# 5

# *Register Set*

# Section 5: REGISTER SET

I/O mapped IMAGINE 128 registers must be accessed as DWORDs (32 bits).

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

## 5.1 Addressing Configuration Registers

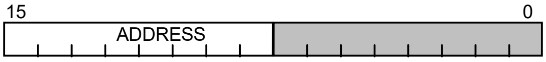
The addressing configuration group of registers are I/O mapped. The address of these registers is defined as follows. The value written into PCI base address register 5 is concatenated with the fixed address of the register to determine the system address of the register.

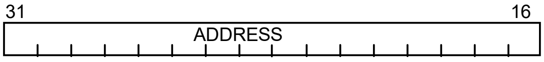
### 5.1.1 Register Base Address for the Global Register Block {PCIB5, (0x0000)}

**Name: RBASE_G**

**Type: I/O space read write**

This register defines the start address for the memory mapped global register block. In PCI mode, 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 |

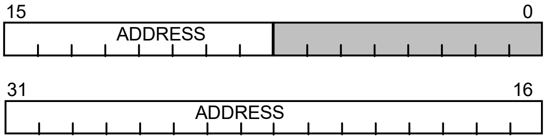
**Note: PCIB5 = The value written into PCI Base 5 Register.**

### 5.1.2 Memory Windows™ Register Block Base Address Register {PCIB5, (0x0004)}

**Name:** RBASE_W

**Type:** I/O space read write

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



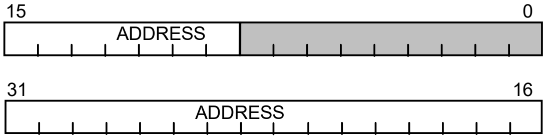
| Bits | Name | Default | Function |
|---|---|---|---|
| RBASE_W [31:8] | ADDRESS | | Address decode value |

### 5.1.3 Register Base Address Drawing Engine {PCIB5,(0x0008)}

**Name:** RBASE_D

**Type:** I/O space read write

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



| Bits | Name | Default | Function |
|---|---|---|---|
| RBASE_D[31:8] | ADDRESS | | Address decode value |

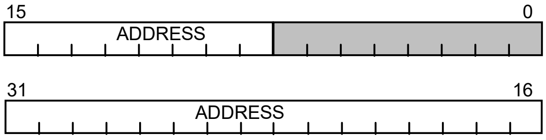
**Note: PCIB5 = The value written into PCI Base 5 Register.**

### 5.1.4 Register Base Address Global Interrupt Registers {PCIB5,(0x0010)}

**Name: RBASE_I**

**Type: I/O space read write**

This register defines the start address for the memory mapped global interrupt register block. In PCI mode, 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 |

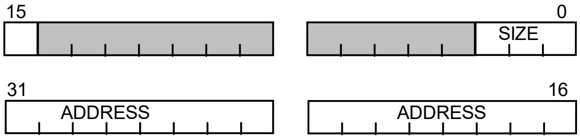
**Note: PCIB5 = The value written into PCI Base 5 Register.**

### 5.1.5 Register Base Address/Size EPROM registers {PCIB5, (0x0014)}

**Name: RBASE_E**

**Type: I/O space read write**

This register defines the start address for the memory mapped EPROM. In PCI mode, this register will be written with the same value as the expansion ROM Base Address Register and the size will be determined by CJ[10:8].



| Bits | Name | Default | Function |
|---|---|---|---|
| RBASE_E[31:15] | ADDRESS | | EPROM Address decode value<br>0x000C (default) |
| RBASE_E[2:0] | SIZE | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | EPROM Size<br>32 KB (default)<br>64 KB<br>128 KB<br>256 KB<br>512 KB<br>1 MB<br>2 MB<br>2 MB |

**Note: PCIB5 = The value written into PCI Base 5 Register.**

## 5.2 VGA DAC Shadow Registers

VGA DAC accesses are routed to the 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 IMAGINE 128 will own the VGA DAC cycles. IMAGINE 128 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 are controlled by the “vde” bit in the VGA-CTRL register, *see section E.2.3.*

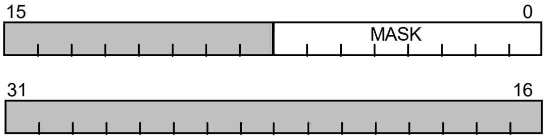
### ~~5.1.1~~5.2.1 Pixel Mask Registers 0 x 03C6

**Формат:** Список

**Name: PEL_MASK**

**Type: I/O or Memory mapped read write**

The contents of this register is logically AND’ed with the pixel input to the VGA color palette.



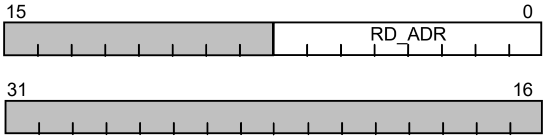
### ~~5.1.2~~5.2.2 Read Address Register 0x03C7

**Формат:** Список

**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.

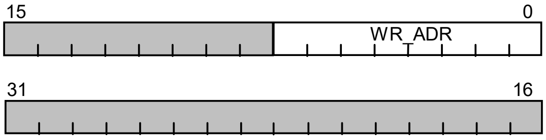


### ~~5.1.3~~5.2.3 Write Address Register 0x03C8

**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.



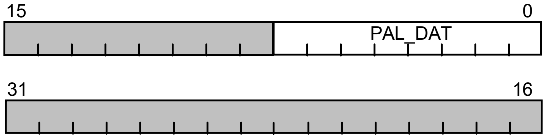
### ~~5.1.4~~5.2.4 Palette Data Register 0x03C9

**Формат:** Список

**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.



## 5.3 Miscellaneous I/O Registers

### ~~5.1.1~~5.3.1 ID Register {PCIB5,(0x0018)}

**Name: ID**

**Type: I/O space mapped read only**

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

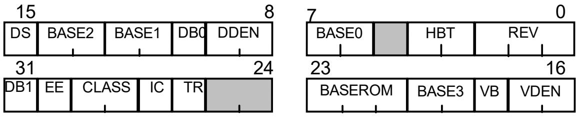
| 15 | | | | 8 | 7 | | | 0 |
|---|---|---|---|---|---|---|---|---|
| DS | BASE2 | BASE1 | DB0 | DDEN | BASE0 | ▨ | HBT | REV |

| 31 | | | | | 24 | 23 | | | 16 |
|---|---|---|---|---|---|---|---|---|---|
| DB1 | EE | CLASS | IC | TR | ▨ | BASEROM | BASE3 | VB | DDEN |

| BITS | NAME | JUMPER | VALUE | DESCRIPTION |
|---|---|---|---|---|
| ID[2:0] | REV | | | Chip Revision |
| ID[4:3] | HBT | | <br>0x0<br>0x1<br>0x2<br>0x3 | Host bus type<br>PCI Local Bus<br>Reserved<br>Reserved<br>Reserved |
| ID[7:6] | BASE0 | <br><br>CJ[1:0] = 00<br>CJ[1:0] = 01<br>CJ[1:0] = 10<br>CJ[1:0] = 11 | <br><br>0x0<br>0x1<br>0x2<br>0x3 | PCI Base 0 Address Register Size<br>( Linear Memory Window 0 )<br>4 Megabyte Memory Space Requested<br>8 Megabyte Memory Space Requested<br>16 Megabyte Memory Space Requested<br>32 Megabyte Memory Space Requested |
| ID[9:8] | DDEN | <br>CJ[26:25] = 00<br>CJ[26:25] = 01<br>CJ[26:25] =10<br>CJ[26:25] = 11 | <br>0x0<br>0x1<br>0x2<br>0x3 | Display buffer density<br>No memory present<br>256K bits by N memory chips<br>16Mbit SGRAM memories<br>Reserved |
| ID[31],ID[10] | DB1,DB0 | CJ[43:27]<br>00<br>01<br>10<br>11 | <br>0x00<br>0x01<br>0x10<br>0x11 | Number of memory banks in Display buffer<br>One bank<br>Two banks<br>Three banks<br>Four banks |

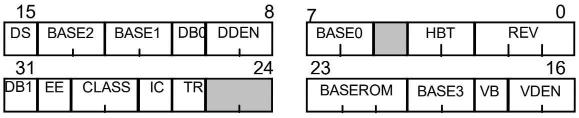
**Формат:** Список

**ID Register {PCIB5,(0x0018)} ( Continued )**



| BITS | NAME | JUMPER | VALUE | DESCRIPTION |
|---|---|---|---|---|
| ID[12:11] | BASE1 | CJ[3:2] = 00<br>CJ[3:2] = 01<br>CJ[3:2] = 10<br>CJ[3:2] = 11 | 0x0<br>0x1<br>0x2<br>0x3 | PCI Base 1 Address Register Size<br>( Linear Memory Window 1)<br>4 Megabyte Memory Space Requested<br>8 Megabyte Memory Space Requested<br>16 Megabyte Memory Space Requested<br>32 Megabyte Memory Space Requested |
| ID[14:13] | BASE2 | CJ[5:4] = 00<br>CJ[5:4] = 01<br>CJ[5:4] = 10<br>CJ[5:4] = 11 | 0x0<br>0x1<br>0x2<br>0x3 | PCI Base 2 Address Register Size<br>( XY Memory Window 0 )<br>4 Megabyte Memory Space Requested<br>8 Megabyte Memory Space Requested<br>16 Megabyte Memory Space Requested<br>32 Megabyte Memory Space Requested |
| ID[15] | DS | CJ[39] = 0<br>CJ[39] = 1 | 0x0<br>0x1 | IMAGINE 128™ Pixel Data Bus Size:<br>This Jumper defined bit specifies the actual pixel bus width. This bit is shadowed to the "one28" bit in the CONFIG1 register.<br>64 bit pixel data bus<br>128 bit pixel data bus |
| ID[17:16] | VDEN | CJ[29:28] = 00<br>CJ[29:28] = 01<br>CJ[29:28] =10<br>CJ[29:28] = 11 | 0x0<br>0x1<br>0x2<br>0x3 | Virtual buffer density<br>No memory present<br>256K bits by N memory chips<br>1M bit by N memory chips<br>Reserved |
| ID[18] | VB | CJ[30] = 0<br>CJ[30] = 1 | 0x0<br>0x1 | Number of banks in Virtual buffer<br>One bank<br>Two banks |
| ID[20:19] | BASE3 | CJ[7:6] = 00<br>CJ[7:6] = 01<br>CJ[7:6] = 10<br>CJ[7:6] = 11 | 0x0<br>0x1<br>0x2<br>0x3 | PCI Base 3 Address Register Size<br>( XY Memory Window 1 -- RESERVED)<br>4 Megabyte Memory Space Requested<br>8 Megabyte Memory Space Requested<br>16 Megabyte Memory Space Requested<br>32 Megabyte Memory Space Requested |

**ID Register {PCIB5,(0x0018)} ( Continued )**



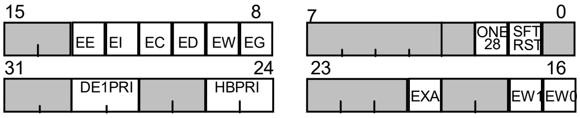
| BITS | NAME | JUMPER | VALUE | DESCRIPTION |
|---|---|---|---|---|
| ID[23:21] | BASEROM | <br>CJ[10:8] = 000<br>CJ[10:8] = 001<br>CJ[10:8] = 010<br>CJ[10:8] = 011<br>CJ[10:8] = 100<br>CJ[10:8] = 101<br>CJ[10:8] = 110<br>CJ[10:8] = 111 | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | PCI EPROM Base Address Register Size<br>32 Kilobyte Memory Space Requested<br>64 Kilobyte Memory Space Requested<br>128 Kilobyte Memory Space Requested<br>256 Kilobyte Memory Space Requested<br>512 Kilobyte Memory Space Requested<br>1 Megabyte Memory Space Requested<br>2 Megabyte Memory Space Requested<br>2 Megabyte Memory Space Requested |
| ID[25:24] | Reserved | | 0x0 | Reserved |
| ID[26] | TRAL | <br>CJ[33] = 0<br>CJ[33] = 1 | <br>0x0<br>0x1 | Extend RAS low timing. (see timing diagram for EDO configuration).<br>Normal timing<br>Extended timing<br>This is a "don't care" bit for SGRAM and WINDOW RAM configuration. |
| ID[27] | IC | <br>CJ[34] = 0<br>CJ[34] = 1 | <br>0<br>1 | PCI Interrupt Capability<br>No PCI Interrupt Capability<br>PCI Interrupt Capability |
| ID[29:28] | CLASS | <br>CJ[37:36] = 00<br>CJ[37:36] = 01<br>CJ[37:36] =10<br>CJ[37:36] = 11 | <br>0x0<br>0x1<br>0x2<br>0x3 | PCI Device Sub-Class<br>VGA<br>XGA<br>Other Display Controller<br>Reserved |
| ID[30] | EE | <br>CJ[38] = 0<br>CJ[38] = 1 | <br>0x0<br>0x1 | EPROM Enable:<br>This configuration jumper controlled signal is the master EPROM enable signal. If this signal is not asserted, then all EPROM access will be disabled.<br>EPROM Decode disabled<br>EPROM Decode Enabled |

### ~~5.1.2~~5.3.2 Configuration Register One {PCIB5,(0x001C)}

**Name: CONFIG1**

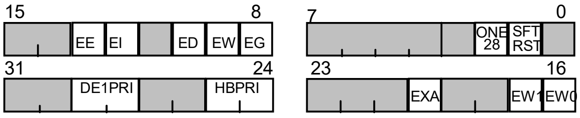
**Type: I/O space mapped read write**

This register contains information for chip configuration. There are three groups of control bits in this register. The arbitration priority bits control the amount of memory bandwidth allocated to the drawing engine and the host interface. 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. The third group of control bits specifies such functions as local buffers data bus width and soft reset generation.



| BITS | NAME | JUMPER | VALUE | DESCRIPTION |
|---|---|---|---|---|
| CONFIG[1] | SFT_RST | | | Software reset bit. |
| | | | 0 | No software reset generated. (default) |
| | | | 1 | Software reset generated. |
| CONFIG[2] | ONE28 | | | Set Pixel Bus Width: |
| | | | | This bit is set on power up by CJ[39} and may be altered by software at a later time. |
| | | CJ[39] = 0 | 0 | Pixel bus width is set to 64 bits. |
| | | CJ[39] = 1 | 1 | Pixel bus width is set to 128 bits. |

**Configuration Register One {PCIB5,(0x001C)} (Continued)**



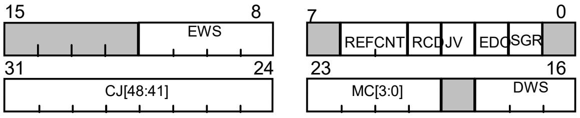
| BITS | NAME | JUMPER | VALUE | DESCRIPTION |
|---|---|---|---|---|
| CONFIG[21:8] | ENABLE DECODER | | 0<br>1 | Disable Decode for a specific block<br>Enable Decode for a specific block |
| | | | (Defaults) | |
| CONFIG[8] | EG | | 0 | Enable Global Decoder. |
| CONFIG[9] | EW | | 0 | Enable Mem Windows Reg Decoder. |
| CONFIG[10] | ED | | 0 | Enable Drawing Engine Decoder. |
| CONFIG[12] | EI | | 0 | Enable Global Interrupt Decoder. |
| CONFIG[13] | EE | | 1 | Enable EPROM Decoder. |
| CONFIG[16] | EW0 | | 0 | Enable Mem Window 0 Decoder. |
| CONFIG[17] | EW1 | | 0 | Enable Mem Window 1 Decoder. |
| CONFIG[20] | EXA | | 0 | Enable XY window Decoder. |
| CONFIG[25:24] | HBPRI | | | Host Interface Memory Bus Priority |
| | | | 0x0 | Low (default) |
| | | | 0x1 | Medium |
| | | | 0x2 | High |
| | | | 0x3 | Reserved |
| CONFIG1[29:28 | DE1PRI | | | Drawing Engine 1 Memory Bus Priorit |
| | | | 0x0 | Low (default) |
| | | | 0x1 | Medium |
| | | | 0x2 | High |
| | | | 0x3 | Reserved |

### ~~5.1.3~~5.3.3 Configuration Register Two {PCIB5,(0x0020)}

**Name: CONFIG2**

**Type: I/O space mapped read write**

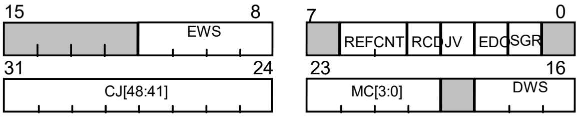
This register contains configuration information for IMAGINE 128☒ and peripheral devices supported by IMAGINE 128☒. The upper eight bits of this register are read-only and contain the settings for CJ[48:41]



| BITS | NAME | VALUE | DESCRIPTION |
|---|---|---|---|
| CONFIG2[2:1] | {ED0,SGR}<br>(CJ[49],CJ[35] ) | <br>00<br>01<br>1x | These two bits defines the type of memory timing generated for the local buffers. These bits can not be set by software.<br>Generate Window RAM timing.<br>Generate SGRAM timing.<br>Generate EDO timing |
| CONFIG2[3] | JV (CJ[50]) | <br>0<br>1 | This read only bit enables serial port read transfers to occur in all banks of Display buffer, regardless of the highest order address bit.<br>Transfer cycles are generated to a single bank of Display buffer<br>Transfer cycles are generated to all banks of Display buffer.<br>This bit has a meaning with dual ported memories only (VRAM or WINDOW RAM). |
| CONFIG2[4] | RCD(CJ[51]) | <br>0<br>1 | RAS to CAS delay select , this bit defines the delay between the falling edge of RAS and the falling edge of CAS for EDO type memories. Typically this bit should be set to one. For more details see EDO timing diagram.<br>Short delay<br>Normal delay<br>***For WINDOW RAM and SGRAM configuration this bit should be set to 1.*** |

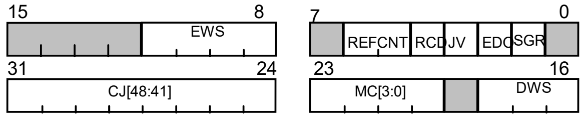
| | | | |
|---|---|---|---|
| CONFIG[6:5] | REFCNT<br>(CJ[53:52]) | | Refresh Count Select, these two bits define how often the DRAM refresh cycles occur. |
| | | 00 | Generate DRAM refresh every 768 mclocks |
| | | 01 | Generate DRAM refresh every 1024 mclocks |
| | | 10 | Generate DRAM refresh every 1280 mclocks |
| | | 11 | Generate DRAM refresh every 3584 mclocks |

**Configuration Register Two ( Continued)**



| | | | |
|---|---|---|---|
| CONFIG2[11:8] | EWS | | EPROM Wait States :<br>Each EPROM wait state adds one Memory Clock to the EPROM Access time. See IMAGINE 128 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 :<br>Each DAC wait state adds one Memory Clock to the DAC Access time. See IMAGINE 128 timing diagrams for more information. |
| | | 0x0 | Zero DAC wait states |
| | | 0x1 | One DAC wait states |
| | | 0x2 | Two DAC wait states |
| | | 0x3 | Three DAC wait states (Default) |
| | | 0x4 | Four DAC wait states |
| | | .0x5 | Five DAC wait states |
| | | 0x6 | Six DAC wait states |
| | | 0x7 | Seven DAC wait states |

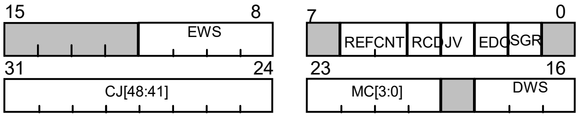
**Configuration Register Two (Continued)**



| BITS | NAME | VALUE | DESCRIPTION |
|---|---|---|---|
| CONFIG2[20] | MC[0] | <br><br>0<br>1 | Memory Control 0 - Display Buffer Enable<br>This bit will tristate all Display Buffer control signals (address, data, control). This is to allow another device access to IMAGINE 128's Display Buffer<br>Display Buffer signals are tristate (*Default*)<br>Display Buffer signals are driven |
| CONFIG2[21] | MC[1] | <br><br>0<br>1 | Memory Control 1 - DRAM Refresh Control<br>This bit controls whether DRAM refresh cycles are generated by IMAGINE 128<br>DRAM Refresh cycles generated (*Default*)<br>DRAM refresh disabled |
| CONFIG2[22] | MC[2] | <br><br>0<br>1 | Memory Control 2 - Data Sampling Control<br>This bit controls the sampling of data during read cycles from a IMAGINE 128 memory buffer.<br>Normal Data sampling (*Default*)<br>Delayed Data sampling<br>Typically this bit should be set to:<br>0 - for EDO configuration<br>1 - for Window Ram configuration<br>0 - for SGRAM configuration |
| CONFIG2[23] | MC[3] | <br><br>0 | Memory Control 3 - Memory Control Skew<br>This bit causes certain memory control signals to be skewed to allow more access time at higher MCLK rates.<br>Normal memory control signals (*Default*)<br>Value of this bit should remain zero. |

| | | | |
|---|---|---|---|
| CONFIG2[24] | FBB | <br><br><br><br>CJ[41] = 0<br>CJ[41] = 1 | Fast Back-to-Back Transfer Enable **(Read Only)**<br>This bit determines whether IMAGINE 128® supports fast back-to-back transfers. It is can also be read from bit 23 of the PCI Status Register.<br>Fast Back-to-Back Transfers are not supported<br>Fast Back-to-Back Transfers are supported |

**Configuration Register Two ( Continued)**



| BITS | NAME | JUMPER | DESCRIPTION |
|---|---|---|---|
| CONFIG2[25] | RVD | CJ[42] | RESERVED |
| CONFIG2[26] | Reserved | 0 | Reserved |
| CONFIG2[27] | CONT | <br><br>CJ[44] = 0<br>CJ[44] = 1 | Continuous Memory Cycle Enable (Read Only)<br>This bit determines whether IMAGINE 128® will concatenate multiple memory request that fall within the same row into a single memory cycle.<br>All new memory requests result in RAS precharge<br>New memory request may be combined |
| OK | RVD | CJ[45] | RESERVED |
| CONFIG2[29] | PRE | <br>CJ[46] = 1<br>CJ[46] = 0 | Linear Memory Windows Pre-fetch (Read Only)<br>Linear Memory Windows are pre-fetchable<br>Linear Memory Windows are not pre-fetchable |
| CONFIG2[31:30] | RVD | CJ[48:47] | Reserved (Read Only)<br>These jumpers can be used to indicate specific board configuration such as DAC speed, type of frequency synthesizer, etc. |

### ~~5.1.4~~5.3.4 SGRAM CONFIGURATION REGISTER {CJ,(0x0024}

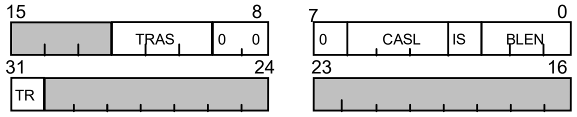
**Name: SGR_CONFIG**

**Type: I/O space mapped read write**

Programs SGRAM memories.

After a power up cycle , but before any of local memory buffers have been accessed for the very first time, the SGRAM 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 parameters is required. For example: switching from VGA mode to a high resolution mode may require reprogramming SGRAM.

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 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 | | Burst length.<br>BLEN has to be set to 001. |
| SGR_CONFIG[3] | IS | | Interleaved or sequential access between banks.<br>IS has to be set to 0 (sequential access). |
| SGR_CONFIG [6:4] | CASL | | CAS Latency- Set CASL to: 011 in Imagine mode. 010 in VGA mode |
| SGR_CONFIG[9:7] | vendor special opr | | Write 000 to these bits. |
| SGR_CONFIG[12:10] | TRAS | | RAS active time (minimum).<br>Value in this register should be calculated based on tRAS parameter from SGRAM specification and the actual period of MCLK (memory clock).<br>TRAS =(tRAS/MCLK) -1<br>Then round up the result to an interger number. Example: tRAS=70ns, MCLK period=11ns. TRAS=70/11-1=5.36=>6. Write TRAS=110. |
| SGR_CONFIG[31] | TR | | A write only trigger bit. A “one” written to this bit initialize configuration cycle. |

Typically the value in SGR_CONFIG[15:0] register should be set to:
0x1420 in VGA mode at 83 Mhz. 0x1020 in VGA mode at 60 Mhz.
0x1830 in Imagine mode at 100 Mhz. 0x1430 in Imagine mode at 83 Mhz.

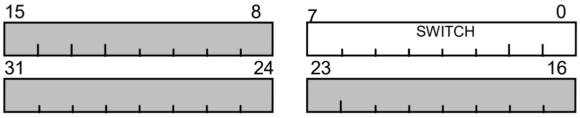
### ~~5.1.5~~5.3.5 Soft Switch Register {PCIB, (0x0028}

**Формат:** Список

**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.

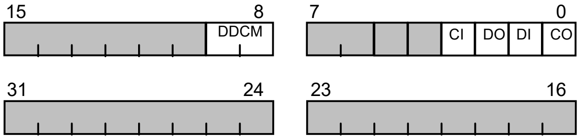


| **BITS** | **NAME** | **DEFAULT VALUE** | **DESCRIPTION** |
|---|---|---|---|
| SOFT_SW[7:0] | SWITCH | 0x00 | This register contains the same data as the external soft switch register. |

### ~~5.1.6~~5.3.6 DDC Register {PCIB5, (0x002C)}

**Name:** **DDC**

**Type:** **I/O mapped read write**



| Bits | Name | Value | Function |
|---|---|---|---|
| DDC[0] | C0 | 0x0<br>0x1 | DDC CLK OUT.<br>In DDC1 mode, this bit connects to the monitor vsync signal.<br>In DDC2B mode, this bit connects to the SCL line.<br>Forces clock signal low.<br>Forces clock signal to “Z”<br>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).<br>This bit shows actual value on the serial data line (DDC1 and/or DDC2B). |
| DDC[2] | DO | 0x0<br>0x1 | DDC2B Data OUT.<br>In DDC2B mode this bit connects to the Serial Data line..<br>This bit is N/A. in DDC1 mode.<br>Forces data signal low.<br>Forces data signal to “Z”<br>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).<br>This bit shows actual value on the serial clock line (SCL) . |
| DDC[9:8] | DDCM | 0x0<br>0x1<br>0x2<br>0x3 | DDC Mode.<br>DDC is disabled.<br>DDC1 is enabled.<br>DDC2B is enabled.<br>DDC is disabled. |

## 5.4 Global Interrupt Registers

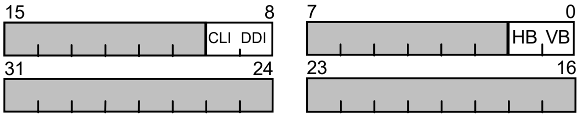
### ~~5.1.1~~5.4.1 Global Interrupt Register RBASE_I+(0x0000)

Формат: Список

**Name: GINTP**

**Type: Memory mapped read write**

This register contains the concatenation of all the IMAGINE 128 interrupts. The drawing engine interrupts in this register are read only and must be cleared in their respective interrupt registers.



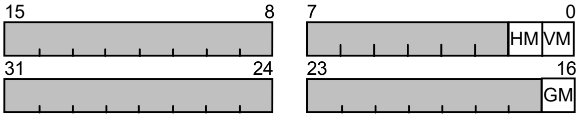
| Bits | Name | Value | Function |
|---|---|---|---|
| GINTP[0] | VB_INT | 0<br>1 | Vertical blank count match has not occurred<br>Vertical blank count match has occurred<br>See also INT_VCNT register in CRT Registers section |
| GINTP[1] | HB_INT | 0<br>1 | Horizontal blank count match has not occurred<br>Horizontal blank count has occurred<br>See also INT_HCNT register in CRT Registers section |
| GINTP[8] | DD_INT | 0<br>1 | Drawing engine operation not complete.<br>Drawing engine operation is complete. |
| GINTP[9] | CL_INT | 0<br>1 | Drawing engine clip interrupt has not occurred.<br>Drawing engine clip interrupt has occurred. |

### ~~5.1.2~~5.4.2 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 | <br>0<br>1 | Vertical Blank Mask<br>Vertical blank count interrupt disabled. (*Default*)<br>Vertical blank count interrupt enabled. |
| GINTM[1] | HB_MSK | <br>0<br>1 | Horizontal Blank Mask<br>Horizontal blank count interrupt disabled. (*Default*)<br>Horizontal blank count interrupt enabled. |
| GINTM[16] | GM | <br>0<br>1 | Global Interrupt mask<br>Disable all interrupts. (*Default*)<br>Enable un-masked interrupts |

## 5.5 RAM DAC Registers

The following 16 registers provide access to an external RAM DAC. The first four registers maintain VGA LUT compatibility and are therefore also accessible from I/O space *See Section 5.2 "VGA DAC Shadowing".* When accessing these registers in memory space, the VGA snooping enable bit in the PCI command register has no effect.

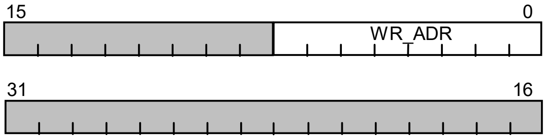
### ~~5.1.1~~5.5.1 Write Address Register RBASE_G + (0x0000) or RBASE_G + (0x0070)

**Формат:** Список

**Name: DAC0**

**Type: Memory mapped read write or I/O**

This register defines the starting write address of the VGA color palette. This register is auto incrementing.



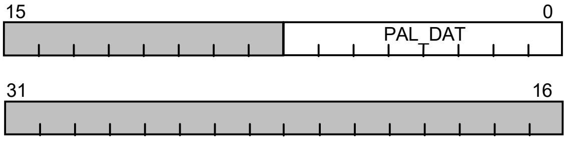
### ~~5.1.2~~5.5.2 Palette Data Register RBASE_G + (0x0004) or RBASE_G + (0x0074)

**Формат:** Список

**Name: DAC1**

**Type: Memory mapped read write or I/O**

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

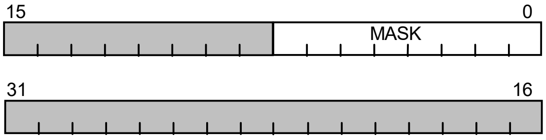


### 5.5.3 Pixel Mask Registers RBASE_G + (0x0008) or RBASE_G + (0x0078)

**Name: DAC2**

**Type: Memory mapped read write or I/O**

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

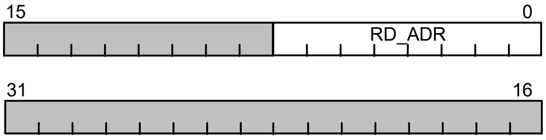


### 5.5.4 Read Address Register RBASE_G + (0x000C) or RBASE_G + (0x007C)

**Name: DAC3**

**Type: Memory mapped read write or I/O**

This register defines the read address of the VGA color palette. This register is auto incrementing.

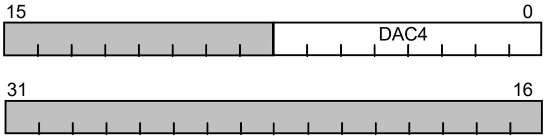


### 5.5.5 DAC Register 4 RBASE_G + (0x0010) or RBASE_G + (0x0080)

**Name: DAC4**

**Type: Memory mapped read write**

This register corresponds to the 4th register address of the RAMDAC.

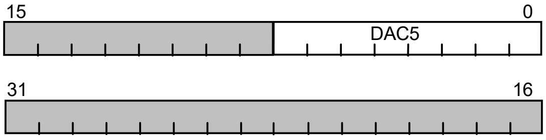


### 5.5.6 DAC Register 5 RBASE_G + (0x0014) or RBASE_G + (0x0084)

**Name: DAC5**

**Type: Memory mapped read write**

This register corresponds to the 5th register address of the RAMDAC.

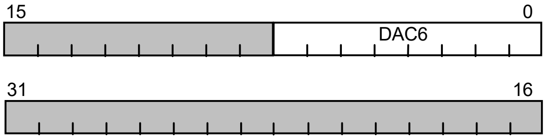


### 5.5.7 DAC Register 6 RBASE_G + (0x0018) or RBASE_G + (0x0088)

**Name: DAC6**

**Type: Memory mapped read write**

This register corresponds to the 6th register address of the RAMDAC.

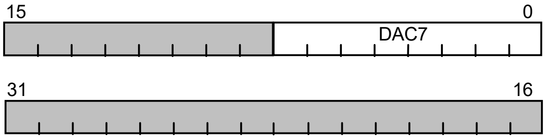


### 5.5.8 DAC Register 7 RBASE_G + (0x001C) or RBASE_G + (0x008C)

**Name: DAC7**

**Type: Memory mapped read write**

This register corresponds to the 7th register address of the RAMDAC.

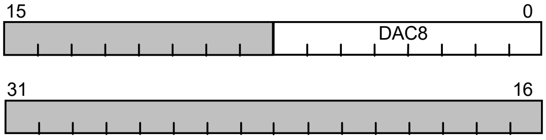


### 5.5.9 DAC Register 8 RBASE_G + (0x0090)

**Name: DAC8**

**Type: Memory mapped read write**

This register corresponds to the 8th register address of the RAMDAC.

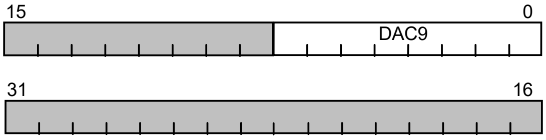


### 5.5.10 DAC Register 9 RBASE_G + (0x0094)

**Name: DAC9**

**Type: Memory mapped read write**

This register corresponds to the 9th register address of the RAMDAC.

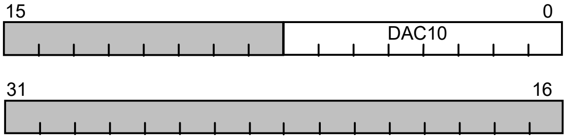


### 5.5.11 DAC Register 10 RBASE_G + (0x0098)

**Name: DAC10**

**Type: Memory mapped read only**

This register corresponds to the 10th register address of the RAMDAC.

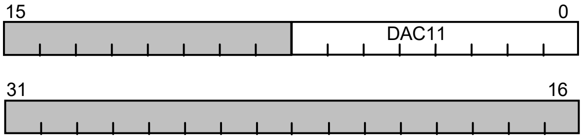


### 5.5.12 DAC Register 11 RBASE_G + (0x009C)

**Name: DAC11**

**Type: Memory mapped read write**

This register corresponds to the 11th register address of the RAMDAC.

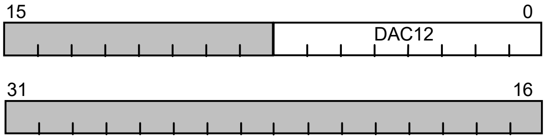


### 5.5.13 DAC Register 12 RBASE_G + (0x00A0)

**Name: DAC12**

**Type: Memory mapped read only**

This register corresponds to the 12th register address of the RAMDAC.

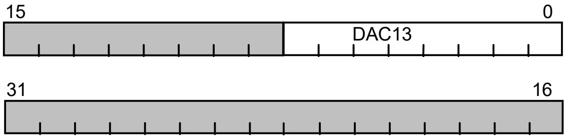


### 5.5.14 DAC Register 13 RBASE_G + (0x00A4)

**Name: DAC13**

**Type: Memory mapped read write**

This register corresponds to the 13th register address of the RAMDAC.

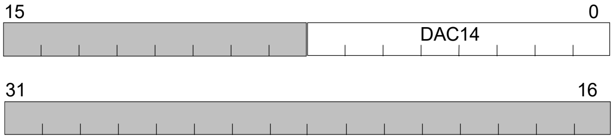


### 5.5.15 DAC Register 14 RBASE_G + (0x00A8)

**Name: DAC14**

**Type: Memory mapped read only**

This register corresponds to the 14th register address of the RAMDAC.

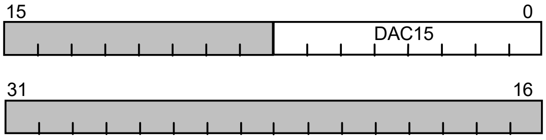


### 5.5.16 DAC Register 15 RBASE_G + (0x00AC)

**Name: DAC15**

**Type: Memory mapped read write**

This register corresponds to the 15th register address of the RAMDAC.



## 5.6 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.

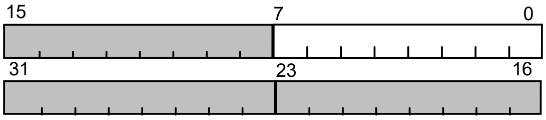
### ~~5.1.1~~5.6.1 Vertical Interrupt Count Register RBASE_G+(0x0020)

**Формат:** Список

**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<br>0x01<br>...<br>0xfe<br>0xff | Interrupt on every vertical field.<br>Interrupt on every other vertical field.<br>...<br>Interrupt on every 255 vertical fields.<br>Interrupt on every 256 vertical fields. |

### ~~5.1.2~~5.6.2 Horizontal Interrupt Count Register RBASE_G+(0x0024)

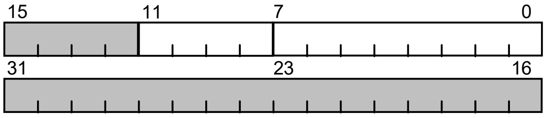
**Формат:** Список

**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.

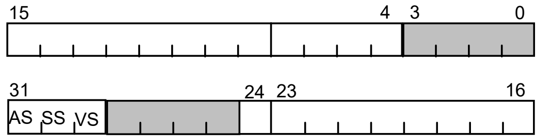


### ~~5.1.3~~5.6.3 CRT Display Start Address (RBASE_G + 0x0028)

**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 byte aligned. The start address is either 16-byte aligned or 128 byte aligned depending upon the type of memory used.



| Bits | Name | Value | Function |
|---|---|---|---|
| DB_ADR[24:0] | Start Address | | Display start address.<br>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.<br>Memory Type Alignment<br>WRAM 128-byte aligned<br>SGRAM 16-byte aligned<br>EDO DRAM 16-byte aligned<br>VRAM 16-byte aligned |
| DB_ADR[29] | VS | | Vertical Blank Status (read only) |
| | | 0 | Vertical Blank is 0 (screen blanked) |
| | | 1 | Vertical Blank is 1 (screen active) |
| DB_ADR[30] | SS | | Source select status (read only)<br>Any write into SEN bit in CRT_2CON sets this bit to one. At the time of accepting a new value in SEN, the SS bit gets cleared. |
| | | 0 | A write into SEN bit in CRT_2CON register has been synchronized. |
| | | 1 | A write into SEN bit in CRT_2CON register hasn't been synchronized yet |

| | | | |
|---|---|---|---|
| DB_ADR[31] | AS | | Display Address Status (read only)<br>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. |
| | | 0 | A write into DB_ADDR[24:0] has been synchronized. |
| | | 1 | A write into DB_ADDR[24:0] hasn't been synchronized yet |

Three most significant bits in this register are "read only" bits intended for use with applications using double buffering technique.

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.

For single ported memory configuration (DRAM, SGRAM) the Virtual Buffer can effectively become a secondary display buffer used for double buffering. Bit SEN in CRT_2CON register selects the displayable buffer. Selection of the displayable buffer is synchronized in similar way as the start address and the status of synchronization can be read from SS bit.

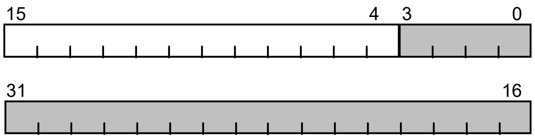
### ~~5.1.4~~5.6.4 Display Buffer Pitch (RBASE_G + 0x002C)

**Формат:** Список

**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. *For WINDOW RAM configuration align the pitch value to 128 bytes.*



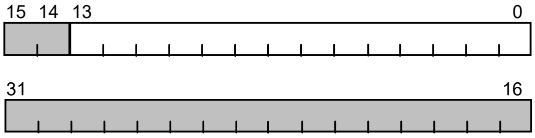
### ~~5.1.5~~5.6.5 CRT Horizontal Active Line (RBASE_G + 0x0030)

**Формат:** Список

**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.



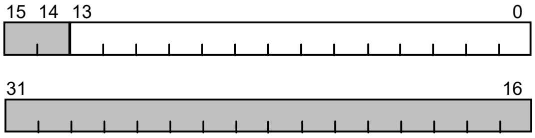
### ~~5.1.6~~5.6.6 CRT Horizontal Blank Width (RBASE_G + 0x0034)

**Формат:** Список

**Name: CRT_HBL**

**Type: Memory mapped read write**

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



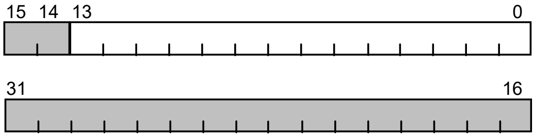
### ~~5.1.7~~5.6.7 CRT Horizontal Front Porch Width (RBASE_G + 0x0038)

**Формат:** Список

**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.



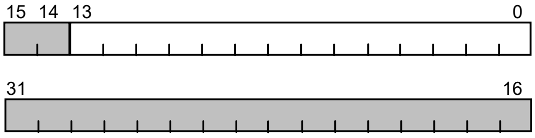
### ~~5.1.8~~5.6.8 CRT Horizontal Sync Width (RBASE_G + 0x003C)

**Формат:** Список

**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. In interlaced mode the smallest value is 2.

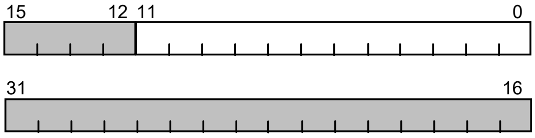


### ~~5.1.9~~5.6.9 CRT Vertical Field Active (RBASE_G + 0x0040)

**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. In interlaced mode (when a frame consists of two fields), it is recommended to write an odd number into this register; thus every field contains an integral number of lines plus a half-line displayed at the beginning of the even field and at the end of the odd field.



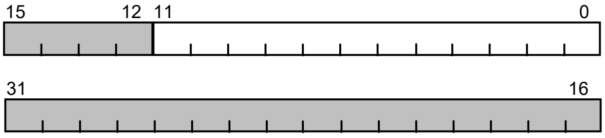
### ~~5.1.10~~5.6.10 CRT Vertical Blank Width (RBASE_G + 0x0044)

**Формат:** Список

**Name: CRT_VBL**

**Type: Memory mapped read write**

This register specifies the number of lines blanked during one frame. In interlaced mode it is recommended to write an even number into this register, so every half-frame (field) consist of integral number of blanked lines (this also is true for blank between an odd and even field, where blank starts and ends with half lines but total still gives full number of lines).



### ~~5.1.11~~5.6.11 CRT Vertical Front Porch Width (RBASE_G + 0x0048)

**Формат:** Список

**Name:** **CRT_VFP**

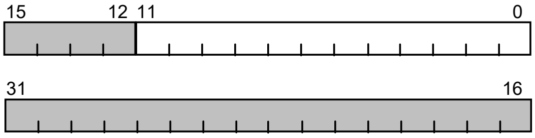
**Type:** **Memory mapped read write**

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

a) number of lines in non interlaced mode.

b) number of half-lines in interlaced mode.

This is to support some interlaced standards which require front porch to be set to a non integral number of lines.



### ~~5.1.12~~5.6.12 CRT Vertical Sync Width (RBASE_G + 0x004C)

**Формат:** Список

**Name:** **CRT_VS**

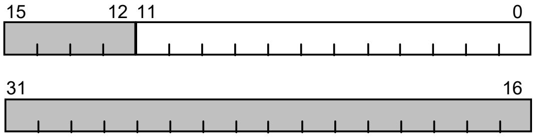
**Type:** **Memory mapped read write**

This register specifies the width of the vertical sync impulse The units of this register are:

a) number of lines in non interlaced mode

b) number of half-lines in interlaced mode.

This is to support some interlaced standards which require the vertical impulse to last a non integral number of lines.



### ~~5.1.13~~5.6.13 CRT Line Counter (RBASE_G + 0x0050)

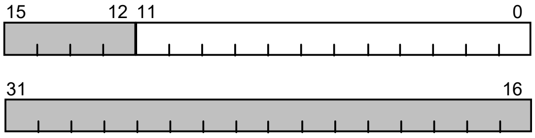
**Formataт:** Список

**Name: CRT_LCNT**

**Type: Memory mapped read only**

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

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

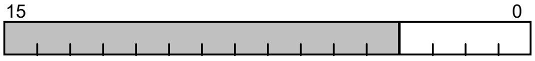


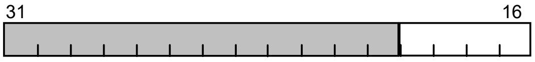
### ~~5.1.14~~5.6.14 CRT Display Buffer Zoom Factor (RBASE_G + 0x0054)

**Name: CRT_ZOOM**

**Type: Memory mapped read write**

This register defines the vertical/horizontal zoom factor for the display output stage.





| Bits | Name | Value | Function |
|---|---|---|---|
| CRT_ZOOM[3:0] | ZOOM[3:0] | 0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>.....<br>0xd<br>0xe<br>0xf | No Zoom<br>Zoom 2X<br>Zoom 3X<br>Zoom 4X<br>Zoom 5X<br>.............<br>Zoom 14X<br>Zoom 15X<br>Zoom 16X |

**Note: In interlaced mode, only zoom factors of 2, 4, 8, and 16 are available.**

| Bits | Name | Value | Function |
|---|---|---|---|
| CRT_ZOOM [19:16] | HSFT[3:0] | 0x0<br>0x1<br>0x3<br>0x7<br>0xF | No Divide<br>Divide By 2<br>Divide by 4<br>Divide by 8<br>Divide by 16 |

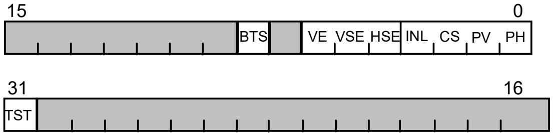
HSFT[3:0] bits cause simple right shifting of all horizontal sync parameters. Using this feature for zoom purpose may require (in some applications) dividing VCLK by the same factor.

**Формат:** Список

### ~~5.1.15~~5.6.15 CRT Configuration Register 1 (RBASE_G + 0x0058)

**Name: CRT_1CON**

**Type: Memory mapped read write**

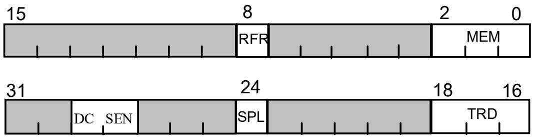


| Bits | Name | Value | Function |
|---|---|---|---|
| CRT_1CON[0] | PH | <br>0<br>1 | Horizontal Sync Polarity.<br>Negative horizontal sync.<br>Positive horizontal sync. |
| CRT_1CON[1] | PV | <br>0<br>1 | Vertical Sync Polarity.<br>Negative vertical sync.<br>Positive vertical sync. |
| CRT_1CON[2] | CS | <br>0<br>1 | Composite Sync Enable.<br>The polarity of composite sync is **always negative**. If composite sync is enabled, then Vertical Sync becomes a one clock wide impulse (always negative).<br>Separate HSYNC and VSYNC are generated<br>Composite sync is output on HSYNC |
| CRT_1CON[3] | INL | <br>0<br>1 | Interlaced mode select<br>Non-interlaced display<br>Interlaced display (supported in VRAM Mode only) |
| CRT_1CON[4] | HSE | <br>0<br>1 | Horizontal Sync Enable<br>HSYNC disabled (HSYNC = 1)<br>Normal HSYNC generation |
| CRT_1CON[5] | VSE | <br>0<br>1 | Vertical Sync Enable<br>VSYNC disabled (VSYNC = 1)<br>Normal VSYNC generation |
| CRT_1CON[6] | VE | <br>0<br>1 | Video Enable<br>BLANK disabled (BLANK= 0)<br>Normal BLANK and BORDER generation |
| CRT_1CON[8] | BTS | <br>0<br>1 | SCLK Pin direction<br>SCLK is an input to IMAGINE 128®<br>SCLK is an output from IMAGINE 128®:<br>The CRT clock (VCLK) is AND-ed with an internal blank signal and then output to SCLK pin to be used as VRAM or WINDOW RAM shift clock. |
| CRT_ICON[31] | TST | 0x0<br>0x1 | CRT Test Mode Bit Normal Operation.<br>Test Mode. |

### ~~5.1.16~~5.6.16 CRT Configuration Register 2 (RBASE_G + 0x005C)

**Name: CRT_2CON**

**Type: Memory mapped read write**



| Bits | Name | Value | Function |
|---|---|---|---|
| CRT_2CON[0] | MEM[0] | <br>0<br>1 | Serial port depth<br>VRAM serial port is 512 deep<br>VRAM serial port is 256 deep<br><br>For Window RAM configuration write 0 to this bit |
| CRT_2CON[2:1] | MEM[2:1] | <br><br><br>0x0<br>0x1<br>0x2<br>0x3<br><br><br>0x0<br>0x1<br>0x2<br>0x3 | Total shift register size (in one bank of display buffer) is:<br>in VRAM mode:<br>4kB<br>2kB<br>8kB<br>Reserved<br><br>in Window RAM mode:<br>1kB (128 bit bus)<br>512B (64 bit bus)<br>Reserved<br>Reserved |
| CRT_2CON[8] | RFR | <br><br><br>0<br>1 | Enables screen refresh (transfers to serial registers of VRAM/WINRAM buffer or to<br>CRT FIFO in DRAM/SGRAM mode)<br>- refresh disabled<br>- refresh enabled |
| CRT_2CON[18:16] | TRD | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | "Memory to register" transfer delay<br>no delay<br>1 VCLK delay<br>2 VCLKs delay<br>3 VCLKs delay<br>4 VCLKs delay<br>5 VCLKs delay<br>6 VCLKs delay<br>7 VCLKs delay |

continued..

| Bits | Name | Value | Function |
|---|---|---|---|
| CRT_2CON[24] | SPL | 0<br>1 | Split transfer Control (VRAM only)<br>- disabled<br>- enabled<br>Write 0 for Window RAM |
| CRT_2CON[28] | SEN | 0x0<br>0x1 | Source Enable<br>Display from Display Buffer<br>Display from Virtual Buffer<br>(In this case the Virtual Buffer of Imagine becomes effectively a display buffer. This can be used with single ported memories only as DRAM or SGRAM)<br>This bit can be written to at any time, however, changes have no effect until following vertical blank interval. At the time a value in SEN is actually accepted , bit SS in DB_ADDR register gets cleared to zero. |
| CRT_2CON[29] | DC | 0x0<br>0x1 | Display Configuration<br>Dual ported memories (VRAM/WINRAM)<br>Single ported memories(DRAM/SGRAM) |

## 5.7 Memory Windows$^{TM}$ 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.

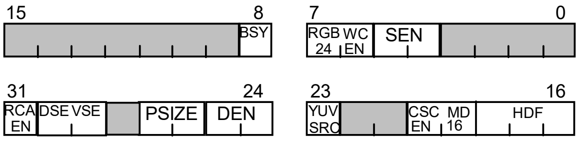
### ~~5.1.1~~5.7.1 Memory Window Control Register

**Формат:** Список

**window zero RBASE_W + (0x0000)**
**window one RBASE_W + (0x0028)**

**Name: MW0_CTRL, MW1_CTRL,**

**Type: Memory mapped read write**



This register contains the buffer control bits for the linear memory window. Contained within this register are the source control bits and the destination control bits.

The source enable bits (SEN) select which of the local buffers is the source of data during a read cycle. It is possible to set SEN so that none of the local buffers are enabled during read cycles. This effectively creates a "write only" memory window.

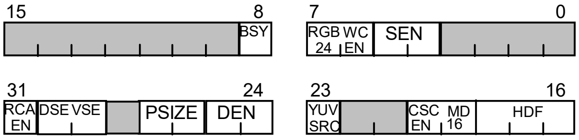
The destination enable bits (DEN) select which of the local buffers is the destination of data during a write cycle. It is possible to set DEN so that none of the local buffers are enabled during write cycles. This effectively creates a "read only" memory window.

Multiple buffers may be written to simultaneously by enabling the appropriate shadow enable bit. There are two shadow enable bits, one corresponding to each of the two local buffers. Any combination of shadow enable bits is allowed

The pixel size bits define the pixel depth for the current memory window.

**Memory Window Control Register (Continued)**

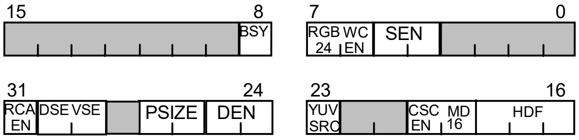
**window zero RBASE_W + (0x0000)**
**window one RBASE_W + (0x0028)**



| Bits | Name | Value | Function |
|---|---|---|---|
| MWX_CTRL[3:0] | Reserved | 0x0 | Reserved |
| MWX_CTRL[5:4] | SEN[1:0] | <br>0x0<br>0x1<br>0x2<br>0x3 | Source Read Enable.<br>Enable reads from the Display buffer.<br>Enable reads from the Virtual buffer.<br>Reserved.<br>All reads disabled. |
| MWX_CTRL[6] | WCEN | 0x0<br>0x1 | Enable write caching<br>Disable write caching |
| MWX_CTRL[7] | Reserved | 0x0 | Reserved |
| MWX_CTRL[8] | BSY | <br>0x0<br>0x1 | Memory Window Busy **(read only)**<br>Memory Window is busy<br>Memory Window is not busy |
| MWX_CTRL[16] | HDF[0] | 0x0<br>0x1 | Bit ordering is not modified<br>Swaps bits within each byte |
| MWX_CTRL[17] | HDF[1] | 0x0<br>0x1 | Byte ordering is not modified<br>Swap bytes within each word |
| MWX_CTRL[18] | HDF[2] | 0x0<br>0x1 | Word ordering is not modified<br>Swap words within each DWORD |
| MWX_CTRL[19] | MD16 | <br>0x0<br>0x1 | 16 Bits per pixel mode<br>0=5RED,5GREEN,5BLUE<br>1=5RED,6GREEN,5BLUE |
| MWX_CTRL[20] | CSC_EN | <br>0x0<br>0x1 | Color space conversion enable<br>Disable color space conversion<br>Enable color space conversion |
| MWX_CTRL[22:21] | Reserved | 0x0 | Reserved |
| MWX_CTRL[23] | YUV_SRC | <br>0x0<br>0x1 | YUV source format.<br>YUV 422. (Default)<br>YUV 444. |

**Memory Window Control register (Continued)**

**window zero RBASE_W + (0x0000)**
**window one RBASE_W + (0x0028)**



| Bits | Name | Value | Function |
|---|---|---|---|
| MWX_CTRL[25:24] | DEN[1:0] | | Destination Write Enable. |
| | | 0x0 | Enable writes to the Display buffer. |
| | | 0x1 | Enable writes to the Virtual buffer. |
| | | 0x2 | Reserved. |
| | | 0x3 | All writes disabled. |
| MWX_CTRL[27:26] | PSIZE[1:0] | | Destination pixel size, number of bits per pixel |
| | | 0x0 | Eight bits per pixel |
| | | 0x1 | Sixteen bits per pixel |
| | | 0x2 | Thirty-two bits per pixel |
| | | 0x3 | Twenty-four |
| MWX_CTRL[28] | Reserved | 0x0 | Reserved |
| MWX_CTRL[29] | VSE | | Virtual Buffer shadow enable. |
| | | 0 | Disable shadowed writes to the Virtual buffer. |
| | | 1 | Enable shadowed writes to the Virtual buffer. |
| MWX_CTRL[30] | DSE | | Display Buffer shadow enable. |
| | | 0 | Disable shadowed writes to the Display buffer. |
| | | 1 | Enable shadowed writes to the Display buffer. |
| MWX_CTRL[31] | RCA_EN | | Read cache enable. |
| | | 0 | Enable read caching. (Default) |
| | | 1 | Disable read caching. |

## COLOR SPACE CONVERSION CONTROL TRUTH TABLE.

| PSIZE | CSC_EN | Description |
|---|---|---|
| 00 | 1 | Direct access, No packing. |
| 00 | 0 | Direct access, No packing. |
| 01 | 1 | Destination 16 bit, source YUV.. |
| 01 | 0 | Direct access, No packing. |
| 10 | 1 | Destination 32 bit, source YUV. |
| 10 | 0 | Direct access, No packing. |
| 11 | 0 | Direct access, No packing. |

### ~~5.1.2~~5.7.2 Memory Windows Address Registers

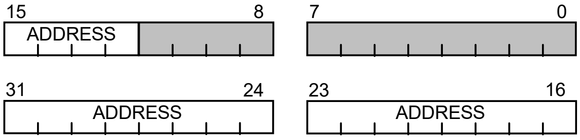
**window zero** **RBASE_W + (0x0004)**
**window one** **RBASE_W + (0x002C)**

**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 IMAGINE 128® 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 section 4.8.3. 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.

In PCI mode, the system BIOS is responsible for allocation of all system resources. The allocation process is facilitated by the PCI base registers described in section 3. IMAGINE 128® will request 4, 8, 16, or 32 megabytes of memory space for each of the memory windows based on the configuration jumper settings. The system will then allocate memory to IMAGINE 128® 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. In VL mode, it is software's responsibility to locate the memory windows so that there is no conflict with other devices in a system.



| Bits | Name | Value | Function |
|---|---|---|---|
| MWX_AD[31:12] | ADDRESS | | Address decode value |

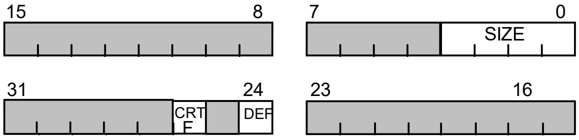
### ~~5.1.3~~5.7.3 Memory Window Size

**window zero RBASE_W + (0x0008)**
**window one RBASE_W + (0x0030)**

**Name:MW0_SZ, MW1_SZ**

**Type: Memory mapped read write**

This register defines the size of a Memory window. In PCI mode, 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<br>0x1<br>0x2<br>...<br>0xD<br>0xE<br>0xF | Size equals 4K<br>Size equals 8K<br>Size equals 16K<br>...<br>Size equals 32MB<br>RESERVED<br>RESERVED |
| MWX_SZ_DIB[24] | DEF | <br>0x0<br>0x1 | Drawing Engine Flush<br>0 = disable. (Default)<br>1 = enable. |
| MWX_SZ_DIB[26] | CRTF | <br>0x0<br>0x1 | CRT Flush<br>0 = disable. (Default)<br>1 = enable. |

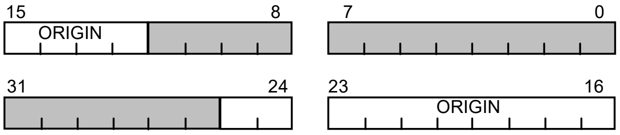
### ~~5.1.4~~5.7.4 Memory Window Origin

**window zero** **RBASE_W + (0x0010 or 0x0014)**
**window one** **RBASE_W + (0x0038 or 0x003C)**

**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] | | | Upper order memory window address bits |
| | MWX_ORG[24:12] | | Upper order address for 4 kilobyte window |
| | MWX_ORG[24:13] | | Upper order address for 8 kilobyte window |
| | MWX_ORG[24:14] | | Upper order address for 16 kilobyte window |
| | • | | • |
| | • | | • |
| | • | | • |
| | MWX_ORG[24:23] | | Upper order address for 8 megabyte window |
| | MWX_ORG[24] | | Upper order address for 16 megabyte window |

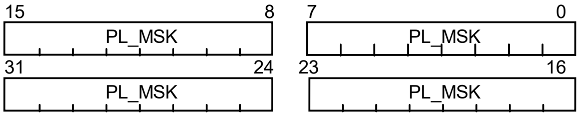
### ~~5.1.5~~5.7.5 Memory Window Plane Mask

**window zero RBASE_W + (0x0024)**
**window one RBASE_W + (0x004C)**

**Name: MW0_MASK, MW1_MASK,**

**Type: Memory mapped read write**

The MASK register allows for selective writing of pixel planes to the Virtual or Display Buffer. Using this feature, bit maps may be partitioned into logical groups for selective updating. A bit value of zero disables writing to 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.



| Bits | Name | Value | Function |
|---|---|---|---|
| MWX_MASK [31:0] | PL_MSK | | Plane Masking Register |

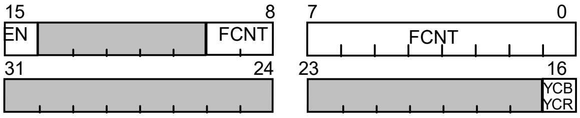
### ~~5.1.6~~5.7.6 Memory Window Cache Flush Counter

**RBASE_W + (0x0050)**

**Name:** **MWC_FCNT**

**Type:** **Memory mapped read write**

This register defines the time out for flushing the Memory Windows cache. When bit [15], cache count is enabled, bits [9:0] define the time in terms of host clocks before the data in their cache will be flushed to memory. If no value is written to bits [[9:0] it defaults to “0xff” this will allow 256 host clocks between linear window rights without automatically flushing the cache to local memory.



| Bits | Name | Value | Function |
|---|---|---|---|
| MWC_FCNT[9:0] | FCNT | 0x000<br>to<br>0x3ff | Number of host clocks allowed between writes without an automatic cache flush.<br>0x000 = 1 clock between flush.<br>0x3ff = 1024 clocks between flush. |
| MWX_MASK[15] | EN | <br>0x0<br>0x1 | Flush count enable bit.<br>0 = counter disabled.<br>1 = counter enabled. |
| MWX_MASK[16] | YCB/YCR | <br>0x0<br>0x1 | YCB / YCR Selector.<br>0 = YCB.<br>1 = YUV. |

### ~~5.1.7~~5.7.7 Memory Window Flush Trigger

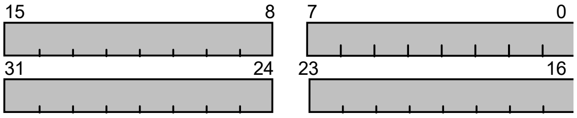
**Формат:** Список

**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.



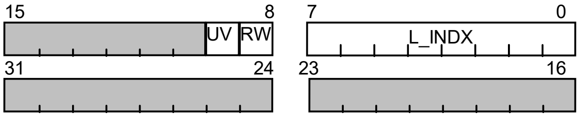
### ~~5.1.8~~5.7.8 YUV LUT Index Register

**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<br>to<br>0xff | Address pointer into the color space conversion LUT. |
| YUV_ADR[8] | RE | <br>0x0<br>0x1 | LUT read enable bit.<br>0 = Normal operation. (Default)<br>1 = Read back mode. |
| YUV_ADR[9] | UV | <br>0x0<br>0x1 | LUT select.<br>0 = Select V LUT. (Default)<br>1 = Select U LUT. |

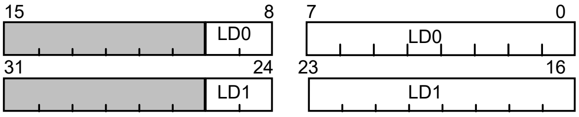
### ~~5.1.9~~5.7.9 YUV LUT Data Register

Формат: Список

**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. |

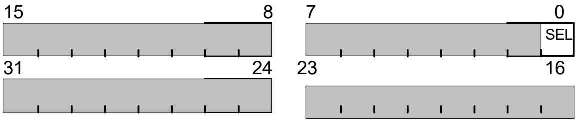
### ~~5.1.10~~5.7.10 Memory Window Shared Control Register

Формат: Список

**RBASE_W + (0x0060)**

**Name:** **MWS_CTRL**

**Type:** **Memory mapped read, write**



| Bits | Name | Value | Function |
|---|---|---|---|
| MWS_CTRL[0] | SEL | <br>0x0<br>0x1 | Selects Window0 / Window1 Size Register<br>0 = Window 0<br>1 = Window 1 |

## 5.8 Drawing Engine Command and Parameter Registers

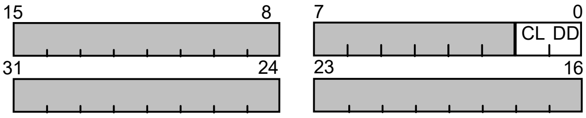
The following group of registers are memory mapped and are used to issue commands and parameters to the drawing engine. The address of these registers is offset by RBASE_D.

### ~~5.1.1~~5.8.1 Interrupt Register RBASE_D + (0x0000)

**Формат:** Список

**Name: INTP**

**Type: Memory mapped read write**



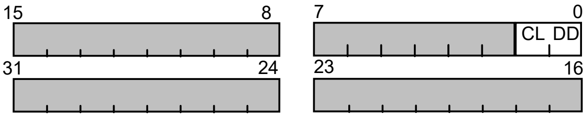
| Bits | Name | Value | Function |
|---|---|---|---|
| INTP[0] | DD_INT | 0<br>1 | Drawing operation not completed<br>Drawing operation is completed |
| INTP[1] | CL_INT | 0<br>1 | Clip interrupt has not occurred<br>Clip interrupt has occurred |
| INTP[31:2] | Reserved | 0x0 | Reserved |

### ~~5.1.2~~5.8.2 Interrupt Mask Register RBASE_D + (0x004)

**Формат:** Список

**Name: INTM**

**Type: memory mapped read write**



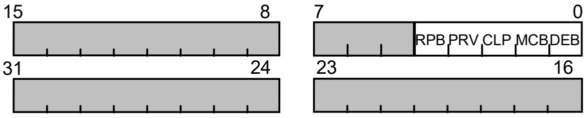
| Bits | Name | Value | Function |
|---|---|---|---|
| INTM[0] | DD_MSK | 0<br>1 | Drawing completed interrupt disabled<br>Drawing completed interrupt enabled |
| INTM[1] | CL_MSK | 0<br>1 | Clip interrupt disabled<br>Clip interrupt enabled |
| INTM[31:2] | Reserved | 0x0 | Reserved |

### ~~5.1.3~~5.8.3 Flow Control Register RBASE_D +(0x0008)

Формат: Список

**Name: FLOW**

**Type: read only**



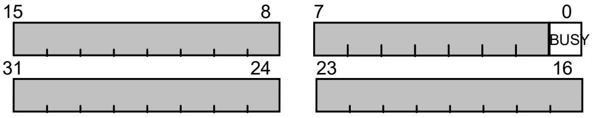
| Bits | Name | Value | Function |
|---|---|---|---|
| FLOW[0] | DEB | 0<br>1 | Drawing engine is not busy<br>Drawing engine is busy |
| FLOW[1] | MCB | 0<br>1 | Memory controller is not busy<br>Memory controller is busy |
| FLOW[2] | CLP | 0<br>1 | No clipping was applied to previous drawing command<br>Clipping was applied to previous drawing command |
| FLOW[3] | PRV | <br>0<br>1<br><br>0<br>1 | During a **write transfer command**, this bit is interpreted as follows:<br>The drawing engine cache is not ready to accept new data<br>The drawing engine cache can accept new data (at least half the cache is empty)<br>During **all other commands** this bit is interpreted as follows:<br>The previously triggered command has completed<br>The previously triggered command is still executing |
| FLOW[4] | RPB | | Renderer Pipeline or Pixel Cache both busy. 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. |
| | | 0 | Renderer Pipeline and Pixel Cache both not busy. |

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

### ~~5.1.4~~5.8.4 BUSY Register RBASE_D +(0x000C)

**Name: BUSY**

**Type: memory mapped read only**



| Bits | Name | Value | Function |
|---|---|---|---|
| BUSY[0] | BUSY | 0<br>1 | IMAGINE 128® is ready to accept a new command. *Command pipeline not busy*<br>IMAGINE 128® is not ready to accept a new command. *Command pipeline busy* |

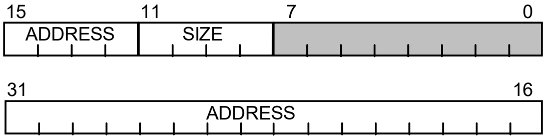
### ~~5.1.5~~5.8.5 XY Window Address RBASE _D + (0x0010)

Формат: Список

**Name: XYW_AD**

**Type: Memory mapped read write**

This register defines the system address range that the host interface will decode for an X-Y memory window. Before programming this register, the EXA (EXB) bit in CONFIG1 should be set to 0, preventing spurious X-Y window decode.

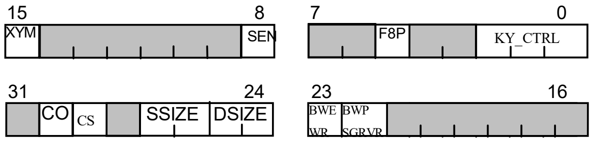


| Bits | Name | Value | Function |
|---|---|---|---|
| XYW_AD[11:8] | SIZE | 0x0<br>0x1<br>0x2<br>...<br>0xC<br>0xD<br>0xE<br>0xF | X-Y Window size equals 4K<br>X-Y Window size equals 8K<br>X-Y Window size equals 16K<br>...<br>X-Y Window size equals 16MB<br>X-Y Window size equals 32 MB<br>RESERVED<br>RESERVED |
| XYW_AD[31:12] | ADDRESS | | Address decode value |

### ~~5.1.6~~5.8.6 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. ~~Contained within this register are the source and destination control bits.~~

The source enable bits (SEN) select ~~which of the local buffers is the source of data during a read cycle. It is also possible to select IMAGINE 128's internal data cache as the source of data.~~whether Silverhammer's internal Drawing engine cache is selected, or if the on board memory is selected.

~~The destination enable bits (DEN) select which of the local buffers is the destination of data during a write cycle. During read-modify-write cycles, the DEN bits specify both the source and destination of the data.~~

~~The two bits, texture enable (TEN) and Z enable (ZEN), select whether the texture and Z buffers are stored in the primary buffer, or the virtual buffer.~~

~~Multiple buffers may be written to simultaneously by enabling the appropriate shadow enable bit. There are two shadow enable bits, one corresponding to each of the local buffers. Any combination of shadow enable bits is allowed.~~

Cache On (CO) is meaningful only when the source data is to be read from IMAGINE 128'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 Source and Destination Pixel Size bits define the pixel depth for the current drawing operation. From here, the destination framebuffer size can be selected, as well as the source data format for 3D operations. Texture cache formats are defined seperately in the TEX_CTRL register. It is important to note that all internal Silverhammer pixel calculations are performed at 32 bits per pixel and dithered to 16, if necessary, prior to writing to memory.

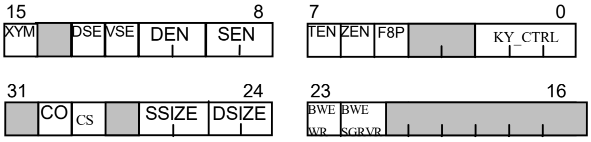
The Key Control field is used in conjunction with the DE_KEY register to set up a 2D transparency operation. 3D keying is selected in the 3D_CTRL register and is independent of 2D color keying. 2D keying is only reliable if not doing a 3D operation involving specular, fog, or filtering, and 3D keying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

The Cache Select (CS) bit is used to select host bus writing access to either the XY windows or the ~~Texel cache (or palette)~~user programmable Fog Tables. ~~The Texture Cache memory is divided into texture and palette regions. Bit 13 selects texture (0) or palette (1). The palette consists of 32 bytes of memory which wrap to fill the entire 8KB region.~~

**Buffer Control Register RBASE _D + (0x0020) (Continued)**

**Name: BUF_CTRL**

**Type: Memory mapped read write**



| Bits | Name | Value | Function |
|---|---|---|---|
| BUF_CTRL[2:0] | KY_CTRL | <br>0xx<br>100<br>101<br>110<br>111 | Key Control<br>No Keying<br>Source Keying, mask on key color<br>Destination Keying, mask on key color<br>Source Keying, mask on NOT key color<br>Destination Keying, mask on NOT key color |
| BUF_CTRL[4:3] | Reserved | 0 | Reserved |
| BUF_CTRL[5] | F8P | <br>0<br>1 | Force 8 Page Mode.<br>Drawing Engine runs 8 & 16 page accesses<br>Drawing Engine Runs 8 page access only |
| BUF_CTRL[7:6] | ~~ZEN~~Reserved | <br>~~0~~<br>~~1~~0x0 | ~~Z buffer selector~~<br>~~Enable Z buffer in the primary buffer~~<br>~~Enable Z buffer in virtual buffer~~Reserved |
| BUF_CTRL[~~9:~~8] | SEN~~[1:0]~~ | <br>0x0<br>0x1~~1~~<br>~~0x2~~<br>~~0x3~~ | Source Read Enable.<br>Enable reads from ~~the Display buffer.~~external memory.<br>~~Enable reads from the Virtual buffer.~~<br>~~Reserved~~<br>Enable reads from IMAGINE 128❸ cache. |
| BUF_CTRL[14:9] | Reserved | 0x0 | Reserved |
| BUF_CTRL[15] | XYM | <br>0x0<br>0x1 | Origin Mode<br>Normal linear origin mode<br>XY origin mode |
| BUF_CTRL[21:16] | Reserved | 0x0 | Reserved |
| BUF_CTRL[22] | BWESGRVR | <br>0x0<br>0x1 | Block Write Enable (For SGRAM and VRAM)<br>Block writes disabled<br>Block writes enabled |
| BUF_CTRL[23] | BWEWR | <br>0x0<br>0x1 | Block Write Enable (For WRAM)<br>Block writes disabled<br>Block writes enabled |
| BUF_CTRL[25:24] | DSIZE[1:0] | <br><br>0x0<br>0x1<br>0x2<br>0x3 | Destination Data Pixel Size and Format - Number of Bits per Pixel.<br>Eight bits per pixel<br>Sixteen bits per pixel - 1555<br>Thirty-two bits per pixel<br>Sixteen bits per pixel - 565 |

## Buffer Control Register RBASE D + (0x0020) (Continued)

**Name: BUF_CTRL**

**Type: Memory mapped read write**

| 15 | | | | | 8 |
|---|---|---|---|---|---|
| XYM | ▨ | DSE | VSE | DEN | SEN |

| 7 | | | | 0 |
|---|---|---|---|---|
| TEN | ZEN | F8P | ▨ | KY_CTRL |

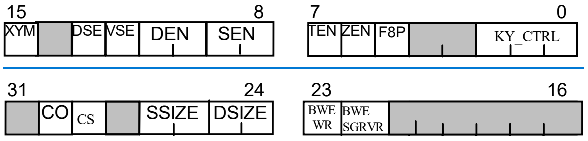
| 31 | | | | | 24 |
|---|---|---|---|---|---|
| ▨ | CO | CS | ▨ | SSIZE | DSIZE |

| 23 | | 16 |
|---|---|---|
| BWE WR | BWE SGRVR | ▨ |

**~~Buffer Control Register RBASE_D + (0x0020)~~ ~~*Continued*~~**

**~~Name: BUF_CTRL~~**

**~~Type: Memory mapped read write~~**

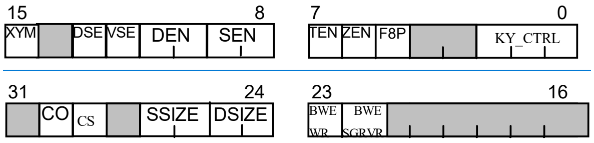


| | | | |
|---|---|---|---|
| BUF_CTRL[27:26] | SSIZE[1:0] | <br>0x0<br>0x1<br>0x2<br>0x3 | Source Pixel Size and Format - Number of Bits per Pixel. (Non Texel Source for Drawing Engine)<br>Eight bits per pixel<br>Sixteen bits per pixel - 1555<br>Thirty-two bits per pixel<br>Sixteen bits per pixel - 565 |
| BUF_CTRL[28] | Reserved | 0x0 | Reserved |
| BUF_CTRL[29] | CS | <br>0<br>1 | Cache Select<br>XY Windows<br>User Programmable Fog Tables |
| BUF_CTRL[30] | CO | <br>0<br>1 | Cache On<br>This bit should be set when source data is read from the cache and the cache contains the appropriate data. This bit is **not reset** by the drawing engine after a command completes, therefore the cache is always assumed to have valid data when CO is set. This setting is useful when the cache is partitioned into halves by software.<br>Cache has not yet been written with the correct data<br>Cache contains appropriate data for the next command |
| BUF_CTRL[31] | Reserved | 0x0 | Reserved |

~~**Buffer Control Register RBASE _D + (0x0020)**~~ ~~***Continued***~~

~~**Name: BUF_CTRL**~~

~~**Type: Memory mapped read write**~~

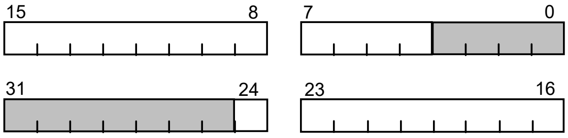


### ~~5.1.7~~5.8.7 Drawing Engine Source Origin RBASE_D + (0x0028)

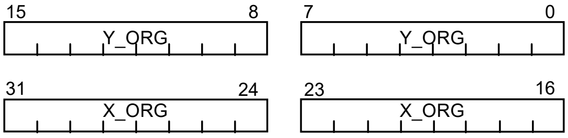
**Name: DE_SORG**

**Type: Memory mapped read write**

This register defines the linear or XY address offset into the selected buffer or buffers for drawing engine read cycles. In linear mode this is a 128 bit word address when accessing the ~~Virtual or~~ Display Buffers. In XY mode this is a pixel address offset.



| Bits | Name | Value | Function |
|---|---|---|---|
| DE_SORG[24:4] | SORGL | | Linear origin mode |



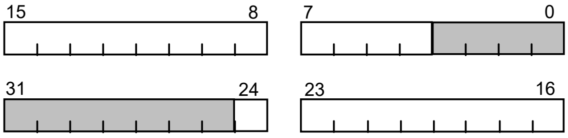
| Bits | Name | Value | Function |
|---|---|---|---|
| DE_SORG[31:0] | SORGXY | | XY origin mode |

### ~~5.1.8~~5.8.8 Drawing Engine Destination Origin RBASE_D + (0x002C)

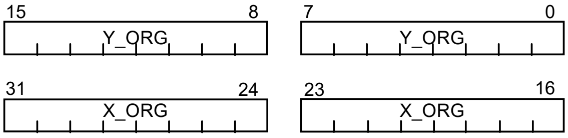
**Name: DE_DORG**

**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 128 bit word address when accessing the ~~Virtual or~~ Display Buffers. In XY mode this is a pixel address offset.



| Bits | Name | Value | Function |
|---|---|---|---|
| DE_DORG[24:4] | DORGL | | Linear origin mode. |



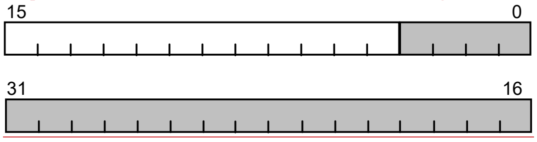
| Bits | Name | Value | Function |
|---|---|---|---|
| DE_DORG[31:0] | DORGXY | | XY origin mode. |

### 5.8.9 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.



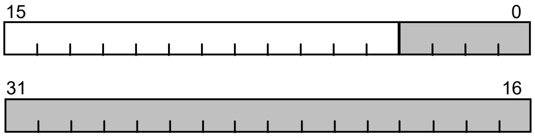
### ~~5.1.10~~5.8.10 DE Source Pitch RBASE_D + (0x003C)

**Формат:** Список

**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.



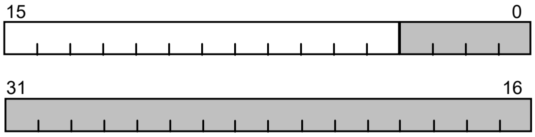
### 5.8.11

### 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.



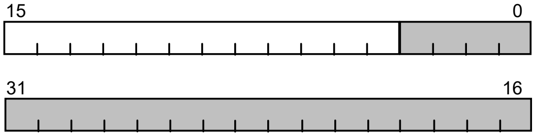
### ~~5.1.12~~5.8.12 DE Destination Pitch RBASE_D + (0x0044)

**Формат:** Список

**Name: DE_DPTCH**

**Type: Memory mapped read write**

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

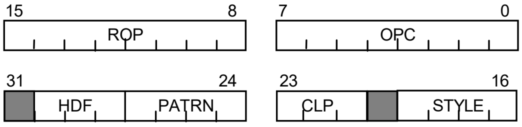


### ~~5.1.13~~5.8.13 Command Register RBASE_D +(0x0048)

**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[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 |

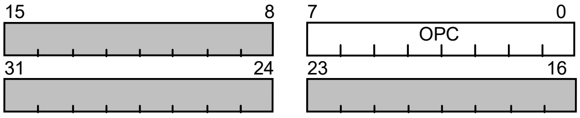
### 5.8.14

**Формат:** Список

### ~~5.1.14~~5.8.15 Command Opcode RBASE_D +(0x0050)

**Name:** **CMD_OPC**

**Type:** **Memory mapped read write**



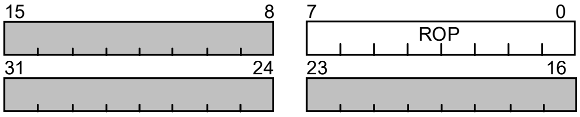
| Value | Name | Function |
|---|---|---|
| 0x00 | NOOP | Transfer parameters no drawing |
| 0x01 | BITBLT | BIT BLT command |
| 0x02 | LINE | Line command |
| 0x03 | ELINE | Error Line |
| 0x04 | Reserved | Reserved |
| 0x05 | PLINE | Poly line command. |
| 0x06 | RXFER | Host X-Y window read image transfer |
| 0x07 | WXFER | Host X-Y window write image transfer |
| 0x08 | LINE_3D | 3D Line command, Texture mapped, Z buffer, Gouraud shaded, setup |
| 0x09 | TRIAN_3D | 3D Triangle Command, Full floating point setup, Texture Mapped, Perspective Corrected, Z buffered, Gouraud Shaded. |
| 0x0A | TEX_INV~~LD_TEX~~ | ~~Texture Load Command~~Invalidate Texture Cache |
| 0x0B | LD_PAL | Palette Load Command |
| 0x0C---0xFF | Reserved | For future expansion (no action taken) |

### ~~5.1.15~~5.8.16 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.



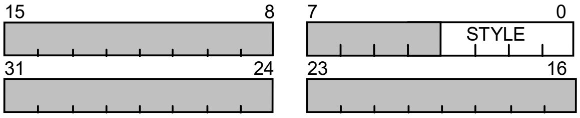
| Value | Name | Function |
|---|---|---|
| 0x00 | clear | 0 (zero) |
| 0x01 | nor | (NOT source) AND (NOT destination) |
| 0x02 | andInverted | (NOT source) AND destination |
| 0x03 | copyInverted | (NOT source) |
| 0x04 | andReverse | source AND (NOT destination) |
| 0x05 | invert | (NOT destination) |
| 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 |

### ~~5.1.16~~5.8.17 Line/Fill Style Register RBASE_D +(0x0058)

Формат: Список

**Name: CMD_STYLE**

**Type: Memory mapped read write**



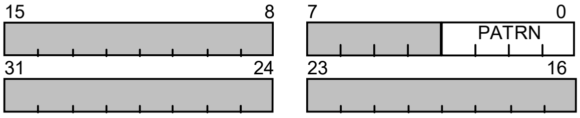
| Bits | Name | Value | Function |
|---|---|---|---|
| CMD_STYLE[0] | SOLID | | Solid<br>When SOLID is set to one the source of the data for all operations is the FORE register. |
| CMD_STYLE[1] | TRNSP | | Transparency<br>When TRNSP is set to one during a line , stippled bit blt, stippled write transfer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

### ~~5.1.17~~5.8.18 Patterning Register RBASE_D +(0x005C)

**Формат:** Список

**Name: CMD_PATRN**

**Type: Memory mapped read write**

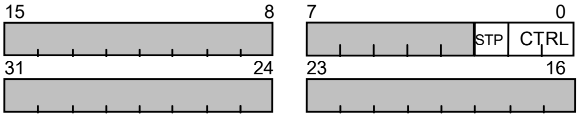


| Bits | Name | Function |
|---|---|---|
| CMD_PATRN[1:0] | APAT[1:0] | Area Pattern<br>When APAT is set to two it forces the source area pattern to be a 32x32 screen locked pattern. |
| | 0x0 | Area pattern mode is off. |
| | 0x1 | Area pattern mode equals 8x8. |
| | 0x2 | Area pattern mode equals 32x32.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

### ~~5.1.18~~5.8.19 Clipping Control Register RBASE_D +(0x0060)

**Name: CMD_CLP**

**Type: Memory mapped read write**



| Bits | Name | Value | Function |
|---|---|---|---|
| CMD_CLP[1:0] | CCTRL[1:0] | 0x0<br>0x1<br>0x2<br>0x3 | No clipping<br>No clipping<br>Draw inside rectangle<br>Draw outside rectangle |
| CMD_CLP[2] | CSTOP | 0<br>1 | Don't stop on clip boundary<br>Stop on clip boundary |

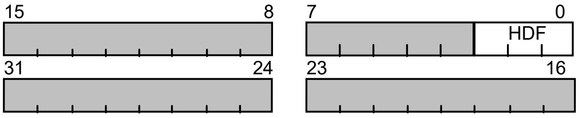
### ~~5.1.19~~5.8.20 Host Data Format Register RBASE_D +(0x0064)

**Формат:** Список

**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 byte order from 3-2-1-0 to 0-1-2-3. 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<br>0x1 | Bit ordering is not modified<br>Swap bits within each byte |
| CMD_HDF[1] | BYTE_SWAP | 0x0<br>0x1 | Byte ordering is not modified<br>Swap bytes within each word |
| CMD_HDF[2] | WORD_SWAP | 0x0<br>0x1 | Word ordering is not modified<br>Swap words within each dword |

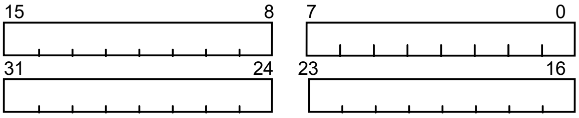
### ~~5.1.20~~5.8.21 Foreground RBASE_D +(0x068)

Формат: Список

**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.



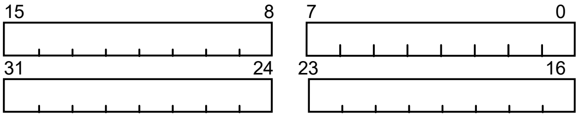
### ~~5.1.21~~5.8.22 Background RBASE_D +(0x006C)

Формат: Список

**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.



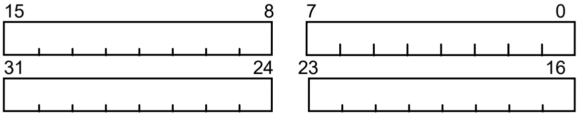
### ~~5.1.22~~5.8.23 Plane Mask RBASE_D +(0x0070)

Формат: Список

**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.



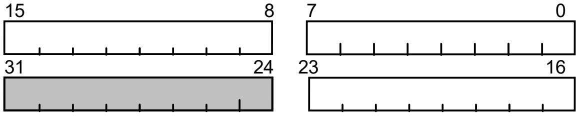
### ~~5.1.23~~5.8.24 Color Key Register Mask RBASE_D +(0x0074)

Формат: Список

**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.



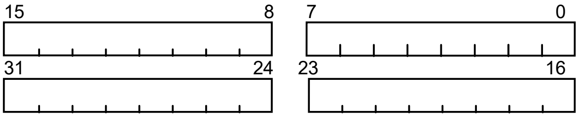
### ~~5.1.24~~5.8.25 Line Pattern Register RBASE_D +(0x0078)

Формат: Список

**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.

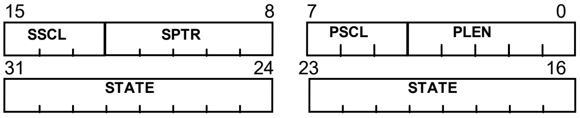


### ~~5.1.25~~5.8.26 Line Pattern Control Register RBASE_D +(0x007C)

**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.



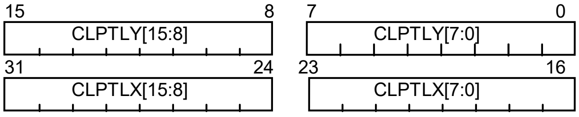
| Bits | Name | Value | Function |
|---|---|---|---|
| PCTL[4:0] | PLEN | Pattern length. The length assumes a scale factor of 1x.<br>0x0<br>0x1<br>...<br>0x1f | <br>32 bit long pattern<br>1 bit long pattern<br><br>31 bit long pattern |
| PCTL[7:5] | PSCL | Pattern scale factor (stretch)<br>0x0<br>0x1<br>...<br>0x7 | <br>x1 scaling<br>x2 scaling<br><br>x8 scaling |
| PCTL[12:8] | SPTR | Pattern offset<br>0x00<br>0x01<br>...<br>0x1f | <br>1st bit of pattern is LPAT[0]<br>1st bit of pattern is LPAT[1]<br><br>1st bit of pattern is LPAT [31] |
| PCTL[15:13] | SSCL | Initial scale offset<br>0x0<br>0x1<br>...<br>0x7 | <br>no offset<br>offset by one<br><br>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. | |

### ~~5.1.26~~5.8.27 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. |

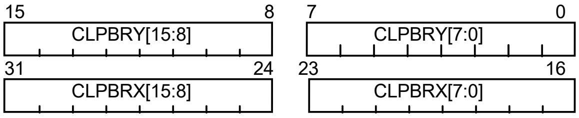
### ~~5.1.27~~5.8.28 Bottom Right of Clip Area RBASE_D + (0x0084)

**Формат:** Список

**Name: CLPBR**

**Type: Memory mapped read write**

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



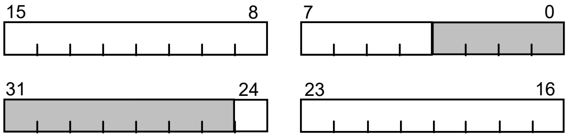
| Bits | Name | Function |
|---|---|---|
| CLPBR[15:0] | CLPBRY | Y coordinate of the bottom right corner of the clipping rectangle. |
| CLPBR[31:16] | CLPBRX | X coordinate of the bottom right corner of the clipping rectangle. |

### ~~5.1.28~~5.8.29 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 128 bit word address when accessing the ~~Virtual or~~ Display Buffer.



| Bits | Name | Value | Function |
|---|---|---|---|
| DE_ZORG[24:4] | ZORGL | | Linear origin mode |

### ~~5.1.29~~5.8.30 Drawing Engine MipMap Origins RBASE_D + (0x0104 through 0x0114)

**Формат:** Список

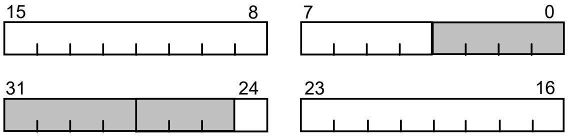
**Name: LOD0_ORG,LOD2~~1~~_ORG,LOD4~~2~~_ORG,LOD6~~3~~_ORG,LOD0_SIZE~~4_ORG~~**

**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. This is a 128 bit word address when accessing the Virtual or Display Buffer. 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 3D_CNTRL2 and the number of mipmaps also specified. Mipmaps must be loaded sequentially, no gaps are allowed. SilverHammer calculates the origins of intermediate mipmaps internally. This allow for up to 10 LOD's to be used (512x512 to 1x1) when mipmapping is enabled. The following table shows how to load the mipmap origins, and what Silverhammer provides for additional setup of mipmaps.

| Parameter | Action |
|---|---|
| LOD0_ORG | User supplied LOD0_ORG |
| LOD1_ORG | Internally calculated |
| LOD2_ORG | User Supplied |
| LOD3_ORG | Internally calculated |
| LOD4_ORG | User Supplied |

| Parameter | Action |
|---|---|
| LOD5_ORG | Internally calculated |
| LOD6_ORG | User Supplied |
| LOD7_ORG | Internally calculated |
| LOD8_ORG | Internally calculated |
| LOD9_ORG | Inteernally calculated |



| Bits | Name | Value | Function |
|---|---|---|---|
| DE_LOD0ORG[24:4] | LOD0ORGL | | Linear origin mode |

### 5.8.31

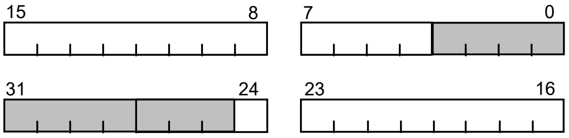
### 5.8.32 –Drawing Engine Palette Origin RBASE_D + (0x0118)

**Формат:** Список

**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 ~~Virtual or~~ Display Buffer.



| Bits | Name | Value | Function |
|---|---|---|---|
| DE_TPALORG[24:4] | TPAL_ORG | | Linear origin mode |

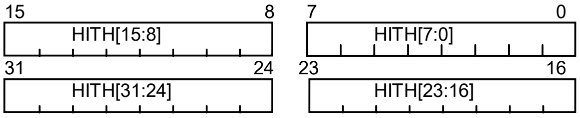
### ~~5.1.31~~5.8.33 HITHER Clip Plane RBASE_D + (0x011C)

**Формат:** Список

**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. |

* bits 31:0 for 32 bit Z, 23:0 for 24 bit Z, or 15:0 for 16 bit Z.

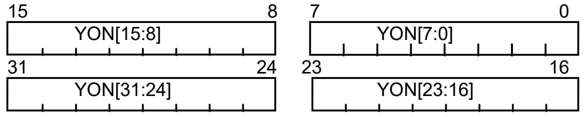
**Формат:** Список

### ~~5.1.32~~5.8.34 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. |

* bits 31:0 for 32 bit Z, 23:0 for 24 bit Z, or 15:0 for 16 bit Z.

### 5.8.35

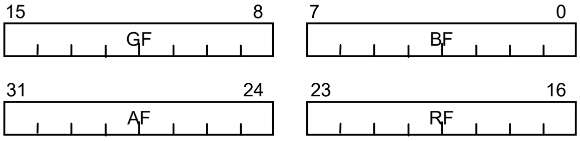
### 5.8.36 Fog Color Register RBASE_D + (0x0124)

**Формат:** Список

**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 |

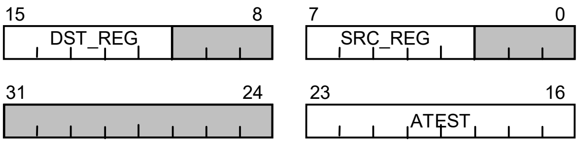
**Формат:** Список

### ~~5.1.34~~5.8.37 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 and the source and destination alpha override register for blending.



| 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 |

### 5.8.38

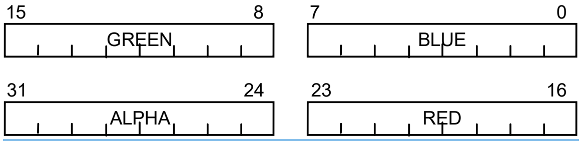
### 5.8.39 Texture Boarder Color Register RBASE_D + (0x012C)

Формат: Список

**Name:** TBOARD_COL

**Type: Memory mapped read write**

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

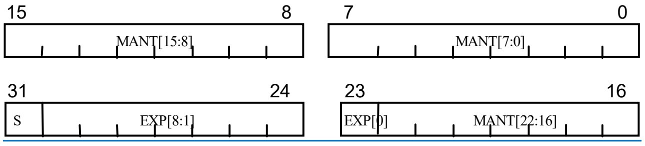


### 5.8.40 Floating Point Interface to the color registers RBASE_D + (0x0130 – 0x015C)

**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 SilverHammer, 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 read only, and they cause the three vertex color registers to be loaded with the new color value when written to with the enable bit located in 3D_CTRL set to one.



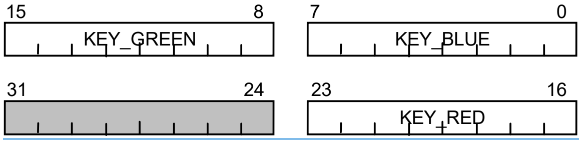
### 5.8.41 3D Color key compare registers RBASE_D + (0x0160, 0x0164)

Формат: Список

**Name: KEY_3D_LOW, KEY_3D_HIGH**

**Type: Memory mapped read write**

SilverHammer 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. Note:

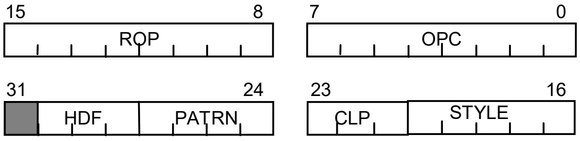


### ~~5.8.31~~5.8.42 Command Register RBASE_D +(0x0168)

**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.



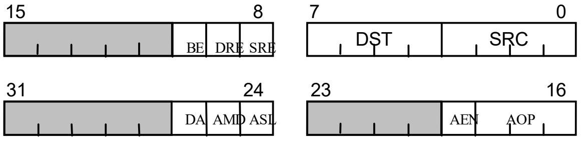
| Bits | Name | Function |
|---|---|---|
| CMD[7:0] | OPC | Drawing command opcode |
| CMD[15:8] | ROP | Drawing Raster or logical operation |
| CMD[20:16] | STYLE | Solid, Transparency, Stipple control |
| CMD[23:21] | CLP | Clipping control |
| CMD[27:24] | PATRN | No last, pat reset, area pattern,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

### ~~5.1.39~~5.8.43 Command Register RBASE_D +(0x016C)

**Name: ACNTRL**

**Type: Memory mapped read write**

The alpha control register sets up the Imagine 3 blending unit for use with the drawing engine commands.



| Bits | Name | Value | Function |
|---|---|---|---|
| ACNTRL[3:0] | SRC | see below | Blending Source Function |
| ACNTRL[7:4] | DST | see below | Blending Destination Function |
| ACNTRL[8] | SRE | <br>0x0<br>0x1 | Source Register Enable<br>Use Alpha in the color<br>Use source alpha register |
| ACNTRL[9] | DRE | <br>0x0<br>0x1 | Destination Register Enable<br>Use Alpha in the color<br>Use destination alpha register |
| ACNTRL[10] | BE | <br>0x0<br>0x1 | Blending Enable<br>Disable blending<br>Enable blending |
| ACNTRL[15:11] | Reserved | 0x0 | Reserved |
| ACNTRL[18:16] | AOP | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | Alpha operator<br>Never.<br>Always.<br>Less than.<br>Less than or equal.<br>Equal.<br>Greater than or equal.<br>Greater than.<br>Not Equal. |
| ACNTRL[19] | AEN | <br>0x0<br>0x1 | Alpha Compare<br>Disabled<br>Enabled |
| ACNTRL[23:20] | Reserved | 0x0 | Reserved |
| ACNTRL[24] | ASL | <br>0x0<br>0x1 | Alpha Select<br>Texture Alpha<br>Interpolated Alpha |
| ACNTRL[25] | AMD | <br>0x0<br>0x1 | Alpha Modulate<br>Disabled<br>Enabled |
| ACNTRL[26] | DA | <br>0x0<br>0x1 | Decal Alpha Mode<br>Disabled<br>Enabled |
| ACNTRL[31:27] | Reserved | 0x0 | Reserved |

The Imagine 3 alpha blending modes follow the standard OpenGL conventions which are outlined in the following two tables:

**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) |

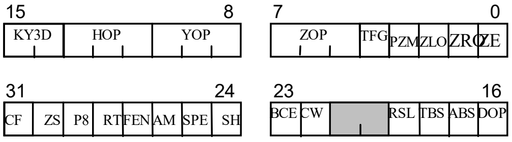
**Destination Factor Table**

| 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) |
| 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) |

### ~~5.1.40~~5.8.44 3D Control Register RBASE_D +(0x0170)

**Name: 3D_CTRL**

**Type: Memory mapped read write**

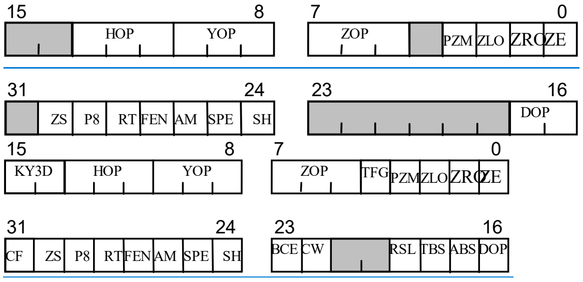


| Bits | Name | Value | Function |
|---|---|---|---|
| 3D_CTRL[0] | ZE | <br>0x0<br>0x1 | Z buffer enable bit.<br>No Z buffer update or compare.<br>Z buffer compare enabled, and updated if ZRO = 0. |
| 3D_CTRL[1] | ZRO | <br>0<br>1 | Z read only mode.<br>Updating of Z buffer enabled.<br>Z buffer in read only mode. |
| 3D_CTRL[2] | ZLO | <br>0x0<br>0x1 | Low resolution Z (32bpp only)<br>Z is 32 bpp (User must set up Z and supply gradients)<br>Z is 16 or 24 bpp (Z is set up and perspectively corrected in 16 bit mode, only set up in 24 bit mode) |
| 3D_CTRL[3] | PZM | <br>0x0<br>0x1 | Perspective Z mode<br>Z is Orthogonal, W must be supplied and set to 1<br>Z is perspectively corrected (only 16 bit Z supported) |
| 3D_CTRL[4] | ~~Reserved~~FSL | ~~0x0~~<br>0x0<br>0x1 | ~~Reserved~~Fog Selector<br>Vertex Fogging<br>1/W Table based Fogging |
| 3D_CTRL[7:5] | ZOP | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | Z operator<br>Never.<br>Always.<br>Less than.<br>Less than or equal.<br>Equal.<br>Greater than or equal.<br>Greater than.<br>Not Equal. |
| 3D_CTRL[10:8] | YOP | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | YON operator<br>Never.<br>Always.<br>Less than.<br>Less than or equal.<br>Equal.<br>Greater than or equal.<br>Greater than.<br>Not Equal. |

**3D Control Register RBASE_D + (0x0170) *Continued***

**Name: 3D_CTRL**

**Type: Memory mapped read write**



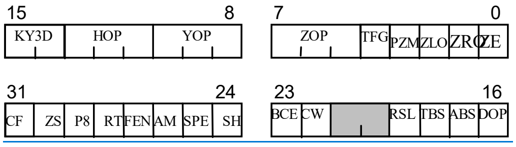
| | | | |
|---|---|---|---|
| 3D_CTRL[13:11] | HOP | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7 | Hither operator<br>Never.<br>Always.<br>Less than.<br>Less than or equal.<br>Equal.<br>Greater than or equal.<br>Greater than.<br>Not Equal. |
| 3D_CTRL[15:14] | ~~Reserved~~KY3 | <br>0x00<br>0x01<br>0x10<br>0x11 | ~~Reserved~~3D Keying<br>Keying Disabled<br>Keying Disabled<br>Inclusive (In range)<br>Exclusive (Out of Range)<br>NOTE: 3D and 2D keying are not exclusive. |
| 3D_CTRL[~~17:~~16] | DOP | <br>0x0<br>~~0x1~~<br>~~0x2~~<br>0x1~~3~~ | Dither Operator<br>No Dithering<br>~~2x2~~<br>~~4x4~~<br>8x8 |
| 3D_CTRL[17] | ABS~~BCE~~ | See Below<br>~~0x0~~<br>~~0x1~~ | ~~Backface Culling~~Alpha Blend Select<br>~~Disabled~~<br>~~Enabled~~ |
| 3D_CTRL[18] | TBS | See Below~~0x0~~ | Texture Blend Select |
| 3D_CTRL[19] | RSEL | See Below~~0x0~~ | RGB Select |
| 3D_CTRL[21:20] | Reserved | 0x0 | Reserved |
| 3D_CTRL[22] | CW | <br>0x0<br>0x1 | CW/CCW selector for culling<br>CW<br>CCW |
| 3D_CTRL[23~~3:2018~~] | ~~Reserved~~BCE | <br>0x0<br>0x1 | ~~Reserved~~Backface culling Enable<br>Disabled<br>Enabled |

| | | | |
|---|---|---|---|
| 3D_CTRL[24] | SH | <br>0x0<br>0x1 | Gouraud Shading<br>Disabled<br>Enabled |
| 3D_CTRL[25] | SE | <br>0x0<br>0x1 | Specular Lighting<br>Disabled<br>Enabled |
| 3D_CTRL[26] | AMSK | <br>0x0<br>0x1 | Alpha Mask for Fog<br>Alpha is not fogged<br>Alpha is Fogged |

### 3D Control Register RBASE_D + (0x0170) *Continued*

**Name: 3D_CTRL**

**Type: Memory mapped read write**



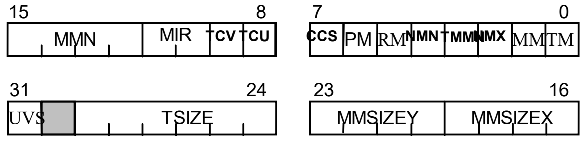
| | | | |
|---|---|---|---|
| 3D_CTRL[27] | FEN | <br>0x0<br>0x1 | Fog<br>Disabled<br>Enabled |
| 3D_CTRL[28] | RT | <br>0x0<br>0x1 | Rectangle<br>Draw triangle in normal mode.<br>Draw rectangle (P0, P1 and P2 must be sorted) |
| 3D_CTRL[29] | P8 | <br>0x0<br>0x1 | 8bpp Palettized video support<br>8bpt normal mode<br>8bpt palette mode* |
| 3D_CTRL[30] | ZS | <br>0x0<br>0x1 | Z Scaling mode<br>Z is not scaled by Imagine 3<br>Z is scaled by Imagine 3 |
| 3D_CTRL[31] | ~~RESERVED~~CF | <br>0x0<br>0x1 | ~~RESERVED~~Color Format<br>ARGB loaded in CP5,13,21<br>Floating point colors loaded into 0x12C-0x158 |

*** TEX_CTRL [28:24] (TSIZE) must be set to 0 x 0D (8bpt 0332) texture map pixel format.**

### ~~5.1.41~~5.8.45 Texture Mapping Control Register RBASE _(D) + (0x0174)

**Name:** **TEX_CNTRL**

**Type:** **Memory mapped read write**

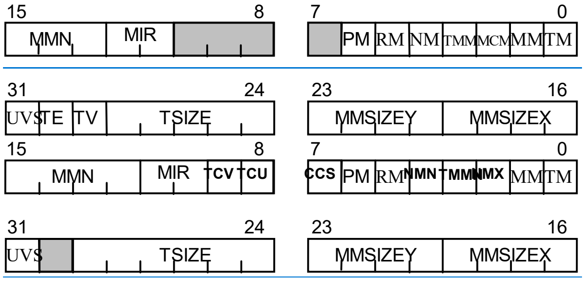


| Bits | Name | Value | Function |
|---|---|---|---|
| TEX_CTRL[0] | TM | <br>0x0<br>0x1 | Texture Mode<br>Texture mapping disabled<br>Texture mapping enabled |
| TEX_CTRL[1] | MM | <br>0x0<br>0x1 | Mipmap mode<br>Mipmapping disabled<br>Mipmapping enabled |
| TEX_CTRL[2] | ~~MCM~~NMX | <br>0x0<br>0x1 | ~~Mipmap Correction~~Nearest Mode Magnification<br>Disabled<br>Enabled |
| TEX_CTRL[3] | TMMN | <br>0x0<br>0x1 | Trilinear Mipmapping<br>Disabled<br>Enabled |
| TEX_CTRL[4] | NMN | <br>0x0<br>0x1 | Nearest Mode<br>Disabled<br>Enabled |
| TEX_CTRL[5] | RM | <br>0x0<br>0x1 | RGB Modulation<br>Disabled<br>Enabled |
| TEX_CTRL[6] | PM | <br>0x0<br>0x1 | Perspective Correction Mode<br>Disabled<br>Enabled |
| TEX_CTRL[7] | CCS | <br>0x0<br>0x1 | Clamp Color Selector<br>Use Last Pixel<br>Use Boarder Color |
| TEX_CTRL[8] | TCU | <br>0x0<br>0x1 | Texture Clamp in U<br>Disabled<br>Enabled |
| TEX_CTRL[9] | TCV | <br>0x0<br>0x1 | Texture Clamp in V<br>Disabled<br>Enabled |
| TEX_CTRL[11:10] | MIR | <br>0x0<br>0x1<br>0x2<br>0x3 | Texture Mirror Mode<br>No mirroring<br>Mirror X<br>Mirror Y<br>Mirror X & Y |

**Texture Mapping Control Register RBASE_D +(0x0174)** ***Continued***

**Name: TEX_CNTRL**

**Type: Memory mapped read write**

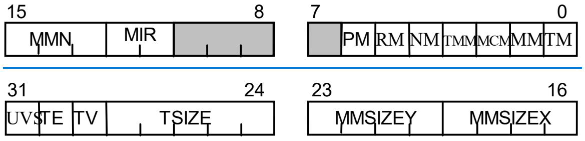


| | | | |
|---|---|---|---|
| TEX_CTRL[15:1~~3~~2] | MMN | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5 - 0x7 | Number of Mipmaps<br>1 mipmap<br>2 mipmaps<br>3 mipmaps<br>4 mipmaps<br>5 mipmaps<br>Reserved |
| TEX_CTRL[19:16] | MMSIZEX | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7<br>0x8<br>0x9<br>0x10- 0x15 | Mip Map X size<br>1 texel<br>2 texels<br>4 texels<br>8 texels<br>16 texels<br>32 texels<br>64 texels<br>128 texels<br>256 texels<br>512 texels<br>Reserved |
| TEX_CTRL[23:20] | MMSIZEY | <br>0x0<br>0x1<br>0x2<br>0x3<br>0x4<br>0x5<br>0x6<br>0x7<br>0x8<br>0x9<br>0x10- 0x15 | Mip Map Y size<br>1 texel<br>2 texels<br>4 texels<br>8 texels<br>16 texels<br>32 texels<br>64 texels<br>128 texels<br>256 texels<br>512 texels<br>Reserved |

**Texture Mapping Control Register RBASE_D +(0x0174)** ***Continued***

~~**Name: TEX_CNTRL**~~

~~**Type: Memory mapped read write**~~

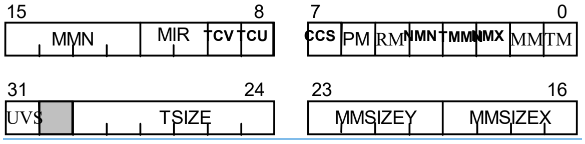


| | | | |
|---|---|---|---|
| TEX_CTRL[29~~8~~:24] | TSIZE | | Texture map pixel format |
| | | | Palettized |
| | | 0x0 | 1 bpt 1555 |
| | | 0x1 | 1 bpt 0565 |
| | | 0x2 | 1 bpt 4444 |
| | | 0x3 | 1 bpt 8888 |
| | | 0x18 | 1 bpt 8332 |
| | | 0x4 | 2 bpt 1555 |
| | | 0x5 | 2 bpt 0565 |
| | | 0x6 | 2 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 |
| | | 0x1A | 8 bpt 1555 |
| | | 0xF | 8 bpt 0565 |
| | | 0x1C | 8 bpt 4444 |
| | | 0x1D | 8 bpt 8888 (Exact mode, must set nearest) |
| | | 0x3E | 8 bpt 8888 (auto convert to 4444) |
| | | 0x3F | 8 bpt 8888 (auto convert to 0565) |
| | | 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] | ~~TEB~~FIS | | ~~Texture in cache and completely loaded~~ |
| | | | ~~False~~ |
| | | ~~0x0~~0x0 | ~~True~~Fog Index Selector |
| | | | Z |
| | | ~~0x1~~0x1 | W |

## Texture Mapping Control Register RBASE_D +(0x0174) *Continued*

**Name: TEX_CNTRL**

**Type: Memory mapped read write**



| | | | |
|---|---|---|---|
| TEX_CTRL[31] | UVS | <br>0x0<br>0x1 | U-V Scaling<br>Imagine 3 won't scale U and V<br>Imagine 3 will scale U and V by texture extent |

### ~~5.8.42~~5.8.46 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.

### ~~5.8.43~~5.8.47 XY Parameter Registers RBASE_D + (0x0088 through 0x0098)

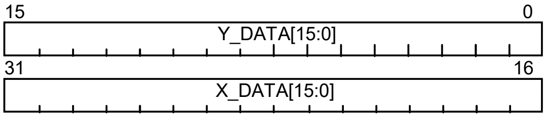
**Формат:** Список

**Name: XY0, XY1, XY2, XY3, XY4**

**Type: Memory mapped read write**

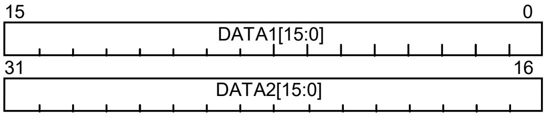
The XY parameter registers are general purpose registers used to hold different data values based on the type of command to be executed. The Command set chapter describes the use of these registers for each command. The data in these registers is one of two data formats.

X-Y FORMAT



| Bits | Function |
|---|---|
| Y_DATA[15:0] | Y coordinate of a pixel location, or the Y portion of a dimension.<br>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.<br>X_DATA is an integer value with range of +32767 to -32767. |

I-I FORMAT



The I-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 word, the most significant byte of XY1 must be written for a command to be triggered.

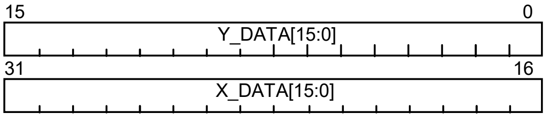
### ~~5.8.44~~5.8.48 CP Parameter Registers RBASE_D + (0x0178 through 0x01D8~~, 0x01E4 through 0x01F8~~)

**Name: CP0 through CP24~~30~~**

**Type: Memory mapped read write**

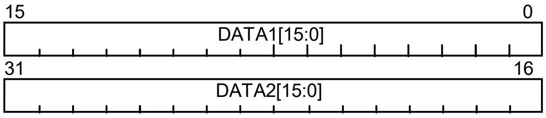
The CP parameter registers are general purpose registers used to hold different data values based on the type of command to be executed. The Command set chapter describes the use of these registers for each command. The data in these registers is one of two data formats.

X-Y FORMAT



| Bits | Function |
|---|---|
| Y_DATA[15:0] | Y coordinate of a pixel location, or the Y portion of a dimension.<br>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.<br>X_DATA is an integer value with range of +32767 to -32767. |

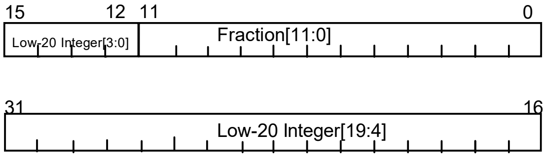
I FORMAT



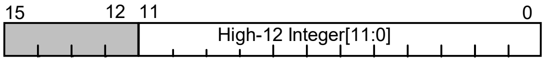
The I format contains two general purpose 16 bit integer data values. (Data can be less than 16 bits as in direction bits for bit blt).

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

I-F Format (20.12)



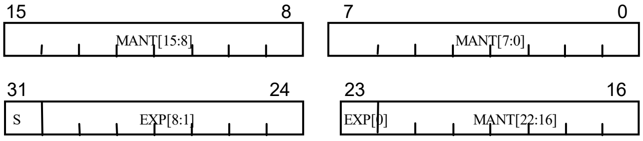
**I-12 Format (12)**



These two formats ( I-F[20:12] and I[11:0]) are for 32-bit Z Mode. Z values are formatted to be a 32-bit Integer with 12 Fractional Bits.

**Float Format**

IEEE Single Precision Floating Point



## 5.9 Display List Processor Registers

~~***Register Read Note:*** The only thing which cannot be properly read back from the CNTRL is the end address. The following bits can be read back:~~

| ~~bit~~ | ~~meaning~~ |
|---|---|
| ~~7~~ | ~~OVD~~ |
| ~~8~~ | ~~DLF~~ |
| ~~9~~ | ~~SEN~~ |
| ~~10~~ | ~~STOP~~ |

~~this is only for reading back registers.~~

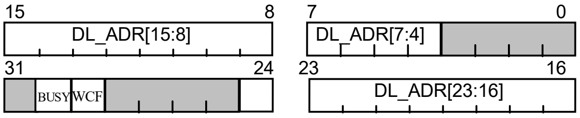
### ~~5.1.1~~5.9.1 Display List Processor Address Register RBASE_D + (0x00F8)

**Формат:** Список

**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 the start address 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 Imagine 3 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 and Imagine will re-try if as third address is attempted to be written.



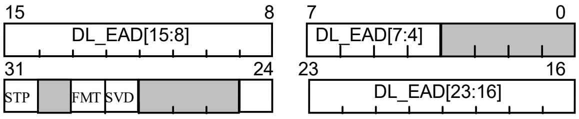
| Bits | Name | Value | Function |
|---|---|---|---|
| DL_ADR[3:0] | Reserved | 0x0 | Reserved |
| DL_ADR[24:4] | DL_ADR | | Display List Start Address |
| DL_ADR[28:25] | Reserved | 0 | Reserved |
| DL_ADR[29] | WCF | | Wait for Linear windows cache to flush<br>- Wait for cache flush (wait for linear window’s cache to flush before executing the list.) ~~***Read-only Bit***~~ |
| | | 0x0 | Disabled |
| | | 0x1 | Enabled |
| DL_ADR[30] | BUSY | | Display List Busy<br>- The display list is currently full and cannot accept any more start addresses. Start Address must NOT be written when this bit is set. Unpredictable results would occur. ***Read-only Bit*** |
| | | 0x0 | Not Busy |
| | | 0x1 | Busy |
| DL_ADR[31] | Reserved | 0x0 | Reserved |

### ~~5.1.2~~5.9.2 Display List Processor Control Register RBASE_D + (0x00FC)

**Name: DL_CNTRL**

**Type: Memory mapped read write**

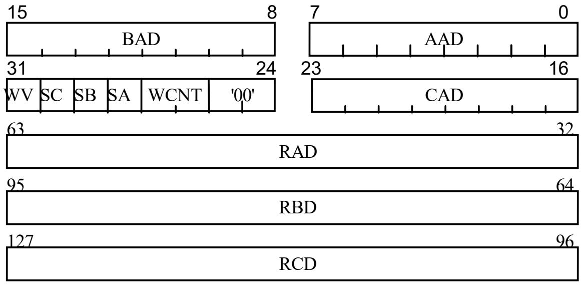
This register contains the control bits and the termination address of the display list.



| 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[27:25] | Reserved | 0x0 | Reserved |
| DL_CNTRL[28] | DL_SVD | <br>0x0<br>0x1 | Display List Source Enable Override<br>Use DL_SEN<br>Use DMA |
| DL_CNTRL[29] | DL_FMT | <br>0x0<br>0x1 | Display List Format<br>Display List Format 0, Write register defined in display list.<br>Display List Format 1, Write only XY0,XY2,XY3,XY1. |
| DL_CNTRL[30] | ~~DL_SEN~~Reserved | <br>~~0x0~~<br>~~0x1~~0x0 | ~~Display List Source~~<br>~~Display list source is display buffer.~~<br>~~Display List source is Virtual buffer.~~Reserved |
| DL_CNTRL[31] | DL_STP | <br><br>0x0<br>0x1 | Display List Processor Stop.<br>This read write bit is used to control the operation of the DLP.<br>The DLP is currently running.<br>And Busy is zero the DLP is idle. |

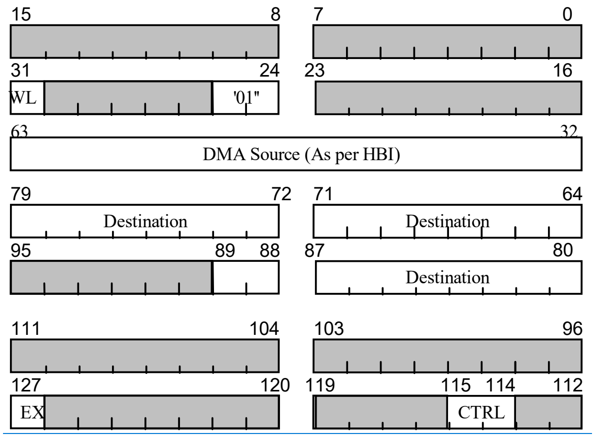
### ~~5.1.3~~5.9.3 Display List Instruction Word, Format Zero No DMA “DL_FMT = 0 & DL_IW[25:24] = 00”

~~Non DMA Format~~



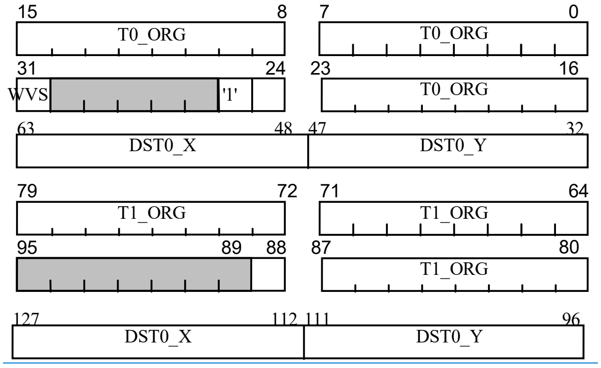
| Bits | Name | Value | Function |
|---|---|---|---|
| DL_IW[7:0] | AAD | | Address for register A data. |
| DL_IW[15:8] | BAD | | Address for register B data. |
| DL_IW[23:16] | CAD | | Address for register C data. |
| DL_IW[25:24] | SELECT | 0x0 | Must be 0 for this format |
| DL_IW[27:26] | WCNT | | 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 |

### 5.9.4 Display List Instruction Word, 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 |
| DL_IW[31] | WL | <br>0x0<br>0x1 | Wait for DMA done<br>Don’t wait for current (and previous DMA’s) to finish<br>Wait for pending and this DMA to finish |
| DL_IW[63:32] | SRC | | DMA Source as per HBI |
| DL_IW[89:64] | DST | | DMA destination as per HBI |
| DL_IW[113:90] | Reserved | 0x0 | Reserved |
| DL_IW[115:114] | CTRL | | DMA Control as per HBI |
| DL_IW[126:116] | Reserved | 0x0 | Reserved |
| DL_IW[127] | EX | <br>0x0<br>0x1 | Expedite<br>Normal<br>Expedited |

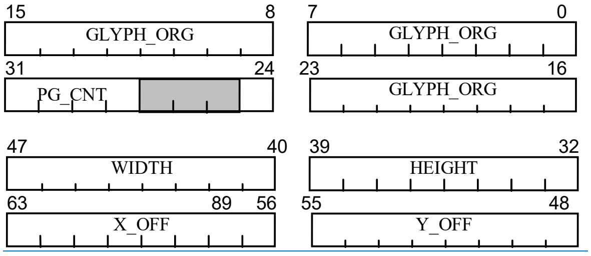
### 5.9.5 Display List Instruction Word, Format Zero TEXT "DL_FMT = 0 & DL_IW[25] = 1"



| 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 | | Character Count |
| | | 0x0 | One character |
| | | 0x1 | Two characters |
| DL_IW[30:28] | Reserved | 0x0 | Reserved |
| DL_IW[31] | WVS | | Wait for vertical sync after this character |
| | | 0x0 | Disabled |
| | | 0x1 | Enabled |
| DL_IW[47:32] | DST0_Y | | Destination 0 X |
| DL_IW[63:48] | DST0_X | | Destination 0 Y |
| DL_IW[88:64] | T1_ORG | | Table 1 Origin |
| DL_IW[95:89] | Reserved | 0x0 | Reserved |
| DL_IW[111:96] | DST1_Y | | Destination 1 X |
| DL_IW[127:112] | DST1_X | | Destination 1 Y |

### 5.9.6 Text Table Format

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. 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 |

**Display List Instruction Word, Format One "DL_FMT = 1"**

31 0
XY0 REGISTER DATA[31:0]

63 32
XY2 REGISTER DATA[31:0]

95 64
XY3 REGISTER DATA[31:0]

127 96
XY1 REGISTER DATA[31:0]

| 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 |

### 5.9.7 DLP Notes

- Waiting after DMA will cause the DLP to hold the DE busy and will not execute another command until the DMA pipeline is empty
- HBI DMA cannot be used at the same time as DLP DMA. The DMA pipeline must be empty prior to executing a DLP 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, DMA and 3 register writes may all be mixed in the same physical list. The 4 register mode must be executed in it's own list.

**Формат:** Список

**Формат:** Список