in memory; the y address distinguishes between these write masks. For example, if your system has 1024 lines, you would need one shade register for address (0,0) and another for address (0,512), bearing in mind that LORES mode requires you to specify twice the coordinates desired.

In HIRES mode, there is one shade register for every 1024 lines (usually the maximum in a system).

4.4.3.2 LORES Mask Mode Write Operations

To perform a mask mode write in LORES:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. Load the write mask in LORES mode; see above for instructions.

2. SET THE SHADE REGISTER AS NEEDED. This step may be omitted if the shade register has already been set. Set the shade register in LORES mode; see above for instructions.

3. LOAD THE MASK-MODE ADDRESS INTO THE IKONAS INTERFACE. The IKONAS bus address to use is a 24-bit quantity; the x-address register holds bits 0-9, and the y-address register holds bits 10-23. This address will depend on the LORES pixels being modified:

Bit 0	is unused.
1-5	contain the top 5 bits of the x address.
6-14	contain the y address.
15-19	must be zero.
20-23	must contain 0110 for mask mode writes.

If the mask-mode address is 011000000yyyyyyyyyxxxxx0

THEN the pixels accessed will be the 16 pixels at y-address yyyyyyyyy and x-address xxxxx0000 through xxxxx1111, where the x's and y's are construed as binary numbers.

4. SET THE FUNCTION CODE

24 octal

and the SENDER ID into the IKONAS interface command register. The sender id determines which write mask and shade value are used for the write operation.

5. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed or DMA); refer to the chapter "Host/IKONAS Interface"

in this manual for details.

There is no corresponding mask-mode READ. See the preceding section on the DR64 to read in pixel or word mode.

4.4.3.3 HIRES Mask Mode Write Operations

To perform a HIRES mask-mode write:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. Load the write mask in HIRES mode; see above for instructions.

2. SET THE SHADE REGISTER AS NEEDED. This step may be omitted if the shade register has already been set. Set the shade register in HIRES mode; see above for instructions.

3. LOAD THE MASK-MODE ADDRESS INTO THE IKONAS INTERFACE. The IKONAS bus address to use is a 24-bit quantity; the x-address register holds bits 0-9, and the y-address register holds bits 10-23. This address will depend on the HIRES pixels to be modified:

Bits 0-4	contain the top 5 bits of the x address.
5-14	contain the y address.
15-19	must be zero.
20-23	must contain 0110 for mask mode writes.

If the mask-mode address is 01100000yyyyyyyyyyxxxxx

THEN the pixels modified will be the 32 pixels at y-address yyyyyyyyyy and x-address xxxxx00000 through xxxxx11111, where the x's and y's are construed as binary numbers.

4. SET THE FUNCTION CODE

20 octal

and the SENDER ID into the IKONAS interface command register. The sender id determines which write mask and shade value are used for the write operation.

5. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed or DMA); refer to the chapter "Host/IKONAS Interface" in this manual for details.

There is no corresponding mask-mode READ. See the preceding section on the DR64 to read in pixel or word mode.

4.4.3.4 Effects of the Write Mask

When the write mask is set to protect certain bitplanes in LORES or HIRES mode, those bitplanes will be protected against mask-mode writes also. If you want to allow modifications to all bitplanes, write all ones (37777777777 octal) to the write mask in LORES or HIRES mode.

4.5 GM256

An IKONAS system consisting of GM256 memory cards allows two different kinds of write operations:

- o pixel mode
- o mask mode

This section explains:

- o how to read and write pixels in LORES and HIRES modes;
- o how to use mask mode;
- o what the visible effects of mask-mode operations are;
- o how to use the write masks to write only desired bits.

This section also provides a description of IKONAS framebuffer memory configurations.

4.5.1 Pixel Mode

In pixel mode, GM256's function in the same way as DR256's. All GM256's can be viewed as one continuous memory that is addressable in x and y. Each x,y memory location corresponds to a pixel. The bit depth of each memory location -- the number of bits which contribute to each pixel -- depends on the system configuration, as follows:

In low resolution (LORES) mode, each GM256 can provide 1,048,576 pixels of eight bits each, corresponding to a 512x512 display. Depending on the number of GM256's in the system configuration, the bit depth may be 8, 16, 24, or 32 bits. X addresses range from 0 to 1023. The y address range depends on the number of GM256's and the bit depth; see Appendix 4 of this chapter for bit depth and y address limits.

In high resolution (HIRES) mode, each GM256 can provide 4,194,304 pixels of 2 bits each, corresponding to a 2048 x 2048 pixel display. Depending on the number of GM256's in the system configuration, the bit depth may be any multiple of 2 from 2-24 bits. X addresses range from 0 to 2047. The y address range depends on the number of GM256's and the bit depth; see Appendix 4.

4.5.1.1 Examples of Configuration

Here are some example IKONAS memory organizations:

1. (Minimum system) The user has one GM256. It has a bit depth of 8 bits, x address range 0-1023, y address range 0-1023 (LORES mode).

2. (Full-color 1024x1024 system) The user has three GM256's. In LORES: bit depth is 24 bits. Card 0 supplies bits 0-7 (red), card 1 supplies bits 8-15 (green), and card 2 supplies bits 16-23 (blue). The x address range is 0-1023, y address range 0-1023.

   In HIRES: bit depth is 6 bits. Card 0 supplies bits 0-1, card 1 supplies bits 2-3, and card 2 supplies bits 4-5, so the user can use pseudocolor on a 6-bit HIRES image. The x address range is 0-2047; the y address range is 0-2047.

3. (Large system) The user has twelve GM256's. In LORES: the bit depth is 24 bits. The x address range is 0-1023; the y address range is 0-2047. (Note that in LORES mode, you cannot use all twelve cards.)

   In HIRES: the bit depth is 24 bits, with each card contributing 2 bits, for a full-color HIRES image. The x address range is 0-2047; the y address range, 0-2047.

NOTE: in LORES mode, the x and y coordinates must be multiplied by two.

4.5.2 Pixel-mode Addressing of the Framebuffer

In pixel mode (LORES or HIRES), each pixel is addressed independently of all the others; however, several GM256's may respond to the same pixel address, depending on the bit depth. (See the preceding section on configuration.)

When a program WRITES to a pixel address, any bits which are not configured into any GM256 are ignored.

When a program READS from a pixel address, any bits which are not configured to any GM256 are read as zero, but displayed as 1.

4.5.2.1 Pixel I/O

To write a pixel, you must do the following:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. See below for a discussion of the write mask.

2. LOAD THE PIXEL ADDRESS INTO THE IKONAS INTERFACE. The IKONAS interface contains two address registers, corresponding to x-address and y-address, so you can set the interface address registers with (x,y) coordinates.

However, for LORES mode only, you must load the x and y registers with twice the coordinate desired. For example, since the x or y address is a number between 0 and 1023, you will load the interface address registers with an even number between 0 and 2046. If your y address range is from 0-2047, you will load the y-address register with an even number between 0 and 4094.

These registers correspond to a 24-bit IKONAS bus address. In HIRES, bits 10-19 contain the y address; bits 0-9, the x address. In LORES, bits 11-19 contain the y address; bits 1-9, the x address.

3. SET THE FUNCTION CODE:

22 octal for LORES
23 octal for HIRES

and the desired sender id into the IKONAS interface command register. The sender id selects the write mask you will use (see below).

4. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed I/O or DMA); refer to the chapter "Host/IKONAS Interface" in this manual for details.

TO READ PIXELS, follow the same steps, with these exceptions:

1. you do not need to load the write mask;

2. use a function code of

02 octal for LORES
03 octal for HIRES

Again, see the chapter concerning the interface specification for details; in particular, for byte mode transfers, you must set the "byte mode" bit in the IKONAS status/command register.

4.5.2.2 Write Mask

The write mask is a set of eight registers, one for each sender id, which allows you to write only selected bits of a pixel when you write to the framebuffer. The write mask is used for all writes to the framebuffer, but it does not apply to any other component of the IKONAS system (for example, the LUVO). Each bit of the write mask controls a single bit of pixels written to the framebuffer. When a bit of the write mask is on (set to 1), the corresponding bit of each pixel in the framebuffer may be modified by writes to the framebuffer. When a bit if the write mask is off, however, the corresponding bit of each pixel in the framebuffer is protected, and will not be modified by writes to the framebuffer, even though other bits in the same pixel may be modified.

The sender id is part of each IKONAS bus write operation. For the GM256, it specifies which write mask is to be set or used. Having multiple write masks allows you to have different bitplanes open to modification by different processes, for example, character graphics planes to be modified by the CGM4 character generator, image planes to be modified by the BPS32 processor, and background planes to be modified by the host. The FBC always uses sender id 0 for its display-and-erase function, so you may have different write masks for erasing and writing memory.

The write mask must have been set before you issue a write, or the results are unpredictable; but once you set the write mask, it remains unchanged until you set it again.

There is only one write mask per sender id, but how it is interpreted depends on the function code used to set it. There are two 'set-write-mask' IKONAS bus function codes, one for LORES and one for HIRES. Normally, you should use the set-write-mask function corresponding to the type of write you wish to control.

TO SET THE WRITE MASK, perform all the steps for writing a single pixel to address (0,0), but use a function code of

32 octal for LORES
33 octal for HIRES

The data written will be used as the write mask for subsequent write operations.

In LORES mode, there is a separate set of write masks for every 1024 lines in memory; the y address distinguishes between these write masks. For example, if your system has 2048 lines, you would need one write mask for address (0,0) and another for address (0,1024), bearing in mind that LORES mode requires you to specify twice the coordinates desired.

In HIRES mode, there is one write mask for every 2048 lines (usually the maximum in a system).

In wire-wrapped GM256's, there is only one write mask, shared by all sender ids.

4.5.2.3 Z-buffer

The Z-buffer board, when present, allows you to do a conditional pixel write to certain bitplanes, which have been configured by wire-wrap jumpers as z-planes. When you perform a conditional write, the value to be written to these bitplanes is compared against their current value. The values are compared as unsigned integers.

If the old value is less than the new value, the write is suppressed and the bitplanes remain unchanged.

If the old value is greater than or equal to the new value, the write is performed. Function codes to use are:

26 octal for LORES

27 octal for HIRES

4.5.3 Mask Mode

Mask mode is a feature unique to the GM64 and GM256 which allows you to write up to 32 pixels in HIRES, or up to 16 pixels in LORES, with one operation. Each of the 32 data bits (for LORES, the lower 16 bits) corresponds to a pixel. If the bit has a value of 1, the pixel remains unchanged; if it has a value of 0, the pixel is written with the shade previously set in the shade register.

While WORD-mode writes in a DR256 modify only a single card, all GM256 cards respond to a mask-mode write operation.

4.5.3.1 Shade register

The shade register contains the value to be written to each pixel selected in a mask mode write. There are eight shade registers, one for each sender id. Each mask mode write sets a sender id which selects both the shade register and the write mask for that operation. Having multiple sender id's allows you to have different bitplanes open to modification by different processes, and to have different shade values from which to select.

The shade register must have been set before you issue a write, or the results are unpredictable; however, once you set the shade register, it remains unchanged until you set it again.

There is only one shade register for each sender id, but the function code to set it must correspond to the mask mode write you plan to do -- HIRES or LORES. To set the shade register, perform all the steps for writing a pixel to address (0,0), but use a function code of

36 octal for LORES

37 octal for HIRES

The data written will be used as the shade value for all subsequent mask mode writes using this sender id.

In LORES mode, there is a separate shade of shade registers for every 1024 lines in memory; the y address distinguishes between these write masks. For example, if your system has 2048 lines, you would need one write mask for address (0,0) and another for address (0,1024), bearing in mind that LORES mode requires you to specify twice the coordinates desired.

In HIRES mode, there is one write mask for every 2048 lines (usually the maximum in a system).

4.5.3.2 LORES Mask Mode Write Operations

To perform a mask mode write in LORES:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. Load the write mask in LORES mode; see above for instructions.

2. SET THE SHADE REGISTER AS NEEDED. This step may be omitted if the shade register has already been set. Set the shade register in LORES mode; see above for instructions.

3. LOAD THE WORD ADDRESS INTO THE IKONAS INTERFACE. The IKONAS bus address to use is a 24-bit quantity; the x-address register holds bits 0-9, and the y-address register holds bits 10-23. This address will depend on the LORES pixels to be modified:

Bit 0	is unused.
1-5 and 17	contain the top 6 bits of the x address (bit 17 is the high-order x bit).
6-14 and 16	contain the y address (bit 16 is the high-order y bit).
15 and 18-19	must be zero.
20-23	must contain 0110 for mask mode writes.

If the mask-mode address is 011000xy0yyyyyyyyyxxxxx0

THEN the pixels accessed will be the 16 pixels at y-address yyyyyyyyyy and x-address xxxxxx0000 through xxxxxx1111, where the x's and y's are construed as binary numbers.

4. SET THE FUNCTION CODE

24 octal

and the SENDER ID into the IKONAS interface command register. The sender id determines which write mask and shade value are used for the write operation.

5. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed or DMA); refer to the chapter "HOST/IKONAS INTERFACE" in this manual for details.

4.5.3.3 HIRES Mask Mode Write Operations

To perform a HIRES mask-mode write:

1. SET THE WRITE MASK AS NEEDED. This step may be omitted if the write mask has already been set. Load the write mask in HIRES mode; see above for instructions.

2. SET THE SHADE REGISTER AS NEEDED. This step may be omitted if the shade register has already been set. Set the shade register in HIRES mode; see above for instructions.

3. LOAD THE WORD ADDRESS INTO THE IKONAS INTERFACE. The IKONAS bus address to use is a 24-bit quantity; the x-address register holds bits 0-9, and the y-address register holds bits 10-23. This address will depend on the HIRES pixels to be modified:

Bits 0-4 and 17	contain the top 6 bits of the x address (bit 17 is the high-order x bit).
5-14 and 16	contain the y address (bit 16 is the high-order y bit).
15 and 18-19	must be zero.
20-23	must contain 0110 for mask mode writes.

If the mask-mode address is 01100000yyyyyyyyyyxxxxx

THEN the pixels modified will be the 32 pixels at y-address yyyyyyyyyyy and x-address xxxxxx00000 through xxxxxx11111, where the x's and y's are construed as binary numbers.

4. SET THE FUNCTION CODE

20 octal

and the SENDER ID in the IKONAS interface command register. The sender id determines which write mask and shade value are used for the write operation.

5. START THE I/O OPERATION. Exact details depend upon your interface and mode of transfer (programmed or DMA); refer to the chapter "Host/IKONAS Interface" in this manual for details.

There is no corresponding mask-mode READ. See the preceding section on the DR256 to read in pixel or word mode.

4.5.3.4 Effects of the Write Mask

When the write mask is set to protect certain bit-planes in LORES or HIRES mode, those bitplanes will be protected against mask-mode writes also. If you want to allow modifications to all bitplanes, write all ones (37777777777 octal) to the write mask in LORES or HIRES mode.

APPENDIX O. FUNCTION CODE SUMMARY

All function codes are in octal.

Function	DR64 DR256	GM64 GM256
Pixel mode write - LORES	22	22
Pixel mode write - HIRES	23	23
Word mode write	20	--
Mask mode write - LORES	--	24
Mask mode write - HIRES	--	20
Set shade - LORES	--	36
Set shade - HIRES	--	37
Set write mask - LORES	32	32
Set write mask - HIRES	33	33
Conditional write - LORES (Z-buffer)	--	26
Conditional write - HIRES (Z-buffer)	--	27
Pixel mode read - LORES	02	02
Pixel mode read - HIRES	03	03
Word mode read	00	00

APPENDIX 1

STANDARD DR64 CONFIGURATIONS

CARD ADDRESS	WORD MODE ADDRESS RANGE	HIRES X range	HIRES Y range	HIRES Bit range	LORES X range	LORES Y range	LORES Bit range	IKMTS RESPONSES
10	1000$0-1077$1777	0-1023	0-1023	0-1	0-511	0-511	0-7	10
11	1100$0-1177$1777	0-1023	0-1023	2-3	0-511	0-511	8-15	11,,10,2
12	1200$0-1277$1777	0-1023	0-1023	4-5	0-511	0-511	16-23	12,,20,4
13	1300$0-1377$1777	0-1023	0-1023	6-7	0-511	0-511	24-31	13,,30,6
14	1400$0-1477$1777	0-1023	0-1023	8-9	0-511	512-1023	0-7	14,1,,10
15	1500$0-1577$1777	0-1023	0-1023	10-11	0-511	512-1023	8-15	15,1,10,12
16	1600$0-1677$1777	0-1023	0-1023	12-13	0-511	512-1023	16-23	16,1,20,14
17	1700$0-1777$1777	0-1023	0-1023	14-15	0-511	512-1023	24-31	17,1,30,16
01	0100$0-0177$1777	0-1023	0-1023	16-17	0-511	1024-1535	0-7	1,2,,20
02	0200$0-0277$1777	0-1023	0-1023	18-19	0-511	1024-1535	8-15	2,2,10,22
03	0300$0-0377$1777	0-1023	0-1023	20-21	0-511	1024-1535	16-23	3,2,20,24
04	0400$0-0477$1777	0-1023	0-1023	22-23	0-511	1024-1535	24-31	4,2,30,26

NOTES:

1. Card addresses, word mode addresses, and IKMTS responses are given in OCTAL; all other numbers are DECIMAL. The card address occupies bits 16-19 of the word address.

2. LORES addresses must be multiplied by 2.

3. The column headed "IKMTS RESPONSES" shows the responses to be used in the memory test for DR64/GM64 boards (WORD MODE BANK, IMAGE NUMBER, LORES BIT #, HIRES BIT #). See "IKMTS" in the IKONAS INSTALLATION GUIDE.

STANDARD 1-CARD SYSTEM: card 10 (red)
STANDARD 3-CARD SYSTEM: cards 10, 11, 12 (red, green, blue)
STANDARD 4-CARD SYSTEM: cards 10, 11, 12, 13 (red, green, blue, alpha)
STANDARD 12-CARD SYSTEM: cards 10 - 04 (3 sets of red, green, blue, alpha cards)

APPENDIX 2

STANDARD DR256 CONFIGURATIONS

CARD ADDRESS	WORD MODE ADDRESS RANGE	HIRES X range	HIRES Y range	HIRES Bit range	LORES X range	LORES Y range	LORES Bit range	IKMTS RESPONSES
10	1000$0-1377$1777	0-2047	0-2047	0-1	0-1023	0-1023	0-7	10
14	1400$0-1777$1777	0-2047	0-2047	2-3	0-1023	0-1023	8-15	14,,10,2
04	0400$0-0777$1777	0-2047	0-2047	4-5	0-1023	0-1023	16-23	4,,20,4
20	2000$0-2377$1777	0-2047	0-2047	6-7	0-1023	0-1023	24-31	20,,30,6
24	2400$0-2777$1777	0-2047	0-2047	8-9	0-1023	1024-2047	0-7	24,1,,10

NOTES:

1. Card addresses, word mode addresses, and IKMTS responses are given in OCTAL; all other numbers are DECIMAL. The card address occupies bits 18-21 of the word mode address.
2. LORES pixel coordinates must be multiplied by 2.
3. The column headed "IKMTS RESPONSES" shows the responses to be used in the memory test for DR256/GM256 boards (WORD MODE BANK, IMAGE NUMBER, LORES BIT #, HIRES BIT #). See "IKMTS" in the IKONAS INSTALLATION GUIDE.

STANDARD 1-CARD SYSTEM: card 10 (red)
STANDARD 3-CARD SYSTEM: cards 10, 14, 4 (red, green, blue)
STANDARD 4-CARD SYSTEM: cards 10, 14, 4, 20 (red, green, blue, alpha)

APPENDIX 3

STANDARD GM64 CONFIGURATIONS

CARD ADDRESS	HIRES X range	HIRES Y range	HIRES Bit range	LORES X range	LORES Y range	LORES Bit range	IKMTS RESPONSES
10	0-1023	0-1023	0-1	0-511	0-511	0-7	10
11	0-1023	0-1023	2-3	0-511	0-511	8-15	11,,10,2
12	0-1023	0-1023	4-5	0-511	0-511	16-23	12,,20,4
13	0-1023	0-1023	6-7	0-511	0-511	24-31	13,,30,6
14	0-1023	0-1023	8-9	0-511	512-1023	0-7	14,1,,10
15	0-1023	0-1023	10-11	0-511	512-1023	8-15	15,1,10,12
16	0-1023	0-1023	12-13	0-511	512-1023	16-23	16,1,20,14
17	0-1023	0-1023	14-15	0-511	512-1023	24-31	17,1,30,16
01	0-1023	0-1023	16-17	0-511	1024-1535	0-7	1,2,,20
02	0-1023	0-1023	18-19	0-511	1024-1535	8-15	2,2,10,22
03	0-1023	0-1023	20-21	0-511	1024-1535	16-23	3,2,20,24
04	0-1023	0-1023	22-23	0-511	1024-1535	24-31	4,2,30,26

NOTES:

1. Card addresses and IKMTS responses are given in OCTAL; all other numbers are DECIMAL.
2. LORES pixel coordinates must be multiplied by 2.
3. The column headed "IKMTS RESPONSES" shows the responses to be used in the memory test for DR64/GM64 boards (WORD MODE BANK, IMAGE NUMBER, LORES BIT #, HIRES BIT #). See "IKMTS" in the IKONAS INSTALLATION GUIDE.

STANDARD 1-CARD SYSTEM: card 10 (red)
STANDARD 2-CARD SYSTEM: cards 10, 11, 12 (red, green, blue)
STANDARD 4-CARD SYSTEM: cards 10, 11, 12, 13 (red, green, blue, alpha)
STANDARD 12-CARD SYSTEM: cards 10 - 04 (3 sets of red, green, blue, alpha)

APPENDIX 4

STANDARD GM256 CONFIGURATIONS

CARD ADDRESS	HIRES X range	HIRES Y range	HIRES Bit range	LORES X range	LORES Y range	LORES Bit range	IKMTS RESPONSES
10	0-2047	0-2047	0-1	0-1023	0-1023	0-7	10
14	0-2047	0-2047	2-3	0-1023	0-1023	8-15	14,,10,2
04	0-2047	0-2047	4-5	0-1023	0-1023	16-23	4,,20,4
20	0-2047	0-2047	6-7	0-1023	0-1023	24-31	20,,30,6
24	0-2047	0-2047	8-9	0-1023	1024-2047	0-7	24,1,,10

NOTES:

1. Card addresses and IKMTS responses are given in OCTAL; all other numbers are DECIMAL.
2. LORES pixel coordinates must be multiplied by 2.
3. The column headed "IKMTS RESPONSES" shows the responses to be used in the memory test for DR256/GM256 boards (WORD MODE BANK, IMAGE NUMBER, LORES BIT #, HIRES BIT #). See "IKMTS" in the IKONAS INSTALLATION GUIDE.

STANDARD 1-CARD SYSTEM: card 10 (red)
STANDARD 3-CARD SYSTEM: cards 10, 14, 4 (red, green, blue)
STANDARD 4-CARD SYSTEM: cards 10, 14, 4, 20 (red, green, blue, alpha)

5.0 FRAME BUFFER CONTROLLER (FBC) PROGRAMMING GUIDE

NOTE: Bit 0=LSB, bit 31=MSB

5.1 INTRODUCTION

The FBC controls the sequencing of data from the frame buffer memory to the color map memory. It also performs video clock generation is performed by the FBC. From a user standpoint, there are eight 32-bit wide registers which control the mode of operation, timing, video display parameters, and cursor display.

5.1.1 REGISTER ACCESS

Data can be written into any of the eight 32-bit control registers by using the correct IKONAS bus address along with a WRITE function command during a data transfer. This is accomplished from the host processor by setting data and address registers properly, setting the WRITE bit in the interface control word and initiating a transfer.

For the FBC the upper 14 bits of address must be 30000 (octal). The low 10 bits of address (bits 0-9) are used to specify the register to be modified. Address bits 3-9 should be zero; the lower 3 bits of address are decoded to determine which register will have data written into it. In the following section, only the lower address bits will be used to identify registers.

5.2 FBC REGISTER DEFINITION

5.2.1 VIEWPORT LOCATION (REGISTER 30000$0)

The lower 16 bits of the VIEWPORT LOCATION register determine the horizontal starting position of the displayed data on the monitor screen. Only bits 2-9 are actually used by the controller, so that positioning can be selected in four-pixel increments only.

The upper 16 bits of the VIEWPORT LOCATION register determine the vertical starting position of the displayed data on the monitor screen. Only bits 18-25 are actually used by the controller, so that positioning can be selected in four-line increments only.

When REPEAT FIELD is set (see below), the y viewport value should be multiplied by two.

5.2.2 VIEWPORT SIZE (REGISTER 30000$1)

The lower 16 bits of the VIEWPORT SIZE register determine the width (number of pixels) of the displayed data on the monitor screen. Only bits 2-9 are actually used by the controller, so that displayed width can be selected in four-pixel increments only.

The upper 16 bits of the VIEWPORT SIZE register determine the height (number of lines) of the displayed data on the monitor screen. Only bits 18-25 are actually used by the controller, so that displayed height can be selected in four-line increments only.

5.2.3 WINDOW LOCATION (REGISTER 30000$2)

The lower 16 bits of the WINDOW LOCATION register determine the starting X address (within the frame buffer memory) of the data to be displayed

within the viewport. In LORES, bits 2-10 give the starting pixel number (bits 0-1 are unused); in HIRES, bits 0-9 give the starting pixel number. In either case, the number in the appropriate bits is a pixel number, so single-pixel scrolling is possible.

The upper 16 bits of the WINDOW LOCATION register determine the starting Y address (within the frame buffer memory) of the data to be displayed within the viewport. Bits 16-28 are used by the controller, so that single-line scrolling is possible.

5.2.3.1 MULTIPLE LORES IMAGES

In systems with more than four DR64 or GM64 image memories, the image memories (in LORES display mode) will be grouped into 512x512 images, numbered starting at 0. For purposes of video display, the 512x512 images are stacked vertically; that is, the first line of image 1 immediately follows the last line of image 0. In a system with three images, for example, the images are addressed just as if they were one image of 1536 lines and 512 columns. To display image 1, the y window would be set to 512; to display the bottom half of image 0 and the top half of image 1, the y window would be set to 256.

5.2.3.2 Note on Y Viewport and Y Window

The following describes exactly the operation of the y viewport and y window registers; the explanation is detailed, and you may want to skip over it to the formulas and examples.

5.2.3.2.1 Introductory Note on Video Terminology

The visible portion of the picture is preceded by the vertical blanking interval, which is an integral number of lines (20 for RS-170A sync, 40 for RS-343 sync).

Vertical drive is a signal which occurs during the vertical blanking interval and lasts for three lines (RS-170A) or four lines (RS-343). A CRT monitor starts the downward scan of the electron beam when vertical drive occurs.

In the Y viewport and Y window settings, you should remember that the first line of the screen, line 0, is within the vertical blanking interval. The vertical blanking interval continues for 17 lines after the start of vertical drive in RS-170A mode, or 34 lines after the start of vertical drive in RS-343 mode.

5.2.3.2.2 Y Viewport

The Y viewport value specifies the number of lines to be held black from the beginning of vertical drive. The number of lines to be blanked will be four more than the number specified in the viewport register. Remember that the two low-order bits of the viewport register are forced to zero.

The picture is forced to black during the vertical blanking interval, regardless of the viewport setting. If you specify a viewport setting less than the first unblanked line, the end of the picture will be calculated from the specified starting line using the viewport size value; but the start of the picture will be the first unblanked line. The exact number of lines between start of vertical drive and the first visible line is the maximum of the blanking interval and the y-viewport-setting+4 (call this number the top margin).

Remember that when the REPEAT FIELD bit (described below) is set, the y viewport setting should be multiplied by two.

5.2.3.2.3 Y Window

The starting y coordinate (the y window value) is defined to be the line number displayed on the first line after vertical drive. Note that this line will always be invisible; the first visible line will be at least 17 lines later.

Therefore, the y coordinate of the first visible line will be a function of the top margin, y window, and width of vertical drive.

5.2.3.2.4 Formulas for computing y window

If you have chosen a y viewport setting and want a specified line to be the first visible line of the screen, calculate the y window setting using the formulas below. If the result is negative, express it in two's complement form, but use at least twelve bits.

First, calculate the top margin as follows (in these formulas TM=top margin, YV=y viewport, YW=y window, FL=desired first line):

For RS-170A sync: TM=MAX(35,YV+4)

For RS-343 noninterlaced: TM=MAX(34,(YV+4)/2)

For RS-343 interlaced: TM=MAX(69,YV+4)

Now calculate the y window:

For RS-170A sync: YW=FL-TM+7

For RS-343 noninterlaced: YW=FL-TM+4 For

RS-343 interlaced: YW=FL-TM+9

Example: you are using RS-170A sync. The y viewport is set to 20 (decimal). What should the y window be to cause line 0 to be displayed as the first visible line?

Solution: the top margin is 24 (maximum of 20+4 and 17); therefore the y window should be set to 0-24+7=-17, or, in octal, 7757 (two's complement to twelve bits).

5.2.4 ZOOM (REGISTER 30000$3)

The lower 16 bits of the ZOOM register determine the pixel replication factor for horizontal zooming. Bits 0-7 are used by the controller so that horizontal zoom ratios of 1:1 to 256:1 in integer steps can be selected. (But see "Notes" at the end of this chapter for an exception in HIRES mode.)

The upper 16 bits of the ZOOM register determine the line replication factor for vertical zooming. Bits 16-33 are used by the controller so that vertical zoom ratios of 1:1 to 256:1 in integer steps can be selected.

5.2.5 DISPLAY RATE CONTROL (REGISTER 3000$4)

The lower 16 bits of the DISPLAY RATE control register determine the horizontal scan frequency. Bits 0-11 are used to determine the number of 4.77 MHz clock cycles per line (209.5 NSEC resolution).

The upper 16 bits of the DISPLAY RATE control register determine the vertical scan frequency. Bits 16-27 are used to determine the number of scan lines per video frame. An even number of scan lines must be used for interlaced field operation.

5.2.6 VIDEO CONTROL (REGISTER 30000$5)

5.2.6.1 CURSOR CONTROL

Bit 2 = 0 Cursor OFF
= 1 Cursor ON

5.2.6.2 RESOLUTION CONTROL

Bit 3 = 0: Low Resolution
(512x512 display). In this mode a DR64 furnishes 8 512x512 bitplanes.
= 1: High Resolution
(1024x1024 display). In this mode a DR64 furnishes 2 1024x1024 bitplanes.

5.2.6.3 AUTO-CLEAR CONTROL

Bit 5 = 1 "Display-then-clear" memory operation.

Permits rapid (one frame) clear of all displayed memory locations.

5.2.6.4 DISPLAY SYNC SELECT

Bit 6 Select one of two SYNC methods

= 1 External SYNC (RS-343 or RS-170)
= 0 Internal SYNC from DISPLAY RATE CONTROL specification

5.2.6.5 COLOR MAP PAGE

Bit 7 determines which one of two color map pages is used for display. This is used for addressing of the 1024x24 LUV024 or LUV024/HS color maps. The colormap address is derived from the pixel data, the cursor, and the page bit (see section 2.7 of this chapter).

Bit 8 is an additional page bit which can be used in systems with a video crossbar switch for additional control over color map selection. (See section 3.1 of the "Crossbar Switch Programming Guide" in this manual.)

5.2.6.6 SYNC TIMING SELECTION

Bit 9 = 0 for RS-170 sync standard (525 line NTSC standard)
Bit 9 = 1 for RS-343 sync standard (high resolution monitors)

Bit 10 = 0 interlaced field operation
Bit 10 = 1 repeat field operation

5.2.6.7 PIXEL CLOCK RATE SELECT

Bits 16-22 = 20-77 (octal): number of nanoseconds per pixel (in HIRES); in LORES, the number of nanoseconds per pixel is twice the number given.

5.2.7 CURSOR LOCATION (REGISTER 30000$6)

The lower 16 bits of the CURSOR LOCATION register determine the cursor horizontal position in pixels. Bits 0-9 are used.

The upper 16 bits of the CURSOR LOCATION register determine the cursor vertical position in lines. Bits 16-25 are used.

A cursor y-location of 0 corresponds to the the first line after vertical drive (see discussion of y viewport and y window above). The screen line containing the top left point of the cursor will have the y-coordinate calculated as (y cursor setting+y window setting).

5.2.8 PROGRAMMABLE CURSOR

The FBC contains a 32x32 bit programmable cursor. This is a 32x32 pixel area, whose upper left corner is specified by the cursor location register (see 2.6). Each pixel has a programmable 1-bit value inserted into the pixel output. For pixels outside the 32x32 area, the bit is 0; for pixels within the area, the bit is 1 or 0 depending on the programmed value. The cursor bit is applied as bit 8 of the colormap address; bit 9 is the colormap page bit (see 2.5.5); bits 0-7 are pixel data.

The display output of the colormap depends on all three settings, as follows: The page bit selects one of two colormap pages. Within each page, there is a 'cursor section' and a 'noncursor section'. The output of the colormap is the value in the memory addressed by pixel value: from the cursor section, if the pixel is in the cursor and the cursor bit is 1; from the noncursor section, otherwise. Thus, the value displayed at a cursor bit may be made to depend on the pixel value (of course, loading the cursor section to constant, will produce a solid-colored cursor).

To set the programmable cursor, load the Frame Buffer Controller for the 256 words starting at Y=30000, X=400. Each of the 256 words specifies four bits of cursor. The four bits to be loaded must be the low-order four bits of the data written

to the FBC; the upper 28 bits are ignored. The particular four cursor bits modified are decoded from the eight low-order address bits as follows:

```
Low 8 address bits (Bit 8=1):   yyyyyxxx
Bit number                      76543210
```

where yyyyy is the y-offset of the data within the cursor;

xxx00 is the x-offset of the data within the cursor.

The x and y offsets range from 0 to 31. y-offset 0 is the TOP cursor line, 31 is the bottom. x-offset 0 is the LEFTMOST column, 31 is the rightmost. Of the four bits written at one time, bit 0 is the leftmost, bit 3 is the rightmost.

Here is a map of the cursor as it is displayed. Each digit corresponds to one bit loaded into a cursor location; the markings show which locations and bits correspond to displayed cursor pixels. The cursor address register (30000$6) gives the (x,y) location of the top left point of the cursor (30000$400 bit 0).

```
        |--------------address 30000$400
        |
        |    |---------address 30000$401
        |    |
        |    |   |------address 30000$402 bit 0
        |    |   ||-----address 30000$402 bit 1
        |    |   |||----address 30000$402 bit 2
        |    |   ||||---address 30000$402 bit 3
       |--||--|||||
       01230123012301230123012301230123 cursor line 0
30000$410->>01230123012301230123012301230123 cursor line 1
30000$420->>01230123012301230123012301230123 cursor line 2
30000$430->>01230123012301230123012301230123 cursor line 3
          (...cursor lines 4-27...)
30000$740->>01230123012301230123012301230123 cursor line 28
30000$750->>01230123012301230123012301230123 cursor line 29
30000$760->>01230123012301230123012301230123 cursor line 30
30000$770->>01230123012301230123012301230123 cursor line 31
       |                                    |
       |                                    |
       cursor col 0                         cursor col 31
```

5.3 Notes

Currently, in HIRES mode, horizontal zoom must be accomplished as follows:

1. To go from a ratio of 1:1 to 2:1, you must double the pixel clock rate, rather than use the zoom register.

2. Thereafter you can increment the zoom register, while leaving the pixel clock rate doubled.

6.0 CROSSBAR SWITCH (XBS34) PROGRAMMING GUIDE

6.1 Introduction

The crossbar switch (XBS34) is a video pipeline stage which accepts pixels at video rate and produces output pixels at video rate. The output is 34 bits per pixel, where each of the 34 output bits is taken directly from some specified one of the input bits. Thus the name crossbar switch, since any configuration of the XBS34 does no more than rearrange, duplicate, or discard input bits. Each pixel is treated separately, and the same operation is performed on each pixel.

6.2 Input to the XBS34

Input to the XBS34 is usually from the FBC. 35 bits per pixel are used: bits 0-31 are the pixel data bits, originating in the DR64; bits 32-34 are the page bits, originating in the FBC. Bit 32 is the cursor bit, on when the cursor is active; bits 33 and 34 are determined by bits 7-8 of the FBC control register at 30000$5. See the "Frame Buffer Controller (FBC) Programming Guide" for details on programming the cursor and page bits.

6.3 Output from the XBS34

The output of the XBS34 goes to the next pipeline stage, usually the LUVO. The output of the XBS34 is the form required by the LUVO: 32 pixel data bits and two page bits, 34 bits in all. If the LUVO has the overlay option, bits 24-31 of the pixel data are construed as overlay information--see the "LUVO Programming Guide" for details.

6.4 Programming the XBS34 data path

The XBS34 contains 34 write-only registers addressed in the range 30200$0-30200$41. There is one register for each output bit. The value loaded into the register for one output bit is the number of the input bit which is to be connected to that output bit, or 63 (decimal) if the output bit is to be held at zero regardless of the input.

The register address for output bit x is 30200$0+x.

6.5 Restrictions on output bits 0-29

Output bits 0-29 cannot be connected to the page bits (input bits 32-34). They may be connected to any pixel data bit (input bits 0-31) or to constant zero.

Output bits 30-33 may be connected to any pixel data bit, any page bit, or constant zero.

7.0 LUVO PROGRAMMING GUIDE

7.1 Introduction

The LUVO (Lookup and video output) accepts data at video rates and produces analog signals for red, green, and blue. The input data is sent first through a three-channel crossbar switch to allow software selection of full-color or pseudocolor operation; the three channel outputs of the crossbar, representing red, green, and blue information, are sent to individual lookup tables for conversion to the output value; the output values are sent to video D/A converters where they are converted to analog voltages with proper video sync and blanking information added.

Each of these components is described below.

7.2 Three-channel crossbar switch (LUVOB only)

7.2.1 Format of video data into the LUVO

The LUVO receives video data on the 72-pin ribbon cable at the top. Video data comes from whatever is the previous element in the video pipeline, usually either the frame buffer controller (FBC) or the 34-bit crossbar switch (XBS). The cable allows for 32 bits of video data and two extra 'page' bits. In systems with an XBS crossbar switch, the assignment of memory bits to video data bits is governed by the crossbar. In systems without a crossbar, the video data bits come directly from the memory cards through the FBC to the LUVO, and the assignment of memory bits to video data bits is determined by the configuration of the memory cards and is specified by the customer when the system is configured.

In any case, the 32 video data bits are divided into four eight-bit channels: bits 0-7 (red), 8-15 (green), 16-23 (blue), and 24-31 (overlay). The red, green, and blue channels will be referred to as

background channels.

7.2.2 Channel crossbar switch

The three eight-bit background channels are the inputs to a crossbar switch. The crossbar switch has three output channels; each output channel can be taken from any of the three input channels under software control. The usual methods of setting up the crossbar switch are:

Full-color operation.

The output red channel is taken from the input red channel, the output green channel is taken from the input green channel, and the output blue channel is taken from the input blue channel. The crossbar switch is tranparent; the output data equals the input data.

Pseudocolor operation.

All three output channels are taken from one of the input channels. All three lookup tables receive the same data.

Other methods of setting up the crossbar are allowed at the user's option.

The crossbar switch control register resides at location 20301$0 and is write-only. Any word written to 20301$0 will set the crossbar switch control; only the six low-order bits of the data written are used, and they are in the following format:

```
bit 543210
    bbggrr
```

where

rr is the two-bit code specifying which input channel will be connected to the red crossbar output:

00 = red input
01 = green input

10 = blue input

(11 is reserved for future use);

gg is the two-bit code for the green output;

bb is the two-bit code for the blue output.

7.2.3 Examples

The data to write to the crossbar for the usual cases is as follows: For full-color operation, write 44 (octal) to 20301$0; for pseudocolor with data coming from the red channel, write 0; for pseudocolor from green, write 25 octal; for pseudocolor from blue, write 52 octal.

7.3 Lookup table operation

The lookup tables in the LUV0 are 1024x8 (for the LUV024) or 1024x10 (for the LUV030). There are three lookup tables, corresponding to the three output channels of the channel crossbar.

7.3.1 Lookup table addressing from video data

Each color lookup table is addressed from one channel of the channel crossbar switch. The 10-bit lookup table address is derived as follows:

```
bit 9876543210
    ppcccccccc
```

where

cccccccc is the eight-bit channel from the crossbar switch;

pp are the two page bits from the video cable.

The page bits are applied to all three video lookups simultaneously; the channel information may be different if the crossbar switch is set up for full-

color operation. In systems with an XBS crossbar switch, the page bits are assigned by the XBS; in systems without an XBS, page bit 0 (the lower page bit) is on if the cursor is active, and page bit 1 is set by the user in a control register on the frame buffer controller. See the "Frame Buffer Controller Programming Guide" for details on the cursor and page bits.

The data present at the output of a lookup table after video data has been supplied at the address of the lookup table will be sent to the D/A converters. The lookup memory must have been initialized from the IKONAS bus for meaningful operation.

7.3.2 IKONAS bus addressing

The three lookup tables are accessed from the IKONAS bus using normal IKONAS read and write operations. The lookup tables occupy locations 20300$0 to 20300$1777 in the IKONAS address space. A single access to a location between 20300$0 and 20300$1777 will access all three lookup tables: the ten low-order bits of the IKONAS bus address are applied to each lookup table; the correspondence of channel data bits to IKONAS bus data bits is as follows:

```
 bit  3         2         1
     10987654321098765432109876543210
     xxbbbbbbbbbbggggggggggrrrrrrrrrr
```

where

rrrrrrrrrr (bits 0-9) correspond to the data bits of the red channel;

gggggggggg (bits 10-19) correspond to the data bits of the green channel;

bbbbbbbbbb (bits 20-29) correspond to the data bits of the blue channel;

xx (bits 30-31) are reserved for future use.

The data written by IKONAS bus writes to a location of a lookup memory will be displayed as the intensity of that channel when the video address (derived as noted above) selects that location. Note that there is no direct path from the video data to the D/A converters without going through the lookup operation.

The LUVO24 has only eight bits of data per lookup table; the low-order two bits of data for each channel (bits 0-1, 10-11, 20-21) are ignored for writes, and read as 0. To compensate for the dropped bits, you would write values of 0-1023 octal, in increments of 4, to effectively load the color map with values between 0 and 255.

7.4 D/A converters

The output of each video lookup table is sent to a D/A converter. The ten-bit (for LUVO24, eight-bit) value is converted to a voltage between -1 volt and 0 volts if noncomposite sync has been requested. If the composite sync option has been requested, the value is converted to a voltage between -0.7 volts and 0 volts, and composite sync information is added.

7.5 Overlay option

The overlay option of the LUVO allows full-color graphics overlays without change to the background picture.

The programming of the LUVO from the IKONAS bus is unchanged by the overlay option, but the video addressing is modified as described below.

Overlay information is received in video data bits 24-31. When the bits of the overlay channel are all zero, operation of the LUVO is similar to that described above, except that bit 8 of each lookup memory address is set to 0 and bit 9 of each lookup memory address is taken from page bit 0. Bits 0-7 of

the lookups are taken from the crossbar switch.

When any of the overlay bits is one, the video address of each lookup table is the same, and is derived as follows: bits 0-7 of the address are taken from the overlay data; bit 8 is 1; and bit 9 is taken from page bit 0.

When the overlay bits are 0, the background picture is displayed normally. When <u>any</u> overlay bit is present, the eight overlay bits are construed as an eight-bit color and applied to the lookup addresses; simultaneously, bit 8 of the lookup address is set to 1; so there are different lookup pages for the overlay and background data.

8.0 BIPOLAR PROCESSOR (BPS) HOST PROGRAMMING GUIDE

This chapter tells how to allow the Bipolar Processor (BPS) to execute programs. This chapter does not tell how to write and assemble programs; that information is given in the Bipolar Processor (BPS) Programming Guide. This chapter describes the format of microcode words, the BPS status register, and the host programming required to start, stop, and single-step the processor.

8.1 Microcode word format

Microcode words are 64 bits long and are addressed as 64-bit words by the program counter (PC) of the BPS. The same memory is addressed as 32-bit words from the IKONAS bus. For example, one MCM4 contains 4096 64-bit microcode words, so a single program can be up to 4096 instructions long. The same MCM4 occupies 8192 locations of the IKONAS bus address space.

Each 64-bit microcode word corresponds to an even-odd pair of 32-bit IKONAS words. The address of the even word of the pair is:

```
bit  2   2         1         0
     3   0         0         0
     bbbbbbbbbbbmmmmmmmmmmmmm
```

where

bbbbbbbbbbb is the 11-bit high-order base address for the MCM4. The first microcode memory is addressed at 20000$0; bbbbbbbbbbb will be 10000000000.

mmmmmmmmmmmm is the address of the 64-bit microcode word as given by the PC.

The address of the odd word of the pair is one more than the address of the even word. The even word is the less-significant word (bits 0-31 of the instruction); the odd word is instruction bits 32-63.

What this all boils down to is this: to get the IKONAS bus address of instruction x, use 20000$0+(2*x) for the even address. For example, instruction 5 is

at 20000$12 and 20000$13.

8.1.1 BPS status register

Each BPS in a system has a status register whose address is given in the configuration sheet for the system. For systems with a single processor, the status register is at 20500$0. This register can be read to see the PC of the next instruction to be executed, or can be written to set the processor mode.

8.1.1.1 Reading the status register (PC)

When the processor is stopped and the status register is read, the low 16 bits of the result contain the PC of the second-next instruction to be executed. The PC can be read only when the processor is stopped.

The address in the PC is the address of the first instruction which can be modified to change the operation of the processor; in other words, if the PC is read as x, you can modify instruction x and the modified instruction will be executed.

Note that instruction x is not the next instruction to be executed, but is instead the instruction after the next instruction. The next instruction to be executed has already begun to be processed and will be executed when the processor is started.

8.1.1.2 Writing the status register

When the status register is written, the three low-order bits of the data written control the operation of the processor. Bits 3-31 of the data written are ignored.

Bit 0 of the status register is the stop bit. When the bit is on, the processor is halted; when it is

off the processor is allowed to run (subject to the host-interface 'run' line, described later).

Bit 1 of the status register is the single-step bit. When the bit is on, the processor will stop after each instruction is executed; when the bit is off, the processor runs freely.

Bit 2 resets the PC to location 0.

When the processor is reset (see below), the status register is initialized to 0: free running.

8.1.1.3 Stop and single-step using the status register

When the stop bit in the status register is on, the processor does not run.

When the single-step bit is on, any store to the status register with the stop bit off causes the processor to be started. To single-step, then, you program as follows:

- stop the processor (store 1 to status register)
- to execute an instruction, set 'single step, run': store 2 to the status register.
- if you want to single-step another instruction, return to step 2.
- if you want to release the processor for free running, store 0 to the status register.

8.1.2 Host programming of the BPS

The operation of the processor is controlled by a combination of the status register on the BPS, two control lines in the host interface, and the front-panel reset button. The chapter entitled "Host/IKONAS Interface" tells you how to set the bits in the interface; this section explains the use of the two bits controlling the BPS.

8.1.2.1 RUN line

The RUN bit in the IKONAS control/status register of the host interface must be on to allow the processor to execute instructions. The processor will execute instructions when the RUN bit in the interface's IKONAS control/status register ison, and the stop bit (bit 0) in the processor status register is off.

8.1.2.2 RESET line

The RESET bit in the IKONAS control/status register of the host interface is set to force the program counter (PC) of the processor to zero. As long as the RESET bit is on, the processor will repeatedly execute the instruction at location 0. Note: for the RESET bit to be effective, the processor must be running. To start the processor at location 0:

- Set the RESET and RUN bits in the IKONAS control/status register of the interface. The processor will be executing at 0.
- Turn off the RESET bit in the IKONAS control/status register, leaving the RUN bit on. The processor will run freely, beginning at location 0.

Note: setting this bit sets program counters in ALL devices to zero. To reset the PC of the BPS only, set bit 2 of the BPS status register (see "Writing the status register," earlier in this chapter).

8.1.2.3 Front-panel RESET button

When the RESET button on the front panel is pressed, the effect is identical to setting the RESET bit in the host interface's IKONAS control/status register.

9.0 SCRATCHPAD/MICROCODE MEMORY MODULE

10.0 MA1024 HOST PROGRAMMING GUIDE

10.1 Introduction

This document describes how to use the microcode routines supplied by IKONAS for the MA1024 Multiplier-Accumulator to perform

- o 3-D rotation of vector end points and polygon vertices (MM33 and MM32)
- o clipping assist (MM33C)
- o perspective division (MM33P and MM32P, not available on all systems)

It also describes how to use custom microcode routines for the MA1024. It does not tell how to write microcode for the MA1024. For an explanation of how to write your own microcode, see the MA1024 Programming Guide, available from IKONAS.

Note that the IKONAS routine MMEOL is no longer supported. It must be replaced by the new routines listed above.

The steps in using MA1024 microcode routines are as follows:

- o Load the microprogram into program memory using IKLOD. For a description of the steps to assemble these programs, see "Software Installation" in this manual.
- o Load the coefficient list or matrix into coefficient memory, located at addresses 20400$0-20400$1777. The MA1024 can hold up to 64 coefficient lists or matrices; each is composed of 16 consecutive 16-bit words. Their format and contents are described later in this chapter.

  Computation of the coefficient matrix is the responsibility of the user. Helpful formulas can be found in Rogers & Adams, Mathematical Elements

for Computer Graphics, pp. 47-58, 54-55, 207-208; and Newman & Sproull, Principles of Interactive Computer Graphics, pp. 333-352.

- Load the points to be transformed into SR8 memory, located at addresses 20200$0-20277$1777. Section 2 defines the data format for input points.

- Load the MA1024 control registers to contain the program address, coefficient address, and input/output list offsets. See section 6 for a further description of these registers.

- Start the MA1024. Use a function code of 25 to store a 0 into address 20402$7.

10.2 Data Points

Input data points can be points in three-space [x y z] or points in homogeneous coordinates [x y z w]. IKONAS-supplied routines expect [x y z] coordinates.

Each input point requires two consecutive 32-bit IKONAS data words, in the following format:

For [x y z] points:

First word:

Bits	0-15	x coordinate
	16-31	y coordinate

Second word:

Bits	0-15	z coordinate
	16-31	tag field (bit 30 = end-of-list bit, set in the last point to be rotated)

For [x y z w] points:

First word:

Bits	0-15	x coordinate
	16-31	y coordinate

Second word:

Bits	0-15	z coordinate
	16-31	w coordinate

Each coordinate is a 16-bit two's-complement fraction, ranging in value from -1.0 to +0.9999.... The most significant bit (bit 15) is the sign bit; the implied binary point lies between bit 14 and bit 15. These fractional values correspond to 32,768 points to the left, and 32,767 points to the right of a central axis (at 0).

For [x y z] coordinates, the second data word contains a tag field. This 16-bit field can be used for

such information as pixel shade value; it is passed, unchanged, with the point to the output.

For IKONAS-supplied routines, bit 30 must be set for end-of-list.

Output points have the same basic format, with these exceptions:

- MM32 and MM32P routines: These routines have the same input but produce [x' y'] coordinates as output, using only one 32-bit word per point. Note that the tag field information is lost in these cases, except for the last point in the list, whose tag field is copied after the x'y' coordinates, so that the end-of-list bit will not be lost.

- MM33C - Clipping assist: This routine uses bits 24-29 of the second word for each point to hold clipping flags. See the format below:

First word:

Bits	0-15	x coordinate
	16-31	y coordinate

Second word:

Bits	0-15	z coordinate
	16-23	tag field
	24-29	clipping flags
	30	end-of-list bit, set in the last point to be rotated

10.2.1 Conversion to Pixel Coordinates

The output coordinates are in the range -1.0 to +0.9999.... To convert them to pixel coordinates, the coefficient matrix must perform the following calculations:

- multiply each coordinate by 1000 octal (for HIRES coordinates) or 400 octal (for LORES), to shift the number 6 or 7 places to the right.

  The user must mentally move the implied binary point to the right of the low-order bit. The result is a value in the range -512 to +511 (HIRES) or -256 to +255 (LORES).

- add 512 (for HIRES) or 256 (for LORES) to the results, to produce pixel coordinates in the range 0-1023 (HIRES) or 0-511 (LORES). These pixel values will be stored in the 10 low-order bits of each coordinate field.

  NOTE: the numbers given above may need to be adjusted for y-range values greater than 511, or for DR256 or GM256 boards. See "Programming the Framebuffer Memory," in this manual, for address ranges.

10.3 Rotation and Translation

There are two IKONAS-supplied routines for rotation: MM33 takes input points [x y z] and produces output points [x' y' z'].

MM32 (the "short form") takes input points [x y z] and produces output points [x' y']. MM32 needs only one word, rather than two, to store each output point; this routine can be used to save space when the z' coordinate is not necessary.

10.3.1 Coefficient Matrix

The coefficient matrix for MM33 and MM32 is defined as

$$[x'\ y'\ z'\ 1] = [x\ y\ z\ 1] \begin{bmatrix} Cxx' & Cxy' & Cxz' \\ Cyx' & Cyy' & Cyz' \\ Czx' & Czy' & Czz' \\ Tx & Ty & Tz \end{bmatrix}$$

where

[x' y' z'] is the output point;

[x y z] is the input point;

Cij' is the contribution of input coordinate i to output coordinate j';

Ta is the amount of translation (lateral movement) in coordinate a.

This coefficient matrix has the format:

Word	0	Cxx'
	1	Cyx'
	2	Czx'
	3	Tx
	4	Cxy'
	5	Cyy'
	6	Czy'
	7	Ty
	8	Cxz'
	9	Cyz'
	10	Czz'
	11	Tz

10.4 Clipping Assist

The clipping assist routine converts the input list to clipping coordinates while it performs rotation and translation. Clipping coordinates are coordinates in the range -1.0 to +0.9999..., with flags set for points which are outside any of 6 clipping planes. After the clipping flags have been set, the points should be sent to the BPS to replace or discard clipped vectors, before perspective division is performed.

The IKONAS routine for clipping assist is MM33C. Rotation and translation are performed just as described in section 3. Conversion to clipping coordinates consists of

- calculating w, the scale factor for the [x y z] coordinates. If the transformation is to simulate the view of a telescopic lens (i.e., the object is to appear close to the observer), the value of w should be less than the value of z. If the transformation is to simulate a wide angle lens (i.e., the object is to appear far away), x and y should be scaled down instead. It is impossible to scale up using the MA1024. Since x and y will be divided by w later, the effect will be the same as scaling up w;

- flagging each point for which the x or y coordinate is greater than w in absolute value;

- flagging each point for which the z coordinate does not fall between 0 and the zrange. The control registers hold the zrange, the distance from the near clipping plane (hither plane) to the far clipping plane (yon plane).

These steps define a viewing pyramid in which

$$-w < x < w$$
$$-w < y < w$$
$$0 < z < \text{zrange}$$

All points within this viewing pyramid are visible; all points outside it are off the screen and must be clipped. For each output point, bits 24-29 of the second word hold the clipping flags, as follows:

```
bit 24 is set if x is less than -w
    25 is set if x is greater than w
    26 is set if y is less than -w
    27 is set if y is greater than w
    28 is set if z is less than zero
    29 is set if z is greater than zrange
```

If w is negative, these comparisons are reversed--e.g., bit 24 is set if x is greater than -w.

10.4.1 Coefficient Matrix

The coefficient matrix for rotation with clipping assist is as follows:

$$[x'\ y'\ z'\ 1] = [x\ y\ z\ 1] \begin{bmatrix} Cxx' & Cxy' & Cxz' & Cxw \\ Cyx' & Cyy' & Cyz' & Cyw \\ Czx' & Czy' & Czz' & Czw \\ Tx & Ty & Tz & k \end{bmatrix}$$

where

[x' y' z'] is the output point;

[x y z] is the input point;

Cij is the contribution of input coordinate i to output coordinate j;

Ta is the amount of translation (lateral movement) in coordinate a, with this exception:

Tz contains the amount of translation in coordinate z, MINUS the distance from the observer to the near clipping plane;

k is a constant.

Computation of rotation and translation coefficients is the responsibility of the user. For standard usage, the computation of w depends on the viewing angle, as follows:

<u>For a viewing angle</u> <= <u>90 degrees</u>:

Cxw: Cxz * tan (B/2)
Cyw: Cyz * tan (B/2)
Czw: Czz * tan (B/2)
k: Tz * tan (B/2)

where

B is the viewing angle;

tan (B/2) is the tangent of 1/2 the viewing angle;

Tz is the amount of translation in z WITHOUT adjusting for the distance from the observer to the near clipping plane.

For a viewing angle > 90 degrees, x and y would be scaled down, as follows:

Cxx:	1/tan (B/2)	(to scale down x)
Cyy:	1/tan (B/2)	(to scale down y)
Cxw:	Cxz	
Cyw:	Cyz	
Czw:	Czz	
k:	Tz	

The coefficient matrix has the format:

Word	0	Cxx'
	1	Cyx'
	2	Czx'
	3	Tx
	4	Cxy'
	5	Cyy'
	6	Czy'
	7	Ty
	8	Cxz'
	9	Cyz'
	10	Czz'
	11	Tz
	12	Cxw
	13	Cyw
	14	Czw
	15	k

10.5 Perspective Division

After the BPS has handled the clipped points, they can be sent back to the MA1024 for perspective division--division of the x′ and y′ coordinates by w.

The IKONAS routines which perform perspective division are MM33P and MM32P. Given input points [x′ y′ z′], MM33P produces output points [x′′ y′′ z′′] and MM32P produces output points [x′′ y′′]. MM32P uses only one word to store each output point; this routine can be used to save space when the z′′ coordinate is not needed.

Perspective division uses a coefficient list to:

- calculate w, the perspective coordinate. Since w is not stored with the point, it must be recalculated from z′. Any points with coordinate values greater than w must have been clipped by the BPS before perspective division;

- divide x′ and y′ by w;

- convert the results to pixel coordinates.

10.5.1 Coefficient List

The coefficient list for perspective division has the format:

Word 0	Cx'w
1	Cy'w
2	Cz'w
3	C1w
4	Xscale
5	2 * Xt
6	Yscale
7	2 * Yt
8	Zscale
9	2 * Zt

The routine MM33P* uses this coefficient list to produce

$$x'' = \frac{x'}{w} * Xscale + Xt$$

$$y'' = \frac{y'}{w} * Yscale + Yt$$

$$z'' = z' * Zscale + Zt$$

where w is calculated as

$$(x' * Cx'w) + (y' * Cy'w) + (z' * Cz'w) + C1w$$

(MM32P uses the same input and coefficient list but produces [x'' y''] coordinates only.)

Cx'w, Cy'w

For perspective division using a single vanishing point, the values of x' and y' do not contribute to the calculation of w. Therefore, Cx'w and Cy'w should contain zeroes. The value of w will be based on z' plus the distance to the near clipping plane--since z was originally adjusted by subtracting this

---

*As of version 2.1.

distance

Cz'w, Clw

The contents of Cz'w and Clw depend on the viewing angle, as follows:

For a viewing angle <= 90 degrees:

Cz'w: tan (B/2)
Clw: ncp * tan (B/2)

For a viewing angle > 90 degrees:

Cz'w 1
Clw ncp

where

B is the viewing angle;

tan (B/2) is the tangent of 1/2 the viewing angle;

ncp is the distance to the near clipping plane (previously subtracted from z).

Xscale, (Yscale), Zscale

To convert the results of perspective division to pixel coordinates, Xscale and Yscale should contain the values 1000 octal for HIRES, or 400 octal for LORES, to shift the number 6 or 7 places to the right, as well as any other scaling factor. The result is a value in the range -511 to +511 (HIRES) or -255 to +255 (LORES).

Xt, Yt, Zt

These values are added to the results of the scaling and perspective division. Because the routine divides Xt, Yt, and Zt by 2, these coefficients must contain twice the value needed.

To complete the conversion to pixel coordinates, xt and yt should contain 1024 for HIRES (512x2) or 512 for LORES (256x2). The result will be a value in the range 0-1023 (HIRES) or 0-511 (LORES). (DR256 or GM256 boards will require different values.)

Xt and Yt may also contain a translation factor.

10.6 Control registers

The control registers for the MA1024 occupy four words in the range 20402$0 - 20402$3 (accessible from the IKONAS bus). The registers have the following contents:

Word 20402$0

Bits 0-9 program address: the offset to the MA1024 microprogram to be used.

For custom microcode, the program address should contain the same value as in the ORG statement in the IKASM assembly of the microcode

For IKONAS-supplied routines, the offsets are as follows:

MM33:	0
MM32:	300
MM33C:	100
MM33P:	200
MM32P:	400

10-12 unused.

13 w: this bit should be set to 0 for IKONAS routines. If the bit is on, the MA1024 expects input points in homogeneous coordinates [x y z w]. IKONAS-supplied routines use points in three-space [x y z].

14 normalization: this bit is set to suppress normalization during perspective division and computation of w. This bit should be off unless perspective division is being performed. It must be off when clipping assist is on (bit 15). For compatibility with the IKONAS RDS-2000, it must be off

15 clipping assist: if this bit is set, bits 24-29 of the tag field of each output point will contain the clipping flags (used with MM33C only).

16-25 coefficient address: the offset to the coefficient matrix or list to be used.

26-31 unused.

Word 20402$1
Bits 0-15 Input list offset in SR8
16-31 Output list offset in SR8

These are the offsets to the input and output areas in SR8 memory. For example, if bits 16-31 specify an offset of xxxx, the area at 20200$xxxx will be used for the output list.

This register is undefined after program execution.

Word 20402$2
Bits 0-11 Loop 0 counter: may be used by any microcode routine except MM33. This counter should be loaded with the two's complement representation of the number of times the loop is to be executed. The counter is incremented under microprogram control.

For IKONAS-supplied routines other than MM33, the loop 0 counter must be initialized and may be used in place of setting the end-of-list bit (bit 30 of the second data word). These routines automatically increment the counter as they process each point in the input list. End-of-list is assumed when the counter reaches 0 or when the end-of-list bit is set, whichever comes first.

This register is undefined after program execution.

Word 20402$3
Bits 0-15 zrange for clipping assist: the distance between the near and far clipping planes. This register needs to be set only when clipping assist is engaged.

10.7 Readback Registers

The IKONAS BUS can read from the PC Readback Register and three result registers. The result registers reflect the current value of each variable, from the last point processed:

Address	Contents
20403$0	X/Y REGISTER
Bits 0-15	x output coordinate
16-31	y output coordinate
20403$1	Z/SHADE REGISTER
Bits 0-15	z output coordinate
16-31	tag field: may contain o shade information o clipping flags (bits 24-29) o the end-of-list bit (bit 30)
20403$2	W REGISTER
Bits 0-15	current value of w
20403$3	PC READBACK REGISTER
Bits 0-9	current value of program counter (PC)
10	BUSY: this bit is set when the MA1024 is executing
11	SREOL: this bit is set when the end-of-list flag is encountered in the input list

APPENDIX G. SUMMARY OF ADDRESSES USED BY THE MA1024

20200$0 - 20200$1777	Input and output lists (SR8 memory)
20400$0 - 20400$1777	Coefficient memory (CM)
20401$0 - 20401$1777	Microprogram memory (MPM)
20402$0 - 20402$3	Control registers
20402$7	MA1024 start--writing to this address with a function code of 25 will start the MA1024.
20403$0 - 20403$2	Result registers
20403$3	PC readback register

11.0 MPC PROGRAMMING GUIDE

All addresses in this chapter are in hexadecimal unless otherwise noted. The information in this chapter describes the printed circuit board MPC, which is available with 32K, 256K, or 512K bytes of memory. For information about wire-wrapped versions of the MPC, which are no longer supported, see Appendix 2.

11.1 Introduction

The MPC 68000 Microprocessor incorporates a Motorola 68000 16-bit microprocessor with up to 512K bytes of memory. It is often used to relieve the host of computational tasks. It can also serve as a building block for workstation and standalone configurations.

Along with the optional Peripheral Control Panel (PCP), the MPC provides an interface to interactive devices such as a data tablet, trackball, three-axis joystick, terminal, lighted function switches (button box), and control dials. Software for the MPC includes a multi-tasking, real-time monitor that contains interactive routines and provisions for user-written routines.

11.2 Memory

MPC memory currently consists of up to 512K bytes of RAM and 8K bytes of ROM.

11.2.1 ROM

The standard amount of ROM in the system is 8K bytes, addressed from 0000 to 1FFF. This ROM contains the power-up vectors and enough code to start the MPC. Addresses 2000-3FFF are reserved for possible future expansion.

11.2.2 RAM

Memory sizes of 32K, 256K, and 512K bytes are available.

Like program ROM, data RAM is addressed starting at location 0. When data RAM and program ROM are at the same addresses, the RAM is "data only" (no execution allowed). For an MPC containing 8K bytes of ROM, this will then affect the first 8K bytes of RAM. RAM that does not have the same address as ROM can be used for data or instruction storage.

11.2.3 Order of Bytes

The 68000 is a word-oriented machine. All memories are 16 bits wide. Sixteen-bit items must be aligned on a 16-bit boundary; i.e., the byte address must be even. Within a 16-bit word, the EVEN byte is the high-order byte, and the ODD byte is the low-order byte.

11.3 I/O Ports

Because 68000 I/O ports are mapped to the memory space, they are accessed with the same instructions that access memory. I/O ports defined for the 68000, and their addresses, are listed in Figure 11-1 below.

Device	MPC Address (in hexadecimal)
IK bus	800000-BFFFFF
IK address translator	C00000-FFFFFF
*PCP	4FFC80-4FFCFF
Serial I/O ports	4FFD00-4FFD0F
Programmable Timer	4FFD30-4FFD3E
*Memory Management Unit (MMU)	4FFD80-4FFD8F
Host interrupt	4FFDE00

Figure 11-1. I/O ports defined for the 68000. Starred ports (*) are readable from the IK bus.

Each of these ports is discussed in the sections following. The VERSAbus port, which does not fall into the same category as the I/O ports described above, is discussed later in the chapter.

11.3.1 IK Bus Access

The upper half of the MPC address space (addresses 800000-FFFFFF) is used for IK bus references. The IK bus is mapped into the address space of the MPC through 512 windows, each 8192 bytes long. Each window corresponds to a contiguous range of 2048 32-bit words on a 2048-word boundary.

When you perform an IK bus operation from the MPC, the IK bus address must first pass through a user-defined translation control table. This table can hold a maximum of 512 entries, or translation control blocks, one for each window. Each block contains the Y address bits of the IK bus address, the function code, the sender id, and MPC control information. The X address bits do not pass through the translation table. Thus, a window consists of a single Y address and a range of X addresses.

11.3.1.1 How to Set Up the Translation Table

Each translation control block is located at an address 400000 greater than that of the corresponding mapping window. The address of a translation control block is given by

```
bit 23                          bit 0
    11sssssssss0000000000000
```

where

bits 22-23 are always set, and

bits 13-21 (sssssssss) contain the window id--the number of the mapping window to be used.

The translation control block at such an address will control references to the corresponding window, covering the address range

10sssssssss0000000000000-10sssssssss1111111111111

Each translation control block occupies four 16-bit words. The first two words contain user-defined information. The last two words are used by the control program to manage window allocation and may not be modified by user programs. The user-defined information is as follows, in terms of a 32-bit IK word:

Bits	30-31	Must be zero
	16-29	Y address bits 0-13 (bits 10-23 in a standard IK bus address)
	12-15	No function - reserved for future use. Unpredictable when read.
	8-11	Write trigger mask
	7	External window select
	4-6	IK bus sender id
	0-3	IK bus function code

Y address bits - the y component of the IK bus address.

Write trigger mask - This four-bit field determines the time for the start of the IK bus transfer. Each bit of this field corresponds to one byte in a 32-bit IK word. When a bit is set, a store to the corresponding byte triggers an IK bus write. For example, if the low-order bit in this field (bit 8) is set, then an IK bus write begins when byte 0 is accessed during the write. For most cases involving simple 32-bit transfers, only

this bit needs to be set.

External window select - This bit should be set for all IK bus accesses.

It should have a value of zero if the window is not to be mapped to the IK bus. In this case, no IK bus operation will be performed; some external peripheral device must respond to the address.

IK bus sender id - the sender id (in the range 0..7) to be used for write operations to image memory (low-order bit = bit 4).

IK bus functionbits - the function code for the operation (low-order bit = bit 0). (Note: the read/write bit is not included; it is deduced by the 68000 from the operation specified in the user program.)

11.3.1.2 How to Specify the IK Bus Address

Within a program running on the MPC, IK bus addresses are specified in the following manner:

```
bit 23                          bit 0
    10sssssssssxxxxxxxxxxxbb
```

where

bits 22-23 always contain 10;

sssssssss is the window id;

xxxxxxxxxxx contains the X address bits for the IK bus address;

bb is the byte being written to. This field is important only when the write trigger mask is set in the translation control table entry. For all reads and all simple 32-bit writes, these bits should be zero. The 68000 MOVE.L instruction increments these bits automatically.

Note that in a translation control block, the y address field is 14 bits long, and x addresses for image memory can be up to 11 bits long. The most significant bit (msb) of the x field and the least significant bit (lsb) of the y field overlap when you are accessing non-image memory. They are decoded properly only if the lsb of the y field is cleared.

When accessing non-image memory, you can use the full 2048-word range of each window. In setting up the translation table, set the Y address bits to fall on a 2048-word boundary (Y address bit 0 clear). Then use all 11 X address bits when necessary. (X-address bit 11, when set, will become the low-order Y bit.) You can then access the full address space of an SR8, for example, using only four windows. If you use consecutive windows, you can access the entire SR8 without having to consider which window is needed--incrementing the address changes the window transparently.

When you are accessing image memory, the overlap of the msb of the X field and the lsb of the Y field is not important. In this case the addresses are decoded differently, because DR256/GM256 memories need the full 11 bits of both X and Y. For image memory access, the decoding circuits:

- examine both the high-order Y address bit and the IK function code in the relevant translation control block entry, to determine whether the access is LORES, HIRES, or WORD mode.

- shift LORES addresses left one bit (neither HIRES nor WORD mode addresses are shifted);

- place the msb of both the X and y fields onto the proper address lines.

For DR64/GM64 boards, a window can be used for 512 pixels in LORES mode and 1024 pixels in HIRES mode. These pixels must all have the same Y address. In WORD mode, each window will contain 1024 addresses, with each window on a 1024-word boundary (Y bit 0 translates to IK address bit 10).

For DR256/GM256 boards, a window can be used for 1024 pixels in LORES mode and 2048 pixels in HIRES mode. These pixels must all have the same Y address. In WORD mode, each window will contain 1024 addresses, with each window on a 1024-word boundary (Y bit 0 translates to IK address bit 10).

11.3.1.3 Correspondence of 68000 Words to IK Words

IK words are 32 bits long, so two 68000 words make up one IK word. 68000 words are assigned to IK words in the following way: each pair of 68000 words having the same value in address bits 2-23 makes up one IK word. Of each pair, the word with address bit 1 = 1 becomes the less significant, corresponding to IK bits 0-15; the word with address bit 1 = 0 becomes the more significant, corresponding to IK bits 16-31.

11.3.1.4 When a 3000 operation is started

A 68000 read cycle to the Adage 3000 always starts a 3000 read. A write to any byte whose write trigger mask is set will start a IK bus write. For read operations, the unused part of the IK word is ignored. For write operations, 32 bits are always written. There is a separate 32-bit buffer for each window for writes to the 3000. Each byte in the IK word is whatever was last stored in that part of the buffer by a previous write from the 68000.

If you want to write fewer than 32 bits to the IK bus, you may be able to get a speed improvement by writing only to the odd address of the 3000 word. You can start a 3000 operation with only one 68000 cycle by doing this. Note, however, that all 32 bits will be written to the IK bus, using whatever valid data were last stored into the upper bits.

11.3.2 PCP Port

The PCP has 64 addresses internal to it. These addresses are mapped into 68000 addresses 4FFC80 - 4FFCFE. PCP addresses are accessed in the same manner as MPC addresses. For PCP address assignments, see Chapter 12, "PCP Programming Guide."

11.3.3 Serial I/O Ports

The MPC provides four RS232 serial ports into which terminals, data tablets, and other peripheral devices can be plugged. The baud rate for each serial port must be set by software.

Each serial port is an MC6850 ACIA chip. For programming specifications of this port, consult the MC6850 data sheets.

The serial ports occupy odd addresses in the range 4FFD01-4FFD0F, and the baud rate generators are in the corresponding even addresses, as follows:

Port number	Control/ status reg address	Data reg address	Baud rate address
0	4FFD01	4FFD03	4FFD02
1	4FFD05	4FFD07	4FFD06
2	4FFD09	4FFD0B	4FFD0A
3	4FFD0D	4FFD0F	4FFD0E

The 68000 must write to the baud rate generator for a serial port before the port will start running. If the baud rate is to be set by software, the following data bits should be used (assuming the 6850 is set to use a frequency of 16xbaud rate, which is normal):

Baud rate	Data value	Baud rate	Data value	Baud rate	Data value	Baud rate	Data value
50	0F	150	0B	1800	07	4800	03
75	0E	300	0A	2000	06	7200	02
110	0D	600	09	2400	05	9600	01
134	0C	1200	08	3600	04	19200	00

11.3.3.1 Serial port interrupts

The serial ports share interrupt level 2, vector address $68. When an interrupt at level 2 occurs, the serial ports must be polled to determine which one generated the interrupt. A serial port will not generate interrupts unless it has been programmed to interrupt (see the 6850 programming guide from Motorola). The interrupt routine must service the interrupting device to ensure that the interrupt is not taken repeatedly.

11.3.3.2 Serial port Programming Sequence

Following is a description of the serial port programming sequence for setup, reads, and writes:

1. Setup

    1. Store control register=03 to reset. See Figure 11-2 for control register bit definitions.

    2. Set character length, receiver interrupt enable (example: control reg=AD for 7 bits, odd parity, one stop bit, all interrupts enabled).

    3. Set baud rate (example: baud rate generator=01 for 9600 baud).

2. Read sequence

    1. Interrupt to level 2 (serial port interrupt level).

    2. Read status registers to determine source of interrupt--detect a received-character interrupt. See Figure 11-3 for status register bit definitions.

    3. Read the character from the data register. This clears the interrupt request

    4. Process the character.

5. Wait for interrupt.

3. Write sequence

1. Write character to data register and enable transmitter interrupt.

2. Wait for interrupt.

3. Interrupt to level 2--decode and detect transmitter interrupt.

4. If another character is ready to send, write it and wait; else disable transmitter interrupt
Note: the transmitter interrupt is set whenever the transmitter buffer is empty. In order to continue processing, disable the interrupt after the string is finished.

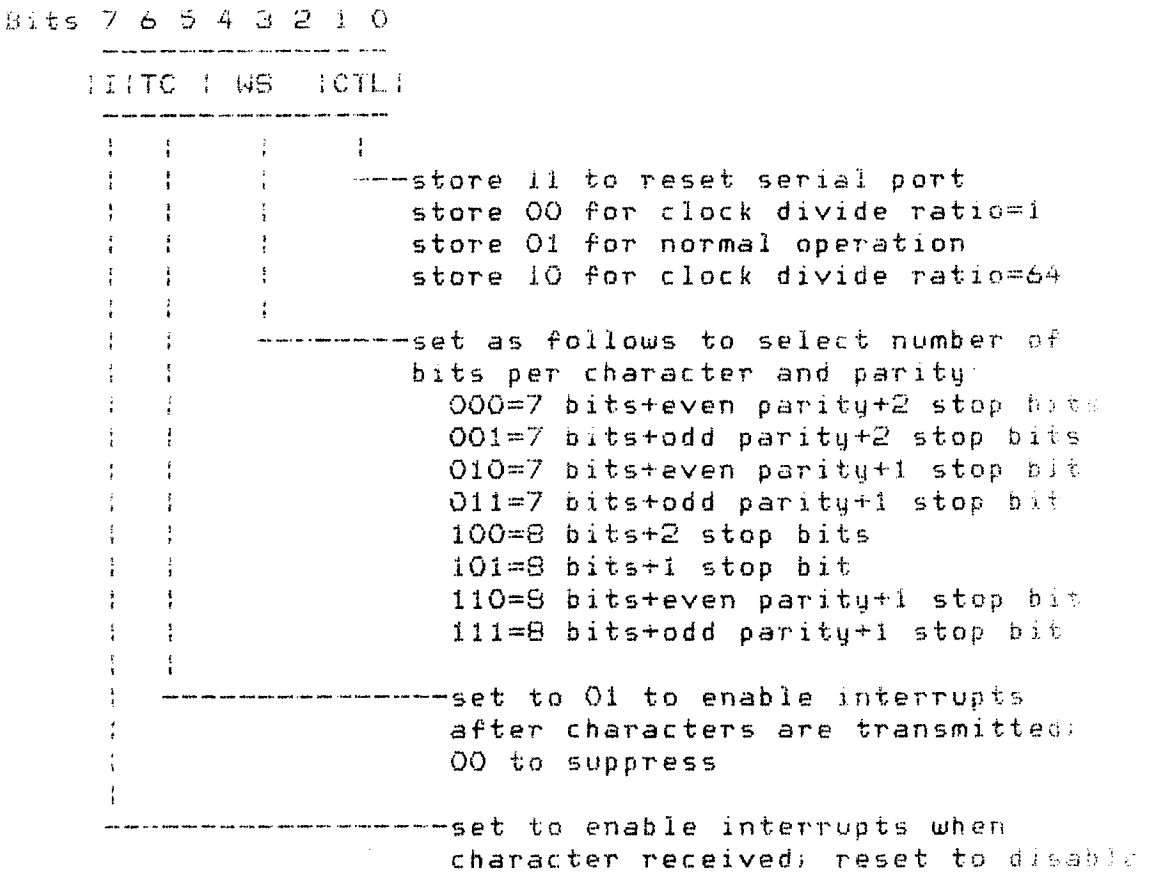


Figure 11-2. Serial port control register bit definitions.

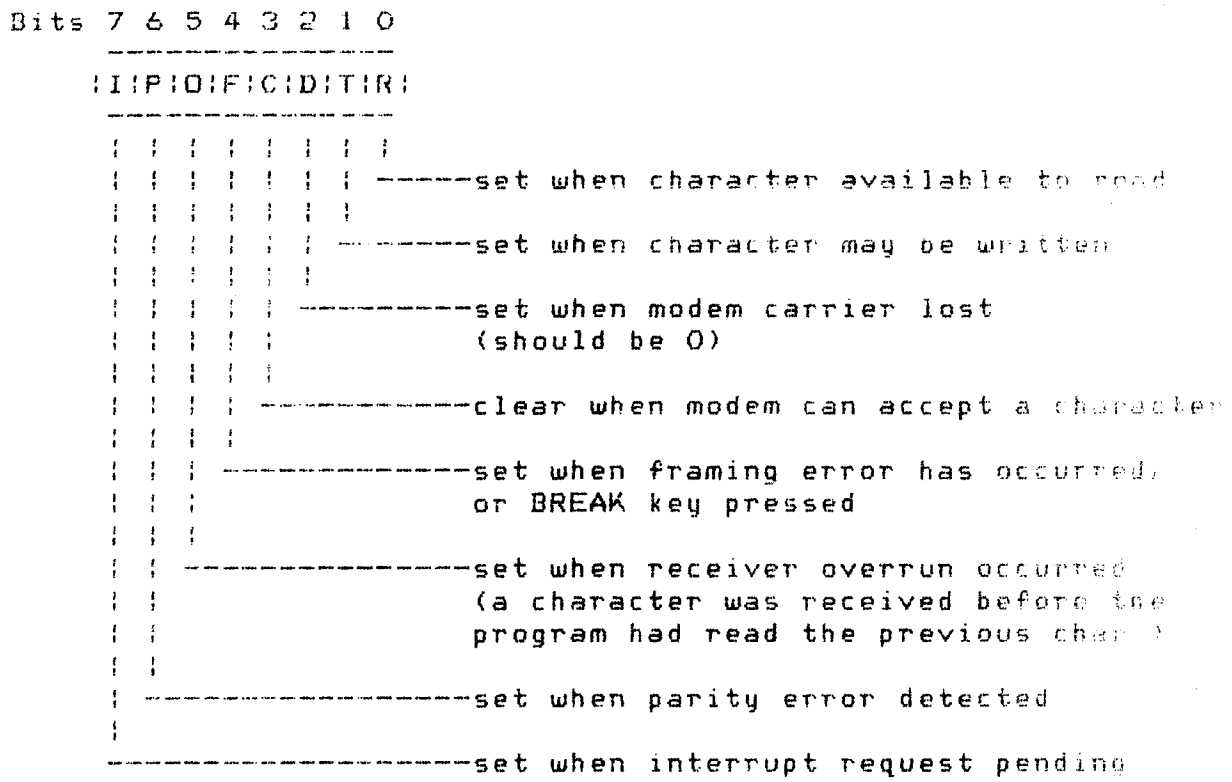


Figure 11-3. Serial port status register bit definitions.

11.3.4 Programmable Timer

The MPC includes a Motorola MC6840 Programmable Timer. The timer occupies odd addresses in the range 4FFD31-4FFD3F. For instructions on programming the timer, consult the Motorola data sheets for the MC6840. The 6840 should be used as a timer; all the G inputs are unused. The internal timers may be cascaded if that is necessary; in that case, timer 1 varies the fastest and timer 3 varies the slowest. Cascading is performed by enabling the C inputs of timers 2 and 3.

If the 6840 is programmed to allow interrupts at end of count, the interrupt will be taken to level 5, vector address $74. The interrupt routine must reset the 6840 to ensure that the interrupt is not taken repeatedly.

11.3.4.1 Example of Timer Use

The following example illustrates use of the programmable timer.

Setting a real-time clock or interval timer

Problem: to create a timer interrupt every 16 milliseconds (60 Hz interrupt rate) using timer number 1.

1. Select control register 1 by storing 01 at 4FFD33 (control register2).

2. Set the timeout value: 16000 (the number of 1-microsecond clock ticks in 1 milliseconds), stored into the timer 1 value, locations 4FFD35 (upper byte) and 4FFD37 (lower byte).

3. Set the timer mode by storing C2 into control register 1, location 4FFD31. Bit 0 of each timer control register has special meaning; for example, bit 0 in the control register for timer 1 is a master-reset bit; so if you wanted to set a different timer, the setting of bit 0 in the timer control register would depend on the timer being set.

4. The timer is started. Wait for an interrupt.

5. When the timer expires, interrupt level 3 will be entered.

6. The interrupt routine should read the timer status register (4FFD33) to determine which timer interrupted. The three low-order bits of the status register indicate which timers have interrupted. bit 0 on=timer 1; bit 1=timer 2; bit 2=timer 3.

7. For each timer which has interrupted, the interrupt routine should read the counter value. Reading the counter value clears the interrupt from that counter.

8. The interrupt routine performs its processing.

9. If a continuous real-time clock is needed, the interrupt routine exits; the timer will interrupt again when the time expires. If a single interval is needed, the interrupt should be disabled by storing 0 into control register 1.

11.3.5 Memory Management Unit Port

The MPC supports full memory mapping and protection using the Motorola MC68451 Memory Management Unit (MMU). The MMU occupies addresses 4FFD80-4FFDBF. Consult the MC68451 data sheets for instructions on how to set up the MMU.

When the MPC is reset (either by power-up, IK-bus reset, or the RESET instruction), the MMU is initialized to a null state which allows addresses to pass through unchanged, as described in the MC68451 programming manual

Accesses from the IK bus to the MPC address space are translated by the MMU before being accepted. Accesses from the IK bus are distinguished by the assertion of function code bit 3 on the MMU. The system software may protect memory from IK bus access or modification by proper setting of the MMU.

If the MMU generates an interrupt (it must be programmed to do so), the interrupt will be to level 6, interrupt vector $78, regardless of the vector address set in the MMU. This is the highest-level interrupt used in the MPC. The interrupt routine must remove the interrupt request in the MMU to avoid taking the interrupt repeatedly.

11.3.6 Adage 3000 Host Interrupt Port

Any store to the word at 4FFE00 will cause the host interface to interrupt the host. This feature may be used for communication between the MPC and the host computer.

11.3.7 LED Registers

There are nine light-emitting diodes (LED's) at the top of MPC boards (printed-circuit-board versions only). These are labeled LED 0-6, LED S, and LED D.

LED's 0 through 6 are controlled by setting bits 0-6 at address 4FFE80. When a bit is set, the corresponding LED is off.

Register bits 6 and 7 have these additional functions:

- Bit 6, when set, causes every write to MPC memory to have bad parity. (Using this bit is not recommended.)

- Bit 7 has no associated LED, but when set it prevents the IK bus program reset from affecting the MPC.

LED S and LED D are not user-alterable. LED S reflects the state of 68000 function bit 2; LED D reflects the state of 68000 function bit 0.

11.3.8 VERSAbus Port

The VERSAbus port is not a port to which I/O is performed explicitly, but accesses to addresses which appear on the VERSAbus will cause the VERSAbus to respond. The addresses used by the VERSAbus are determined by the cards in your VERSAbus; but the VERSAbus must not overlap addresses used within the MPC.

When a VERSAbus device is performing a DMA operation, it has direct access to the physical address space of the MPC; that is, it bypasses the MMU. VERSAbus devices may access the MPC memory and I/O space, including the IK bus, and other VERSAbus devices.

VERSAbus devices whose addresses fall in the range C00000-FFFFFF have a potential address conflict with the IK bus segment space. The conflict can be averted by setting bit 11 of the appropriate translation register to deactivate the segment for IK bus accesses.

11.4 Accessing 68000 Memory from the IK Bus

When the IK bus accesses 68000 memory, only the lower 16 bits of the IK data bus are used. IK data bits 0-15 map to 68000 memory bits 0-15.

11.4.0.1 Accessing Data RAM

The data RAM on the MPC may be accessed from the IK bus at the following locations:

with 32K bytes of RAM:	34000$0-34037$1777 (octal)
with 256K bytes of RAM:	34000$0-34377$1777
with 512K bytes of RAM:	34000$0-34777$1777

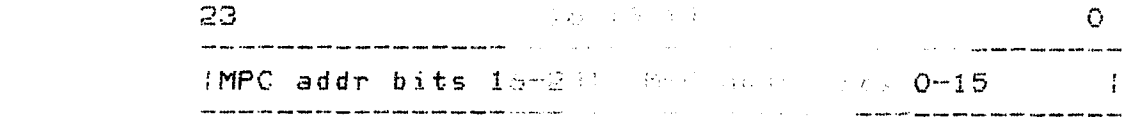
IK bus accesses pass through the MMU for translation IK address bit 0 is hard-wired to MPC address bit 1. Therefore, to access the MPC from some other device on the IK bus, you must divide the internal MPC address by two and OR in the card address of 340 octal.

11.4.0.2 Accessing I/O Space

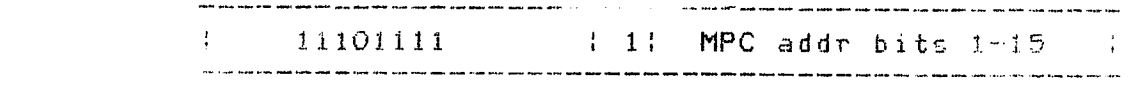
The I/O space of the MPC can be accessed from the IK bus at locations 35777$0 to 35777$1777 (octal).

To form the IK bus address, you must shift the lower 16 bits of the internal MPC address right one bit, shifting a one into the high-order bit position. Place the upper 8 bits of the internal MPC address in the low octal (E7 hex).

MPC internal address:



IK bus address:



11.5 IK-bus-directed Reset and Interrupt

When the IK bus accesses the MPC with a function code of 26 octal, the MPC will be reset. This reset is indistinguishable from the power-on reset.

When the IK bus accesses the MPC with IK bus function bit 2 set (i.e. function code 24 octal for write or 4 octal for read), the requested operation will be performed and an interrupt request to level 3, interrupt vector $6C will be raised. The interrupt request will be removed when the interrupt is serviced. This IK-bus-directed interrupt may be used for communication between the host or the BPS32 and the MPC. A region of storage may be set aside for communication, and when an interrupt code is stored there (using function code 24 octal), the interrupt will be raised.

11.6 Interrupts and Exceptions

Interrupt levels are assigned as follows:

Level	Vector address	Use
1	$64	PCP interrupts
2	$68	Serial port interrupts
3	$6C	IK-bus-generated interrupt
4	$70	Timer interrupts
5	$74	Frame interrupts
6	$78	MMU interrupts

For the MPC, interrupts from devices on the VERSAbus will be taken to the interrupt vectors specified by the devices. Internal interrupts have priority over VERSAbus interrupts, regardless of interrupt levels.

Accesses to non-existent memory result in a bus error and an interrupt to the bus-error interrupt point, vector address $8.

11.7 Power On

Power on follows 68000 protocol: the SP is fetched from location 0 in the ROM and the PC is fetched from location 4 in the ROM. Control then passes to the address in the PC.

APPENDIX 1. MPC MEMORY MAP

(All addresses in hex)

	15 8	7 0
800000-FFFFFF	1k bus address space and address translation table	
4FFE02-7FFFFF	reserved for VERSAbus devices	
4FFE00-4FFE01	reserved	host interrupt
4FFDC0-4FFDFF	reserved for VERSAbus devices	
4FFD80-4FFDBF	MMU translation & control registers	
4FFD40-4FFD7F	reserved for VERSAbus devices	
4FFD3C-4FFD3F		timer no. 3 value
4FFD38-4FFD3B		timer no. 2 value
4FFD34-4FFD37	even byte	timer no. 1 value
4FFD32-4FFD33	reserved	ctl reg. 2/status
4FFD30-4FFD31		ctl reg. 1 or 3
4FFD10-4FFD2F	reserved for VERSAbus devices	
4FFD0F	port 3 baud rate	port 3 xmt/rcv dta
4FFD0C	reserved	port 3 ctl/status
4FFD0B	port 2 baud rate	port 2 xmt/rcv dta
4FFD08	reserved	port 2 ctl/status
4FFD07	port 1 baud rate	port 1 xmt/rcv dta
4FFD04	reserved	port 1 ctl/status
4FFD03	port 0 baud rate	port 0 xmt/rcv dta
4FFD00	reserved	port 0 ctl/status
4FFC80-4FFCFF	PCP registers, as defined by PCP	
4FFC20-4FFC7F	reserved for VERSAbus devices	
040000-4FFBFF	reserved for VERSAbus devices	
008000-03FFFF	program or data memory	
000000-003FFF*	program ROM and data RAM-same address	

*000000-001FFF for 32K ROMs
 000000-003FFF for 64K ROMs

APPENDIX 2. WIRE-WRAPPED MPC'S

The following details should be noted concerning wire-wrapped MPC's, which are no longer supported by Adage:

The MPC32 contains 32K bytes of static RAM. It has no MMU, no programmable timer, and no VERSAbus port. It can be accessed from the IK bus at locations 34000$0 - 34037$1777 (octal).

The MPC256 contains 256K bytes of dynamic RAM. It contains a programmable timer, MMU, and a VERSAbus port. It can be accessed from the IK bus at locations 34000$0-34377$1777 (octal).

12.0 PERIPHERAL CONTROL PANEL (PCP) PROGRAMMING GUIDE

12.1 Initialization

12.1.1 Reset

Two methods of reset are available; both types affect the four serial* ports and the eight digital ports:

1. Power on Reset - comes about when power is first applied to the PCP;

2. Programmed Reset - occurs when a 0 and then 1 is written to control register 0 bit 1 (address XXXX74 octal)

*Upon receiving either reset, the serial ports must be re-initialized

12.1.2 Device Initialization

12.1.2.1 Serial Ports

The baud rate clocks are user selected via hardware straps on the PCP (as delivered all pots set for 9600 baud); however, parity/character length/stop bit length must be programmed.

Upon reset, program initialization must occur before data can be sent or received. Program initialization consists of writing first a mode control byte and then a command byte.

*****The first byte written to the command register is the mode control byte. Successive bytes are command bytes, which instruct the serial port that the mode**

control is to be changed.***

See Table 1 for the mode control word format, and Table 2 for the command word format.

The lower eight bits of data words sent and received are the bytes used for command/data. Status is read from the command/status register. See table 3 for status byte definition.

Two interrupts originate from each serial I/O port: TxRDY and RxRDY. Reading either the transmit buffer or the receive buffer (whichever is causing interrupt) will clear its respective interrupt.

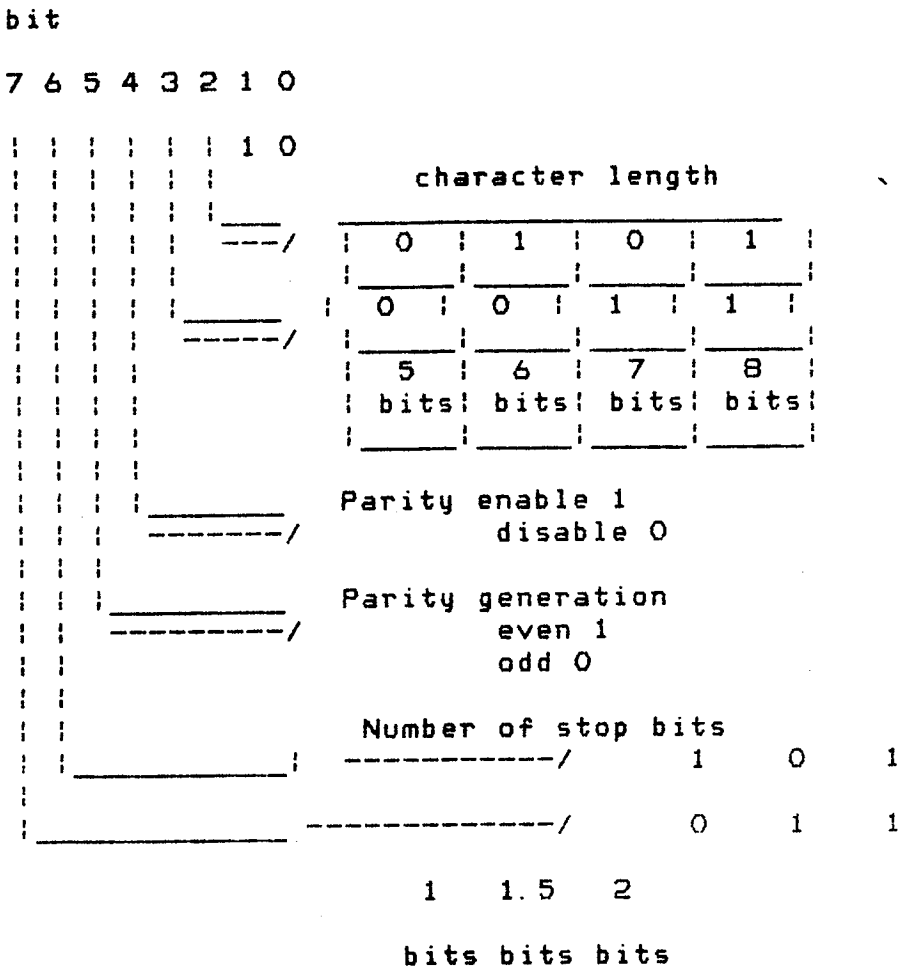


TABLE 1. USART MODE WORD

A typical mode word would then be 316 for 2 stop bits, no parity, 8 bits common length.

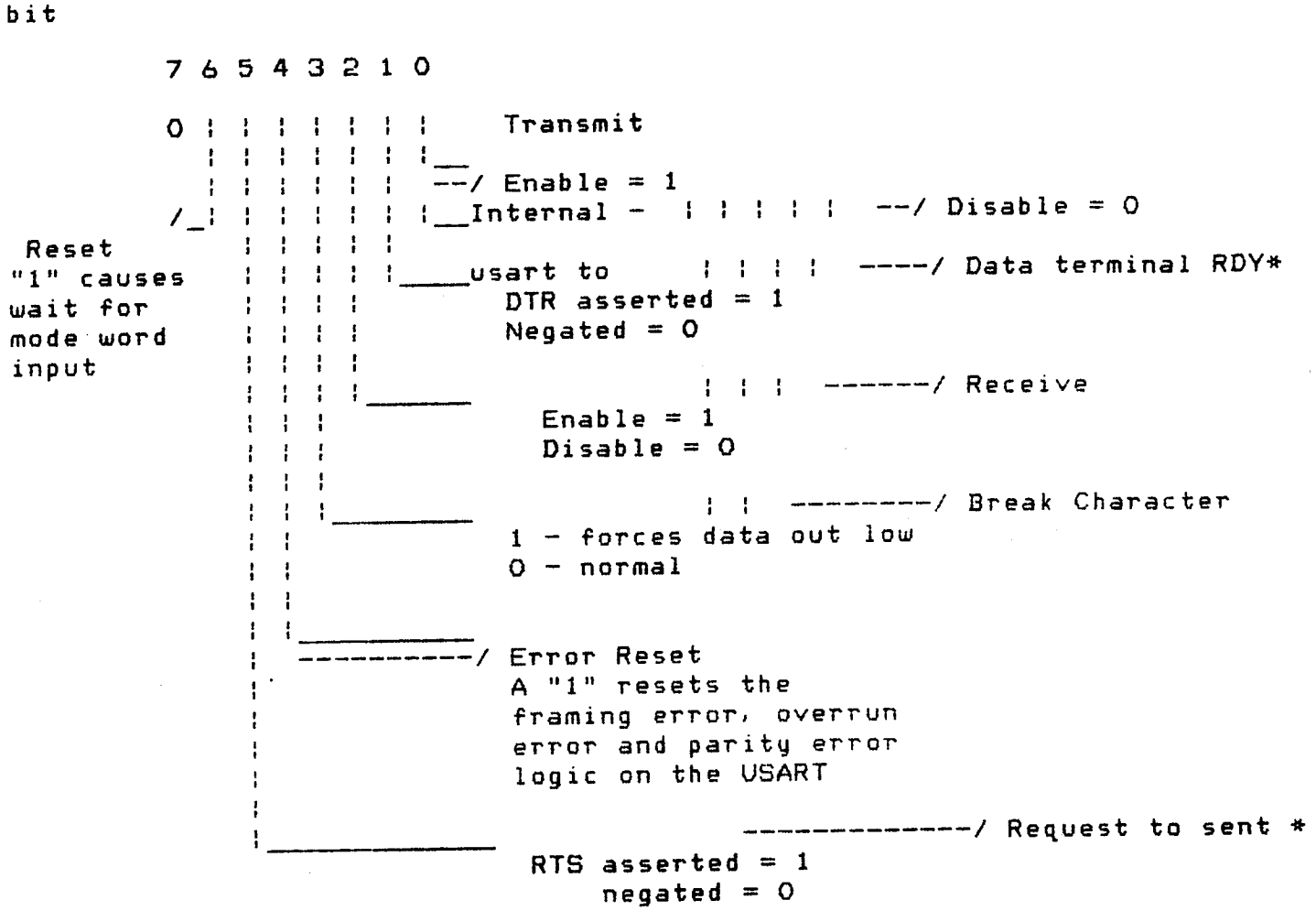


TABLE 2. COMMAND WORD USART

A typical command word would then be 5 octal.

*Typically used in modem control.

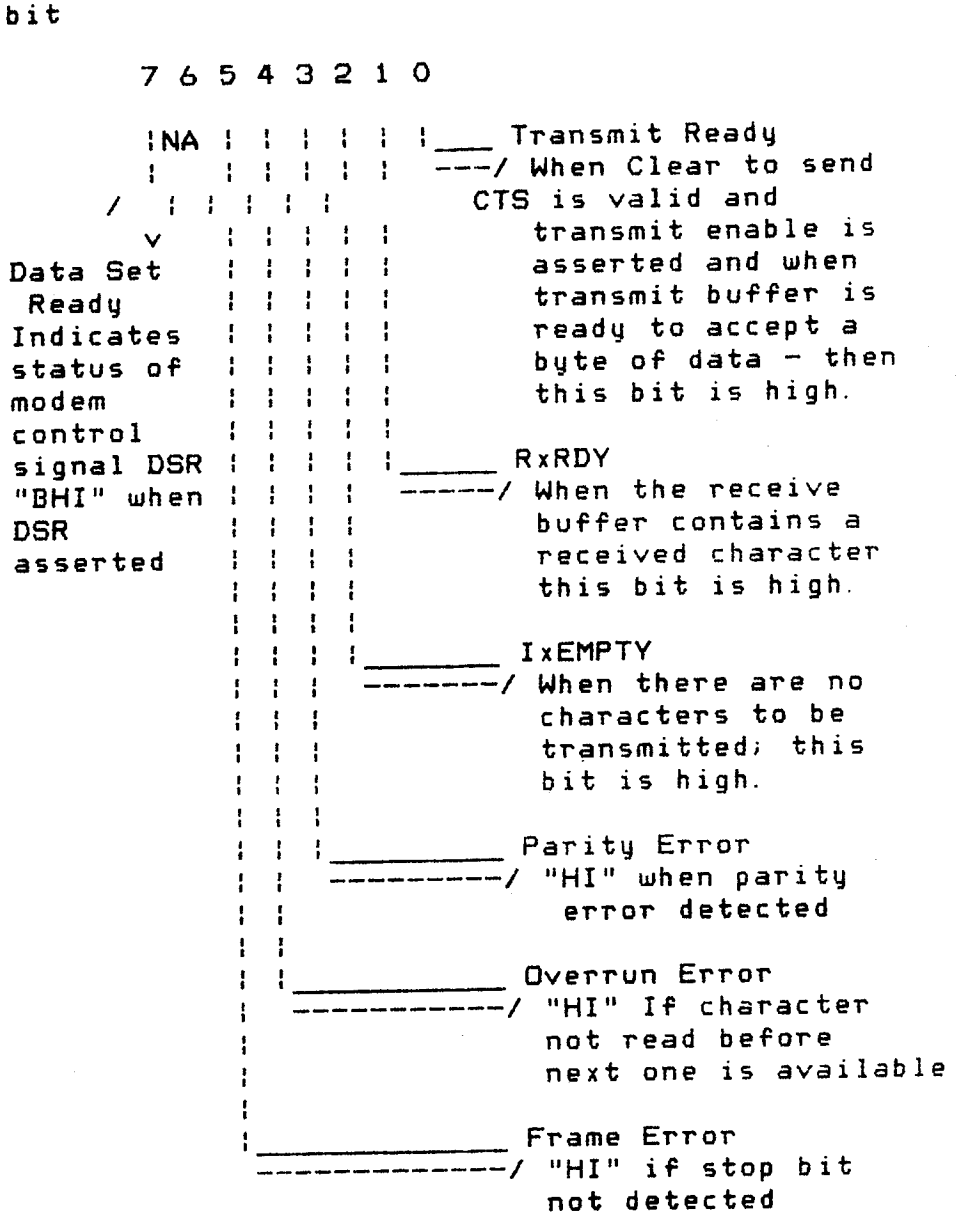


TABLE 3. USART STATUS WORD

12.1.2.2 Digital I/O Port

Eight 16-bit digital I/O ports are provided for. Initialization is dependent on the device attached. Each I/O port receives the combined power on/programmed reset pulse.

A device present signal is activated whenever a port is in use--bits 0-7 of control register 0 (address 74 octal). Asserted level for device present is a "HI".

12.1.2.3 Analog to Digital Converter

Sixteen analog channels are provided for. Each line has a range of +/- 10 volts and a resolution of 12 bits. Approximately 33 microseconds is required for a converson allowing a sample rate of 30KHz.

A conversion is started by reading the channel to be digitized. Polling bit 8 of control register 0 (address 74 octal) for a "HI" signifies an end of converson.

Reading the digitized* channel after a valid end of conversion will give the sampled value in the lower twelve bits of the returned data.

*Reading any channel will return the digitized data and start a conversion on the new channel. By this means all sixteen channels may be read in seventeen reads or any combination of channels may be read by n+1 reads.

12.2 Interrupts

12.2.1 Preparation

12.2.1.1 Serial Ports

Each serial port is the origin of two interrupts; TxRDY and RxRDY. TxRDY states that the transmit buffer is ready for another character. RxRDY states that the receive buffer has a character for the CPU.

12.2.1.2 Parallel Ports

Each parallel port can initiate an interrupt action.

12.2.2 INTERRUPT SERVICE

1. There are sixteen possible interrupt origins:

a) Four serial port receive buffer full.

b) Four serial port transmit buffer empty.

c) Eight digital I/O port device interrupts.

2. Interrupt Masking

a) Control register two (address location XXXX76 octal) contains interrupt mask.

b) Each bit in control used masks one possible interrupt line. See Table 4.

3. The host system receives one interrupt.

4. The origin of the interrupt is determined by reading a vector at control register 0 (address 74 octal). Bits 15-12 contain the vector; see Table 5 for device vectors.

The vectors are priority encoded; the vector read is the highest-priority device presenting an interrupt. Servicing an interrupt--reading from or writing to a particular device--clears that device's interrupt.

Devices may simultaneously cause interrupts; however, only the highest priority presents its vector. When the highest device is serviced (or masked), the next highest priority device presents its vector. The interrupt line is negated only when all interrupts have been serviced (or masked).

Control Register Bit	"HI" enables Interrupt		"LO" Masks Interrupt	
15	RxRDY	Serial	Port	0
14	"	"	"	1
13	"	"	"	2
12	"	"	"	3
11	TxRDY	Serial	Port	0
10	"	"	"	1
9	"	"	"	2
8	"	"	"	3
7	Digital	I/O	Port	0
6	"	"	"	1
5	"	"	"	2
4	"	"	"	3
3	"	"	"	4
2	"	"	"	5
1	"	"	"	6
0	"	"	"	7

TABLE 4. INTERRUPT MASK

Vectors Priority Encoded

Command Register bits							Priority
15	14	13	12				
0	0	0	0	digital	I/O	Port 7	0
0	0	0	1	"	"	" 6	1
0	0	1	0	"	"	" 5	2
0	0	1	1	"	"	" 4	3
0	1	0	0	"	"	" 3	4
0	1	0	1	"	"	" 2	5
0	1	1	0	"	"	" 1	6
0	1	1	1	"	"	" 0	7
1	0	0	0	TxRDY	Serial	Port 3	8
1	0	0	1	"	"	" 2	9
1	0	1	0	"	"	" 1	10
1	0	1	1	"	"	" 0	11
1	1	0	0	RxRDY	Serial	Port 3	12
1	1	0	1	"	"	" 2	13
1	1	1	0	"	"	" 1	14
1	1	1	1	"	"	" 0	15

0 = lowest priority
15 = highest priority

TABLE 5. INTERRUPT VECTORS

12.3 DATA TRANSFER

12.3.1 DIGITAL DATA TRANSFER

1. Eight 16 bit parallel I/O points available.

a) Devices may force particular ports to input only or output only configuration.

b) Each port has interrupt capabilities

c) PCP control register zero bits 0 thru 7 contain information as to whether a device is present; "HI" means a device is present.

d) A read will read data from the addressed port, and a write will write to the addressed port. See table 6 for address map.

2. Four RS232C compatable serial ports are available.

a) Baud rates user selected by hardware strapping

b) Parity, character length, and stop bit length selected by user program action.

c) Lower eight bits of data written or read are the bits used by the USART.

d) User program action should first initialize mode and command words.

e) Two interrupts possible from each port

1 - TxRDY transmit buffer can accept a character for transmission

2 - RxRDY receive buffer can accept a charactor for transmission.

f) Mode in control is completed through the command/status words. See table 6 for address mapping.

12.3.2 ANALOG DATA TRANSFER

1. Sixteen analog channels available.

2. +/-10V is range of each channel, and 12 bits accuracy (11 bits magnitude 1 bit sign) for each channel.

3. Approximately 30KHz is maximum conversion rate for the A/D channels.

4. Bit 8 of command register 0 (see table 6) is an END OF CONVERSION signal to signify when a valid sample is ready. ("HI" signifies EOC has completed)

address (octal)				
0	0	Digital	I/O Port	0
0	1	"	"	1
0	2	"	"	2
0	3	"	"	3
0	4	"	"	4
0	5	"	"	5
0	6	"	"	6
0	7	"	"	7
1	0	Serial	Port 0	Data Rd/Wr
1	1	"	" 0	Command mode/Status
1	2	"	" 1	Data Rd/Wr
1	3	"	" 1	Command mode/Status
1	4	"	" 2	Data
1	5	"	" 2	Command mode/Status
1	6	"	" 3	Data
1	7	"	" 3	Command mode/Status
2	0	Analog	Channel	0
2	1	"	"	1
2	2	"	"	2
2	3	"	"	3
2	4	"	"	4
2	5	"	"	5
2	6	"	"	6
2	7	"	"	7
3	1	"	"	8
3	1	"	"	9
3	2	"	"	10
3	3	"	"	11
3	4	"	"	12
3	5	"	"	13
3	6	"	"	14
3	7	"	"	15
7	4	Command register 0		
	Read	bit 0-7	Device Present	
		8	end of A/D conversion	
		12-15	Interrupt device Vector	
	Write	bit 1	Programmable reset	
7	6			
	Read	**bit 0-15**	**Interrupt Mask**	
	Write	**bit 0-15**	**Interrupt Mask**	

TABLE 6. ADDRESS MAP

13.0 VIDEO INPUT MODULE (VI8/VI24) PROGRAMMING GUIDE

NOTE: Bit 0=LSB, bit 31=MSB

13.1 Introduction

The VI8/VI24 Video Input Module accepts data from an RS-170A video camera and stores it into the DR64 framebuffer memory. Input data is accepted at real-time rates, sampled at a rate from 9 to 11 MHz, and written to the DR64. The sampling frequency, the size of the sampled frame, the location of the sampled frame within the input scene, and the location of the frame written to the output framebuffer may be programmed. Sampling may be continuous or may be set to stop after one frame has been digitized. The input scene may be compressed (minified) by an integer divisor, to allow storing a scene into a small section of the framebuffer.

13.1.1 The VI8

The VI8 is a one-channel, eight-bit digitizer. It accepts a single video signal and digitizes it into an eight-bit output frame.

13.1.2 The VI24

The VI24 accepts three video channels simultaneously and digitizes them into a 24-bit output frame. The input channels must be synchronized with each other.

13.2 Programming the VI8/VI24

Setting up the VI8/VI24 consists of performing the analog adjustments corresponding to your input signal and initializing the control registers.

13.2.1 Analog adjustments

The following description will assume that the VI8/VI24 is viewed from the component side with the 108-pin edge connectors facing down.

The VI8/VI24 has four adjustable potentiometers (trimmer pots) for each input channel. The trimmer pots for each channel are immediately to the left of the BNC connector for the channel. The two pots at the left are the gain controls; the two pots at the right are the offset controls. These pots may be adjusted to select a range of the input signal to be digitized.

The gain control is used to select what input voltage spread will correspond to the minimum and maximum digitized values.

The offset control is used to select what input voltage will correspond to the minimum digitized values.

These pots may be set up so that a standard one-volt signal is digitized into the full range of 256 levels; or they may be set up so that a small region of interest in the input signal is digitized into the 256 levels, with values outside that range being treated as equal to the upper or lower limit, as appropriate.

For each adjustable component (gain and offset), there are two settings: the minimum value and the departure from the minimum. The minimum-value setting is established by the factory and should not be disturbed. The minimum gain is set by the long pot at the far left (set for unity gain by the factory). To increase the gain, adjust the smaller pot at the second-from-left position. The minimum gain which can be used is 1 (unity gain); the maximum gain is about 8.

The minimum offset is set by the pot to the far right (set for zero offset by the factory). To increase the offset, adjust the second-rightmost pot. The offset can span a large enough range to digitize any portion of the input signal at any gain up to the maximum.

13.2.2 Programmed initialization

Data may be written to six VI8/VI24 control registers whose addresses are in the range 30100$0 to 30100$5. These registers are write-only; when one of these addresses is read from, the data returned is unpredictable. Function code 20 octal should be used to write to the registers of the VI8/VI24. The meaning of each register is defined below.

13.2.2.1 VIEWPORT LOCATION (REGISTER 30100$0)

The lower 16 bits of the VIEWPORT LOCATION register determine the horizontal starting position of the first sample taken from the input signal. Sampling on each line starts at the leading edge of the horizontal drive pulse. This pulse occurs 9.4 microseconds before the video picture data.

The upper 16 bits of the VIEWPORT LOCATION register determine the vertical starting position of the first line to be sampled. Sampling each frame starts at the end of the vertical drive pulse, which is fourteen lines before the start of the picture data.

By proper setting of the VIEWPORT LOCATION register, any desired section of the input picture may be sampled. The number of sample times between horizontal drive and the area of interest, and the number of lines between vertical drive and the area of interest, are written to 30100$0.

13.2.2.2 MINIFICATION FACTOR (30100$1)

The MINIFICATION FACTOR is the factor by which the picture is compressed before it is stored into the DR64 framebuffer memory. The minification is achieved by reducing the sampling frequency by an integral divisor. Minification in x and y are independent.

The lower 16 bits of the MINIFICATION FACTOR register are the x minification factor; the amount of "squeeze" in the x-direction, as an integral divisor of the sampling frequency; the number written to the register should be the desired minification factor minus 1. For example, a setting of 1 corresponds to a minification factor of 2 and means that the x-direction will be squeezed to half the normal width.

The upper 16 bits of the MINIFICATION FACTOR register are the y minification factor, the amount of "squeeze" in the y-direction, as an integral divisor of the sampling frequency. The number written to the register should be the desired minification factor minus 1. For example, a setting of 1 corresponds to a minification factor of 2, and means that the y-direction will be squeezed to half the normal height.

13.2.2.3 WINDOW LOCATION (30100$2)

The lower 16 bits of the WINDOW LOCATION register determine the starting X address (within the frame buffer memory) into which the digitized picture data are to be written. For valid results, this number must be a multiple of 16.

The upper 16 bits of the WINDOW LOCATION register determine the starting Y address (within the frame buffer memory) into which the digitized picture data are to be written.

13.2.2.4 VIEWPORT SIZE (30100$3)

The VIEWPORT SIZE register gives the size of the sampled picture, in number of lines and number of columns. The lower 16 bits of the VIEWPORT SIZE register give the number of samples per line, and must be a multiple of 16 for proper operation. The upper 16 bits of the VIEWPORT SIZE register give the number of lines to be sampled.

13.2.2.5 CONTROL REGISTER (30100$4)

The CONTROL REGISTER contains bits which set the operating mode of the VI8/VI24.

13.2.2.5.1 ONE-FRAME BIT (BIT 1)

When the 'run' bit (bit 2) if off, and a logical 1 is written to the 'one-frame' bit (bit 1), the VI8/VI24 will digitize one frame and stop. The VI8/VI24 may be restarted by writing a 1 to the one-frame or run bit.

13.2.2.5.2 RUN BIT (BIT 2)

When the 'run' bit (bit 2) is on, the VI8/VI24 will continuously digitize frames into the DR64 frame-buffer. When the 'run' bit is on, the 'one-frame' bit is ignored.

13.2.2.5.3 SAMPLING CLOCK SOURCE (BITS 5-7)

Bits 5-7 of 30100$4 select the source of the sampling clock. When bits 5-7=000, the internal oscillator (controlled by 30100$5) is used; when bits 5-7=111, the pixel clock is taken from the BNC connector at the top to the VI8/VI24. All other combinations of bits 5-7 are invalid.

13.2.2.6 SAMPLING FREQUENCY (30100$5)

The sampling frequency of the internal sampling clock is controlled by bits 0-3 of location 30100$5. The time per sample varies from about 95 nanoseconds per pixel (setting=0000) to about 125 nanoseconds per pixel (setting=1111), varying in 2-nanosecond increments. It can be seen that the sampling frequency increases as the setting of the sampling frequency register decreases.

14.0 HARDWARE CHARACTER GENERATOR PROGRAMMING GUIDE

15.0 ROUTINES FOR COMMUNICATING WITH THE IKONAS PROCESSOR

15.1 I/O Routines

Users should have a standard set of I/O routines to allow them to run diagnostics supplied by IKONAS. The following facilities must be provided by a standard I/O package, callable from FORTRAN:

- o pixel I/O
- o word I/O
- o single-pixel I/O

IKBDMA* is a set of I/O routines from IKONAS which perform these functions. The routines in IKBDMA are described below. The arguments are of two types: (1) FORTRAN INTEGER variables (unless otherwise specified); (2) IKONAS 32-bit variables. 32-bit variables may be declared as INTEGER*4, or two-element arrays, according to user's machine.

*RT-11 version:	IKBDMA.MAC
RSX version:	IKBDMARSX.MAC
VAX/VMS version:	IKVAXDMA.MAR

15.1.1 Pixel I/O

Functions at this level fall into three categories:

1. Functions to perform movement of data (read, write, wait);
2. Functions to set control lines in the IKONAS machine;
3. Functions to set up host interface.

Note: At this basic level we will be dealing with data in IKONAS form--that is, 32-bit data words, (X,Y) addressing.

Data-movement

1. IKPRD (code, x addr, y addr, length, data, IKONAS status, system status)

code = IKONAS function code.
Default = 2;
pixel-mode bit (bit 1) will be set;
write bit (bit 4) will be cleared.

x addr = 10-bit x-value

y addr = 14-bit y-value

length = number of 32-bit IKONAS words to read. One word of whatever size is the default on the user's machine (to allow this argument to be specified as a constant)

data = IKONAS data, sequential 32-bit variables.

This entry point starts to read, then returns to caller.

The status arguments are optional. Errors detected by the IKONAS system during a transfer are reported in the IKONAS status argument. Errors detected by the Host operating system are reported in the system status argument. IKONAS status of 0 indicates normal completion; system status is defined by the operating system. If an error occurs, and the corresponding status argument is not given, the user's program is terminated.

2. IKPWR (code, x addr, y addr, length, data, IKONAS status, system status)

code = IKONAS function code.
Default = 22 octal;
the pixel-mode bit (bit 1) and
the write bit (bit 4) will be set

(address, length, data, status, as for IKPRD)

3. IKBWT(Ikonas status, system status)

This function returns to caller when the previous I/O is completed.

IKONAS Control

Control lines in the IKONAS machine are those lines which are independent of the host machine. They include: invisible I/O, increment, run processor, IKONAS reset, frame interrupt, processor interrupt.

Control lines in the host interface are those lines which may change from one host to the next. They include: Extended bus address, Halfword, DMA Enable.

1. IKBSEI (code)

   code = IKONAS status mask
   (see "Host/IKONAS Interface" in this manual)

This includes: invisible I/O, increment, run processor, clear, frame int., proc. int. Default is 'increment'.

2. IKBGEI (code)

   returns the current IKONAS status

Host Control

1. IKOPEN

performs any operating-system-dependent functions to open a channel to the IKONAS.

2. IKBSEH (code)

   code = Host status mask.

This mask is dependent on the host.

3. IKBGEH (code)

This routine returns the host status mask.

15.1.2 Word I/O

Functions to perform word I/O differ from pixel I/O only in the method of addressing. The address is specified as a 24-bit number instead of X and Y addresses.

1. IKBRD (code, address, length, data, IKONAS status, system status)

   code = IKONAS function code.
   Default is 0;
   The write bit (bit 4) will be cleared.

   address is a 24-bit address

2. IKBWR (code, address, length, data, IKONAS status, system status)

   code = IKONAS function code.
   Default is 20 octal;
   The write bit (bit 4) will be set.

   address is a 24-bit address;
   the 24-bit address must occupy the low 24 bits of a 32-bit variable

   Other parameters as for IKPRD, IKPWR

15.1.3 Single-Pixel I/O

Two routines will be available to programs such as "PAINT" which require high-speed I/O for single pixels. In multiuser systems which have high overhead for I/O, these routines should be implemented through some fast path.

1. IKPRD1 (code, x addr, y addr, pixel data, IKONAS status, system status)

2. IKPWR1 (code, x addr, y addr, pixel data, IKONAS status, system status)

   Parameters are as for IKPRD and IKPWR

On certain operating systems it may be necessary to buffer several pixels together for a single write. For this reason, an application should call IKBWT to ensure that all pixels are out.

15.2 Routines for 32-bit arithmetic

Since the IKONAS is a 32-bit machine, to achieve machine independence it is necessary to have a set of routines which perform 32-bit arithmetic. The following routines, callable from FORTRAN, are available in the file ALU32.MAC. These routines assume that the host machine has some way of declaring 32-bit numbers (e.g., as integer*4).

1. JOIN32 (vbl32, vblhi, vbllo)

creates a 32-bit word by JOINing two 32-bit integers.

vbl32 = resultant 32-bit word

vblhi = 16-bit integer variable which will become the high-order bits of vbl32;
vbllo = 16-bit variable which will become the low-order bits of vbl32.

Once JOIN32 has combined two integer variables into one IKONAS word, only the routines listed below may use the variable.

2. BRK32(vbl32, vblhi, vbllo)

splits a 32-bit number into two 16-bit components. It is the complement of JOIN32.

vbl32 = 32-bit number;
vblhi = 16-bit number resulting from the high-order bits of vbl32;
vbllo = 16-bit number resulting from the low-order bits of vbl32.

3. SET32(target, source)

assigns the contents of source to target. Both target and source are 32-bit numbers.

4. ADD32(augent, addend)

adds one variable to another. Both augend and addend are 32-bit numbers. ADD32 adds the contents of addend to augend and stores the result in augend.

5. SUB32(minuend, subtrahend)

subtracts one variable from another. Both minuend and subtrahend are 32-bit numbers. SUB32 subtracts the contents of subtrahend from minuend and stores the result in minuend.

6. ICMP32(minuend, subtrahend)

compares two numbers and returns a result indicating their relative value. Both minuend and subtrahend are 32-bit numbers. ICMP32 returns a value of 0 if minuend = subtrahend; -1 if minuend < subtrahend; +1 if minuend > subtrahend.

16.0 LINKER/LOADER FILE DESCRIPTION

16.1 Introduction

This document describes the format of disk files which contain data used by IKLINK, the Ikonas relocating linkage-editor, and by IKLOD, the Ikonas absolute loader. The linkage-editor can be used to link microcode routines, IDL routines, libraries, colormap files, scratch pad data files, and character fonts all into one loadable absolute code file, which is then loaded into the Ikonas system by the absolute loader. Each record in a linker/loader file is written by an unformatted FORTRAN write, or equivalent. Each element in the record is an integer variable (size=compiler default).

16.2 File Record Format

Any file used by IKLOD or IKLINK may consist of several different records, each containing different types and amounts of data. Records may contain relocation information, program descriptor information, or global symbol pointer data. The record types and their contents are defined as follows:

Variable	Definition
1	The number of integer variables in the rest of the record.
2	Record type (see below).
3-nnn	Data, the meaning of which depends on the record type.

The recognized record types are defined as follows:

TYPE 256: IKASM MODULE HEADER

3	IKASM version number
4-12	Program name
13-14	IKONAS Bus starting address of module (Hi-Lo)
15-16	Length of module in IKONAS words
17	Data field width of microcode words (DATFW)
18	Microcode word width (MCWID)
19-26	Reserved for future use.

TYPE 257: GLOBAL SYMBOL DIRECTORY (GSD) ENTR

3-10	Symbol name
11	GSD flags: bit 0 ON=Global, OFF=External 1 ON=Relative, OFF=Absolute 2 3
12	Number of this GSD entry
13-14	Value of the symbol
15-18	Reserved for future use.

TYPE 258: RELOCATION DIRECTORY (RLD) ENTRY

3-4	Offset in module of the symbol
5	GSD number of the symbol this is relative to
6-10	Reserved for future use.

TYPE 259: IDL MODULE HEADER

3	IDL version number
4-12	Program name
13-14	IKONAS Bus starting address of the module (Hi-Lo)
15-16	Length of the module in IKONAS words
17-26	Reserved for future use.

TYPE 1: IKONAS DATA

3	GSD number of symbol these data are relative to
4	High-order IKONAS Bus address of the data (8 bits)
5	Low order IKONAS Bus address of the data (16 bits)
6-9	Reserved for future use
10-nnn	IKONAS data, 32-bit words

16.3 Order of Records in File

The records in a linker/loader file are ordered in a specific manner. The loader will ignore any records which are not related to its task. The linker uses most of the defined record types, but does not necessarily require that all types be present in the file. The records are ordered as follows:

```
Module Header (type 256, 259, etc.)
GSD (type 257)
GSD
.
.
.
Ikonas data (type 1)
Ikonas data
.
.
.
RLD (type 258)
RLD
.
.
.
```

16.4 IKLOD - IKONAS Absolute Loader

IKLOD is a subroutine which reads files in loader format, then writes the data to the Ikonas Bus. The calling format is:

```
IKLOD(iunit)
```

where

iunit is the unit number of the file containing the data to load. The unit number should be defined by the calling procedure before entry to IKLOD.

IKLODF is similar to IKLOD, except that it is used for ASCII-formatted files.

17.0 SOFTWARE INSTALLATION

17.1 Introduction

The following sections describe the steps necessary to:

1. install IKASM, the IKONAS Assembler;

2. install and execute PREP, the program which builds the mnemonic definition tables for IKASM;

3. assemble a microcode file using IKASM;

4. compile, link, and run BMTST, a diagnostic test for the BPS32 bipolar processor;

5. compile, link, and run MATST, a diagnostic test for the MA1024;

6. install IKONAS-supplied routines for the MA1024 Multiplier-Accumulator.

All FORTRAN programs supplied by IKONAS have been written in STANDARD FORTRAN IV (1966). All programs and data needed for installation have been stored on magnetic tape using DOS-11 format (a standard DEC formatting interface).

17.2 IKASM Installation

To install IKASM:

1. Modify and compile

   ALU32.MAC and
   ASMIO.FTN

These files contain operating system-dependent routines callable from FORTRAN. For system compatibility, you may need to modify one or both of these files. ALU32 describes and implements a set of routines to perform 32-bit arithmetic on a PDP-11. ASMIO contains all necessary I/O routines to open and close files and to interface with the terminal. The code was written for implementation under RSX-11M 3.2 (operating system for PDP-11 computers).

2. Compile

   IKASM4.FTN - the main program for IKASM
   IKALIB.FTN - a set of subroutines used by IKASM

3. Link IKASM4, IKALIB, ASMIO, and ALU32.

17.3 Mnemonic Definition File

To assemble microcode using IKASM, you need both a source file and a mnemonic definition table (symbol table). PREP builds the mnemonic definition table, using one of two input files:

   BPS.TXT - for the BPS32 bipolar processor;
   MAMNEN.TXT - for the MA1024 Multiplier-Accumulator

To install and execute PREP:

1. Compile PREP.FTN.

2. Link PREP, IKALIB, ALU32, and ASMIO. See the previous section on IKASM for descriptions of the last

three files.

3. Modify the mnemonic definition file (BPS.TXT or MAMNEN.TXT), if necessary for system compatibility.

4. Run PREP, responding to the prompts as shown below:

```
OPCODE DEF. FILE:   input filename

PREP OUTPUT FILE:   output filename
```

where

input filename is the name of the mnemonic definition file -- either BPS.TXT or MAMNEN.TXT;

output filename is the name of the symbol table produced by PREP -- BPS.PRP or MAMNEN.PRP.

17.4 IKASM Execution

To assemble a microcode file using IKASM: run IKASM, responding to the prompts as shown below:

```
PREPROCESSED SYMBOL TABLE FILE:  symbol table filename
LISTING FILE:   program-name.LST
SOURCE FILE:    program-name.ASM
OBJECT FILE:    program-name.MOB
```

where

symbol table filename can be

```
BPS.PRP (for the BPS32) or
MAMNEN.PRP (for the MA1024)
```

program-name is the name of your source program.

17.5 BMTST

BMTST is a set of diagnostics for the BPS32 bipolar processor; it exercises a series of test cases with filenames BPTxxx.MOF (where "xxx" represents the test case number). To run BMTST:

1. Compile

   BMTST.FOR - the main procedure for BMTST;
   IKBMP.FOR - subroutines for BMTST.

2. Link BMTST, IKBMP, IKLODF, ALU32, and IKBDMA. See the sections for IKLOD and IKASM for descriptions of the last three files.

3. Run BMTST, responding to the prompts as follows:

   ENTER STARTING TEST CASE: starting-filename

   ENTER ENDING TEST CASE: ending-filename

where

starting-filename is the filename of the first case to be tested; enter BPT003.MOF;

ending-filename is the name of the last case to be tested; enter BPT079.MOF.

17.5.1 BMTST Messages

While each test is running a message will appear describing the test -- for example,

   BPT004.MOF,OUTPUT = (205\0,0\1)

After all the test cases have executed, the following message will appear:

   SUCCESSFUL COMPLETION

If an error has occurred during the test, the following messages may appear:

MESSAGE: STARTING TEST CASE NOT FOUND
ENTER STARTING TEST CASE

EXPLANATION: Invalid test case name.

USER RESPONSE: Retype starting test case name.

MESSAGE: ERROR WHILE EXECUTING
BPTXXX.MOF
ADDRESS ACTUAL RESULT EXPECTED RESULT
202 XXXXXX YYYYYY
CONTINUATION IS NOT FEASIBLE

EXPLANATION: A hardware error has occurred.

USER RESPONSE: Check for disconnected cables.
If a cable is disconnected,
recable it and rerun test;
otherwise, contact IKONAS.

17.6 MATST

MATST is a diagnostic sequence for the MA1024. To install and run MATST:

1. Compile

   MATST.FOR - the main procedure

   IKMA.FOR - subroutines for MATST

2. Link MATST, IKMA, IKLODF, ALU32, and IKBDMA. See the first two sections of this chapter for descriptions of the last three files.

3. Run MATST, responding to the prompts as follows:

   ENTER STARTING TEST CASE: MA035A.MOF

   ENTER ENDING TEST CASE: MA055.MOF

17.6.1 MATST Messages

While each test is running, a message will appear describing the test -- for example:

MA004.MOF,OUTPUT = (205\0,0\1)

After all the test cases have executed, the following message will appear:

SUCCESSFUL COMPLETION

If an error has occurred during the test, the following messages may appear:

MESSAGE: STARTING TEST CASE NOT FOUND
ENTER STARTING TEST CASE

EXPLANATION: **Invalid test case name.**

USER RESPONSE: **retype starting test case name.**

MESSAGE: ERROR WHILE EXECUTING
MAXXX.MOF
ADDRESS ACTUAL RESULT EXPECTED RESULT
202 XXXXXX YYYYYY
CONTINUATION IS NOT FEASIBLE

EXPLANATION: A hardware error has occurred.

USER RESPONSE: Check for disconnected cables. If a cable is disconnected, recable it and rerun the test; otherwise, contact IKONAS.

17.7 IKONAS Routines for the MA1024

There are several IKONAS-supplied routines for 3-D transformation, clipping, and perspective division. To install these routines, run IKASM, responding to the system prompts as follows:

```
PREPROCESSED SYMBOL TABLE FILE:   MAMNEN.PRP
                 LISTING FILE:   program-name.LST
                  SOURCE FILE:   program-name.ASM
                  OBJECT FILE:   program-name.MOB
```

where

program-name is the name of the desired routine (MM33, MM32, MM33C, MM33P, or MM32P). See the "MA1024 Host Programming Guide" in this manual for descriptions of these routines.

APPENDIX O

PRIORITY LEVELS FOR IKONAS DEVICES

IKONAS devices have been assigned the following priority levels. A level of 7 is the highest priority. Two modules with the same priority level will not be part of the same system.

Level	Module
7	FBC0 (Frame Buffer Controller)
6	VI8/VI24 (Video Input Module)
5	FBC1 (second Frame Buffer Controller) or VPM (Video Processing Module)
4	IF0 (Interface for FBC0)
3	IF1 (Interface for FBC1)
2	MPC (Multifunction Peripheral Controller)
1	CGM4 (Character Generator) or BPS1 (Bit Slice Processor)
0	BPS0 (Bit Slice Processor)