PERITEK CORPORATION TECHNOLOGY SECRET

CONFIDENTIAL DOCUMENT

DOCUMENT TITLE: BOREALIS TECHNICAL REFERENCE MANUAL

REV. NO.: **1.0**

DATE: **April 10, 2002**

REF. NO.: **L2B1188-004**

UNAUTHORIZED USE OR DISCLOSURE OF THIS INFORMATION COULD IMPACT PERITEK CORPORATION'S COMPETITIVE ADVANTAGE. THIS DOCUMENT CONTAINS CONFIDENTIAL INFORMATION.

Borealis

Technical Reference Manual

High Performance

128-Bit Graphics and

Multimedia Processor

Peritek Corporation

Copyrights

Notice

Information in this document is subject to change without notice and does not represent a commitment on the part of Peritek Corporation. Peritek Corporation reserves the right to revise this publication and to make changes from time to time in its content without obligation of Peritek Corporation to notify any person or organization of such revision or change.

Trademarks

Borealis is a trademark of, and the Peritek logo and Peritek Corporation are registered trademarks of Peritek Corporation. Some of the product names used herein have been used for identification purposes only and may be trademarks or registered trademarks of their respective manufacturers and sellers.

Borealis Reference Manual

TABLE OF CONTENTS

Chapter 1: *General Information*

Overview

Borealis is the fourth generation in the family of high performance visual processors. It is implemented in a single gate array using 0.35 micron 3.3 volt CMOS process and is packaged in a 388 PBGA (Plastic Ball Grid Array).

Key Device Features

Borealis includes the following features:

- 66/133 MHz AGP Interface with AGP DMA Bus Mastering (read only)
- 33/66 MHz PCI Interface with PCI Bus Master Event Notification
- 66/33 MHz Host interface clock for PCI Interface
- 100 MHz SGRAM/SDRAM Memory Controller with Block Writes
- Texture Mapping Capabilities
- Perspective Correction
- Point Sampling, Bilinear and Trilinear filtering
- Full Level-of-Detail Per-Pixel MIP Mapping
- Separate Texture Mipmapping Minification and Magnification filtering
- Full OpenGL Texture Decal, Blend and Modulation Modes
- RGB Modulation Lighting Effects
- Support Palettized Textures: 1, 2, 4 and 8 bit
- Support Non-Palette Textures: 8, 16, and 32 bit
- Floating-point Triangle Setup with Vertex Level Commands
- Flat and Gouraud Shaded line drawing, with patterning
- Table and Vertex Fog
- Specular Highlighting
- Full Alpha blending
- Alpha Compare Testing
- 3D Color Keying with Color Range Support
- Backface Culling
- Bilinearly Filtered Scaling
- Power of 2 Display Zooming
- Support 8, 16, 32 bits per pixels Destination Format
- 16 and 24 bit per pixel Z Buffer Support
- Hardware 3D Volume Clipping
- 16-bit Logical Addressing in both X and Y
- Two configurable Memory Windows
- Transparent BLT and Two operand Bit Blts
- Display List Processor
- Color Space Converter: YUV-RGB Conversion
- VGA
- 250MHz RAMDAC

Chapter 2: *Functional Description*

Overview

This chapter describes the architecture and includes a block diagram for Borealis. Borealis provides a high performance, AGP and PCI compliant interface with no additional external logic required.
Software may interact with Borealis by directly manipulating pixels through the Memory Windows interface or by setting up parameters in the register area to trigger one of the Borealis drawing engine commands.
The **Drawing Engine** commands provide all of the normally required operations including:

- BIT BLT
- 2D Line
- 3D Line with setup
- 3D Triangle with full setup and vertex sorting

Borealis is implemented using a highly pipelined graphic processor architecture. This architecture allows for high performance 2D and 3D Rendering. After a sequence of commands and parameters are written, Borealis executes the selected command without any further host processor intervention.

Frame Buffer

Borealis supports one local frame buffer up to 32Mbytes with a data bus width of 128 bits of SGRAM/SDRAM memory.
These buffers may be accessed as linear buffers through the Memory windows interface or through the drawing engine.
The local buffer may be used as a display buffer, as well as off-screen memory to be used for the storage and manipulation of bitmaps, texture maps, Z buffering or fonts.

Register Map

This section provides a brief overview of all Borealis registers. A full description may be found in Chapters 4 and 5 of this manual.

I/O Mapped VGA DAC Registers

The following is a list of the I/O mapped VGA DAC shadowing registers. The addresses of these registers are absolute I/O addresses.
Borealis can easily be configured to respond or not respond to the following four I/O addresses. There are two modes of operation. If VGA DAC register access is enabled in the VGA_CTRL register (see *Appendix D, Ticket to Ride VGA Specification* for details), it can be in one of two modes: snooping and owning (see *Chapter 5, Register Set* for details).

I/O ADDRESS	REGISTER NAME	DESCRIPTION
0x03C6	PEL_MASK (DAC_02)	Pixel Mask
0x03C7	RD_ADR (DAC_03)	Read Address
0x03C8	WR_ADR (DAC_00)	Write Address
0x03C9	PAL_DAT (DAC_01)	Palette Data

I/O Mapped Configuration Registers

The following is a list of the I/O mapped configuration registers. The base I/O address of these registers is set via PCI base register 4 at boot time.

I/O ADDRESS	REGISTER NAME	DESCRIPTION
0x0000	RBASE_G	Base address for the global register
0x0004	RBASE_W	Base address for Mem windows config registers.
0x0008	RBASE_D	Base address for Drawing Engine Register Set
0x000C	Reserved	Reserved
0x0010	RBASE_I	Base address for the Global Interrupt Registers
0x0014	RBASE_E	Base address for the local PROM
0x0018	ID	Chip ID register
0x001C	CONFIG1	Chip Configuration Register 1
0x0020	CONFIG2	Chip Configuration Register 2
0x0024	SGR_CONFIG	SGRAM/SDRAM Configuration Register
0x0028	SSWTCH	Soft Switch Register
0x002C	DDC	DDC Register
0x0030	VGA_CTRL	VGA Control Register
0x0034 - 0x003C	Reserved	Reserved
0x0040	MW1_CTRL	Memory Window 1 control (shadowed)
0x0044	MW1_AD	Memory Window 1 Address (shadowed)
0x0048	MW1_SZ	Memory Window 1 Size (shadowed)
0x004C	Reserved	Reserved
0x0050	MW1_ORG	Memory Window 1 Origin (shadowed)
0x0054	MW1_ORG	Memory Window 1 Origin (shadowed)
0x0058	Reserved	Reserved
0x005C	Reserved	Reserved
0x0060	Reserved	Reserved
0x0064	MW1_MASK	Memory Window 1 Plane Mask (shadowed)
0x0068	BIOS1	General Purpose BIOS Register 1
0x006C	BIOS2	General Purpose BIOS Register 2
0x0070	BIOS3	General Purpose BIOS Register 3
0x0074	BIOS4	General Purpose BIOS Register 4
0x0078	Reserved	Reserved
0x007C	Reserved	Reserved
0x0080	DAC_00	DAC Register 00
0x0084	DAC_01	DAC Register 01
0x0088	DAC_02	DAC Register 02
0x008C	DAC_03	DAC Register 03
0x0090	DAC_04	DAC Register 04
0x0094	DAC_05	DAC Register 05
0x0098	DAC_06	DAC Register 06
0x009C	DAC_07	DAC Register 07

I/O ADDRESS	REGISTER NAME	DESCRIPTION
0x00A0	DAC_08	DAC Register 08
0x00A4	DAC_09	DAC Register 09
0x00A8	DAC_0A	DAC Register 0A
0x00AC	DAC_0B	DAC Register 0B
0x00BO	DAC_0C	DAC Register 0C
0x00B4	DAC_0D	DAC Register 0D
0x00B8	DAC_0E	DAC Register 0E
0x00BC	DAC_0F	DAC Register 0F
0x00D0	DMA_SRC	AGP DMA Address Source Register
0x00D4	DMA_DST	AGP DMA Address Destination Register
0x00D8	DMA_CMD	AGP DMA Command Register
0x00DC	Reserved	Reserved
0x00E0	PCI_BMWA	PCI Bus Master Write Address Register
0x00E4	PCI_BMTM	PCI Bus Master Trigger Mask Register

Memory Mapped Global Registers

The following is a list of the memory mapped global registers. The base address of these registers is offset by the value loaded in the RBASE_G register in Memory space. The first set of DAC registers, 0x0000-0x001C, should only be accessed in non-burst mode.

MEMORY ADDRESS	REGISTER NAME	DESCRIPTION
0x0000	DAC_00	DAC Register 00
0x0004	DAC_01	DAC Register 01
0x0008	DAC_02	DAC Register 02
0x000C	DAC_03	DAC Register 03
0x0010	DAC_04	DAC Register 04
0x0014	DAC_05	DAC Register 05
0x0018	DAC_06	DAC Register 06
0x001C	DAC_07	DAC Register 07
0x0020	INT_VCNT	Vertical Interrupt Counter
0x0024	INT_HCNT	Horizontal interrupt counter
0x0028	DB_ADR	Display start address
0x002C	DB_PTCH	Display pitch
0x0030	CRT_HAC	Horizontal active line width
0x0034	CRT_HBL	Horizontal blank width
0x0038	CRT_HFP	Horizontal front porch width
0x003C	CRT_HS	Horizontal sync width
0x0040	CRT_VAC	Vertical active field width
0x0044	CRT_VBL	Vertical blank width
0x0048	CRT_VFP	Vertical front porch width
0x004C	CRT_VS	Vertical sync width
0x0050	CRT_LCNT	CRT Line Counter
0x0054	CRT_ZOOM	Display zoom factor
0x0058	CRT_1CON	CRT config. register 1.
0x005C	CRT_2CON	CRT config. register 2.
0x0060	DB_ADR2	CRT Display Start Address 2
0x0070	DAC_00	DAC Register 00
0x0074	DAC_01	DAC Register 01
0x0078	DAC_02	DAC Register 02

MEMORY ADDRESS	REGISTER NAME	DESCRIPTION
0x007C	DAC_03	DAC Register 03
0x0080	DAC_04	DAC Register 04
0x0084	DAC_05	DAC Register 05
0x0088	DAC_06	DAC Register 06
0x008C	DAC_07	DAC Register 07
0x0090	DAC_08	DAC Register 08
0x0094	DAC_09	DAC Register 09
0x0098	DAC_0A	DAC Register 0A
0x009C	DAC_0B	DAC Register 0B
0x00A0	DAC_0C	DAC Register 0C
0x00A4	DAC_0D	DAC Register 0D
0x00A8	DAC_0E	DAC Register 0E
0x00AC	DAC_0F	DAC Register 0F

Memory Mapped Global Interrupt/Control Registers

The following is a list of the memory mapped global interrupt/control registers. The base address of these registers is offset by the value loaded in the RBASE_I register in Memory space.

MEMORY ADDRESS	REGISTER NAME	DESCRIPTION
0x0000	GINTP	Global Interrupt Register.
0x0004	GINTM	Global Interrupt Mask Register.
0x0080	RBASE_G	Base address for the global register
0x0084	RBASE_W	Base address for the Memory windows config registers.
0x0088	RBASE_D	Base address for Drawing Engine register set
0x008C	Reserved	Reserved
0x0090	RBASE_I	Base address for the Global Interrupt registers
0x0094	RBASE_E	Base address for the local PROM
0x0098	ID	Chip ID register
0x009C	CONFIG1	Chip Configuration register 1
0x00A0	CONFIG2	Chip Configuration register 2
0x00A4	SGRAM/SDRAM	SGRAM/SDRAM Configuration Register
0x00A8	SSWTCH	Soft Switch register
0x00AC	DDC	DDC Register
0x00B0	VGA_CTRL	VGA Control Register
0x00B4	BIOS1	General Purpose BIOS Register 1
0x00B8	BIOS2	General Purpose BIOS Register 2
0x00BC	BIOS3	General Purpose BIOS Register 3
0x00C0	BIOS4	General Purpose BIOS Register 4
0x00D0	DMA_SRC	AGP DMA Address Source Register
0x00D4	DMA_DST	AGP DMA Address Destination Register
0x00D8	DMA_CMD	AGP DMA Command Register
0x00DC	Reserved	Reserved
0x00E0	PCI_BMWA	PCI Bus Master Write Address Register
0x00E4	PCI_BMTM	PCI Bus Master Trigger Mask Register

Memory Windows Registers

The following is a list of the memory mapped Memory Windows™ registers. The base address of these registers is offset by the value loaded in the RBASE_W register in Memory space.

MEMORY ADDRESS	REGISTER NAME	DESCRIPTION
0x0000	MW0_CTRL	Memory window 0 control
0x0004	MW0_AD	Memory window 0 address
0x0008	MW0_SZ	Memory window 0 size
0x000C	Reserved	Reserved
0x0010 or 0x0014	MW0_ORG	MW0 origin
0x0018	Reserved	Reserved
0x001C	Reserved	Reserved
0x0020	Reserved	Reserved
0x0024	MW0_MASK	MW0 Plane Mask
0x0028	MW1_CTRL	Memory window 1 control
0x002C	MW1_AD	Memory window 1 address
0x0030	MW1_SZ	Memory window 1 size
0x0034	Reserved	Reserved
0x0038 or 0x003C	MW1_ORG	MW1 origin
0x0040	Reserved	Reserved
0x0044	Reserved	Reserved
0x0048	Reserved	Reserved
0x004C	MW1_MASK	MW1 Plane Mask
0x0050	MWC_FCNT	Memory Window Cache Flush counter
0x0054	MWC_FLSH	Manual Cache Flush
0x0058	YUV_LI	YUV LUT index
0x005C	YUV_LA	YUV LUT address
0x0060	MW_CTRL	Memory Window 0 and 1 Control
0x0064 - 0x00FC	RESERVED	RESERVED

Memory Mapped Drawing Registers

The following is a list of the memory mapped registers. All of these registers are pipelined so that they are synchronous with the drawing engine. This register block is placed in system memory space by the setting of RBASE_D in the Memory mapped register block.

REGISTER ADDRESS	REGISTER NAME	DESCRIPTION
0x0000	INTP	Interrupt register
0x0004	INTM	Interrupt Mask register
0x0008	FLOW	Flow status register
0x000C	BUSY	Busy status bit
0x0010	DEW_AD	Drawing Engine Window Address
0x0014	Reserved	Reserved
0x0018	Reserved	Reserved
0x001C	Reserved	Reserved
0x0020	BUF_CNTRL	Buffer enables
0x0024	Reserved	Reserved
0x0028	DE_SORG	Drawing Source origin
0x002C	DE_DORG	Drawing destination origin
0x0030	Reserved	Reserved
0x0034	Reserved	Reserved
0x0038	DE_TPTCH	Texture Pitch of LOD0
0x003C	DE_ZPTCH	Z buffer pitch
0x0040	DE_SPTCH	Source pitch
0x0044	DE_DPTCH	Destination pitch
0x0048	CMD	Command register
0x004C	Reserved	Reserved
0x0050	CMD_OPC	CMD opcode field
0x0054	CMD_ROP	CMD raster op
0x0058	CMD_STYLE	CMD line style
0x005C	CMD_PATRN	CMD line pattern control
0x0060	CMD_CLP	CMD clip control
0x0064	CMD_PF	CMD Pattern Fetch
0x0068	FORE	Foreground color register
0x006C	BACK	Background color register
0x0070	MASK	Plane Mask
0x0074	DE_KEY	Color Key
0x0078	LPAT	Line pattern register
0x007C	PCTRL	Line pattern control register
0x0080	CLPTL	Clip Rectangle Top Left Corner
0x0084	CLPBR	Clip Rectangle Bottom Right Corner
0x0088	XY0	XY0
0x008C	XY1	XY1
0x0090	XY2	XY2
0x0094	XY3	XY3
0x0098	XY4	XY4
0x009C	Reserved	Reserved
0x00A0	Reserved	Reserved
0x00A4	Reserved	Reserved
0x00A8	Reserved	Reserved
0x00AC	Reserved	Reserved

REGISTER ADDRESS	REGISTER NAME	DESCRIPTION
0x00B0	Reserved	Reserved
0x00B4	Reserved	Reserved
0x00B8	Reserved	Reserved
0x00BC	Reserved	Reserved
0x00C0	Reserved	Reserved
0x00C4	Reserved	Reserved
0x00C8	Reserved	Reserved
0x00CC	Reserved	Reserved
0x00D0	LOD0	Level of detail 0 Origin
0x00D4	LOD1	Level of detail 1 Origin
0x00D8	LOD2	Level of detail 2 Origin
0x00DC	LOD3	Level of detail 3 Origin
0x00E0	LOD4	Level of detail 4 Origin
0x00E4	LOD5	Level of detail 5 Origin
0x00E8	LOD6	Level of detail 6 Origin
0x00EC	LOD7	Level of detail 7 Origin
0x00F0	LOD8	Level of detail 8 Origin
0x00F4	LOD9	Level of detail 9 Origin
0x00F8	DL_ADR	Display list address
0x00FC	DL_CNTRL	Display list control
0x0100	DE_ZORG	Z-buffer origin
0x0104	Reserved	Reserved
0x0108	Reserved	Reserved
0x010C	Reserved	Reserved
0x0110	Reserved	Reserved
0x0114	Reserved	Reserved
0x0118	TPAL_ORG	Texture Palette origin
0x011C	HITH	Hither Register
0x0120	YON	Yon Register
0x0124	FOG_COL	Fog Color (Af,Rf,Gf,Bf)
0x0128	ALPHA	Alpha Register(Atest,Asrc,Adst)
0x012C	TEX_BORDER	Texture Border Color (RGB)
0x0130	V0_A_FP	Vertex0 Alpha Channel Floating point Input
0x0134	V0_R_FP	Vertex0 Red Channel Floating Point Input
0x0138	V0_G_FP	Vertex0 Green Channel Floating Point Input
0x013C	V0_B_FP	Vertex0 Blue Channel Floating Point Input
0x0140	V1_A_FP	Vertex1 Alpha Channel Floating Point Input
0x0144	V1_R_FP	Vertex1 Red Channel Floating Point Input
0x0148	V1_G_FP	Vertex1 Green Channel Floating Point Input
0x014C	V1_B_FP	Vertex1 Blue Channel Floating Point Input
0x0150	V2_A_FP	Vertex2 Alpha Channel Floating Point Input
0x0154	V2_R_FP	Vertex2 Red Channel Floating Point Input
0x0158	V2_G_FP	Vertex2 Green Channel Floating Point Input
0x015C	V2_B_FP	Vertex2 Blue Channel Floating Point Input
0x0160	KEY_3D_LOW	3D Key value, low color
0x0164	KEY_3D_HI	3D Key value, High color
0x0168	CMD	Command Register
0x016C	A_CNTRL	Alpha Control Register
0x0170	3D_CNTRL	3D Control Register
0x0174	TEX_CNTRL	Texture Control Register

REGISTER ADDRESS	REGISTER NAME	DESCRIPTION
0x0178	CP0	Command Parameter 0
0x017C	CP1	Command Parameter 1
0x0180	CP2	Command Parameter 2
0x0184	CP3	Command Parameter 3
0x0188	CP4	Command Parameter 4
0x018C	CP5	Command Parameter 5
0x0190	CP6	Command Parameter 6
0x0194	CP7	Command Parameter 7
0x0198	CP8	Command Parameter 8
0x019C	CP9	Command Parameter 9
0x01A0	CP10	Command Parameter 10
0x01A4	CP11	Command Parameter 11
0x01A8	CP12	Command Parameter 12
0x01AC	CP13	Command Parameter 13
0x01B0	CP14	Command Parameter 14
0x01B4	CP15	Command Parameter 15
0x01B8	CP16	Command Parameter 16
0x01BC	CP17	Command Parameter 17
0x01C0	CP18	Command Parameter 18
0x01C4	CP19	Command Parameter 19
0x01C8	CP20	Command Parameter 20
0x01CC	CP21	Command Parameter 21
0x01D0	CP22	Command Parameter 22
0x01D4	CP23	Command Parameter 23
0x01D8	CP24	Command Parameter 24
0x01DC	3D_TRIG	Trigger Register for 3D
0x01E0	GLBLENDC	OpenGL Blend Color
0x01E4	Reserved	Reserved
0x01E8	Reserved	Reserved
0x01EC	Reserved	Reserved
0x01F0	Reserved	Reserved
0x01F4	Reserved	Reserved
0x01F8	Reserved	Reserved
0x01FC	Reserved	Reserved

Block Diagram

Borealis is partitioned into the following functional sections:

- Ï Host Bus Interface
- Ï Aperture Controller
- Ï Drawing Engine
- Ï CRT Controller
- Ï Memory Controller
- Ï Internal VGA
- Ï Internal RAMDAC

The **Host Bus Interface** provides an interface to the system bus (PCI or PCI/AGP). It implements a full PCI slave interface, responding to reads and writes of configuration, memory, and I/O cycles. It also implements a PCI master interface for specific memory writes (see *Chapter 5, Register Set*). Lastly, it implements an AGP master interface for DMA reads (see *Chapter 5, Register Set*). It also generates peripheral bus control for a FLASH EPROM, an external RAMDAC, and an external soft switch register.

The **Linear Windows Controller** provides address decoding, address translation, color space conversion between the host interface and the local memory system. It also provides a mechanism for caching reads and writes from the host bus to the local buffers. In write mode, up to eight 32 bit words may be written to the host bus cache. The cache continuously monitors the address of each word written to determine if they are in the same page. If the words are not in the same page, or if the cache word count reaches eight, the cache will request the required number of memory writes from the Memory Controller. At this time the cache controller swaps access to its second cache and continues to accept host writes. If another page fault is detected during the secondary cache fill, a system stall will occur. This situation can be avoided by testing the cache and by doing cache line fills. During reads latency will be incurred for initial accesses or any page fault conditions. Software should make an effort to maintain scan line coherency during any access to the local buffers for optimal performance.

The **Drawing Engine** provides all the required logic to implement BITBLT, LINE, LINE_3D, TRIAN_3D, and HOST XFER commands. The Drawing Engine, when triggered, transfers command and parameter information from the host accessible registers to its own local working registers where it begins its setup phase. When the Drawing Engine is done with its setup, it begins the execution of a specific algorithm for the associated command.

For non-rendering commands, after the setup phase, the Drawing Engine begins requesting memory access from the Memory Controller. For 2D and/or 3D rendering commands, the object is piped through the algorithmic rendering engine which begins requesting memory access from the Memory Controller as soon as the first pixel/texel is generated. Up to 2 rendering commands can be piped through the rendering engine at the same time.

If read data is requested, the memory controller will control the loading of the data into the Drawing Data path and will notify the Drawing Engine that the data is now available. If write data is requested, the data will have been previously setup in the drawing data path and the Memory Controller will control the output of that data to the selected memory buffer.

The **Display List Processor** (DLP) is used to feed a set of commands to the Drawing Engine. It can also be used to schedule AGP DMA block transfers. The DLP uses a 128-bit instruction word. The instruction formats allow for each word to write up to three Drawing Engine registers, two text glyphs, or a DMA transfer. There is a four register mode which only writes XY0, XY1, XY2, and XY3. This mode cannot be mixed with any other mode.

The **CRT Controller** provides programmable CRT timing signals: horizontal, vertical blanks and syncs. It is also responsible for generating requests to the memory controller for screen refresh cycles. A free running frame counter which generates interrupts to the Host is also provided. This is useful for synchronizing bit map copies. **CRT Controller** also provides display refresh data for the internal RAMDAC.

The **Memory Controller** arbitrates and controls all access to the local memory buffer by the Host Interface, the CRT controller, and the Drawing Engine. This unit provides support for the SGRAM/SDRAM memory.

The Borealis incorporates an IBM-compatible **VGA** core. The VGA core implements the standard VGA register set for the various VGA components (CRT controller, sequencer, graphics controller, attribute controller, etc.) and is capable generating the standard VGA modes (00h - 07h, 00h - 13h). The control of memory and CRT signals can be switched between the VGA core and the Borealis. (See the description of the VGA_CTRL register in Chapter 7, *VGA Specification*.) The VGA memory space is shared with the Borealis frame buffer and is sparsely mapped within it.

The **Internal RAMDAC** transforms the raw data from the CRT controller into signals that an analog or digital monitor can understand. In the process, it can add gamma correction and a high resolution cursor. The RAMDAC also provides two programmable clocks: one for the memory controller, and the other for the pixel data. These clocks can range from 25 MHz to 250 MHz.

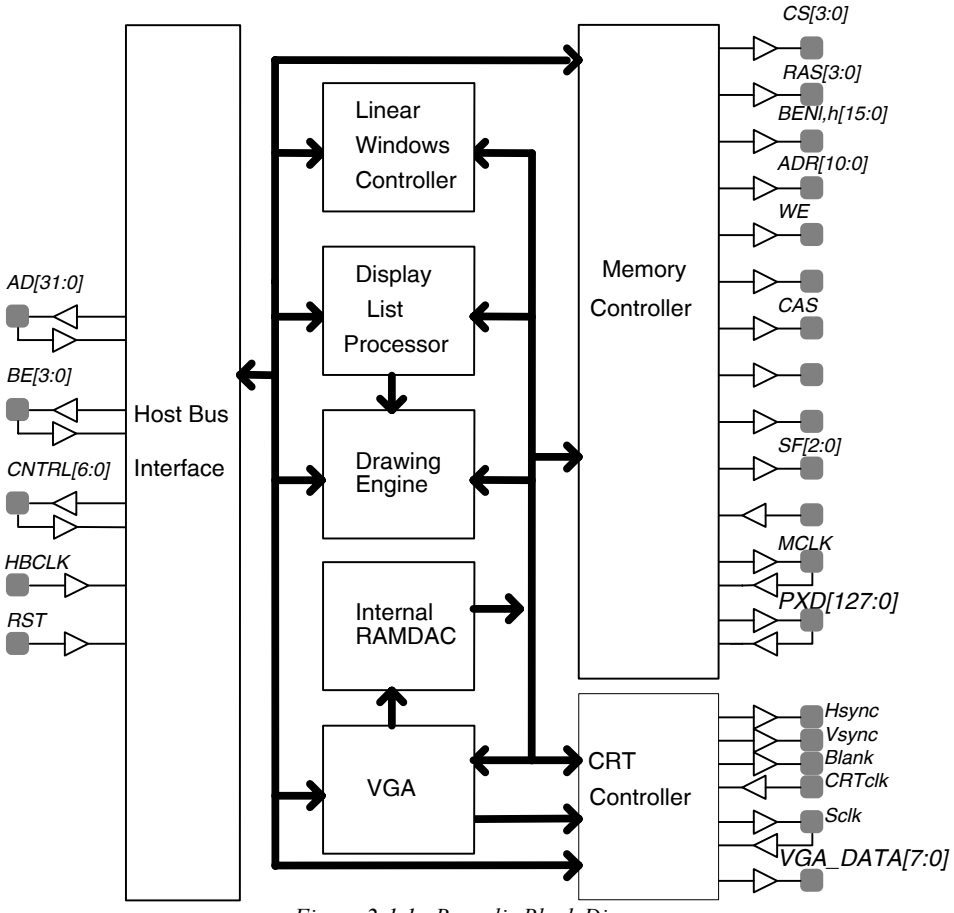


Figure 2-1.1. Borealis Block Diagram

Coordinate System

The screen coordinate system has its origin at the upper left hand corner of the screen, with the X coordinates incrementing left to right and the Y coordinates incrementing top to bottom. The coordinate system for a 1280 by 1024 display is shown in Figure 2-2.

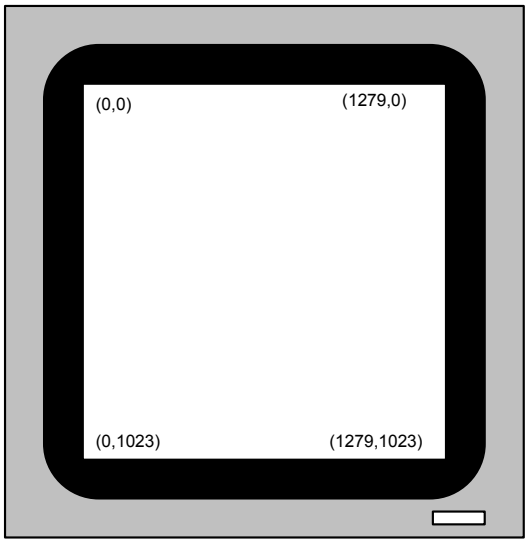


Figure 2-1.2. Coordinate System 1280x1024 Display

Destination X and Y coordinates are 16-bit 2's complement integers. All registers specified in a XY format will be interpreted as 2's complement integers. All internal arithmetic operations are done in 2's complement format; therefore no overflows will be detected or reported. Care must be taken for drawing operations not to step outside of the 16 bit coordinate space.

In the case of the rendering commands, the X and Y coordinates are in IEEE single precision floating point format. The setup for X and Y is done in IEEE single point floating point and is converted to 16-bit 2's complement integers. Again, care must be taken for drawing operations not to step outside of the 16 bit coordinate space.

The display buffer is accessed in this format by specifying the coordinate, the source and/or destination space origin, and the buffer pitch. From this organization it can be seen that the pitch of the display buffer can be changed on a command by command basis.

The Z buffer can be either 16bits in 16bpp mode or 24bits (packed into the lower 3 bytes of a 4byte DWORD) in 32bpp mode. As with the display buffer, the Z buffer can be accessed by specifying the Z buffer origin, Z buffer pitch and the (x,y) coordinate of the Z buffer. As with the X and Y coordinates, the Z values goes through an IEEE single precision floating point setup engine and are converted to the appropriate format (16 or 24 bits) before it is stored into the Z buffer.

Chapter 3: *Borealis I/O Information*

Signal Descriptions

Borealis signal pins are categorized into the following five groups:

- PCI/AGP Bus Interface
- Local Memory Buffer Interface
- Peripheral Devices Interface
- CRT Control
- Miscellaneous

NOTE: The "**#**" indicates an active low signal.

Table 3-1.1. PCI/AGP Bus Interface Signals

NAME	I/O	DESCRIPTION
AD[31:0]	I/O	Multiplexed address and data bus
C_BE[3:0]	I/O	Multiplexed command and byte enables (Active Low as byte enables.)
HCLK	I	Host Bus Clock
RST#	I	System Reset
FRAME#	I/O	Frame
IRDY#	I/O	Initiator Ready
DEVSEL#	I/O	Device Select
STOP#	I/O	Disconnect
INTRP#	O	Interrupt
PAR	I/O	Parity indicator
IDSEL/PIPE#	I/O	PCI Configuration Select/AGP Pipe
REQ#	O	Request bus mastering
GNT#	I	Grant bus mastering
SB_STB	O	AGP SideBand Strobe
SBA[7:0]	O	AGP SideBand Address
ST[2:0]	I	AGP Status (For PCI applications, an external source is required to tie all 3 bits to a logic 1)
AD_STB1	I	AGP High Word Data Strobe (For PCI applications, an external source is required to tie this signal to a logic 1)
AD_STB0	I	AGP Low Word Data Strobe (For PCI applications, an external source is required to tie this signal to a logic 1)

Local Memory Buffer Control Signals

The local memory buffer of Borealis may be built using SGRAM or SDRAM memories:

- SGRAM
- SDRAM

Names and/or function of some signals in this group change accordingly with selected type of memories.

Table 3-1.3. SGRAM/SDRAM Configuration

NAME	I/O	DESCRIPTION
MCLK	I	Memory Clock Feedback To meet timing requirements of SGRAM memories and to facilitate skew control between SCLK clock and all others memory signals, the clock from SCLK pin should be connected to memories and fed back into MCLK pin.
SCLK	I/O	Memory Clock Output Output In SGRAM mode, only the output function of this pin is used providing clock to SGRAM chips.
RAD [10:0]	O	Local Memory Buffer Address RAD[9] always generates SGRAM bank select *For 16Mbit chips with bank select on A[10] pin* *RAD[8] will connect to A[9]* *RAD[9] will connect to A[10]* *RAD[10] will connect to A[8]*
PDAT [127:0]	I/O	Local Memory Buffer Data Bus All PDAT pins have internal pull up resistors
CSn# [3:0]	O	Local Memory Buffer Chip Select Local Memory buffer consist of up to 4 banks. Bank size is 4MB for 8Mbit chips (8MB for 16Mbit chips). CSn[0] corresponds to the address range 0MB-4MB (0MB-8MB) … … CSn[3] corresponds to the address range 12MB-16MB (24MB-32MB All CSn# pins have internal pull up resistors
DQM [15:0]	O	Local Memory Buffer Byte Enable All DQM pins have internal pull up resistors
WEn#	O	Local Memory Buffer Write Enable
DSF	O	Local Memory Buffer Special Function Signal DSF pin has internal pull up resistor
RASn#	O	Local Memory Buffer Row Address Strobe Enable
CASn#	O	Local Memory Buffer Column Address Strobe Enable

NAME	I/O	DESCRIPTION
EMREQn#	I	External memory controller request. An external device may request access to local memories by setting EMREQn pin to low level. EMREQn has to stay low until EMGRNTn will go low (access granted) and through all the time the external device is controlling the memory bus. EMREQn does not have to be synchronous with SCLK If unused EMREQn should be externally pulled up to 3.3V
EMGRNTn#	0	External memory controller access granted CRT_1CON register bit[29]=0 Low level on EMGRNTn indicates that T2R IV finished any pending memory cycles and none of memory bus signals is actively driven (Z or Z with pull up/down). CRT_1CON register bit[29]=1 This setting can be used only if no external device is to share the memory . EMGRNTn pin outputs then the status of currently displayed line in stereo line sequential mode toggling between left image line and right image line EMGRNTn has internal pull up resistor

Table 3-1.4. Peripheral Devices Control Signals

NAME	I/O	DESCRIPTION
PA [7:0]	I/O	Output function: Address to external EPROM or external RAMDAC (PA[2:0] connect to RS[2:0] inputs of the RAMDAC) Address to EPROM is multiplexed on PA[7:0] Higher address byte is output first and should be externally latched by 74F373 type latch enabled by a signal from PCSn pin Lower address byte is output second and should be directly connected to EPROM inputs. Input function: used as configuration pins CP[31:24].For more details see paragraph “Configuration Pins” in the same chapter. All PA pins have internal pull down resistors
PD [7:0]	I/O	Data Bus : to/from external EPROM , to/from external RAMDAC, to external soft switch Inputs are also used as configuration pins CP[23:16].For more details see paragraph “Configuration Pins” in the same chapter. All PD pins have internal pull down resistors

NAME	I/O	DESCRIPTION
BENh[11]	I/0	Frequency Select Output function: VGA Control Register[0]=1: BENh[11] provide frequency select signal to an external clock source (commonly known as FS[0] signal) Input function see BENh[15:0] below BENh[11] pin has internal pull-up resistor
BENh[15:0]	I/O	Configurations Pins *Output functions of these pins are described in Table 3-2 and Table 3-5.* Input functions: BENh[15:0] are used as Configuration Pin CP[15:0]. For more details see paragraph "Configuration Pins" in the same chapter. All BENh[15:0] pins have internal pull-up resistors
SOEn[0] #	O	Write Strobe to external flash EPROM CONFIG1 bit [13] =1: SOEn[0] generates write enable (active low) to be used by external flash EPROM SOEn[0] pin has internal pull up resistor.
SOE[1]	O	EPROM Output Enable CONFIG1 bit [13] =1: SOEn[1] generates output enable (active low) to be used by external EPROM SOEn[1] pin has internal pull up resistor.
PCSn#	O	EPROM Chip Select Also used to latch higher byte of address sent to EPROM PCSn pin has internal pull up resistor
PWRn#	O	External RAMDAC Write PWRn pin has internal pull up resistor
PRDn#	O	External RAMDAC Read PRDn pin has internal pull up resistor
PSFT	O	Load Soft Switch PSFT pin has internal pull down resistor

Table 3-1.5. CRT Controller Signals

NAME	I/O	DESCRIPTION
CRTCLK	I	CRT controller clock or Reference Clock CONFIG2 Register bit[0]=1: Reference clock to internal clock synthesizers generating internal clocks for CRT and memory controller.
SCLK	I/O	In SGRAM mode only the output function of this pin is used providing clock to SGRAM chips.
HSYNC	O	Horizontal Sync
VSYNC	O	Vertical Sync
CBLANK	O	Composite Blank
DDC_DAT	I/O	DDC Data to/from monitor (SDA)
DDC2_CLK	I/O	DDC clock (SCL)
LD_CLK	O	RAMDAC Load Clock Used only with external RAMDAC In SGRAM mode with internal RAMDAC this pin may be used to verify actual frequency of internal CRT controller clock
BENh[[15:12, 7:4]	I/O	VGA pixel data to external RAMDAC Output function: VGA Control Register[0]=1: BENh[15:12, 7:4] form 8-bit VGA pixel data DDAT[7:0] to an external RAMDAC , Input function: See Table 3-4 “Peripheral Devices Control signals” All BENh pins have internal pull up resistor.

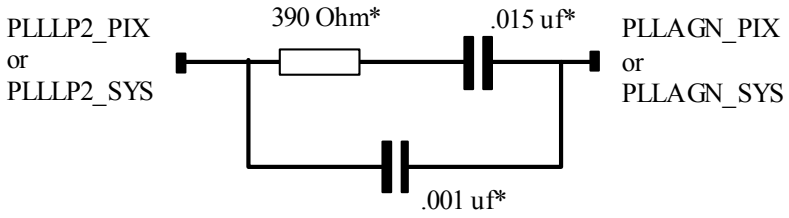
Note: Support for digital RGB output is accomplished by setting CRT_1CON register bit [8] to 1 and ROM_BASE register bit [0] to 0. The chip has to be set to SGRAM/SDRAM configuration. Under these conditions, 24 bits of RGB data appears on I/O pins: RED[7:0] on pins PA[7:0], GREEN[7:0] on pins BENh[15:12,7:4], BLUE[7:0] on pins PD[7:0]. Pixels 0,1,2, etc. can be captured into external registers/transistors on rising edge of LD_CLK. Pixels 1,3,5,etc. can be captured into external registers/transisters on falling edge of LD_CLK. An inverted version of LD_CLK is available (only in this mode) on pin SF[0].

Table 3-1.6. Miscellaneous Signals

NAME	I/O	DESCRIPTION
DECLK	I	Drawing Engine Clock
SECLK	I	Setup Engine Clock CONFIG1 register bits[5:4]=11 Setup Engine runs of external clock on SECLK pin CONFIG1 register bits[5:4] with any other values than 11 SECLK pin is unused and should be connected to GND or pulled high.
BLUE	0	Analog output - blue channel
RED	0	Analog output - red channel
GREEN	0	Analog output - green channel
RGB_RET		return for blue, red, green channel. Should be connected to common GND (in such a way that noise in analog circuits is minimized)
IREF	0	Current reference of nominal value 3.307 mA should be connected between IREF pin and DACAGN
DACAGN		GND for analog circuitry of the internal DAC , should be connected to common GND (in such a way that noise in analog circuits is minimized)
DACVDD		+3.3V VDD for analog circuitry of the internal DAC
PLLAGN_PIX		GND for analog circuitry of the internal pixel clock synthesizer , should be connected to common GND (in such a way that noise in analog circuits is minimized)
PLLAGN_SYS		GND for analog circuitry of the internal memory clock synthesizer , should be connected to common GND (in such a way that noise in analog circuits is minimized)
PLLBIAS_PIX		75kOhm(*) resistor should be connected between PLLBIAS_PIX and PLLVSS_PIX pins 0.1 uF capacitor should be connected between PLLBIAS_PIX and PLLVDD_PIX
PLLBIAS_SYS		75kOhm(*) resistor should be connected between PLLBIAS_SYS and PLLVSS_SYS pins 0.1 uF capacitor should be connected between PLLBIAS_SYS and PLLVDD_SYS
PLLLP2_PIX		an RC network should be connected between PLLLP2_PIX and PLLAGN_PIX pins as shown in figure below
PLLLP2_SYS		an RC network should be connected between PLLLP2_SYS and PLLAGN_SYS pins as shown in figure below
PLLVDD_AGP		PLL +3.3V (host bus interface)
PLLVDD_MC		PLL +3.3V (SGRAM skew control)
PLLVDD_PIX		PLL +3.3V (pixel clock synthesizer)
PLLVDD_SYS		PLL +3.3V (memory clock synthesizer)
PLLVSS_AGP		PLL GND (host bus interface)

NAME	I/O	DESCRIPTION
PLLVSS_MC		PLL GND (SGRAM skew control)
PLLVSS_PIX		PLL GND (pixel clock synthesizer)
PLLVSS_SYS		PLL GND (memory clock synthesizer)
PLLTST	I	TEST PIN, must be connected to GND
PLL_PB	I	TEST PIN, must be connected to GND
IDD_TEST	I	TEST PIN, must be connected to GND
CLTST	I	TEST PIN, must be connected to GND for SGRAM/SDRAM
PROC_MON	O	TEST PIN pin , must be left unconnected

(*) from reference schematic



Pin Assignments

Borealis is packaged in a 388 pin Plastic Ball Grid Array (PBGA).

388 PLASTIC BALL GRID ARRAY

SIGNAL NAME	SOCKET
AD.0	L3
AD.1	L2
AD.2	L1
AD.3	K4
AD.4	K3
AD.5	K2
AD.6	K1
AD.7	J4
AD.8	J1
AD.9	H3
AD.10	H2
AD.11	H1
AD.12	G4
AD.13	G3
AD.14	G2
AD.15	G1
AD.16	D3
AD.17	D1
AD.18	C2
AD.19	C1
AD.20	B1
AD.21	B3
AD.22	A3
AD.23	B4
AD.24	A4
AD.25	D5
AD.26	C5
AD.27	B5
AD.28	A5
AD.29	C6
AD.30	B6
AD.31	A6
AD_STB0	J3
AD_STB1	D7
BENh.0	V1
BENh.1	AC3
BENh.2	AD8
BENh.3	AE14
BENh.4	AC19
BENh.5	AE26
BENh.6	V26
BENh.7	L24
BENh.8	R2
BENh.9	Y4
BENh.10	AC5
BENh.11	AE11
BENh.12	AC17
BENh.13	AD22
BENh.14	W24

SIGNAL NAME	SOCKET
BENh.15	R26
BENl.0 // DQM.0	U4
BENl.1 // DQM.1	AC2
BENl.2 // DQM.2	AE8
BENl.3 // DQM.3	AF14
BENl.4 // DQM.4	AD19
BENl.5 // DQM.5	AD25
BENl.6 // DQM.6	U25
BENl.7 // DQM.7	L25
BENl.8 // DQM.8	R1
BENl.9 // DQM.9	Y3
BENl.10 // DQM.10	AD5
BENl.11 // DQM.11	AF11
BENl.12 // DQM.12	AD16
BENl.13 // DQM.13	AE22
BENl.14 // DQM.14	AA26
BENl.15 // DQM.15	P25
BLUE	C23
C_BEn.0	J2
C_BEn.1	F3
C_BEn.2	D2
C_BEn.3	C4
CAS // WEn	AD26
CBLANK	B24
CLTST	D24
CRTCLK	B20
DACAGN	A22
DACVDD	D22
DDC	C25
DDC2_CLK	A21
DECLK	M26
DEVSELn	E3
EMGRNTn	C26
EMREQn	D25
FRAMEn	E2
GRANTn	C11
GREEN	C22
HCLK	C12
HSYNC	A24
IDD_TEST	M1
INTRP	D12
IRDYn	E1
IREF	B23
LD_CLK	C19
MCLK	G25
OEn // SF	E26
PA.0	B15
PA.1	A15

SIGNAL NAME	SOCKET
PA.3	B16
PA.4	A16
PA.5	D17
PA.6	C17
PA.7	C18
PAR	F2
PCSn	D20
PD.0	A12
PD.1	B13
PD.2	A13
PD.3	C13
PD.4	B14
PD.5	A14
PD.6	D15
PD.7	C15
PDAT.0	R3
PDAT.1	R4
PDAT.2	T1
PDAT.3	T2
PDAT.4	T3
PDAT.5	U1
PDAT.6	U2
PDAT.7	U3
PDAT.8	AA1
PDAT.9	AA2
PDAT.10	AA3
PDAT.11	AB1
PDAT.12	AB2
PDAT.13	AB3
PDAT.14	AB4
PDAT.15	AC1
PDAT.16	AF6
PDAT.17	AE6
PDAT.18	AD6
PDAT.19	AF7
PDAT.20	AE7
PDAT.21	AD7
PDAT.22	AC7
PDAT.23	AF8
PDAT.24	AD11
PDAT.25	AF12
PDAT.26	AE12
PDAT.27	AD12
PDAT.28	AC12
PDAT.29	AF13
PDAT.30	AE13
PDAT.31	AD13
PDAT.32	AF17
PDAT.33	AE17

SIGNAL NAME	SOCKET
PDAT.34	AD17
PDAT.35	AF18
PDAT.36	AE18
PDAT.37	AD18
PDAT.38	AF19
PDAT.39	AE19
PDAT.40	AC22
PDAT.41	AF23
PDAT.42	AE23
PDAT.43	AD23
PDAT.44	AB23
PDAT.45	AC24
PDAT.46	AB24
PDAT.47	AC25
PDAT.48	W25
PDAT.49	Y26
PDAT.50	V23
PDAT.51	V24
PDAT.52	V25
PDAT.53	W26
PDAT.54	U23
PDAT.55	U24
PDAT.56	N23
PDAT.57	N24
PDAT.58	N25
PDAT.59	P26
PDAT.60	M23
PDAT.61	M24
PDAT.62	M25
PDAT.63	N26
PDAT.64	M4
PDAT.65	N1
PDAT.66	N2
PDAT.67	N3
PDAT.68	P1
PDAT.69	P2
PDAT.70	P3
PDAT.71	P4
PDAT.72	V2
PDAT.73	V3
PDAT.74	W1
PDAT.75	W2
PDAT.76	W3
PDAT.77	W4
PDAT.78	Y1
PDAT.79	Y2
PDAT.80	AD1
PDAT.81	AD2
PDAT.82	AF2

SIGNAL NAME	SOCKET
PDAT.83	AF4
PDAT.84	AE4
PDAT.85	AD4
PDAT.86	AF5
PDAT.87	AE5
PDAT.88	AF9
PDAT.89	AE9
PDAT.90	AD9
PDAT.91	AC9
PDAT.92	AF10
PDAT.93	AE10
PDAT.94	AD10
PDAT.95	AC10
PDAT.96	AD14
PDAT.97	AC14
PDAT.98	AF15
PDAT.99	AE15
PDAT.100	AD15
PDAT.101	AC15
PDAT.102	AF16
PDAT.103	AE16
PDAT.104	AF20
PDAT.105	AE20
PDAT.106	AD20
PDAT.107	AC20
PDAT.108	AF21
PDAT.109	AE21
PDAT.110	AD21
PDAT.111	AF22
PDAT.112	AB25
PDAT.113	AC26
PDAT.114	AA24
PDAT.115	AA25
PDAT.116	AB26
PDAT.117	Y23
PDAT.118	Y24
PDAT.119	Y25
PDAT.120	T24
PDAT.121	T25
PDAT.122	U26
PDAT.123	R23
PDAT.124	R24
PDAT.125	R25
PDAT.126	T26
PDAT.127	P24
PIPEn	D10
PLL_BP	B17
PLLAGN_PIX	B19
PLLAGN_SYS	F26

SIGNAL NAME	SOCKET
PLLBIAS_PIX	D18
PLLBIAS_SYS	G24
PLLLP2_PIX	C20
PLLLP2_SYS	G23
PLLTST	D13
PLLVDD_AGP	C14
PLLVDD_MC	K26
PLLVDD_PIX	A19
PLLVDD_SYS	E25
PLLVSS_AGP	B12
PLLVSS_MC	K23
PLLVSS_PIX	A20
PLLVSS_SYS	F25
PRDn	C21
PROC_MON	B18
PSFT	A17
PWRn	B21
RAD.0	H26
RAD.1	J24
SOEn.3 // RAD.10	K24
RAD.2	F24
RAD.3	H24
RAD.4	J25
RAD.5	J26
RAD.6	H23
RAD.7	L26
RAD.8	K25
SOEn.2 // RAD.9	H25
RED	A23
REQn	B11
RGB_RET	B22
RSTn	A11
SB_STB	C8
SBA.0	A9
SBA.1	B9
SBA.2	C9
SBA.3	A8
SBA.4	D8
SBA.5	A7
SBA.6	B7
SBA.7	C7
SCLK	D26
SECLK	A18
SF.0	G26
SF.1 // RASn	E24
SF.2 // CASn	E23
SOEn.0	M3
SOEn.1	M2
ST.0	A10

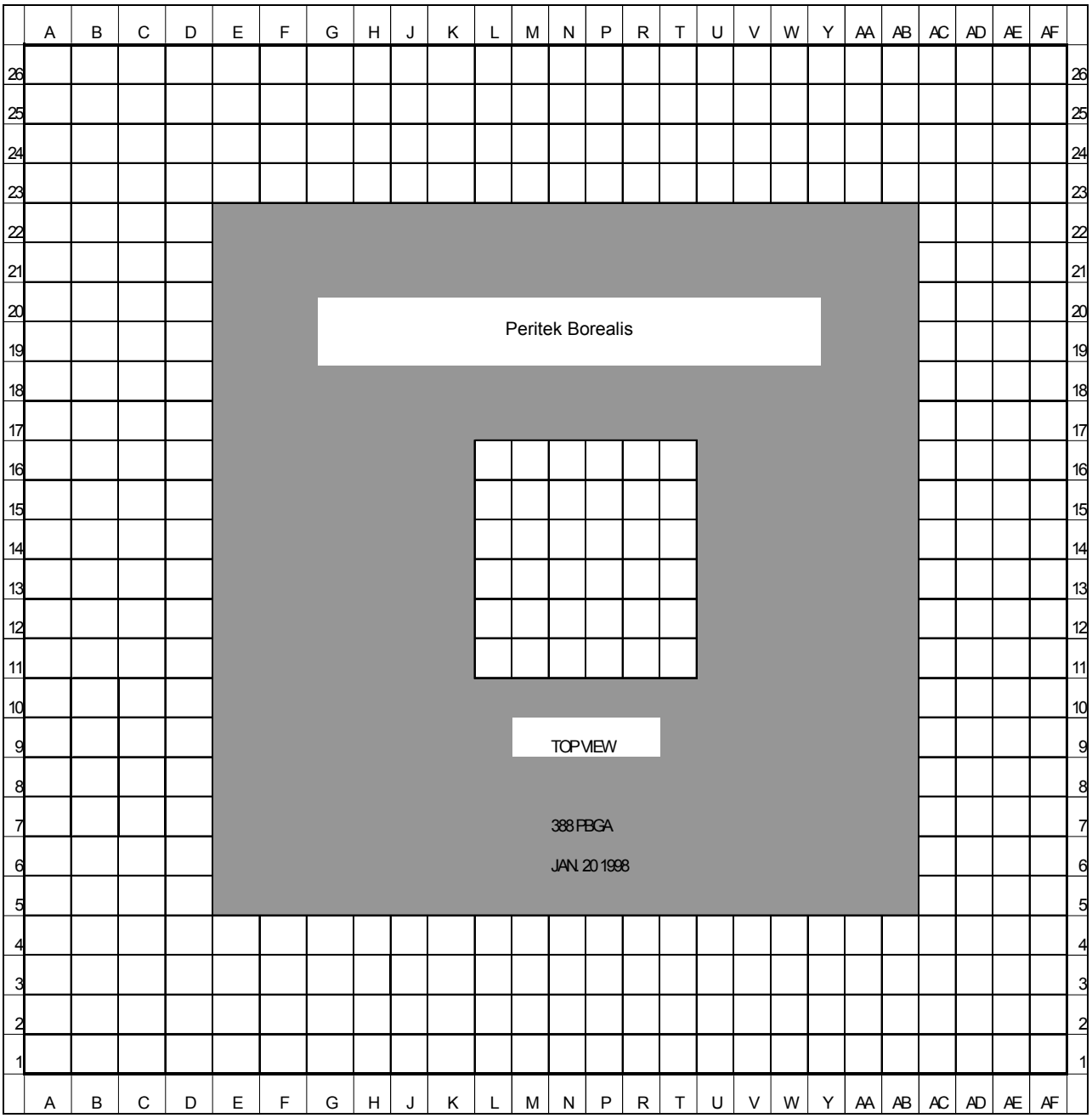
SIGNAL NAME	SOCKET
ST.1	B10
ST.2	C10
STOPn	F1
TRDYn	E4
VDD	AA23
VDD	AA4
VDD	AC11
VDD	AC16
VDD	AC21
VDD	AC6
VDD	D11
VDD	D16
VDD	D21
VDD	D6
VDD	F23
VDD	F4
VDD	L23
VDD	L4
VDD	T23
VDD	T4
VRASn.0 // CSn.0	AE24
VRASn.1 // CSn.1	AF24
VRASn.2 // CSn.2	AE3
VRASn.3 // CSn.3	AF3
VSS	A1
VSS	A2
VSS	A26
VSS	AC13
VSS	AC18
VSS	AC23
VSS	AC4
VSS	AC8
VSS	AD24
VSS	AD3
VSS	AE1
VSS	AE2
VSS	AE25
VSS	AF1
VSS	AF25
VSS	AF26
VSS	B2
VSS	B25
VSS	B26
VSS	C24
VSS	C3
VSS	D14
VSS	D19
VSS	D23
VSS	D4

Power Pins

VDD	D6,D11,D16,D21,F4,F23,L4,L23,T4,T23,AA4,AA23, AC6,AC11,AC16,AC21;

VSS	A1,A2,A26,B2,B25,B26,C3,C24,D4,D9,D14,D19, D23,H4,J23,L11,L12,L13,L14,L15,L16,M11,M12,M13,M14,M15,M16,N4, N11,N12,N13,N14,N15,N16,P11,P12,P13,P14,P15,P16,P23,R11,R12,R13, R14,R15,R16,T11,T12,T13,T14,T15,T16,V4,W23,AC4,AC8,AC13,AC18, AC23,AD3,AD24,AE1,AE2,AE25,AF1,AF25,AF26;

Figure 2-1.1. 388PBGA (Top View)


A B C D E F G H J K L M N P R T U V W Y AA AB AC AD AE AF
26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
Peritek Borealis
TOP VIEW
388 PBGA
JAN. 20 1998

Configuration Pins

As Borealis is being reset during the power up sequence, the logic states of *32 configuration pins* are used to configure specific functions. The values on configuration pins are latched into Borealis on the rising edge of the RST# signal and may be read at any time in the ID or CONFIG registers. The notation "CP[0]" corresponds to the logic state on CONFIGURATION_PIN[0] at the rising edge of RST#. Likewise, CP[27:26] would correspond to CONFIGURATION_PIN[27:26].
The configuration pins *are not* dedicated pins. They actually correspond to other functional pins of the chip as follows:
CONFIGURATION_PIN[15:0] use BENh[15:0] pins
CONFIGURATION_PIN[23:16] use PD[7:0] pins
CONFIGURATION_PIN[31:24] use PA[7:0] pins
The PA and PD pins are internally pulled down to logic zero. A value of logic one can be achieved on CP[31:16] through the use of an external pull-up resistor.
The BENh pins are internally pulled up to logic one. A value of logic zero can be achieved on CP[15:0] through the use of an external pull-down resistor.

Table 3-1.11.

PINS	NAME	VALUE	DESCRIPTION
CP[15:0]	SVID		Subsystem Vendor ID This 16 bit field is the PCI Subsystem Vendor ID. It can be read at address 0x2C in PCI configuration space. This field will correspond to CP[15:0] unless CP[16] is set to zero, in which case the SVID will be set to 0x105D; the vendor ID for Peritek Corporation
CP[16]	SSEL	 0 1	Subsystem Vendor ID Select This bit allows the Subsystem Vendor ID fields in PCI configuration space to be set according to CP[15:0]. Subsystem Vendor ID = 0x105D; Subsystem Vendor ID selected by CP[15:0]
CP[22:17]	SID		Subsystem ID AGP Sideband capability enable. 1 enable 0 disable This **6 bit** field corresponds to the lower 6 bits of the PCI Subsystem ID; the upper 10 bits of the SID will always be zero. It can be read at address 0x2E in PCI configuration space. This field will **always** correspond to CP[22:17] (regardless of the value of CP[16]).
CP[23]	HBT	 0 1	Host Bus Type AGP Bus PCI Bus It can be read in ID register bit[3]
CP[24]	SGR	 1	Local memory type: SGRAM/SDRAM It can be read in CONFIG2 REGISTER bit[1]
CP[25]	IDAC	 0 1	Internal RAMDAC enable External RAMDAC enabled Internal RAMDAC enabled It can be read in CONFIG2 REGISTER bit[0] This configuration pin selects default value (after reset) of CONFIG2 REGISTER bit[0].

PINS	NAME	VALUE	DESCRIPTION
CP[27:26]	DDEN	 00 01 10 11	Local buffer memory density/type reserved 256K bits by N memory chips 16MBit chips (512K bits by 32) - if SGRAM memory 16MB chips (2k x 256 x 2 x 16) - if SDRAM memory It can be read in ID REGISTER bits[9:8]
CP[28]	CLASS	 0 1	PCI Device Sub-Class VGA Other It can be read in ID REGISTER bit[28] Setting this configuration pin to a 1 disables the VGA component (i.e., VGA decodes will never occur regardless of VGA_CTRL settings).
CP[29]	EE	 0 1	EPROM Boot Enable: This pin provides the power-on default value for CONFIG1[13], which is the EPROM (RBASE_E) decode enable. EPROM Boot Decode disabled EPROM Boot Decode Enabled It can be read in ID REGISTER bit[30]
CP[31:30]	BASE0/1	 00 01 10 11	PCI Base 0 Address Register Size and PCI Base 1 Address Register Size 4 Megabyte Memory Space Requested 8 Megabyte Memory Space Requested 16 Megabyte Memory Space Requested 32 Megabyte Memory Space Requested These two bits select memory space requested for Linear Memory Window 0 and 1. **Both memory windows request same amount of memory**. Value of CP[31:30] can be read in ID REGISTER bits[7:6] or ID REGISTER bits[12:11]

Chapter 4: *PCI & AGP Configuration*

PCI/AGP Configuration Space

To be both PCI and AGP compliant, Borealis implements the required 256 bytes of configuration space. This configuration space allows configuration of the device by the system BIOS at boot time. This is sometimes referred to as "plug and play". These registers are read from and written to via special PCI configuration cycles.

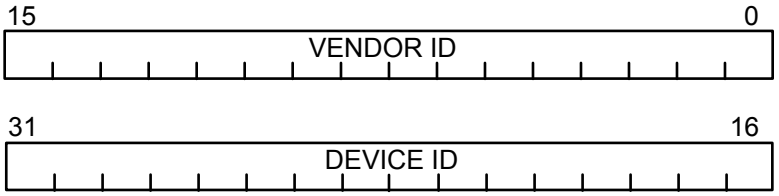
Borealis PCI Registers

The diagram on the following page illustrates the PCI configuration registers implemented within Borealis. The values in parentheses indicate power up default values except in the case of the base registers, where Borealis register mapping is indicated.

NOTE: Greyed out fields indicate unimplemented PCI functions and will always be read back as a zero value.

PCI Configuration Register 0 (00h)

Type: PCI configuration read only



BITS	NAME	DEFAULT	FUNCTION
[15:0]	VENDOR ID	105Dh	Company Identifier for Number Nine
[31:16]	DEVICE ID	5348h	Borealis Identifier

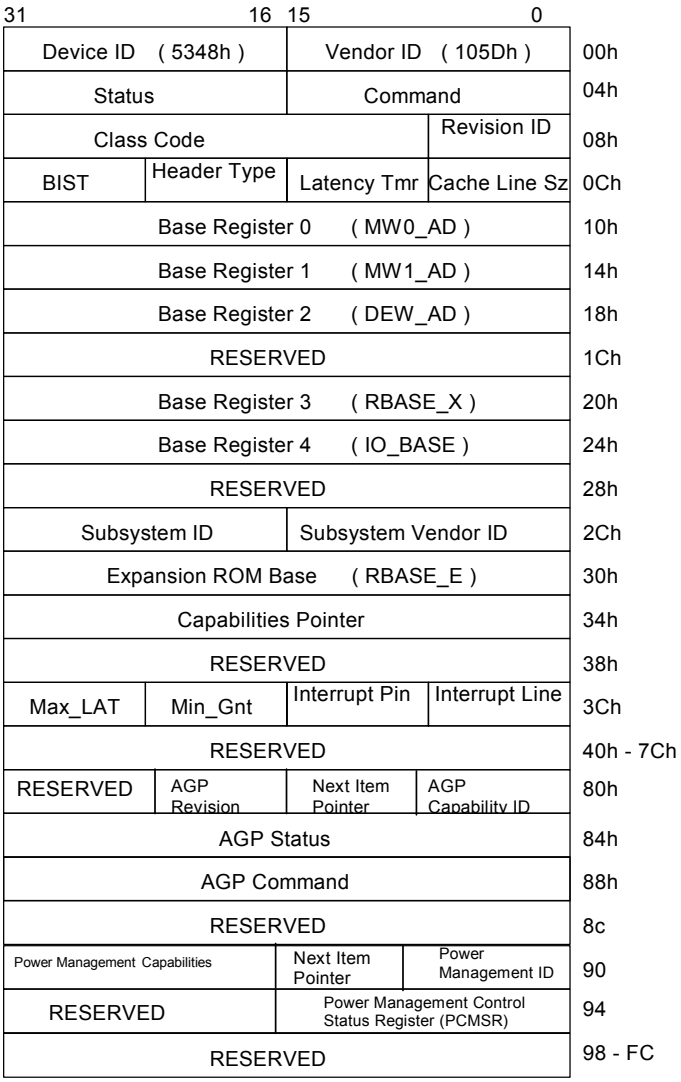
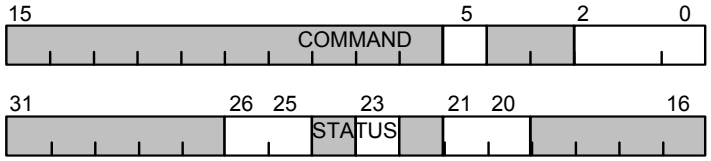


Figure 2-1.1. Borealis Configuration Registers

PCI Configuration Register 1 (04h)

Type: PCI configuration read write

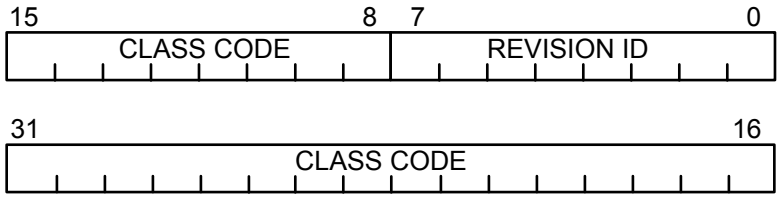


BITS	NAME	DEFAULT	FUNCTION
[15:0]	COMMAND	0	PCI Command Register
0	I/O Space	0	Allows Borealis to respond to I/O cycles when set to 1
1	Memory Space	0	Allows Borealis to respond to Memory cycles when set to 1
2	Bus Master	0	PCI Bus Master enable (write)
5	VGA Palette Snooping	0	Enables VGA Palette snooping when set to 1
[31:16]	STATUS	02B0h	PCI Status Register
[20]	Capabilities List	1	Device implements a Capabilities list (read-only)
[21]	66 MHz	1	66 MHz capable (read-only)
[23]	Fast back to back	1	Indicates that Borealis supports fast back to back transactions (**read only**)
[26:25]	DEVSEL Timing	01	These two bits indicate that Borealis will assert the PCI DEVSEL signal within the "MEDIUM" time frame. **(read only)**

PCI Configuration Register 2 (08h)

Type: PCI configuration read only

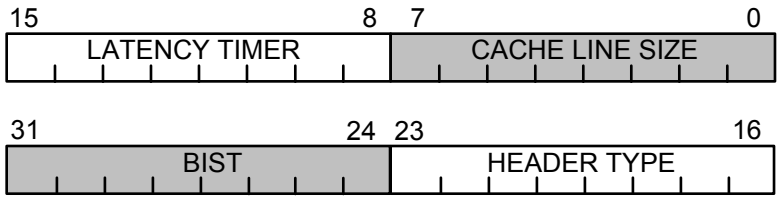
The sub-class field of the class code register is determined by configuration pin [28]. All other fields in this register are hardwired to their default values.



BITS	NAME	DEFAULT	FUNCTION
[7:0]	REVISION ID		Borealis Revision Number
[31:8] [31:24]	CLASS CODE Base Class	 03h	PCI DEVICE CLASS Borealis will always identify itself as a display controller which is base class 3.
[23:16]	Sub Class CP[28] = 0 CP[28] = 1	 00h 80h	The two sub-classes for display controllers supported by Borealis are: VGA compatible controller Other Display Controller
[15:8]	Programming Interface	00h	Always defined as 0

PCI Configuration Register 3 (0Ch)

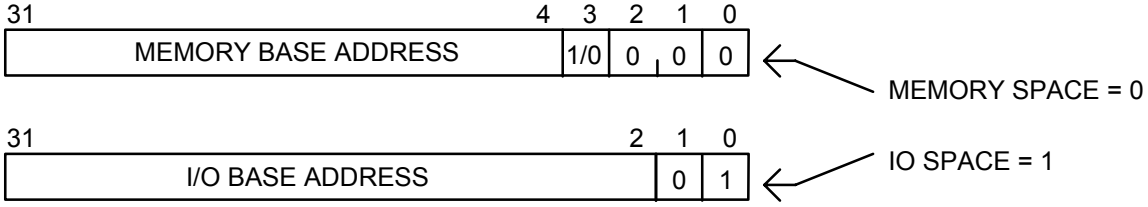
Type: PCI configuration read only



BITS	NAME	DEFAULT	FUNCTION
[7:0]	CACHE LINE SIZE	00h	Cache line size is not supported.
[15:8]	LATENCY TIMER	00h	PCI Bus Master Minimum Clock Count for a Burst Transaction before another master can gain bus grant.
[23:16]	HEADER TYPE	00h	Indicates that Borealis is a single function device with the default layout of configuration space.
[31:24]	BIST	00h	Device is not BIST capable.

Borealis PCI Base Address Registers

To allow for maximum configuration capabilities, PCI configuration space contains five base address registers, and one EPROM base address register. A PCI compliant device such as Borealis uses the base registers to inform the system of its memory and I/O requirements. The system then allocates memory and I/O resources at boot time to all PCI devices. The base address registers are loaded by the system to indicate which memory or I/O locations a board has been assigned. The general format of a base address register is shown below.

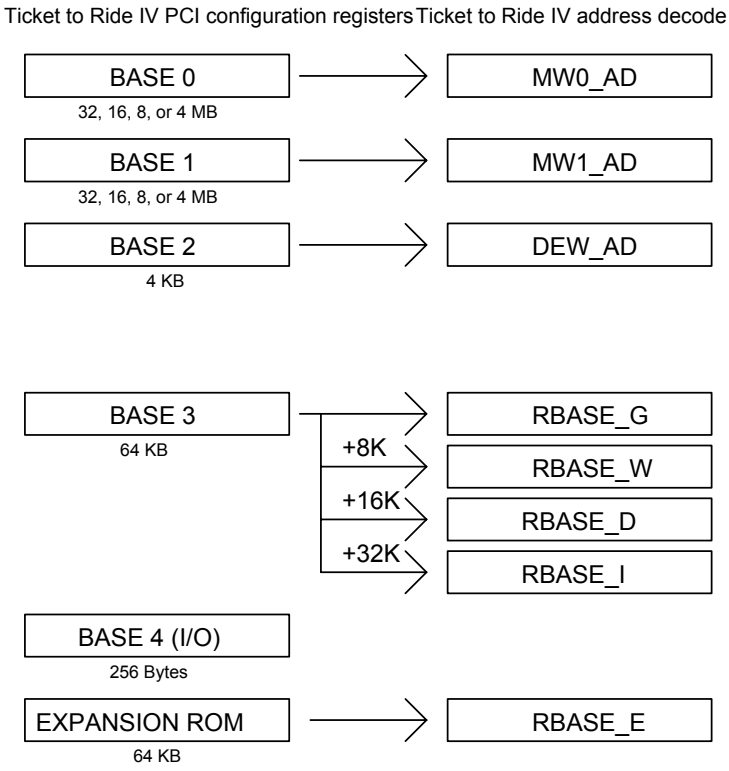


A PCI device sets bit 0 in a base register to one for I/O mapping and to zero for memory mapping. Bit 0 is read only. If the register is set to I/O, bit 1 must be read back as zero. The remaining bits (31:2) are used to specify the I/O address to which the device will respond. If the register is set to memory mapping, Borealis will set bits 1 and 2 both to zero indicating that the allocated memory space may be anywhere in 32 bit address space. Bit 3 indicates whether the area of memory corresponding to the base address register is pre-fetchable. Bit 3 is set to zero for all base registers except for the linear memory windows registers, where it is hardwired to 1.

The amount of memory or I/O space that Borealis requests from the host is determined by the number of upper order base address bits that are implemented. To request enough memory for a four megabyte memory window, for example, Borealis would implement base address bits (31:22) with the remaining bits (21:4) hardwired to zero. After allocating all PCI memory requests, the host would assign a four megabyte segment of memory to Borealis by setting bits (31:22) to the assigned memory space.

Each base register will allocate memory or I/O space for a specific Borealis function Borealis allocates two base registers for linear memory windows, one for the DRAWING ENGINE CACHE, one for memory mapped registers, one for I/O mapped registers, and one for memory mapped EPROM. *When a PCI base address register is written, the write will be shadowed to the associated internal register.* The diagram on the following page illustrates how Borealis's PCI configuration registers are shadowed to Borealis's address decode registers. For example, base address register 0 allocates memory for linear memory window 0. When the PCI system BIOS writes to base register 0, the write will also occur to MW0_AD, the starting address of memory window 0. The first 3 PCI address registers are shadowed to MW0_AD, MW1_AD, DEW_AD), respectively. Under most circumstances, there is no need to alter the values that have been written to MW0_AD, MW1_AD, DEW_AD(A), These registers are, however, accessible in Borealis memory space. The size of base registers 0 and 1 is determined by configuration pins (4, 8, 16, or 32 megabytes). *The size of other base registers is fixed.*

When base address register 4 is written, the value is directly shadowed to RBASE_G. RBASE_W will be written with the value in base 4 plus an offset of 8 kilobytes. RBASE_D, and RBASE_I are written as shown on the following page. RBASE_G is the pointer to the starting memory location of the global register block. RBASE_W points to the memory windows registers. RBASE_D points to the drawing engine. RBASE_I points to the interrupt registers. The "RBASE" registers are I/O mapped and pointed to by the value in base address register 4.



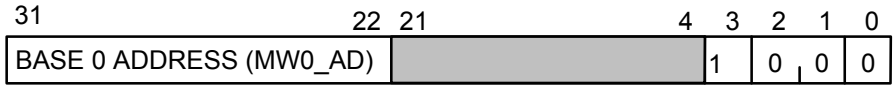
Whenever one of the PCI base registers is written, the corresponding RBASE register is updated. Updating the RBASE registers has no effect on the PCI base reigisters.

Figure 2-1.2. PCI Configuration Registers and Address Decode Registers

PCI Base Address Register 0 (10h)

Type: PCI configuration read write

This base address register requests 4, 8, 16, or 32 megabytes of memory space based on configuration pins[31:30]. It corresponds to the starting address of linear memory window 0. When this register is written, MW0_AD will also be written.

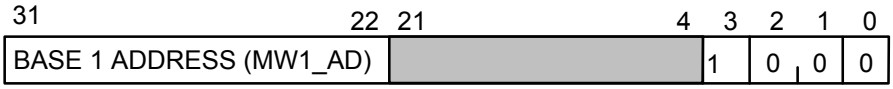


BITS	NAME	DEFAULT	FUNCTION
[2:0]	Base Address Type	0x0	Indicates that base address register 0 corresponds to a memory device **(read only)**
[3]	Memory Space pre-fetchable	1	Linear Memory Window 0 is pre-fetchable **(read only)**
[31:22] [31:23] [31:24] [31:25]	Base Address 0 CP[31:30] = 00 Base Address 0 CP[31:30] = 01 Base Address 0 CP[31:30] = 10 Base Address 0 CP[31:30] = 11		Address decode for 4 megabytes Address decode for 8 megabytes Address decode for 16 megabytes Address decode for 32 megabytes

PCI Base Address Register 1 (14h)

Type: PCI configuration read write

This base address register requests 4, 8, 16, or 32 megabytes of memory space based on configuration pins[31:30]. It corresponds to the starting address of linear memory window 1. When this register is written, MW1_AD will also be written.

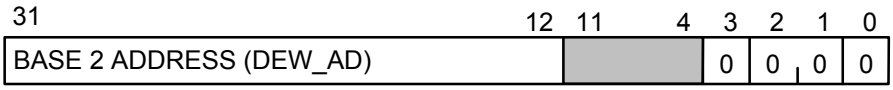


BITS	NAME	DEFAULT	FUNCTION
[2:0]	Base Address Type	0x0	Indicates that base address register 1 corresponds to a memory device **(read only)**
[3]	Memory Space pre-fetchable	1	Linear Memory Window 1 is pre-fetchable **(read only)**
[31:22] [31:23] [31:24] [31:25]	Base Address 1 CP[31:30] = 00 Base Address 1 CP[31:30] = 01 Base Address 1 CP[31:30] = 10 Base Address 1 CP[31:30] = 11		Address decode for 4 megabytes Address decode for 8 megabytes Address decode for 16 megabytes Address decode for 32 megabytes

PCI Base Address Register 2 (18h)

Type: PCI configuration read write

This base address register requests 4 KB of memory. It corresponds to the starting address of the memory window for drawing engine. When this register is written, DEW_AD will also be written.



BITS	NAME	DEFAULT	FUNCTION
[3:0]	Base Address Type	0x0	Indicates that base address register 2 corresponds to a non-prefetchable memory device **(read only)**
[31:12]	Base Address 2		Address decode for 4 kilobytes

PCI Base Address Register 3 (20h)

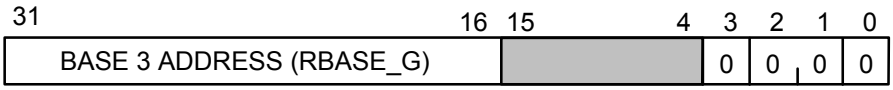
Type: PCI configuration read write

This base address register requests 64 kilobytes of memory space for the memory mapped Borealis registers. When this register is written, the write will also occur to RBASE_G, the base address of the Global register block. The following registers will also be loaded:

RBASE_W will be loaded with the value in RBASE_G plus 8 kilobytes

RBASE_D will be loaded with the value in RBASE_G plus 16 kilobytes

RBASE_I will be loaded with the value in RBASE_G plus 32 kilobytes

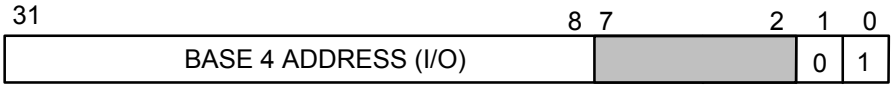


BITS	NAME	DEFAULT	FUNCTION
[3:0]	Base Address Type	0x0	Indicates that base address register 3 corresponds to a non-prefetchable memory device **(read only)**
[31:16]	Base Address 3		Address decode for 64 kilobytes

PCI Base Address Register 4 (24h)

Type: PCI configuration read write

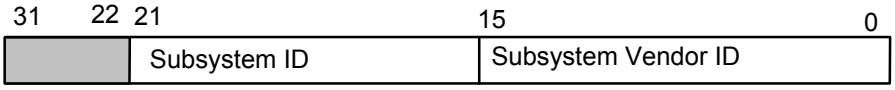
This base address register requests 256 bytes of I/O space for the I/O mapped Borealis registers. This register is not shadowed to any other registers.



BITS	NAME	DEFAULT	FUNCTION
[1:0]	Base Address Type	01	Indicates that base address register 4 corresponds to I/O space. **(read only)**
[31:8]	Base Address 4		Address decode for 256 bytes

PCI Configuration Register 4 (2Ch)

Type: PCI configuration read only

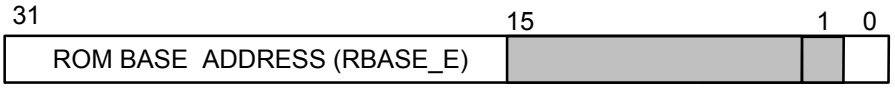


BITS	NAME	DEFAULT	FUNCTION
[15:0]	Subsystem Vendor ID		Value in this register is selected by Configuration pin CP[16]. If CP[16] = 0 Subsystem Vendor ID[15:0] = 105Dh If CP[16] =1 Subsystem Vendor ID[15:0] contain settings from configuration pins CP[15:0].
[21:16]	Subsystem ID		Configuration pins CP[22:17] provide value for this 6 bit field. It may be used to get information about board configuration, other devices on the board etc. (CP[16] has no effect for this field!) **NOTE**: CP22 is AGP sideband enable, therefore, all 6 bit SIDs will be 1xxxxxb is sideband is enabled.

PCI ROM Base Address Register (30h)

Type: PCI configuration read write

The ROM base address register requests 64 kilobytes of memory space for a PROM device. When this register is written, the write will also occur to RBASE_E, the EPROM base address register. Bit 0 of this register is used by the PCI BIOS to enable or disable EPROM decode. The EPROM Enable bit in the CONFIG1 register must also be set for EPROM decode to be active.

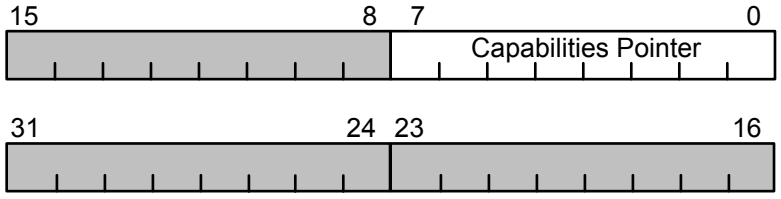


BITS	NAME	DEFAULT	FUNCTION
[0]	PCI ROM Enable	0 1	Disable EPROM address decode (default) Enable EPROM address decode
[31:16]	ROM Base Address		ROM Address decode for 64 kilobytes

PCI Configuration Register 5 (34h)

Type: PCI configuration read only

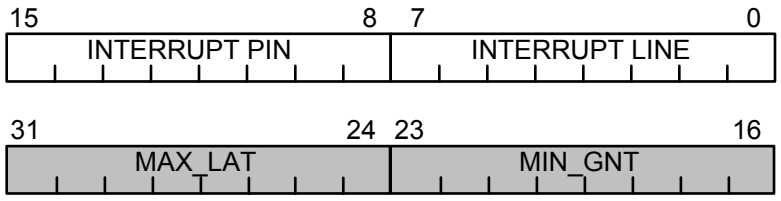
Borealis implements a capabilities list (linked list) to support its AGP and Power Management features. The list is located in Configuration Space at offset 80h. The Capabilities Register contains a pointer to that list.



BITS	NAME	DEFAULT	FUNCTION
[7:0]	Capabilities Pointer	80h	Pointer to a Capabilities List in Configuration Space.
[31:8]	Reserved		Reserved.

PCI Configuration Register 6 (3Ch)

Type: PCI configuration read write

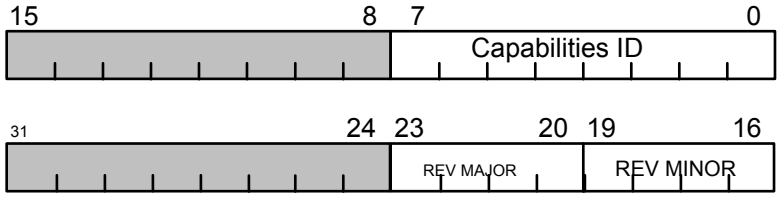


BITS	NAME	DEFAULT	FUNCTION
[7:0]	INTERRUPT LINE		Indicates which input of the system interrupt controller Borealis's interrupt signal is connected to. (Written by system software)
[15:8]	INTERRUPT PIN	01h	Indicates that Borealis's interrupt pin is connected to PCI connector pin INTA#.**(read only)**

PCI Configuration Register 7(80h)

Type: PCI configuration read only

The T2R IV has the ability to act as an AGP Master. This register is the beginning of the AGP Capabilities Structure.

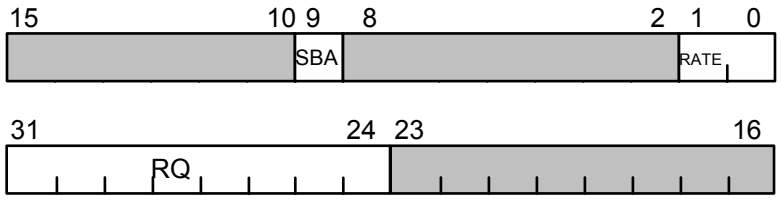


BITS	NAME	DEFAULT	FUNCTION
[7:0]	AGP Capabilities ID	02h	Identifies Borealis as an AGP device.
[15:8]	Next Item Pointer	90h	Pointer to PCI Power Management Capabilities Structure.
[19:16]	REV MINOR	0h	Minor AGP spec. revision that device complies to.
[23:20]	REV MAJOR	1h	Major AGP spec. revision that device complies to.
[31:24]	Reserved		Reserved

PCI Configuration Register 8 (84h)

Type: PCI configuration read only

This is the AGP Status Register and is part of the AGP Capabilities Structure.



BITS	NAME	DEFAULT	FUNCTION
[1:0]	Rate	11	Device supports both 1x & 2x AGP transfers.
[8:2]	Reserved		Reserved.
[9]	SBA (CP22)	0	Side Band Addressing Capabilities Bit Config Pin 22 (CP22).
[23:10]	Reserved		Reserved.
[31:24]	RQ	0Fh	We can have 16 outstanding AGP requests at a time. (00h = 1; FFh = 256.) Do not set to 0, 1, 2 or greater than F.

PCI Configuration Register 9 (88h)

Type: PCI configuration read write

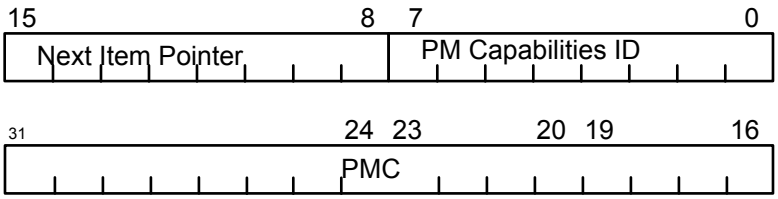
This is the AGP Command Register and is part of the AGP Capabilities Structure.

BITS	NAME	DEFAULT	FUNCTION
[1:0]	Data Rate	00	Determines AGP data transfer rate. (Set only one bit. Bit0: 1x, Bit1: 2x)
[7:2]	Reserved.		Reserved.
[8]	AGP EN	0	Enable AGP functions
[9]	SBA EN	0	Enable side band addressing
[23:10]	Reserved		Reserved.
[31:24]	RQ DEPTH	00h	Set the Request Queue depth. **NOTE**: This should not exceed 0Fh or the RQ of the corelogic.

PCI Configuration Register 10 (90h)

Type: PCI configuration read only

The T2R IV is a PCI Power Management Device. This register is the beginning of the PCI Power Management Capabilities Structure.

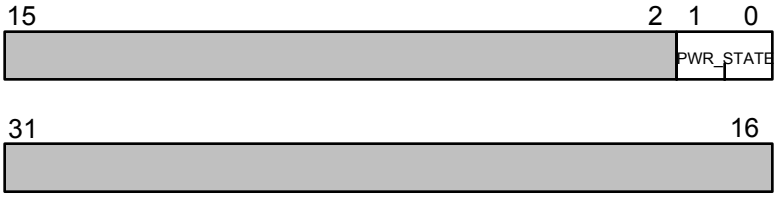


BITS	NAME	DEFAULT	FUNCTION
[7:0]	PM Capabilities ID	01h	Identifies Borealis as a PCI Power Management device.
[15:8]	Next Item Pointer	00h	This structure is the last in the linked list.
[31:16]	PMC	21h	Power Management Capabilities T2R IV conforms to the PCI PM specification 1.0, requires device specific initialization (DSI) and does not support PME#, D1, or D2.

PCI Configuration Register 11 (94h)

Type: PCI configuration read write

This is the PCI Power Management Control Status Register.



BITS	NAME	DEFAULT	FUNCTION
[1:0]	PWR_STATE	0	PMCSR Power State 0 D0 1 Reserved 2 Reserved 3 D3hot
[31:2]	Reserved		Reserved.

Chapter 5: *Register Set*

Addressing Configuration Registers

NOTE: Undefined bits in the registers are always read back to zero and should be written as zero.

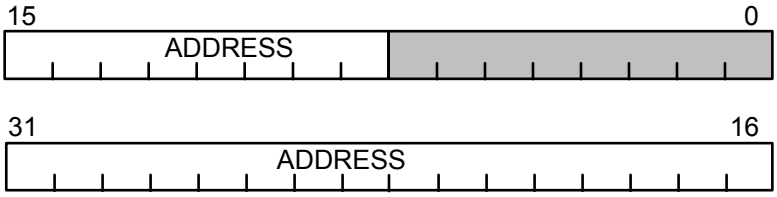
The addressing configuration group of registers are I/O mapped. The address of these registers is defined as follows. The value written into PCI base address Register 4 is concatenated with the fixed address of the register to determine the system address of the register.

Register Base Address for the Global Register Block {PCIB4, (0x0000)}

Name: RBASE_G

Type: I/O space read write

This register defines the start address for the memory mapped global register block. This register will be written with the same value as Base Address Register 4.



BITS	NAME	DEFAULT	FUNCTION
RBASE_G[31:8]	ADDRESS		Address decode value

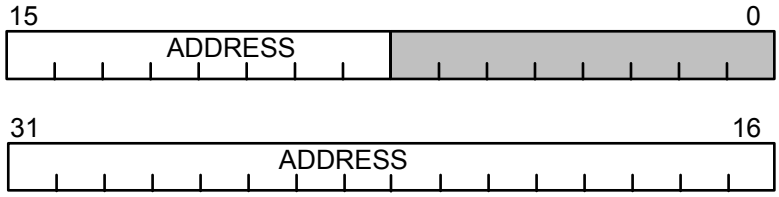
NOTE: PCIB4 = The value written into PCI Base 4 Register.

Memory Windows™ Register Block Base Address Register {PCIB4, (0x0004)}

Name: RBASE_W

Type: I/O space read write

This register defines the start address for the Memory Windows configuration register block. This register will be written with the same value as Base Address Register 4 plus an 8 kilobyte offset.



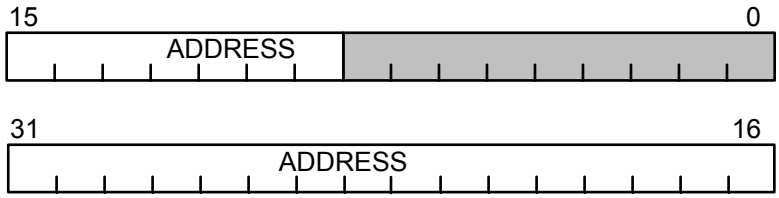
BITS	NAME	DEFAULT	FUNCTION
RBASE_W [31:8]	ADDRESS		Address decode value

Register Base Address Drawing Engine {PCIB4,(0x0008)}

Name: RBASE_D

Type: I/O space read write

This register defines the start address for the memory mapped register block. This register is written with the same value as Base Address Register 4 plus a 16 kilobyte offset.



BITS	NAME	DEFAULT	FUNCTION
RBASE_D[31:9]	ADDRESS		Address decode value

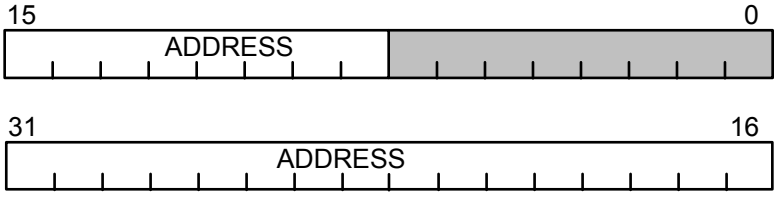
NOTE: PCIB4 = The value written into PCI Base 4 Register.

Register Base Address Global Interrupt Registers {PCIB4,(0x0010)}

Name: RBASE_I

Type: I/O space read write

This register defines the start address for the memory mapped global interrupt register block. This register will be written with the same value as Base Address Register 4 plus a 32 kilobyte offset.



BITS	NAME	DEFAULT	FUNCTION
RBASE_I[31:8]	ADDRESS		Address decode value

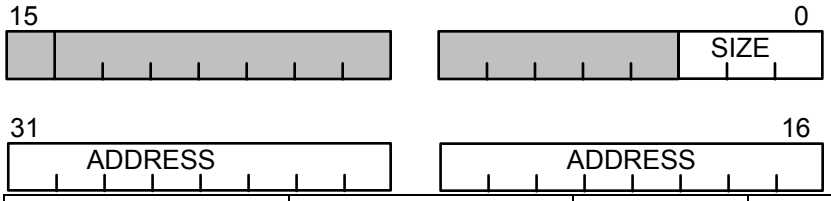
NOTE: PCIB4 = The value written into PCI Base 4 Register.

Register Base Address/Size EPROM registers {PCIB4, (0x0014)}

Name: RBASE_E

Type: I/O space read write

This register defines the start address for the memory mapped EPROM. This register will be written with the same value as the expansion ROM Base Address Register and the size will be set to 64KB.



BITS	NAME	DEFAULT	FUNCTION
RBASE_E[31:16]	ADDRESS	0x0000	EPROM Address decode value
RBASE_E[2:0]	SIZE	0x1	EPROM Size 64 KB (**read only**)

NOTE: PCIB4 = The value written into PCI Base 4 Register.

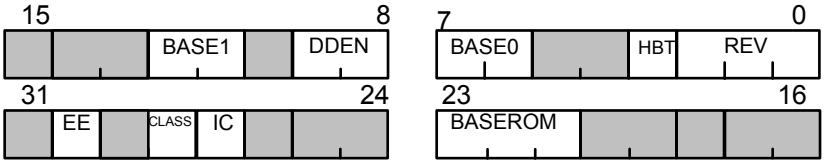
Miscellaneous I/O Registers

ID Register {PCIB4,(0x0018)}

Name: ID

Type: I/O space mapped read only

The ID Register contains the hardwired chip revision as well as configuration information. Configuration pins, as indicated, are used to initialize most of the fields in this register.



BITS	NAME	CONFIG. PIN	VALUE	DESCRIPTION
ID[2:0]	REV			Chip Revision
ID[3]	HBT	CP[23]=0 CP[23]=1	0x0 0x1	Host bus type AGP Bus PCI Local Bus
ID[7:6]	BASE0	CP[31:30] = 00 CP[31:30] = 01 CP[31:30] = 10 CP[31:30] = 11	0x0 0x1 0x2 0x3	PCI Base 0 Address Register Size (Linear Memory Window 0) 4 Megabyte Memory Space Requested 8 Megabyte Memory Space Requested 16 Megabyte Memory Space Requested 32 Megabyte Memory Space Requested
ID[9:8]	DDEN	CP[27:26] = 00 CP[27:26] = 01 CP[27:26] =10 CP[27:26] = 11	0x0 0x1 0x2 0x3	Display buffer density Reserved 256K bits by N memory chips if **SGRAM/SDRAM** - 16Mbit chips (1k x 256 x 2 x 32) if **SDRAM** - 16Mbit chips (2k x 256 x 2 x 16)
ID[12:11]	BASE1	CP[31:30] = 00 CP[31:30] = 01 CP[31:30] = 10 CP[31:30] = 11	0x0 0x1 0x2 0x3	PCI Base 1 Address Register Size (Linear Memory Window 1) 4 Megabyte Memory Space Requested 8 Megabyte Memory Space Requested 16 Megabyte Memory Space Requested 32 Megabyte Memory Space Requested
ID[23:21]	BASEROM		0x1	PCI EPROM Base Address Register Size 64 Kilobyte Memory Space Requested
ID[27]	IC		1	PCI Interrupt Capability is on

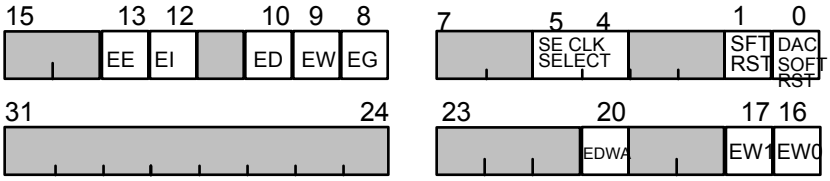
BITS	NAME	CONFIG. PIN	VALUE	DESCRIPTION
ID[28]	CLASS	CP[28] = 0 CP[28] = 1	0x0 0x1	PCI Device Sub-Class VGA Other Display Controller
ID[30]	EE	CP[29] = 0 CP[29] = 1	0x0 0x1	EPROM Boot Decode Enable: This is the read of CP[29] for EPROM boot enable bit. EPROM Boot Decode disabled EPROM Boot Decode Enabled

Configuration Register One {PCIB4,(0x001C)}

Name: CONFIG1

Type: I/O space mapped read write

The decoder enable bits allow certain address decode functions to be individually enabled. Before any address decode register is programmed, the corresponding decode enable bit should be set to 0 (disabled), and then set to 1 after the address range has been programmed. Bit[0] or Bit[1] allow for soft reset generation.



BITS	NAME	VALUE	DESCRIPTION
CONFIG1[0]	DAC_SOFT _RESET	0 1	DAC and Borealis Software reset bit. Default, No software reset to the DAC and Borealis. Software reset the DAC and Borealis.
CONFIG1[1]	SFT_RST	0 1	Software reset bit. No software reset generated. (default) Software reset generated.
CONFIG1[3:2]	Reserved		Reserved
CONFIG1[5:4]	SE_CLOCK _SELECT	00 01 10 11	Setup Engine Clock Source Select Default, Host Bus Clock as SE Clock Internally derived 2X Host Bus Clock (for use only in PCI 33MHz systems) Internally derived DE_CLK/2 SE_CLK pin
CONFIG1[7:6]	Reserved	0	Reserved

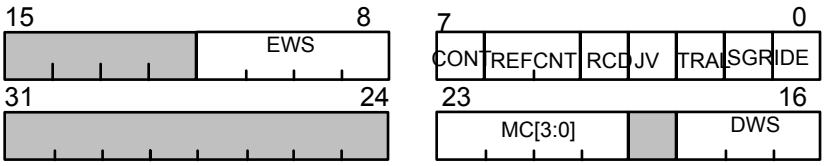
BITS	NAME	VALUE	DESCRIPTION
CONFIG1[20:8]	ENABLE DECODER	0 1 (Defaults)	Disable Decode for a specific block Enable Decode for a specific block
CONFIG1[8]	EG	0	Enable Global Decoder.
CONFIG1[9]	EW	0	Enable Mem Windows Reg Decoder.
CONFIG1[10]	ED	0	Enable Drawing Engine Decoder.
CONFIG1[11]	Reserved	0	Reserved
CONFIG1[12]	EI	0	Enable Global Interrupt Decoder.
CONFIG1[13]	EE	CP[29]	Enable EPROM Decoder.
CONFIG1[15:14]	Reserved	0	Reserved
CONFIG1[16]	EW0	0	Enable Mem Window 0 Decoder.
CONFIG1[17]	EW1	0	Enable Mem Window 1 Decoder.
CONFIG1[19:18]	Reserved	0	Reserved
CONFIG1[20]	EDWA	0	Enable Drawing Engine Window Decoder.
CONFIG1[31:21]	Reserved	0	Reserved

Configuration Register Two {PCIB4,(0x0020)}

Name: CONFIG2

Type: I/O space mapped read write

This register contains configuration information for Borealis and peripheral devices supported by Borealis. All bits are write/read except SGR bit which specifies the memory type in the local buffer and is set by Configuration Pin CP[24].



BITS	NAME	VALUE	DESCRIPTION
CONFIG2[0]	IDE (CP[25])	0 1	INTERNAL RAMDAC ENABLE Internal RAMDAC and PLLs disabled Internal RAMDAC and PLLs enabled
CONFIG2[1]	SGR (CP[24])	1	This bit (**read only**) defines the type of memory timing generated for the local buffers. Generate SGRAM/SDRAM/SDRAM timing.
CONFIG2[2]	TRAL	0 1	Extended RAS low timing Normal timing (default) Extended timing
CONFIG2[3]	JV	0	Reserved
CONFIG2[4]	RCD	0 1	RAS to CAS delay select , this bit defines the delay between the falling edge of RAS and the falling edge of CAS. Short delay Normal delay (default) **NOTE**: Typically this bit should be set to one.
CONFIG[6:5]	REFCNT	00 01 10 11	Refresh Count Select, these two bits define how often the DRAM refresh cycles occur. Generate DRAM refresh every 768 mclocks Generate DRAM refresh every 1024 mclocks Generate DRAM refresh every 1280 mclocks (default) Generate DRAM refresh every 3584 mclocks
CONFIG2[7]	CONT	0 1	Continuous Memory Cycle Enable This bit determines whether Borealis will concatenate multiple memory request that fall within the same row into a single memory cycle. All new memory requests result in RAS precharge New memory request may be combined (default). Typically this bit should be set to 1.

BITS	NAME	VALUE	DESCRIPTION
CONFIG2[11:8]	EWS		EPROM Wait States : Each EPROM wait state adds one Memory Clock to the EPROM Access time. See Borealis timing diagrams for more information.
		0x0	Zero EPROM wait states
		0x1	One EPROM wait states
		0x2	Two EPROM wait states
		0x3	Three EPROM wait states
		•	•
		•	•
		•	•
		.0xD	Thirteen EPROM wait states
		0xE	Fourteen EPROM wait states
		0xF	Fifteen EPROM wait states (Default)
CONFIG2[18:16]	DWS		DAC Wait States. These bits control WRITE/READ signals to internal and external RAMDAC as follows:
		0x0	WR/RD low = 3 clocks ; WR/RD high min. = 4 clocks
		0x1	WR/RD low = 3 clocks ; WR/RD high min. = 16 clocks
		0x2	WR/RD low = 5 clocks ; WR/RD high min. = 4 clocks
		0x3	WR/RD low = 5 clocks ; WR/RD high min. = 16 clocks
		0x4	WR/RD low = 7 clocks ; WR/RD high min. = 4 clocks
		0x5	WR/RD low = 7 clocks ; WR/RD high min. = 16 clocks
		0x6	WR/RD low = 9 clocks ; WR/RD high min. = 4 clocks
		0x7	WR/RD low = 9 clocks ; WR/RD high min. = 16 clocks Default is 0x3. Clocks refer to PCI interface clock. For optimum performance of the internal RAMDAC set this register to: 3 - if 33MHz PCI bus 5 - if AGP or 66MHz PCI bus
CONFIG2[20]	MC[0]		Memory Control 0 - Display Buffer Enable This bit will tristate all Display Buffer control signals (address, data, control).
		0	Display Buffer signals are tristate (*Default*)
		1	Display Buffer signals are driven
CONFIG2[21]	MC[1]		Memory Control 1 - RAM Refresh Control This bit controls whether DRAM refresh cycles are generated by Borealis
		0	RAM Refresh cycles generated (*Default*)
		1	RAM refresh disabled
CONFIG2[22]	MC[2]		Memory Control 2 - Data Sampling Control This bit controls the sampling of data during read cycles from a Borealis memory buffer. Normal Data sampling (*Default*) Delayed Data sampling Typically this bit should be set to:
		1	Special configuration
		0	SGRAM/SDRAM configuration

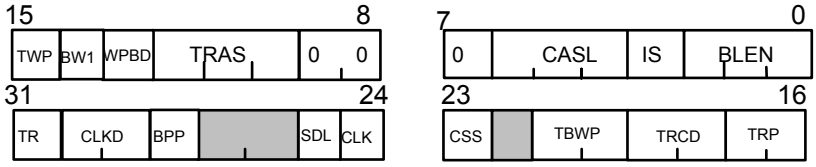
BITS	NAME	VALUE	DESCRIPTION
CONFIG2[23]	MC[3]	1 0	Memory Control 3 - Memory Control Skew This bit causes certain memory control signals to be skewed to allow more access time at higher MCLK rates. Normal memory control signals (*Default*) Typical value of this bit should remain zero.

SGRAM/SDRAM Configuration Register {PCIB4,(0x0024}

Name: SGR_CONFIG

Type: I/O space mapped read write

Programs SGRAM/SDRAM memories.
After a power up cycle, but before any of local memory buffers have been accessed for the very first time, the SGRAM/SDRAM memories must be initialized with a special configuration cycle. This configuration cycle may be repeated at any time later if a new set of SGRAM/SDRAM parameters is required. For example: switching from VGA mode to a high resolution mode may require reprogramming SGRAM/SDRAM.
Bit [31] of this register is a "trigger bit". Any time a "one" is written to this bit, the configuration cycle begins and the other bits in this register define SGRAM/SDRAM parameters. No other cycles are allowed to occur at the time when the configuration cycle is pending.



BITS	NAME	DEFAULT VALUE	DESCRIPTION
SGR_CONFIG[2:0]	BLEN	0	Has to be always set to 000, (this setting programs SGRAM/SDRAM memories for burst length of one).
SGR_CONFIG[3]	IS	0	Interleaved or sequential access between banks. IS has to be set to 0 (sequential access).
SGR_CONFIG [6:4]	CASL	0	CAS Latency in memory clocks. (In VGA mode set CASL to 010)
SGR_CONFIG[9:7]	vendor special op	0	Write 000 to these bits.

BITS	NAME	DEFAULT VALUE	DESCRIPTION
SGR_CONFIG[12:10]	TRAS	0	RAS active time (minimum). Value in this register should be calculated based on tRAS parameter from SGRAM/SDRAM specification and the actual period of MCLK (memory clock). TRAS =(tRAS/MCLK) -1 Then round up the result to an integer number. Example: tRAS=70ns, MCLK period=11ns. TRAS=70/11-1=5.36=>6. Write TRAS=110.
SGR_CONFIG[13]	WPBD	0	0- writes per bit enabled 1 -writes per bit disabled
SGR_CONFIG[14]	BW1	0	0 - two clocks block writes 1 - one clock block writes
SGR_CONFIG[15]	TWP	0	0 - two clocks from write to precharge 1 -one clock from write to precharge
SGR_CONFIG[17:16]	TRP	00	Precharge cycle (in clocks) 00 - 3 clocks 01 - 1 clock 10 - 2 clock 11 - 3 clocks
SGR_CONFIG[19:18]	TRCD	00	TRCD time (in clocks) 00 - 3 clocks 01 - 1 clock 10 - 2 clocks 11 - 3 clocks
SGR_CONFIG[21:20]	TBWP	00	Block write to precharge time (in clocks) 00 - 2 clock 01 - 1 clock 10 - 3 clocks 11 - 3 clocks
SGR_CONFIG[23]	CSS	0	Chip select skew 0 -normal 1 -delayed
SGR_CONFIG[24]	CLK	0	This bit enables PLL controlling skew between internal and external clock. 0 - PLL disabled (a clock from REFCLK pin is sent instead) 1 - PLL enabled (enable this bit after memory clock synthesizer is properly programmed and stable)
SGR_CONFIG[25]	SDL	0	Read data sampling delay 0 - normal 1 - delayed This bit is OR'ed with CONFIG2[22] bit , so any of these two bits if set to one will delay sampling.

BITS	NAME	DEFAULT VALUE	DESCRIPTION
SGR_CONFIG[28]	BPP	0	This bit enables PLL controlling skew between internal and external clock. 0 - PLL enabled 1 - PLL disabled (delayed internal clock is sent to memories, but the skew is not controlled by the PLL)
SGR_CONFIG[30:29]	CLKD	0	Clock delay control 00 - 2 ns + 0 ns 01 - 2ns + 1ns 10 - 2ns + 2 ns 11 - 2 ns + 3 ns
SGR_CONFIG[31]	TR		A write only trigger bit. A "one" written to this bit initialize configuration cycle. The memory controller has to be in idle state to execute this cycle. (Turn off ram refresh, screen/CRT refresh, drawing engine has to be idle)

NOTE: Bits[23:10] and bit[25] are "don't care" in VGA mode.

Typically the value in SGR_CONFIG[31:0] register should be set to:

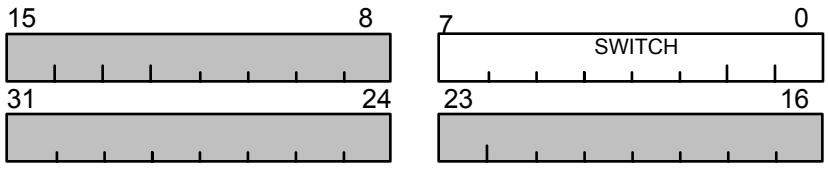
- 8Mbit memories 128kx32x2 -8 (Samsung KM4132G271A-8)
- 0xA100_0020 in VGA mode
- 0xA108_9030 in Imagine mode (83MHz <clock <= 100 MHz).
- 0xA10A_8c20 in Imagine mode at 83 MHz and below
- 16Mbit memories 256kx32x2 -H (Samsung KM4132G512-H)
- 0xA100_0020 in VGA mode.
- 0xA11A_D020 in Imagine mode (83MHz <clock <= 100 MHz).
- 16Mbit memories 256kx32x2 -8 (Samsung KM4132G512-8)
- 0xA118_d430 for -8 chips 100 MHz up

Soft Switch Register {PCIB4, (0x0028}

Name: SOFT_SW

Type: I/O space mapped read write

This register contains the data that was written to the external soft switch register. If no switches are present, this can be used as a general purpose register.



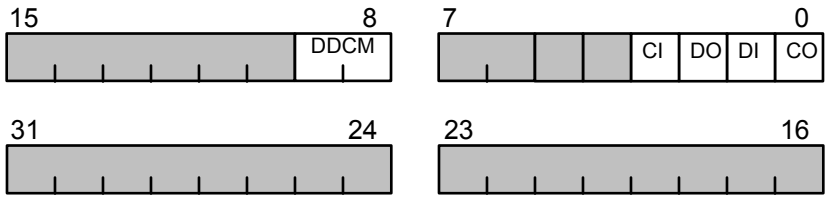
BITS	NAME	DEFAULT VALUE	DESCRIPTION

SOFT_SW[7:0]	SWITCH	0x00	This register contains the same data as the external soft switch register.

DDC Register {PCIB4, (0x002C)}

Name: DDC

Type: I/O mapped read write



BITS	NAME	VALUE	FUNCTION
DDC[0]	C0		DDC CLK OUT. In DDC1 mode, this bit connects to the monitor vsync signal. In DDC2B mode, this bit connects to the SCL line. 0x0 Forces clock signal low. 0x1 Forces clock signal to "Z" External pull-up resistor will cause the clock to stay high if no other device is driving this signal low.
DDC[1]	DI		DDC Data Line Status (read only) (SDA). This bit shows actual value on the serial data line (DDC1 and/or DDC2B).
DDC[2]	DO		DDC2B Data OUT (SDA) In DDC2B mode this bit connects to the Serial Data line. This bit is N/A in DDC1 mode. 0x0 Forces data signal low. 0x1 Forces data signal to "Z" External pull-up resistor will cause the data signal to stay high if no other device is driving this signal low.
DDC[3]	CI		DDC2B Clock Line Status (read only). (SCL) This bit shows actual value on the serial clock line (SCL) .
DDC[9:8]	DDCM		DDC Mode. 0x0 DDC is disabled. 0x1 DDC1 is enabled. 0x2 DDC2B is enabled. 0x3 DDC is disabled.

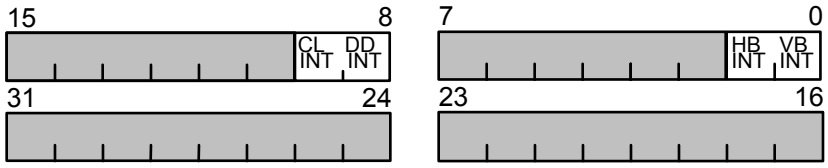
Global Interrupt Registers

Global Interrupt Register RBASE_I+(0x0000)

Name: GINTP

Type: Memory mapped read write

This register contains the concatenation of all the Borealis interrupts. The drawing engine interrupts in this register are read only and must be cleared in their respective interrupt registers.



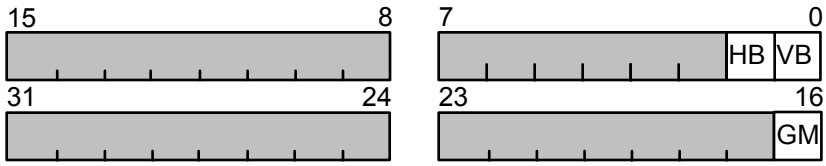
BITS	NAME	VALUE	FUNCTION
GINTP[0]	VB_INT	0 1	Vertical blank count match has not occurred Vertical blank count match has occurred **NOTE**: See also INT_VCNT register in CRT Registers section
GINTP[1]	HB_INT	0 1	Horizontal blank count match has not occurred Horizontal blank count match has occurred **NOTE**: See also INT_HCNT register in CRT Registers section
GINTP[8]	DD_INT	0 1	Drawing engine operation not complete. Drawing engine operation is complete.
GINTP[9]	CL_INT	0 1	Drawing engine clip interrupt has not occurred. Drawing engine clip interrupt has occurred.

Global Interrupt Mask Register RBASE_I+(0x0004)

Name: GINTM

Type: Memory mapped read write

This register contains the interrupt mask bits for the "VB_MSK" and the "HB_MSK" interrupts.



BITS	NAME	VALUE	FUNCTION
GINTM[0]	VB_MSK		Vertical Blank Mask
		0	Vertical blank count interrupt disabled. (*Default*)
		1	Vertical blank count interrupt enabled.
GINTM[1]	HB_MSK		Horizontal Blank Mask
		0	Horizontal blank count interrupt disabled. (*Default*)
		1	Horizontal blank count interrupt enabled.
GINTM[16]	GM		Global Interrupt mask
		0	Disable all interrupts. (*Default*)
		1	Enable un-masked interrupts

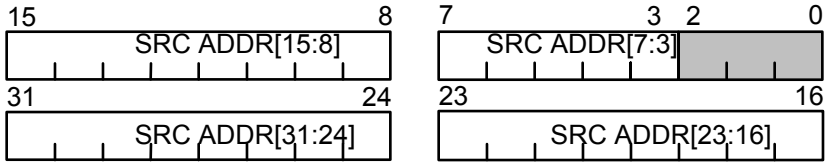
AGP DMA Registers

DMA registers are located in both I/O and RBASE_I space. The T2R IV is capable of generating DMA reads from system memory to local memory using AGP bus transactions. **Note**: A Qword is 64 bits.

DMA Source Address Register PCIB4 or RBASE_I + (0x00D0)

Name: DMA_SRC

Type: Memory or I/O mapped read write

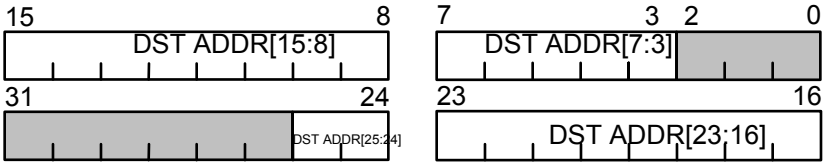


BITS	NAME	VALUE	FUNCTION
DMA_SRC[2:0]	Reserved		Reserved
DMA_SRC[31:3]	SRC_ADDR		Source Address - address of 1st Qword in system memory.

DMA Destination Address Register PCIB4 or RBASE_I + (0x00D4)

Name: DMA_DST

Type: Memory or I/O mapped read write



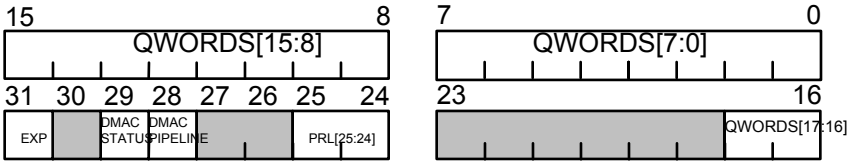
BITS	NAME	VALUE	FUNCTION
DMA_DST[2:0]	Reserved		Reserved
DMA_DST[25:3]	DST_ADDR		Destination Address - address of 1st Qword in local memory.
DMA_DST[31:26]	Reserved		Reserved

DMA Command Register PCIB4 or RBASE_I + (0x00D8)

Name: DMA_CMD

Type: Memory or I/O mapped read write

NOTE: Any work to this register triggers an AGP DMA transaction.



BITS	NAME	VALUE	FUNCTION
DMA_CMD[17:0]	QWORDS		Number of Qwords requested **NOTE**: Do not set to zero.
DMA_CMD[23:18]	Reserved		Reserved
DMA_CMD[25:24]	PRL	 00 01 10 11	Preferred Request Length for AGP transfers 4 Qwords 8 Qwords Reserved Reserved (See Appendix E, *Errata* for additional information.) **NOTE**: A PRL = “00” is an illegal combination for 2x transfers. (“00” will be effectively promoted a “01”)

BITS	NAME	VALUE	FUNCTION
DMA_CMD[27:26]	Reserved		Reserved
DMA_CMD[28]	DMAC_PIPELINE	 0 1	DMAC_PIPELINE is **read only.** DMAC can accept another request. DMAC cannot accept another request.
DMA_CMD[29]	DMAC_STATUS	 0 1	DMAC_STATUS is **read only**. DMAC is not idle DMAC is idle.
DMA_CMD[30]	Reserved		Reserved
DMA_CMD[31]	EXP	 0 1	Expedite **NOTE**: Do not start a high priority DMA request if the DMAC is busy processing a low priority request. Low Priority High Priority

PCI Bus Master Registers

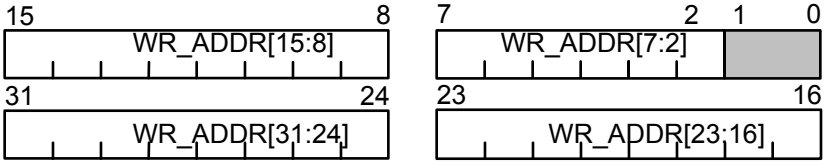
PCI Bus Master registers are located in both I/O and RBASE_I space. The T2R IV is capable of generating a specific PCI Bus Master Write to system memory of event status.

PCI Bus Master Write Address Register PCIB4 or RBASE_I + (0x00E0)

Name: PCI_BMWA

Type: Memory or I/O mapped read write

Address where Status Table is written.



BITS	NAME	VALUE	FUNCTION
PCI_BMWA[1:0]	Reserved		Reserved
PCI_BMWA[31:2]	WR_ADDR		Write Address (system memory physical address)

Status Table Format

This is the format of the status DWORD written into WR_ADDR.

BITS	STATUS NAME	REFERENCE	DESCRIPTION
[0]	DEB	FLOW[0]	Drawing Engine is busy
[1]	MCB	FLOW[1]	Memory Controller is busy
[2]	CLP	FLOW[2]	Clipping on previous command

[3]	PRV	FLOW[3]	Previous Command still executing
[4]	RPB	FLOW[4]	Rendering Pipeline and/or Pixel Cache Busy
[5]	DEPF	BUSY[0]	Drawing Engine Command Pipeline is full.
[6]	DLPB	DL_ADR[30]	DLP is busy
[7]	DMAC_PIPELINE	DMA_CMD[28]	DMAC pipeline is full.
[8]	DMAC_STATUS	DMA_CMD[29]	DMAC is idle
[9]	HBINT	GINTP[1]	HB Interrupt occurred
[10]	VBINT	GINTP[0]	VB Interrupt occurred
[11]	VBLANK	DB_ADR[29]	Vertical Blanking time
[30:12]	Reserved	N/A	RESERVED
[31]	REPORTED	N/A	PCI Bus Master always writes a 1. Software must clear this bit in order to detect a new write.

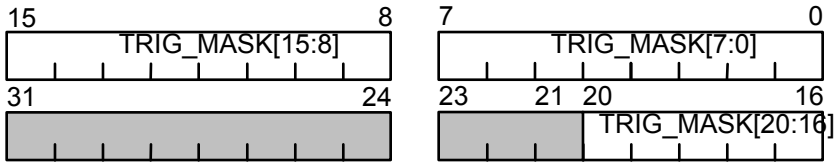
PCI Bus Master Trigger Mask Register PCIB4 or RBASE_I + (0x00E4)

Name: PCI_BMTM

Type: Memory or I/O mapped read write

Description: Borealis's PCI bus master can be triggered on separate rising or falling edge events. The appropriate mask bit must be cleared (reset to 0) in order for a bus master cycle to run for the given event. When an unmasked trigger event occurs, the bus master will write the **current** state of that status bit to WR_ADDR.

NOTE: A rising edge mask means that an event trigger will occur when the referenced status bit transitions from a 0 value to a 1 value. Falling edge indicates a 1 to 0 transition.



BITS	NAME	VALUE	FUNCTION
PCI_BMTM[20:0]	TRIG_MASK	0X1FFFFF	Trigger Mask for PCI Bus Master
PCI_BMTM[31:21]	Reserved		Reserved

BITS	NAME	VALUE	FUNCTION	STATUS TABLE BIT
[0]	DMAC_PIPELINE_FE_MASK	1	DMAC Pipeline falling edge mask	7
[1]	DMAC_PIPELINE_RE_MASK	1	DMAC Pipeline rising edge mask	7
[2]	DMAC_STATUS_FE_MASK	1	DMAC status falling edge mask	8
[3]	DMAC_STATUS_RE_MASK	1	DMAC status rising edge mask	8
[4]	DEB_FE_MASK	1	Drawing Engine busy falling edge mask	0
[5]	DEB_RE_MASK	1	Drawing Engine busy rising edge mask	0
[6]	HBINT_RE_MASK	1	HB Interrupt rising edge mask	9
[7]	VBINT_RE_MASK	1	VB Interrupt rising edge mask	10
[8]	VBLANK_FE_MASK	1	V Blank rising edge mask	11

BITS	NAME	VALUE	FUNCTION	STATUS TABLE BIT
[9]	VBLANK_RE_MASK	1	V Blank falling edge mask	11
[10]	MCB_FE_MASK	1	Memory Controller busy falling edge mask	1
[11]	MCB_RE_MASK	1	Memory Controller busy rising edge mask	1
[12]	CLP_RE_MASK	1	Clipping bit rising edge mask	2
[13]	PRV_FE_MASK	1	Previous Command bit falling edge mask	3
[14]	PRV_RE_MASK	1	Previous Command bit rising edge mask	3
[15]	RPB_FE_MASK	1	Rendering Pipeline/Pixel Cache busy bit falling edge mask	4
[16]	RPB_RE_MASK	1	Rendering Pipeline/Pixel Cache busy bit rising edge mask	4
[17]	DEPF_FE_MASK	1	Drawing Engine Pipeline Full falling edge mask	5
[18]	DEPF_RE_MASK	1	Drawing Engine Pipeline Full rising edge mask	5
[19]	DLPB_FE_MASK	1	DLP busy falling edge mask	6
[20]	DLPB_RE_MASK	1	DLP busy rising edge mask	6
[31:21]	Reserved		Reserved	N/A

VGA DAC Registers

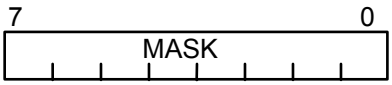
VGA DAC accesses (IO ports 0x03C6 - 0x03C9) are routed to the internal or external RAM DAC in two ways: snooping and owning. When snooping is enabled, VGA DAC writes will be routed to the RAM DAC, but ignored on the bus. Read cycles will be ignored. If snooping isn't enabled, then Borealis will own the VGA DAC cycles. Borealis will claim VGA DAC cycles on the bus. Both reads and writes will be routed to the RAM DAC. Snooping is controlled by bit 5 in PCI configuration register 1. VGA DAC cycles decode is controlled by the "vde" bit in the VGA_CTRL register, (*see Appendix F, Errata)*.

Pixel Mask Registers 0 x 03C6

Name: PEL_MASK

Type: I/O or Memory mapped read write

The contents of this register are logically ANDed, with the pixel input to the VGA color palette.

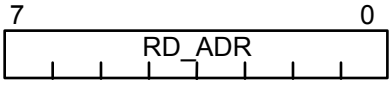


Read Address Register 0x03C7

Name: RD_ADR

Type: I/O or Memory mapped read write

This register defines the read address of the VGA color palette. This register is auto incrementing (with 3 successive reads of 0x3C9).

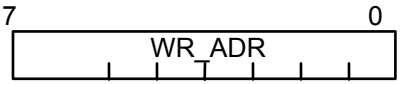


Write Address Register 0x03C8

Name: WR_ADR

Type: I/O or Memory mapped read write

This register defines the starting write address of the VGA color palette. This register is auto incrementing (with 3 successive writes of 0x3C9).

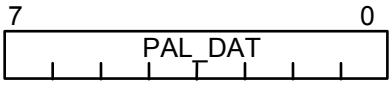


Palette Data Register 0x03C9

Name: PAL_DAT

Type: I/O or Memory mapped read write

This register contains data to be read from or written to the VGA color palette.



RAMDAC Registers

The following 16 registers provide access to a RAM DAC (external or internal). The first four registers maintain VGA LUT compatibility and are therefore also accessible from VGA I/O. When accessing these registers as T2R IV addresses, the VGA snooping enable bit in the PCI command register has no effect. The T2R IV implements a generic DAC 8-bit register interface through 16 DAC registers, which map to actual internal or external DAC registers through a register select (RS[3:0]) mechanism. External DACs are connected to the T2R IV peripheral bus (see Appendix B) and external DAC board designs will connect the peripheral bus address lines (PA[3:0]) to the DAC register selects (RS[3:0]). See the external DAC data sheet for register descriptions (indexed by RS[3:0]).
The T2R IV DAC registers may be accessed through the following I/O or memory addresses:

DAC Register	RS[3:0]	T2R IV addresses	VGA address
DAC00	**0000**	**PCIB4+80h, RBASEG+70h, RBASEG+00h**	**3C8h**
DAC01	**0001**	**PCIB4+84h, RBASEG+74h, RBASEG+04h**	**3C9h**
DAC02	**0010**	**PCIB4+88h, RBASEG+78h, RBASEG+08h**	**3C6h**
DAC03	**0011**	**PCIB4+8Ch, RBASEG+7Ch, RBASEG+0Ch**	**3C7h**
DAC04	**0100**	**PCIB4+90h, RBASEG+80h, RBASEG+10h**	
DAC05	**0101**	**PCIB4+94h, RBASEG+84h, RBASEG+14h**	
DAC06	**0110**	**PCIB4+98h, RBASEG+88h, RBASEG+18h**	
DAC07	**0111**	**PCIB4+9Ch, RBASEG+8Ch, RBASEG+1Ch**	
DAC08	**1000**	**PCIB4+A0h, RBASEG+90h**	
DAC09	**1001**	**PCIB4+A4h, RBASEG+94h**	
DAC10	**1010**	**PCIB4+A8h, RBASEG+98h**	
DAC11	**1011**	**PCIB4+ACh, RBASEG+9Ch**	

DAC12	**1100**	**PCIB4+B0h, RBASEG+A0h**
DAC13	**1101**	**PCIB4+B4h, RBASEG+A4h**
DAC14	**1110**	**PCIB4+B8h, RBASEG+A8h**
DAC15	**1111**	**PCIB4+BCh, RBASEG+ACh**

CRT Registers

The CRT registers specify the CRT timing sent to the DAC. All of the horizontal parameters are in terms of the VCLK, whereas the vertical parameters are in terms of horizontal lines. VCLK is defined as the number of pixels delivered to the DAC in one clock cycle.
For example:

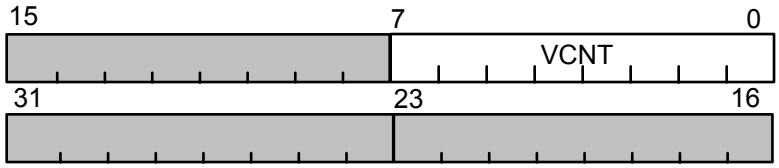
640 pixels at 16bpp with a 64 bit DAC bus width is 640/(64/16) = 160 VCLKs

Vertical Interrupt Count Register RBASE_G+(0x0020)

Name: INT_VCNT

Type: memory mapped read write.

This register defines the vertical field count. The contents of this register is compared with the field counter. When they match, a host interrupt is generated and the field counter is reset to zero. The range of this register is 0 to 255.



BITS	NAME	VALUE	FUNCTION
INT_VCNT[7:0]	VCNT	0x00 0x01 ... 0xfe 0xff	Interrupt on every vertical field. Interrupt on every other vertical field. ... Interrupt on every 255 vertical fields. Interrupt on every 256 vertical fields.

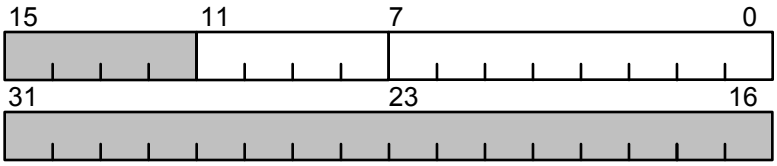
Horizontal Interrupt Count Register RBASE_G+(0x0024)

Name: INT_HCNT

Type: memory mapped read write.

This register defines a horizontal line before which an interrupt is generated.

A value 0x0ff in HCNT, for example, means that an interrupt will be generated at the beginning of the horizontal blank between line number 254 and 255.

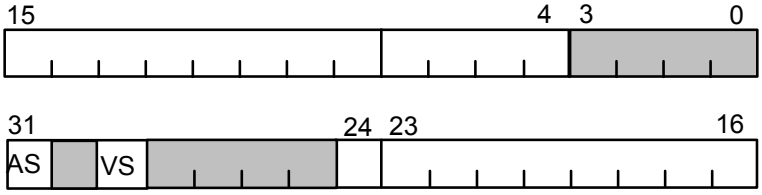


CRT Display Start Address (RBASE_G + 0x0028)

Name: DB_ADR

Type: Memory mapped read write

Display Start Address specifies the linear address of the first pixel to be displayed after vertical refresh (the upper left corner of screen). The start address is 16/128/256-byte aligned in SGRAM/SDRAM/WINRAM/WINRAM interleaved modes respectively.



BITS	NAME	VALUE	FUNCTION
DB_ADR[24:0]	Start Address		Display start address. Display start address can be changed at any time, however, changes have no effect until following vertical blank interval. At the time a value in DB_ADDR[24:0] gets actually accepted, bit AS gets cleared to zero. Memory Type Alignment SGRAM/SDRAM 16-byte aligned
DB_ADR[29]	VS	 0 1	Vertical Blank Status (read only) Vertical Blank is 0 (screen blanked) Vertical Blank is 1 (screen active)
DB_ADR[31]	AS	 0 1	Display Address Status (read only) Any write into DB_ADDR[24:0] sets this bit to one. At the time of accepting a new value in DB_ADDR[24:0], the AS bit gets cleared. A write into DB_ADDR[24:0] has been synchronized. A write into DB_ADDR[24:0] hasn't been synchronized yet

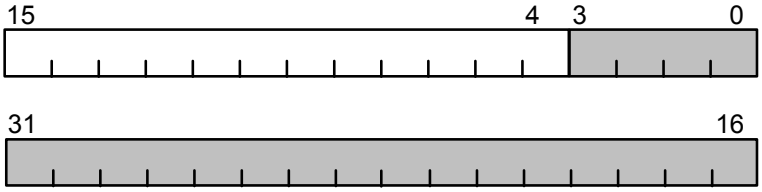
To avoid a "display tearing effect" when switching the display start address, a new address is actually passed to CRT controller only once per frame during vertical blank period.
DB_ADDR[24:0] can be written to at any time, however, to avoid skipping frames, the address should be updated only if AS bit reads zero. For proper operation, the most significant byte of the address (bit[24]), has to be always updated along with the other address bits.

Display Buffer Pitch (RBASE_G + 0x002C)

Name: DB_PTCH

Type: Memory mapped read write

This register defines the display pitch of the Display buffer. It is the address offset between two vertically adjacent pixels. The display pitch is 16 byte aligned.

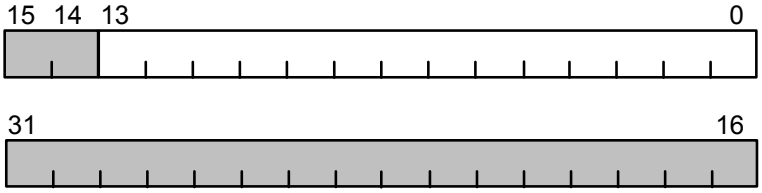


CRT Horizontal Active Line (RBASE_G + 0x0030)

Name: CRT_HAC

Type: Memory mapped read write

The horizontal active line register specifies the horizontal resolution in terms of CRT clock periods. The value in this register equals the number of CRT clocks (VCLK) during the active part of one line.

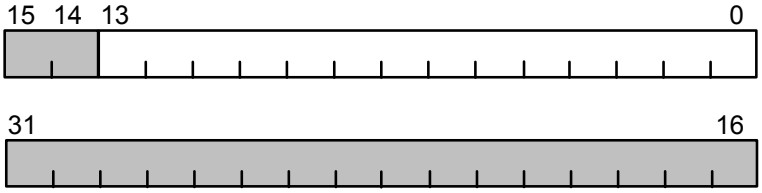


CRT Horizontal Blank Width (RBASE_G + 0x0034)

Name: CRT_HBL

Type: Memory mapped read write

This register specifies width of horizontal blank in number of CRT clocks (VCLK).

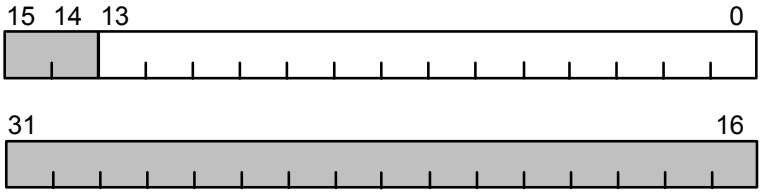


CRT Horizontal Front Porch Width (RBASE_G + 0x0038)

Name: CRT_HFP

Type: Memory mapped read write

This register specifies the width of horizontal front porch in number of CRT clocks (VCLK). The value of CRT_HFP is equal to the number of CRT clock periods between the beginning of horizontal blank and the following horizontal sync impulse. The smallest value is 0.

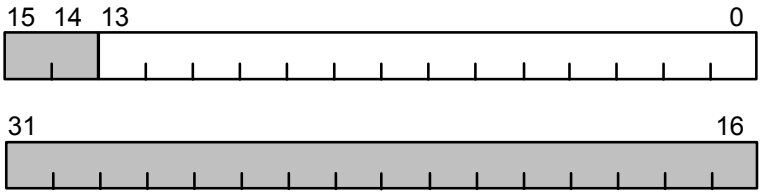


CRT Horizontal Sync Width (RBASE_G + 0x003C)

Name: CRT_HS

Type: Memory mapped read write

This register specifies the width of the horizontal sync impulse in number of CRT clock periods (VCLK) The smallest value for non-interlaced displays is 1.

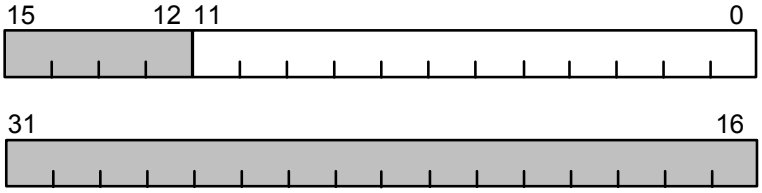


CRT Vertical Field Active (RBASE_G + 0x0040)

Name: CRT_VAC

Type: Memory mapped read write

Vertical Field Active register specifies the vertical resolution in number of lines. It is the number of displayed lines during one frame.

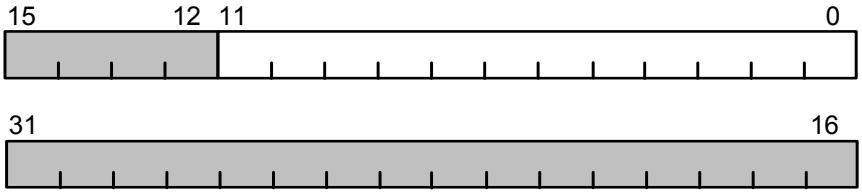


CRT Vertical Blank Width (RBASE_G + 0x0044)

Name: CRT_VBL

Type: Memory mapped read write

This register specifies the number of lines blanked during one frame.

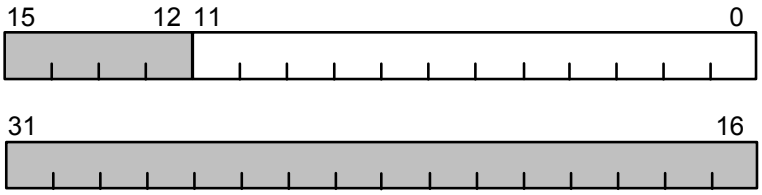


CRT Vertical Front Porch Width (RBASE_G + 0x0048)

Name: CRT_VFP

Type: Memory mapped read write

This register specifies the width of the vertical front porch. The units of this register are the number of lines .

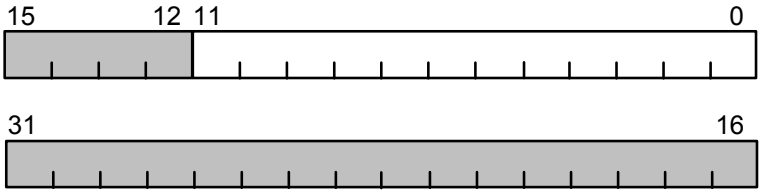


CRT Vertical Sync Width (RBASE_G + 0x004C)

Name: CRT_VS

Type: Memory mapped read write

This register specifies the width of the vertical sync impulse The units of this register are the number of lines .



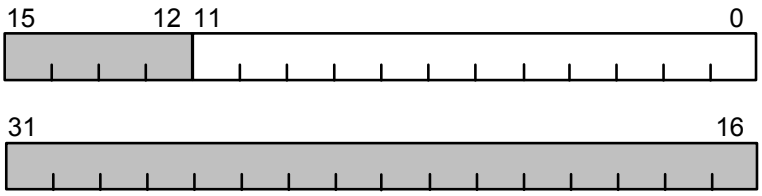
CRT Line Counter (RBASE_G + 0x0050)

Name: CRT_LCNT

Type: Memory mapped read only

This read only register shows the actual value of the display line counter.

A zero corresponds to beginning of a vertical blank period. A value greater than the number of lines in a vertical blank (see CRT_VBL register) indicates active portion of video, etc.

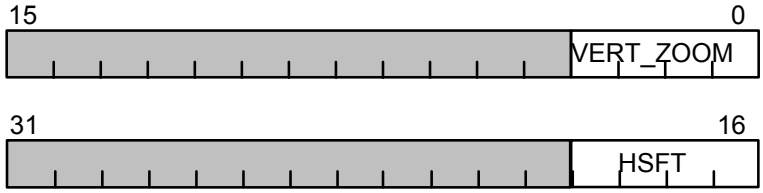

15
12
11
0
31
16

CRT Display Buffer Zoom Factor (RBASE_G + 0x0054)

Name: CRT_ZOOM

Type: Memory mapped read write

This register defines the vertical zoom and horizontal shift factors for the display output stage.

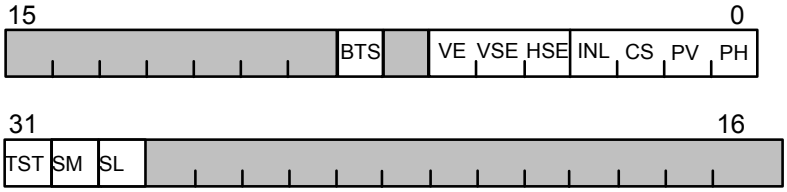


BITS	NAME	VALUE	FUNCTION
CRT_ZOOM[3:0]	VERT_ZOOM[3:0] **NOTE**: Vertical Zoom has no effect in line sequential stereo mode. See CRT_1CON[30].	0x0 0x1 0x2 0x3 0x4 ..... 0xd 0xe 0xf	No Zoom Zoom 2X Zoom 3X Zoom 4X Zoom 5X ..... Zoom 14X Zoom 15X Zoom 16X
CRT_ZOOM [19:16]	HSFT[3:0] **NOTE**: HSFT[3:0] bits cause simple right shifting of all horizontal sync parameters. Using this feature for zoom purpose may require (in some applications) dividing VCLK by the same factor.	0x0 0x1 0x3 0x7 0xF	No Divide Divide By 2 Divide by 4 Divide by 8 Divide by 16

CRT Configuration Register 1 (RBASE_G + 0x0058)

Name: CRT_1CON

Type: Memory mapped read write



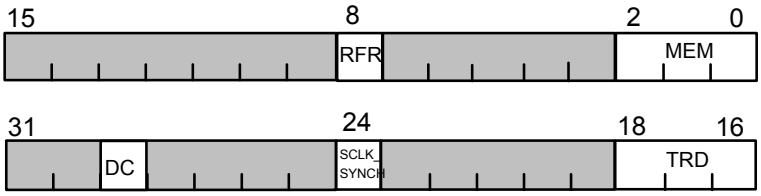
BITS	NAME	VALUE	FUNCTION
CRT_1CON[0]	PH	0 1	Horizontal Sync Polarity. Negative horizontal sync. Positive horizontal sync. **NOTE**: Typically this bit is set to negative sync. If positive sync is desired, invert the sync polarity in the DAC.
CRT_1CON[1]	PV	0 1	Vertical Sync Polarity. Negative vertical sync. Positive vertical sync. **NOTE**: Typically this bit is set to negative sync. If positive sync is desired, invert the sync polarity in the DAC.
CRT_1CON[2]	CS	0 1	Composite Sync Enable. The polarity of composite sync is **always negative**. If composite sync is enabled, then Vertical Sync becomes a one clock wide impulse (always negative). Separate HSYNC and VSYNC are generated Composite sync is output on HSYNC
CRT_1CON[3]	INL	0	This bit must be set to 0.
CRT_1CON[4]	HSE	0 1	Horizontal Sync Enable HSYNC disabled (HSYNC = 1) Normal HSYNC generation
CRT_1CON[5]	VSE	0 1	Vertical Sync Enable VSYNC disabled (VSYNC = 1) Normal VSYNC generation
CRT_1CON[6]	VE	0 1	Video Enable BLANK disabled (BLANK= 0) Normal BLANK generation
CRT_1CON[8] **NOTE**: In SGRAM mode, this bit enables digital RBG outputs only if PCI ROM BASE Address Register bit [0]=0.	BTS	0 1	SCLK Pin direction/Flat Panel Display Selector **SGRAM/SDRAM mode: Digital RBG Enable** Digital RGB outputs off. Digital RGB outputs enabled.
CRT_1CON[29]	SL	0 1	EMGNT# pin used as left/right line indicator. This setting should be used only with SM bit=1.

BITS	NAME	VALUE	FUNCTION
CRT_1CON[30]	SM	0 1	Normal CRT operation Enables line sequential stereo mode.
CRT_1CON[31]	TST	0 1	CRT Test Mode Bit Normal Operation. Test Mode.

CRT Configuration Register 2 (RBASE_G + 0x005C)

Name: CRT_2CON

Type: Memory mapped read write



BITS	NAME	VALUE	FUNCTION
CRT_2CON[2:0]	MEM[2:0]		Reserved
CRT_2CON[8]	RFR	 0 1	Enables screen to CRT FIFO in SGRAM/SDRAM mode) - refresh disabled - refresh enabled
CRT_2CON[18:16]	TRD	 0x0 0x1 0x2 0x3 0x4 0x5 0x6 0x7	"Memory to register" transfer delay no delay 1 VCLK delay 2 VCLKs delay 3 VCLKs delay 4 VCLKs delay 5 VCLKs delay 6 VCLKs delay 7 VCLKs delay
CRT_2CON[24]	SCLK_SYNCH	0 1	This bit should be 0. (SCLK synched with CRTCLK. This bit should be always be 0, but if set to 1, SCLK not synched with CRTCLK. Should only be used with DACs that have asynchronous clocks (LCLK - CRTCLK) like Brooktree and IBM (not TI).
CRT_2CON[29]	DC	 0x1	Display Configuration SGRAM/SDRAM

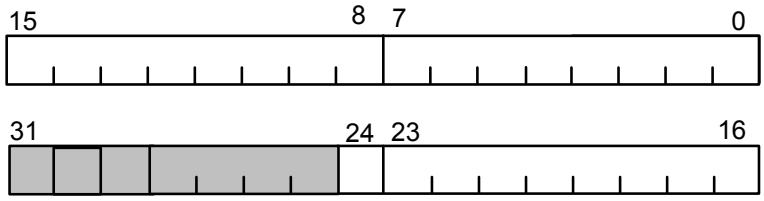
NOTE: Do not switch DC values if RFR=1.

CRT Display Start Address 2 (RBASE_G + 0x0060)

Name: DB_ADR2

Type: Memory mapped read write

Display Start Address2 specifies the linear address of the secondary display buffer used only in stereo mode. Value in this register is valid only if SM bit in CRT_1CON register is set to one. The start address is 16 byte aligned in SGRAM/SDRAM mode.



BITS	NAME	VALUE	FUNCTION
DB_ADR2[24:0]	Start Address2		Display start address.

If stereo mode is set, the first line after vertical blank and then every other line will be displayed from the primary display buffer defined by DB_ADR register. Second line after vertical blank and then every other line will be displayed from the secondary display buffer defined by DB_ADR2. For example, the CRTC will output line 0 from DB_ADR, line 0 from DB_ADR2, line 1 from DB_ADR, line 1 from DB_ADR2, etc.

Memory Windows™ Configuration Registers

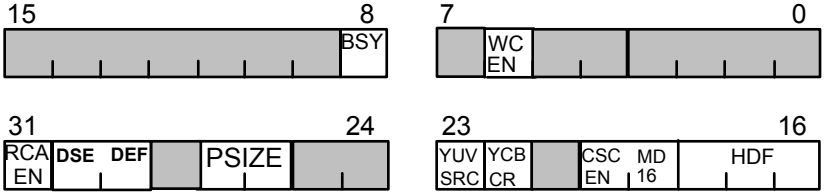
The Memory Windows™ configuration registers set the size, control, system memory location and local buffer offset of the Memory Windows™. There are two identical sets of these memory mapped registers. The address of these registers is set by the value written to the RBASE_W register.

Memory Window Control Register

Name: MW0_CTRL, MW1_CTRL

Type: Memory mapped read write

window zero RBASE_W + (0x0000)
window one RBASE_W + (0x0028) , PCIB4 + (0x0040)
The register controls one of two memory windows that Borealis supports.



BITS	NAME	VALUE	FUNCTION
MWX_CTRL[5:0]	Reserved	0x0	Reserved
MWX_CTRL[6]	WCEN	0x0 0x1	Enable write caching Disable write caching
MWX_CTRL[7]	Reserved	0x0	Reserved
MWX_CTRL[8]	BSY	 0x0 0x1	Memory Window Busy **(read only)** Memory Window is busy Memory Window is not busy

BITS	NAME	VALUE	FUNCTION
MWX_CTRL[16]	HDF[0]	0x0 0x1	Bit ordering is not modified Swaps bits within each byte
MWX_CTRL[17]	HDF[1]	0x0 0x1	Byte ordering is not modified Swap bytes within each word
MWX_CTRL[18]	HDF[2]	0x0 0x1	Word ordering is not modified Swap words within each DWORD
MWX_CTRL[19]	MD16	 0x0 0x1	16 Bits per pixel mode 0=5RED,5GREEN,5BLUE 1=5RED,6GREEN,5BLUE
MWX_CTRL[20]	CSC_EN	 0x0 0x1	Color space conversion enable Disable color space conversion Enable color space conversion
MWX_CTRL[21]	Reserved	0x0	Reserved
MWX_CTRL[22]	YCBCR	 0X0 0X1	YCBCR Selector (color space format) YCBCR YUV
MWX_CTRL[23]	YUV_SRC	 0x0 0x1	YUV source format. YUV 422. (Default) YUV 444.
MWX_CTRL[25:24]	Reserved	 0x0	Reserved.
MWX_CTRL[27:26]	PSIZE[1:0]	 0x0 0x1 0x2 0x3	Destination pixel size, number of bits per pixel Eight bits per pixel Sixteen bits per pixel Thirty-two bits per pixel Reserved
MWX_CTRL[28]	Reserved	0x0	Reserved
MWX_CTRL[29]	DEF	 0 1	Drawing Engine Flush Cache on 2D/3D Trigger Access Enable (default). Disable.
MWX_CTRL[30]	DSE	 0 1	CRT Flush on Vertical Blank Interrupt Enable(default). Disable.
MWX_CTRL[31]	RCA_EN	 0 1	Read cache enable. Enable read caching. (Default) Disable read caching.

window zero **RBASE_W + (0x0000)**
window one **RBASE_W + (0x0028) , PCIB4 + (0x0040)**

Memory Windows Address Registers

window zero **RBASE_W + (0x0004)**
window one **RBASE_W + (0x002C) , PCIB4 + (0x0044)**

Name: MW0_AD, MW1_AD

Type: Memory mapped read write

The address contained in this register specifies the start of system memory space that is mapped to Borealis local memory. The granularity of this register is determined by the memory window size that is specified by the SIZE field which is described in Chapter 4. In the case of the smallest memory window which is 4 Kb, MWX_AD[31:12] are decoded for the memory window. For an 8 KB memory window,

MWX_AD[31:13] are decoded. This pattern continues to the maximum window size of 32 MB, where MWX_AD[31:25] are decoded.
The system BIOS is responsible for allocation of all PCI system resources. The allocation process is facilitated by the PCI base registers described in Chapter 3. Borealis will request 4, 8, 16, or 32 megabytes of memory space for each of the memory windows based on the configuration pin settings. The system will then allocate memory to Borealis by writing the starting address of the memory block for memory window 0 to base address register 0 and the starting address of the memory block for memory window 1 to base address register 1. These values will be shadowed to MW0_AD and MW1_AD respectively. Software can then program each of the memory windows to reside anywhere within the allocated memory block. The size of each window can then be set from a minimum of 4 kilobytes to a maximum determined by the amount of memory allocated by the PCI system.

BITS	NAME	VALUE	FUNCTION
MWX_AD[31:12]	ADDRESS		Address decode value

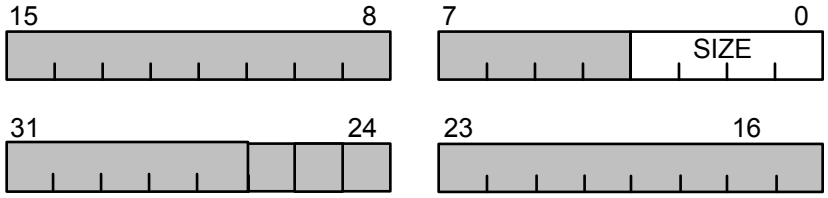
Memory Window Size

window zero RBASE_W + (0x0008)
window one RBASE_W + (0x0030) , PCIB4 + (0x0048)

Name:MW0_SZ, MW1_SZ

Type: Memory mapped read write

This register defines the size of a Memory window. The size of a memory window should be set equal to or less than the amount of memory allocated by the PCI system.



BITS	NAME	VALUE	FUNCTION
MWX_SZ_DIB[3:0]	SIZE	0x0 0x1 0x2 ... 0xD 0xE 0xF	Size equals 4K Size equals 8K Size equals 16K ... Size equals 32MB RESERVED RESERVED

Memory Window Origin

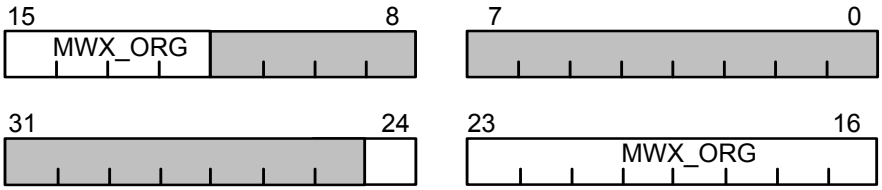
window zero RBASE_W + (0x0010 or 0x0014)
window one RBASE_W + (0x0038 or 0x003C) , PCIB4 + (0x0050 or 0x0054)

Name: MW0_ORG,MW1_ORG

Type: Memory mapped read write

This register specifies the starting address of a memory window within a selected buffer or buffers. The number of valid bits in this register depends on the size of the memory window. As the size of the memory

window increases, fewer bits will be taken from this register and more will be taken from the host address. In the case of a 4 kilobyte memory window, the address presented to the memory controller will be a concatenation of bits[24:12] from this register with the lower 12 bits coming from the host. An 8 kilobyte memory window will cause bits[24:13] to be taken from this register with the lower 13 bits coming from the host. A 32 megabyte memory window would not use any bits from this register.



BITS	NAME	VALUE	FUNCTION
MWX_ORG[25:12]	MWX_ORG[24:12] MWX_ORG[24:13] MWX_ORG[24:14] • • • MWX_ORG[24:23] MWX_ORG[24]		Upper order memory window address bits Upper order address for 4 kilobyte window Upper order address for 8 kilobyte window Upper order address for 16 kilobyte window • • • Upper order address for 8 megabyte window Upper order address for 16 megabyte window

Memory Window Plane Mask

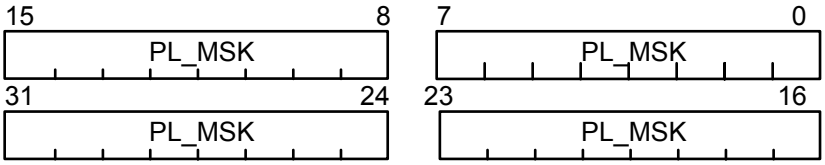
<<Need an introduction here.>>
window zero RBASE_W + (0x0024)
window one RBASE_W + (0x004C) , PCIB4 + (0x0064)

Name: MW0_MASK, MW1_MASK

Type: Memory mapped read write

The MASK register allows for selective writing of pixel planes to the Display Buffer. Using this feature, bit maps may be partitioned into logical groups for selective updating. A bit value of zero disables writing to a specific plane while a one enables writing to that plane. In eight bit per pixel mode, the plane mask in MASK[7:0] should be replicated four times into the entire register. In sixteen bits per pixel mode, MASK[15:0] should be replicated two times into the entire register.

NOTE: The MASK registers will have no effect if the memory clips used with Borealis does not support the write-per-bit feature.



BITS	NAME	VALUE	FUNCTION
MWX_MASK [31:0]	PL_MSK		Plane Masking Register

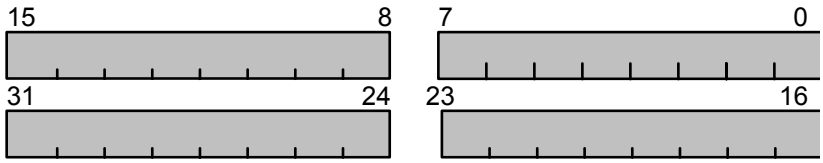
Memory Window Flush Trigger

<<Need an introduction here.>>
RBASE_W + (0x0054)

Name: MWC_FLSH

Type: Memory mapped write, always reads back as zero

This register when written triggers a flush of the memory windows cache. The value of the data written to this register is a don't care.



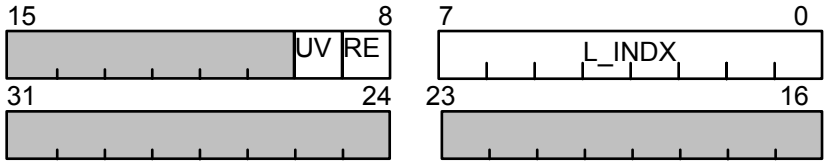
YUV LUT Index Register

RBASE_W + (0x0058)

Name: YUV_ADR

Type: Memory mapped read, write

This is a test register is used to indirectly address the YUV LUT.



BITS	NAME	VALUE	FUNCTION
YUV_ADR[7:0]	L_INDX	0x00 to 0xff	Address pointer into the color space conversion LUT.
YUV_ADR[8]	RE	 0x0 0x1	LUT read enable bit. 0 = Normal operation. (Default) 1 = Read back mode.
YUV_ADR[9]	UV	 0x0 0x1	LUT select. 0 = Select V LUT. (Default) 1 = Select U LUT.

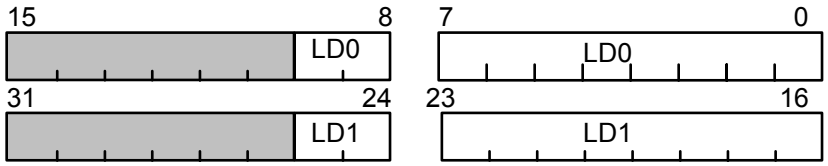
YUV LUT Data Register

RBASE_W + (0x005C)

Name: YUV_DAT

Type: Memory mapped read only

This is a test register. It returns the value of the LUT location pointed to by YUV_ADR.



BITS	NAME	VALUE	FUNCTION
YUV_DAT[9:0]	LD0		LUT read data L_INDX.
YUV_ADR[25:16]	LD1		LUT read data L_INDX + 1.

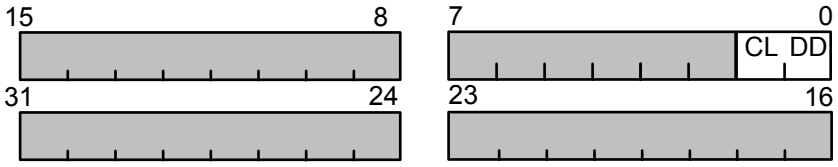
Drawing Engine Command and Parameter Registers

The following group of registers are memory mapped and are used to issue commands and parameters to the drawing engine. The address of these registers is offset by RBASE_D.

Interrupt Register RBASE_D + (0x0000)

Name: INTP

Type: Memory mapped read write

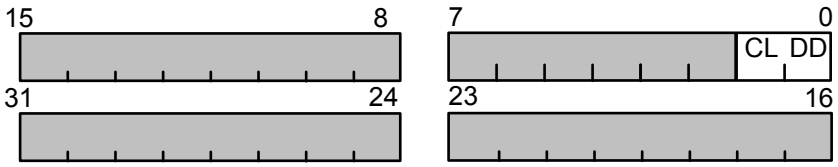


BITS	NAME	VALUE	FUNCTION
INTP[0]	DD_INT	0 1	Drawing operation not completed Drawing operation is completed
INTP[1]	CL_INT	0 1	Clip interrupt has not occurred Clip interrupt has occurred
INTP[31:2]	Reserved	0x0	Reserved

Interrupt Mask Register RBASE_D + (0x004)

Name: INTM

Type: memory mapped read write

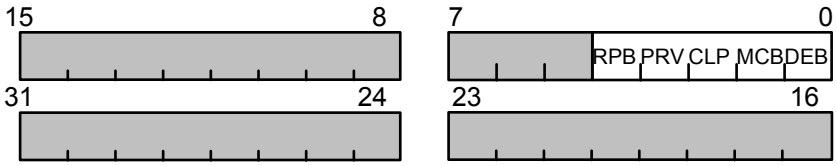


BITS	NAME	VALUE	FUNCTION
INTM[0]	DD_MSK	0 1	Drawing completed interrupt disabled Drawing completed interrupt enabled
INTM[1]	CL_MSK	0 1	Clip interrupt disabled Clip interrupt enabled
INTM[31:2]	Reserved	0x0	Reserved

Flow Control Register RBASE_D +(0x0008)

Name: FLOW

Type: read only



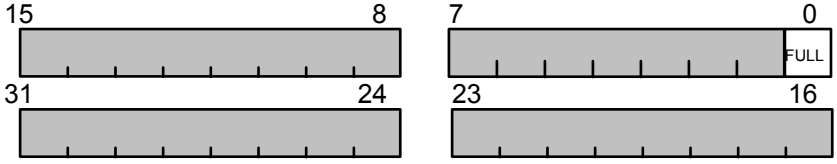
BITS	NAME	VALUE	FUNCTION
FLOW[0]	DEB	0 1	Drawing engine is not busy Drawing engine is busy
FLOW[1]	MCB	0 1	Memory controller is not busy Memory controller is busy
FLOW[2]	CLP	0 1	No clipping was applied to previous drawing command Clipping was applied to previous drawing command
FLOW[3]	PRV	 0 1	Previous Command Busy The previously triggered command has completed its execution and the memory controller for the same command has subsequently gone not busy at least once. The previously triggered command is still executing or the busy bit is asserted.
FLOW[4]	RPB	0 1	Renderer Pipeline and Pixel Cache both not busy. Renderer Pipeline and/or Pixel Cache busy. **NOTE**: The Drawing engine will complete before the renderer and pixel cache complete. It is possible that the memory controller may be read as not busy before the pipeline and cache are flushed.

NOTE: Drawing Engine Busy (DEB) and Renderer Pipeline Busy (RPB) are independent busy status bits.

BUSY Register RBASE_D +(0x000C)

Name: BUSY

Type: memory mapped read only



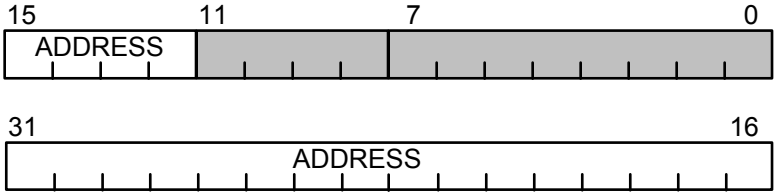
BITS	NAME	VALUE	FUNCTION
BUSY[0]	FULL	0	Borealis is ready to accept a new command. *Command pipeline not full.*
		1	Borealis is not ready to accept a new command. *Command pipeline is full.*

Drawing Engine Window Address RBASE _D + (0x0010)

Name: DEW_AD

Type: Memory mapped read write

This register defines the system address range that the host interface will decode for the Drawing Engine memory window. Before programming this register, the EDWA bit in CONFIG1 should be set to 0.

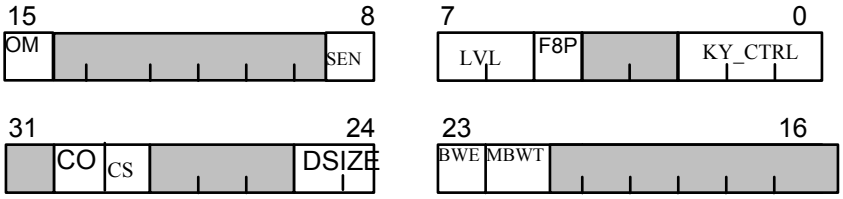


BITS	NAME	VALUE	FUNCTION
DEW_AD[31:12]	ADDRESS		Address decode value

Buffer Control Register RBASE _D + (0x0020)

Name: BUF_CTRL

Type: Memory mapped read write



This register contains the buffer control bits for the drawing engine.
The source enable bits (SEN) select whether Borealis's internal 2D Drawing Engine Cache is selected, or if the on board memory (local buffer) is selected.
Cache On (CO) is meaningful only when the source data is to be read from Borealis's internal 2D data cache. Setting CO to 1 indicates that the 2D cache always will contain valid data. This is useful when software partitions the cache into two halves, guaranteeing that one half of the cache contains valid data. Resetting CO to O is done by software to indicate it is no longer using the 2D cache.
The Cache Select (CS) bit is used to select host bus writing access to either the 2D Drawing Engine Cache or the user programmable Fog Table.
The Destination Pixel Size (DSIZE) bits define the destination pixel depth and format for the current drawing operation. The source pixel size and format is 32bpp, 8888 format and all drawing operations will be performed with this format. Right before alpha blending, z compare and ROP's, the pixel will be re-formatted to the specified destination pixel size and format. Texture cache formats are defined separately in the TEX_CTRL register, but will be expanded to 32bpp, 8888 format before any operation is performed on the texels. This preserves precision during pixel and texel operations.
BWE and MBWT are the memory block write mode bits. BWE is the block write enable bit and the MBWT indicates which memory block write type you are enabling.
OM is the origin mode bit which defines how the origin is defined and how addresses are calculated. See SORG and/or DORG for origin and address calculation.

The Key Control field is used in conjunction with the DE_KEY register to set up a 2D transparency operation. 3D keying is selected in the 3D_CTRL register and is independent of 2D color keying. 2D keying is not reliable if doing a 3D operation involving texture-mapping, specular, fog, or texture filtering. 3D keying should be used only under these circumstances.

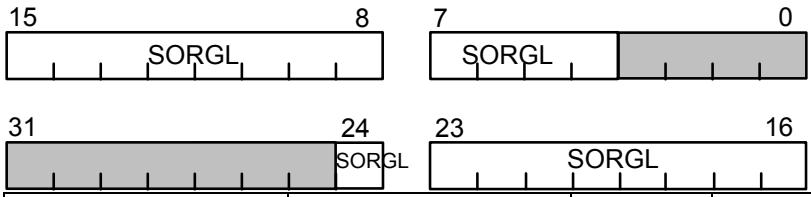
BITS	NAME	VALUE	FUNCTION
BUF_CTRL[2:0]	KY_CTRL		Key Control
		0xx	No Keying
		100	Source Keying, mask on key color
		101	Destination Keying, mask on key color
		110	Source Keying, mask on NOT key color
		111	Destination Keying, mask on NOT key color
BUF_CTRL[4:3]	Reserved	0	Reserved
BUF_CTRL[5]	F8P		Force 8 Page Mode.
		0	Drawing Engine runs 8 & 16 page accesses
		1	Drawing Engine Runs 8 page access only
BUF_CTRL[7:6]	LVL		Line Vertical Limit This bit sets the limit to the number of vertical requests which the pixel cache can make:
		0x0	0 = no limit (4 pages)
		0x1	1 = 1 page
		0x2	2 = 2 pages
		0x3	3 = 3 pages.
BUF_CTRL[8]	SEN		Source Read Enable.
		0x0	Enable reads from local memory.
		0x1	Enable reads from Drawing Engine 2D Cache.
BUF_CTRL[14:9]	Reserved	0x0	Reserved
BUF_CTRL[15]	OM		Origin Mode
		0x0	Normal linear origin mode
		0x1	XY origin mode
BUF_CTRL[21:16]	Reserved	0x0	Reserved
BUF_CTRL[22]	MBWT		Memory Block Write Type
		0x1	SGRAM/SDRAM Block write
BUF_CTRL[23]	BWE		Block Write Enable
		0x0	Block writes disabled
		0x1	Block writes enabled
BUF_CTRL[25:24]	DSIZE[1:0]		Destination Data Pixel Size and Format - Number of Bits per Pixel.
		0x0	8 bits per pixel - 332 Format (Blue Channel)
		0x1	16 bits per pixel - 1555 Format
		0x2	32 bits per pixel - 8888 Format
		0x3	16 bits per pixel - 565 Format
BUF_CTRL[28:26]	Reserved	0x0	Reserved
BUF_CTRL[29]	CS		Cache Select (PCI/AGP Cache Access Only)
		0	Drawing Engine 2D Cache
		1	User Programmable Fog Table
BUF_CTRL[30]	CO		Drawing Engine Source Enable **(Must to the same as SEN)**
		0	When SEN=0, turn this bit on 0 to enable read access to local buffer..
		1	When SEN=1, turn this bit to 1 enable read access to Drawing Engine 2D Cache.
BUF_CTRL[31]	Reserved	0x0	Reserved

Drawing Engine Source Origin RBASE_D + (0x0028)

Name: DE_SORG

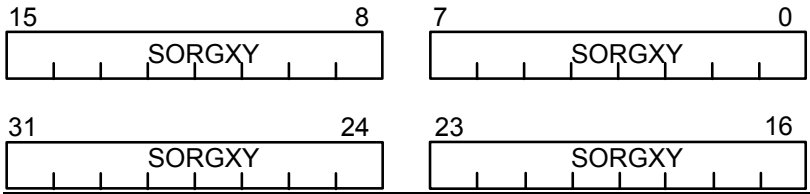
Type: Memory mapped read write

This register defines the linear or XY address offset into the local buffer for drawing engine read cycles. In linear mode this is a 16 byte aligned address when accessing the local buffer. In XY origin mode this is a pixel address offset.



BITS	NAME	VALUE	FUNCTION
DE_SORG[24:4]	SORGL		Linear origin mode.

Origin = DE_SORG.
Address = Y*SPTCH + X*(Bytes per Pixel) + DE_SORG.



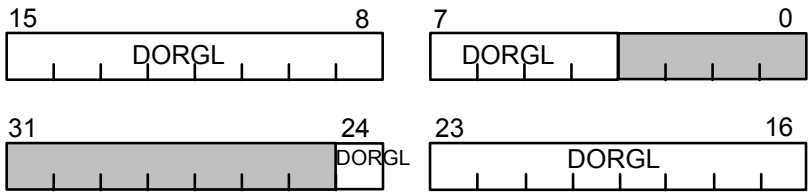
BITS	NAME	VALUE	FUNCTION
DE_SORG[31:0]	SORGXY		XY origin mode

Origin = Y_ORG*SPTCH + X_ORG*(Bytes per Pixel)
Address = (Y+Y_ORG)*SPTCH + (X+X_ORG)*(Bytes per Pixel)

Name: DE_DORG

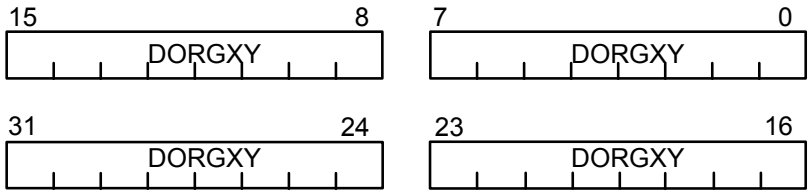
Type: Memory mapped read write

This register defines the linear or XY address offset into the selected buffer or buffers for drawing engine write cycles. In linear mode this is a 16 byte aligned address when accessing the local buffer. In XY origin mode this is a pixel address offset.



BITS	NAME	VALUE	FUNCTION
DE_DORG[24:4]	DORGL		Linear origin mode.

Origin = DE_DORG.
Address = Y*SPTCH + X*(Bytes per Pixel) + DE_DORG.



BITS	NAME	VALUE	FUNCTION
DE_DORG[31:0]	DORGXY		XY origin mode.

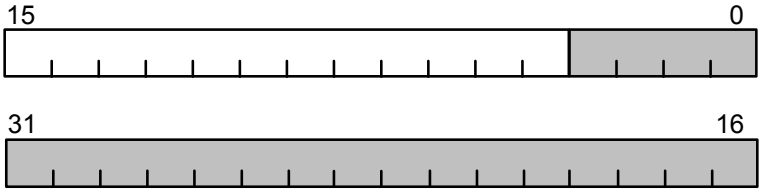
Origin = Y_ORG*SPTCH + X_ORG*(Bytes per Pixel)
Address = (Y+Y_ORG)*SPTCH + (X+X_ORG)*(Bytes per Pixel)

DE Texture Pitch RBASE_D + (0x0038)

Name: DE_TPTCH

Type: Memory mapped read write

This register defines the pitch of the texture map in LOD0 in bytes. All subsequent texture maps have a pitch equal to the previous texture/2, with a minimum size of 1. It is aligned to a 16 byte boundary.

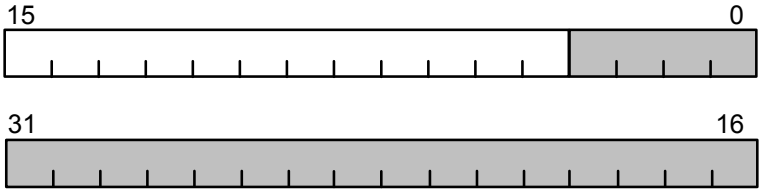


DE Z Buffer Pitch RBASE_D + (0x003C)

Name: DE_ZPTCH

Type: Memory mapped read write

This register defines the pitch of the Z buffer in bytes. It is aligned to a 16 byte boundary.

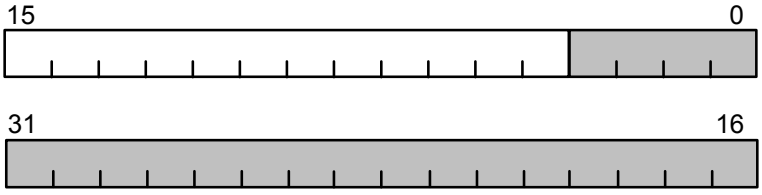


DE Source Pitch RBASE_D + (0x0040)

Name: DE_SPTCH

Type: Memory mapped read write

This register defines the pitch of the source bit map in bytes. It is aligned to a 16 byte boundary.

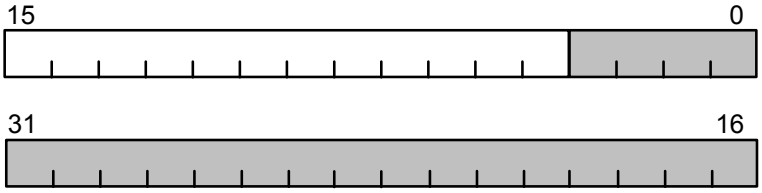


DE Destination Pitch RBASE_D + (0x0044)

Name: DE_DPTCH

Type: Memory mapped read write

This register defines the pitch of the destination bit map in bytes. It is aligned to a 16 byte boundary.

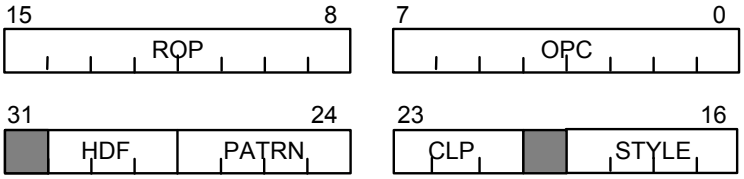


Command Register RBASE_D +(0x0048) (Shadowed with 0x0168)

Name: CMD

Type: Memory mapped read write

The drawing engine command register can be accessed as a single 32 bit register or individual fields can be accessed in their own register space as described in the following pages.

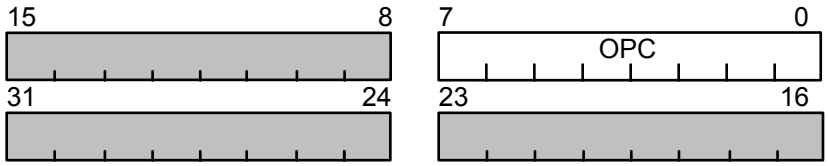


BITS	NAME	FUNCTION
CMD[7:0]	OPC	Drawing command opcode
CMD[15:8]	ROP	Drawing Raster or logical operation
CMD[19:16]	STYLE	Solid, Transparency, Stipple control
CMD[20]	Reserved	Reserved
CMD[23:21]	CLP	Clipping control
CMD[27:24]	PATRN	No last, pat reset, area pattern, and pattern cache enable.
CMD[30:28]	HDF	Host Data Format
CMD[31]	Reserved	Reserved

Command Opcode RBASE_D +(0x0050)

Name: CMD_OPC

Type: Memory mapped read write



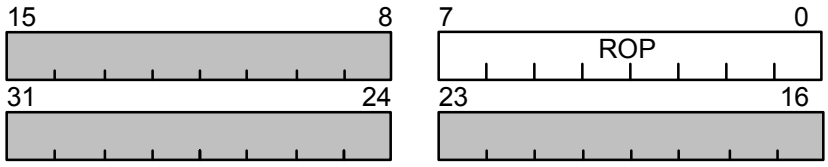
VALUE	NAME	FUNCTION
0x00	NOOP	Transfer parameters no drawing
0x01	BITBLT	BIT BLT command
0x02	LINE	2D Line command
0x03	ELINE	2D Error Line
0x04	Reserved	Reserved
0x05	PLINE	2D Poly Line command.
0x06	Reserved	Reserved
0x07	Reserved	Reserved
0x08	LINE_3D	3D Line command
0x09	TRIAN_3D	3D Triangle Command
0x0A	TEX_INV	Invalidate Texture Cache
0x0B	LD_PAL	Palette Load Command
0x0C---0xFF	Reserved	For future expansion (no action taken)

Command Raster Operation RBASE_D +(0x0054)

Name: CMD_ROP

Type: Memory mapped read write

The ROP field sets the logical operation for all drawing commands.

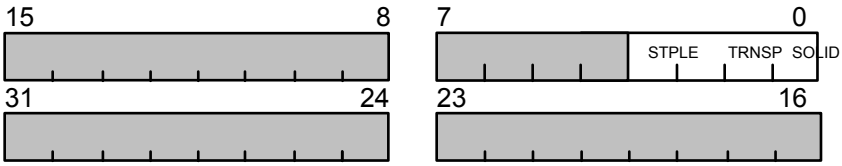


VALUE	NAME	FUNCTION
0x00	clear	0 (zero)
0x01	nor	(NOT source) AND (NOT destination)
0x02	andInverted	(NOT source) AND destination
0x03	copyInverted	(NOT source)
0x04	andReverse	source AND (NOT destination)
0x05	invert	(NOT destination)
0x06	xor	source XOR destination
0x07	nand	(NOT source) OR (NOT Destination)
0x08	and	source AND destination
0x09	equiv	(NOT source) XOR destination
0x0A	noop	Destination
0x0B	orInverted	(NOT source) OR destination
0x0C	copy	source
0x0D	orReverse	source OR (NOT destination)
0x0E	or	source OR destination
0x0F	set	1 (one)
0x10 – 0xFF	RESERVED	reserved for future expansion

Line/Fill Style Register RBASE_D +(0x0058)

Name: CMD_STYLE

Type: Memory mapped read write

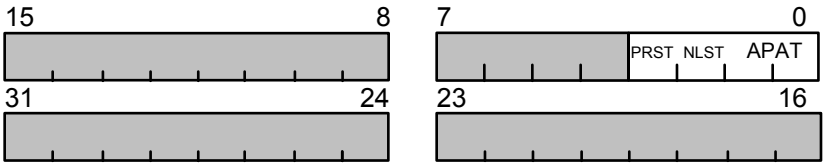


BITS	NAME	VALUE	FUNCTION
CMD_STYLE[0]	SOLID		Solid When SOLID is set to one the source of the data for all operations is the FORE register.
CMD_STYLE[1]	TRNSP		Transparency When TRNSP is set to one during a line , stippled bit blt, or stippled triangles a background condition will cause the destination data to be unmodified. Transparency overrides solid.
CMD_STYLE[3:2]	STPLE	 0x0 0x1 0x2 0x3	Stipple No stipple Reserved Packed stipple (padded to 32 bit boundary) Packed stipple (padded to 8 bit boundary)
CMD_STYLE[4]	Reserved	0x0	Reserved.

Patterning Register RBASE_D +(0x005C)

Name: CMD_PATRN

Type: Memory mapped read write

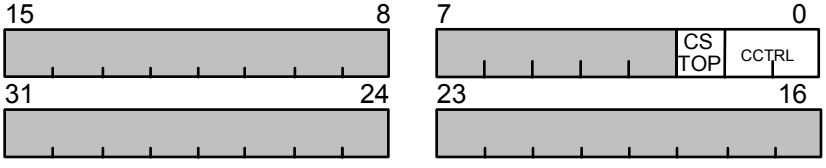


BITS	NAME	FUNCTION
CMD_PATRN[1:0]	APAT[1:0]	Area Pattern When APAT is set to two it forces the source area pattern to be a 32x32 screen locked pattern.
	0x0 0x1 0x2 0x3	Area pattern mode is off. Area pattern mode equals 8x8. Area pattern mode equals 32x32. RESERVED
CMD_PATRN[2]	NLST	No last pixel When NLST is set to one during a line operation it forces the last pixel in the line not to be drawn.
	0 1	Draw the last pixel in a line Do not draw the last pixel in a line
CMD_PATRN[3]	PRST	Pattern reset bit When PRST is set to a one during a line operation, the pattern pointers will be reset before each line is started.
	0 1	Don't reset pattern pointers at the start of each line Reset pattern pointers at the start of each line

Clipping Control Register RBASE_D +(0x0060)

Name: CMD_CLP

Type: Memory mapped read write



BITS	NAME	VALUE	FUNCTION

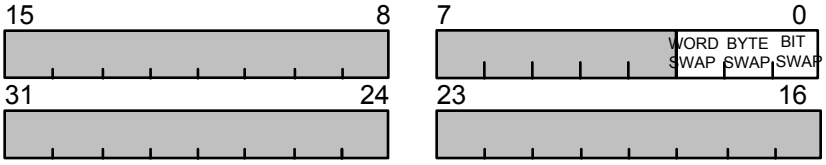
BITS	NAME	VALUE	FUNCTION
CMD_CLP[1:0]	CCTRL[1:0]	0x0 0x1 0x2 0x3	No clipping No clipping Draw inside rectangle Draw outside rectangle
CMD_CLP[2]	CSTOP	0 1	Don't stop on clip boundary Stop on clip boundary

Host Data Format Register RBASE_D +(0x0064)

Name: CMD_HDF

Type: Memory mapped read write

This register controls how host data is passed to or from the drawing engine cache. It doesn't affect drawing engine registers. If all bits within this register are set to 0, the host data is unmodified. Setting the WORD_SWAP bit will cause the upper word of data to be exchanged with the lower word. The BYTE_SWAP bit will reverse the bytes within each word. The BIT_SWAP bit will reverse the bit ordering within a byte: in byte 0, bits[7:0] will be passed as bits[0:7]. All combinations of "SWAP" bits are valid.

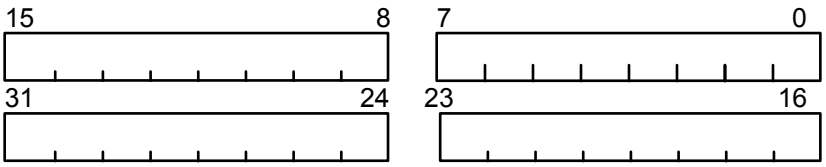


BITS	NAME	VALUE	FUNCTION
CMD_HDF[0]	BIT_SWAP	0x0 0x1	Bit ordering is not modified Swap bits within each byte
CMD_HDF[1]	BYTE_SWAP	0x0 0x1	Byte ordering is not modified Swap bytes within each word
CMD_HDF[2]	WORD_SWAP	0x0 0x1	Word ordering is not modified Swap words within each dword

Name: FORE

Type: Memory mapped read write

The FORE register holds the foreground color used during line drawing and stipple operations as well as any operation with the SOLID bit set. This register is 32 bits at 32 bits per pixel, 16 bits at 16 bits per pixel, and 8 bits at 8 bits per pixel.

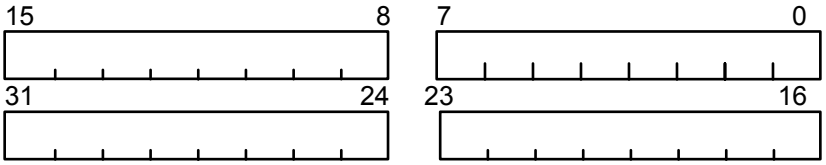


Background RBASE_D +(0x006C)

Name: BACK

Type: Memory mapped read write

The BACK register holds the background color used during patterned line and stippled operations. During these operations, a zero value in the pattern or stipple will select the BACK register as the source to be written out as the pixel value. This register is not used if transparency is set. This register is 32 bits at 32 bits per pixel, 16 bits at 16 bits per pixel, and 8 bits at 8 bits per pixel.



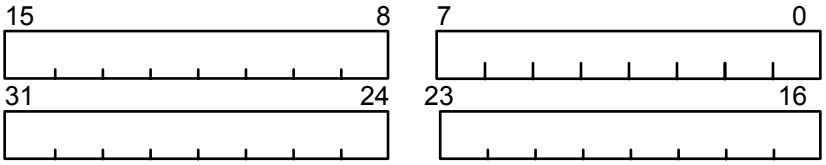
Plane Mask RBASE_D +(0x0070)

Name: MASK

Type: Memory mapped read write

The MASK register allows for selective writing of pixel planes. Using this feature, bit maps may be partitioned into logical groups for selective updating. A bit value of zero disables writing while a one enables writing to the corresponding plane. <<Ask Jim about masking for different bit depths>>

NOTE: The MASK registers will have no effect if the memory clips used with Borealis does not support the write-per-bit feature.

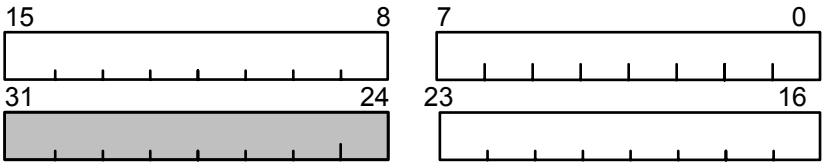


Color Key Register Mask RBASE_D +(0x0074)

Name: DE_KEY

Type: Memory mapped read write

The DE_KEY register provides a 24 bit register for a 2D color key compare. The 2D Key control register is located in BUF_CTRL.



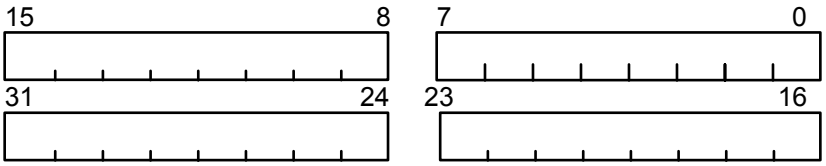
Line Pattern Register RBASE_D +(0x0078)

Name: LPAT

Type: Memory mapped read write

The LPAT register holds a 32 bit pattern in effect during line drawing. A one in the LPAT register corresponds to rendering in the foreground color while a zero corresponds to the background color or

destination color if transparency is enabled. LPAT[0] represents the first bit of the pattern in a line assuming that the pattern reset bit is set in the CMD_PAT register. Solid lines may be drawn by setting the pattern register to 0xffffffff and setting Shade bit to Zero or by setting the SOLID bit in the CMD_STYLE register.

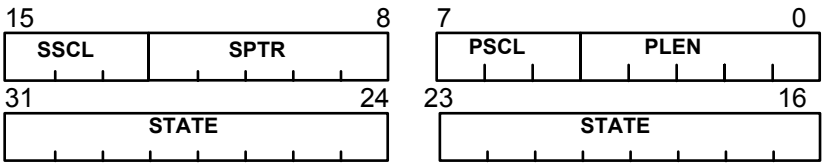


Line Pattern Control Register RBASE_D +(0x007C)

Name: PCTRL

Type: Memory mapped read write

The pattern control register allows the modifying of the pattern in LPAT by specifying a scaling factor, a length, and specifying a starting condition. The SSCL field (Starting Scale factor) allows for accurate alignment of scaled patterns.



BITS	NAME	VALUE FUNCTION
PCTL[4:0]	PLEN	Pattern length. The length assumes a scale factor of 1x. 0x0 32 bit long pattern 0x1 1 bit long pattern ... 0x1f 31 bit long pattern
PCTL[7:5]	PSCL	Pattern scale factor (stretch) 0x0 x1 scaling 0x1 x2 scaling ... 0x7 x8 scaling
PCTL[12:8]	SPTR	Pattern offset 0x00 1st bit of pattern is LPAT[0] 0x01 1st bit of pattern is LPAT[1] ... 0x1f 1st bit of pattern is LPAT [31]
PCTL[15:13]	SSCL	Initial scale offset 0x0 no offset 0x1 offset by one ... 0x7 offset by seven
PCTL[31:16]	STATE	State always contains the current state of the pattern controller. By reading the contents of the field and moving it directly to {SSCL,SPTR,PSCL,PLEN} the line pattern will continue from exactly where it left off. This is very useful for context switching.

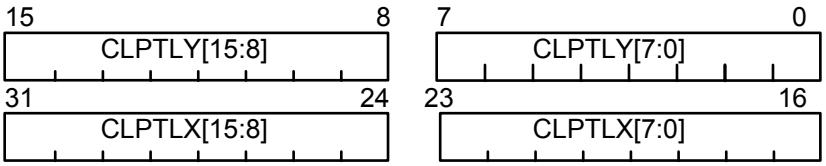
<<Jim, what is the intention of the STATE field in PCTL>>

Top Left of Clip Area RBASE_D + (0x0080)

Name: CLPTL

Type: Memory mapped read write

The CLPTL defines the upper left corner of the clipping rectangle.



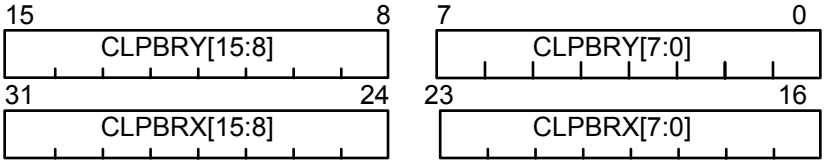
BITS	NAME	FUNCTION
CLPTL[15:0]	CLPTLY	Y coordinate of the top left corner of the clipping rectangle.
CLPTL[31:16]	CLPTLX	X coordinate of the top left corner of the clipping rectangle.

Bottom Right of Clip Area RBASE_D + (0x0084)

Name: CLPBR

Type: Memory mapped read write

The CLPBR defines the bottom right corner of the clipping rectangle.



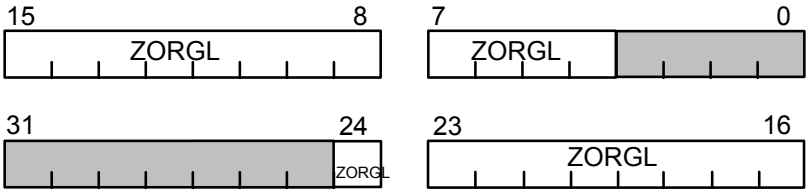
BITS	NAME	FUNCTION
CLPBR[15:0]	CLPBRY	Y coordinate of the bottom right corner of the clipping rectangle.
CLPBR[31:16]	CLPBRX	X coordinate of the bottom right corner of the clipping rectangle.

Drawing Engine Z Origin RBASE_D + (0x0100)

Name: DE_ZORG

Type: Memory mapped read write

This register defines the linear address offset into the selected buffer or buffers for drawing engine Z cycles. This is a 16 byte-aligned address when accessing the Display Buffer.



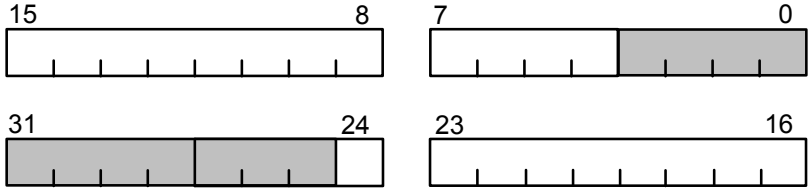
BITS	NAME	VALUE	FUNCTION
DE_ZORG[24:4]	ZORGL		Linear origin of the Z buffer

Drawing Engine MipMap Origins RBASE_D + (0x00D0 through 0x00F4)

Name: LOD0_ORG, LOD1_ORG, LOD2_ORG, LOD3_ORG, LOD4_ORG, LOD5_ORG, LOD6_ORG, LOD7_ORG, LOD8_ORG, LOD9_ORG

Type: Memory mapped read write

This register defines the linear address offset into the selected buffer or buffers for Texture mapping with Mip-Mapping enabled. When accessing the Display Buffer, this appears as a 128-bit word address. LOD0_ORG must be loaded with the origin of the largest mipmap (or the single texture map origin in non-mipmapped mode) with the smallest mipmap in a higher numbered origin. Each mipmap must be a power of 2 smaller than the previous mipmap with the largest mipmap size specified in TEX_CNTRL and the number of mipmaps also specified. The Texture Pitch (TPTCH) must also be specified for LOD0. Borealis then calculates the pitches for all subsequent mipmap levels. The pitch must be at a 128 bit boundary, so when the pitch falls below 128, the pitch still is 128. Therefore, a 2x2 texture at 32 bpp still would take up 2x128, (1/2 is not used). (Pixels are packed in the lower order bits, i.e. in 32bpp, the 2 pixels will be loaded in [63:0] in each 128-bit row.) Borealis supports up to 10 LOD's to be used (512x512 to 1x1) when mipmapping is enabled. Rectangular power of 2 Mipmaps are supported.



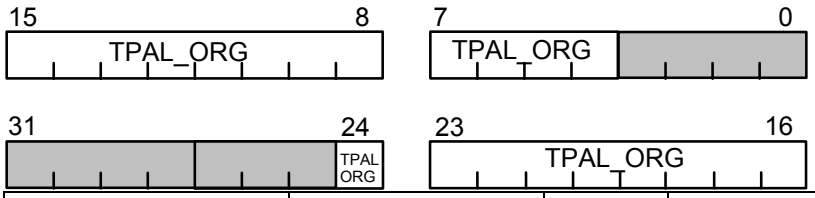
BITS	NAME	VALUE	FUNCTION
LOD0_ORG[24:4]	LOD0_ORG		Linear origin mode for LOD0
LOD1_ORG[24:4]	LOD1_ORG		Linear origin mode for LOD1
LOD2_ORG[24:4]	LOD2_ORG		Linear origin mode for LOD2
LOD3_ORG[24:4]	LOD3_ORG		Linear origin mode for LOD3
LOD4_ORG[24:4]	LOD4_ORG		Linear origin mode for LOD4
LOD5_ORG[24:4]	LOD5_ORG		Linear origin mode for LOD5
LOD6_ORG[24:4]	LOD6_ORG		Linear origin mode for LOD6
LOD7_ORG[24:4]	LOD7_ORG		Linear origin mode for LOD7
LOD8_ORG[24:4]	LOD8_ORG		Linear origin mode for LOD8
LOD9_ORG[24:4]	LOD9_ORG		Linear origin mode for LOD9

Drawing Engine Palette Origin RBASE_D + (0x0118)

Name: DE_TPALORG

Type: Memory mapped read write

This register defines the linear address offset into the selected buffer or buffers for the texture cache palette. This is a 128 bit word address when accessing the Display Buffer.



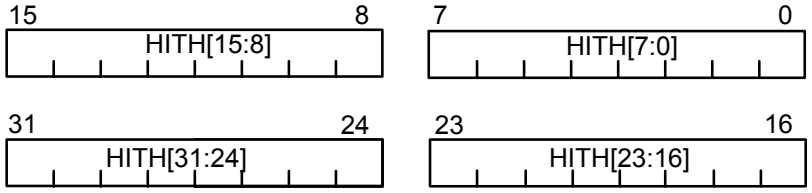
BITS	NAME	VALUE	FUNCTION
DE_TPALORG[24:4]	TPAL_ORG		Linear origin of the Texture Palette

HITHER Clip Plane RBASE_D + (0x011C)

Name: HITH

Type: Memory mapped read write

The HITH defines the front most clipping plane.



BITS	NAME	FUNCTION
HITH[31:0]	HITH	Z coordinate hither clipping plane.

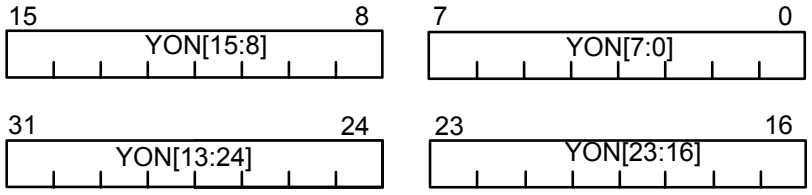
* 23:0 for 24 bit Z, or 15:0 for 16 bit Z.

YON Clip plane RBASE_D + (0x0120)

Name: YON

Type: Memory mapped read write

The YON defines the back most clipping plane.



BITS	NAME	FUNCTION
YON[31:0]	YON	Z coordinate YON clipping plane.

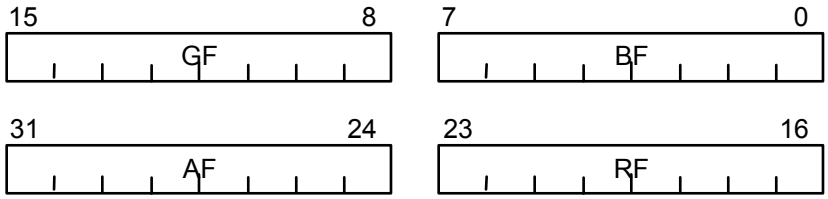
* 23:0 for 24 bit Z, or 15:0 for 16 bit Z.

Fog Color Register RBASE_D + (0x0124)

Name: FOG_COL

Type: Memory mapped read write

The color of the fog which is present across a primitive. It is specified in an ARGB format.



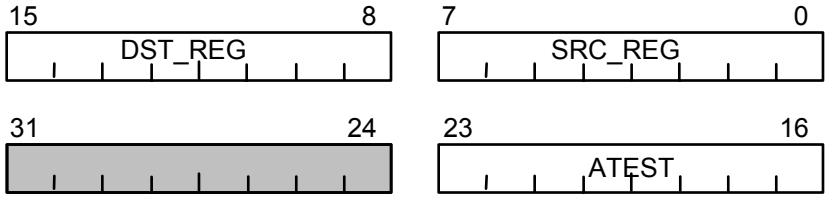
BITS	NAME	FUNCTION
FOG_COL[7:0]	BF	Blue component of the Fog Color
FOG_COL[15:8]	GF	Green component of the Fog Color
FOG_COL[23:16]	RF	Red component of the Fog Color
FOG_COL[31:24]	AF	Alpha component of the Fog Color

Alpha Register RBASE_D + (0x0128)

Name: ALPHA_REG

Type: Memory mapped read write

The Alpha register contains the alpha test value for alpha compare mode. It also contains the source and destination alpha values that are used for alpha blending under the control of ACNTRL.



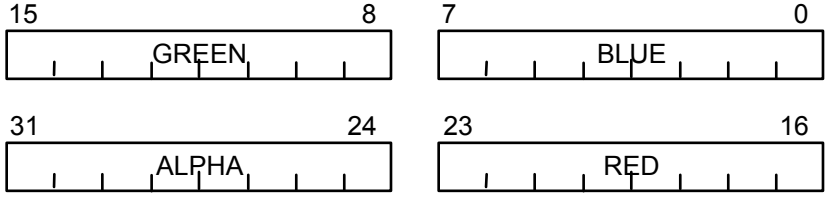
BITS	NAME	FUNCTION
ALPHA[7:0]	SRC_REG	Alpha source register for blending. Lower three bits are considered 0.
ALPHA[15:8]	DST_REG	Alpha destination register for blending. Lower 3 bits are considered 0.
ALPHA[23:16]	ATEST	Alpha test register for alpha compare
ALPHA[31:24]	Reserved	Reserved

Texture Border Color Register RBASE_D + (0x012C)

Name: TBORD_COL

Type: Memory mapped read write

The Texture border color register contains the RBG value to clamp the color value to when Bordering is turned on in TEX_CTRL and the texture map U,V coordinates extend beyond the size of the texture map.

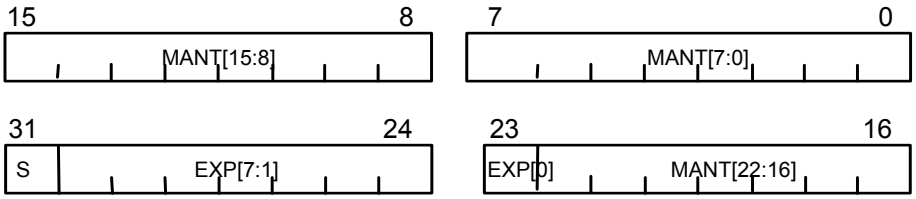


Floating Point Interface to the color registers RBASE_D + (0x0130 – 0x015C)

Name: V0_A_FP, V0_R_FP, V0_G_FP, V0_B_FP, V1_A_FP, V1_R_FP, V1_G_FP, V1_B_FP, V2_A_FP, V2_R_FP, V2_G_FP, V2_B_FP

Type: Memory mapped write only

In Borealis, an interface is in place which will allow the three triangle vertex color registers to be loaded using floating point colors. This removes software overhead when updating vertices and removes the float to fixed conversion that software must do for these registers. These registers are write only. These floating point color registers can be used at any time, for a given command, it should either all be floating point or all integer.

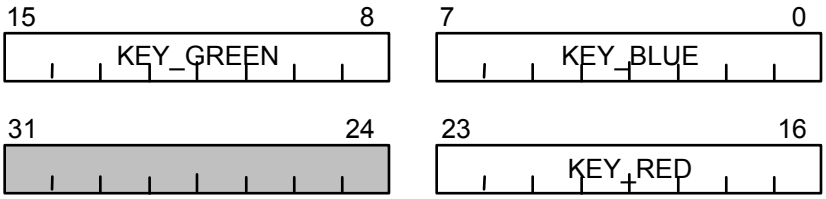


3D Color key compare registers RBASE_D + (0x0160, 0x0164)

Name: KEY_3D_LOW, KEY_3D_HIGH

Type: Memory mapped read write

Borealis allows 3D color keying to be performed on a range of colors. This range is specified in the two 3D color key registers. The range can be either between the two key colors, or outside the range of the two key colors. In texture mapping, 3D color keying is always done on the nearest Texel prior to any additive operations or filtering.

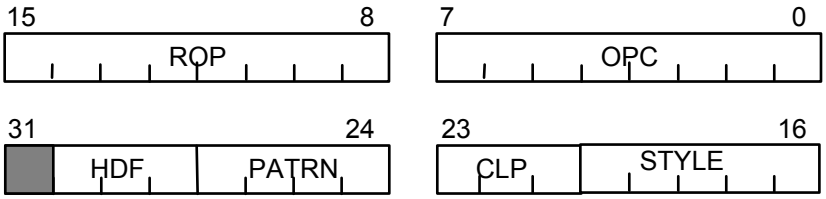


Command Register RBASE_D +(0x0168) (Shadowed with 0x0048)

Name: CMD

Type: Memory mapped read write

The drawing engine command register can be accessed as a single 32 bit register or individual fields can be accessed in their own register space as described in the following pages.



BITS	NAME	FUNCTION
CMD[7:0]	OPC	Drawing command opcode
CMD[15:8]	ROP	Drawing Raster or logical operation
CMD[20:16]	STYLE	Solid, Transparency, Stipple control
CMD[23:21]	CLP	Clipping control
CMD[27:24]	PATRN	No last, pat reset, area pattern, and pattern cache enable.
CMD[30:28]	HDF	Host Data Format
CMD[31]	Reserved	Reserved

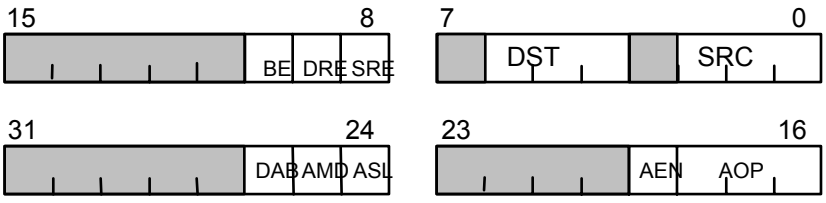
NOTE: See CMD at 0x0050 - 0x0064 for specification of each CMD field.

Alpha Control RBASE_D +(0x016C)

Name: ACNTRL

Type: Memory mapped read write

The alpha control register sets up the Imagine 4 blending unit for use with the drawing engine commands.



BITS	NAME	VALUE	FUNCTION
ACNTRL[2:0]	SRC	see below	Blending Source Function
ACNTRL[6:4]	DST	see below	Blending Destination Function
ACNTRL[8]	SRE	 0x0 0x1	Source Register Enable Use Alpha in the Pixel Use source alpha register

BITS	NAME	VALUE	FUNCTION
ACNTRL[9]	DRE		Destination Register Enable
		0x0	Use Alpha in the Pixel
		0x1	Use destination alpha register
ACNTRL[10]	BE		Blending Enable
		0x0	Disable blending
		0x1	Enable blending
ACNTRL[15:11]	Reserved	0x0	Reserved
ACNTRL[18:16]	AOP		Alpha operator
		0x0	Never.
		0x1	Always.
		0x2	Less than.
		0x3	Less than or equal.
		0x4	Equal.
		0x5	Greater than or equal.
		0x6	Greater than.
		0x7	Not Equal.
ACNTRL[19]	AEN		Alpha Compare
		0x0	Disabled
		0x1	Enabled
ACNTRL[23:20]	Reserved	0x0	Reserved
ACNTRL[24]	ASL		Alpha Select
		0x0	Selects Texture Alpha
		0x1	Selects Vertex Alpha
ACNTRL[25]	AMD		Alpha Modulate
		0x0	Disabled
		0x1	Enabled
ACNTRL[26]	DAB		Decal Alpha Blend Mode
		0x0	Disabled
		0x1	Enabled
ACNTRL[31:27]	Reserved	0x0	Reserved

The Borealis alpha blending modes follow the standard OpenGL conventions which are outlined in the following two tables:

Table 3-1.1. Source Blending Table

SOURCE FACTOR	DEFINITION	OPERATION
0x0	SRC_ZERO	(0,0,0,0)
0x1	SRC_ONE	(1,1,1,1)
0x2	SRC_DST_COLOR	(Ad,Rd,Gd,Bd)
0x3	SRC_ONE_MINUS_DST	(1,1,1,1)-(Ad,Rd,Gd,Bd)
0x4	SRC_SRC_ALPHA	(As,As,As,As)
0x5	SRC_ONE_MINUS_SRC_ALPHA	(1,1,1,1)-(As,As,As,As)
0x6	SRC_DST_ALPHA	(Ad,Ad,Ad,Ad)
0x7	SRC_ONE_MINUS_DST_ALPHA	(1,1,1,1)-(Ad,Ad,Ad,Ad)

Table 3-1.2. Destination Factor Table

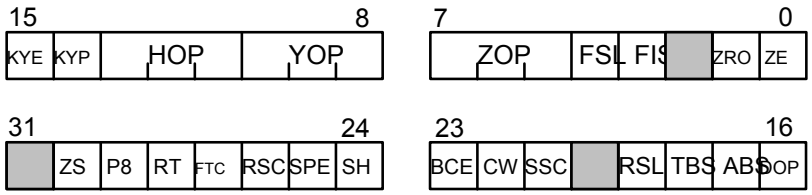
DESTINATION FACTOR	DEFINITION	OPERATION
0x0	DST_ZERO	(0,0,0,0)
0x1	DST_ONE	(1,1,1,1)
0x2	DST_SRC_COLOR	(As,Rs,Gs,Bs)

DESTINATION FACTOR	DEFINITION	OPERATION
0x3	DST_ONE_MINUS_SRC	(1,1,1,1)-(As,Rs,Gs,Bs)
0x4	DST_SRC_ALPHA	(As,As,A,s,As)
0x5	DST_ONE_MINUS_SRC_ALPHA	(1,1,1,1)-(As,As,As,As)
0x6	DST_DST_ALPHA	(Ad,Ad,Ad,Ad)
0x7	DST_ONE_MINUS_DST_ALPHA	(1,1,1,1)-(Ad,Ad,Ad,Ad)

3D Control Register RBASE_D +(0x0170)

Name: 3D_CTRL

Type: Memory mapped read write



BITS	NAME	VALUE	FUNCTION
3D_CTRL[0]	ZE		Z buffer enable bit.
		0x0	No Z buffer update or compare.
		0x1	Z buffer compare enabled, and updated if ZRO = 0.
3D_CTRL[1]	ZRO		Z read only mode.
		0	Updating of Z buffer enabled.
		1	Z buffer in read only mode.
3D_CTRL[2]	Reserved	Reserved	Reserved
3D_CTRL[3]	FIS		Fog Index Select
		0x0	Selects Z as Index to Fog Table
		0x1	Selects 1/w as Index to Fog Table
3D_CTRL[4]	FSL		Fog Selector
		0x0	Vertex Fogging
		0x1	Table based Fogging
3D_CTRL[7:5]	ZOP		Z operator
		0x0	Never.
		0x1	Always.
		0x2	Less than.
		0x3	Less than or equal.
		0x4	Equal.
		0x5	Greater than or equal.
		0x6	Greater than.
		0x7	Not Equal.
3D_CTRL[10:8]	YOP		YON operator
		0x0	Never.
		0x1	Always.
		0x2	Less than.
		0x3	Less than or equal.
		0x4	Equal.
		0x5	Greater than or equal.
		0x6	Greater than.
		0x7	Not Equal.

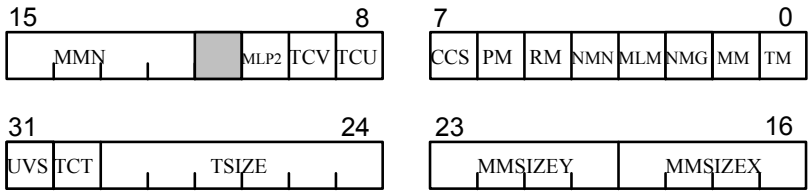
BITS	NAME	VALUE	FUNCTION
3D_CTRL[13:11]	HOP		Hither operator
		0x0	Never.
		0x1	Always.
		0x2	Less than.
		0x3	Less than or equal.
		0x4	Equal.
		0x5	Greater than or equal.
		0x6	Greater than.
		0x7	Not Equal.
3D_CTRL[14]	KYP		3D Key Polarity
		0x0	Inclusive (In range)
		0x1	Exclusive (Out of Range)
			NOTE: 3D and 2D keying are not exclusive.
3D_CTRL[15]	KYE		3D Key Enable
		0x0	3D Key Disabled
		0x1	3D Key Enabled
			NOTE: 3D keying must be disabled when texture mapping is disabled.
3D_CTRL[16]	DOP		Dither Operator
		0x0	No Dithering
		0x1	8x8
3D_CTRL[17]	ABS		Alpha Blend Select
		0x0	Disabled
		0x1	Enabled
3D_CTRL[18]	TBS		Texture Blend Select
		0x0	Disabled
		0x1	Enabled
3D_CTRL[19]	RSL		RGB Select
		0x0	Selects Texture RGB
		0x1	Selects Vertex RGB
3D_CTRL[20]	Reserved	Reserved	Reserved
3D_CTRL[21]	SSC		XY Setup Spatial Center
		0x0	Integer Centered
		0x1	0.5 Centered
3D_CTRL[22]	CW		CW/CCW selector for culling
		0x0	CW
		0x1	CCW
3D_CTRL[23]	BCE		Backface culling Enable
		0x0	Disabled
		0x1	Enabled
3D_CTRL[24]	SH		Gouraud Shading
		0x0	Disabled
		0x1	Enabled
3D_CTRL[25]	SPE		Specular Lighting
		0x0	Disabled
		0x1	Enabled
3D_CTRL[26]	RSC		UV Renderer Spatial Center
		0x0	Centered
		0x1	0.5 Centered

BITS	NAME	VALUE	FUNCTION
3D_CTRL[27]	FTC	0x0 0x1	Fog/Texture Composite Mode Disabled Enabled **NOTE**: When Mipmap Linear is enabled, this mode bit becomes the Texture Composite Mode bit.
3D_CTRL[28]	RT	0x0 0x1	Rectangle Draw triangle in normal mode. Draw rectangle (P0, P1 and P2 must be sorted)
3D_CTRL[29]	P8	0x0 0x1	8 Bits Per Pixel Format Mode Normal 8bpp Mode (Format = Blue Channel) **Example** Source Color 44556677 (Hex) Destination Color 77 (Hex) 332 8bpp Mode (Format = 332) **Example** Source Color 44556677 (Hex) 01000100 01010101 01100110 01110111 (Binary) 010 011 01 (Binary) 01001101 (Binary) Destination Color 4D (Hex)
3D_CTRL[30]	ZS	0x0 0x1	Z Scaling mode Z is not scaled Z is scaled
3D_CTRL[31]	Reserved	Reserved	Reserved

Texture Mapping Control Register RBASE _(D) + (0x0174)

Name: TEX_CNTRL

Type: Memory mapped read write



BITS	NAME	VALUE	FUNCTION
TEX_CTRL[0]	TM	0x0 0x1	Texture Mode Texture mapping disabled Texture mapping enabled
TEX_CTRL[1]	MM	0x0 0x1	Mipmap mode Mipmapping disabled Mipmapping enabled

BITS	NAME	VALUE	FUNCTION
TEX_CTRL[2]	NMG		Nearest Mode Magnification
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[3]	MLM		MipMap Linear Mode
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[4]	NMN		Nearest Mode Minification
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[5]	RM		RGB Modulation
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[6]	PM		Perspective Correction Mode
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[7]	CCS		Clamp Color Selector
		0x0	Use Edge Color
		0x1	Use Border Color
TEX_CTRL[8]	TCU		Texture Clamp in U
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[9]	TCV		Texture Clamp in V
		0x0	Disabled
		0x1	Enabled
TEX_CTRL[10]	MLP2		MipMap Linear Pass
		0x0	Pass 1
		0x1	Pass 2
TEX_CTRL[15:12]	MMN		Number of Mipmaps
		0x0	1 mipmap
		0x1	2 mipmaps
		0x2	3 mipmaps
		0x3	4 mipmaps
		0x4	5 mipmaps
		0x5	6 mipmaps
		0x6	7 mipmaps
		0x7	8 mipmaps
		0x8	9 mipmaps
		0x9	10 mipmaps
		0xA-0xF	Reserved

BITS	NAME	VALUE	FUNCTION
TEX_CTRL[19:16]	MMSIZEX		MipMap X size
		0x0	1 texel
		0x1	2 texels
		0x2	4 texels
		0x3	8 texels
		0x4	16 texels
		0x5	32 texels
		0x6	64 texels
		0x7	128 texels
		0x8	256 texels
		0x9	512 texels
		0xA- 0xF	Reserved
			Note: Must be specified for all texture mapped commands.
TEX_CTRL[23:20]	MMSIZEY		MipMap Y size
		0x0	1 texel
		0x1	2 texels
		0x2	4 texels
		0x3	8 texels
		0x4	16 texels
		0x5	32 texels
		0x6	64 texels
		0x7	128 texels
		0x8	256 texels
		0x9	512 texels
		0xA- 0xF	Reserved
			Note: Must be specified for all texture mapped commands.

BITS	NAME	VALUE	FUNCTION
TEX_CTRL[29:24]	TSIZE		Texture map pixel format
			Palettized
		0x0	1 bpt 1555
		0x1	1 bpt 0565
		0x2	1 bpt 4444
		0x3	1 bpt 8888
		0x18	1 bpt 8332
		0x4	2 bpt 1555
		0x5	2 bpt 0565
		0x6	2 bpt 4444
		0x7	2 bpt 8888
		0x19	2 bpt 8332
		0x8	4 bpt 1555
		0x9	4 bpt 0565
		0xA	4 bpt 4444
		0xB	4 bpt 8888
		0xE	4 bpt 8332
		0xF	8 bpt 0565
		0x1A	8 bpt 1555
		0x1C	8 bpt 4444
		0x1D	8 bpt 8888 (Exact mode, must set to nearest mode)
		0x3E	8 bpt 8888 (auto converted to 4444 into palette)
		0x3F	8 bpt 8888 (auto converted to 0565 into palette)
		0x1E	8 bpt 8332
			Non-Palettized
		0xC	8 bpt 1232
		0xD	8 bpt 0332 (when 3D_CTRL [29] = 0)
		0xD	8-bit index (when 3D_CTRL [29] = 1)
		0x10	16 bpt 4444
		0x11	16 bpt 1555
		0x12	16 bpt 0565
		0x13	16 bpt 8332
		0x14	32 bpt 8888
			OpenGL modes
		0x20	Alpha4
		0x21	Alpha8
		0x24	Luminance4
		0x25	Luminance8
		0x28	Luminance4_Alpha4
		0x29	Luminance6_Alpha2
		0x2A	Luminance8_Alpha8
		0x2C	Intensity4
		0x2D	Intensity8
		0x30	RGBA2
		Everything else	Reserved
TEX_CTRL[30]	TCT		Texture Cache Tile Mode
		0x0	Disabled
		0x1	Enabled
			NOTE: Readback path is broken.
TEX_CTRL[31]	UVS		U-V Scaling
		0x0	Scale U and V Disabled
		0x1	Scale U and V Enabled

NOTE: All texels are reformatted to 8888 by replicating most significant bits into the lower order bits. as it is read from the cache, a 1555 texel (0xAAAA) is read as a 8888 texel (0xFF52AD52).

3D Command Trigger Register RBASE_D + (0x01DC)

Name: 3D_TRIG

Type: Memory mapped read write

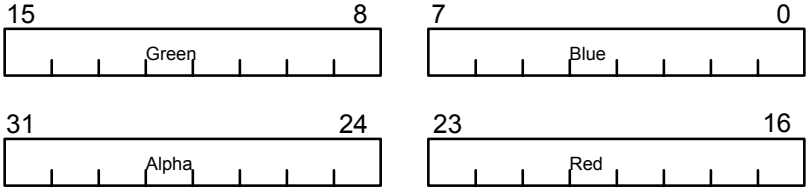
The 3D trigger register causes a 3D command to be loaded and executed when written to. The register needs to only be written to, no data need be supplied, and no data is contained within.

OpenGL Blend Color Register RBASE_D + (0x01E0)

Name: GLBLENDC

Type: Memory mapped read write

This color register is used in OpenGL for its Texture Blend Function. It is specified in an ARGB format.



BITS	NAME	FUNCTION
GLBLENDC[7:0]	Blue	Blue component of the OpenGL Blend Color
GLBLENDC[15:8]	Green	Green component of the OpenGL Blend Color
GLBLENDC[23:16]	Red	Red component of the OpenGL Blend Color
GLBLENDC[31:24]	Alpha	Alpha component of the OpenGL Blend Color

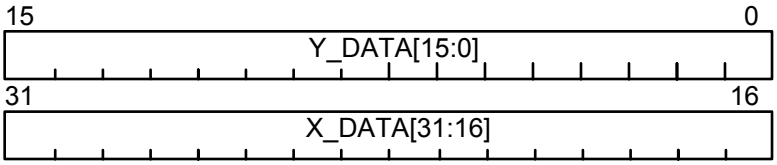
XY Parameter Registers RBASE_D + (0x0088 through 0x0098)

Name: XY0, XY1, XY2, XY3, XY4

Type: Memory mapped read write

The XY parameter registers are general purpose registers used to hold different data values based on the type of command to be executed. The Command set chapter describes the use of these registers for each command. The data in these registers is one of two data formats.

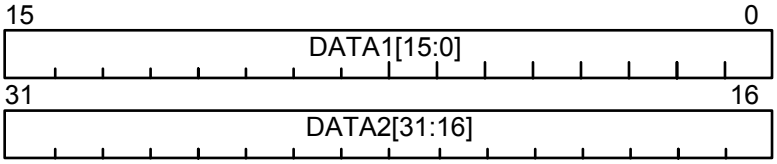
X-Y FORMAT



BITS	FUNCTION
Y_DATA[15:0]	Y coordinate of a pixel location, or the Y portion of a dimension. Y_DATA is an integer value with range of +32767 to -32767.

X_DATA[31:16]	X coordinate of a pixel location, or the X portion of a dimension. X_DATA is an integer value with range of +32767 to -32767.

I FORMAT



The I format contains two general purpose 16 bit integer data values. (Data can be less than 16 bits as in direction bits for bit blt.)

XY1 is the trigger register for all 2D drawing operations. All other XY registers and parameter registers should be programmed before writing to XY1. Although XY1 can be written to as a byte, word, or Dword, the most significant byte of XY1 must be written for a command to be triggered.

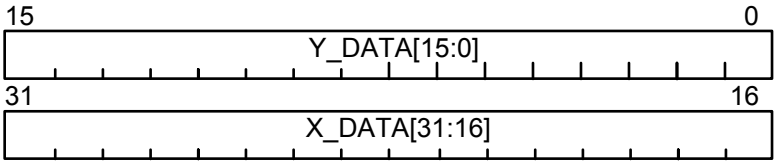
CP Parameter Registers RBASE_D + (0x0178 through 0x01D8)

Name: CP0 through CP24

Type: Memory mapped read write

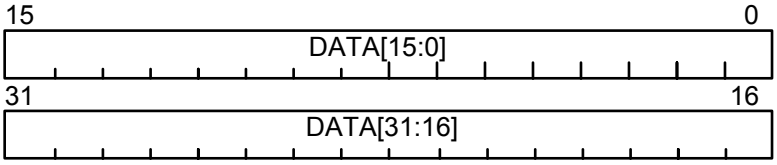
The CP parameter registers are general purpose registers used to hold different data values based on the type of command to be executed. The Command set chapter describes the use of these registers for each command. The data in these registers is one of three data formats.

X-Y FORMAT



BITS	FUNCTION
Y_DATA[15:0]	Y coordinate of a pixel location, or the Y portion of a dimension. Y_DATA is an integer value with range of +32767 to -32767.
X_DATA[31:16]	X coordinate of a pixel location, or the X portion of a dimension. X_DATA is an integer value with range of +32767 to -32767.

I FORMAT

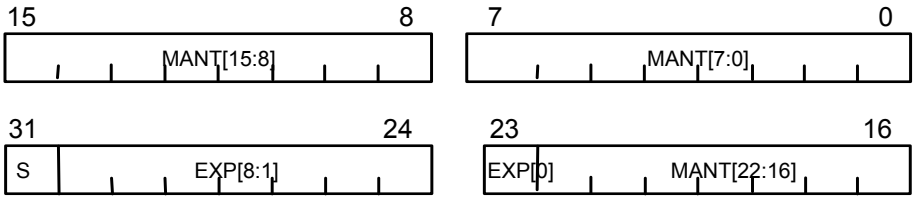


The I format contains one general purpose 32 bit integer data values.

3D_Trigger is the trigger register for all 3D operations. All other CP registers and parameter registers should be programmed before writing to 3D_TRIGGER.

Float Format

IEEE Single Precision Floating Point



S = Sign Bit
EXP = 8-bit Biased Exponent
MANT = 23 Bit MANTISSA with Implied Leading 1.

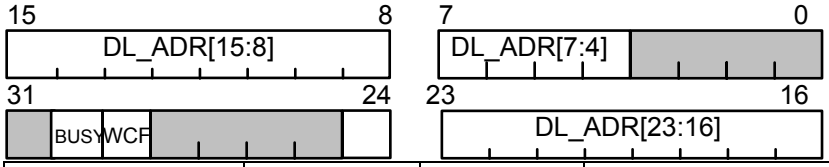
Display List Processor Registers

Display List Processor Address Register RBASE_D + (0x00F8)

Name: DL_ADR

Type: Memory mapped read write

This register defines the address in local RAM at which the display list processor will begin execution. A start address is written into DL_ADR at which time the DLP will accept end addresses which are associated with the start address. The list can be started or stopped at any time and the end address updated asynchronously. The DLP can also accept a new start address as a "pending list". Once the new start address is written, the DLP will continue executing the current list and seamlessly switch to the new list when the executing list is completed. All new end addresses written after a start address are associated with the new list, and the executing list is NOT accessible. Only 2 start addresses can be loaded at any time. A third start address must not be written when the busy bit (DL_ADR[30]) is set.



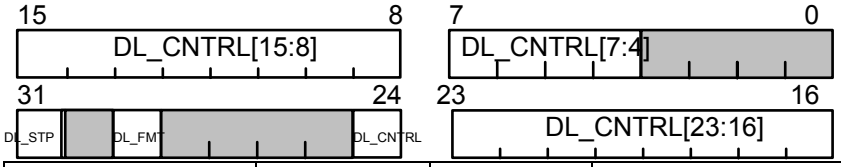
BITS	NAME	VALUE	FUNCTION
DL_ADR[3:0]	Reserved	0x0	Reserved
DL_ADR[24:4]	DL_ADR		Display List Start Address
DL_ADR[28:25]	Reserved	0	Reserved
DL_ADR[29]	WCF		Wait for Linear windows cache to flush - Wait for cache flush (wait for linear window's cache to flush before executing the list.)
		0x0	Enabled
		0x1	Disabled
DL_ADR[30]	BUSY		Display List Busy
		0x0	Not Busy (Can Accept a New Start address)
		0x1	Busy (Can Not Accept a New Start Address) **Note: Do Not Send a New Start Address when this bit is set.**
DL_ADR[31]	Reserved	0x0	Reserved

Display List Processor Control Register RBASE_D + (0x00FC)

Name: DL_CNTRL

Type: Memory mapped read write

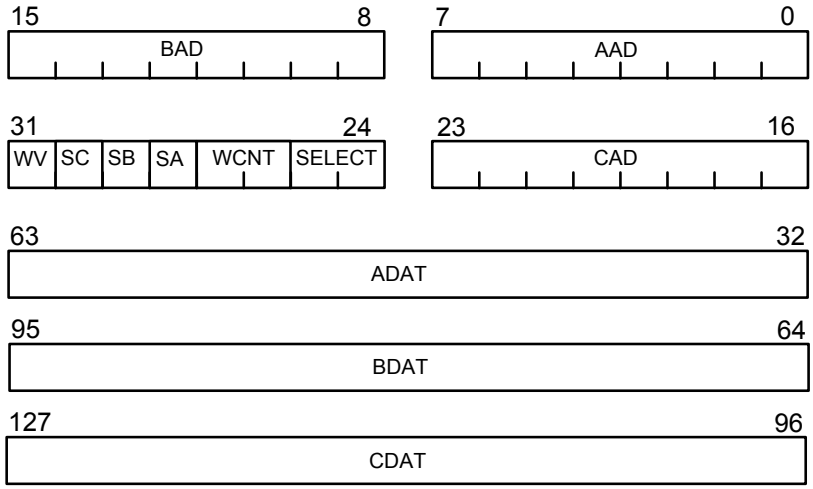
This register contains the control bits and the termination address of the display list.



BITS	NAME	VALUE	FUNCTION
DL_CNTRL[3:0]	Reserved	0x00	Reserved
DL_CNTRL[24:4]	DL_EAD		Display List End Address, When the Display List Processor reaches this address it terminates execution and sets the DL_STP bit.
DL_CNTRL[28:25]	Reserved	0x0	Reserved
DL_CNTRL[29]	DL_FMT	0x0 0x1	Display List Format Display List Format 0, Write register defined in display list. Display List Format 1, Write only XY0,XY2,XY3,XY1.
DL_CNTRL[30]	Reserved	0x0	Reserved
DL_CNTRL[31]	DL_STP	0x0 0x1	Display List Processor Stop. This read/write bit is used to control the operation of the DLP. DLP Trigger. Writing a 0 will trigger the DLP. It also indicates that the DLP is not idle. DLP is IDLE. Borealis will write a 1 when the DLP is done. **NOTE**: If a 1 is written by the user, it will halt the DLP.

Display List Instruction Word

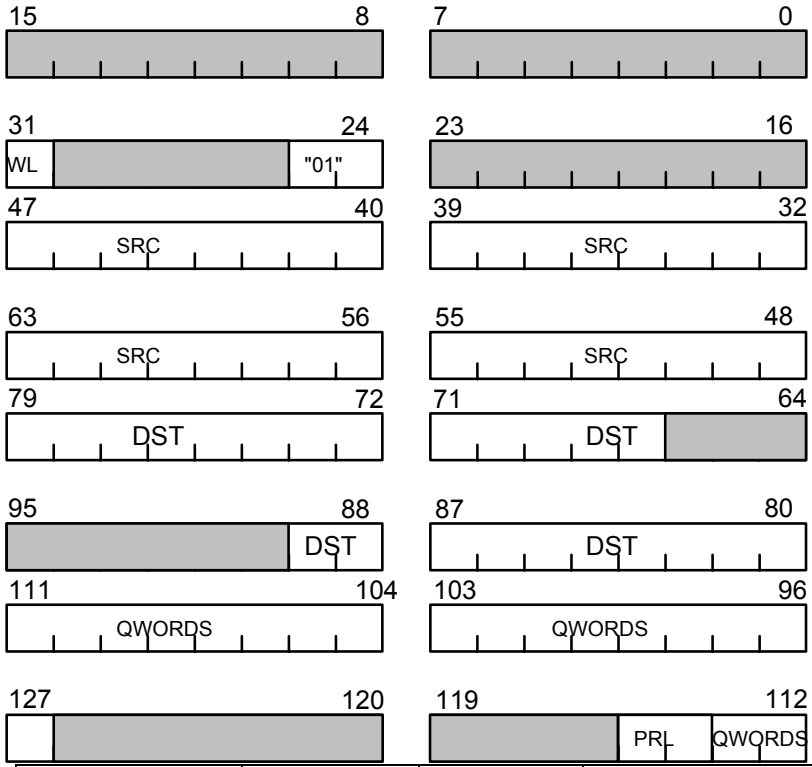
(Format Zero REG3 format "DL_FMT = 0 & DL_IW[25:24] =00")



BITS	NAME	VALUE	FUNCTION
DL_IW[7:0]	AAD		Address for register A data.
DL_IW[15:8]	BAD		Address for register B data.
DL_IW[23:16]	CAD		Address for register C data.
DL_IW[25:24]	SELECT	0x0	Must be 0 for this format

BITS	NAME	VALUE	FUNCTION
DL_IW[27:26]	WCNT		Word Count
		0x0	3 Words
		0x1	1 Word
		0x2	2 Words
		0x3	3 Words
DL_IW[28]	SA		Register A Address Selector
		0x0	DE registers 0x000 through 0x0FC
		0x1	DE registers 0x100 through 0x1FC
DL_IW[29]	SB		Register B Address Selector
		0x0	DE registers 0x000 through 0x0FC
		0x1	DE registers 0x100 through 0x1FC
DL_IW[30]	SC		Register C Address Selector
		0x0	DE registers 0x000 through 0x0FC
		0x1	DE registers 0x100 through 0x1FC
DL_IW[31]	WV		Wait for vertical blank
		0x0	Do not wait for vertical interrupt before executing current command.
		0x1	Wait for vertical interrupt before executing current command.
DL_IW[63:32]	ADAT		Data to be written to register addressed by AAD
DL_IW[95:64]	BDAT		Data to be written to register addressed by BAD
DL_IW[127:96]	CDAT		Data to be written to register addressed by CAD

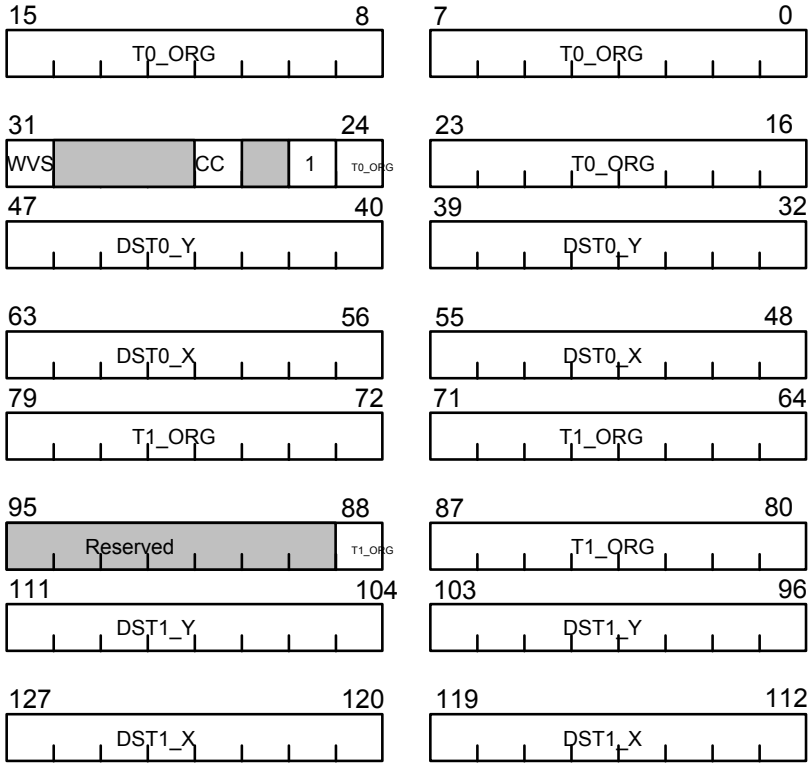
(Format Zero DMA “DL_FMT = 0 & DL_IW[25:24] = 01”)



BITS	NAME	VALUE	FUNCTION
DL_IW[23:0]	Reserved	0x0	Reserved
DL_IW[25:24]	SELECT	0x01	Must be 01 for this format
DL_IW[30:26]	Reserved	0x0	Reserved

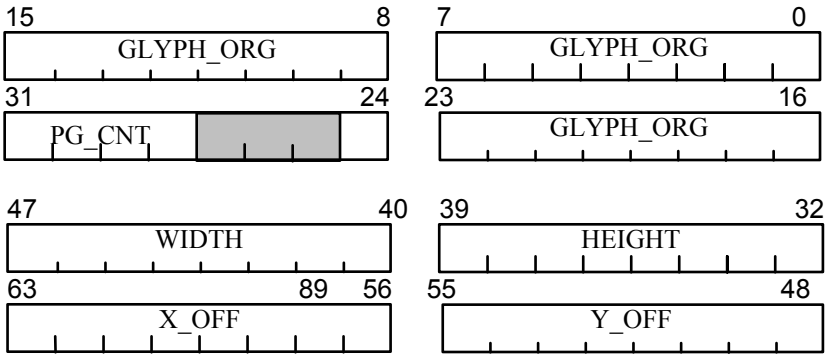
BITS	NAME	VALUE	FUNCTION
DL_IW[31]	WL	 0x0 0x1	Wait for AGP DMA done Don't wait for current (and previous DMA's) to finish Wait for pending and this DMA to finish
DL_IW[63:32]	SRC		AGP DMA Source (see DMA_SRC)
DL_IW[66:64]	Reserved	0x0	Reserved
DL_IW[89:67]	DST		AGP DMA destination (see DMA_DST)
DL_IW[95:90]	Reserved	0x0	Reserved
DL_IW[113:96]	QWORDS		Number of QWORDS Requested (do not set to 0)
DL_IW[115:114]	PRL	 00 01 10 11	Preferred Request Length for AGP transfers 4 Qwords 8 Qwords Reserved Reserved (See Appendix E, *Errata* for additional information.) **NOTE**: A PRL = "00" is an illegal combination for 2x transfers. ("00" will be effectively promoted to "01")
DL_IW[126:116]	Reserved	0x0	Reserved
DL_IW[127]	EX	 0x0 0x1	Expedite Normal Expedited

(Format Zero TEXT "DL_FMT = 0 & DL_IW[25] =1")



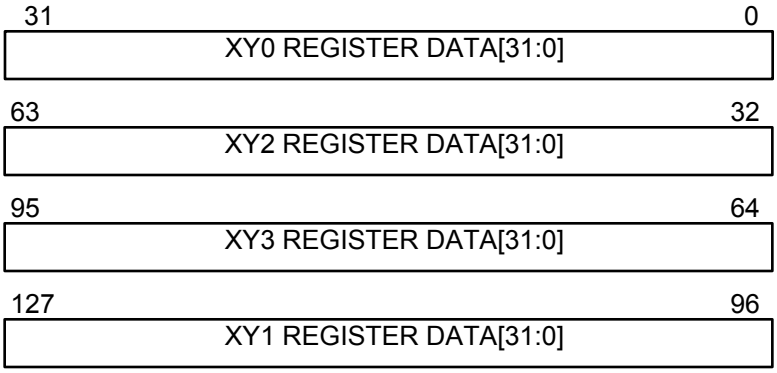
BITS	NAME	VALUE	FUNCTION
DL_IW[24:0]	T0_ORG		Table 0 Origin
DL_IW[25]	1	0x1	Must be '1' for this format
DL_IW[26]	Reserved	0x0	Reserved
DL_IW[27]	CC	 0x0 0x1	Character Count One character Two characters
DL_IW[30:28]	Reserved	0x0	Reserved
DL_IW[31]	WVS	 0x0 0x1	Wait for vertical sync after this character Disabled Enabled
DL_IW[47:32]	DST0_Y		Destination 0 Y
DL_IW[63:48]	DST0_X		Destination 0 X
DL_IW[88:64]	T1_ORG		Table 1 Origin
DL_IW[95:89]	Reserved	0x0	Reserved
DL_IW[111:96]	DST1_Y		Destination 1 Y
DL_IW[127:112]	DST1_X		Destination 1 X

The DLP uses the concept of "text tables" which contain information about character glyphs stored in off screen memory. A table entry is 64 bits long and can be packed such that two entries can exist in a 128 bit memory word. The origin in the DLP instruction points to the 64 bit memory word which contains the text entry. This 64-bit table entry must be aligned to a 8-byte boundary. The definition of the table is as follows:



BITS	NAME	VALUE	FUNCTION
TEXT[24:0]	GLYPH_ORG		Glyph Origin loaded into SORG
TEXT[27:25]	Reserved	0x0	Reserved
TEXT[31:28]	PG_CNT		Page Count loaded into XY3
TEXT[39:32]	Height		Height loaded into XY2
TEXT[47:40]	Width		Width, Loaded into XY2
TEXT[55:48]	Y_OFF		Y offset, subtract from dest_y
TEXT[63:56]	X_OFF		X Offset, add to dest_x

(Format One "DL_FMT = 1")



BITS	NAME	VALUE	FUNCTION
DL_IW[31:0]	XY0_DAT		Data to be written to register XY0
DL_IW[63:32]	XY2_DAT		Data to be written to register XY2
DL_IW[95:64]	XY3_DAT		Data to be written to register XY3
DL_IW[127:96]	XY1_DAT		Data to be written to register XY1

DLP Notes

- Waiting for AGP DMA done will cause the DLP to hold the DE busy and will not execute another command until the DMA pipeline is empty
- HBI AGP DMA cannot be used at the same time as DLP AGP DMA. The AGP DMA pipeline must be empty prior to executing a DLP AGP DMA list.
- Text mode only updates the SORG, XY2, XY3, and XY1 registers. All other registers (including command) must be set prior to DLP text operations.
- Text, AGP DMA and 3 register writes may all be mixed in the same physical list. The 4 register mode must be executed exclusively in its own list.

Chapter 6: *Drawing Engine Command Set*

NOTE: The following is a special programming note:

Shadowed Register: CMD(0x0048) = CMD(0x0168)

Noop

Command Name: No operation
Command Mnemonic: NOOP
Command Description:
The **NOOP** command performs a null operation. The CHIP returns to its idle states.
Command Capabilities:
None

0x0	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY0			
XY1*			Command Trigger
XY2			
XY3			
XY4			

* Command Trigger

Table 3-1.1. Active Parameters

BUF_CTRL	-
DE_SORG	-
DE_DORG	-
DE_ZORG	-
DE_SPTCH	-
DE_DPTCH	-
DE_ZPTCH	-
CMD_OPC	0x0
CMD_ROP	-
CMD_STYLE	-
CMD_PATRN	-
CMD_CLP	-
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	-
BACK	-
MASK	-
DE_KEY	-
LPAT	-
PCTRL	-
CLPTL	-
CLPBR	-
ALPHA	-
A_CNTRL	-
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

BitBlt

Command Name: Bit Block Transfer
Command Mnemonic: BITBLT
Command Description:
The BITBLT command manipulates rectangular areas in the local buffers. The source or destination can be in the Display buffer. The BITBLT command requires the host to calculate the scanning direction and provide the correct corner of both the source and destination rectangles. The scanning direction will always be from left to right, top to bottom regardless of the DIR field for any bitblt with zoom or any packed mode bitblt.
Command Capabilities:

- Transparent BLT
- Raster operations as defined by the ROP field of the command register
- Fast area fills with selectable pixel values
- 32x32 and 8x8* area patterning
- Color expansion
- Screen door transparency in stipple mode
- Bit plane masking
- OpenGL Alpha Blending

* In 8bpp full color area patterning, 8x8 area pattern mode is treated as a 16x8 area pattern.

0x1	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY0	Source	X-Y	Corner of source
XY1*	Destination	X-Y	Corner of destination
XY2	Width/Height	X-Y	Width/Height of Rect
XY3**	DIR	I	Direction right justified
XY4	YZOOM	NA-Y	Zoom Factor

* Command Trigger

** Regarding monochrome expansion bitBLTs, XY3 register specifies the number of pages.

DIRECTION	CODE	CORNER REQUIRED
L → R, T →B	0x0	Upper Left
L → R, B →T	0x1	Bottom Left
R → L, T →B	0x2	Upper Right
R → L, B →T	0x3	Bottom Right

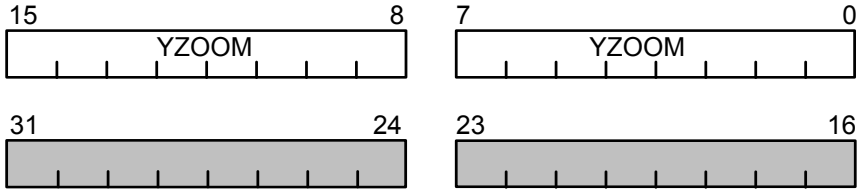
Table 3-1.2. ACTIVE Parameters

BUF_CTRL	✓
DE_SORG	✓
DE_DORG	✓
DE_ZORG	-
DE_SPTCH	✓
DE_DPTCH	✓
DE_ZPTCH	-
CMD_OPC	0x1
CMD_ROP	✓
CMD_STYLE	✓
CMD_PATRN	✓
CMD_CLP	✓
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	✓
BACK	✓
MASK	✓
DE_KEY	✓
LPAT	-
PCTRL	-
CLPTL	✓
CLPBR	✓
ALPHA	✓
A_CNTRL	✓
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-

TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

XY4 Register Zoom Data Format

This register contains the Y zoom factors for the BITBLT command.



BITS	NAME	VALUE	FUNCTION
XY4[15:0]	YZOOM		Y direction Zoom factor
		0x0 - 0x1	No Zoom
		0x2	2X Zoom
		0x3	3X Zoom
		0x4	4X Zoom
		0x5	5X Zoom
		0x6	6X Zoom
		0x7	7X Zoom
		0x8	8X Zoom
		0x9 - 0xFFFF	9X to 65535X Zoom

LINE

Command Name: Line
Command Mnemonic: LINE
Command Description:
The LINE command draws straight lines in to memory using a modified Bresenham's Algorithm to ensure that a line drawn from point A to point B will exactly match a line drawn from point B to point A. This command accepts directly in X-Y format the start and end point for a line.
Command Capabilities:

- 16 raster operations as defined by the ROP field of the command register
- Gouraud shading
- 32 bit scaled patterning
- Color expansion
- Screen door transparency
- Bit plane masking
- Open GL Alpha Blending

OPCODE

0x2

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY0	Source	X-Y	Start point of line
XY1*	Destination	X-Y	End point of line
XY2	NA		
XY3	NA		
XY4	NA		

* Command Trigger

Table 3-1.3. ACTIVE Parameters

BUF_CTRL	✓
DE_SORG	-
DE_DORG	✓
DE_ZORG	-
DE_SPTCH	-
DE_DPTCH	✓
DE_ZPTCH	-
CMD_OPC	0x2
CMD_ROP	✓
CMD_STYLE	✓
CMD_PATRN	✓
CMD_CLP	✓
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	✓
BACK	✓
MASK	✓
DE_KEY	✓
LPAT	✓
PCTRL	✓
CLPTL	✓
CLPBR	✓
ALPHA	✓
ACNTRL	✓
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

Line with Initial Error

Command Name: Draw line with initial error
Command Mnemonic: ELINE
Command Description:
The ELINE is a variant of the LINE command which allows the setting of the initial and incremental values of the Bresenham error term. All the characteristics of the LINE command also apply to the ELINE command. The ELINE may be used when sub-pixel accuracy is required. By pre-loading the error terms, a line segment can be drawn and the pixels will match exactly as if the entire line had been drawn. In addition, the ELINE command allows triangle boundaries to be drawn to match precisely the pixels drawn by the triangle command.
Command Capabilities:

- 16 raster operations as defined by the ROP field of the command register
- 32 bit scaled patterning
- Color expansion
- Screen door transparency
- Bit plane masking
- OpenGL Alpha Blending

0x3	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY0	Source	X-Y	Start point of line
XY1*	Destination	X-Y	End point of line
XY2	ERR	X-Y	Initial Error
XY3	ERRI	X-Y	Error Increment
XY4	NA		

ERR={(-major_delta),16'h0};
ERRI={2(major_delta),2(minor_delta)};

* Command Trigger

Table 3-1.4. ACTIVE parameters

BUF_CTRL	✓
DE_SORG	-
DE_DORG	✓
DE_ZORG	-
DE_SPTCH	-
DE_DPTCH	✓
DE_ZPTCH	-
CMD_OPC	0x3
CMD_ROP	✓
CMD_STYLE	✓
CMD_PATRN	✓
CMD_CLP	✓
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	✓
BACK	✓
MASK	✓
DE_KEY	✓
LPAT	✓
PCTRL	✓
CLPTL	✓
CLPBR	✓
ALPHA	✓
ACNTRL	✓
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

PLINE

Command Name: Poly Line
Command Mnemonic: PLINE
Command Description:
The PLINE command draws straight lines beginning at the destination of the last command. It uses modified Bresenham's Algorithm to ensure that a line drawn from point A to point B will exactly match a line drawn from point B to point A. This command accepts end point for a line in XY format.
Command Capabilities:

- 16 raster operations as defined by the ROP field of the command register
- Gouraud shading
- 32 bit scaled patterning
- Color expansion
- Screen door transparency
- Bit plane masking
- Open GL Blending

0x5	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY0	NA		
XY1*	Destination	X-Y	End point of line
XY2	NA		
XY3	NA		
XY4	NA		

* Command Trigger

Table 3-1.5. ACTIVE Parameters

BUF_CTRL	✓
DE_SORG	-
DE_DORG	✓
DE_ZORG	-
DE_SPTCH	-
DE_DPTCH	-
DE_ZPTCH	✓
CMD_OPC	0x5
CMD_ROP	✓
CMD_STYLE	✓
CMD_PATRN	✓
CMD_CLP	✓
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	✓
BACK	✓
MASK	✓
DE_KEY	✓
LPAT	✓
PCTRL	✓
CLPTL	✓
CLPBR	✓
ALPHA	✓
ACNTRL	✓
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

3D Lines with Setup

Command Name: Draw line with full setup.
Command Mnemonic: LINE_3D
Command Description: LINE_3D draws 3D Gouraud shaded, fogged, lit lines and points.
Command Capabilities:

- 16 raster operations as defined by the ROP field of the command register
- Specular Highlighting
- Fog
- Gouraud Shading
- Alpha blending and compare
- 16 or 32 bit Z buffering
- bit scaled patterning
- Color expansion
- Screen door transparency in stipple mode
- Bit plane masking

0x8	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
CP9	V1_X	Float	Vertex 0 X
CP10	V1_Y	Float	Vertex 0 Y
CP11	V1_Z	Float	Vertex 0 Z
CP12	Reserved	0x0	Reserved
CP13	V1_C	I	Vertex 0 Color {A, R, G, B}
CP14	V1_S	I	Vertex 0 Specular {F, Rs, Gs, Bs}
CP15	Reserved	0x0	Reserved
CP16	Reserved	0x0	Reserved
CP17	V2_X	Float	Vertex 1 X
CP18	V2_Y	Float	Vertex 1 Y
CP19	V2_Z	Float	Vertex 1 Z
CP20	Reserved	0x0	Reserved
CP21	V2_C	I	Vertex 1 Color {A, R, G, B}
CP22	V2_S	I	Vertex 1 Specular {F, Rs, Gs, Bs}
CP23	Reserved	0x0	Reserved
CP24	Reserved	0x0	Reserved
TRIGGER *	3D Trigger	NA	Triggers 3D commands

* Command Trigger

Table 3-1.5. ACTIVE Parameters

BUF_CTRL	✓
DE_SORG	-
DE_DORG	✓
DE_ZORG	✓
DE_SPTCH	-
DE_DPTCH	✓
DE_ZPTCH	✓
CMD_OPC	0x8
CMD_ROP	✓
CMD_STYLE	✓
CMD_PATRN	✓
CMD_CLP	✓
3D_CNTRL	✓
HITH	✓
YON	✓
FOG_COLOR	✓
FORE	✓
BACK	✓
MASK	✓
DE_KEY	✓
LPAT	✓
PCTRL	✓
CLPTL	✓
CLPBR	✓
ALPHA	✓
A_CNTRL	✓
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

3D Triangle with Full Setup and Vertex Sorting

Command Name: 3D Triangle
Command Mnemonic: TRIAN_3D
Command Description:
The TRIAN_3D command draws a Textured, Gouraud shaded, Specular lighted or flat shaded triangle. This command requires only vertex-level parameters.
Command Capabilities:

- 16 raster operations as defined by the ROP field of the command register
- Perspective corrected texture mapping
- Trilinear MipMap Filtering, Bilinear Filtering
- Palletized texture maps
- Linear MIP mapping
- Specular Lighting, Fog
- Gouraud Shading
- Alpha blending and compare
- 16, 24, fixed point Z buffering
- 32 bit floating point Z buffering
- 32x32 and 8x8 area patterning
- Color expansion
- Screen door transparency in stipple mode
- Bit plane masking
- 3D Color Keying
- Optional Float color values
- Optional Backface culling w/ CW/CCW selection

0x9	**OPCODE**

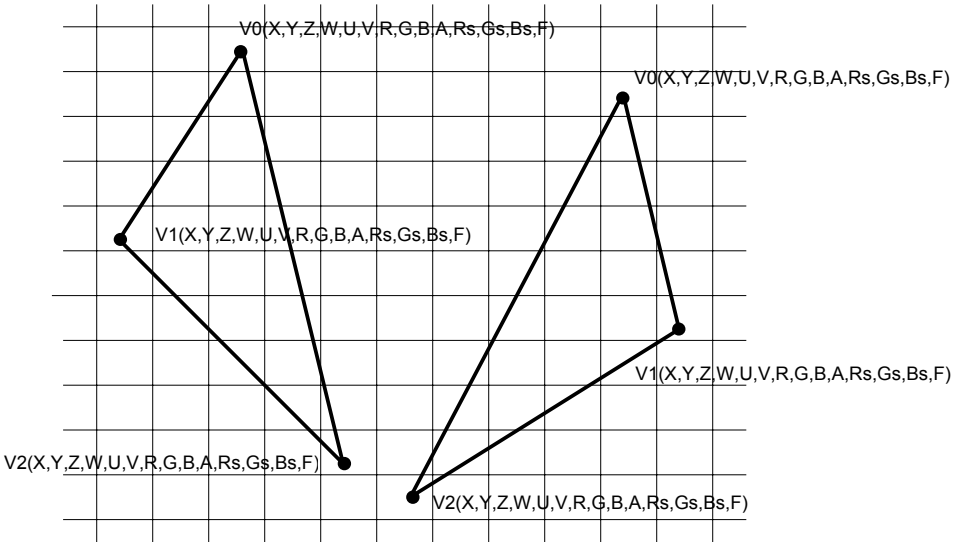


Figure 2-1.1. <<Steve, need figure caption text here.>>

REGISTER	PARAMETER	FORMAT	DESCRIPTION
CP0	PPTR	X-Y	Pattern Pointer
CP1	V0_X	Float	Vertex 0 X
CP2	V0_Y	Float	Vertex 0 Y
CP3	V0_Z	Float	Vertex 0 Z
CP4	V0_W	Float	Vertex 0 W
CP5	V0_C	I	Vertex 0 Color {A, R, G, B}
CP6	V0_S	I	Vertex 0 Specular {F, Rs, Gs, Bs}
CP7	V0_U	Float	Vertex 0 U
CP8	V0_V	Float	Vertex 0 V
CP9	V1_X	Float	Vertex 1 X
CP10	V1_Y	Float	Vertex 1 Y
CP11	V1_Z	Float	Vertex 1 Z
CP12	V1_W	Float	Vertex 1 W
CP13	V1_C	I	Vertex 1 Color {A, R, G, B}
CP14	V1_S	I	Vertex 1 Specular {F, Rs, Gs, Bs}
CP15	V1_U	Float	Vertex 1 U
CP16	V1_V	Float	Vertex 1 V
CP17	V2_X	Float	Vertex 2 X
CP18	V2_Y	Float	Vertex 2 Y
CP19	V2_Z	Float	Vertex 2 Z
CP20	V2_W	Float	Vertex 2 W
CP21	V2_C	I	Vertex 2 Color {A, R, G, B}
CP22	V2_S	I	Vertex 2 Specular {F, Rs, Gs, Bs}
CP23	V2_U	Float	Vertex 2 U
CP24	V2_V	Float	Vertex 2 V
TRIGGER *	3D Trigger	NA	Triggers 3D commands

* Command Trigger

Important Programming Note: The W parameter for each vertex must be set before setting the corresponding u and v parameters.

Table 3-1.6. ACTIVE Parameters

BUF_CTRL	✓
DE_SORG	✓
DE_DORG	✓
DE_ZORG	✓
DE_SPTCH	✓
DE_DPTCH	✓
DE_ZPTCH	✓
CMD_OPC	0x9
CMD_ROP	✓
CMD_STYLE	✓
CMD_PATRN	✓
CMD_CLP	✓
3D_CNTRL	✓
HITH	✓
YON	✓

FOG_COLOR	✓
FORE	✓
BACK	✓
MASK	✓
DE_KEY	✓
LPAT	✓
PCTRL	✓
CLPTL	✓
CLPBR	✓
ALPHA	✓
A_CNTRL	✓
TEX_CNTRL	✓
LOD0_ORG	✓
LOD1_ORG	✓
LOD2_ORG	✓
LOD3_ORG	✓
LOD4_ORG	✓
LOD5_ORG	✓
LOD6_ORG	✓
LOD7_ORG	✓
LOD8_ORG	✓
LOD9_ORG	✓
DE_TPTCH	✓
TPAL_ORG	
TEX_BORDER	✓
KEY_3D_LOW	✓
KEY_3D_HI	✓
GLBLENDC	✓

DE_SORG: Origin for pattern pointer

DE_SPTCH: Pitch for pattern pointer space.

Texture Invalidate

Command Name: Texture Invalidate
Command Mnemonic: INV_TEX
Command Description:
The texture invalidate command is set before switching to a new texture. It instructs the Borealis chip that the next triangle coming through has a different texture than previously used.

0xA	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY1*	Trigger		Trigger command

* Command Trigger

Important Programming Note: To enable Texture Cache Tile Mode, TEX_CTRL[30] must be set before triggering the INV_TEX command.

Table 3-1.7. ACTIVE Parameters

BUF_CTRL	-
DE_SORG	-
DE_DORG	-
DE_ZORG	-
DE_SPTCH	-
DE_DPTCH	-
DE_ZPTCH	-
CMD_OPC	0xA
CMD_ROP	-
CMD_STYLE	-
CMD_PATRN	-
CMD_CLP	-
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	-
BACK	-
MASK	-
DE_KEY	-
LPAT	-
PCTRL	-
CLPTL	-
CLPBR	-
ALPHA	-
A_CNTRL	-
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	-
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

Load Texture Palette

Command Name: Load Texture Palette
Command Mnemonic: LD_TPAL
Command Description:
The load texture palette command loads a palette from the system or frame buffer memory into the Borealis internal texture cache palette. This command must be used to load a palette before using a palletized texture for the first time, or whenever a new texture is required.
Command Capabilities:

- 8 bit palletized, 32 bit exact
- 8 bit palletized, 16 bit filtered
- 16 bit or 32 bit color palettes at 1,2,4 bit per texel
- automatic conversion of 32 bit to either 4444 or 0565 for 8 bit palette filtering

0xB	**OPCODE**

REGISTER	PARAMETER	FORMAT	DESCRIPTION
XY0	NA		
XY1*	NA		Trigger
XY2	NA		
XY3	NA		
XY4	NA		

* Command Trigger

Table 3-1.8. ACTIVE Parameters

BUF_CTRL	-
DE_SORG	-
DE_DORG	-
DE_ZORG	-
DE_SPTCH	-
DE_DPTCH	-
DE_ZPTCH	-
CMD_OPC	0xB
CMD_ROP	-
CMD_STYLE	-
CMD_PATRN	-
CMD_CLP	-
3D_CNTRL	-
HITH	-
YON	-
FOG_COLOR	-
FORE	-
BACK	-
MASK	-
DE_KEY	-
LPAT	-
PCTRL	-
CLPTL	-
CLPBR	-

ALPHA	-
A_CNTRL	-
TEX_CNTRL	-
LOD0_ORG	-
LOD1_ORG	-
LOD2_ORG	-
LOD3_ORG	-
LOD4_ORG	-
LOD5_ORG	-
LOD6_ORG	-
LOD7_ORG	-
LOD8_ORG	-
LOD9_ORG	-
DE_TPTCH	-
TPAL_ORG	✓
TEX_BORDER	-
KEY_3D_LOW	-
KEY_3D_HI	-
GLBLENDC	-

Chapter 7: VGA Specification

Internal VGA

Borealis has an internal VGA core. VGA operation is enabled by configuration jumper. CP[28] is pulled down/up respectively to set the PCI device class to VGA. No VGA operation is enabled if the device class is not set to VGA.

Supported Modes

The core supports all standard VGA modes. The following table shows the modes that are supported in the Borealis:

Table 3-1.1. Standard VGA Mode Table

MODE NO.	SCREEN FORMAT	COLORS	MODE TYPE
00 / 01	320 X 200	16	Text
00* / 01*	320 X 350	16	Text
00+ / 01+	360 X 400	16	Text
02 / 02	640 X 200	16	Text
02* / 03*	640 X 350	16	Text
02+ / 03+	720 X 400	16	Text
04 / 05	320 X 200	4	Graphics
06	640 X 200	2	Graphics
07	720 X 350	2	Text
07+	720 X 400	2	Text
0D	320 X 200	16	Graphics
0E	640 X 200	16	Graphics
0F	640 X 350	2	Graphics
10	640 X 350	16	Graphics
11	640 X 480	2	Graphics
12	640 X 480	16	Graphics
13	320 X 200	256	Graphics

*EGA Compatibility using 14 line font +Enhanced VGA modes using 16 line font

Borealis VGA Architecture

The block diagram below shows how the VGA core subsystem interfaces to the rest of the Borealis Design:

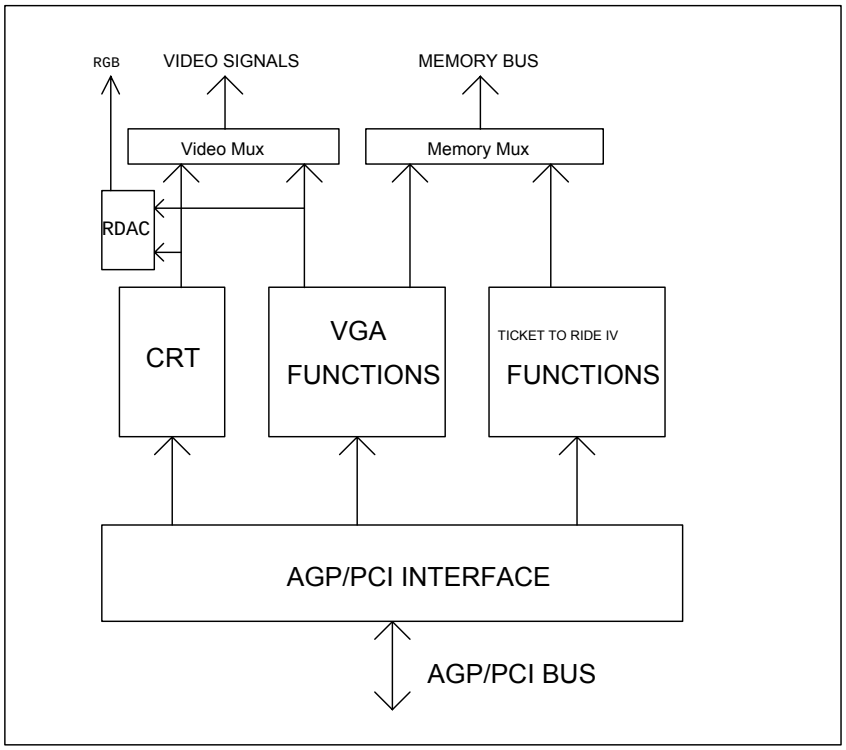


Figure 2-1.1. VGA Block Diagram

Borealis VGA Architecture

The VGA subsystem in Borealis consists of three parts: the VGA core, the VGA host interface, and the VGA memory interface. The VGA host interface accepts decoded PCI cycles from the PCI Host Interface block. The decoded cycles are then presented to the VGA core which processes the cycles. The VGA core will then request memory cycles to the VGA memory interface. The VGA memory interface in turn will generate the actual memory timing. The VGA subsystem runs off of the memory controller clock. The VGA video timing is generated from the CRT clock.

Borealis VGA Operation

The VGA core subsystem in Borealis is controlled by the VGA_CTRL register. (The programmer interface for the VGA_CTRL register is shown at the end of this document.) Before enabling VGA, the chip must be initialized for normal operation: enable the appropriate register block and memory window decode in CONFIG1, set the memory bus control in CONFIG2, and program the appropriate wait states also in CONFIG2. For boards with SGRAM or SDRAM, the SGR_CONFIG register must also be programmed.

On power up, the VGA_CTRL register is reset to zero, indicating that all VGA cycle decode on the PCI Bus is disabled. If the PCI device class was set to VGA (CP[28]=1), the following settings in the VGA_CTRL register will enable VGA operation:

1 Set VGA_EN = 1. This bit enables PCI decode of VGA cycles. If this bit is left in its default state of 0, all VGA decode, both memory and I/O, is disabled.

2 Set VGA_MUX = 1. Setting this bit to 1 enables the VGA generated memory timing signals to be present on the memory bus. Leaving this bit a 0 allows Borealis memory signals to be present on the memory bus. Setting this bit to 1 also enables the VGA generated video timing and data signals to be present on the sync/blank pins and pixel data bus of an external/internal RAMDAC. Leaving this bit a 0 allows Borealis generated video timing to be present on the sync/blank pins of an external/internal RAMDAC.

3 In DDC1 mode, the VSYNC pin is not affected by this bit.

RAMDAC and PROM timing are always generated from Borealis host bus controller, not from the VGA subsystem.

Set MEM_EN = 1. This enables VGA memory decode.

Set VDE = 1. This enables VGA DAC access.

VGA Decode

The VGA subsystem in Borealis has a number of options that determine what I/O and memory space is decoded on the PCI bus. For any VGA decode to be enabled, both the device class must be set to VGA (via configuration jumper) and VGA_EN must be set to 1. If either of these conditions is not true, then no memory or I/O decode will be enabled. Please note that this means that VGA DAC decode (IO x3C6 - x3C9) will also be completely disabled.

Memory decode is based on the VGA mode. The VGA will decode one of the following ranges, based on the VGA register GR6, bits [3:2]:

- 0xA0000 - 0xBFFFF
- 0xA0000 - 0xAFFFF
- 0xB0000 - 0xB7FFF
- 0xB8000 - 0xBFFFF

All VGA memory decode may be disabled by setting the MEM_EN bit in VGA_CTRL to 0. Doing so will not affect the I/O decode. This feature is provided so that linear memory window 1 may be programmed to decode VGA memory space. This can be achieved by programming the MW1 registers through I/O space. Scratch registers are also provided in I/O space so the original MW1 values may be preserved. The I/O mapping of the DDC, VGA_CTRL, MW1, SCRATCH, and DAC registers is attached at the end of this document.

VGA I/O decode is also dependent on the mode. The following I/O locations are decoded for the VGA subsystem. An X indicates “B” for monochrome modes and “D” for color modes:

- 3C0, 3C1, 3C2, 3C4, 3C5, 3CA, 3CC, 3CE, 3CF
- 3XA, 3X4, 3X5

I/O 3C6 - 3C9 are decoded to the normal DAC decode logic and not to the VGA subsystem.

Borealis VGA Control Register

Read /write at address = PCIB4 + 0x30
All registers default to 0 on reset

7	6	5	4	3	2	1	0
vde	win_rst	stretch	reserved	mem_en	reserved	vga_en	vga_mux

vde

This bit controls whether VGA DAC accesses are decoded. If VGA DAC accesses are enabled, then the cycles will be "owned" or "snooped' based on the PCI DAC snoop bit in the PCI command register. All DAC accesses are routed to the internal/external RAMDAC.

vde = 1 VGA DAC cycles are decoded
vde = 0 VGA DAC cycles are ignored (default)

win_rst

This bit is don't care with SGRAM/SDRAM memories and normally this bit is set to 0

Stretch/Horizontal Zoom

In SGRAM/SDRAM mode with digital RGB outputs enabled: 0-horizontal zoom is off, 1-horizontal zoom x2 is on.
This bit affects display in both VGA and high resolution modes if digital RGB outputs are enabled. If digital RGB support is off, this bit is a "don't care" in SGRAM/SDRAM modes.

mem_en

This bit enables overall VGA decode of the A000 and B000 segments of memory. If enabled, the appropriate portion of A000 or B000 segments will be decoded, depending on the setting of VGA register GR6[3:2]. If disabled, the VGA subsystem will not decode any memory space, although I/O access will still be enabled.

mem_en = 1 VGA Memory decode enabled
mem_en = 0 VGA Memory decode disabled (default)

vga_en

This is the VGA master decode enable. If this bit is 0, all VGA decode on the host bus (PCI or AGP/PCI). Bus is disabled.

vga_en = 1	Enable VGA Decode	
vga_en = 0	Disable VGA Decode	(default)

vga_mux

This bit controls whether the VGA or Borealis high resolution subsystem has access to the frame buffer and the RAMDAC (syncs and video data).

vga_mux = 1	VGA subsystem has access to the frame buffer and DAC
vga_mux = 0	Imagine has Frame buffer/DAC access (default)

NOTE: See Chapter 2, *Functional Description,* for a listing of I/O mapped configuration registers.

Chapter 8: *RAMDAC Device*

Palette DAC Description

The internal palette DAC brings high speed and advanced features for high-resolution true-color. The translucent cursor clears the way to improved graphical user interface functionality. Pixel re-synchronization circuitry eliminates the susceptibility to control signal timing variations. It provides vibrant color with unsurpassed image stability and bright, flicker-free display on large-screen, high-resolution monitors.

Palette DAC Highlights

The following lists the palette DAC highlights:

- 250 MHz operation
- 64-bit wide pixel data bus
- Fine-grained PLL programming optimizes display
- Pixel re-synchronization ensures integrity of all display modes
- Large Screen ISO-compliant refresh rates
- 8/16/32-bits per pixel
- Direct color
- Gamma correction
- 256-shade gray scale
- Three 256x8 color palette RAMs
- Triple monotonic 8-bit DACs
- 64x64/32x32 translucent hardware cursor
- 100 MHz 8-bit VGA data input
- On-chip diagnostic functions
- Power-down modes

Microprocessor Access

Two indirect schemes are used to access all of the internal registers and arrays through eight PCI addresses. The first scheme is standard VGA, which maps to PCI addresses 0x3C6, 0x3C7, 0x3C8, and 0x3C9. Address 0x3C6 maps to the Pixel mask register, 0x3C7 maps to the Palette read address register, 0x3C8 maps to the Palette write address register, and 03C9 maps to the Palette data register. The second scheme is an indexed scheme and is used to access all of the remaining registers including the cursor array. This scheme operates when AD[4] =1. Of the four I/O addresses then available using AD[3:2], two are used to load an index register (Low and High). The third address is used to write or read the register or array position pointed to by the index register. The fourth address is used to directly access a register which controls whether the index register automatically increments following an indexed register access.

The eight I/O addresses selected by AD[4:2] are listed in the following table:

Table 3-1.1. I/O Addresses

AD[4:2]	REGISTER
000	Palette Address (Write Mode)
001	Palette Data
010	Pixel Mask
011	Palette Address (Read Mode)
100	Index Low
101	Index High
110	Index Data (Indexed Registers)
111	Index Control

Table 3-1.2. Index Registers

AD[4:2]	INDEX	R/W	RESET VALUE	REGISTER NAME
000	-	√	U	Palette Address (Write Mode)
001	-	√	U	Palette Data
010	-	√	U	Pixel Mask
011	-	√	U	Palette Address (Read Mode)
100	-	√	U	Index Low
101	-	√	U	Index High
110	-	√	U	Index Data (Indexed Registers)
111	-	√	U	Index Control
110	0x0002	√	0x01	Miscellaneous Clock Control
110	0x0003	√	0x00	Sync Control
110	0x0004	√	0x00	Horizontal Sync Position
110	0x0005	√	0x00	Power Management
110	0x0006	√	0x02	DAC Operation
110	0x0007	√	0x00	Palette Control
110	0x0008	√	0x01	System Clock Control
110	0x0009	-	-	(Reserved)
110	0x000a	√	U	Pixel Format
110	0x000b	√	U	8 BPP Control
110	0x000c	√	U	16 BPP Control
110	0x000e	√	U	32 BPP Control
110	0x000f	-	-	(Reserved)
110	0x0010	√	0x00	Pixel PLL Control 1
110	0x0011	√	0x00	Pixel PLL Control 2
110	0x0012 0x0013	-	-	(Reserved)
110	0x0015	√	0x08	SYSCLK N (System PLL Reference Divider)
110	0x0016	√	0x41	SYSCLK M (System PLL VCO Divider)
110	0x0017	√	U	SYSCLK P
110	0x0018	-	-	(Reserved)
110	0x0019 0x001f	-	-	(Reserved)

AD[4:2]	INDEX	R/W	RESET VALUE	REGISTER NAME
110	0x0020	√	0x005	Pixel M0 (M0)
110	0x0021	√	0x0e	Pixel N0 (N0)
110	0x0022	√	0x00	Pixel P0 (M1)
110	0x0023	-	-	(Reserved)
110	0x0024	√	0x00	Pixel M1 (M2)
110	0x0025	√	0x00	Pixel N1 (N2)
110	0x0026	√	0x00	Pixel P1 (M3)
110	0x0027	-	-	(Reserved)
110	0x0030	√	0x00	Cursor Control
110	0x0031	√	U	Cursor X Low
110	0x0032	√	U	Cursor X High
110	0x0033	√	U	Cursor Y Low
110	0x0034	√	U	Cursor Y High
110	0x0035	√	U	Cursor Hot Spot X
110	0x0036	√	U	Cursor Hot Spot Y
110	0x0037	√	0x00	Advanced Cursor Control
110	0x0038	√	U	Advanced Cursor Attribute
110	0x0039 0x003f	-	-	(Reserved)
110	0x0040	√	U	Cursor Color 1 Red
110	0x0041	√	U	Cursor Color 1 Green
110	0x0042	√	U	Cursor Color 1 Blue
110	0x0043	√	U	Cursor Color 2 Red
110	0x0044	√	U	Cursor Color 2 Green
110	0x0045	√	U	Cursor Color 2 Blue
110	0x0046	√	U	Cursor Color 3 Red
110	0x0047	√	U	Cursor Color 3 Green
110	0x0048	√	U	Cursor Color 3 Blue
110	0x0049 0x005f	-	-	(Reserved)
110	0x0063 0x0067	-	-	(Reserved)
110	0x0069 0x006b	-	-	(Reserved)
110	0x006d- 0x006f	-	-	(Reserved)
110	0x0070	√	U	Miscellaneous Control 1
110	0x0071	√	U	Miscellaneous Control 2
110	0x0073- 0x0077	-	-	(Reserved)
110	0x0079- 0x0081	-	-	(Reserved)
110	0x0082	RO	U	DAC Sense
110	0x0083	RO	0x00	SR Status
110	0x0084	RO	U	SR Red
110	0x0085	-	-	(Reserved)
110	0x0086	RO	U	SR Green

AD[4:2]	INDEX	R/W	RESET VALUE	REGISTER NAME
110	0x0087	-	-	(Reserved)
110	0x0088	RO	U	SR Blue
110	0x0089-0x008b	-	-	(Reserved)
110	0x008c	RO	U	Pixel P Input
110	0x008d	-	-	(Reserved)
110	0x008e	RO	0x05 (FS[1:0]=00) 0x0e (FS[1:0]=01) 0x00 (FS[1:0]=10 or 11)	Pixel M Input (Pixel PLL VCO Divider Input)
110	0x008f	RO	0x05	Pixel N Input (Pixel PLL Reference Divider Input)
110	0x00P0-0x00P3	RO	U	VRAM Mask Low
110	0x00C0-0x00C3	RO	U	VRAM Mask High
110	0x0092-0x00ff	-	-	(Reserved)
110	0x0100-0x04ff	√	U	Cursor Array
110	0x0500 0x07ff	-	-	(Reserved)

VGA Access

Palette

Internally the three 256x8 palettes are accessed by the microprocessor as a single 256x24 palette, with all 24 bits written or read in one operation.

A single Palette Address register points to 1 of 256 locations for writing or reading the 24 bits. Two different Register Select addresses are used to access the Palette Address register.

A write to AD[4:2] = 000 (Palette Address Write Mode) initializes the palette logic for write operations. Subsequent writes to Palette Data (AD[4:2] = 001) will load internal palette color registers and cause these register contents to be written into the palettes.

A write to AD[4:2] = 011 (Palette Address Read Mode) initializes the palette logic for read operations. Data from the palettes will be loaded into internal palette color registers. Subsequent reads from Palette Data (AD[4:2] = 001) will read these palette color registers.

Every three accesses of Palette Data (AD[4:2] = 001) will cause the Palette Address register to be incremented. An increment past 0xff will “wrap around” to 0x00.

A read from either Palette Address (Write Mode) or Palette Address (Read Mode) will read the Palette Address register. The same register is used for writing and reading, thus, changing modes destroys the contents of the previous mode’s palette address. For example, if some reads are performed and then Palette Address (Write mode) is written, the read address will be lost and a read of either Palette Address (Write Mode) or Palette Address (Read Mode) will produce the same result: the address that was written into Palette Address (Write Mode).

Palette Write

Palette writes must be initialized by writing the Palette Address (Write Mode) register. This provides a starting address for writes and initializes the internal circuitry for palette write operations.
Palette writes are then performed by writing to Palette Data in a red, green, blue... sequence. These writes will load internal palette data registers in sequence. Immediately following every third write, an internal write will be triggered to the palette of the 24 bits contained in the internal palette data registers, at the address contained in the Palette Address register.
Immediately following the internal palette write triggered by the third write to Palette Data, the Palette Address register will be incremented. Thus, continuous writes to Palette Data will load the palette, stepping through the palette addresses in ascending order.

Palette Read

Palette reads must be initialized by writing the Palette Address (Read Mode) register. This provides a starting address for reads and initializes the internal circuitry for palette read operations.
Immediately following the writing of Palette Address (Read Mode), a read of the palette will be performed at the address just written. Internal palette data registers are loaded with the read data, and the Palette Address register is incremented.
Palette reads are then performed by reading from Palette Data. Red, green, blue... data from the preloaded internal registers will be presented in sequence. Immediately following every third read, an internal read of the palette to the 24 bits contained in the internal registers will be performed at the address contained in the Palette Address register.
Immediately following the internal palette read triggered by the third read of Palette Data, the Palette Address register will be incremented. Thus, continuous reads of Palette Data will read the palette, stepping through the palette addresses in ascending order.

6/8 Bit Palette Access

The original VGA had 6-bit DACs and 6-bit palette entries, and the low order 6 bits from/to the microprocessor port were written/read into the palette.
The DACs and palette entries are 8 bits. For non-VGA emulation all 8 bits are used. To emulate 6-bit VGA operation the upper 6 bits of the palette hold the VGA 6-bit color and the two low order bits are set to 00. The COL RES bit (color resolution) of the Miscellaneous Control 2 register determines if the access is 6-bit or 8-bit.
The reset condition is to emulate VGA using the 6 low order microprocessor data bits. COL RES is set to 6 bits. In this mode, for writing, microprocessor bits [7:6] are discarded, bits [5:0] are shifted to bits [7:2], and bits [1:0] are set to 00 before being written into the internal data registers. For reading, the internal data register bits [7:2] are shifted to bits [5:0], and bits [7:6] are set to 00 before being presented on the microprocessor data signals.
If COL RES is set to 8 bits then all 8 bits from/to the microprocessor will be written to and read from the color palette registers.
Note that the 6-to-8 bit translation is only done between the microprocessor port and the internal data registers. Internally, on writes, all 8 bits of the internal registers are written to the palette, and on reads, the internal registers retain all 8 bits read from the palette. Thus, if the palette is loaded with 8-bit values with COL RES set to 8 bits, and then the palette is read with COL RES set to 6 bits, the internal palette color registers will still be loaded with the 8 bits that were written into the palette. But the data read on the microprocessor data lines will be 6 bits.

Palette Clocking

Palette accesses are synchronized internally with the pixel clock. On writes, the pixel values of the previous cycle are held and displayed during the write cycle. Both of these features minimize disturbance of displayed pixels when the palette is accessed.
The pixel clock (as selected by the PCLK SEL bits in Miscellaneous Control 2) must be running for palette access to be valid.

The timings for the microprocessor signals are specified in units of pixel clocks. These specifications are derived from the requirement for the pixel clock to be running for palette access, as well as to allow time for the Palette Accesses and Palette Address increments to occur internally following a palette access.

Palette Access Status

The original VGA logic had an override for read accesses of the Palette Address (Read Mode) register. Instead of reading the Palette Address register, a value was returned that indicates the status of the last palette access, write or read.
The reset condition of the palette DAC is to return the address value for a read of Palette Address (Read Mode). The VGA logic may be emulated by setting the RADR RFMT bit in Miscellaneous Control 1. This causes the status of the last palette access to be returned.
The value of the status returned is 0x00 if the last write to Palette Address was Write Mode, and 0x03 if the last write to Palette Address was Read Mode.

Pixel Mask

The pixel mask is an 8-bit register addressed with AD[4:2] = 010. It can be accessed at any time without disturbing a palette write or read sequence.
Accesses to the pixel mask are asynchronous to the pixel clock. Temporary color disturbances can be expected if the mask is changed while displaying pixels through the palette.

Indexed Access

The cursor array and a number of control registers are addressed with an internal 11-bit index register. The microprocessor accesses this as Index High (AD[4:2] = 101) and Index Low (AD[4:2] = 100).
A write or read to Index Data (AD[4:2] = 110) actually writes or reads the register/cursor array location addressed by the Index register.
Following a write or read of Index Data, the index register will increment if the INDX CNTL bit is set. The Index Control register (AD[4:2] = 111) contains this bit. To allow for future expansion, wraparound from 0x07ff to 0x0000 is not supported.
In general, access of Index Low, Index High, Index Control, or any of the Indexed registers is independent of the palette access and will not disturb a palette write or read sequence. However, as described above the PADR RFMT bit in Miscellaneous Control 1, the COL RES bit in Miscellaneous Control 2, and the 6BIT ACC bit in Palette Control all affect palette access.
Also, as described above, the pixel clock must be running for valid access of the palette, and the pixel clock is affected by a number of indexed registers.

Cursor Array

In general, the indexed registers may be written or read at any time, using the address held in Index High and Index Low. This address may be set by writing to Index High or Index Low, or the value may result from the auto-increment action of a previous access.
However, as described in section, *Cursor Array Reads*, later in this appendix, to access the cursor array a write to Index High or Index Low must be performed first. That is, the cursor array cannot be accessed by auto-increment from address 0x00ff to 0x0100.
Also, as with the palette, the pixel clock must be running to access the cursor array.

Clocking

Clock Generators

There are two on-board clock generators: pixel clock and system clock (SYSCLK). Each clock generator uses a separate programmable phase locked loop (PLL).
The pixel clock generator provides the fundamental “dot” timings; it serves generally as the clock both for internal chip clocking and for on-card CRT timings.

The system clock generator is used to provide all timing for the graphics memory subsystem.

PLL Input

REFCLK

The REFCLK input is a reference clock that the PLLs use in conjunction with programming registers to produce a wide variety of frequencies.
In general, REFCLK can be any frequency from 1 MHz through 100 MHz. However, as discussed below, the two clock generators each come up at "start up" frequencies which are dependent on the REFCLK frequency. This can govern the value chosen for REFCLK, depending on the application requirements at "reset" time.
Following a reset, the PLL driving the SYSCLK output is enabled with the start-up frequency:

SYSCLK frequency = (33/16) x REFCLK frequency

The pixel clock PLL has two start-up frequencies:

F0 frequency = (7/4) x REFCLK frequency

F1 frequency = (79/40) x REFCLK frequency

The desired frequency is selected using the FS input from the VGA core .
The start-up values are chosen for use with REFCLK = 14.31818 MHz (a common graphics adapter frequency):

SYSCLK frequency = (33/16) x 14.31818 = 29.53 MHz
F0 frequency = (7/4) x 14.31818 = 25.057 MHz
F1 frequency = (79/40) x 14.31818 = 28.278 MHz

This causes the SYSCLK start up frequency to be approximately 30 MHz and the pixel clock frequencies to be approximately the standard VGA frequencies 25.175 MHz and 28.322 MHz.
With a 14.31818 MHz REFCLK the start-up pixel clock frequencies selected by FS are:

Table 3-1.3. Start-up Pixel Clock Frequency

FS	FREQUENCY SELECTED	VALUE
0	F0	25.057 MHz
1	F1	28.278 MHz

Pixel PLL Outputs

The pixel PLL is used internally as the pixel clock. The maximum allowed generated frequency is 250 MHz.
The pixel PLL Output is not available directly. However, a divided versions is provided on output signal:

Ï LD_CLK

Table 3-1. Table 8-4. LD_CLK Frequencies

BPP	LD_CLK Calculations
8	÷8
15/16	÷4
32	÷2
VGA	÷1

Additional Clocks

Pixel Clock (Dot Clock)

The pixel clock, or dot clock, is the internal clock used to clock pixel data up through the DACs. It is also required to be running to access the palette and the cursor array. The maximum frequency of this clock is 250 MHz.

There are several sources of the pixel clock, as selected by the PCLK SEL bits in the Miscellaneous Control 2 register:

LCLK input This is the reset default. It is intended to be used when the VGA port is selected as the pixel source.

Pixel PLL output This is intended to be used when the VRAM pixel port is selected as the pixel source. It provides the highest pixel clock operation.

REFCLK input This is intended for laboratory bring up.

When LCLK is selected as the pixel clock all internal pixel operations are synchronous with LCLK. If the pixel clock is sourced by the pixel PLL output or REF-CLK, then the incoming pixels and video controls are expected to be derived from SCLK. After latching the signals with LCLK, the signals are clocked with an internal SCLK, and then clocked with the internal pixel clock.

PLL Operation and Programming

The two PLLs are generally identical in their operation and programming. A simplified diagram of the PLL is shown in Figure 8-1. The PLL takes the incoming reference clock, REFCLK, and generates the CLOCKOUT output. The frequency of CLOCKOUT is determined by three programming values contained in registers, M, N and P.

The heart of the PLL is the VCO (voltage controlled oscillator). The VCO can operate over the range of 65 MHz to 250 MHz.

The VCO voltage input value is produced by comparing a divided version of the VCO output with a reference frequency. The value M sets the divide value for the VCO output. Values for M can be 2 through 127 (0 and 1 are illegal). '1' is added to M, to produce a divide value of 3 through 128.

The internal reference frequency (f_{INTREF}) is produced by dividing the incoming REFCLK, with the divide value set by N. N can range from 0 through 63, and '1' is added to this value to produce a divider value of 1 through 64.

The divided VCO output frequency is compared to the internal reference f_{INTREF} by a phase comparator. The phase comparator drives a charge pump which drives a filter connected to the VCO. A capacitor in the filter develops the voltage supplied to the VCO. The voltage goes up or down depending on whether the phase comparator is driving the charge pump current up or down.

When the divided VCO frequency is equal to f_{INTREF} no pump current is produced, the voltage to the VCO stays constant, and the VCO frequency stays constant. If the VCO tends to drift in frequency the phase comparator will detect the difference and will drive filter voltage, via the charge pump, in the appropriate direction. This adjusts the VCO frequency such that the frequency difference at the input to the phase comparator again becomes zero. With a constant f_{INTREF} the VCO frequency will be "locked" to the internal reference.

The internal reference frequency is:

$$(1)\quad f_{INTREF} = \frac{f_{REFCLK}}{N + 1}$$

The VCO frequency is:

$$(2)\quad f_{VCO} = f_{INTREF} \times (M + 1) = f_{REFCLK} \times \frac{M + 1}{N + 1}$$

The output of the VCO is divided down by 1, 2, 4, 6 or 8, depending on the P programming value, to produce the final programmed frequency. For example, a clock frequency of 28 MHz is produced by programming the VCO to oscillate at 4 x 28 = 112 MHz, and then setting P to divide the VCO output by 4.

Values for P are 0 through 4 for the Pixel PLL and 1 through 4 for the SYSCLK PLL. When P is 0 the divide value is 1; when P = 1, 2, 3 or 4 the divide factor is 2 x P.
The generated CLOCK OUT frequency is:

(3a) $f_{CLOCKOUT} = f_{REFCLK} \times \frac{M + 1}{N + 1}$; P = 0

(3b) $f_{CLOCKOUT} = f_{REFCLK} \times \frac{M + 1}{(N + 1) \times 2P}$; P = 1, 2, 3, 4

Additional Constraints

Internal Reference

To avoid excessive jitter the internal reference frequency f_{INTREF} must not be less than 1 MHz.

VCO Frequency Range

The minimum VCO frequency is 65 MHz. The maximum frequency is the maximum speed rating of the product: 250 MHz. (E.g., if it is desired to have a Dot Clock of 150 MHz it is not permitted to program the PLL to run at 300 MHz with a P value of 1 to divide down to 150 MHz.)

Programming Summary

1. Select M, N and P values that produce the desired frequency using equation (3a) or (3b). Note that for the SYSCLK PLL a value of '0' for P is not permitted, so equation (3a) is not valid for this PLL.
2. Verify that f_{INTREF} is not less than 1 MHz, using equation (1).
3. Verify that f_{VCO} is not less than 65 MHz and not greater than 250 MHz, using equation (2).
4. If all of the above conditions are met then M, N and P are valid values.

Glitching on Frequency Change

When the operating frequency of either PLL is changed by changing one of the programming register values, the transition from the original frequency to the new frequency can either occur smoothly or can glitch, depending upon the following:

1. If the P bits are not changed, then changing the M and N bits will not cause a glitch.
2. ***If the P value is changed then the PLL output can glitch. For the Pixel PLL there is no protection against this situation.***

   For the SYSCLK PLL there is additional logic (not shown) which causes the "2,4,6,8" VCO divider circuit to update synchronously with the VCO output when the P value is changed, such that changing P will not cause SYSCLK to glitch.

The Pixel PLL has selectable banks of programming values (described below). Note that switching between banks has the same glitching considerations as changing single M, Nor P values.

MN Programming Model vs. MNP Programming Model

PLL Compatibility Programming

The palette DAC provides a new method of programming the PLLs. This new method is less restrictive and is preferred for new applications.
But for software compatibility with the other palette DACs the palette DAC additionally provides the same, "old" method of PLL programming. When reset the Palette DAC's PLL control registers are set for "old" programming, and the programming registers contain power on values appropriate for "old" programming.

PLL Programming

The two PLLs are programmed identically. Three values are used:
REF DIV COUNT (Reference Divide Count) This number provides a count value for dividing down the incoming REFCLK. It must be between 2 and 31. Operation of the PLL is indeterminate if this number is 0 or 1.
VCO DIV COUNT (VCO Divide Count) This number provides a count value for the divider in the PLL feedback loop. The value can range from 0 through 63. Internally, 65 is added to VCO DIV COUNT, so that the PLL feedback divider value ranges from 65 through 128.
DF (Desired Frequency) These are two bits with values of 00, 01, 10, and 11. The intent of these bits is to divide the operation of the PLL into four frequency ranges. Following the divide of the REFCLK provided

by the REF DIV COUNT there is an additional divide-by-two which is selected or bypassed with the DF bits. Also, the output of the PLL has a divider stage, or postscaler, that is controlled by the DF bits.

Table 8-4, "PLL Equations," gives the general formulas for programming the PLLs. Because of the action of the DF bits there are four equations, one for each DF bit setting.

Table 3-1.4. PLL Equations

DF	Output Frequency	Internal VRF	Max Output Freq. (MHz)
00	$\frac{\text{FREF} \times (\text{VCO DIV COUNT} + 65)}{(\text{REF DIV COUNT}) \times 8}$	$\frac{\text{FREF}}{(\text{REF DIV COUNT}) \times 2}$	62.5
01	$\frac{\text{FREF} \times (\text{VCO DIV COUNT} + 65)}{(\text{REF DIV COUNT}) \times 4}$	$\frac{\text{FREF}}{(\text{REF DIV COUNT}) \times 2}$	125 (*)
10	$\frac{\text{FREF} \times (\text{VCO DIV COUNT} + 65)}{(\text{REF DIV COUNT}) \times 2}$	$\frac{\text{FREF}}{(\text{REF DIV COUNT}) \times 2}$	250 (*)
11	$\frac{\text{FREF} \times (\text{VCO DIV COUNT} + 65)}{\text{REF DIV COUNT}}$	$\frac{\text{FREF}}{\text{REF DIV COUNT}}$	250 (*)

1. FREF = REFCLK frequency
2. Frequencies marked with (*) are maximum pixel PLL frequencies. The SYSCLK PLL maximum output frequency is 100 MHz.

It is possible to program the PLLs with values that generate illegal operating conditions:

1. The reference frequency VRF (Video Reference Frequency), which is internal to the PLL, cannot be less than 1 MHz.
2. The internal VCO (Voltage Controlled Oscillator) cannot exceed the rated speed of the product (250 MHz).
3. The SYSCLK output driven by the SYSCLK PLL cannot exceed 150 MHz.

Table 8-4, "PLL Equations," gives the equations for calculating the internal VRF. Table 8-4 also gives the maximum allowable output frequency for each setting of DF. This reflects the action of the VCO postscaler. If the PLLs are programmed so that these maximum dot clock frequencies are not exceeded then the maximum VCO frequency will not be exceeded.

PLL Frequency Selection

The REF DIV COUNT, VCO DIV COUNT, and DF bits are provided to the PLLs in a pair of 8-bit registers. REF DIV COUNT is 5 bits and occupies one register, with the 3 high order bits unused. The 6 VCO DIV COUNT bits occupy the second register, with the 2 high order bits used by DF.

For the SYSCLK PLL, the two programming registers are the System PLL Reference Frequency, which holds REF DIV COUNT, and the System PLL VCO Divider, which holds VCO DIV COUNT and the DF bits.

For the pixel PLL, there are actually 8 registers which can be used to hold the programming values: Two or three registers are selected from this group to provide the pixel PLL programming values. This selection is controlled by the pixel PLL Control 1 and pixel PLL Control 2 registers.

M over N In this scheme both register values are changed to program a new frequency. The name refers to the general PLL concept in which

Output frequency = Input reference x (M/N)

where VCO DIV COUNT serves as M and REF DIV COUNT serves as N, with modifications to the equation as shown in Table 8-9, "PLL Equations."

For the SYSCLK PLL there is not much distinction between the two programming styles. Both registers are written to provide an initial operating frequency. Then to change frequency either one register is changed (System PLL VCO Divider), or both registers are changed, depending on the programming style.

With the M/N style programming 4 pairs of M and N values are availible (M0,N0,M1,N1M2,N2,M3,N3). This allows 4 preprogrammed pixel clock frequencies.

The selection of the programming registers, either 1 of the 4 M/N pairs, or 1 of 2 M/N/P triplets is done either from the VGA core FS signal , or internally with the INT FS1:0] bits of Pixel PLL Control 2 register. One of two frequencies can be select via the VGA FS line, while one of four frequencies can be selected from PLL control register 2 in M/N mode.
The programming style and selection source is chosen with the EXT/INT bits of the Pixel PLL Control 1 register.

M/N Programming

For the "M over N" programming style use Table 8-4, "PLL Equations," in the following steps:

1. Select values for REF DIV COUNT, VCO DIV COUNT, and DF that generate the desired frequency (or come close enough). Note that the values 0 and 1 are illegal for REF DIV COUNT under all conditions.
2. Calculate the internal reference frequency VRF. Verify that this frequency is not less than 1 MHz.
3. Verify that the dot clock frequency does not exceed the maximum value specified in the table.
4. If conditions 2 and 3 are not met then the selected values cannot be used.

Start-up Values

As mentioned previously when the palette DAC is reset the PLLs are initialized for "old" style programming.
The System PLL VCO Divider (index 0x0016) is reset to 0x41. This yields a DF value of '01', and a VCO DIV COUNT value of decimal 1.
The System PLL Reference Divider (index 0x0015) is reset to 0x08. This gives a REF DIV COUNT of decimal 8.
Using the equation in Table 8-4, "PLL Equations," for DF = 01, the SYSCLK frequency will be REFCLK times:

(1 + 65)/(8 x 4) = 66/32 = 33/16

The F0 and F1 (indices 0x0020 and 0x0021) hold the VCO Divider values for the Pixel PLL. F0 is initialized to 0x05, for a DF value of 00 and a VCO DIV COUNT of decimal 5. F1 is initialized to 0x0e, for a DF value of 00 and a VCO DIV COUNT of decimal 14.
The Fixed Pixel PLL Reference Divider (index 0x0014) is initialized to 0x05, for a REF DIV COUNT of decimal 5.9
Using the equation in Table 8-4 for DF = 00, the F0 frequency will be REFCLK times:

(5 + 65)/(5 x´ 8) = 70/40 = 7/4

and the F1 frequency will be REFCLK times:

(14 + 65)/(5 x 8) = 79/40

Programming Registers

SYSCLK PLL

When the PROG MODE bit of the System Clock Control register (index 0x0008) is set to '1' the SYSCLK N, M, and P registers (indices 0x0015, 0x0016, 0x0017 respectively) are used to provide the SYSCLK PLL programming values.
When the PROG MODE bit is set to '0' the registers at 0x0015 and 0x0016 are used for compatibility mode and the register at 0x0017 is ignored.

Pixel PLL

For the Pixel PLL two banks of registers, each containing a set of M, N, and P values are provided. The selection source of these banks, external or internal, is controlled by the EXT/INT bits of the PLL Control 1 register (index 0x0010).
When EXT/INT = '100' the selection is external, with the values on the VGA core FS signal is to select the register bank.

When EXT/INT = '101' the selection is internal, with the value of the Pixel PLL Control 2 register used to select the register bank.

Table 3-1.5. Pixel PLL Bank Selection

FS OR PLL CONTROL 2 [0]	PROGRAMMING REGISTERS	REGISTER INDICES
0	M0,N0,P0	0x0020 - 0x0023
1	M1,N1,P1	0x0024 - 0x0027

When EXT/INT = '001' or '011' the registers at 0x0020 through 0x0027 are used for compatibility mode programming as described in the section "PLL Compatibility Programming" in this chapter. A register at 0x0014, the Fixed PLL Reference Divider, is used for compatibility mode programming but is ignored when EXT/INT = '100' or '101'.

Diagnostic Readback

The read-only registers Pixel M Input (index 0x008e), Pixel N Input (index 0x008f), Pixel P Input (index 0x008c) contain the programming values actually used by the pixel PLL. These registers can be used to verify that the desired programming registers are the ones actually selected.
When compatibility mode programming is used the registers at 0x008e and 0x008f contain the Pixel PLL programming values as described in the section "PLL Compatibility Programming" later in this appendix, and the values in the registers at 0x008c and 0x008d are undefined.

PLL Disable

Following a reset the PLLs are "enabled" and will change frequencies with changes to the programming values. Each PLL can be separately "disabled". In the disabled state the PLL does not respond to the M and N programming values. This causes the internal VCO to oscillate somewhere in the range of 20 KHz to 1 MHz.
The VCO output continues to be affected by the P value, so the actual PLL output will be (20 KHz through 1 MHz) divided by 1, 2, 4, 6 or 8, for the Pixel PLL, and (20 KHz through 1 MHz) divided by 2, 4, 6 or 8, for the SYSCLK PLL.
The Pixel PLL is disabled by setting the PPLL ENAB in the Miscellaneous Clock Control register to 0. The SYSCLK PLL is disabled by setting the SPPL ENAB bit in the System Clock Control register to 0.

Modes of Operation

Pixel data can come from the VGA port or the VRAM pixel port, as selected by the PORT SEL bit of the Miscellaneous Control 2 register.
If the VRAM pixel port is selected, the pixel format can be 8 BPP (bits per pixel), 15/16 BPP, or 32 BPP, selected by the Format bits of the Pixel Format register.
VGA data is always used to indirectly generate 24 bits of color by indexing into the 256 entry palettes. The Pixel Mask register is used to selectively mask off the index bits as desired.
8 BPP, 15/16 BPP, and 32 BPP from the VRAM pixel port can either be indirect (through the palettes) or direct (bypassing the palettes).
Each of these formats has an associated control register with bits to select indirect or direct color.
As with VGA, the Pixel Mask is used to mask off palette address bits with indirect color access for 8, 15/16, and 32 BPP.

VGA Port

VGA uses 8 bits per pixel. When the VGA port is selected only indirect mode is used. The 8 bits are masked with the Pixel Mask register and presented to the red, green, and blue palettes as indices into the 256 entries of each palette. The masked data is used as the same index into each of the three color palettes.

VRAM Pixel Formats

Bit Ordering

Bit order is high-to-low. For 8 BPP, the MSB is '7' and the LSB is '0'; for 16 BPP the MSB is '15' and the LSB is '0', and so on.
When the VRAM pixel port is selected the default condition is to access the pixels from low to high. For each LCLK, the first pixel used is at the end with bit PIX[00], and the last pixel used is at the end with bit PIX[63]. For example, for 8 BPP, the first pixel is PIX[07:00], the second pixel is PIX[15:08], and so on.

Pixel Format Tables

Table 8-5 shows the bit assignments of the pixel data port for each supported pixel format. Prefixes A - H identify individual pixels, and numbers 0 - 7 identify the bit within the pixel. For 8 bit pixels, it is the data seen by the three color palettes in indirect color mode, and it is the data seen by the three DACs in direct color mode. The suffixes (blu, grn, red) identify the data seen by each of the color palettes (indirect mode) or each of the DACs (direct mode) for 16, and32 bit RGB pixels.

8 BPP

With 8 BPP format 8 pixels are obtained for each pixel port data access.
8 BPP can be indirect or direct, under control of the B8 DCOL bit of the 8 BPP Control register. If indirect, the 8 bits are masked with the Pixel Mask register and presented to the red, green, and blue palettes as indices into the 256 entries of each palette.
If direct, the 8 bits are presented to the red, green, and blue DACs. Note that since the red, green, and blue colors are identical the displayed image will be monochrome.

16 BPP

With 15 BPP or 16 BPP format 4 pixels are obtained for each pixel port data access. The 15 or 16 bits are expanded to 24 bits, under control of the 16 BPP Control register.
The 16 BPP Control register provides a number of options for using the 16 BPP format:

1. The incoming pixel can be 15 bits (555 format) or 16 bits (565 format).
2. The color path can be indirect (through the palettes) or direct (bypassing the palettes).
3. If direct color is used the pixel bits are sent to the DAC high order bits. The low order bits can be zero filled, or the low order bits can be filled with the high order bits of the pixel data. (See description of ZIB/LIN bit below.)

555/565 Formats

The 555/565 bit determines if the pixel is 15 bits (5:5:5 format) or 16 bits (5:6:5 format). The format designator, 5:5:5 or 5:6:5, refer to the bit allocations, high-to-low, for red:green:blue.
With 15 BPP the high order bit of each two bytes (PIX[15], PIX[31], PIX[47], PIX[63]) is discarded.

Color Path Selection

The expansion to 24 bits varies depending on whether the color path is indirect or direct.
Indirect Color: The palette addressing is contiguous. For 555 format there is 1 partition with 32 entries. For 565 format there is 1 partition. All 64 entries in the green palette are addressed. Only the lower 32 entries of the red and blue palettes are used; the high 32 entries are not used.
Direct Color: To expand the 5 or 6 bits of color from the pixel data to 8 bits, the ZIB/LIN bit of the 16 BPP Control register specifies the generation of the low order 3 or 2 bits. If ZIB (Zero Intensity Black), the low order bits are made 0. If LIN (Linear), the low order bits are made equal to the high order bits. This causes the 5 or 6 bits to expand to 8 bits in a linear fashion, with both zero scale and full scale values used. With Zero Intensity Black, full scale cannot be achieved.

32 BPP

With 32 BPP format 2 pixels are obtained for each pixel port data access.

6 Bit Linear Palette Output

The 6BIT LIN (6 bit linear) bit of the Palette Control register affects the format of the color data read from the palettes and presented to the DACs in indirect color mode. It only has effect when the color resolution is set to 6 bits with the COL RES bit of the Miscellaneous Control 2 register and DCOL CNTL is set to indirect color.

If the palettes contain data with the two low order bits set to 00 (which will be the case when the palettes are loaded with COL RES set to 6 bits), without special processing the data values presented to the DACs will range from 0x00 through 0xfd. The maximum output of the DACs will be approximately 1.5% less than full scale (0xff). This will occur when 6BIT LIN is set to 1.

When 6BIT LIN is set to 0 (the default), then the outputs of the palettes will be modified to allow the DACs to reach full scale output. The modification consists of discarding the two low order bits from the palettes, and substituting the two high order bits for the two low order bits presented to the DACs. (i.e., the palette bits presented to a DAC will be bits 7 6 5 4 3 2 7 6).

With this bit substitution there will be a “linear” mapping of the palette data range (0x00 - 0xfd) to the DAC data range (0x00 - 0xff), and the DACs will operate over their full range.

If COL RES = 1 (8-bit color resolution) the palette outputs are presented to the DACs unchanged, and 6BIT LIN has no effect. The DACs will operate over the 8-bit range from completely off to full scale on.

Palette linear output is intended for emulation of the VGA 6-bit DACs in which the palette is loaded with 6-bit colors in the 6 high-order bits by setting COL RES to 6-bits. However, regardless of how the palette was loaded or what the pixel format is (VGA, 8, 15/16, 32 BPP), if enabled (DCOL = indirect, COL RES = 6 bit, 6BIT LIN = 0) the palette outputs will be affected as discussed above.

In summary, with the default conditions for VGA mode (indirect color, 6-bit color resolution, 6BIT LIN = 0), there will be a linear mapping of the 6-bit VGA palette data to the DACs, and the DACs will operate over their full range. The mapping can be turned off by setting 6BIT LIN to 1, in which case the 8 bits from the palettes are presented to the DACs unmodified. With 00 in the two low order bits of the palettes the DACs will not reach full scale output.

With 8-bit color resolution (indirect color), or with direct color, the setting of 6BIT LIN has no effect.

Table 3-1.6. Pixel Format Table

PIXEL PORT BIT	8 BPP	555 CONTIG	565 CONTIG	32 BPP
0	A0	A0BLU	A0BLU	A0
1	A1	A1BLU	A1BLU	A1
2	A2	A2BLU	A2BLU	A2
3	A3	A3BLU	A3BLU	A3
4	A4	A4BLU	A4BLU	A4
5	A5	A0GRN	A0GRN	A5
6	A6	A1GRN	A1GRN	A6
7	A7	A2GRN	A2GRN	A7
8	B0	A3GRN	A3GRN	A8
9	B1	A4GRN	A4GRN	A9
10	B2	A0RED	A5GRN	A10
11	B3	A1RED	A0RED	A11
12	B4	A2RED	A1RED	A12
13	B5	A3RED	A2RED	A13
14	B6	A4RED	A3RED	A14
15	B7	UNUSED	A4RED	A15
16	C0	B0BLU	B0BLU	A16
17	C1	B1BLU	B1BLU	A17
18	C2	B2BLU	B2BLU	A18
19	C3	B3BLU	B3BLU	A19
20	C4	B4BLU	B4BLU	A20
21	C5	B0GRN	B0GRN	A21
22	C6	B1GRN	B1GRN	A22
23	C7	B2GRN	B2GRN	A23
24	D0	B3GRN	B3GRN	A24
25	D1	B4GRN	B4GRN	A25
26	D2	B0RED	B5GRN	A26
27	D3	B1RED	B0RED	A27
28	D4	B2RED	B1RED	A28
29	D5	B3RED	B2RED	A29
30	D6	B4RED	B3RED	A30
31	D7	UNUSED	B4RED	A31
32	E0	C0BLU	C0BLU	B0
33	E1	C1BLU	C1BLU	B1
34	E2	C2BLU	C2BLU	B2
35	E3	C3BLU	C3BLU	B3
36	E4	C4BLU	C4BLU	B4
37	E5	C0GRN	C0GRN	B5
38	E6	C1GRN	C1GRN	B6
39	E7	C2GRN	C2GRN	B7
40	F0	C3GRN	C3GRN	B8
41	F1	C4GRN	C4GRN	B9
42	F2	C0RED	C5GRN	B10
43	F3	C1RED	C0RED	B11
44	F4	C2RED	C1RED	B12
45	F5	C3RED	C2RED	B13
46	F6	C4RED	C3RED	B14

PIXEL PORT BIT	8 BPP	555 CONTIG	565 CONTIG	32 BPP
47	F7	UNUSED	C4RED	B15
48	G0	D0BLU	D0BLU	B16
49	G1	D1BLU	D1BLU	B17
50	G2	D2BLU	D2BLU	B18
51	G3	D3BLU	D3BLU	B19
52	G4	D4BLU	D4BLU	B20
53	G5	D0GRN	D0GRN	B21
54	G6	D1GRN	D1GRN	B22
55	G7	D2GRN	D2GRN	B23
56	H0	D3GRN	D3GRN	B24
57	H1	D4GRN	D4GRN	B25
58	H2	D0RED	D5GRN	B26
59	H3	D1RED	D0RED	B27
60	H4	D2RED	D1RED	B28
61	H5	D3RED	D2RED	B29
62	H6	D4RED	D3RED	B30
63	H7	UNUSED	D4RED	B31

Controls

Blanking Control

BLANK is latched by the rising edge of LCLK. When BLANK is active (low), the data presented to the DACs is forced to zeroes. When BLANK is inactive (high), the pixel data or VGA data is considered valid, and the data is presented to the DACs as determined by the current mode of operation. Cursor data will override pixel data when the cursor is to be displayed.

Vertical Blanking

When BLANK is active (low) an internal counter is used to determine whether or not the current blanking interval is vertical blanking. If the counter reaches its maximum count of 2048 pixels, an internal signal will become active to indicate that the end of the current frame has been reached. This internal signal will remain active until BLANK becomes inactive (high). This vertical blanking detection is used by the cursor logic to position the cursor (if enabled) in the following frame. It is also used by the SR (if enabled) to control the accumulation of a signature for one complete frame of pixel data.

Sync Control

Three sync signals are brought into the device on two pins, HCSYNCIN and VSYNCIN.
Four registers control what is done with these signals:

- Sync Control (index 0x0003)
- Horizontal Sync Position (index 0x0004)
- DAC Operation (index 0x0006)
- Power Management (index 0x0005)
- Miscellaneous Control 1 (index 0x0070)

Horizontal sync on HCSYNCIN is processed and sent out on HSYNCOUT. Vertical sync on VSYNCIN is processed and sent out on VSYNCOUT.

The intent of processing horizontal sync is to delay it to match the delay seen by the pixel data from the inputs (VGA[7:0] or PIX[63:0]) to the DAC outputs. In addition, the signal may be inverted, forced low or high, or tristated.
A mismatch between pixel delay and horizontal sync delay can cause a visible effect, that is, the display may not be centered horizontally on the screen. The vertical display timings are generally such that mismatches are not visible. Vertical sync is brought in on VSYNCIN and sent out on VSYNCOUT to provide the same invert, force low or high, and tristate controls as provided for horizontal sync.
Composite sync on HCSYNCIN may be injected onto the Green DAC output for composite-sync-on-green. This function is enabled by setting the SOG bit of the DAC Operation register. If this bit is off HCSYNCIN generally is not used for composite sync.
However, in lieu of providing a composite sync externally, if horizontal sync is provided on the HCSYNCIN and vertical sync is provided on the VSYNCIN, a “synthetic” composite sync may be generated internally by exclusive ORing these two signals. This is done by set-ting the XOR SYNC bit of the Miscellaneous Control 1register.
If XOR SYNC is set and SOG is set the internally XORed sync signal will be injected onto the Green DAC output. If XOR SYNC is set but SOG is not set, then the synthetic composite sync will be presented on the HSYNCOUT output, but will not be injected on the Green DAC output.
Composite sync, whether it is the signal presented on HCSYNCIN or the synthetic internal composite sync, is delayed internally to match the pixel pipeline delay.
Since external horizontal and composite sync are shared on the same pin, only one of them should be enabled at a given time. For example, if the signal on HCSYNCIN is horizontal sync and XOR SYNC is not being used, then the SOG bit on the DAC control register should be off. Or, if SOG is on to inject composite sync on HCSYNCIN onto the green DAC output, some decision must be made on how to handle the HSYNCOUT output (force low, high, tristate, or leave unconnected).

Clocking and Pipeline Delay

Horizontal Sync

The clocking and delay of HCSYNCIN to HSYNCOUT depends on the SOG bit in the DAC operation register bit of the Sync Control register.
If SOG is off (no composite sync), then HCSYNCIN is latched on the rising edge of LCLK and delayed internally to match the pixel pipeline delay before being sent out on HSYNCOUT. Also, additional delay may be added with the Horizontal Position register (see section below).
If SOG is on (composite sync) then HCSYNCIN is passed directly to HSYNCOUT without latching and without pipeline delay matching. (This is not a typical use for this input, it is just a by-product of sharing horizontal sync with composite sync.) In this case, the sync signals presented on the green channel are delayed internally to match the pixel pipeline delay. Again, additional delay may be added with the horizontal position register.

Vertical Sync

VSYNCIN is passed directly to VSYNCOUT without latching and without pipeline delay matching.

Composite Sync

The HCSYNCIN input is always latched on the rising edge of LCLK for use as composite sync. When enabled with the SOG bit, it is delayed internally to match the pipeline delay of the pixel data, and then is injected onto the Green DAC output. As with horizontal sync, additional delay can be added with the Horizontal Sync Position register.

Horizontal Position Control

Additional delay of 0 to 15 pixel clock periods may be added to the horizontal sync and composite sync signals with the Horizontal Sync Position register.

The intent of this additional delay is to provide a “fine tune” control of the horizontal screen position. Typically the incoming sync signals can only be adjusted in multiples of the pixel clock. The additional delay added with the Horizontal Position Control register adjusts the screen position with pixel increments. The Horizontal Position register can be used on horizontal sync when SOG is off. The register can be used with composite sync when SOG is on.

Additional Sync Control

The polarity of the received HCSYNCIN input may be inverted before it is applied to the green DAC using the CSYN INVT bit of the Sync Control register.
The polarity may be inverted between HCSYNCIN and HSYNCOUT using the HSYN INVT bit, and the polarity may be inverted between VSYNCIN and VSYN-COUT using the VSYN INVT bit.
The HSYNCOUT and VSYNCOUT signals may be individually forced low, forced high, or forced to high impedance using the HSYN CNTL and VSYN CNTL bits of the Sync Control register.
As discussed in the section *Clocking Power*, the clocks to the sync delay circuits can be shut off with the SYNC PWR bit of the Power Management register.

Cursor Operation

The cursor is a 32x32 or 64x64 pixel pattern that is overlaid on the display pixels just before presentation to the DACs. The cursor size, 32x32 or 64x64 is set with the CURS SIZE bit of the Cursor Control register. Pixel columns are numbered left to right starting with 0. Pixel rows are numbered top to bottom starting with 0.

Cursor Enable

The cursor is enabled when the CURSOR MODE bits of the Cursor Control register are not 00. When enabled, the cursor will display if it has not been moved off-screen. If disabled (CURSOR MODE = 00), the cursor will not be displayed.
The cursor may be used with either pixel port (VGA or PIX), with any of the pixel formats (VGA, 8, 15/16, 32 BPP), and with indirect or direct color.

Cursor Array

The cursor image is stored in the Cursor Array. The array is organized 1024x8 (1024 bytes). It is accessed as Indexed Data using index addresses 0x0100 through 0x04ff.
Each pixel of the cursor uses 2 bits, thus 4 cursor pixels are stored in each byte of the array. The entire array is used to contain the 64x64 cursor image (4 pixels/byte ´ 1024 bytes = 4096 pixels = 64x64).
For the 32x32 cursor only 256 bytes are required (4 pixels/ byte ´ 256 bytes = 1024 pixels = 32x32.) The cursor array is divided into four contiguous slots to allow the storage of four (32x32) cursor images. The SMLC SLOT bits of the Cursor Control register are used to select one of the four slots for display. The SMLC SLOT bits have no effect when the cursor size is 64x64.
Storage of the cursor within the array starts with the top row. For the 64x64 cursor the first 16 bytes hold row 0, the next 16 bytes hold row 1, and so on, starting with the first byte in the array at index address 0x0100.
For the 32x32 cursor the first 8 bytes hold row 0, the next 8 bytes hold row 1, and so on, starting with the first byte in a slot (index addresses 0x0100, 0x0200, 0x0300 or 0x0400).
Within a row the pixels are stored left to right in groups of four. The first byte holds pixels 0, 1, 2, 3, the next byte holds pixels 4, 5, 6, 7, and so on.
Within a byte the four pixels may be stored right to left or left to right, depending on the PIX ORDR bit of the Cursor Control register. If PIX ORDR = 0 the pixels are stored right to left (3, 2, 1, 0); if PIX ORDR = 1 the pixels are stored left to right (0, 1, 2, 3).

Cursor Array Access

Cursor Array writes and reads are synchronized with the internal pixel clock, so the pixel clock must be running for microprocessor accesses to be valid. If this condition is met, the cursor array may be written or read at any time.
Microprocessor read accesses of the cursor array may disturb the cursor image if it is being displayed at that time. However, no more than one cursor pixel will be disturbed per cursor read access. Microprocessor write accesses of the cursor array will not disturb the cursor.

Cursor Array Writes

A write to the cursor array is accomplished by writing the Index High and Index Low registers with an index address for the array (0x0100 – 0x04ff), followed by a write of the desired data to Index Data. If auto-increment is turned on, the entire array may be written sequentially by repeated writes to Index Data.

Cursor Array Reads

To meet the bus timings for reads, the cursor array read data is pre-fetched. A pre-fetch is triggered by writing the Index High or Index Low register such that the resulting index address is for an entry in the array (0x0100 - 0x04ff). At the end of the write cycle the cursor array will be read at the address held in the index address registers, and the read data will be held in an internal register. A subsequent read of Index Data will read this pre-fetched data. At the end of the read another pre-fetch will be triggered. If auto-increment is turned on, this pre-fetch will be for the next address in the array. Thus, the entire array can be read by repeated reads from Index Data.
The pre-fetching of cursor array data will stop if

1. The index register auto-increments beyond 0x04ff
   OR
2. A write is done to Index Data

Cursor Modes

Each pixel of the cursor is specified with 2 bits. There are two fundamental ways to interpret these bits: Standard Modes and Advanced Mode.
The ACA ENAB (Advanced Cursor Attribute Enable) bit in the Advanced Cursor Control register selects either the Standard Cursor (ACA ENAB = ‘0’) or the Advanced Cursor (ACA ENAB = ‘1’).
For either cursor type, there are three cursor colors that may be displayed. The colors are stored in the Cursor Color 1 Red, Green, Blue, Cursor Color 2 Red, Green, Blue, and Cursor Color 3 Red, Green, Blue registers.
Each red, green, and blue register is 8 bits, yielding a full 24-bit color for each of the three cursor colors. The cursor color is always 24 bits, and is not affected by the COL RES or 6BIT LIN control bits, or any of the pixel formats (VGA, 4, 8, 15/16, 24, 32 BPP).
The Advanced Cursor has an additional color, Color 0, which is always Black. That is, all 24 bits are ‘0’s.
Both cursor types can generate three types of displayed pixels:

1. Transparent - the underlying pixel (the normally displayed pixel) is displayed. This pixel will either be a palette output or a formatted VRAM pixel, depending on whether the pixel format is VGA, indirect color, or direct color.
2. Solid Color - One of the three cursor colors is displayed (Cursor Color 1, 2 or 3). The Advanced Cursor can also display Cursor Color 0.
3. Highlighted - a bitwise complement of the underlying display pixel is displayed. The intent is to highlight the cursor by “reversing” the color of the background pixels.

The Advanced Cursor has an additional display type:

4. Translucent - the underlying pixel is “mixed” with one of the Cursor Colors (0, 1, 2 or 3), such that a “dimmed” version of the underlying pixel appears to “shine through” a translucent version of the Cursor Color.
   This is achieved by shifting each red, green and blue component of the underlying pixel right by 1 bit (dimming the pixel by dividing by 2). The cursor color is mixed in by setting the now vacant high order

bit of each component of the display pixel with the high order bit of the corresponding Cursor Color component.

Standard Cursor

The Standard Cursor modes are used when the cursor is ON (CURSOR MODE not equal to '00'), and the Advanced Attributes are OFF (ACA ENAB = '0').
The Standard Cursor has three modes of display, specified with the remaining bit values of CURSOR MODE (CURSOR MODE = '01', '10' or '11'). The two cursor pixel bits can select one of three colors, one of two colors plus highlighting, or just two colors (where the selected cursor colors are solid colors):

Table 3-1.7. Standard Cursor Modes

	DISPLAY PIXEL		
CURSOR PIXEL	**Mode 0 CURSOR MODE = 01**	**Mode 1 CURSOR MODE = 10**	**Mode 2 CURSOR MODE = 11**
00	Transparent	Cursor Color 1	Transparent
01	Cursor Color 1	Cursor Color 2	Transparent
10	Cursor Color 2	Transparent	Cursor Color 1
11	Cursor Color 3	Highlighted	Cursor Color 2

Advanced Cursor

The Advanced Cursor mode is used when the cursor is ON (CURSOR MODE not equal to '00'), and the Advanced Attributes are ON (ACA ENAB = '1').
For the Advanced Cursor, the two cursor pixel bits simply select one of four colors:

Table 3-1.8. Advanced Cursor Colors

CURSOR PIXEL	CURSOR COLOR
00	Color 0
01	Color 1
10	Color 2
11	Color 3

Associated with each color is a two bit Attribute. The Attribute specifies the display of the cursor pixel:

Table 8-9. Advanced Cursor Attributes

ATTRIBUTE	DISPLAY PIXEL
00	Transparent
01	Solid color
10	Translucent
11	Highlighted

The attribute values for each color are set in the Advanced Cursor Attribute register. This register contains a two bit attribute value for each of the four colors.
Example: Suppose a given cursor pixel value read from the cursor array is '01'. From Table 8-9, Cursor Color 1 will be selected.
Bits 3 and 2 of the Advanced Cursor Attribute register contain the attribute value for Color 1. Suppose these bits are '01'. This specifies a solid color, so the values in the Cursor Color 1 Red, Green and Blue registers will be displayed.
If Bits 3 and 2 of the Advanced Cursor Attribute register were '10', a translucent pixel would be selected, and the values in the Cursor Color 1 Red, Green and Blue registers would be mixed in with the underlying pixels for display.

Cursor Hot Spot

The hot spot is the point within the cursor that is used to locate the cursor's position on the screen. Any pixel within the cursor may be identified as the hot spot.
The Cursor Hot Spot X and Cursor Hot Spot Y registers hold the unsigned cursor pixel X (column) and Y (row) coordinates for the hot spot. The range for the X and Y values is 0 to 31 for the 32x32 cursor and 0 to 63 for the 64x64 cursor.

Cursor Position

The Cursor X Low, Cursor X High, Cursor Y Low, and Cursor Y High registers specify the position of the cursor (the cursor hot spot) on the screen.
The X and Y positions are specified as signed numbers in two's complement format. The High and Low pairs yield 16-bit position registers, of which 12 bits plus a sign bit are used.
The hardware automatically extends the sign bit into the unused bit positions of the position registers. The valid X and Y ranges are -4096 to +4095.
The X and Y screen coordinates are for display pixels. (0,0) is the upper left corner pixel of the screen. The X value increases positively left-to-right, and the Y value increases positively top-to-bottom. Negative X values are to the left of the display area and negative Y values are above the top of the display area.
The cursor is clipped by the edges of the screen. For example, if the hot spot is (0,0) the full cursor will be displayed in the upper left corner if the X position is +0 and the Y position is +0. If the X value is changed to -1 (0xffff) only columns 1 through 31 of the cursor will be displayed. If the X value is -31 (0xffe1) only column 31 of the cursor will be displayed. If the X value is more negative than -31 the cursor will not be displayed.

Cursor Update and Display

Position

Writing any of the Cursor X Low, Cursor X High, or Cursor Y Low registers will not affect the position of the cursor on the screen. When the Cursor Y High register is written, the X and Y positions are captured in a second set of registers.
When vertical blanking is detected (see the section, "Vertical Blanking", earlier in this appendix), the "captured" X and Y positions are sampled. The sampled position is saved until it is re-sampled on the next vertical blanking time. Between vertical blanking times the sampled position is used, along with the Cursor Hot Spot, to calculate which pixels of the cursor are used and where they are displayed on the screen.

Controls

When vertical blanking is detected the Cursor Control register is sampled along with the X and Y position registers. This allows the cursor to be toggled on and off on a frame-by-frame basis with the CURSOR MODE bits, and if the cursor is 32x32, it allows toggling among the four slots on a frame-by-frame basis using the SMLC SLOT bits.

Other

Changes to the Cursor Color registers and the Cursor X and Y Hot Spot registers are propagated to the cursor logic as soon as they are made, so if they are updated while the cursor is being displayed the cursor image will be disturbed.
Changes to the cursor array are also propagated to the cursor logic as soon as they are made. Also, as noted above, microprocessor read accesses of the cursor array may interfere with the cursor display logic. For example, with a 32x 32 cursor being displayed from slot 0, micro-processor read accesses to slot 1 may cause the display of the slot 0 cursor to be disturbed.
It is recommended that Cursor Array Reads and changes to the Cursor Color registers and the Cursor X and Y Hot Spot registers be made only when the cursor is disabled, off screen, or during vertical blanking time.

DAC Control

Several miscellaneous features of the DACs are controlled by the DAC Operation register.

SOG - Composite Sync-On-Green

When the SOG bit is set, composite sync will be merged with the pixel data on the green DAC.
The source of the composite sync can be the signal on the HCSYNCIN input, or an internally generated (synthetic) sync signal formed by the XOR of the signals on the HCSYNCIN and VSYNCIN inputs. The incoming HCSYNCIN signal may be inverted and/or delayed before presentation at the DAC. See sections 6.5 through 6.7 for more details.

BRB - Blank Red and Blue DACs

When this is set the red and blue DACs are set to the blanking level. This is intended for use when a monochrome display is driven by the green DAC.

DPE - DAC Blanking Pedestal Enable

When off, the DAC pedestal is disabled (blanking level = 0 IRE). When on, the pedestal is enabled (7.5 IRE).

Power Management

The following registers are used to control power dissipation:

- Power Management (index 0x0005)
- Miscellaneous Clock Control (index 0x0002)
- Sync Control (index 0x0003)
- Miscellaneous Control 1 (index 0x0070)

DAC Power

The analog portion of the DACs can be shut down with the DAC PWR bit of the Power Management register. A small amount of current (approximately 100 µA) will continue to be drawn through the VREFIN input. This can be eliminated if the voltage on VREFIN is reduced to 0 V.

Clocking Power

Most of the digital logic power dissipation occurs as a result of clocking. The ICLK PWR, SCLK PWR, and SYNC PWR bits of the Power Management register are used to inhibit the digital logic clocking.
The ICLK PWR bit, when set, inhibits all internal clocking except for the following:

- The PLL.
- The palette arrays and the cursor array control logic. The clocks to the internal logic are left running because this is required for microprocessor access.
- SCLK The circuitry that generates this clock is left running in case external components need to run off these clocks.
- The horizontal and vertical sync delay circuits. These circuits are left running to allow sync signals to propagate to the display monitor.

When the ICLK PWR bit is set the DAC outputs will remain stuck at whatever was last clocked into the DACs, unless the DACs are shut down with DAC PWR.
The SCLK PWR bit may be set to disable the clocking to the SCLK generator. The resultant static SCLK output may be left at either the low or high state. As noted above, the SCLK output may be tristated with the SCLK DSAB bit of the Miscellaneous Clock Control register.

The SYNC PWR bit may be set to disable the clocking to the horizontal and vertical sync circuits. These outputs may be left at either the low or high state. (But note that the outputs can be forced high or low or tristated with the HSYN CNTL and VSYN CNTL bits of the Sync Control register.)
The starting and stopping of clocks with the SCLK PWR, DDOT PWR, and SYNC PWR bits is asynchronous. Thus, "chopped" pulses may be produced on the SCLK, DDOTCLK, HSYNCOUT and VSYNCOUT outputs when these bits are changed.
Similarly, changing the ICLK PWR bit can disturb the stopping and starting of the internal clocks such that the display is disturbed for a frame. It is recommended that the DACs be blanked with the BLANK CNTL bit of the Miscellaneous Control 2 register before shutting off the clocks, and that a frame be run by after turning on the clocks before the DACs are unblanked again with BLANK CNTL.

PLL Power

The PLL can be shut off with the PLL ENAB bit of the Miscellaneous Clock Control register. This, in conjunction with turning off the DACs, produces the lowest power consumption.
Note that in general the PLL drives SCLK, and the incoming LCLK is generally derived externally from SCLK. If the PLL is disabled, SCLK will stop running regardless of the setting of SCLK PWR, internal clocking will stop regardless of the setting of ICLK PWR, sync signal clocking will stop regardless of the setting of SYNC PWR, and external circuitry running off SCLK and/or DDOTCLK will stop running.
If EXTCLK is used instead of the PLL the same effect can be achieved by stopping EXTCLK.

Diagnostic Support

SR

The SR (Signature Register) consists of three registers, SR Red, SR Green and SR Blue. Together the three registers form a 24-bit shift register with feedback to accumulate a signature of the data presented to the DACs during one screen frame.
Signature accumulation is controlled by the SR CNTL bit in the Miscellaneous Control 1 register, and detection of vertical blanking.
Signature accumulation uses the following sequence:

1. Write the Miscellaneous Control 1 register to change the SR CNTL bit from '0' to '1'.
2. The next detection of vertical blanking will "arm" the SR circuits.
3. The start of the next frame (in non-interlace mode when the BLANK input goes from low to high), and starts the accumulation of the signature.
4. A signature is accumulated for all pixel data presented to the DACs in a frame. Note that this includes cursor pixels if they are displayed.
5. Signature accumulation stops with the next detection of vertical blanking,. At this point the three SR registers are frozen.

The accumulated signature can now be read from the SR Red, SR Green and SR Blue registers.

DAC Comparators

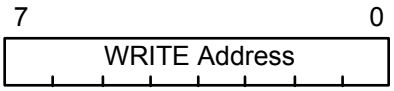
Each DAC output is connected to a comparator. Both latched and unlatched copies of the comparator outputs can be read from the DAC Sense register. The logical AND of either the latched or unlatched comparator bits is presented on the SENSE output. With the internally applied reference voltage of 0.35 V, the corresponding Sense bit will be '1' when the DAC output is 0 to 0.28 V or '0' when it is 0.42 V to 0.70 V.
These values apply when the DAC is doubly terminated in 75 W, RREF=698 W, and no sync or blank is present.

Register Descriptions

Direct Access Registers

The direct access registers are addressed using AD[4:2] inputs.

Palette Address (Write Mode)



AD[4:2]: 000
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 WRITE Address - Palette address in write mode.

Operation of this register is discussed in the section, “Microprocessor Access”, earlier in this appendix.

Palette Data

AD[4:2]: 001
Access: Read/Write
Power on Value: Undefined

The format of the palette data depends on the color resolution, 6 or 8 bit.

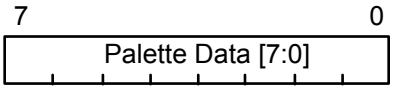
6 bit color resolution
Miscellaneous Control 2 COL RES = 0

Bits 7 - 6 00
Bits 5 - 0 6 bit palette data

On WRITEs bits 7:6 from the microprocessor are discarded, bits 5:0 are written to bits 7:2 internally, and internal bits 1:0 are set to ‘00’.

On READs internal bits 7:2 are read as bits 5:0, and bits 7:6 are returned as ‘00’.

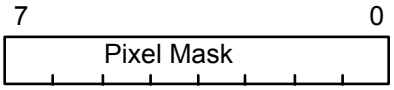
8 bit color resolution
Miscellaneous Control 2 COL RES = 1



Bits 7-0 8 bit palette data. Bits 7:0 are written/read internally as bits 7:0

Operation of this register is discussed in 1.0 “Microprocessor Access” on page 1.

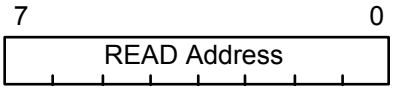
Pixel Mask



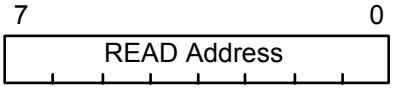
AD[4:2]: 010
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Pixel Mask

In indirect color modes this register masks the pixel values used to index into the palettes. Each bit is ANDed with its corresponding pixel bit. A value of 0xff is required to pass the pixel values to the palettes unchanged. The same mask is applied to each of the red, green, and blue pixel addresses into the palettes.

Palette Address (Read Mode) / Palette Access State



AD[4:2]: 011
Access: Write
Power on Value: -
Bits 7 - 0 READ Address - Palette address in read mode.

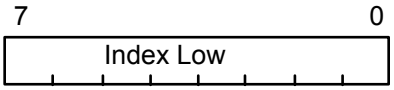


AD[4:2]: 011
Access: Read
Power on Value: Undefined
PADR RFMT: 0
Bits 7 - 0 READ Address - Palette address in read mode.

AD[4:2]: 011
Access: Read
Power on Value: Undefined
PADR RFMT: 1
Bits 7 - 2 Reserved
Bits 1 - 0 ACC STATE - Palette Access State. Reports which mode was used on last write of Palette Address Register.
00 Write Mode
11 Read Mode

Note that the palette address to be read is written into this register, but the contents that are read depends on the PADR RFMT bit in the Miscellaneous Control 1 register. Operation of these registers is discussed in the section, “Microprocessor Access”, earlier in this appendix.

Index Low



AD[4:2]: 100
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Index Low

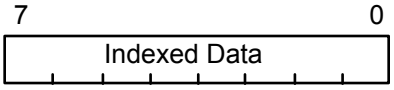
This register, together with Index High, forms the internal index register. It selects the register that will be accessed when the Indexed Data register is written or read.

Index High

AD[4:2]: 101
Access: Read/Write
Power on Value: Undefined
Bits 7 - 3 Reserved
Bits 2 - 0 Index High

This register provides the high-order bits of the internal index register. If auto-increment is turned on, the resulting index is not defined if an increment past the maximum index value occurs.

Index Data



AD[4:2]: 110
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Indexed Data

A write or read to this register will write or read the register addressed by the internal index register (Index High and Index Low). Following a write or read to Indexed Data, the index register will be incremented if auto-increment is turned on (INDX CNTL bit of the Index Control register).

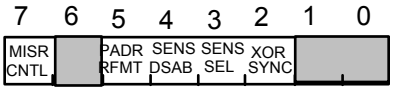
Index Control

AD[4:2]: 111
Access: Read/Write
Power on Value: Undefined
Bits 7 - 1 Reserved
Bit 0 INDX CNTL - Index Control. Controls auto-increment of the index register.
0 (Off) - no auto-increment.
1 (On) - the index register (Index High and Index Low) will increment by one following a write or read to Indexed Data.

Indexed Registers

The indexed registers are accessed by setting the desired address into the internal index register (Index High and Index Low) and writing or reading the Indexed Data register.

Miscellaneous Control 1



Index: 0x0070
Access: Read/Write
Power on Value: 0x00

Bit 7 SR CNTL

0 Off. If the SR is running, it will stop at the beginning of the next frame.
1 On. The SR will start accumulating a signature at the start of the next frame (end of vertical blanking).

Bit 6 Reserved

Bit 5 PADR RFMT

Palette Address Register (Read Mode) Format. Specifies the contents returned from the Palette Address register, read mode (AD[4:2] = 011)

0 Return the eight bits of the read address
1 Return the palette access state in the two low order bits (00 write and 11 read)

Bit 4 SENS DSAB

SENSE Driver Disable

0 SENSE driver enabled
1 SENSE driver disabled (tristated)

Bit 3 SENS SEL

Sense Select. Selects which bit of the DAC Sense register is presented on the SENSE driver.

0 Bit 3 - Unlatched Sense
1 Bit 7 - Latched Sense

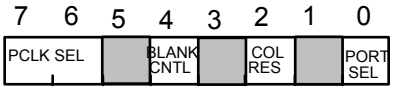
Bit 2 XOR SYNC

Composite Sync Source - Selects source of composite sync.

0 Composite sync taken from HCSYNCIN input.
1 Composite sync generated internally from XOR of HCSYNCIN input and VSYN-CIN input.

Bits 1, 0 Reserved

Miscellaneous Control 2



Index: 0x0071
Access: Read/Write
Power on Value: 0x00

Bits 7 - 6 PCLK SEL - Pixel Clock Select. Specifies the source of the internal pixel clock.

00 LCLK input
01 Internal PLL output
10 REFCLK input
11 Reserved

NOTE: A selection of 00 (LCLK input) for the pixel clock is required and only valid when PORT SEL= 0 (VGA data inputs).

Bit 5 Reserved

Bit 4 BLANK CNTL - Blanking Control

0 Normal operation.
1 DACs are blanked.

No pixel data is presented on the DACs, but all other operations remain normal, including the collection of a signature if the MISR is turned on.

Bit 3 Reserved

Bit 2 COL RES - Color Resolution

0 6-bit
1 8-bit

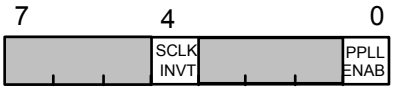
With 6-bit color resolution only 6 bits of microprocessor data are loaded into the palettes. Microprocessor data bits D[5:0] are written to/read from palette bits [7:2]. Internally 00 is written to palette bits [1:0], and on reads D[7:6] are forced to 00. Also with 6-bit color resolution the two low order bits presented from the palettes to the DACs are controlled by Palette Control bit 6BIT LIN.

With 8-bit color resolution all 8 bits from/to the microprocessor are written/ read to the palette, and the 8 bits presented to the DACs are unmodified. The 6BIT LIN bit has no effect.

Bit 1 Reserved

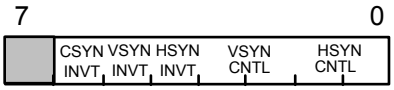
Bit 0 PORT SEL - Port Select
0 VGA Data inputs.
1 VRAM pixel port inputs.

Miscellaneous Clock Control



Index: 0x0002
Access: Read/Write
Power on Value: 0x01
Bit 7 - 5 Reserved
Bit 4 SCLK INVT - Inverts the SCLK output.
Bits 3 - 1 Reserved
Bit 0 PPLL ENAB - Pixel PLL Enable
0 Pixel PLL programming disabled.
1 Pixel PLL programming enabled.

Sync Control



Index: 0x0003
Access: Read/Write
Power on Value: 0x00
Bit 7 Reserved
Bit 6 CSYN INVT - Invert incoming
HCSYNCIN when used as Composite Sync
0 Do not invert incoming HCSYNCIN
1 Invert incoming HCSYNCIN

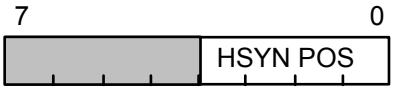
Bit 5 VSYN INVT - Vertical Sync Invert
0 Do not invert incoming VSYNCIN
1 Invert incoming VSYNCIN

Bit 4 HSYN INVT - Invert incoming
HCSYNCIN when used as Horizontal Sync
0 Do not invert incoming HCSYNCIN
1 Invert incoming HCSYNCIN

Bits 3 - 2 VSYN CNTL - Vertical Sync Output Control
00 Normal output
01 VSYNCOUT forced high
10 VSYNCOUT forced low
11 VSYNCOUT disabled (3- stated)

Bits 1 - 0 HSYN CNTL - Horizontal Sync Output Control
00 Normal output
01 HSYNCOUT forced high
10 HSYNCOUT forced low
11 HSYNCOUT disabled (tristated)

Horizontal Sync Control



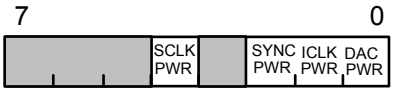
Index: 0x0004
Access: Read/Write
Power on Value: 0x00
Bits 7 - 4 Reserved
Bits 3 - 0 HSYN POS - Horizontal Sync Position.

Specifies number of additional pixel delays to add to the horizontal sync signal and the composite sync signal.

If the SOG bit of the DAC Operation register is on, the additional pixel delays are not added to HSYNCOUT, but are added to the syncs presented on the green channel instead.

0000	0 pixels
0001	1 pixel
0010	2 pixels
0011	3 pixels
0100	4 pixels
0101	5 pixels
0110	6 pixels
0111	7 pixels
1000	8 pixels
1001	9 pixels
1010	10 pixels
1011	11 pixels
1100	12 pixels
1101	13 pixels
1110	14 pixels
1111	15 pixels

Power Management



Index: 0x0005
Access: Read/Write
Power on Value: 0x00
Bits 7 - 5 Reserved

Bit 4 SCLK PWR - SCLK Power Control
0 Normal Operation
1 Disable clocks to SCLK generator

Bit 3 Reserved

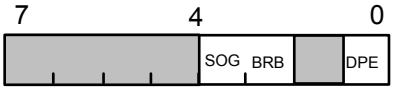
Bit 2 SYNC PWR - Sync Power Control
0 Normal Operation
1 Disable clocks to horizontal and vertical sync circuits

Bit 1 ICLK PWR - Internal Clock Power Control
0 Normal Operation
1 Disable all internal clocks except those for the SCLK generator, and horizontal and vertical sync circuits.

A clock is left running to allow microprocessor access of the palette and cursor array, but otherwise the palette and cursor RAMs will not be clocked. The two RAMs will retain their contents.

Bit 0 Reserved

DAC Operation



Index: 0x0006
Access: Read/Write
Power on Value: 0x02
Bits 7 - 4 Reserved

Bit 3 SOG - Composite Sync-On-Green
0 Sync is disabled on Green DAC.
1 Sync is enabled on Green DAC.

Bit 2 BRB - Blank Red and Blue DACs
0 Red and Blue DACs have normal function.
1 Red and Blue DACs are always blanked.

Bit 1 Reserved

Bit 0 DPE - DAC blanking Pedestal Enable
0 Blanking pedestal disabled (0 IRE)
1 Blanking pedestal enabled (7.5 IRE)

Palette Control

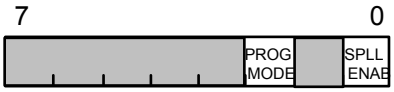
Index: 0x0007
Access: Read/Write
Power on Value: 0x00
Bit 7 6BIT LIN - 6 Bit Linear Color

0 Apply linear palette output
1 Do not apply linear palette output

This bit only has effect with indirect color modes, and when Color Resolution is set to 6 bits (Miscellaneous Control 2 COL RES bit = 0). For the 8 bits of palette output for each color, the high order two bits 7 and 6 will be substituted for the two low order bits 1 and 0.

Bits 6 - 0 Reserved.

System Clock Control



Index: 0x0008
Access: Read/Write
Power on Value: 0x01
Bit 7 - 3 Reserved

Bit 2 PROG MODE - SYSCLK PLL programming mode

0 Compatibility mode – program with registers at 0x0015 and 0x0016 as described in the section, “PLL Compatibility Programming”, later in this appendix.
1 Standard mode - program with N, M, and P (registers at 0x0015, 0x0016, 0x0017) as described in the section, “PLL Operation and Programming”.

Bit 1 Reserved

Bit 0 SPLL ENAB - System PLL enable

0 SYSCLK PLL programming disabled
1 SYSCLK PLL programming enabled

Pixel Representation

Pixel Format

Index: 0x000a
Access: Read/Write
Power on Value: Undefined

Bits 7 - 3 Reserved

Bits 2 - 0 Pixel Format

000	Reserved
001	Reserved
010	Reserved
011	8 BPP
100	15/16 BPP
101	Reserved
110	32 BPP
111	Reserved

This register has no effect when the VGA port is selected.

8 Bit Pixel Control

Index: 0x000b
Access: Read/Write
Power on Value: Undefined

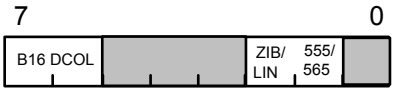
Bits 7 - 1 Reserved

Bit 0 B8 DCOL - 8 BPP Direct Color Control

0	Indirect Color (through the palette).
1	Direct Color (palette bypass).

Since the same 8-bit value will be applied to each of the Red, Green, and Blue DACs a monochrome image will be displayed. This register only affects 8 BPP mode.

16 Bit Pixel Control



Index: 0x000c
Access: Read/Write
Power on Value: Undefined

Bits 7 - 6 B16 DCOL - 16 BPP Direct Color Control

00	Indirect Color (always goes through the palette). Either the 555 or 565 format can be selected.
01	Reserved
10	Reserved
11	Direct Color (always bypasses the palette). Either the 555 or 565 format can be selected. The ZIB/LIN bit determines the expansion to 24 bits (low order bit fill).

Bits 5 - 3 Reserved

Bit 2 ZIB/LIN - Bit fill selection. For direct color this bit specifies how the low order bits of each color, R,G,B are filled when 555 or 565 formats are expanded to 24 bits.

0	ZIB - Zero Intensity Black. Low order bits are set to 0.
1	LIN - Linear. The low order bits are set to the values of the high order bits. For indirect color if CNT (contiguous) addressing is selected, then ZIB/LIN has no effect. If SPR (sparse) addressing is selected then ZIB/LIN must be set to 0 (ZIB). The palette addressing is undefined if LIN bit fill is selected with sparse addressing.

Bit 1 555/565 - Selects 5 bits per color (555) or 5 red, 6 green, 5 blue (565).

0	555
1	565

Bit 0 Reserved

32 Bit Pixel Control

Index: 0x000e
Access: Read/Write
Power on Value: Undefined

Bit 7 – 2 Reserved

Bits 1 - 0 B32 DCOL - 32 BPP Direct Color Control.

00	Indirect Color (always goes through the palette). 24 bits (8 bits each for Red, Green, Blue) are used to index into the palettes. The 8 high order bits (PIX[31:24], PIX[63:56]) are not used.
01	Reserved
10	Reserved
11	Direct Color (always bypasses the palette). 24 bits (8 bits each for Red, Green, Blue) are presented to the DACs. The 8 high order bits (PIX[31:24], PIX[63:56] are not used.

Pixel Clock Frequency Selection

Pixel PLL Control 1

Index: 0x0010
Access: Read/Write
Power on Value: 0x00

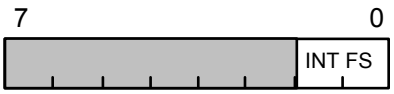
Bits 7 - 3 Reserved

Bits 2 - 0 EXT/INT
Determines the programming mode (standard or compatibility), and the source and selection for the Pixel PLL programming registers.

000	Reserved
001	Compatibility mode with external frequency select. One of two pairs of registers (M0/N0, M1/N1) are selected by VGA core signal FS.
010	Reserved
011	Compatibility mode with internal frequency select. One of four pairs of registers (M0/N0, M1/N1, M2/N2, M3/N3) selected by PLL Control 2[1:0] to provide the VCO divider/reference divider inputs to the Pixel PLL.
100	Standard mode: One of two sets of registers M0/N0/P0, M1/N1/P1 is selected with VGA core signal FS to provide the M, N, and P programming values to the Pixel PLL.
101	Standard mode with internal frequency select.
11x	Reserved

The Pixel PLL frequency is programmed using several values. Multiple sets of these values may be stored in registers sets. The desired set of values can be selected using external signals (the FS[1:0] inputs) or by using the internal FS[3:0] bits of the PLL Control 2 register. When the standard programming method is used then EXT/INT is set to 10x to use external selection or internal selection of frequency values. When one of the two compatibility methods is used then EXT/INT is set to 00x for external selection, and 011 for internal selection.

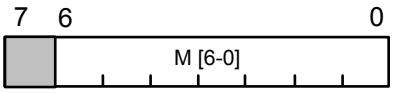
Pixel PLL Control 2



Index: 0x0011
Access: Read/Write
Power on Value: 0x00
Bits 7 - 2 Reserved
Bits 1 - 0 INT FS - Internal Frequency Selection. Identifies which Pixel PLL programming registers to use when Pixel PLL Control 1 register bits EXT/INT specify internal frequency selection (EXT/INT = 011 or 101).

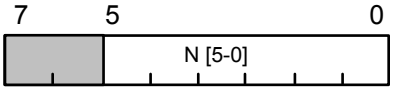
Pixel PLL Programming – Standard Mode

Pixel M0, Pixel M1



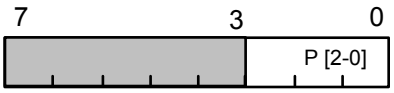
Index: 0x0020, 0x0024
Access: Read/Write
Power on Value: 0x05 (index 0x0020)
0x00 (others)
Bit 7 Reserved
Bits 6 - 0 M

Pixel N0, Pixel N1



Index: 0x0021, 0x0025
Access: Read/Write
Power on Value: 0x0e (index 0x0021)
0x00 (others)
Bit 7-6 Reserved
Bits 5 - 0 N

Pixel P0, Pixel P1



Index: 0x0022, 0x0026
Access: Read/Write
Power on Value: 0x00 (all)
Bits 7 - 3 Reserved
Bits 2 - 0 P

When the standard programming method is chosen for the Pixel PLL (Pixel PLL Control 1 register, EXT/INT bits = '100' or '101'), the registers at 0x0020 - 0x0027 provide the N, M, and P programming values.

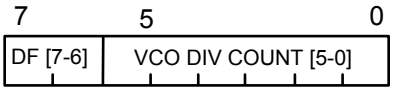
The valid ranges of these bits and their use are described above.

There are two sets of M, N, and P values. One of two sets is chosen under control of the PLL Control 1 and PLL Control 2 registers.

Note that, following a reset, the Pixel PLL is set to use compatibility programming using power on values appropriate for that mode (in registers at 0x0020 and 0x0021). Thus, when switched to standard programming, the power on values found in those registers are actually "left over" values that are not intended for use with the standard programming method.

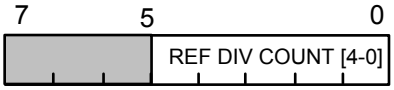
Pixel PLL Programming – Compatibility Mode

M0-M3



Index: 0x0020, 0x0022, 0x0024, 0x0026
Access: Read/Write
Power on Value: 0x05 (index 0x0020)
0x00 (others)
Bits 7 - 6 DF - Desired Frequency
Bits 5 - 0 VCO DIV COUNT - VCO Divide Count

N0-N3

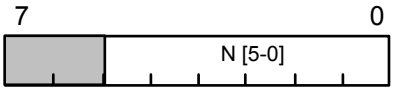


Index: 0x0021, 0x0023, 0x0025, 0x0027
Access: Read/Write
Power on Value: 0x0e (index 0x0021)
0x00 (others)
Bits 7 - 5 Reserved
Bits 4 - 0 REF DIV COUNT - Reference Divide Count

The above diagrams show the formats for the 4 ‘M’ and 4 “N” pixel frequency registers. These formats are selected when the EXT/INT bits (Pixel PLL Control 1 Register, bits 2:0) = 001 or 011.
The 8 registers are grouped into four pairs, M0/N0, M1/N1, M2/N2, M3/N3. For a given pair, the “M” register provides the Pixel PLL with the DF value and the VCO divide count, and the “N” register provides the Pixel PLL with the reference divide count.

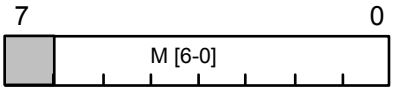
SYSCLK PLL Programming – Standard Mode

SYSCLK N



Index:	0x0015
Access:	Read/Write
Power on Value:	0x08
Bits 7 - 6	Reserved
Bits 5 - 0	N

SYSCLK M



Index:	0x0016
Access:	Read/Write
Power on Value:	0x41
Bit 7	Reserved
Bits 6 - 0	M

SYSCLK P

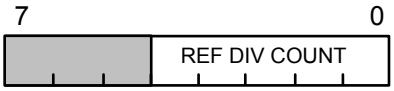
Index:	0x0017
Access:	Read/Write
Power on Value:	Undefined
Bits 7 - 3	Reserved
Bits 2 - 0	P

When the standard programming method is chosen for the SYSCLK PLL (System Clock Control register, PROG MODE bit = '1'), the registers at 0x0015 - 0x0018 provide the N, M, and P programming values. The valid ranges of these bits and their use are described above.

Note that, following a reset, the SYSCLK PLL is set to use compatibility programming using power on values appropriate for that mode. Thus, when switched to standard programming, the power on values found in the SYSCLK N and SYSCLK M registers are actually "left over" values that are not intended for use with the standard programming method.

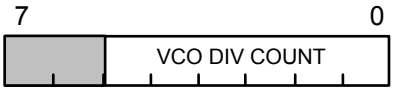
System Clock Frequency Selection –Compatibility Mode

System PLL Reference Divider



Index: 0x0015
Access: Read/Write
Power on Value: 0x08
Bits 7 - 5 Reserved
Bits 4 - 0 REF DIV COUNT - Reference Divide Count

System PLL VCO Divider



Index: 0x0016
Access: Read/Write
Power on Value: 0x41
Bits 7 - 6 DF - Desired Frequency
Bits 5 - 0 VCO DIV COUNT - VCO Divide Count

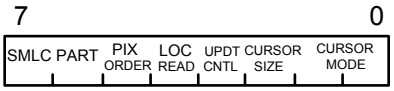
When the compatibility programming method is chosen for the SYSCLK PLL (System Clock Control register, PROG MODE bit = '0'), the registers at 0x0015 - 0x0016 provide the "Reference Divider", "VCO Divider" and DF programming values.

The valid ranges of these bits and their use are described in the section, "PLL Compatibility Programming", below.

The registers at 0x0017 and 0x0018 are not used. Compatibility programming is the default mode when the product is reset. The power on values will cause the PLL to produce a SYSCLK frequency of approximately 30 MHz (29.53 MHz) if the incoming REFCLK is the commonly available 14.31818 MHz frequency.

Cursor

Cursor Control



Index: 0x0030
Access: Read/Write
Power on Value: 0x00

Bits 7 - 6 SMLC PART - Small Cursor Partition. Selects 1 of 4 partitions within the cursor array to use for the 32x32 cursor. No effect when the cursor size is 64x64.
00 0x0100 - 0x01ff
01 0x0200 - 0x02ff
10 0x0300 - 0x03ff
11 0x0400 - 0x04ff

Bits 5 PIX ORDR. Specifies ordering of pixel bits in the bytes of the cursor array.
0 Right-to-left
1 Left-to-right

Bit 4 LOC READ - Read-back Value. Specifies the value obtained by reads of the Cursor X Low, Cursor X High, Cursor Y Low, and Cursor Y High registers.
0 Written Value - the value last written.
1 Actual Location - the location presently used for display. This will be different than the written value if a location register has been written but the location has not yet been updated. Following a cursor location update the "Written Value" and the "Actual Location" will be the same.

Bit 3 UPDT CNTL - Cursor Location Update Control. Controls when Cursor Location registers are sampled to change the cursor position.
0 Delayed - A write to the Cursor Y High register arms the circuitry for the update. The position is then updated (the cursor moves to the new location) when a vertical blanking period is detected.
1 Immediate - Move the cursor immediately following a write to any of Cursor X Low, Cursor X High, Cursor Y Low, or Cursor Y High.

Bit 2 Cursor Size
0 32x32
1 64x64

Bits 1 - 0 Cursor Mode
Standard Cursor (ACA ENAB = '0' in Advanced Cursor Control register):
00 OFF
01 ON - Mode 0 (3 colors)
10 ON - Mode 1 (2 colors and highlighting)
11 ON - Mode 2 (2 colors)
Advanced Cursor (ACA ENAB = '1' in Advanced Cursor Control register):
00 OFF
01, 1x ON

When the Advanced Cursor is enabled the cursor display modes are set in the Advanced Cursor Attribute register.

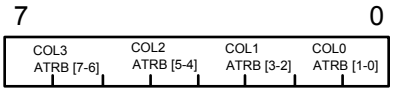
Advanced Cursor Control

Index: 0x0037
Access: Read/Write
Power on Value: 0x00
Bits 7 - 1 Reserved

Bit 0 ACA ENAB - Advanced Cursor Attribute Enable. When the cursor is ON, determines how the cursor bits in the cursor array are interpreted.
0 Standard Cursor - use Cursor Mode bits in Cursor Control register.
1 Advanced Cursor – use Advanced Cursor Attribute register.

NOTE: This register has no effect when the cursor is OFF (Cursor Mode = '00').

Advanced Cursor Attribute



Index: 0x0038
Access: Read/Write
Power on Value: Undefined
Bits 7 - 6 COL3 ATRB - Cursor Color 3 Attribute

Bits 5 - 4 COL2 ATRB - Cursor Color 2 Attribute

Bits 3 - 2 COL1 ATRB - Cursor Color 1 Attribute

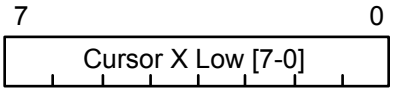
Bits 1 - 0 COL0 ATRB - Cursor Color 0 Attribute

When the cursor is ON (Cursor Control register, Cursor Mode is not '00'), and the Advanced Cursor is enabled (Advanced Cursor Control register, ACA ENAB = '1'), then the attributes in this register control the interpretation of the cursor pixel bits in the cursor array. The interpretations specified by the Cursor Mode bits ('01', '10', and '11') are ignored.
For each of the four colors that can be selected by the cursor pixel bits (Color 3, Color 2, Color 1, Color 0), the COLx ATRB bits specify the display of that color as follows:

00	Transparent
01	Solid Color
10	Translucent
11	Highlighted

When the ACA ENAB bit is off this register has no effect.

Cursor X Low

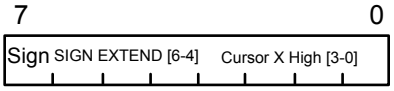


Index: 0x0031
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor X Low. The low order bits of the cursor X (horizontal) position.

A write to this register will update the cursor position:
1. When both the Cursor Y High register is written and vertical blanking is detected,
OR
2. Immediately
under control of the UPDT CNTL bit of the Cursor Control register.

The value read back:
1. Written Value,
OR
2. Actual Location
is controlled by the LOC READ bit of the Cursor Control register:

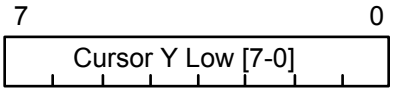
Cursor X High



Index: 0x0032
Access: Read/Write
Power on Value: Undefined
Bit 7 Sign
Bits 6 - 4 SIGN EXTND - Sign Extended.
These bits are always the same as bit 7. On a write these bits are discarded and replaced with the value written to bit 7. On a read they will return the same value as bit 7.
Bits 3 - 0 Cursor X High. The high order bits of the cursor X (horizontal) position.

Cursor X High and Cursor X Low form a combined register that holds a signed cursor X position in two's complement form. The X position range is -4096 to +4095. A write to this register will update the cursor position:
1. When both the Cursor Y High register is written and vertical blanking is detected,
OR
2. Immediately under control of the UPDT CNTL bit of the Cursor Control register
The value read back:
1. Written Value,
OR
2. Actual Location is controlled by the LOC READ bit of the Cursor Control register.

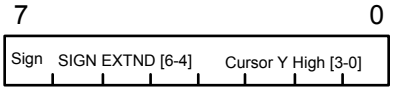
Cursor Y Low



Index: 0x0033
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Y Low. The low order bits of the cursor Y (vertical) position.

A write to this register will update the cursor position:
1. When both the Cursor Y High register is written and vertical blanking is detected,
OR
2. Immediately under control of the UPDT CNTL bit of the Cursor Control register.
3.
The value read back:
1. Written Value,
OR
2. Actual Location is controlled by the LOC READ bit of the Cursor Control register.

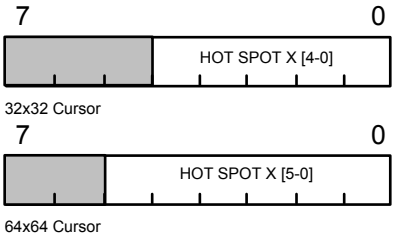
Cursor Y High



Index: 0x0034
Access: Read/Write
Power on Value: Undefined
Bit 7 Sign
Bits 6 - 4 SIGN EXTND - Sign Extended.
These bits are always the same as bit 7. On a write these bits are discarded and replaced with the value written to bit 7. On a read they will return the same value as bit 7.
Bits 3 - 0 Cursor Y High. The high order bits of the cursor Y (vertical) position.

Cursor Y High and Cursor Y Low form a combined register that holds a signed cursor Y position in two's complement form. The Y position range is -4096 to +4095. A write to this register will update the cursor position:
1. When vertical blanking is detected,
OR
2. Immediately under control of the UPDT CNTL bit of the Cursor Control register.
The value read back:
1. Written Value,
OR
2. Actual Location is controlled by the LOC READ bit of the Cursor Control register.

Cursor Hot Spot X



Index: 0x0035
Access: Read/Write
Power on Value: Undefined

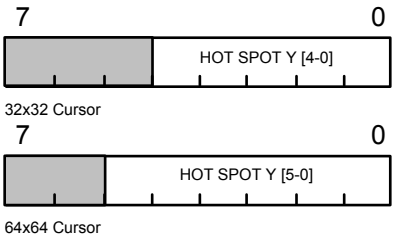
32x32 Cursor

Bits 7 - 5 Reserved
Bits 4 - 0 HOT SPOT X. Specifies which pixel in a cursor row is the X position pixel.

64x64 Cursor

Bits 7 - 6 Reserved
Bits 5 - 0 HOT SPOT X. Specifies which pixel in a cursor row is the X position pixel.

Cursor Hot Spot Y



Index: 0x0036
Access: Read/Write
Power on Value: Undefined

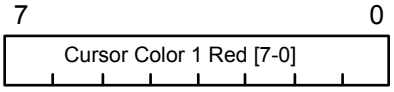
32x32 Cursor

Bits 7 - 5 Reserved
Bits 4 - 0 HOT SPOT Y. Specifies which pixel in a cursor row is the Y position pixel.

64x64 Cursor

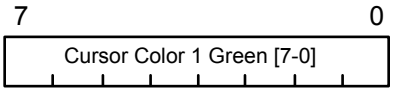
Bits 7 - 6 Reserved
Bits 5 - 0 HOT SPOT Y. Specifies which pixel in a cursor row is the Y position pixel.

Cursor Color 1 Red



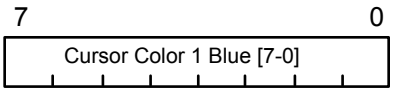
Index: 0x0040
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 1 Red

Cursor Color 1 Green



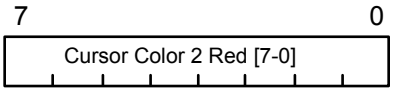
Index: 0x0041
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 1 Green

Cursor Color 1 Blue



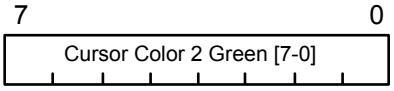
Index: 0x0042
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 1 Blue

Cursor Color 2 Red



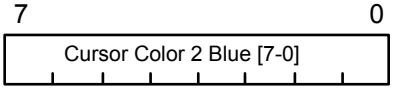
Index: 0x0043
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 2 Red

Cursor Color 2 Green



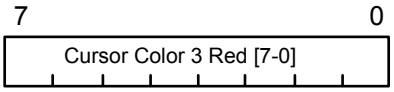
Index: 0x0044
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 2 Green

Cursor Color 2 Blue



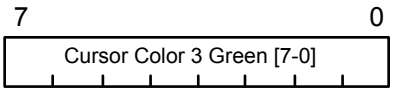
Index: 0x0045
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 2 Blue

Cursor Color 3 Red



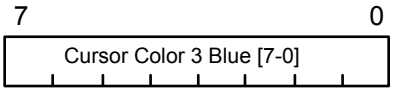
Index: 0x0046
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 3 Red

Cursor Color 3 Green



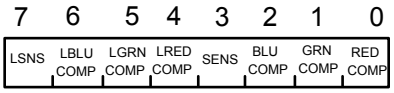
Index: 0x0047
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 3 Green

Cursor Color 3 Blue



Index: 0x0048
Access: Read/Write
Power on Value: Undefined
Bits 7 - 0 Cursor Color 3 Blue

DAC Sense



Index: 0x0082
Access: Read Only
Power on Value: Undefined
Bit 7 LSNS - Latched Sense
Bit 6 LBLU COMP - Latched Blue DAC Comparator Output
Bit 5 LGRN COMP - Latched Green DAC Comparator Output
Bit 4 LRED COMP - Latched Red DAC Comparator Output
Bit 3 SENS - Sense
Bit 2 BLU COMP - Blue DAC Comparator Output
Bit 1 GRN COMP - Green DAC Comparator Output
Bit 0 RED COMP - Red DAC Comparator Output

Bits 2,1,0 are the outputs of the three DAC reference comparators. The DAC output voltages are compared against the 0.35 V internal reference voltage (presented on COMPVREF). These bits are the “raw” outputs of the comparators.

Bits 6,5,4 are latched copies of bits 2,1,0. The latches are clocked during active line time (when BLANK is high).

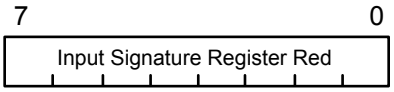
Bit 3 (Sense) represents the combined status of bits 2,1,0. If any of these bits is 0, bit 3 will be 0.

Bit 7 (Latched Sense) represents the combined status of bits 6,5,4. If any of these bits is 0, bit 7 will be 0.

Either bit 3 or bit 7 will be presented on the SENSE output, depending on the SENS SEL bit of the Miscellaneous Control 1 register.

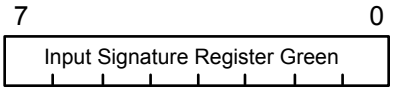
If the selected bit is 0, SENSE will be low.
If the selected bit is 1, SENSE will be high.

SR Red



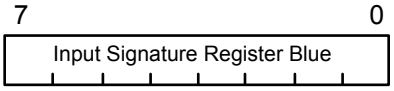
Index: 0x0084
Access: Read Only
Power on Value: Undefined
Bits 7 - 0 Input Signature Register Red
This register along with SR GREEN and SR BLUE is used to accumulate a diagnostic signature on the values presented to the DACs. The input to the Red DAC is the parallel data input to this portion of the MISR.

SR Green



Index: 0x0086
Access: Read Only
Power on Value: Undefined
Bits 7 - 0 Input Signature Register Green
This register along with SR RED and SR BLUE is used to accumulate a diagnostic signature on the values presented to the DACs. The input to the Green DAC is the parallel data input to this portion of the SR.

SR Blue

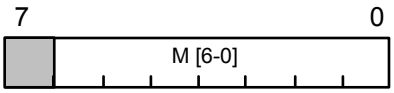


Index: 0x0088
Access: Read Only
Power on Value: Undefined
Bits 7 - 0 Input Signature Register Blue
This register along with SR RED and SR GREEN is used to accumulate a diagnostic signature on the values presented to the DACs. The input to the Blue DAC is the parallel data input to this portion of the SR.
Note: The reset, accumulation, and hold function of the SR are controlled by the SR CNTL bit of the Miscellaneous Control 1 register, and the BLANK input. See “Diagnostic Support” in this chapter for more information.

Pixel P Input

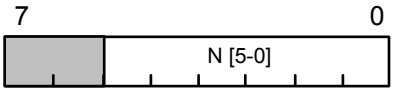
Index: 0x008c
Access: Read Only
Power on Value: Undefined
Bits 7 - 3 Reserved
Bits 2 - 0 P

Pixel M Input



Index: 0x008e
Access: Read Only
Power on Value: 0x05 (FS[1:0] = 00)
0x0e (FS[1:0] = 01)
0x00 (FS[1:0] = 10 or 11)
Bit 7 Reserved
Bits 6 - 0 M

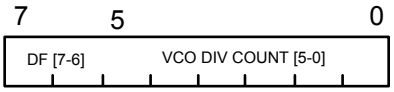
Pixel N Input



Index: 0x008f
Access: Read Only
Power on Value: 0x05
Bits 7 - 6 Reserved
Bits 5 - 0 N

These four registers allow readback of the selected programming values presented to the Pixel PLL. When the standard programming method is used (Pixel PLL Control 1 register, EXT/INT bits = '100' or '101') the above representations are valid.
When compatibility programming is used (EXT/INT bits = '001' or '011') the contents of registers at 0x008c and 0x008d are undefined, and the representation of the registers at 0x008e and 0x008f are given in the next two register descriptions.

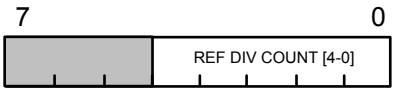
PLL VCO Divider Input



Index: 0x008e
Access: Read Only
Power on Value: 0x05 (FS[1:0] = 00)
0x0e (FS[1:0] = 01)
0x00 (FS[1:0] = 10 or 11)
Bits 7 - 6 DF - Desired Frequency
Bits 5 - 0 VCO DIV COUNT - VCO Divide Count

This register allows readback of the selected PLL VCO divider input. It is one of these registers:
M0, M1, M2, M3
as determined by the PLL Control 1 EXT/INT bits (2:0), the inputs FS[1:0], and PLL Control 2 INT FS bits (1:0).

PLL Reference Divider Input



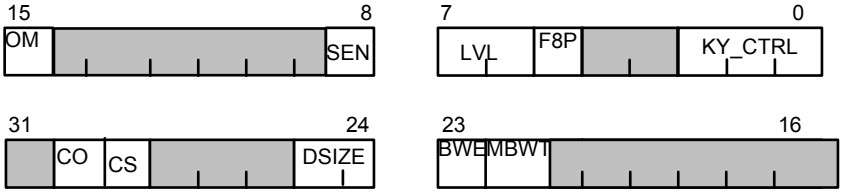
Index: 0x008f
Access: Read Only
Power on Value: 0x05
Bits 7 - 5 Reserved
Bits 4 - 0 REF DIV COUNT - Reference Divide Count

This register allows readback of the input to the PLL reference divider.
N0, N1, N2, N3
as determined by the PLL Control 1 EXT/INT bits (2:0), the inputs FS[1:0], and PLL Control 2 INT FS bits (1:0).
The above register contents are valid when the compatibility programming mode is used (Pixel PLL Control 1 register, EXT/INT bits = ‘001’ or ‘011’).

Appendix A: *Theory of Operation*

Buffer Control

Each linear memory window has a buffer control register (MW0_CTRL, MW1_CTRL) and the drawing engine has a buffer control register (BUF_CTRL). The parameters in a specific buffer control register apply only to the resource associated with that register. For example, MW0_CTRL has no effect on how the drawing engine or the other memory window access the buffer.
The buffer control registers for a linear window and a drawing engine are slightly different, but the drawing engine buffer control register shown below will illustrate all possible buffer control functions.



The **Source Enable** field (SEN) determines whether the internal 2D cache or the local buffer is the source of data during a read cycle.
The **CO** bit is useful when consecutive commands within a group of commands are using the cache as the source of data. To use the CO bit, one programs the appropriate drawing engine registers, loads the cache with data, and then triggers the command by writing XY1. The drawing engine will execute the command without interruption. This technique is useful when one wants to partition the cache into halves. The drawing engine can be executing from one half of the cache while new data for the next command is written into the other half of the cache. The CO bit essentially indicates to the drawing engine that the cache will always contain valid data. CO is not reset by the drawing engine.
The **CS** bit is used to select writing into the 2D Drawing Engine Cache or the User Defined Fog Table.

Linear Memory Windows Operation

A Linear Memory Window is defined as a region of system memory that is mapped Borealis local memory space. Borealis supports two memory windows. The address in system memory space to be mapped is programmed into MWn_AD, where n=0 or 1 for memory window 0 or memory window 1. The size of the memory window is programmed into MWn_SZ. Memory windows may be programmed from 4 Kbytes to 32 Mbytes. A memory window must begin on an address boundary equal to its size. For example, a 4 Kbyte window can be start at any 4 Kbyte address while a 1 Mbyte window must begin on a 1 megabyte boundary.

The memory window can be mapped to any of Borealis's local buffers as described in the previous section. The address in the local buffer to which the memory window is mapped is determined by the MWn_ORG and MWn_PGE registers. The MWn_ORG register determines the starting address of the memory window in local memory space. The MWn_PGE register allows the value in MWn_ORG to be offset from 0 to 31 megabytes. A linear memory window is enabled by setting the appropriate window enable bit (EW0 or EW1) in the CONFIG1 register.

An example of how the local buffer address is calculated for a 4 Kbyte window is shown below. As can be seen from the diagram, host address bits [31:12] are compared with bits [31:12] of the MWn_AD register. If all bits are a match, this access is considered a window hit. The Display buffer address is then generated by adding the DVPGE register to MWn_ORG[24:19] and replacing the lower twelve bits of the result with the lower twelve bits of the host address.

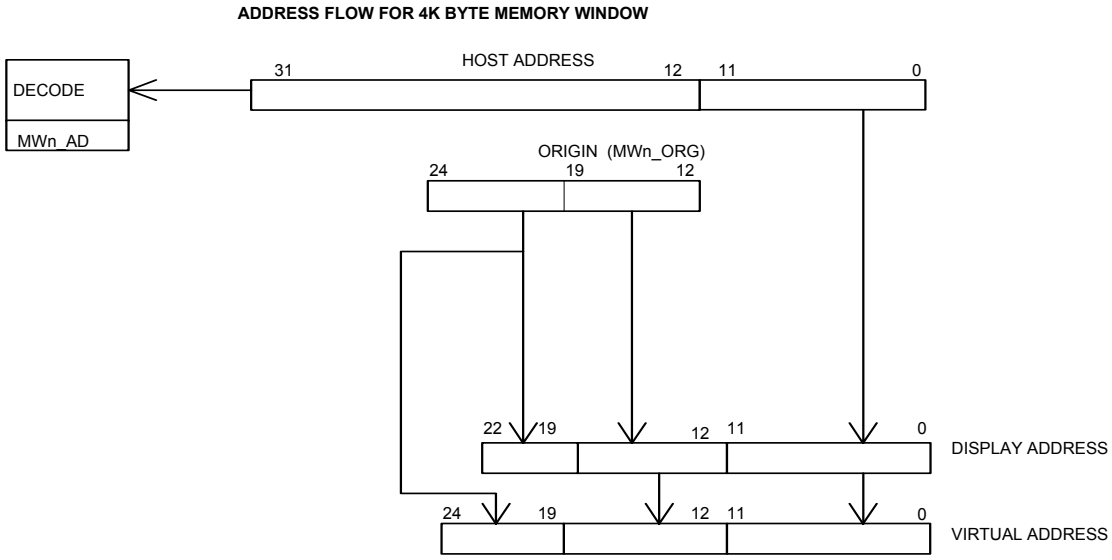


Figure 2-1.1. Address Flow for 4K Byte Memory Window

Control of the Command Pipeline

Control of the command pipeline is a three step process.

Step 1. Ensure that Borealis is ready to accept new data.

Step 2. Update registers.

Step 3. Trigger New command.

Borealis employs extensive pipelining to smooth the process of loading new commands and parameters. When a command begins execution, all of the required parameters are transferred from the host accessible registers to internal working registers. When the parameters have been transferred to internal registers, Borealis is ready to accept a new command and parameters into its host accessible registers.
The command is queued for execution once the command trigger register (XY1) or the 3D_TRIG register (for 3D lines and triangles) has been written. This register must be the last one written in any command sequence. It is important to note that a command will not be triggered unless the most significant byte of XY1 or 3D_TRIG has been written. As soon as the trigger is written, Borealis will acknowledge by asserting the BUSY signal. A command that is ready for execution must still wait for a previous command to complete execution before it will actually start. During this waiting period, BUSY will remain set.
There are five additional bits that may be monitored to control the command pipeline.

Drawing Engine Busy (DEB- FLOW[0]) and Memory Controller Busy (MCB - FLOW[1]) provide additional visibility into the internal state of Borealis. The Drawing Engine asserts DEB when a command is in its Execution Pipe. The Drawing Engine Busy bit gets de-asserted when no commands are currently in the Execution Pipe, *but* may still be in the Renderer Pipeline or Pixel Cache (see FLOW[4]). The memory controller will assert MCB when it is currently running a memory cycle or has a request for a memory cycle in its queue. Note that refresh or display transfer cycles will not cause MCB to be asserted, but indirectly they will cause MCB to be held if there are normal memory requests in the memory controller queue. The MCB bit is very useful in determining the integrity of data to be read back from the local buffers.

CLP (FLOW[2]) will indicate that the command that just completed did so as the result of a clipping condition (i.e. stop on clip boundary). The CLP bit is cleared every time a new command begins execution, so it is important not to pipeline commands if the status of CLP is to be relied on.

PRV (FLOW[3])=1 indicates that the previously triggered command is still executing. It is identical to BUSY, **except** that PRV will not be de-asserted until the drawing engine has completed the *execution* of the previously triggered command and after the **first instance** the memory controller goes not busy for the previously triggered command. PRV is useful when operating the drawing engine in "dual 2D cache mode".
RPB(FLOW[4])=1 indicates that the Renderer Pipeline and/or Pixel Cache is busy.

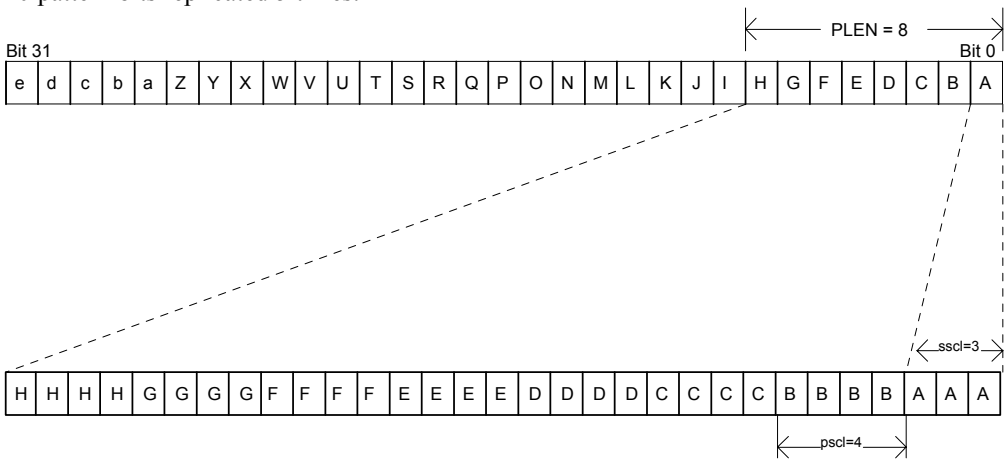
NOTE: This bit in conjunction with DEB and MCB must be read to determine if Rendering Command (e.g. 3D Triangle) has completed. It is entirely possible for DEB and MCB to both de-assert, but RPB to be asserted, thus signifying that the Rendering Command hasn't completed yet.

Draw Style and Patterning

LINE STYLE	SOLID	TRNSP	LPAT BIT	RESULT
Line solid	1	X	X	Foreground
Line on off dash	0	1	0	Destination
Line on off dash	0	1	1	Foreground
Line double dash	0	0	0	Background
Line double dash	0	0	1	Foreground

The LPAT bit is the corresponding bit in the pattern register. In addition, the dash members may be controlled by the settings in the pattern register (LPAT) and the pattern control register (PCTRL). The LPAT register is a simple 32 bit stipple pattern that selects foreground, background or transparent depending whether each bit is set or cleared and the state of the TRNP bit. PCTRL allows for scaling and cropping of the LPAT bit pattern.

The scaling factor is specified in the PSCL register and will cause each bit in the pattern register to be scaled from one to eight times. The SSCL register specifies the scale to be applied to the first bit in the pattern this value must be less than PSCL. The PLEN register specifies the pattern length, starting with LPAT[0]. Note that PLEN specifies pattern bits and not pixels drawn. For example, if PLEN is 10 (decimal) and PSCL is 5 (decimal) the total number of pixels drawn would be 50 (decimal) with each of the 10 pattern bits replicated 5 times.



The PATRN register provides additional information as to how to start and end. The NLST (no last) bit will cause the line algorithm not to draw the last pixel in the line, and not increment the line pattern for that pixel. This should be used in the drawing of poly lines where the end point of one line segment is common with the start point of the next line segment.

The PRST (pattern reset) bit will cause Borealis to initialize the pattern at the start of each new LINE command. If PRST is not set, the pattern will be continuous from line to line.

For BITBLT and TRIAN, the APAT bit in the PATRN register will cause the source to be a 32x32 tile of pixels or stipple pattern if stipple is set. This pattern is locked to the screen, that is to say two abutting triangles or rectangles will have matching patterns at their edges.

Fill Style and Patterning

FILL STYLE	SOLID	TRNSP	STPL	STIPPLE BIT	RESULT
Fill Solid	1	X	X	n/a	fore op dst
Fill Tiled	0	0	0	n/a	src op dst
Fill Stippled	0	1	1	0	destination
Fill Stippled	0	1	1	1	fore op dst
Fill Opaque Stippled	0	0	1	0	back op dst
Fill Opaque Stippled	0	0	1	1	fore op dst

The transparency function is only meaningful when a stipple operation is being executed. Transparency should be set to zero at all other times.

Display List Processor

Borealis contains a Display List Processor (DLP) which can read drawing engine commands from memory and execute them. The DLP can accept four distinct formats, XY format (format 1), REG3 format (format 0), DMA format, or text mode. The format is selected when writing the end address, however a given format must be held constant though an entire list. Therefore list formats cannot change until after writing a new start address, or the list reaches an explicit stop point, and a new end address is written with the new format. It is very important that software maintain coherency by abiding by the above rules or the display list can have unpredictable results.

The XY Format is suitable for applications where only XY0 through 3 need to be accessed. The REG3 format provides the most flexibility where individual registers need to be accessed. The DMA format can start multiple DMA requests over AGP. Finally, the text mode provides a very efficient means of caching glyphs and drawing them rapidly to the screen.

Selecting format 0 specifies the Register/DMA mode. Setting bits 25:24 to 0 specifies the Non-DMA register format. Three registers can be specified by writing the lower 8 bits of the drawing engine registers into the first 3 bytes of the command. The upper 8^{th} bit (or bank select) for each of the registers is located in bits 28-30 respectively. Bit 31 specifies waiting for vertical blank before executing the list. Bits 32 through 27 specify the register values to be written in the three registers, and bits 26-27 select the number of registers to be written (1-3). Registers are always written from A-C, no gaps allowed.

Setting bits 25:24 to a 01 selects the DMA mode. This mode may be used to request Multiple AGP DMA transfers. If bit 31 is set, then the DLP will wait after the current AGP request until it is complete.
Setting bit 25 to a 1 selects Text mode. In text mode, up to two glyphs can be specified, plus their destination addresses. The glyphs each have their own text table entry which provides information about the glyph to enhance rapid accessing. Software must set up all registers not accessed by the DMA, but needed for writing text, prior to executing this command. REG3 mode can be used to do this.

Any version of format 0 are allowed to be mixed within a single list.
After the data is ready for display list execution, the application must set the start and end addresses. The display list start address is loaded into DL_ADR. The display list end address is loaded into DL_CNTRL and the stop bit is set to 0. This triggers execution of the list. DL_FMT selects the format, and DL_SVD selects DMA mode. These three buffers must follow the following rules to maintain coherency:

1. They can never be changed while a list is executing.
2. They can only be changed after a start address is written (for the new list), or only after a list has become idle and a new end address is written.

The DLP maintains two independent lists, an active list and pending list. The active list is the currently running list which is triggered by a start address and end address pair. It is possible to append to the list by writing to just the end address. This allows dynamic list building.
Whenever a list is running, a new start address can be written which will change the internal pointer to the pending list. The old list will continue executing, but all subsequent end addresses will now be associated with the pending lists start address. The pending list can be treated just like the primary list.
If a third start address is written, retries will be issued until the pending list becomes the active list and there is room for a new pending list.

NOTE: The stop bit will stop all lists, including any pending lists and cause the lists to be inaccessible.

Texel Cache

The Borealis is equipped with an advanced 8KB texel cache (TC) which greatly improves texture mapping and video applications. The cache can handle several palette and non-palette texture formats with sizes up to 512x512. Mipmapping is supported up to 10 levels of detail. The TC is capable of generating 4 texels/clock for 1 clock cycle bilinear interpolation. It also employs an advanced look ahead algorithm for minimizing memory reads.

TC Sizes

The texture cache is capable of handling textures (or video frames) of up to 512 x 512 with the following restrictions. Textures must be a power of 2 in both X and Y directions. X and Y sizes are independent, i.e. 2 x 64, 128 x 32, etc.. are allowed.
The cache is optimized for an efficient use of space, however the following formats (or smaller) work best as they will fit directly into the cache:

TEXTURE FORMAT (bits per textured pixel (bpt))	TEXTURE SIZE (XxY)
32 bpt	32x64
16 bpt	64x64
8 bpt (palettized or direct)	128x64
4 bpt (palette)	256x64
2 bpt (palette)	512x64
1 bpt (palette)	512x64

Using textures of the above size or less will result in the maximum number of hits, minimizing misses and greatly improving cache performance.

Texture Formats

Please see the register definition section for TEX_CTRL.
Each palette entry is in the following format: ARGB where A is most significant.
Note: For TSIZE of 3E and 3F, the intent is to allow bilinear or trilinear filtering for a palettized 8bpt 8888 texture source. This is accomplished by T2R IV autoconverting the texture source of format 8888 to either 4444 or 0565 with a specified TSIZE of 3E or 3F, respectively, into the on-chip palette during the palette load command.

Special Commands

There are two commands associated with the texture cache, LD_PAL and INV_TEX.
The LD_PAL command will load a texture palette with the format specified in TEX_CTRL into the Ticket to Ride's Internal Texture Cache Palette:

1) Specify the TPAL_ORG (Origin Address of the Palette to be loaded).
2) Specify in TEX_CTRL the palette format (TEX_CTRL[29:24]…TSIZE).
3) Specify LD_PAL command (Set CMD_OPC = 0xB).
4) Trigger this texture palette load command with the 2D Trigger register, XY1.

Note: The TSIZE will determine the amount of data to be loaded into the texture palette.

The INV_TEX command instructs the texture cache that the next triangle coming through is using a different texture and/or texture palette than the previous command. In essence, it invalidates the texture tags, thereby invalidating the current texture and/or texture palette. The TEX_CTRL register must be set prior to issuing this command, e.g. Texture Cache Tile Mode (TEX_CTRL[30]) must be set prior to issuing the INV_TEX command.

Alpha Blending

The Ticket to Ride implements full OpenGL compatible alpha blending. The blending function can be applied to any Drawing Engine command. The control of blending is performed in the ACNTRL register.
To properly set up a blending command, the following must be done. First, select a source and destination blending function. These are defined in the lower byte of the ACNTRL register. Second, if you are in a mode which doesn't support alpha or are going to use the source or destination alpha registers, you must load them into the lower two bytes of the ALPHA register. If you are in a mode which supports alpha and you want to use the constant registers, you must set the SRE and DRE bits to override the default settings. Finally, the blending enable bit must be set to activate blending.

Programming Borealis 3D Operations

The Borealis is a vertex-based 3D polygon accelerator. It performs the setup for 3D triangles, lines and points in hardware.
There are 2 3D Rendering Commands:

- TRIAN_3D
- LINE_3D

NOTE: LINE_3D can be used to render 3D points. Texture Mapped 3D Lines are not supported.

There are up to three steps to take to program a 3D operation:

1. Control and Mode Setup
2. Parameter Load
3. Trigger

Control and Mode Setup

a)	BUF_CTRL	<25:24>	Destination Pixel Size and Format
b)	DE_SORG:		Source Origin
c)	DE_DORG:		Destination Origin
d)	DE_TPTCH:		Texture Pitch of LOD0
e)	DE_ZPTCH:		Z Pitch
f)	DE_SPTCH:		Source Pitch
g)	DE_DPTCH:		Destination Pitch
h)	CMD:		Command
i)	FORE:		Foreground Color
j)	BACK:		Background Color
k)	MASK:	0xFFFFFFFF	Pixel Plane Mask
l)	DE_KEY:		Color Key
m)	LPAT:		Line Pattern (For 3D Lines)
n)	PCTRL:		Line Pattern Control
o)	DE_ZORG:		Z-Buffer Origin
p)	LOD0_ORG:		Texture Mipmap Level 0 Origin
q)	LOD1_ORG:		Texture Mipmap Level 1 Origin
r)	LOD2_ORG:		Texture Mipmap Level 2 Origin
s)	LOD3_ORG:		Texture Mipmap Level 3 Origin
t)	LOD4_ORG:		Texture Mipmap Level 4 Origin
u)	LOD5_ORG:		Texture Mipmap Level 5 Origin
v)	LOD6_ORG:		Texture Mipmap Level 6 Origin
w)	LOD7_ORG:		Texture Mipmap Level 7 Origin
x)	LOD8_ORG:		Texture Mipmap Level 8 Origin
y)	LOD9_ORG:		Texture Mipmap Level 9 Origin
z)	HITH:		Hither Value (For Z-Buffer Enabled)
aa)	YON:		YON Value (For Z-Buffer Enabled)
bb)	FOG_COL:		Fog Color (For Fog Mode)
cc)	ALPHA:	<23:16>	Alpha Test Value (For Alpha Compare)
dd)	TEX_BORDER:		Texture Border Color
ee)	KEY_3D_LOW:		3D Key Color Low Range
ff)	KEY_3D_HI:		3D Key Color High Range
gg)	A_CNTRL:		Alpha Control Register
hh)	3D_CNTRL:		3D Control Register
ii)	TEX_CNTRL:		Texture Map Control Register

Parameter Load

a) CP0: Pattern Pointer
b) CP1: Vertex 0 X
c) CP2: Vertex 0 Y
d) CP3: Vertex 0 Z
e) CP4: Vertex 0 W
f) CP5: Vertex 0 Color {A,R,G,B}
g) CP6: Vertex 0 Specular {F,Rs,Gs,Bs}
h) CP7: Vertex 0 U
i) CP8: Vertex 0 V
j) CP9: Vertex 1 X
k) CP10: Vertex 1 Y
l) CP11: Vertex 1 Z
m) CP12: Vertex 1 W
n) CP13: Vertex 1 Color {A,R,G,B}
o) CP14: Vertex 1 Specular {F,Rs,Gs,Bs}
p) CP15: Vertex 1 U
q) CP16: Vertex 1 V
r) CP17: Vertex 2 X
s) CP18: Vertex 2 Y
t) CP19: Vertex 2 Z
u) CP20: Vertex 2 W
v) CP21: Vertex 2 Color {A,R,G,B}
w) CP22: Vertex 2 Specular {F,Rs,Gs,Bs}
x) CP23: Vertex 2 U
y) CP24: Vertex 2 V

NOTE: The three sets of parameters (CP1-CP8), (CP9-CP16) and (CP17-CP24) represent the parameters for the three vertices of a 3D triangle. For the 3D Line or Point, (CP9-CP16) and (CP17-CP24) represent the parameters for the two endpoints. Since texture mapped lines are not supported, the (w,u,v) parameters for 3D Line or Point commands are meaningless.

Trigger

To trigger the 3D rendering command, write into the 3D_TRIGGER register. This register needs only to be written. No data needs to be supplied.

3D Operational Modes and Control

For 3D triangle commands, the minimal set of CP parameters you need to set are the x,y coordinate pairs for the three vertices (CP1, CP2), (CP9, CP10), and (CP17, CP18). For 3D line commands, the minimal set of CP parameters you need to set are the x,y coordinate pairs for the two endpoints (CP9, CP10), and (CP17, CP18).

Gouraud Shading Mode

- To enable Gouraud Shading, set ***3D_CNTRL<24>*** = 1.
- Must disable SOLID mode by setting ***CMD<16>*** = 0.
- Set the color values for the three vertices of the triangle (CP5, CP13, CP21), OR- set the color values for the two endpoints of the line (CP13, CP21).

Specular Highlighting Mode

- To enable Specular Highlighting, set ***3D_CNTRL<25>*** = 1.
- Set the Specular color values for the three vertices of the triangle (CP6, CP14, CP22), -OR- set the Specular color values for the two endpoints of the line (CP14, CP22).

Z Buffer Mode

- To enable Z-Buffering, set ***3D_CNTRL<0>*** = 1. If disabled, no z-buffer hidden surface removal or z-buffer updates will occur.
- Z Scale Mode: Silver Hammer supports 16bpp and 24 bpp Z. It accepts either unscaled or scaled Z (normalized to the range of 0 to 1). If the Z Scale Mode bit (***3D_CNTRL<30> = 1***) is set, then the normalized Z value will be scaled to either 2^16 in 16bpp destination pixel mode or 2^24 in 32bpp destination pixel mode.
- At 32bpp, the Z Values in the Z Buffer are loaded in as DWORDS (32 bits), but are loaded in the lower 3 bytes ([23:0]) of each DWORD. The upper byte, 4th byte, of each DWORD must be set to ZERO.
- If Z compares and Z-Buffer updating are desired, disable Read_Only_Z (***3D_CNTRL<1>*** = 0).
- If Read_Only_Z is enabled, then Z compares would occur, but the Z-Buffer would not be updated.

Dest Pixel Size	Clamping	Setup	Z Size	Dest Zsize
16bpp	*16bpp*	Yes	16bpp	16bpp
32bpp	*24bpp*	Yes	24bpp	24bpp

- Z Compare Operators: (***3D_CNTRL<7:5>***) See *Chapter 5, Register Sets* for the table of Z compare operators used in Z-Buffer hidden surface removal operation.

NOTE: If Z Compare Operator is set to ALWAYS (***3D_CNTRL<7:5>*** = 0x1), performance will be enhanced by 40%. An example of an application that would utilize this performance enhancing feature would be Z Fills.

- Yon Compare Operators: (***3D_CNTRL<10:8>***) See 5.8.35 for the table of Yon compare operators used in Yon Z Clipping surface removal operation.
- Hither Compare Operators: (***3D_CNTRL<13:11>***) See 5.8.35 for the table of Hither compare operators used in Hither Z Clipping surface removal operation.
- Set the z values for the three vertices of the triangle (CP3, CP11, CP19), -OR- set the z values for the two endpoints of the line (CP11, CP19).

Fog Mode

To enable Fog, set 3D_CNTRL<27> = 1
*Result_Color = Color * Fog_factor_percentage + Fog_Color * (1 - Fog_factor_percentage)*
The Fog_factor_percentage (or just fog_factor) can either come from the vertices on the polygon or from a user-defined fog table. The user-defined table would be indexed by either the Z-value or the 1/w value of the drawn pixel.

Vertex Fog Mode

(3D_CNTRL[4] = 0)
Set the vertex-based fog_factor value for the three vertices of the triangle (CP6, CP14, CP22), -OR- set the fog_factor value for the two endpoints of the line (CP14, CP22).

Table Fog Mode

(3D_CNTRL[4] = 1)
The fog_factor values for the Fog Table are loaded via the host bus by setting the CS bit in BUF_CNTRL[29]. There are 65 entries in the table, each one byte long. Fog Tables can be indexed by either the per-pixel 1/w or z value. The index range from 0 to 2. Index of 0 will index the 1st entry and index of 2.0 will index the 65th entry. Fog_Index_Select (3D_CNTRL[3]) selects between using 1/w (FIS=1) or using z (FIS=0).

Fog Enable	Fog Selector	Fog Index Select	Fog
0	x	x	No Fog
1	0	x	Vertex Fog
1	1	0	z Table Fog
1	1	1	1/w Table Fog

Alpha Modes and Control

Alpha Comparison

- To enable Pixel Alpha Compare, set A_CNTRL<19> = 1.
- Set the Alpha Test Value in ALPHA_REG<23:16>.
- The Pixel Alpha Compare Operators are set in A_CNTRL<18:16>. See 5.8.35 for the table of alpha compare operators used in Alpha Clipping.
- If the alpha value to be compared is originating from the vertices of the triangle or line, set the alpha values for the three vertices of the triangle (CP5, CP13, CP21), -OR- set the alpha values for the two endpoints of the line (CP13, CP21).

Alpha Select, Alpha Modulation and Alpha Blend Select

- Alpha Select (***A_CNTRL<24>***) will select alpha value from current texel if A_CNTRL<24> is set to 0 and texture mode (***TEX_CNTRL<0>*** = 1) is enabled. Otherwise, the alpha value will be selected from either the interpolated pixel value, foreground or background value, i.e. non_texel_alpha depending upon which pixel color mode is set.
- Enable Alpha Modulation (***A_CNTRL<25>*** = 1) will, with texture mode enabled, modulate the texel alpha value with non_texel_alpha.
  Resultant_Texel_Alpha = Pixel_Alpha * Texel_Alpha
- Enable Alpha Blend Select (OpenGL GL_BLEND mode) (***3D_CNTRL<17>*** = 1) will, with texture mode enabled, alpha blend the alpha value:
  Resultant_Alpha = Pixel_Alpha * (1 - Texel_Alpha) + OpenGL_Blend_Color_Alpha * Texel_Alpha

Alpha Select	Texture Mode	Alpha Modulation	Alpha_Blend_Sel	Alpha Value
1	x	x	x	*Gourard Shaded, Flat Shaded, or Bg/Fg (Line) Alpha*
0	1	0	0	*Texel Alpha*
0	1	0	1	*Alpha Blended Alpha*
0	1	1	0	*Alpha Modulated Alpha*
0	1	1	1	*Texel Alpha*

Decal Alpha Mode

Enabling Decal Alpha Mode (***A_CNTRL<26>*** = 1) would perform the following alpha blending function: **RGB = (Texel_RGB)*(Texel_Alpha) + (Non_Texel_RGB)*(1-Texel_Alpha).**

Rectangle Mode

Enabling Rectangle Mode (***3D_CNTRL<28>***) will render a rectangle with the triangle command, but the triangle must be specified as follows:

1. It is a flat top triangle.
2. Its spatial (x,y) vertex coordinates must be specified as follows:

V0	**V1**	**OR**	**V1**	**V0**
V2	**(V3)**		**(V3)**	**V2**

where VO(x,y) corresponds to (CP1,CP2), V1(x,y) corresponds to (CP9,CP10) and V2(x,y) corresponds to (CP17,CP18). The command will drop a vertical edge from the specified V1 point to (V3) to complete the rectangle, thus achieving a rendered rectangle on a triangle command.

Dither Mode

Dithering is used, among other things, to remove banding artifacts when displaying an image at a lower pixel depth, i.e. from 24bpp to 16bpp. To enable dithering, set (***3D_CNTRL<16> = 1***).

Backface Cull Mode

To enable backface cull mode, set (3D_CNTRL<23> = 1). With this mode turned on, Borealis will not render triangles that are facing the back. This is accomplished by performing a calculation on an ordered set of vertex spatial (x,y) coordinates. The order of the coordinates may be either clockwise or counter-clockwise and must to set by setting the CW/CCW Selector Bit for Culling (3D_CNTRL<22>). Setting this bit to zero denotes clockwise vertex ordering and setting this bit to one denotes counter-clockwise.

Texture Map Modes and Control

NOTE: Before using a new texture, the texel cache must be invalidated. After the TEX_CNTRL register is set up, and before the 3D Texture command is triggered, you must invalidate the texel cache as follows:

1. Set command opcode to 0xA.
2. Trigger this command by writing into XY1 register. Any value is okay because it is only used to trigger this command.

Non-MipMap Texture Map Mode Setup

- Enable Texture Map Mode by setting ***TEX_CNTRL<0>*** = 1.
- To enable Perspective Correction, set ***TEX_CNTRL<6>*** = 1. Otherwise, it would be orthogonal mode and the w values at the vertices are automatically set to 1.
- Set the perspective w and u,v texel coordinate values for the three vertices of the triangle V0(CP4, CP7, CP8), V1(CP12, CP15, CP16), and V2(CP20, CP23, CP24).
- The texture image size is specified by the MMSIZEX (***TEX_CNTRL<19:16>***) and MMSIZEY (***TEX_CNTRL<23:20>***) fields of the TEX_CNTRL register. The texture size must be a power of 2 and not necessarily the same power of 2, i.e. Borealis support power of 2 rectangular texture images.
- The origin of a non-mipmapped (see Mipmap Mode) texture is to be specified in the ***LOD0_ORG*** register. The LOD origin must be 128 bits or 16-byte aligned. The ***TPTCH*** register contains the pitch of LOD0_ORG.
- **NOTE**: Each vertex w must be written before its respective vertex u,v coordinate pairs.
- Disabling texture map mode disables all texture map modes.

MipMap Texture Map Mode Setup

- Enable Texture Map Mode by setting TEX_CNTRL<0> = 1.
- To enable Per-Pixel MipMap Mode, set TEX_CNTRL<1> = 1.
- To enable Perspective Correction, set TEX_CNTRL<6> = 1. Otherwise, it would be orthogonal mode and the w values at the vertices are automatically set to 1.
- Set the perspective w and u,v texel coordinate values for the three vertices of the triangle V0(CP4, CP7, CP8), V1(CP12, CP15, CP16), and V2(CP20, CP23, CP24).
- Ten levels of detail (LOD) Mipmapping is supported in Borealis and is on a per pixel basis. The origin, pitch and Mipmap XSIZE and YSIZE of the largest LOD is to be specified in LOD0_ORG,
- TPTCH, TEX_CNTRL<19:16> and TEX_CNTRL<23:20>, respectively. Subsequent LOD origins are to be specified in LOD1_ORG through LOD9_ORG and the number of mipmap levels are to be specified in TEX_CNTRL<15:12>. Only the origins of the specified number of mipmap levels are required to be specified in LOD0_ORG through LOD9_ORG.
- Borealis will support texture images up to a maximum size of 512x512 to a minimum size of 1x1. The width and height must be a power of 2 and they need not have to be equal, i.e. Borealis supports square and rectangular texture images. Each subsequent LOD mipmap must be half the size in both x and y dimensions until it reaches the size of 1x1. In the case of rectangular texture images, each subsequent mipmap x and y size will be halved until one of the dimension reaches 1. At that level, then the other dimension of each subsequent LOD mipmap gets halved until a 1x1 size is achieved. E.g.

LOD0	64 x 16
LOD1	32 x 8
LOD2	16 x 4
LOD3	8 x 2
LOD4	4 x 1
LOD5	2 x 1
LOD6	1 x 1

Texture Filtering, Lighting, and Coordinate Setup

- Borealis supports magnification and minification filtering, (***TEX_CNTRL<2> and TEX_CNTRL<4>)***, respectively. For Magnification Filtering, setting ***TEX_CNTRL<2> = 0,*** will enable bilinear filtering. Otherwise, it will perform nearest texel sampling. Similarly, setting ***TEX_CNTRL<4> = 0,*** will enable bilinear filtering and disabling this bit will give you nearest texel sampling for minification.
- Texture Format is specified in ***TEX_CNTRL<29:24>***. Please refer to Chapter 5 for formats supported.

NOTE: Texel formats other than 8888 will be expanded to 8888 before texture processing is performed.

For example, 1555 texel format will be expanded as follows:

alpha[0]	->	8{alpha[0]}
red[4:0]	->	{red[4:0], red[4:2]}
green[4:0]	->	{green[4:0], green[4:2]}
blue[4:0]	->	{blue[4:0], blue[4:2]}

- The vertex u,v coordinates may either be specified as a scaled number, normalized to be between 0 to 1 inclusive, or unscaled. To be in UV scale mode, set ***TEX_CNTRL<31> = 1.*** The uv values will be scaled to the Mipmap XSIZE and YSIZE, respectively.
- To enable RGB Modulation Mode, set ***TEX_CNTRL<5>*** = 1. This mode modulates the texture image with an interpolated rgb surface generated by the Gouraud shader. So, Gouraud Shading needs to be enabled by setting ***3D_CNTRL<24>*** = 1 and the three color parameters (or two, depending upon the rendering command) need to be set (see Gouraud shading)**.** For Example:

- Modulated_Texel_Red = Pixel_Red * Texel_Red
- Modulated_Texel_Green = Pixel_Green * Texel_Green
- Modulated_Texel_Blue = Pixel_Blue * Texel_Blue

- To enable Texture Wrap Mode, disable the U and V Texture Clamp Bits, ***TEX_CNTRL<8> = 0 and TEX_CNTRL<9> = 0***. To clamp in U or V coordinate space or both coordinate space, just enable each bit (or both) accordingly. In Texture Clamping, it could either be clamped to a border color specified in TBORD_COL register or to the color at the edge of the textured image. Setting the clamp selector to 1, ***TEX_CNTRL<7> = 1***, will select the Texture Border Color. Otherwise, it would clamp to the texel value at the edge of the texture image.
- Borealis supports Mipmap Linear Filtering. This mode will interpolate between the two nearest LOD mipmap for better texture image quality. It requires two passes to perform this feature.

Mipmap Linear Setup

Refer to section in this Appendix, entitled *MipMap Texture Map Mode Set Up* to set up a mipmap texture triangle command.

Pass 1:
1) Enable Mipmap Linear Mode by setting ***TEX_CNTRL<3> = 1.***
2) Set Mipmap Linear Pass 2 bit ***TEX_CNTRL<10> = 0.***
3) Set the fog color register (**FOG_COL**) to 0.
4) Enable Texture Composite Mode, ***3D_CNTRL<27> = 1***.
5) If Z buffer is enabled (***3D_CNTRL<0> = 1)***,
   enable Z Read Only Bit ***3D_CNTRL<1> = 1.***
6) Disable Alpha Blending Mode, ***A_CNTRL<10> = 0.***
7) Trigger the 3D command, i.e. write into 3D_TRIG register.

Pass 2:
1) Enable Mipmap Linear Mode by setting ***TEX_CNTRL<3> = 1.***
2) Set Mipmap Linear Pass 2 bit ***TEX_CNTRL<10> = 1.***
3) Set the fog color register (**FOG_COL**) to 0.
4) Enable Texture Composite Mode, ***3D_CNTRL<27> = 1***.

5) If Z buffer is enabled (***3D_CNTRL<0> = 1)***,
disable Z Read Only Bit ***3D_CNTRL<1> = 0*** unless Z Read Only Bit ***3D_CNTRL<1>*** was already set to 1 before pass 1.

Another way of interpreting how to set Z Read Only Bit is as follows:

Z_Read_Only = Z_Read_Only_Original | NOT Mipmap_Linear_Pass_2;

2) Enable Alpha Blending Mode, ***A_CNTRL<10> = 1.*** Set Blending Source and Destination Function to SRC_ONE and DST_ONE, respectively, i.e. set ***A_CNTRL<3:0> = 0x1 and A_CNTRL<7:4> = 0x1.***

3) Trigger the 3D command, i.e. write into 3D_TRIG register.

NOTE: Linear Mipmap Linear Mode, also known as Trilinear Mipmapping, is Mipmap Linear with the texture filtering set to bilinear interpolation. If texture filtering is set to Nearest Texel Sampling, then we would be in *Nearest* Mipmap Linear Mode.

- Texture Cache supports two cache configurations: Linear or Tile Mode. Setting the Texture Cache Tile Mode, ***TEX_CNTRL<30> = 1,*** will configure the cache lines to be a 1 page by 8 configuration. This mode would be better for non-horizontal texture mapping applications. In Texture Cache Linear Mode, ***TEX_CNTRL<30> = 0,*** cache will be configured as 8 pages by 1. This mode would be better for horizontal texture mapping applications, e.g. rectangular video clip acceleration.
- To enable 3D Texture Color Range Keying, set (***3D_CNTRL<15> = 1)***. To set the 3D High Color Key value, set KEY_3D_HI. Similarly, the low value would be set in KEY_3D_LOW. Borealis supports inclusive as well as exclusive color range keying. For inclusive color range keying, set the 3D Key Polarity bit (***3D_CNTRL<14>)*** to 0. Otherwise, setting it to one, would enable exclusive 3D color range keying. The Color keying is performed on all texels requested and before any filtering or processing has occurred. In other words, it is performed on the unmodified texels.
- **Note:** 3D Texture Color Range Keying must be disabled if texture mapping is disabled.
- Borealis supports integer- and 0.5-centered pixel and texel coordinates. This gives Borealis flexibility to support various different 3D API's specifications, i.e. Direct3D vs OpenGL. For integer-centered pixels, set (***3D_CNTRL<21>)*** to 0. For 0.5-centered pixels, set (***3D_CNTRL<21>)*** to 1. For integer-centered texels, set (***3D_CNTRL<26>)*** to 0. For 0.5-centered texels, set (***3D_CNTRL<26>)*** to 1.
- For backward compatibility to Imagine 128^4 (T2R IV), Borealis supports a special 8bpt index mode. By setting (***3D_CNTRL<29>)*** to 1, and setting ***(TEX_CNTRL<29:24> = 0x0D, 8-bit index mode)***, the triangle will be texture-mapped (Nearest mode must also be set) with the 8-bit index texture image. This mode will then use 8-bit value to index into the DAC palette to deliver the palettized result.
- To select which RGB value will be displayed, there is an RGB_SELECT bit (***3D_CNTRL<19>)*** which would select the RGB value either from the texture image or the vertices. Setting it to 0 would select the RGB from the texture image. Setting it to 1 will select RGB from the vertices.

Pixel Color Table

Non_Texel_Color Priority

1) Foreground Color if SOLID (***CMD<16>*** = 1) is enabled.

If RGB_Select is set to 1:

2) Background Color if internal pattern bit is set to zero.

3) Foreground Color if internal pattern bit is set to one and Gouraud Shading is not set.

4) Interpolated Gouraud Shaded Color if internal pattern bit is set to one and Gouraud Shading is set.

Else (RGB_Select is set to 0)

Texel_Color Priority

NOTE: (When texture mode (***TEX_CNTRL<0>*** = 1) is enabled.)

1) Modulated Texel Color corresponds to a modulated, blended or filtered Texel Color.

2) Texel Color.

OpenGL Texture Environment and Texture Functions

The following registers are used in the table to generate all of the OpenGL functions.

CONTROL	DEFINITION	REGISTER LOCATION
ABS	Alpha Blend Select	3D_CTRL[17]
TBS	Texture Blend Select	3D_CTRL[18]
RSEL	RGB Select	3D_CTRL[19]
RM	RGB Modulation	TEX_CNTRL[5]
ASL	Alpha Select	ACNTRL[24]
AMD	Alpha Modulation	ACNTRL[25]
DA	Decal Alpha Blend	ACNTRL[26]

NOTE: There can only be at most one RGB texture mode enabled and at most one Alpha texture mode enabled. Otherwise, the results are unpredictable.

The following table shows the OpenGL blend functions and how they can be obtained using the Borealis graphics processor.

TEXTURE FMT	FRAGMENT	REPLACE	MODULATE	DECAL	BLEND
Alpha	RGB	RSEL = 1 RM = x DA = x TBS = x	RSEL = 1 RM = x DA = x TBS = x	Undefined	RSEL = 1 RM = x DA = x TBS = x
	Alpha	ASL = 0 AMD = 0 ABS = 0	ASL = 0 AMD = 1 ABS = 0	Undefined	ASL = 0 AMD = 1 ABS = 0
:Luminance	RGB	RSEL = 0 RM = 0 DA = 0 TBS = 0	RSEL = 0 RM = 1 DA = 0 TBS = 0	Undefined	RSEL = 0 RM = 0 DA = 0 TBS = 1
	Alpha	ASL = 1 AMD = x ABS = x	ASL = 1 AMD = x ABS = x	Undefined	ASL = 1 AMD = x ABS = x
Luminance Alpha	RGB	RSEL = 0 RM = 0 DA = 0 TBS = 0	RSEL = 0 RM = 1 DA = 0 TBS = 0	Undefined	RSEL = 0 RM = 0 DA = 0 TBS = 1
	Alpha	ASL = 0 AMD = 0 ABS = 0	ASL = 0 AMD = 1 ABS = 0	Undefined	ASL = 0 AMD = 1 ABS = 0
Intensity	RGB	RSEL = 0 RM = 0 DA = 0 TBS = 0	RSEL = 0 RM = 1 DA = 0 TBS = 0	Undefined	RSEL = 0 RM = 0 DA = 0 TBS = 1
	Alpha	ASL = 0 AMD = 0 ABS = 0	ASL = 0 AMD = 1 ABS = 0	Undefined	ASL = 0 AMD = 0 ABS = 1
RGB	RGB	RSEL = 0 RM = 0 DA = 0 TBS = 0	RSEL = 0 RM = 1 DA = 0 TBS = 0	RSEL = 0 RM = 0 DA = 0 TBS = 0	RSEL = 0 RM = 0 DA = 0 TBS = 1
	Alpha	ASL = 1 AMD = x ABS = x	ASL = 1 AMD = x ABS = x	ASL = 1 AMD = x ABS = x	ASL = 1 AMD = x ABS = x
RGBA	RGB	RSEL = 0 RM = 0 DA = 0 TBS = 0	RSEL = 0 RM = 1 DA = 0 TBS = 0	RSEL = 0 RM = 0 DA = 1 TBS = 0	RSEL = 0 RM = 0 DA = 0 TBS = 1
	Alpha	ASL = 0 AMD = 0 ABS = 0	ASL = 0 AMD = 1 ABS = 0	ASL = 1 AMD = x ABS = x	ASL = 0 AMD = 1 ABS = 0

SGRAM Programming Settings

This section describes the software changes needed for the SGRAM based Borealis cards. The procedure for programming the SYSCLK PLL on the SGRAM T2R IV card is as follows:

1. Disable SPLL in the DAC by clearing bit 0 of DAC register 0x08.
2. Clear bit 24 of the SGRAM config register (I/O 0ffset 0x24).
3. Program the SYSCLK PLL by writing DAC registers 0x15 and 0x16.
4. Set bit 0 of DAC register 0x08.
5. Delay for about 1 ms or so.
6. Set bit 24 of the SGRAM config register.
7. Delay for 1 ms again.

As a programming note, bit 24 of the SGRAM config register is not affected by the trigger bit (bit 31), so it is not necessary to write this register twice each time you want to change bit 24.

Appendix B: *Peripheral Devices Interfaces*

The Peripheral Bus

In addition to the 128-bit Display Buffer interface, Borealis provides an additional 8-bit peripheral bus interface. The following devices can reside on the peripheral bus:

DEVICE	WIDTH	FUNCTION
RAM DAC	8	Output Stage
Soft Switches	8	Miscellaneous board functions (clock gen, LED)
EPROM	8	VGA BIOS

External RAMDAC

This interface provides direct support for a high performance RAM DAC such as the IBM526. If the RAMDAC is VGA compatible, I/O locations x3C6, x3C7, x3C8, and x3C9 can be mapped to the RAMDAC. See the section on VGA DAC shadowing for more information. In high performance modes, up to sixteen RAMDAC registers are mapped into Borealis memory space. If more than sixteen registers are required an external soft switch may be connected to the highest order register select line on the RAMDAC. The address and data lines for the DAC are supplied from the Peripheral Devices Interface. The RAMDAC data port is assumed to always be eight bits. The read and write control are provided from dedicated control pins. A typical RAMDAC connection is shown below. DAC wait states may be added via the DWS field in the CONFIG2 register.

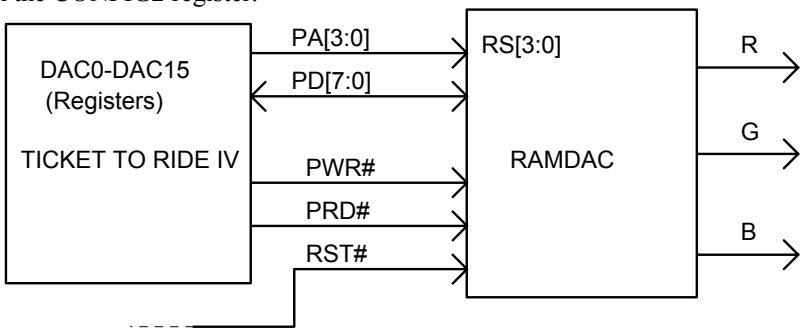


Figure 2-1.1. External RAMDAC Connections

Soft Switches

Up to 8 bits of external soft switch registers may be connected directly to the Peripheral Devices data bus (PD). The soft switches can be used to provide control for other board devices such as a frequency synthesizer, status indicator (LED), or any other device that requires software accessible control. The soft switch register is an I/O mapped device. The soft switch register is shadowed to an internal register that may be read at any time. On reset, the internal as well as external soft switch registers are initialized to 0.

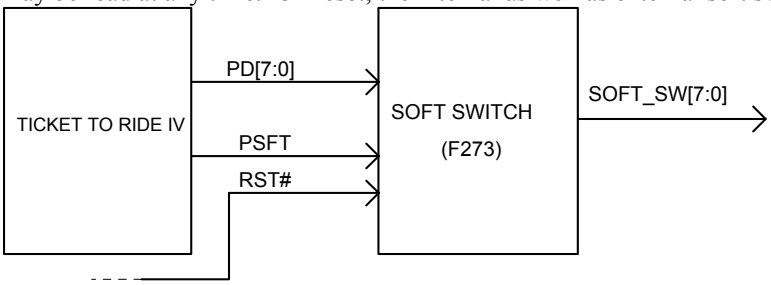


Figure 2-1.2. Soft Switch Connections

EPROM

Borealis can support an EPROM interface. To enable EPROM decode at boot, CP[29] needs to be strapped to a 1. The intended purpose of the EPROM is to provide the VGA BIOS if a VGA mode is enabled. The EPROM is mapped into PCI memory space by the EPROM Base Address Register in PCI configuration space. The amount of PCI memory space allocated to the EPROM is 64KB. When an EPROM read access is detected, Borealis will return an entire DWORD of data regardless of the state of the byte enables. Flash EPROM may also be used. In this case, single byte writes to the EPROM are supported. EPROM wait states may be added via the EWS field in CONFIG2.

To connect an EPROM to Borealis, an F373 latch must be used to construct the full EPROM address. The latch is controlled by the PCS# signal which is also the chip select for the EPROM. One F373 latch along with the unlatched address provides a large enough address to access 64KB of EPROM. The OE and WE pins of the EPROM are connected as shown below.

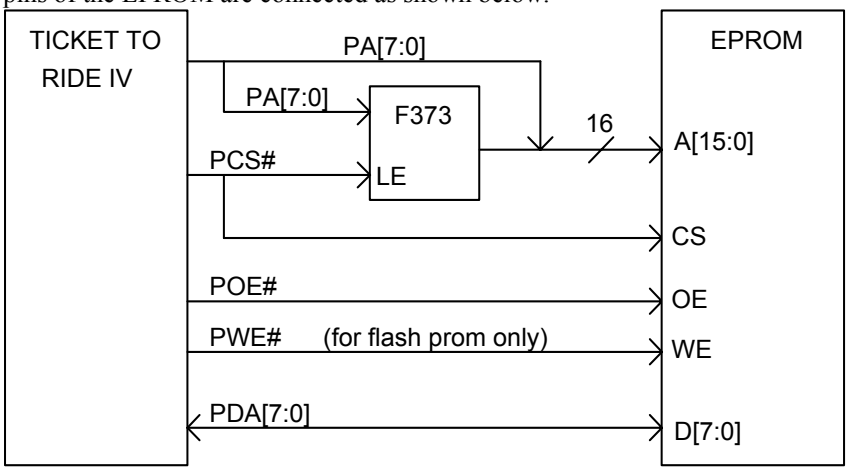


Figure 2-1.3. EPROM Connections

Appendix C: Mechanical Package Specification

388-Pin BGA

REVISIONS

DESCRIPTION	DATE	REV.
SEE DOCUMENT CONTROL TITLE PAGE	12-10-97	8

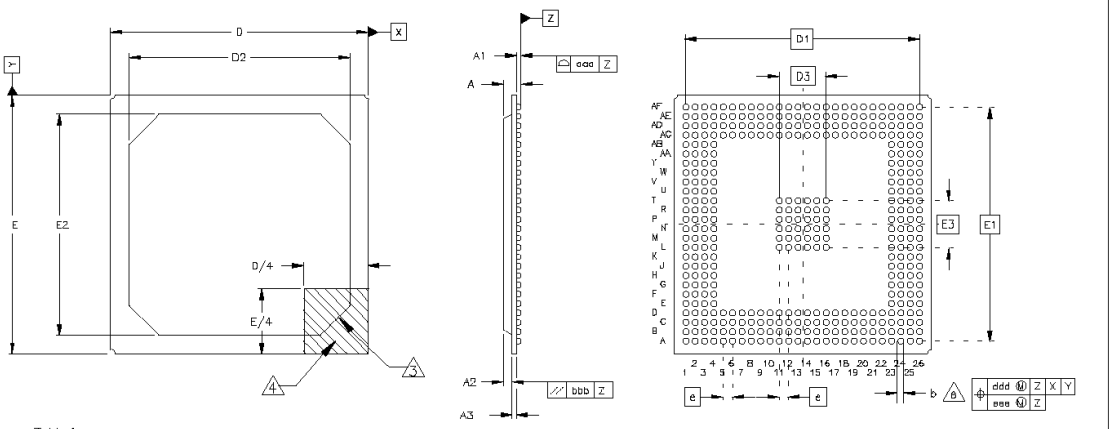


Table: 1

DIMENSIONS

SYM	MIN	NOM	MAX	NOTES
A	2.1	2.3	2.50	
A1	0.50	0.60	0.70	
A2	1.10	1.17	1.25	
A3		0.56 REF		
b	0.60	0.75	0.90	
D	34.70	35.00	35.30	
D1		31.75 BSC		4
D2	29.00	30.00	31.00	
D3		8.36 BSC		
E	34.70	35.00	35.30	
E1		31.75 BSC		4
E2	29.00	30.00	31.00	
E3		6.35 BSC		
e		1.27 BSC		
aaa	–	–	0.15	
bbb	–	–	0.25	
ddd	–	–	0.30	
eee	–	–	0.10	
TOTAL NO. OF PINS		388		

NOTES:

1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M-1994.
2. ALL DIMENSIONS IN MM.
3. CORNER DETAIL IS LSI LOGIC CORPORATION OPTION.
4. DETAILS OF PIN 1 IDENTIFIER ARE OPTIONAL, AND MAY CONSIST OF INK, LASER MARK OR METALLIZED MARKING, BUT MUST BE LOCATED WITHIN THE ZONE INDICATED
5. SOLDER BALLS ARE TYPICALLY 63/37 TIN LEAD.
6. JEDEC CODE MO-151.
7. PRIMARY DATUM Z AND SEATING PLANE ARE DEFINE BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.

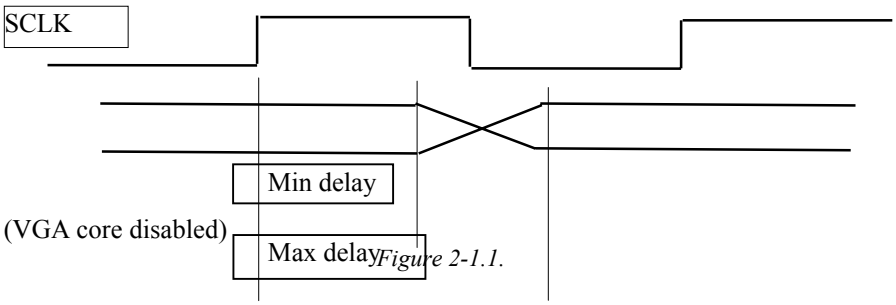
LSI LOGIC CORPORATION
1551 McCARTHY BLVD. MILPITAS, CA. 95035

TITLE: 388 PBGA 35x35mm 1.27mm BALL PITCH
CODE ID: MD97.II
DWG NO: JZ01-000448-00
REV: B
SCALE:
SHEET: 1 of 1

Appendix D: *Memory Interface*

SGRAM INTERFACE AC characteristic

Timing parameters shown below are referenced to rising edge of clock as seen on output of SCLK pin with internal clock skew control enabled .
SGR_CONFIG register [30:29] is set to = 01
(If SGR_CONFIG[30:29] = 00 then all minimum delays increase by 0.8ns, and all maximum delays increase by 1.4 ns).
All timing parameters are specified over operating conditions limited as follows:
Min conditions: Power supply 3.46 V, temperature (junction) 0 C
Max conditions: Power supply 3.13 V, temperature (junction) 85 C



HiRes mode (VGA core disabled)

Figure 2-1.1.

Table 3-1.1.

PIN NAME	Parameter	MIN DELAY (time in ns)	MAX DELAY (time in ns)	COMMENTS
PDAT [127:0]	data_out_delay	1.0	3.6	load 25 pF timing N/A for NOP cycles
RAD [10:0]	address_delay	1.2	3.2	load 60 pF
CSn[3:0]	chip_sel_delay	2.0 a) 2.7 b)	4.2 a) 5.5 b)	load 30 pF a) SGR_CONFIG [23]=0 b) SGR_CONFIG [23]=1
RASn	ras_delay	2.5	4.2	a)load 60 pF
CASn	cas_delay	1.9	3.3	a)load 60 pF
WEn	we_delay	2.3	4.3	a)load 60 pF
SF	sf_delay	2.2	3.7	a)load 60 pF
DQM	mask_delay	1.8	4.4	a)load 20 pF

VGA mode (HiRes mode disabled)

Table 3-1.2.

PIN NAME	**PARAMETER**	**MIN DELAY (time in ns)**	**MAX DELAY (time in ns)**	**COMMENTS**
PDAT [127:0]	data_out_delay	1.8	5.9	load 25 pF timing N/A for NOP cycles
RAD [10:0]	address_delay	1.5	5.7	load 60 pF
CSn[3:0]	chip_sel_delay	1.6	4.1	load 30 pF
RASn	ras_delay	2.14	3.8	a)load 60 pF
CASn	cas_delay	1.9	3.4	a)load 60 pF
WEn	we_delay	2.3	5.0	a)load 60 pF
SF	sf_delay	2.0	3.6	a)load 60 pF
DQM	mask_delay	1.5	4.3	a)load 20 pF

Appendix E: *Borealis Errata*

Errata

1. Wide stipple blits from offscreen memory don't work when the width > 900 or so pixels. The exact width where failures start occurring hasn't been determined yet.
2. Color blits from the drawing engine cache are offset by the low order bits in the destination. This can be worked around easily by writing a negative value to the source X coordinate to compensate. No software currently exists which uses this feature except verification tests.
3. Using XY origin mode for the bitmap start address has problems in simulation. These problems have not yet been fully characterized. No software currently exists which uses this feature except verification tests.
4. The two display list registers, DL_ADR and DL_CNTRL, must be written with long word accesses. Some older compilers will break 32-bit writes into two 16-bit writes, causing the display list to malfunction.
5. There seems to be a problem with the readback path for the TCT bit in the TEX_CTRL drawing engine register. When reading this register, the value read won't necessarily match what was written to it. This shouldn't affect TCT (Texture Cache Tile) mode.
6. Palette snooping is broken. Do not set bit 5 in PCI Config Space Command Register.
7. Do not set AGP DMA Request Queue Depth field to 0, 1, 2, 3 or greater than 0xF.
8. Do not set AGP DMA Preferred Request Length (PRL) to 16 or 32. There are problems with transaction lengths that have a PRL "remainder" at these PRLs. A PRL of 8 is recommended.
9. AD_STB*x* lines may need a series resistor when operating AGP 2X PIPE# or sideband on an Intel 440BX system.
10. When using the HDF (Host Data Format) bits in the linear memory window MWx_CTRL registers, data written is okay, but data read from the linear memory window is not always swapped correctly.
11. In Rev. A and B of the Ticket-to-Ride IV chip, Bit, Byte, and Word ordering does not work properly for PCI reads from Linear Memory Windows. These ordering functions (also called "swizzling") are controlled by the Memory Window Control Registers (MW0_CTRL, MW1_CTRL) bits[18:16]. The last transfer of a PCI read transaction returns "un-swizzled" data. The exact same incorrect data is returned every time swizzling is used. This bug is fixed in Rev. C of the Ticket-to-Ride IV chip.
12. Split-screen in VGA graphics mode 12 (hexadecimal) does not work properly

Index

Index

1

3

5

8

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

V

W

X

Y

Z