



Sinneslöschen

SYSTEMS CORPORATION

CLEAR SIGNALS. RELIABLE SYSTEMS.

CUSTOMER TECHNICAL DOCUMENTATION

VDP-7 Programmer's Reference Manual

Raster / Vector Display Processor

Document Number: RM-VDP7-03
Revision: C
Release Date: May 19, 1998
Product: VDP-7 Raster / Vector Display Processor
Distribution: Customer / OEM / Development

This manual defines the VDP-7 programming model, including host registers, command packet formats, frame-buffer operations, geometric drawing, display timing, interrupts and diagnostic behavior. Electrical characteristics and package specifications are provided separately in DS-VDP7-02.

Sinneslöschen Systems Corporation
1450 Meridian Technology Drive
Sunnyvale, CA 94089 USA
Technical Support: 1-800-555-0142
Documentation Fax: 408-555-0163



Copyright and Technical Notice

Copyright © 1998 Sinneslöschen Systems Corporation. All rights reserved.

The information in this manual is believed to be accurate as of the publication date. Sinneslöschen Systems Corporation reserves the right to revise products, software and documentation without notice. This manual describes programming-visible operation of the VDP-7 Raster / Vector Display Processor and is intended for software developers, OEM engineering groups, system integrators and qualified service personnel.

No part of this publication may be reproduced for commercial distribution without prior written permission. Customers may reproduce reasonable portions for internal development, integration and service purposes.

CAUTION

The VDP-7 controls external display timing and framebuffer access. Programming invalid clock, blanking or memory parameters may produce an unreadable display or bus activity outside the qualified operating range of attached components. Preserve a known-good initialization path before testing new timing values.

This manual does not replace the VDP-7 Data Sheet (DS-VDP7-02), board-level installation instructions or safety documentation supplied with a complete Sinneslöschen system.

Technical Support

North American Technical Support: 1-800-555-0142

Direct: 408-555-0186

Fax: 408-555-0127

E-mail: support@sinnesloschen.com

When requesting programming assistance, provide the VDP-7 hardware revision, host platform, framebuffer configuration, display timing, software release, failing command sequence and the contents of STATUS and ERROR_STATUS.



Document Control

Document	RM-VDP7-03 - VDP-7 Programmer's Reference Manual
Current Revision	C
Release Date	May 19, 1998
Product	VDP-7 Raster / Vector Display Processor
Primary Coverage	Register model, command stream, framebuffer interface, geometric operations, timing control, diagnostics
Related Hardware	VDP-7 Evaluation Board; VDP-7 Development System; SV-97; SV-98D
Related Software	VDP Development Toolkit 3.2; Graphics Library 4.1
Maintaining Organization	Display Systems Laboratory / Technical Documentation

Document Purpose

This reference defines the programming contract between host software and VDP-7 hardware. Electrical limits, package data and board design information are intentionally summarized only where they affect software operation. For electrical characteristics and signal requirements, use DS-VDP7-02.

Applicability

Revision C applies to VDP-7 silicon revisions B1 through C2 unless a section identifies a narrower range. Differences affecting revision A evaluation hardware are collected in Chapter 12 and in SB-97-019.

SERVICE NOTE

Do not infer hardware revision from software driver version. Read the REVISION register and compare the board assembly label when qualification status depends on revision.

Revision History

Rev.	Date	Status	Summary
A	06.05.97	Released	Initial production programming reference for VDP-7 Rev. A/B evaluation and OEM hardware.
B	11.21.97	Superseded	Added high-resolution timing guidance, clarified host-to-framebuffer transfers, expanded interrupt definitions and documented polygon edge cases.
C	05.19.98	Current	Revises command FIFO behavior, adds Graphics Library 4.1 notes, expands VDP-6 compatibility guidance, documents C2 revision status bits and consolidates display-timing corrections.

Related Publications

- DS-VDP7-02 - VDP-7 Raster / Vector Display Processor Data Sheet, Rev. B.
- AN-VDP7-05 - High-Resolution Display Design with the VDP-7, Rev. A.
- AN-VDP7-06 - VDP-6 Application Migration Notes.
- SB-97-019 - VDP-7 Evaluation Board Power-Up Procedure.
- RM-GL41-01 - Graphics Library 4.1 Programmer's Reference.

Change-Control Note

Where a product-specific Service Bulletin conflicts with this manual for the affected hardware range, the Service Bulletin takes precedence. Applications used in validated OEM systems should not be rebuilt solely because a newer manual revision exists.



How to Use This Manual

The VDP-7 can be programmed at three levels: through a supported Sinneslösch graphics library, by constructing command packets in host memory, or by direct register-level control. This manual documents all three from the hardware point of view, but examples are intentionally written at the register/packet boundary so they remain useful across operating systems.

Recommended Reading Paths

New driver development: Chapters 1-6, then Chapters 8-11.

Graphics application optimization: Chapters 6-9 and Appendix B.

Display timing integration: Chapters 4 and 9 plus AN-VDP7-05.

VDP-6 migration: Chapter 12 and AN-VDP7-06.

Field diagnosis: Chapter 11, STATUS/ERROR register pages and Appendix C.

Programming Model Rule

Software shall treat reserved bits as zero on write and ignore reserved bits on read unless a later document revision states otherwise. Read-modify-write operations are required for registers containing both software-controlled and hardware-maintained fields.

Compatibility Principle

A command stream that depends on undocumented register values, reserved command bits or evaluation-board behavior is not considered VDP-7 compatible, even when it appears to operate on a particular revision.

Conventions and Numeric Formats

Notation	Meaning
0xNNNN	Hexadecimal value.
bit 0	Least-significant bit of a 32-bit register.
[31:16]	Inclusive bit field, most-significant bit first.
R/W	Software may read and write the field.
RO	Read-only; writes are ignored unless otherwise stated.
W1C	Writing one clears the corresponding status bit.
WO	Write-only; read value is undefined or implementation-specific.
Reset = 0	Value after hardware reset has completed.
Undefined	Software must not depend on the reset or read value.

Coordinates are signed unless a command states otherwise. Most drawing coordinates use 16.8 fixed-point representation: the upper 16 bits contain the signed integer component and the lower eight significant fractional bits provide subpixel positioning. Integer-only examples set fractional bits to zero.

Framebuffer addresses are byte addresses. Host-visible register offsets are expressed relative to the mapped VDP-7 register aperture. Command packet words are 32 bits and are shown in host order after any bus-specific byte swapping performed by the system interface.

SERVICE NOTE

On systems using the SL-9816 or another host bridge, verify the bridge's endian and write-posting configuration before interpreting command FIFO failures as a VDP-7 error.

Contents 1 of 4

Section	Subject	Pages
Front Matter	Copyright / Document Control / Conventions	2-10
Chapter 1	Introduction	11-18
Chapter 2	Functional Architecture	19-32
Chapter 3	Host Interface and Addressing	33-48
Chapter 4	Reset and Initialization	49-60
Chapter 5	Register Reference	61-116
Chapter 6	Command Processor and FIFO	117-132
Chapter 7	Drawing and Transfer Commands	133-156
Chapter 8	Framebuffer and Pixel Formats	157-170
Chapter 9	Display Timing and Synchronization	171-184
Chapter 10	Cursor, Palette and Color	185-192
Chapter 11	Interrupts and Diagnostics	193-200
Chapter 12	VDP-6 Compatibility	201-206
Appendix A	Register Summary	207-209
Appendix B	Command Summary	210-211
Appendix C	Error Quick Reference	212
Appendix D	Initialization Skeleton	213
Appendix E	Glossary and Support References	214

Documentation Set

This manual should be used with DS-VDP7-02 for electrical/package information and AN-VDP7-05 for board-level high-resolution timing guidance.

Contents 2 of 4

Page	Topic
11	1.1 Product Overview
12	1.2 Supported Functional Blocks
13	1.3 Feature Summary and Limits
14	1.4 Software Layers
15	1.5 Programming Sequence at a Glance
16	1.6 Access Ordering and Barriers
17	1.7 Reset Domains
18	1.8 Revision Identification
19	2.1 Internal Data Flow
20	2.2 Command Processor
21	2.3 Rendering State
22	2.4 Coordinate Representation
23	2.5 Clipping Pipeline
24	2.6 Raster Operation Unit
25	2.7 Framebuffer Scheduler
26	2.8 Display Timing Engine
27	2.9 Cursor Composition
28	2.10 Interrupt Controller
29	2.11 Diagnostic Logic
30	2.12 Clock Domains
31	2.13 Memory Protection Boundaries
32	2.14 Architecture Programming Checklist
33	3.1 Host Interface Overview
34	3.2 Register Aperture
35	3.3 Address Decode and Alignment
36	3.4 Read Semantics
37	3.5 Write Posting
38	3.6 Command Port Writes
39	3.7 FIFO Capacity Management
40	3.8 Host Framebuffer Window
41	3.9 Byte Order
42	3.10 Atomicity
43	3.11 Host Interrupt Interface
44	3.12 Bus Error Handling
45	3.13 Polling vs. Interrupts
46	3.14 Multithreaded Access
47	3.15 Host Interface Diagnostic Test
48	3.16 Host Programming Checklist
49	4.1 Power-On State
50	4.2 Full Initialization Sequence
51	4.3 Command-Engine Reset
52	4.4 Display-Engine Reset
53	4.5 Interrupt Initialization
54	4.6 Framebuffer Initialization
55	4.7 Initial Clip and Drawing State
56	4.8 Timing Bring-Up
57	4.9 Warm Driver Restart
58	4.10 Shutdown Sequence
59	4.11 Initialization Example
60	4.12 Initialization Checklist
61	5.01 Register Reference - DEVICE_ID
62	5.02 Register Reference - REVISION
63	5.03 Register Reference - RESET
64	5.04 Register Reference - STATUS
65	5.05 Register Reference - INT_STATUS
66	5.06 Register Reference - INT_ENABLE
67	5.07 Register Reference - INT_ACK
68	5.08 Register Reference - ERROR_STATUS
69	5.09 Register Reference - HOST_CTRL
70	5.10 Register Reference - HOST_CONFIG
71	5.11 Register Reference - FIFO_STATUS
72	5.12 Register Reference - FIFO_WATERMARK
73	5.13 Register Reference - CMD_PORT
74	5.14 Register Reference - CMD_BASE
75	5.15 Register Reference - CMD_LIMIT
76	5.16 Register Reference - CMD_PTR
77	5.17 Register Reference - FB_BASE
78	5.18 Register Reference - FB_PITCH

Contents 3 of 4

Page	Topic
79	5.19 Register Reference - FB_FORMAT
80	5.20 Register Reference - FB_LIMIT
81	5.21 Register Reference - SCRATCH_BASE
82	5.22 Register Reference - SCRATCH_LIMIT
83	5.23 Register Reference - CLIP_LEFT_TOP
84	5.24 Register Reference - CLIP_RIGHT_BOTTOM
85	5.25 Register Reference - ORIGIN
86	5.26 Register Reference - SCALE_X
87	5.27 Register Reference - SCALE_Y
88	5.28 Register Reference - LINE_CTRL
89	5.29 Register Reference - FG_COLOR
90	5.30 Register Reference - BG_COLOR
91	5.31 Register Reference - ROP
92	5.32 Register Reference - PATTERN
93	5.33 Register Reference - BLT_SRC_BASE
94	5.34 Register Reference - BLT_DST_BASE
95	5.35 Register Reference - BLT_SRC_XY
96	5.36 Register Reference - BLT_DST_XY
97	5.37 Register Reference - BLT_SIZE
98	5.38 Register Reference - BLT_CTRL
99	5.39 Register Reference - POLY_CTRL
100	5.40 Register Reference - EDGE0
101	5.41 Register Reference - EDGE1
102	5.42 Register Reference - VERTEX_PORT
103	5.43 Register Reference - VERTEX_COUNT
104	5.44 Register Reference - CURSOR_CTRL
105	5.45 Register Reference - CURSOR_POS, CURSOR_BASE
106	5.46 Register Reference - CURSOR_COLOR0, CURSOR_COLOR1
107	5.47 Register Reference - DISP_CTRL, H_TOTAL
108	5.48 Register Reference - H_SYNC, H_BLANK
109	5.49 Register Reference - V_TOTAL, V_SYNC
110	5.50 Register Reference - V_BLANK, START_ADDR
111	5.51 Register Reference - LINE_COMPARE, CLOCK_CTRL
112	5.52 Register Reference - DAC_CTRL, LUT_ADDR
113	5.53 Register Reference - LUT_DATA, GPIO_CTRL
114	5.54 Register Reference - GPIO_DATA, DIAG_CTRL
115	5.55 Register Reference - PERF_COUNT0, PERF_COUNT1
116	5.56 Register Reference - SIGNATURE, TEST_STATUS
117	6.1 Command Packet Format
118	6.2 Header Encoding
119	6.3 Fixed-Length Packets
120	6.4 Variable-Length Packets
121	6.5 FIFO Watermark Strategy
122	6.6 Completion Markers
123	6.7 WAIT_IDLE
124	6.8 WAIT_VBLANK
125	6.9 State Commands
126	6.10 Error Recovery
127	6.11 Command Tags
128	6.12 Command Alignment
129	6.13 Command Submission Example
130	6.14 Packet Trace Interpretation
131	6.15 Throughput Considerations
132	6.16 Command Processor Checklist
133	7.01 Command - NOP
134	7.02 Command - SET_STATE
135	7.03 Command - SET_CLIP
136	7.04 Command - SET_ORIGIN
137	7.05 Command - SET_COLOR
138	7.06 Command - LINE
139	7.07 Command - POLYLINE
140	7.08 Command - POINTS
141	7.09 Command - RECT_OUTLINE
142	7.10 Command - RECT_FILL
143	7.11 Command - TRIANGLE_FILL
144	7.12 Command - POLY_FILL
145	7.13 Command - BLIT
146	7.14 Command - STRETCH_BLIT

Contents 4 of 4

Page	Topic
147	7.15 Command - HOST_TO_FB
148	7.16 Command - FB_TO_HOST
149	7.17 Command - PATTERN_FILL
150	7.18 Command - MONO_EXPAND
151	7.19 Command - CLEAR_SURFACE
152	7.20 Command - SET_CURSOR
153	7.21 Command - LOAD_LUT
154	7.22 Command - WAIT_VBLANK
155	7.23 Command - WAIT_IDLE
156	7.24 Command - SYNC_MARKER
157	8.1 Framebuffer Organization
158	8.2 8-bit Indexed Format
159	8.3 RGB565 Format
160	8.4 XRGB8888 Format
161	8.5 Pitch Rules
162	8.6 Surface Alignment
163	8.7 Overlapping Blits
164	8.8 Host Transfers
165	8.9 Bandwidth Calculation
166	8.10 Display and Drawing Contention
167	8.11 Page Flipping
168	8.12 Off-Screen Allocation
169	8.13 Bounds Protection
170	8.14 Framebuffer Programming Checklist
171	9.1 Timing Model
172	9.2 Horizontal Timing
173	9.3 Vertical Timing
174	9.4 Sync Polarity
175	9.5 Pixel Clock Selection
176	9.6 START_ADDR and Panning
177	9.7 Blanking and Forced Blank
178	9.8 Vertical Blank Synchronization
179	9.9 Line Compare
180	9.10 Qualified Example Timing
181	9.11 Timing Update Procedure
182	9.12 High-Resolution Considerations
183	9.13 Timing Fault Recovery
184	9.14 Display Timing Checklist
185	10.1 Hardware Cursor Overview
186	10.2 Cursor Image Format
187	10.3 Cursor Position and Hotspot
188	10.4 Cursor Colors
189	10.5 LUT Programming
190	10.6 Color Interpretation
191	10.7 Palette Animation and Diagnostics
192	10.8 Cursor and Color Checklist
193	11.1 Interrupt Service Model
194	11.2 CMD_DONE and FIFO_LOW
195	11.3 VBLANK Interrupt
196	11.4 Error Interrupt
197	11.5 Performance Counters
198	11.6 Signature Testing
199	11.7 Field Diagnostic Capture
200	11.8 Diagnostic Checklist
201	12.1 Compatibility Scope
202	12.2 Preserved Concepts
203	12.3 Changed Areas
204	12.4 Line Rendering Differences
205	12.5 Timing Migration
206	12.6 Migration Checklist
207	Appendix A.1 - Register Summary
208	Appendix A.2 - Register Summary
209	Appendix A.3 - Register Summary
210	Appendix B.1 - Command Summary (1 of 2)
211	Appendix B.2 - Command Summary (2 of 2)
212	Appendix C - Error and Status Quick Reference
213	Appendix D - Complete Initialization Skeleton
214	Appendix E - Glossary and Support References

Chapter 1 - Introduction

1.1 Product Overview

The VDP-7 is a programmable display processor intended for technical graphics, engineering visualization, instrumentation and specialized OEM systems. It combines vector-oriented drawing functions with raster framebuffer operations, programmable clipping, scaling, cursor support and display timing control.

The processor is designed to operate as a graphics peripheral rather than as a general-purpose CPU. Host software supplies state changes and drawing commands through memory-mapped registers or a command FIFO. The VDP-7 performs drawing operations against an external framebuffer and supplies timing/control information to the display output path.

Principal software-visible capabilities include hardware line and polyline generation, filled rectangles and polygons, framebuffer block transfers, source-to-destination scaling, logical raster operations, programmable clipping rectangles, host transfers, display-start control, hardware cursor operation and interrupt-driven synchronization.

Typical System Context

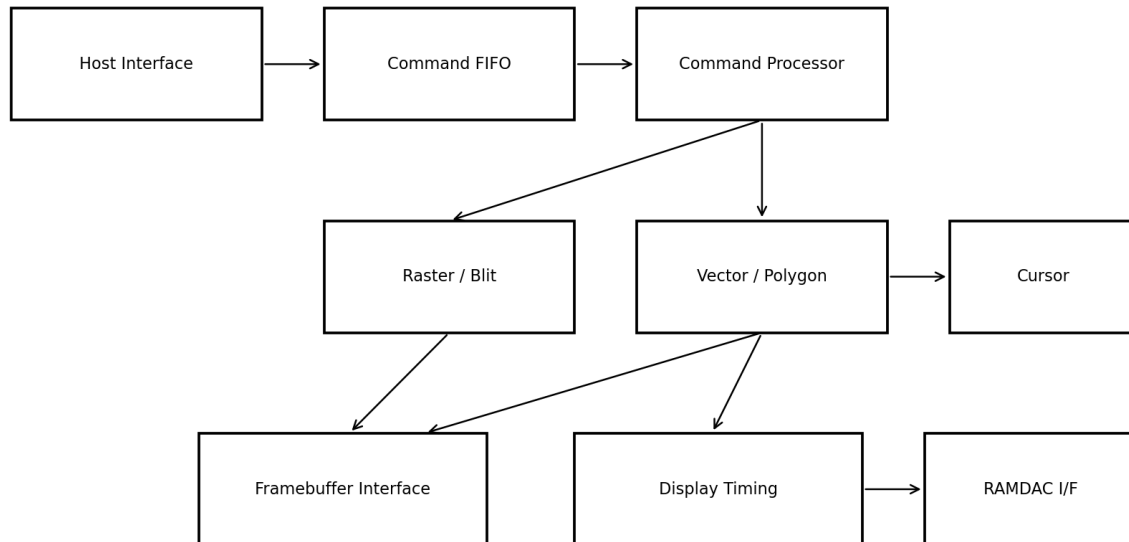
In an SV-97 workstation, the VDP-7 normally operates under Windows NT 4.0 through a Sinneslösch display driver and Graphics Library 4.x. In laboratory or OEM systems, the processor may be driven by a custom host environment. The programming model is independent of the user interface presented by the host system.

Terminology

The term *vector* in VDP-7 naming refers to line-oriented technical graphics capability. The VDP-7 also performs raster framebuffer operations; it is not limited to calligraphic/vector-display output.

1.2 Supported Functional Blocks

VDP-7 Functional Architecture



The host interface presents control/status registers, a command submission path and optional framebuffer access windows. The command processor decodes packets and applies the currently programmed drawing state. Geometry stages perform coordinate interpretation, clipping and edge setup; raster stages generate pixels or spans; the ROP stage combines source, pattern and destination data; the framebuffer interface schedules external memory cycles.

The display timing block operates independently from the drawing pipeline once programmed. It fetches scanout data from the framebuffer while drawing commands may continue to update other regions. Software must account for available memory bandwidth when combining high-resolution scanout with sustained blit or polygon traffic.

The hardware cursor uses a separate image base and position state and is composited during scanout, avoiding the need to save and restore framebuffer pixels during normal pointer movement.

1.3 Feature Summary and Limits

Function	Programming-visible capability
Host interface	32-bit memory-mapped control; posted command writes supported by qualified bridges.
Command FIFO	64-word hardware FIFO with programmable interrupt watermark.
Coordinates	Signed 16.8 fixed-point drawing coordinates.
Clipping	Programmable rectangular clip window; inclusive/exclusive edge rules defined per command.
Raster formats	8-bit indexed, 16-bit RGB, and 32-bit packed RGB modes.
Line drawing	Subpixel endpoints, programmable color, logical ROP, optional endpoint suppression.
Polygon fill	Convex polygons and triangles; software decomposition required for arbitrary concave polygons.
Blit	Screen-to-screen, host-to-screen, screen-to-host, pattern fill and scaling.
Cursor	64 x 64 maximum, two programmable colors plus transparent states.
Timing	Fully programmable horizontal/vertical totals, sync and blanking boundaries.
Interrupts	FIFO, completion, vertical blank, error and diagnostic sources.

These limits describe the base VDP-7 programming model. Complete system limits may be narrower because of RAMDAC, frame-buffer memory, board layout or display restrictions.

CAUTION

Do not select a display mode solely because timing registers can represent it. Pixel clock, RAMDAC, memory bandwidth and monitor limits remain board- and system-dependent.

1.4 Software Layers

A typical software stack contains four layers. Applications call Graphics Library 4.1 or an OEM rendering layer. That library translates operations into a driver-neutral graphics state. A VDP-7 display driver allocates framebuffer resources, validates modes, schedules command submissions and handles interrupts. The hardware interface layer performs memory-mapped register access and manages bridge-specific ordering.

Direct register programming is appropriate for boot diagnostics, embedded systems and driver development. Application software should normally avoid direct register access because it bypasses resource arbitration, mode ownership and compatibility logic supplied by the driver.

State Ownership

The VDP-7 does not maintain process isolation. All drawing state is global to the device. Multitasking systems must serialize state changes and command packets. A driver may shadow state in host memory and emit only differences when switching contexts, but hardware should be treated as containing one active rendering context.

Synchronization Ownership

A host write being accepted by the bus does not mean the associated drawing operation is complete. Completion must be established with command markers, FIFO/idle status or interrupts as described in Chapter 6.

1.5 Programming Sequence at a Glance

A minimal initialization sequence is:

1. Hold display output disabled and verify the DEVICE_ID / REVISION values.
2. Reset the command processor and clear pending interrupt/error state.
3. Program framebuffer base, pitch, format and valid-address limit.
4. Program display timing while scanout remains disabled.
5. Initialize clip, color, ROP and cursor state.
6. Enable selected interrupts and configure FIFO watermark behavior.
7. Enable display timing and verify vertical-blank activity.
8. Submit a known drawing command and confirm completion.

The order is deliberate. Enabling scanout before framebuffer and timing values are coherent can issue unintended memory fetches or generate invalid sync. Similarly, drawing commands must not be accepted while the command processor is in reset or while stale error state is being interpreted as a new failure.

Warm Restart

A warm driver restart may preserve display timing and framebuffer content while resetting only the command processor. Software must distinguish that operation from a full device reset, which returns most programming-visible state to reset values and disables display output.

1.6 Access Ordering and Barriers

VDP-7 register writes may pass through host bridges capable of posted operation. Software must use an ordering read or bridge-specific barrier when subsequent CPU activity depends on a write having reached the device.

A common sequence is to write command words through CMD_PORT and then read FIFO_STATUS before reclaiming a host-side command buffer. The FIFO status read forces preceding posted writes to complete on qualified Sinneslösch host interfaces.

Device Barriers

Three concepts must not be confused:

- **Host ordering barrier:** ensures host writes reached the VDP-7 interface.
- **FIFO acceptance:** ensures command words entered the hardware FIFO.
- **engine completion:** ensures the decoded operation has finished modifying framebuffer state.

Use WAIT_IDLE or a completion marker when CPU or another device must observe completed rendering. FIFO empty alone does not guarantee that the raster/blit engine is idle; the final command may already have left the FIFO and still be executing.

1.7 Reset Domains

The VDP-7 contains distinct reset domains for the host interface, command processor, display timing and diagnostic logic. A board-level hardware reset resets all domains. The RESET register permits software to reset selected internal engines without necessarily disturbing display timing.

This separation allows a driver to recover from a malformed command sequence while maintaining a stable display. However, partial reset requires deliberate state reconstruction: drawing state registers may return to defaults even when display timing and framebuffer base remain active.

Recommended Recovery

If ERROR_STATUS reports a command parser fault, first disable command submission, capture diagnostic state, issue a command-engine reset, clear the error, restore drawing state and test with a NOP + completion marker. Escalate to full-device reset only when the command engine does not return to idle or when framebuffer/display state is already considered lost.

CAUTION

A partial reset does not cancel host writes that remain posted in an upstream bridge. Flush or disable the host path before resetting the command engine.

1.8 Revision Identification

The REVISION register identifies silicon revision and feature level. Production software should use feature bits rather than hard-coded revision comparisons wherever possible.

Revision	Typical use	Programming note
A0/A1	Evaluation hardware	Requires SB-97-019 startup procedure; several diagnostic fields differ.
B1/B2	Early production	Base Revision B command model; high-resolution timing errata addressed by driver workarounds.
C1/C2	Current production	Adds corrected FIFO watermark behavior and expanded TEST_STATUS reporting.

Revision letters on a board assembly are not equivalent to the silicon revision field. A Rev. B evaluation board may contain later silicon after service replacement. Always read the device and inspect the board label when both are relevant.

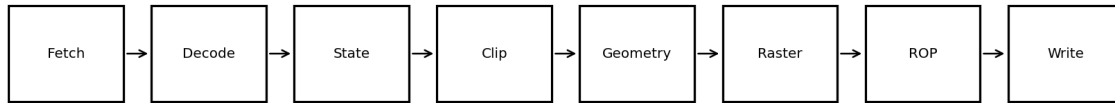
SERVICE NOTE

When collecting a field report, record DEVICE_ID, REVISION, board assembly revision, RAMDAC type and framebuffer memory configuration. These values are sufficient to resolve most programming compatibility questions without requiring board removal.

Chapter 2 - Functional Architecture

2.1 Internal Data Flow

Command Execution Pipeline



State changes are ordered with drawing commands; host reads may require an explicit WAIT_IDLE barrier.

Command words enter through the host interface and are buffered in the command FIFO. The decoder converts packet opcodes into micro-operations that reference global drawing state. Coordinate and clip stages produce accepted primitives or spans; raster logic generates pixel addresses; the raster-operation unit combines source/pattern/destination data; the framebuffer scheduler issues memory cycles. Display scanout is scheduled separately and receives priority sufficient to maintain programmed timing when the mode is within qualified bandwidth.

The internal data path is 32 bits for register, command and core pixel operations. External framebuffer implementation may use a wider physical memory organization; that width is board-specific and does not alter command packet format.

Software-visible stalls occur primarily when the FIFO reaches its high-water condition, when a drawing command waits for framebuffer bandwidth, or when an explicit synchronization command prevents subsequent work from advancing.

2.2 Command Processor

The command processor accepts fixed- and variable-length packets. The first word identifies opcode, packet length and selected state-control flags. Remaining words contain coordinates, colors, addresses or command-specific parameters.

Malformed packet length is treated as a parser error because the processor cannot reliably identify the next command boundary. Once a parser error is latched, subsequent command words are not executed until software resets the command engine and clears the error.

The processor is intentionally simple: complex tessellation, font interpretation and arbitrary polygon decomposition remain host responsibilities. This division keeps hardware command behavior deterministic and allows the same rendering primitives to be used by embedded systems and workstation drivers.

2.3 Rendering State

Rendering state includes clip bounds, origin, scale factors, foreground/background colors, raster operation, pixel format and selected command-mode flags. State is global and persists across commands until explicitly changed or reset.

A command packet may override selected state values when the opcode defines inline parameters, but the override does not necessarily update the corresponding global register. Driver writers should avoid mixing register-programmed and inline state unless the save/restore rules are fully understood.

For context switching, Sinneslöschen software maintains a host-side state image and re-emits state after ownership changes. There is no hardware context RAM in VDP-7.

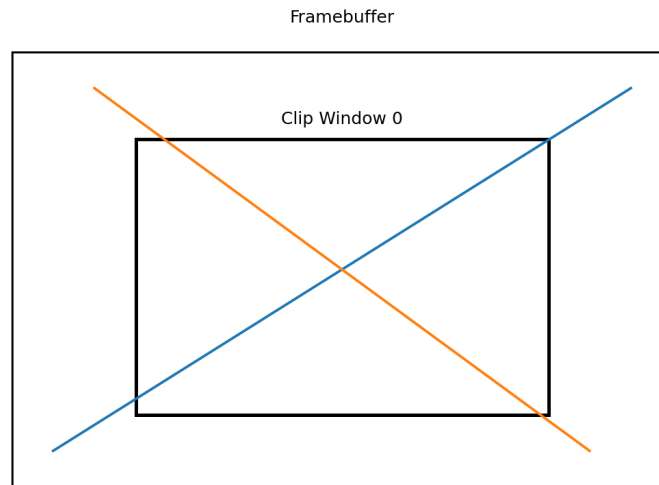
2.4 Coordinate Representation

Drawing coordinates use signed 16.8 fixed-point representation in command packets that support subpixel placement. The integer portion spans -32768 through +32767; fractional values represent 1/256 pixel increments.

Framebuffer addresses are derived after origin, scale and clip processing. Commands that explicitly operate on memory rectangles may use integer pixel coordinates instead of fixed-point geometric coordinates. The command reference identifies the representation for each field.

Overflow during coordinate addition is saturating at the clip/viewport stage rather than wrapping into the opposite side of the address space. Software should still pre-clip very large coordinates because saturation can produce unexpectedly long edge calculations.

2.5 Clipping Pipeline



Pixels outside the active clip rectangle are rejected before raster write.

A programmable rectangular clip window is applied after origin and scale state but before pixel generation. Lines are clipped geometrically; rectangle/blit operations are clipped by adjusting source and destination extents; polygon spans are rejected or truncated against the active bounds.

The default reset clip window rejects all drawing until valid bounds are programmed. This behavior prevents uninitialized software from writing arbitrary framebuffer addresses. Drivers should establish framebuffer limits first and derive the initial clip rectangle from the visible or allocated surface.

The clip registers are independent of display timing. Off-screen surfaces can use a clip rectangle unrelated to the current scanout dimensions as long as framebuffer address limits remain valid.

2.6 Raster Operation Unit

The ROP unit combines source, pattern and destination data after pixel-format interpretation. Sixteen logical operations cover copy, AND, OR, XOR, invert and related boolean combinations. Color expansion commands additionally select foreground or background color according to source bits.

Destination reads required by the selected ROP consume framebuffer bandwidth and may reduce fill/blit throughput. Pure source-copy operations can avoid the destination-read phase on qualified memory configurations.

ROP behavior is defined per stored pixel word, not per displayed component after RAMDAC conversion. In indexed-color mode, logical operations therefore affect palette indices.

2.7 Framebuffer Scheduler

The framebuffer scheduler arbitrates scanout fetches, drawing writes, destination reads, command DMA activity where present, and host access windows. Display fetch has deterministic service requirements; graphics traffic uses remaining slots.

A mode can be electrically valid yet perform poorly if scanout consumes most available memory bandwidth. Drivers should use the bandwidth calculation in Chapter 8 and avoid assuming that a higher pixel clock leaves unchanged graphics throughput.

The scheduler does not guarantee fairness to continuous host framebuffer reads. Diagnostic software should limit large readbacks while an active high-resolution display is operating.

2.8 Display Timing Engine

The timing engine generates horizontal/vertical counters, blanking intervals, sync pulses, display-address progression and vertical-blank interrupt events. Timing runs independently from drawing after DISP_CTRL enables scanout.

Horizontal register values are expressed in pixel clocks subject to board-specific granularity; vertical values are expressed in scan lines. START_ADDR may be changed during vertical blank to implement page flipping without modifying drawing state.

The engine can generate composite combinations of sync polarity and blanking appropriate to technical CRT systems, but the electrical output format is ultimately determined by the board-level RAMDAC and sync interface.

2.9 Cursor Composition

The hardware cursor is fetched from a separate cursor image area and composited after framebuffer pixel fetch. The cursor can therefore move without changing framebuffer content.

Cursor coordinates are screen-relative and may be negative within a limited range so a cursor can be partially visible at the top or left edge. Two programmable cursor colors and transparent states support monochrome and highlighted technical pointers.

Software must keep cursor image storage within the valid framebuffer-access range and aligned according to `CURSOR_BASE` requirements.

2.10 Interrupt Controller

The interrupt controller collects command completion, FIFO threshold, vertical blank, command error, framebuffer fault and diagnostic events. Each source has a status bit and enable bit; active enabled sources assert the external interrupt output through the board interface.

Most event bits use write-one-to-clear semantics. Clearing a status bit does not necessarily remove the underlying condition. For level-like conditions such as FIFO threshold, the status can become active again immediately if the threshold remains satisfied.

Drivers should read `INT_STATUS` once, mask against software-owned enables, service all recognized sources and then clear only the bits that were handled.

2.11 Diagnostic Logic

Diagnostic logic provides engine signature registers, internal test-state reporting and performance counters intended for driver qualification and field service. These facilities are not part of the normal rendering path.

Production applications should not depend on diagnostic register values except where this manual explicitly defines a stable field. Some test modes temporarily disconnect normal rendering or alter memory scheduling.

The VDP Development Toolkit 3.2 uses additional authorized diagnostic sequences; those sequences are not reproduced in application-facing documentation.

2.12 Clock Domains

Host interface, rendering core, framebuffer and display output may operate from related but distinct clocks depending on board implementation. Synchronizers isolate software-visible status transitions where necessary.

A status bit may therefore require a small number of host cycles to reflect a just-completed internal event. Software must poll defined status conditions rather than relying on fixed instruction delays.

Changing display clocking while the display engine is enabled is not supported. Disable scanout, wait for disabled status, program clock/timing state, allow board-level clock lock time and then re-enable.

2.13 Memory Protection Boundaries

FB_BASE, FB_LIMIT and pitch/format state define the address range used for normal surfaces. The hardware performs bounds checks on calculated drawing accesses when protection is enabled in HOST_CONFIG.

Bounds protection is a development and fault-containment feature, not a virtual-memory mechanism. It cannot distinguish surfaces owned by different host processes. Operating-system drivers remain responsible for allocation and isolation.

A bounds violation terminates the affected command, latches an error and leaves completion state implementation-defined for that packet. The driver should reset or resynchronize the command stream before continuing.

2.14 Architecture Programming Checklist

Before enabling an application rendering path, confirm that the host interface can perform ordered 32-bit register accesses, that the command FIFO reports expected depth, framebuffer bounds are valid, clip state is initialized, display timing is qualified for the attached hardware, interrupts are either deliberately configured or masked, and the device can execute a NOP/completion marker without error.

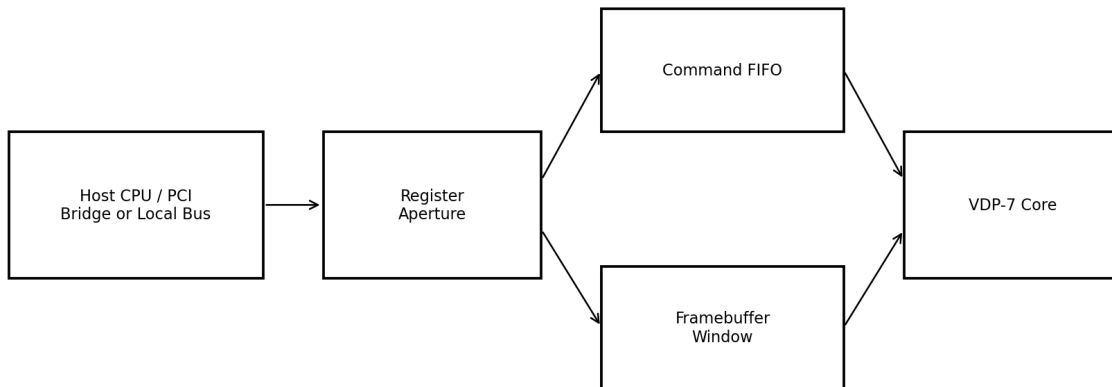
During qualification, also exercise one line, one filled rectangle, one screen-to-screen blit, one host transfer, one cursor update and one vertical-blank wait. These operations touch the principal engine paths and expose most configuration errors before higher-level software is introduced.

Record the resulting status and revision information as the baseline for future field comparison.

Chapter 3 - Host Interface and Addressing

3.1 Host Interface Overview

Host Addressing and Data Paths



The VDP-7 presents a memory-mapped control aperture containing identification, status, command, framebuffer and display registers. Qualified system boards may additionally provide a larger linear framebuffer mapping controlled by the surrounding host bridge. The core device itself does not define operating-system virtual addresses.

All programming-visible registers are 32 bits wide. Byte and halfword accesses are not guaranteed to preserve adjacent fields and should not be used unless a board-specific interface document explicitly supports them.

3.2 Register Aperture

The base register aperture occupies at least 0x200 bytes. Registers described in this manual use offsets relative to the VDP-7 base address supplied by system firmware or the host bridge. Reserved offsets must not be probed by production software.

The aperture is strongly ordered from the device perspective; ordering observed by the CPU may be weaker when PCI or another posted bridge is used. Drivers must use the bridge conventions documented for the host platform.

3.3 Address Decode and Alignment

Register offsets are naturally aligned on 4-byte boundaries. Misaligned accesses are unsupported and may be split by the host bridge in a way the VDP-7 does not interpret as one transaction.

Framebuffer addresses used in commands are byte addresses relative to the programmed memory map. Certain engines impose additional alignment: cursor data requires 256-byte alignment; accelerated blits reach full rate when source and destination begin on 32-byte boundaries; display start addresses must satisfy the scanout granularity for the selected pixel format.

3.4 Read Semantics

Read-only status is sampled into the host domain before being returned. Registers that report FIFO counts, vertical blank state or performance counters can change immediately after the read completes. Software requiring a coherent multi-register snapshot should suspend command submission where practical or use a synchronization marker before reading.

Write-only ports such as CMD_PORT do not return submitted data. Reads may return zero or implementation-specific values and must not be used as a write-posting barrier unless explicitly stated.

3.5 Write Posting

Host bridges may accept a register write before the device has observed it. A subsequent read from a defined readable VDP-7 register forces ordering on qualified Sinneslösch bridges. The recommended barrier target is FIFO_STATUS because it has no side effects.

Do not use INT_STATUS as a generic barrier in interrupt context when another execution path can clear event bits. Do not use framebuffer reads as a barrier because they may be cached or mapped with different bridge semantics.

3.6 Command Port Writes

CMD_PORT accepts one 32-bit command word per successful host write. Software may issue a burst of posted writes only when the bridge guarantees order and the driver has already verified sufficient FIFO capacity.

If FIFO full is reached, behavior depends on the host interface: some boards back-pressure the bus, while others rely on software watermarking. Portable drivers use FIFO_STATUS and never depend on indefinite host retry behavior.

3.7 FIFO Capacity Management

FIFO_STATUS reports free and occupied word counts. The hardware FIFO holds 64 words. Drivers normally reserve a safety margin rather than filling all entries so asynchronous interrupt-service commands can be inserted without deadlock.

A common policy waits until at least 16 words are free before emitting a packet up to 12 words long. Larger variable-length transfers should be chunked or submitted through a board-level DMA mechanism where available.

3.8 Host Framebuffer Window

Some VDP-7 boards map framebuffer memory into host address space. This mapping is a board feature; its caching and write-combining characteristics are controlled by the host platform.

CPU framebuffer writes must be synchronized with VDP-7 drawing when both can touch the same region. The VDP-7 does not snoop CPU caches. Drivers should use uncached/write-combined mappings appropriate to the operating system and insert engine barriers before CPU readback.

3.9 Byte Order

The canonical VDP-7 register and command word definition numbers bits as seen by the device. On little-endian x86 hosts using qualified PCI bridges, 32-bit writes normally appear without software byte swapping. Other hosts may require bridge configuration or software conversion.

Pixel byte order is separately controlled by framebuffer format and board wiring. Do not infer command-word endianness from the visible order of color components in memory.

3.10 Atomicity

Aligned 32-bit register writes are atomic at the VDP-7 interface. Multi-register state updates are not atomic. If a drawing engine can observe partially updated state, software must wait for idle or use command packets that carry state inline.

Display timing changes are particularly sensitive: keep display output disabled while programming the complete timing set. Cursor position is an exception; CURSOR_POS is designed for single-write asynchronous updates during active display.

3.11 Host Interrupt Interface

VDP-7 interrupt sources are combined into one device interrupt indication. The surrounding board maps that indication to PCI, local-bus or system interrupt logic. The VDP-7 manual defines source status and acknowledge behavior; vector assignment and OS registration are platform-specific.

Drivers should mask device interrupts before reprogramming the host interrupt route during initialization or shutdown.

3.12 Bus Error Handling

A host access failure outside the VDP-7 core may not set `ERROR_STATUS`. PCI target abort, parity or bridge timeout indications must be collected from the host interface. Conversely, framebuffer bounds and command parser faults originate inside the VDP-7 and are reported through `ERROR_STATUS`.

Diagnostic logs should identify which layer reported the failure rather than collapsing all events into a generic graphics error.

3.13 Polling vs Interrupts

Boot firmware and diagnostics may poll FIFO or completion status. General-purpose drivers should use interrupts or bounded polling to avoid monopolizing the CPU during long blits or vertical-blank waits.

Any polling loop must include a timeout. A hardware fault, invalid display clock or lost host interrupt can otherwise produce an infinite wait that obscures the original failure.

3.14 Multithreaded Access

The hardware does not arbitrate software threads. A driver lock or equivalent serialization mechanism is required around shared register state and command emission.

High-level libraries may prepare command buffers concurrently, but final packet ordering must be established by the driver. Interleaving words from two variable-length packets causes a parser error and requires command-engine recovery.



3.15 Host Interface Diagnostic Test

A minimal host-interface test reads `DEVICE_ID` and `REVISION` repeatedly, writes and reads a software-owned `HOST_CONFIG` field, verifies FIFO free count after reset, submits a NOP and completion marker, and confirms the expected completion interrupt or status transition.

The test should be run before framebuffer timing tests because it isolates bus/interface problems from graphics-memory or display configuration.

3.16 Host Programming Checklist

Before releasing the device to higher software layers, verify 32-bit alignment, byte order, write-posting barrier behavior, FIFO accounting, interrupt routing, framebuffer mapping policy and timeout handling. Record any bridge-specific requirements in the platform driver rather than in application code.

The VDP-7 programming model assumes that these host-interface responsibilities are satisfied. Symptoms such as random parser errors, stale status and missing commands are frequently caused by host ordering mistakes rather than drawing-engine defects.



Chapter 4 - Reset and Initialization

4.1 Power-On State

After hardware reset, display output is disabled, command processing is held in a clean idle state, interrupts are masked, clip bounds reject drawing and framebuffer parameters are not considered valid. `DEVICE_ID` and `REVISION` are immediately readable once host reset is released.

Software shall not issue drawing packets until `STATUS` reports `READY` and the required framebuffer state has been programmed.

4.2 Full Initialization Sequence

A full driver initialization should identify the device, mask interrupts, clear stale status, reset the command engine, program host configuration, establish framebuffer bounds and format, initialize drawing state, program timing, verify clock lock at board level where applicable, enable scanout and finally enable selected interrupts.

The sequence is deliberately conservative and can be shortened only when the system firmware guarantees a known inherited display state.

4.3 Command-Engine Reset

Writing CMD_RESET in RESET aborts the current drawing operation, flushes decoder state and returns the FIFO/engine to idle after a bounded recovery interval. It does not automatically clear ERROR_STATUS.

After reset, software must restore drawing state that the reset domain invalidates. Display timing can remain active, allowing visible output to survive recovery.

4.4 Display-Engine Reset

Display reset disables scanout and resets horizontal/vertical counters. It should be used when timing state is invalid, when changing clock source or when a board-level display interface must be reinitialized.

Display reset does not clear framebuffer content. After reprogramming timing and start address, software can re-enable output and restore the prior visible surface.

4.5 Interrupt Initialization

Mask all sources before clearing INT_STATUS. Program INT_ENABLE only after the host interrupt route is established and the driver's ISR is ready.

A vertical-blank event can occur immediately after enabling display. If VBLANK interrupt is enabled too early, software may service an event before display ownership structures are initialized.

4.6 Framebuffer Initialization

Program FB_BASE and FB_LIMIT to the accessible memory range, then FB_PITCH and FB_FORMAT. Clear or test representative memory through accelerated fills before enabling scanout in a new board bring-up.

Framebuffer size is a board property; the VDP-7 does not discover installed memory capacity automatically. System firmware or board identification supplies the qualified limit.



4.7 Initial Clip and Drawing State

Set the clip rectangle to the target surface or visible screen before issuing rendering commands. Initialize origin to zero, scale to 1.0, foreground/background colors to known values and ROP to COPY.

Drivers should not rely on reset values for color or ROP even when a particular revision powers up in a useful state.

4.8 Timing Bring-Up

Keep DISP_ENABLE clear while writing horizontal and vertical timing registers. Program START_ADDR and pixel-format related scanout fields before enabling.

On first bring-up, select a conservative qualified mode and verify sync with an oscilloscope or known monitor before attempting custom high-resolution timings. AN-VDP7-05 contains board-level guidance beyond this programming reference.

4.9 Warm Driver Restart

A warm restart can preserve display timing, cursor and framebuffer content while resetting command state. Read and save required display registers before any partial reset that may affect them.

After command recovery, invalidate all host-side assumptions about pending FIFO contents. No partially submitted packet may be resumed; start at a known packet boundary.

4.10 Shutdown Sequence

Disable new command submissions, wait for idle or apply a bounded abort policy, mask interrupts, disable cursor and then disable display output if the system is relinquishing the device.

Do not remove framebuffer mappings while the display engine can still fetch scanout data. On PCI systems, disable bus mastering at the host bridge only after the VDP-7 is quiescent.

4.11 Initialization Example

```
/* Simplified register-level initialization */
write32(INT_ENABLE, 0);
write32(RESET, RESET_CMD);
wait_status(STATUS_CMD_IDLE);
write32(INT_STATUS, 0xffffffff);

write32(FB_BASE, fb_base);
write32(FB_LIMIT, fb_limit);
write32(FB_PITCH, pitch);
write32(FB_FORMAT, FMT_XRGB8888);

set_clip(0, 0, width, height);
program_timing(&mode);
write32(DISP_CTRL, DISP_ENABLE);

write32(INT_ENABLE, INT_CMD_DONE | INT_VBLANK | INT_ERROR);
```

The example omits host-bridge setup, clock programming, RAMDAC initialization and error handling. Production code must verify every wait with a timeout and must not assume that FB_FORMAT values apply to the external RAMDAC without board-specific setup.

4.12 Initialization Checklist

A qualified initialization path records device revision, proves a host ordering barrier, verifies framebuffer range, initializes all drawing state, validates timing before scanout, clears stale interrupts, uses bounded waits and emits one known command before accepting application traffic.

If any of these tests fails, keep higher graphics software detached and collect raw register/bridge state. Continuing through initialization after a partial failure usually converts a simple hardware/configuration issue into a secondary command or display fault that is harder to diagnose.

Chapter 5 - Register Reference

5.01 Register Reference - DEVICE_ID

DEVICE_ID - Offset 0x000

Access	RO
Reset	0x56445037

Bits	Field	Reset	Description
31:16	VENDOR	0x534C	Sinneslöschen device vendor code.
15:0	DEVICE	0x5037	VDP-7 device identifier.

Programming Notes. DEVICE ID participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to DEVICE ID should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.02 Register Reference - REVISION

REVISION - Offset 0x004

Access	RO
Reset	varies

Bits	Field	Reset	Description
31:24	MAJOR	-	Silicon major revision.
23:16	MINOR	-	Silicon minor revision.
15:8	FEATURE	-	Programming feature level.
7:0	STEPPING	-	Manufacturing stepping.

Programming Notes. REVISION participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to REVISION should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.03 Register Reference - RESET

RESET - Offset 0x008

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	CMD_RESET	0	Reset command/raster engine.
1	DISP_RESET	0	Reset display timing domain.
2	DIAG_RESET	0	Reset diagnostic counters.
31:3	RSVD	0	Write zero.

Programming Notes. RESET participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to RESET should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.04 Register Reference - STATUS

STATUS - Offset 0x00C

Access	RO
Reset	0x00000003

Bits	Field	Reset	Description
0	READY	1	Host register interface ready.
1	CMD_IDLE	1	Command engine idle.
2	DISP_ACTIVE	0	Display scanout enabled and active.
3	VBLANK	0	Current scan position is vertical blank.
4	FIFO_EMPTY	1	Command FIFO empty.
5	ERROR	0	One or more ERROR_STATUS bits set.

Programming Notes. STATUS participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to STATUS should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.05 Register Reference - INT_STATUS

INT_STATUS - Offset 0x010

Access	W1C
Reset	0x00000000

Bits	Field	Reset	Description
0	CMD_DONE	0	Completion marker reached.
1	FIFO_LOW	0	FIFO occupancy at/below watermark condition.
2	VBLANK	0	Vertical blank transition.
3	ERROR	0	Error event latched.
4	DIAG	0	Diagnostic event.
31:5	RSVD	0	Write zero.

Programming Notes. INT STATUS participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to INT STATUS should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.06 Register Reference - INT_ENABLE

INT_ENABLE - Offset 0x014

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
4:0	ENABLE	0	One enable bit corresponding to INT_STATUS sources.
31:5	RSVD	0	Write zero.

Programming Notes. INT_ENABLE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to INT_ENABLE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.07 Register Reference - INT_ACK

INT_ACK - Offset 0x018

Access	WO		
Reset	0x00000000		

Bits	Field	Reset	Description
31:0	ACK	0	Optional platform interrupt acknowledge write; use only where board documentation requires.

Programming Notes. INT ACK participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to INT ACK should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.08 Register Reference - ERROR_STATUS

ERROR_STATUS - Offset 0x01C

Access	W1C
Reset	0x00000000

Bits	Field	Reset	Description
0	CMD_FORMAT	0	Malformed opcode/packet length.
1	FIFO_OVERFLOW	0	Command submission exceeded FIFO capacity.
2	FB_BOUNDS	0	Calculated framebuffer access outside programmed range.
3	DISPLAY_TIMING	0	Invalid or inconsistent display timing.
4	MEMORY_TIME-OUT	0	Framebuffer transaction did not complete.
5	DIAG_FAULT	0	Internal diagnostic fault.
31:6	RSVD	0	Write zero.

Programming Notes. ERROR STATUS participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to ERROR STATUS should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.09 Register Reference - HOST_CTRL

HOST_CTRL - Offset 0x020

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	CMD_ENABLE	0	Allow command processor to consume FIFO.
1	BOUNDS_ENABLE	0	Enable framebuffer bounds checks.
2	POSTED_OK	0	Indicate qualified posted-write environment.
31:3	RSVD	0	Write zero.

Programming Notes. HOST_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to HOST_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.10 Register Reference - HOST_CONFIG

HOST_CONFIG - Offset 0x024

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
1:0	BYTE_ORDER	0	Host word ordering selection where supported.
2	STRICT_ORDER	0	Request strict command/register ordering.
7:3	RSVD	0	Write zero.
15:8	HOST_TAG	0	Software diagnostic tag reflected in SIGNATURE.
31:16	RSVD	0	Write zero.

Programming Notes. HOST CONFIG participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to HOST CONFIG should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.11 Register Reference - FIFO_STATUS

FIFO_STATUS - Offset 0x028

Access	RO
Reset	0x00400000

Bits	Field	Reset	Description
6:0	OCCUPIED	0	Words currently held in FIFO.
14:8	FREE	64	Free FIFO words.
16	FULL	0	FIFO full.
17	EMPTY	1	FIFO empty.
18	PARSER_BUSY	0	Decoder executing packet.
31:19	RSVD	0	Ignore.

Programming Notes. FIFO STATUS participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FIFO STATUS should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.12 Register Reference - FIFO_WATERMARK

FIFO_WATERMARK - Offset 0x02C

Access	R/W
Reset	0x00000010

Bits	Field	Reset	Description
6:0	LOW_MARK	16	FIFO free-word threshold for FIFO_LOW event.
31:7	RSVD	0	Write zero.

Programming Notes. FIFO WATERMARK participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FIFO WATERMARK should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.13 Register Reference - CMD_PORT

CMD_PORT - Offset 0x030

Access	WO
Reset	undefined

Bits	Field	Reset	Description
31:0	WORD	-	Next 32-bit command packet word.

Programming Notes. CMD PORT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CMD PORT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.14 Register Reference - CMD_BASE

CMD_BASE - Offset 0x034

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:5	BASE	0	Optional host command-buffer base, 32-byte aligned.
4:0	RSVD	0	Write zero.

Programming Notes. CMD BASE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CMD BASE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.15 Register Reference - CMD_LIMIT

CMD_LIMIT - Offset 0x038

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:5	LIMIT	0	Last valid command-buffer block address.
4:0	RSVD	0	Write zero.

Programming Notes. CMD LIMIT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CMD LIMIT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.16 Register Reference - CMD_PTR

CMD_PTR - Offset 0x03C

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:2	PTR	0	Current command-buffer pointer for board modes supporting memory fetch.
1:0	RSVD	0	Write zero.

Programming Notes. CMD_PTR participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CMD_PTR should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.17 Register Reference - FB_BASE

FB_BASE - Offset 0x040

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:8	BASE	0	Framebuffer address base, 256-byte aligned.
7:0	RSVD	0	Write zero.

Programming Notes. FB BASE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FB BASE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.18 Register Reference - FB_PITCH

FB_PITCH - Offset 0x044

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
15:0	BYTES_PER_LINE	0	Byte distance between adjacent scan lines/surface rows.
31:16	RSVD	0	Write zero.

Programming Notes. FB PITCH participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FB PITCH should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.19 Register Reference - FB_FORMAT

FB_FORMAT - Offset 0x048

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
2:0	PIXEL_FMT	0	0=8bpp indexed, 1=RGB565, 2=XRGB8888.
3	SWAP16	0	Swap bytes within 16-bit pixels.
4	SWAP32	0	Swap halfwords within 32-bit pixels.
31:5	RSVD	0	Write zero.

Programming Notes. FB_FORMAT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FB_FORMAT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.20 Register Reference - FB_LIMIT

FB_LIMIT - Offset 0x04C

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:8	LIMIT	0	Highest accessible framebuffer byte block.
7:0	RSVD	0	Write zero.

Programming Notes. FB_LIMIT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FB_LIMIT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.21 Register Reference - SCRATCH_BASE

SCRATCH_BASE - Offset 0x050

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:8	BASE	0	Scratch/temporary surface base address.
7:0	RSVD	0	Write zero.

Programming Notes. SCRATCH BASE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to SCRATCH BASE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.22 Register Reference - SCRATCH_LIMIT

SCRATCH_LIMIT - Offset 0x054

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
31:8	LIMIT	0	Scratch surface upper bound.
7:0	RSVD	0	Write zero.

Programming Notes. SCRATCH LIMIT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to SCRATCH LIMIT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.23 Register Reference - CLIP_LEFT_TOP

CLIP_LEFT_TOP - Offset 0x058

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
15:0	LEFT	0	Left inclusive clip coordinate.
31:16	TOP	0	Top inclusive clip coordinate.

Programming Notes. CLIP LEFT TOP participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CLIP LEFT TOP should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.24 Register Reference - CLIP_RIGHT_BOTTOM

CLIP_RIGHT_BOTTOM - Offset 0x05C

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
15:0	RIGHT	0	Right exclusive clip coordinate.
31:16	BOTTOM	0	Bottom exclusive clip coordinate.

Programming Notes. CLIP RIGHT BOTTOM participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CLIP RIGHT BOTTOM should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.25 Register Reference - ORIGIN

ORIGIN - Offset 0x060

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	X	0	Signed origin X integer pixels.
31:16	Y	0	Signed origin Y integer pixels.

Programming Notes. ORIGIN participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to ORIGIN should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.26 Register Reference - SCALE_X

SCALE_X - Offset 0x064

Access	R/W
Reset	0x00010000

Bits	Field	Reset	Description
31:0	SCALE	1.0	16.16 scale factor applied where scaling state is enabled.

Programming Notes. SCALE X participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to SCALE X should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.27 Register Reference - SCALE_Y

SCALE_Y - Offset 0x068

Access	R/W		
Reset	0x00010000		

Bits	Field	Reset	Description
31:0	SCALE	1.0	16.16 vertical scale factor.

Programming Notes. SCALE Y participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to SCALE Y should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.28 Register Reference - LINE_CTRL

LINE_CTRL - Offset 0x06C

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	LAST_PIXEL_OFF	0	Suppress final endpoint.
1	AA_HINT	0	Reserved compatibility hint; no full antialiasing guarantee.
3:2	TIE_RULE	0	Diamond-exit/tie behavior selection for compatibility.
31:4	RSVD	0	Write zero.

Programming Notes. LINE_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to LINE_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.29 Register Reference - FG_COLOR

FG_COLOR - Offset 0x070

Access	R/W		
Reset	0xFFFFFFFF		

Bits	Field	Reset	Description
31:0	COLOR	all 1	Foreground color in current framebuffer format.

Programming Notes. FG COLOR participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to FG COLOR should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.30 Register Reference - BG_COLOR

BG_COLOR - Offset 0x074

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
31:0	COLOR	0	Background color for expansion/pattern operations.

Programming Notes. BG COLOR participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BG COLOR should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.31 Register Reference - ROP

ROP - Offset 0x078

Access	R/W		
Reset	0x0000000C		

Bits	Field	Reset	Description
3:0	ROP_CODE	0xC	Logical raster operation; 0xC = source copy.
31:4	RSVD	0	Write zero.

Programming Notes. ROP participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to ROP should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.32 Register Reference - PATTERN

PATTERN - Offset 0x07C

Access	R/W
Reset	0xFFFFFFFF

Bits	Field	Reset	Description
31:0	PATTERN_BITS	all 1	32-bit repeating monochrome pattern source.

Programming Notes. PATTERN participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to PATTERN should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.33 Register Reference - BLT_SRC_BASE

BLT_SRC_BASE - Offset 0x080

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
31:8	BASE	0	Source surface base address.
7:0	RSVD	0	Write zero.

Programming Notes. BLT SRC BASE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BLT SRC BASE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.34 Register Reference - BLT_DST_BASE

BLT_DST_BASE - Offset 0x084

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:8	BASE	0	Destination surface base address.
7:0	RSVD	0	Write zero.

Programming Notes. BLT DST BASE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BLT DST BASE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.35 Register Reference - BLT_SRC_XY

BLT_SRC_XY - Offset 0x088

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	X	0	Source X integer pixels.
31:16	Y	0	Source Y integer pixels.

Programming Notes. BLT SRC XY participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BLT SRC XY should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.36 Register Reference - BLT_DST_XY

BLT_DST_XY - Offset 0x08C

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	X	0	Destination X integer pixels.
31:16	Y	0	Destination Y integer pixels.

Programming Notes. BLT_DST_XY participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BLT_DST_XY should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.37 Register Reference - BLT_SIZE

BLT_SIZE - Offset 0x090

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
15:0	WIDTH	0	Rectangle width in pixels.
31:16	HEIGHT	0	Rectangle height in pixels.

Programming Notes. BLT_SIZE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BLT_SIZE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.38 Register Reference - BLT_CTRL

BLT_CTRL - Offset 0x094

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	X_REVERSE	0	Process right-to-left.
1	Y_REVERSE	0	Process bottom-to-top.
2	SCALE_ENABLE	0	Enable scale factors for stretch blit.
3	COLOR_EXPAND	0	Interpret source as monochrome expansion.
31:4	RSVD	0	Write zero.

Programming Notes. BLT_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to BLT_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.39 Register Reference - POLY_CTRL

POLY_CTRL - Offset 0x098

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	FILL_ENABLE	0	Enable filled polygon generation.
1	EDGE_INCLUSIVE	0	Select edge convention.
2	WINDING_HINT	0	Compatibility hint for vertex ordering.
31:3	RSVD	0	Write zero.

Programming Notes. POLY_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to POLY_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.40 Register Reference - EDGE0

EDGE0 - Offset 0x09C

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
31:0	EDGE_STATE	0	Implementation-defined edge accumulator seed used by low-level diagnostics.

Programming Notes. EDGE0 participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to EDGE0 should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.41 Register Reference - EDGE1

EDGE1 - Offset 0x0A0

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
31:0	EDGE_STATE	0	Second edge accumulator seed; normal software uses polygon commands instead.

Programming Notes. EDGE1 participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to EDGE1 should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.42 Register Reference - VERTEX_PORT

VERTEX_PORT - Offset 0x0A4

Access	WO
Reset	undefined

Bits	Field	Reset	Description
31:0	VERTEX_WORD	-	Packed X/Y or coordinate word for variable-length polygon submission.

Programming Notes. VERTEX PORT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to VERTEX PORT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.43 Register Reference - VERTEX_COUNT

VERTEX_COUNT - Offset 0x0A8

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
7:0	COUNT	0	Expected vertex count for selected low-level polygon mode.
31:8	RSVD	0	Write zero.

Programming Notes. VERTEX COUNT participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to VERTEX COUNT should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.44 Register Reference - CURSOR_CTRL

CURSOR_CTRL - Offset 0x0AC

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
0	ENABLE	0	Enable hardware cursor.
1	FORMAT	0	Cursor encoding selection.
2	HOTSPOT_EN- ABLE	0	Apply hotspot offsets.
31:3	RSVD	0	Write zero.

Programming Notes. CURSOR_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CURSOR_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.45 Register Reference - CURSOR_POS, CURSOR_BASE

CURSOR_POS - Offset 0x0B0

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	X	0	Signed screen X.
31:16	Y	0	Signed screen Y.

Programming Notes. CURSOR POS participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CURSOR POS should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

CURSOR_BASE - Offset 0x0B4

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:8	BASE	0	Cursor image base, 256-byte aligned.
7:0	RSVD	0	Write zero.

Programming Notes. CURSOR BASE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CURSOR BASE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.46 Register Reference - CURSOR_COLOR0, CURSOR_COLOR1

CURSOR_COLOR0 - Offset 0x0B8

Access	R/W
Reset	0x00FFFFFF

Bits	Field	Reset	Description
31:0	COLOR0	white	Cursor programmable color 0.

Programming Notes. CURSOR_COLOR0 participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CURSOR_COLOR0 should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

CURSOR_COLOR1 - Offset 0x0BC

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
31:0	COLOR1	black	Cursor programmable color 1.

Programming Notes. CURSOR_COLOR1 participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CURSOR_COLOR1 should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.47 Register Reference - DISP_CTRL, H_TOTAL

DISP_CTRL - Offset 0x0C0

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	ENABLE	0	Enable display scanout.
1	SYNC_ENABLE	0	Enable sync outputs.
2	BLANK_FORCE	0	Force blanking independent of counters.
3	INTERLACE	0	Interlace timing mode where board supports.
4	HSYNC_POL	0	Horizontal sync polarity.
5	VSYSN_POL	0	Vertical sync polarity.
31:6	RSVD	0	Write zero.

Programming Notes. DISP_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to DISP_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

H_TOTAL - Offset 0x0C4

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
15:0	TOTAL	0	Total pixel clocks per line minus one.
31:16	ACTIVE	0	Active pixels per line minus one.

Programming Notes. H_TOTAL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to H_TOTAL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.48 Register Reference - H_SYNC, H_BLANK

H_SYNC - Offset 0x0C8

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	START	0	Horizontal sync start.
31:16	END	0	Horizontal sync end.

Programming Notes. H_SYNC participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to H_SYNC should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

H_BLANK - Offset 0x0CC

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	START	0	Horizontal blank start.
31:16	END	0	Horizontal blank end.

Programming Notes. H_BLANK participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to H_BLANK should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.49 Register Reference - V_TOTAL, V_SYNC

V_TOTAL - Offset 0x0D0

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	TOTAL	0	Total lines per frame minus one.
31:16	ACTIVE	0	Active lines per frame minus one.

Programming Notes. V TOTAL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to V TOTAL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

V_SYNC - Offset 0x0D4

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	START	0	Vertical sync start line.
31:16	END	0	Vertical sync end line.

Programming Notes. V SYNC participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to V SYNC should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.50 Register Reference - V_BLANK, START_ADDR

V_BLANK - Offset 0x0D8

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
15:0	START	0	Vertical blank start line.
31:16	END	0	Vertical blank end line.

Programming Notes. V BLANK participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to V BLANK should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

START_ADDR - Offset 0x0DC

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
31:3	ADDRESS	0	Display scanout start address with 8-byte granularity.
2:0	RSVD	0	Write zero.

Programming Notes. START_ADDR participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to START_ADDR should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.51 Register Reference - LINE_COMPARE, CLOCK_CTRL

LINE_COMPARE - Offset 0x0E0

Access	R/W
Reset	0x0000FFFF

Bits	Field	Reset	Description
15:0	LINE	0xFFFF	Optional scan line compare event; 0xFFFF disables.
31:16	RSVD	0	Write zero.

Programming Notes. LINE_COMPARE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to LINE_COMPARE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

CLOCK_CTRL - Offset 0x0E4

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
3:0	CLOCK_SEL	0	Board-defined qualified pixel clock selection.
4	CLK_DIV2	0	Optional divide by two.
31:5	RSVD	0	Write zero.

Programming Notes. CLOCK_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to CLOCK_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.52 Register Reference - DAC_CTRL, LUT_ADDR

DAC_CTRL - Offset 0x0E8

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
1:0	DAC_MODE	0	Board-level RAMDAC mode hint.
2	BLANK_POL	0	Blanking polarity control.
31:3	RSVD	0	Write zero.

Programming Notes. DAC_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to DAC_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

LUT_ADDR - Offset 0x0EC

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
7:0	INDEX	0	Palette/LUT address.
31:8	RSVD	0	Write zero.

Programming Notes. LUT_ADDR participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to LUT_ADDR should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.53 Register Reference - LUT_DATA, GPIO_CTRL

LUT_DATA - Offset 0x0F0

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
23:16	R	0	Red component.
15:8	G	0	Green component.
7:0	B	0	Blue component.
31:24	RSVD	0	Write zero.

Programming Notes. LUT DATA participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to LUT DATA should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

GPIO_CTRL - Offset 0x0F4

Access	R/W		
Reset	0x00000000		

Bits	Field	Reset	Description
7:0	DIR	0	Direction bits for board-qualified general-purpose pins.
15:8	ALT	0	Alternate function select.
31:16	RSVD	0	Write zero.

Programming Notes. GPIO_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to GPIO_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.54 Register Reference - GPIO_DATA, DIAG_CTRL

GPIO_DATA - Offset 0x0F8

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
7:0	DATA	0	General-purpose data bits where implemented.
31:8	RSVD	0	Write zero.

Programming Notes. GPIO DATA participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to GPIO DATA should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

DIAG_CTRL - Offset 0x0FC

Access	R/W
Reset	0x00000000

Bits	Field	Reset	Description
0	PERF_ENABLE	0	Enable performance counters.
1	SIGNATURE_ENABLE	0	Enable signature accumulation.
2	TEST_MODE	0	Authorized diagnostic mode; normal software leaves zero.
31:3	RSVD	0	Write zero.

Programming Notes. DIAG_CTRL participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to DIAG_CTRL should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.55 Register Reference - PERF_COUNT0, PERF_COUNT1

PERF_COUNT0 - Offset 0x100

Access	RO
Reset	0x00000000

Bits	Field	Reset	Description
31:0	COUNT	0	Framebuffer write-cycle or command count selected by DIAG_CTRL.

Programming Notes. PERF_COUNT0 participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to PERF_COUNT0 should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

PERF_COUNT1 - Offset 0x104

Access	RO
Reset	0x00000000

Bits	Field	Reset	Description
31:0	COUNT	0	Framebuffer stall or scanout arbitration count selected by DIAG_CTRL.

Programming Notes. PERF_COUNT1 participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to PERF_COUNT1 should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

5.56 Register Reference - SIGNATURE, TEST_STATUS

SIGNATURE - Offset 0x108

Access	RO
Reset	0x00000000

Bits	Field	Reset	Description
31:0	VALUE	0	Diagnostic signature of selected internal activity.

Programming Notes. SIGNATURE participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to SIGNATURE should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

TEST_STATUS - Offset 0x10C

Access	RO
Reset	revision dependent

Bits	Field	Reset	Description
7:0	STATE	-	Internal diagnostic state summary.
15:8	FLAGS	-	Revision-dependent diagnostic flags.
31:16	RSVD	-	Ignore.

Programming Notes. TEST STATUS participates in the VDP-7 programming model exactly as described by the access type above. Software should write reserved fields as zero and should not use undocumented read values for revision detection. When a host bridge performs posted writes, read FIFO_STATUS or another defined readable register before assuming that a change has reached the device.

Interaction. Changes to TEST STATUS should be ordered with commands that depend on the new value. If the register controls display or framebuffer behavior, perform the change while the affected engine is idle or disabled unless this reference explicitly identifies the field as asynchronous-safe.

Diagnostic Guidance. When diagnosing unexpected behavior involving this register, record its value together with STATUS, ERROR_STATUS, REVISION and the command or mode that first exposed the problem. A field value that appears correct in isolation can still be incompatible with framebuffer pitch, pixel format, timing or host ordering.

Chapter 6 - Command Processor and FIFO

6.1 Command Packet Format

Every command begins with a 32-bit header containing opcode and length. Fixed-length commands have one defined length; variable-length commands encode the total word count. The decoder validates enough information to locate the next packet boundary before execution.

Reserved header bits must be zero. A nonzero reserved combination can be rejected as CMD_FORMAT even when the packet length is otherwise correct.

6.2 Header Encoding

Bits	Name	Meaning
31:24	OPCODE	Command operation code.
23:16	LENGTH	Total packet words including header; zero reserved.
15:8	FLAGS	Command-specific execution flags.
7:0	TAG	Software tag copied to selected completion/debug state.

The TAG field has no rendering effect. Drivers may use it to correlate completion markers or diagnostic traces with host command batches.

6.3 Fixed-Length Packets

Fixed-length packets carry a predetermined sequence of words such as coordinate pairs, dimensions or colors. Software should construct packets in host memory first and emit them only after verifying adequate FIFO space.

Do not write a header and then block indefinitely waiting for space for the remainder of the same packet while another software context can access CMD_PORT. Command emission must be serialized for the complete packet.

6.4 Variable-Length Packets

Polyline and polygon operations may carry a variable number of vertices. LENGTH identifies the entire packet. Hardware validates the minimum and maximum length for the opcode.

For large geometry, split work into multiple legal packets rather than exceeding LENGTH or FIFO constraints. Preserve shared endpoints carefully when last-pixel suppression is used.

6.5 FIFO Watermark Strategy

The recommended driver strategy maintains a software reserve of at least eight FIFO words beyond the largest asynchronous packet an interrupt path may need. Applications should never busy-loop on individual word availability when a batch can be buffered in host memory.

On C1/C2 silicon the FIFO_LOW event is based on free-word threshold. Earlier documentation described occupancy threshold; Revision C corrects that ambiguity.

6.6 Completion Markers

A completion marker is an ordered command that generates CMD_DONE after all preceding drawing operations reach the defined completion point. It is the preferred mechanism for fence-style synchronization.

Multiple markers can be outstanding. Software tags allow a driver to associate completion with a host batch, but the base INT_STATUS bit only indicates that at least one marker completed; the driver must consult its own queue/state.

6.7 WAIT_IDLE

WAIT_IDLE prevents following commands from executing until active drawing engines are idle. It is useful before changing state that cannot be safely modified in flight or before CPU framebuffer readback.

WAIT_IDLE is stronger than waiting for FIFO empty. Excessive barriers reduce parallelism and can significantly lower blit/line throughput.



6.8 WAIT_VBLANK

WAIT_VBLANK delays subsequent commands until the next qualified vertical-blank interval. It can be used to order page flips or cursor image updates.

A disabled display never enters normal vertical blank; submitting WAIT_VBLANK while scanout is disabled can stall the stream. Drivers should test DISP_ACTIVE before using the command.

6.9 State Commands

State commands update drawing registers through the ordered command stream. They are useful in buffered/multitasking drivers because state takes effect at a precise packet position relative to rendering operations.

Direct register writes remain appropriate during initialization and while the engine is idle. Mixing direct state writes with queued rendering requires explicit ordering.

6.10 Error Recovery

On CMD_FORMAT, stop new submissions, flush host posting, capture FIFO/STATUS/ERROR state, reset the command engine, clear the error and restore state. Do not attempt to guess the parser position and continue after the malformed packet.

If FIFO_OVERFLOW is reported, audit host-side free-count logic and bridge retry behavior before treating the condition as a hardware defect.

6.11 Command Tags

The eight-bit command tag is copied into selected completion/diagnostic state where the opcode supports it. Tags are not unique identifiers enforced by hardware; software may reuse them after the associated command has completed.

A ring of 256 tags is sufficient for most driver fence queues when combined with a monotonic host sequence number.

6.12 Command Alignment

CMD_PORT accepts words rather than byte streams, so packet words are naturally 32-bit aligned. Memory-fetched command buffers, when supported by the system board, must begin on the alignment specified by CMD_BASE.

Do not concatenate binary command data with packed byte structures without explicit alignment in C or assembly source.

6.13 Command Submission Example

```
static void emit_packet(const uint32_t *p, unsigned words)
{
    while (fifo_free_words() < words)
        wait_fifo_event();

    for (unsigned i = 0; i < words; ++i)
        write32(CMD_PORT, p[i]);

    (void)read32(FIFO_STATUS); /* posting barrier */
}
```

The example assumes one serialized submitter. Production code must handle packets larger than the driver's immediate FIFO policy and should integrate timeouts/error detection.

6.14 Packet Trace Interpretation

VDP Development Toolkit traces represent each packet as header plus decoded parameters. When comparing a trace with application intent, verify not only the opcode but also the state that was active when the command executed.

A correct LINE packet can draw the wrong color if an earlier state packet was omitted or reordered. Trace analysis should therefore preserve state changes, barriers and completion markers.

6.15 Throughput Considerations

Command overhead is smallest when applications batch related primitives with stable state. Reprogramming clip/color/ROP before every primitive consumes FIFO bandwidth and decoder cycles.

Conversely, extremely large batches can increase latency for other clients. Workstation drivers normally balance throughput and responsiveness by limiting batch size and inserting completion points at resource boundaries.

6.16 Command Processor Checklist

A robust command path validates packet lengths, serializes emission, accounts FIFO words before writing, maintains a posting barrier, uses bounded waits, records completion markers and has a defined parser-error recovery path.

If any layer above the driver can write CMD_PORT directly, these guarantees cannot be maintained. Direct application access is therefore unsupported in multitasking workstation environments.

Chapter 7 - Drawing and Transfer Commands

7.01 Command - NOP

Opcode	0x00
Packet length	1 word(s)
Purpose	No operation; used for alignment and diagnostic submission tests.

Packet Parameters

No parameters.

Programming Behavior

Has no rendering side effect; still participates in packet ordering. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x00 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.02 Command - SET_STATE

Opcode	0x01
Packet length	3 word(s)
Purpose	Update a selected drawing-state register through the ordered stream.

Packet Parameters

Word 1 = state selector; Word 2 = value.

Programming Behavior

Only selectors documented for stream programming are legal. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x01 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.03 Command - SET_CLIP

Opcode	0x02
Packet length	3 word(s)
Purpose	Set active clip rectangle.

Packet Parameters

Word 1 = left/top; Word 2 = right/bottom.

Programming Behavior

Right and bottom are exclusive. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x02 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.04 Command - SET_ORIGIN

Opcode	0x03
Packet length	2 word(s)
Purpose	Set signed drawing origin.

Packet Parameters

Word 1 packs signed X/Y integer origin.

Programming Behavior

Applied before clip evaluation. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x03 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.05 Command - SET_COLOR

Opcode	0x04
Packet length	3 word(s)
Purpose	Set foreground and background colors.

Packet Parameters

Word 1 = foreground; Word 2 = background.

Programming Behavior

Values interpreted in current framebuffer format. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x04 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.



7.06 Command - LINE

Opcode	0x10
Packet length	5 word(s)
Purpose	Draw one line segment using current color/ROP/clip state.

Packet Parameters

Words 1-2 = X0/Y0; Words 3-4 = X1/Y1 in 16.8 format.

Programming Behavior

Endpoint convention follows LINE_CTRL. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x10 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.07 Command - POLYLINE

Opcode	0x11
Packet length	variable word(s)
Purpose	Draw connected line segments.

Packet Parameters

Header + packed vertex coordinate words.

Programming Behavior

Shared vertices are not drawn twice when last-pixel rules are enabled. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x11 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.08 Command - POINTS

Opcode	0x12
Packet length	variable word(s)
Purpose	Draw independent single-pixel points.

Packet Parameters

Header + packed coordinate pairs.

Programming Behavior

Useful for markers; clipping applies per point. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x12 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.



7.09 Command - RECT_OUTLINE

Opcode	0x13
Packet length	3 word(s)
Purpose	Draw axis-aligned rectangle outline.

Packet Parameters

Word 1 = X/Y; Word 2 = width/height.

Programming Behavior

Zero width or height produces no pixels. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x13 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.10 Command - RECT_FILL

Opcode	0x14
Packet length	3 word(s)
Purpose	Fill axis-aligned rectangle.

Packet Parameters

Word 1 = X/Y; Word 2 = width/height.

Programming Behavior

Fast path for clears and UI regions. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x14 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.



7.11 Command - TRIANGLE_FILL

Opcode	0x15
Packet length	7 word(s)
Purpose	Fill one triangle.

Packet Parameters

Three 16.8 coordinate pairs.

Programming Behavior

Top-left edge rule used unless POLY_CTRL override selected. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x15 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.12 Command - POLY_FILL

Opcode	0x16
Packet length	variable word(s)
Purpose	Fill a convex polygon.

Packet Parameters

Vertex list; 3-32 vertices.

Programming Behavior

Concave polygons must be decomposed by software. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

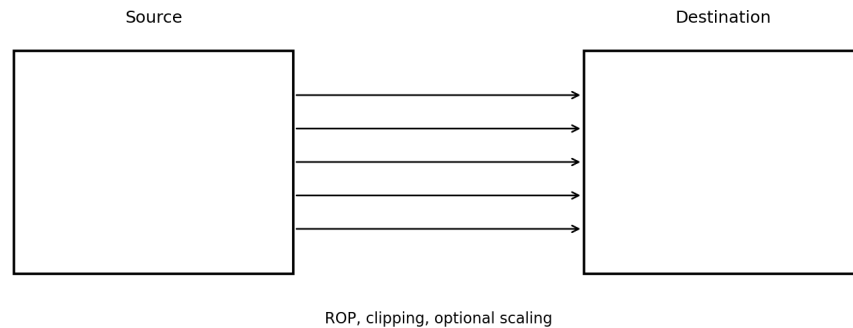
Error Conditions

A packet length inconsistent with opcode 0x16 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.13 Command - BLIT



Opcode	0x20
Packet length	5 word(s)
Purpose	Copy a rectangle between framebuffer surfaces.

Packet Parameters

Source XY, destination XY, width/height and control.

Programming Behavior

Overlap requires correct X/Y reverse selection. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x20 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.14 Command - STRETCH_BLIT

Opcode	0x21
Packet length	7 word(s)
Purpose	Scale a source rectangle into destination rectangle.

Packet Parameters

Source/destination origins plus source and destination dimensions.

Programming Behavior

Nearest-sample technical scaling; not a resampling filter. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x21 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.15 Command - HOST_TO_FB

Opcode	0x22
Packet length	variable word(s)
Purpose	Transfer host-supplied pixel words into framebuffer.

Packet Parameters

Destination plus inline or board-level stream data.

Programming Behavior

Use chunking; format follows FB_FORMAT. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x22 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.16 Command - FB_TO_HOST

Opcode	0x23
Packet length	4 word(s)
Purpose	Prepare framebuffer readback transfer.

Packet Parameters

Source origin and dimensions.

Programming Behavior

Board/driver supplies actual host read mechanism. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x23 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.17 Command - PATTERN_FILL

Opcode	0x24
Packet length	4 word(s)
Purpose	Fill rectangle using repeating PATTERN bits and colors.

Packet Parameters

Destination XY, dimensions, pattern origin.

Programming Behavior

ROP applied after color expansion. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x24 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.18 Command - MONO_EXPAND

Opcode	0x25
Packet length	variable word(s)
Purpose	Expand 1-bpp source data to foreground/background colors.

Packet Parameters

Destination, dimensions, packed monochrome words.

Programming Behavior

Transparent background flag may suppress zero bits. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x25 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.19 Command - CLEAR_SURFACE

Opcode	0x26
Packet length	4 word(s)
Purpose	Clear defined surface region with constant color.

Packet Parameters

Base/pitch/extent selected by packet.

Programming Behavior

May bypass destination read for COPY operation. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x26 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.20 Command - SET_CURSOR

Opcode	0x30
Packet length	3 word(s)
Purpose	Update cursor position/control in ordered stream.

Packet Parameters

Position and control words.

Programming Behavior

Direct CURSOR_POS writes remain lower latency for pointer motion. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x30 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.21 Command - LOAD_LUT

Opcode	0x31
Packet length	variable word(s)
Purpose	Load consecutive palette/LUT entries.

Packet Parameters

Start index followed by RGB words.

Programming Behavior

Prefer vertical blank for large visible palette changes. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x31 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.22 Command - WAIT_VBLANK

Opcode	0x40
Packet length	1 word(s)
Purpose	Wait for next vertical blank boundary.

Packet Parameters

No parameters.

Programming Behavior

Display must be active. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x40 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.23 Command - WAIT_IDLE

Opcode	0x41
Packet length	1 word(s)
Purpose	Wait for drawing engines to become idle.

Packet Parameters

No parameters.

Programming Behavior

Does not disable display scanout. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x41 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

7.24 Command - SYNC_MARKER

Opcode	0x42
Packet length	2 word(s)
Purpose	Generate ordered completion event/tag.

Packet Parameters

Word 1 = software marker value.

Programming Behavior

Preferred fence primitive. The command is ordered with all preceding packets and uses the rendering state active when execution reaches this packet. Coordinates outside the active clip window are rejected or reduced according to the command's clipping rules.

Error Conditions

A packet length inconsistent with opcode 0x42 sets CMD_FORMAT. Framebuffer address calculation outside FB_BASE/FB_LIMIT with bounds checking enabled sets FB_BOUNDS. A memory timeout sets MEMORY_TIMEOUT and terminates the operation.

Performance Notes

Batch commands of the same type when state is stable. Avoid unnecessary WAIT_IDLE packets between independent operations. Destination-read ROPs and unaligned surfaces may reduce throughput relative to simple source-copy or constant-fill paths.

Chapter 8 - Framebuffer and Pixel Formats

8.1 Framebuffer Organization

VDP-7 treats framebuffer memory as a linear byte-addressed store divided into surfaces by base address, pitch and format. The processor does not maintain a surface directory; software supplies addresses and dimensions.

The visible display is simply one surface selected by `START_ADDR` and display pitch/format state. Other off-screen surfaces can coexist in the same memory range and are referenced by blit/command parameters.

8.2 8-bit Indexed Format

In indexed mode each pixel occupies one byte containing a LUT index. Scanout expands the index through the programmed color table. Drawing colors and logical ROP values are therefore index values rather than direct RGB components.

Indexed mode offers low memory bandwidth and is useful for instrumentation displays with controlled palettes. Palette updates can change the visible color of existing framebuffer pixels without redrawing.



8.3 RGB565 Format

RGB565 stores red in five bits, green in six and blue in five. Pixels occupy two bytes. The logical ROP unit operates on the complete 16-bit stored pixel value.

When SWAP16 is used to accommodate board/host byte ordering, CPU-visible memory layout may differ from the canonical bit diagram. Use the driver surface-format description rather than hard-coded byte assumptions.



8.4 XRGB8888 Format

XRGB8888 stores one pixel in 32 bits. The upper unused byte is ignored for display but may participate in certain logical ROP operations because the memory word is 32 bits. Drivers should write a consistent value to the unused bits when surfaces are shared with CPU software.

This format is the preferred mode for Graphics Library 4.1 on SV-97/SV-98D systems when memory bandwidth permits.

8.5 Pitch Rules

Pitch is the byte distance between the start of adjacent rows. It may exceed visible width to satisfy alignment, allocate padding or create virtual screens.

Pitch must be large enough for the surface width in the current format. Certain accelerated operations reach best throughput when pitch is a multiple of 32 bytes. Display scanout may impose stricter board-specific alignment.

8.6 Surface Alignment

FB_BASE and cursor/scratch bases have explicit alignment requirements. General blit source/destination bases should be aligned to at least the natural pixel width and preferably 32 bytes for burst-friendly accesses.

Misalignment is legal in selected operations but can split memory transactions and reduce throughput. Alignment requirements are distinct from clipping; a clipped operation can still begin from an unaligned source.

8.7 Overlapping Blits

When source and destination rectangles overlap in the same surface, copy direction matters. Set `X_REVERSE` and/or `Y_REVERSE` so source pixels are read before being overwritten.

The required direction is analogous to `memmove` rather than `memcpy`. Drivers should calculate overlap and direction automatically; applications should not be expected to choose these low-level flags.

8.8 Host Transfers

Host-to-framebuffer transfer is useful for glyphs, images and initialization data not already resident in graphics memory. Inline FIFO transfer is efficient only for moderate blocks; large transfers should use the board/driver's bulk path where available.

Host reads are more expensive because they contend with scanout and force synchronization. Avoid frequent small readbacks in interactive drawing paths.

8.9 Bandwidth Calculation

Minimum scanout bandwidth is approximately pixel clock multiplied by stored bytes per pixel, before memory protocol overhead. Rendering traffic adds source reads, destination reads and writes depending on command/ROP.

For example, a 110 MHz XRGB8888 display consumes roughly 440 MB/s of raw pixel fetch bandwidth. Whether a specific VDP-7 board qualifies such a mode depends on memory organization and RAMDAC; use AN-VDP7-05 and board documentation rather than this arithmetic alone.

8.10 Display and Drawing Contention

The scheduler protects active scanout from ordinary rendering traffic, but heavy drawing can experience variable latency as resolution/pixel clock increase. Performance counters can quantify arbitration stalls.

Software requiring deterministic update completion should use completion markers and measure worst-case behavior under the target display mode, not under a reduced service resolution.

8.11 Page Flipping

Double buffering uses two framebuffer surfaces with compatible pitch/format. Render into the non-visible surface, wait for vertical blank and update START_ADDR to the completed surface.

The address write should be ordered so it takes effect at a defined blank boundary. Avoid changing format or pitch as part of a normal page flip; those changes require broader timing/resource validation.

8.12 Off-Screen Allocation

Off-screen memory can hold backing stores, icons, patterns, temporary blit surfaces and software-managed caches. VDP-7 does not allocate this space. The driver must prevent scanout, cursor and command surfaces from overlapping incorrectly.

Fragmentation is usually handled by the operating-system driver or graphics library; embedded systems can use a fixed memory map established at initialization.

8.13 Bounds Protection

With bounds checking enabled, calculated graphics accesses outside FB_BASE/FB_LIMIT are blocked and reported. Bounds are coarse device limits, not per-surface limits, so an erroneous command can still overwrite a different valid surface inside the global range.

Drivers should validate surface rectangles in software even when hardware bounds protection is enabled.



8.14 Framebuffer Programming Checklist

For each mode, record pixel format, visible size, pitch, surface base, surface extent, cursor allocation and off-screen allocation. Confirm alignment and total memory use against the board's installed capacity.

Qualification tests should include edge rectangles, clipped negative coordinates, overlapping blits, full-screen fills, page flips and CPU readback. Run tests at the highest intended display bandwidth because memory-scheduler margins are smallest there.

Chapter 9 - Display Timing and Synchronization

9.1 Timing Model

Programmable Display Timing

Front Porch	H Sync	Back Porch	Active Video
-------------	--------	------------	--------------

Front	V Sync	Back Porch	Active Lines
-------	--------	------------	--------------

Register fields store total, sync, blanking and active-region boundaries in pixel clocks or scan lines.

Display timing is defined by horizontal/vertical total, active, sync and blanking boundaries plus board-level pixel clock selection. Register values are counter endpoints rather than monitor model names.

The VDP-7 can represent many timings that are not qualified by the complete board. Always begin from a supported mode table or AN-VDP7-05 and modify only within known limits.

9.2 Horizontal Timing

H_TOTAL contains active-pixel count and total clocks per line. H_SYNC and H_BLANK define intervals within that total. Boundaries must be monotonic in the counter domain used by the selected mode.

Programming a sync end beyond total or an active count incompatible with blanking can set DISPLAY_TIMING or produce undefined external sync on early revisions.

9.3 Vertical Timing

V_TOTAL contains active and total scan-line counts. V_SYNC and V_BLANK define line intervals. Vertical blank is also the synchronization interval used by page-flip and palette-update software.

A vertical-blank interrupt is generated once per frame at the qualified transition defined by the timing engine, not continuously for every line in blank.



9.4 Sync Polarity

HSYNC_POL and VSYNC_POL select logical polarity at the VDP-7 output before any board-level inversion. Board documentation determines the actual connector polarity.

Do not infer monitor mode from polarity alone; many technical CRTs accept several combinations. Use the complete timing specification.

9.5 Pixel Clock Selection

CLOCK_CTRL selects among board-qualified sources and optional division. The numeric pixel frequency is board-specific because oscillators/PLL/RAMDAC are external to the core.

Drivers should represent modes with an abstract clock selector or board timing structure rather than hard-coding a frequency-to-register formula into generic VDP-7 code.

9.6 START_ADDR and Panning

START_ADDR selects the first scanout location. Combined with pitch larger than visible width, address changes can implement page flipping and coarse panning.

Fine horizontal panning below the address granularity is not part of the base VDP-7 programming model and should be handled by drawing into an appropriately aligned surface.

9.7 Blanking and Forced Blank

BLANK_FORCE allows software to suppress visible output while preserving timing. This is useful while changing palette entries, diagnosing a monitor or performing operations that should not be exposed onscreen.

Forced blank is not equivalent to display disable: timing counters and vertical-blank events continue. This distinction is useful for mode transitions.

9.8 Vertical Blank Synchronization

The VBLANK status bit reflects current scan state; the interrupt/event marks a transition. Software that needs exactly one action per frame should use the event or WAIT_VBLANK rather than repeatedly polling the level without edge tracking.

Page flips should update START_ADDR during blank with sufficient margin before active fetch begins.

9.9 Line Compare

LINE_COMPARE can generate a diagnostic/interrupt event at a selected scan line where supported by the driver. It is intended for measurement, split-screen service tests and synchronization experiments, not for general application raster effects.

Set 0xFFFF to disable. Software must account for interlace behavior when using line compare on a qualified interlaced mode.



9.10 Qualified Example Timing

Parameter	1024 x 768 example	1280 x 1024 example
Active	1024 x 768	1280 x 1024
Horizontal total	1344	1688
H sync	1048-1184	1328-1440
Vertical total	806	1066
V sync	771-777	1025-1028
Notes	Representative engineering values; board clock qualification required.	Representative engineering values; board clock qualification required.

These values illustrate register relationships and are not a substitute for the mode table supplied with a particular VDP-7 board or system. The board RAMDAC and memory subsystem determine whether the corresponding pixel clock is supported.

9.11 Timing Update Procedure

To change modes safely: force blank, wait for blank/idle as required, disable display scanout, program clock selection through the qualified board interface, write the complete timing set, update framebuffer format/pitch/start, clear stale timing error state, enable sync/scanout, then release forced blank after stable operation.

If the host cannot verify clock lock directly, use the board-specific delay specified by the system manual rather than an arbitrary short loop.

9.12 High-Resolution Considerations

Higher pixel clocks reduce memory bandwidth available to rendering and increase sensitivity to board signal integrity. Software symptoms can include slower blits or underflow-like display faults even when command logic is correct.

Use performance counters and board diagnostics to distinguish memory contention from command parser or host interface errors. AN-VDP7-05 contains the electrical/design side of this analysis.

9.13 Timing Fault Recovery

On DISPLAY_TIMING, force blank if possible, disable scanout, capture timing registers and revision, restore a known-good mode, clear the error and verify vertical-blank activity before re-enabling application rendering.

Do not repeatedly reapply the failing custom timing in an automatic retry loop; a monitor or board interface can be stressed by unstable sync.

9.14 Display Timing Checklist

For every qualified mode, preserve a complete structure containing pixel format, pitch, clock selector, horizontal total/active/sync/blank, vertical total/active/sync/blank, polarity and start-address alignment. Treat that structure as one version-controlled object.

Mode validation should include cold boot, warm restart, repeated mode switching, blank/unblank, cursor movement, full-screen drawing load and page flipping. A mode that displays a static test pattern is not necessarily qualified for sustained graphics traffic.

Chapter 10 - Cursor, Palette and Color

10.1 Hardware Cursor Overview

The VDP-7 cursor is a scanout overlay independent of framebuffer drawing. It supports a maximum 64 x 64 image with transparent states and two programmable colors. Cursor movement therefore requires only `CURSOR_POS` updates.

Operating-system drivers should use the hardware cursor when the requested shape fits the supported encoding; larger or full-color cursors require software composition.

10.2 Cursor Image Format

The standard cursor encoding uses two bits per cursor pixel, selecting transparent, color 0, color 1 or invert/alternate behavior as defined by `CURSOR_CTRL` format. Rows are stored in fixed-length aligned blocks.

`CURSOR_BASE` must satisfy 256-byte alignment. Update the image while cursor is disabled or during vertical blank when the active image memory is being rewritten.

10.3 Cursor Position and Hotspot

CURSOR_POS contains signed screen coordinates. Negative values permit a cursor to be partially visible at top/left. Optional hotspot offsets allow the logical pointer location to differ from the image origin.

Position writes are designed for asynchronous use and do not require command-engine idle. Still, host ordering must ensure image/color changes reach the device before enabling a newly defined cursor.



10.4 Cursor Colors

CURSOR_COLOR0 and CURSOR_COLOR1 hold direct color values interpreted by the scanout path. In indexed framebuffer modes the cursor can still use programmed direct colors where supported by the board RAMDAC path, allowing visibility independent of application palette.

System board documentation identifies any RAMDAC limitations.

10.5 LUT Programming

LUT_ADDR selects an entry and LUT_DATA supplies 8-bit red, green and blue values. Consecutive writes may auto-increment the address on supported board implementations.

For visible indexed-color modes, large palette changes should be synchronized to vertical blank or performed under forced blank to avoid partial-frame palette transitions.

10.6 Color Interpretation

FG_COLOR, BG_COLOR and cursor colors are logical values. In RGB565 only the low 16 relevant bits are stored for drawing. In XRGB8888, color components map to the packed pixel representation defined by FB_FORMAT. In indexed mode, drawing colors are palette indices.

Graphics Library 4.1 handles these conversions; direct driver code must not reuse a 32-bit RGB constant across formats without conversion.

10.7 Palette Animation and Diagnostics

Palette animation can update selected LUT entries without redrawing the framebuffer, but it is primarily intended for technical visualization and diagnostic applications. Excessive unsynchronized LUT traffic can produce visible color tearing.

Diagnostic software may load a grayscale or color-ramp table to verify RAMDAC path integrity independently from drawing operations.

10.8 Cursor and Color Checklist

Verify cursor base alignment, image encoding, hotspot/negative-coordinate handling, palette consistency, blanking around visible LUT changes and format-aware color packing.

When a cursor appears corrupted while framebuffer graphics remain correct, test the cursor memory allocation and RAMDAC overlay path before modifying line/blit code. When only colors are wrong, compare FB_FORMAT, LUT contents and board DAC mode as separate layers.



Chapter 11 - Interrupts and Diagnostics

11.1 Interrupt Service Model

INT_STATUS latches device events. The ISR reads status, masks with enabled/owned sources, services them and writes one to clear handled edge events. Host-platform interrupt acknowledge is separate.

Shared interrupts require the driver to verify that at least one VDP-7 source is active before claiming the interrupt.

11.2 CMD_DONE and FIFO_LOW

CMD_DONE is associated with ordered completion markers, while FIFO_LOW indicates the configured free-word condition. They serve different purposes: completion synchronizes rendering; FIFO_LOW supports producer flow control.

Do not use FIFO_LOW as a rendering fence. The engine may still be busy after the FIFO drains.

11.3 VBLANK Interrupt

VBLANK is useful for page flip, cursor/palette updates and frame accounting. Service should be short; defer expensive application work to a lower-priority context.

If the display is disabled, no normal VBLANK cadence exists. Drivers should cancel or complete pending frame waits during shutdown/mode change.

11.4 Error Interrupt

ERROR indicates one or more ERROR_STATUS bits. Capture ERROR_STATUS before clearing and preserve the failing command batch identifier if available.

Parser and framebuffer faults normally require command-stream recovery; display timing faults require scanout recovery. The error bit alone is insufficient to choose the reset domain.



11.5 Performance Counters

PERF_COUNT0/1 can count selected engine or memory events while DIAG_CTRL enables them. Use counters for comparative measurements under controlled workloads, not as absolute timing sources.

Counters wrap naturally. Read them before and after a bounded test and calculate unsigned differences.



11.6 Signature Testing

SIGNATURE accumulates a diagnostic value for selected internal activity. Sinneslöschen service procedures use known command/data patterns and compare the resulting signature against revision-specific references.

Application software should not implement its own pass/fail criteria from SIGNATURE because authorized patterns and acceptable values can vary by revision.

11.7 Field Diagnostic Capture

Item	Capture
Identification	DEVICE_ID, REVISION, board assembly revision
Host	Bridge type/revision, mapped base, ordering mode
Framebuffer	Base, limit, pitch, format, installed memory
Display	Clock selector, totals, sync, blanking, start address
Command	FIFO status, last software batch/tag, engine idle
Error	INT_STATUS, ERROR_STATUS, TEST_STATUS
Environment	Driver/library version and failing application sequence

A useful field report preserves raw numeric values before a reset. Interpreted descriptions should be included in addition to, not instead of, the raw state.

11.8 Diagnostic Checklist

Separate failures by layer: host access, command parsing, framebuffer access, drawing correctness, scanout/timing, color/cursor and software synchronization. Prove the lowest failing layer with a minimal test before reintroducing higher graphics software.

The VDP Development Toolkit 3.2 supplies tests for this progression. When Toolkit and raw register tests pass but an application fails, collect the command trace and Graphics Library version rather than replacing hardware reflexively.



Chapter 12 - VDP-6 Compatibility

12.1 Compatibility Scope

VDP-7 provides source-level and command-level migration aids for software developed around VDP-6 concepts, but it is not register-identical. Drivers must use the VDP-7 register map and should not map VDP-6 register offsets onto VDP-7 hardware.

Vector Compatibility Library 2.8 and Graphics Library 4.1 provide the preferred application migration path.



12.2 Preserved Concepts

Line, rectangle, blit, clipping, indexed-color and display-timing concepts remain recognizable. Basic ROP semantics and integer coordinate behavior can often be mapped directly.

This continuity allows established technical applications to migrate without redesigning their graphics model, while the driver/library translates hardware details.



12.3 Changed Areas

VDP-7 expands command FIFO behavior, pixel formats, scaling, polygon fill, cursor capability, interrupt status and timing flexibility. VDP-6 assumptions about FIFO depth, register reset state and endpoint tie behavior must be removed from direct hardware code.

AN-VDP7-06 identifies source-level migration patterns for common VDP-6 applications.

12.4 Line Rendering Differences

Subpixel coordinate support and updated tie rules can change a small number of pixels at shared endpoints or exact diagonal boundaries compared with VDP-6. Compatibility mode in LINE_CTRL reduces differences for validated legacy applications.

New development should use VDP-7 default rules rather than force compatibility unless pixel-identical output is a requirement.



12.5 Timing Migration

VDP-6 mode tables cannot be copied blindly because VDP-7 timing registers encode boundaries differently and may operate with a different RAMDAC/clock subsystem. Convert the complete mode through the VDP-7 driver and requalify against the target board/monitor.

Application code should request a named/validated mode rather than program timing directly.

12.6 Migration Checklist

Inventory VDP-6 direct register accesses, replace them with driver/library calls where possible, convert command packet assumptions, verify pixel format/color handling, compare line/polygon edge cases, requalify display modes and collect performance baselines.

Retain the previous VDP-6 executable during qualification. A migration is complete only when rendering correctness, timing stability and application workflow have been tested under representative load.

Appendix A - Register Summary

Appendix A.1 - Register Summary

Offset	Register	Access	Reset
0x000	DEVICE_ID	RO	0x56445037
0x004	REVISION	RO	varies
0x008	RESET	R/W	0x00000000
0x00C	STATUS	RO	0x00000003
0x010	INT_STATUS	W1C	0x00000000
0x014	INT_ENABLE	R/W	0x00000000
0x018	INT_ACK	WO	0x00000000
0x01C	ERROR_STATUS	W1C	0x00000000
0x020	HOST_CTRL	R/W	0x00000000
0x024	HOST_CONFIG	R/W	0x00000000
0x028	FIFO_STATUS	RO	0x00400000
0x02C	FIFO_WATERMARK	R/W	0x00000010
0x030	CMD_PORT	WO	undefined
0x034	CMD_BASE	R/W	0x00000000
0x038	CMD_LIMIT	R/W	0x00000000
0x03C	CMD_PTR	R/W	0x00000000
0x040	FB_BASE	R/W	0x00000000
0x044	FB_PITCH	R/W	0x00000000
0x048	FB_FORMAT	R/W	0x00000000
0x04C	FB_LIMIT	R/W	0x00000000
0x050	SCRATCH_BASE	R/W	0x00000000
0x054	SCRATCH_LIMIT	R/W	0x00000000
0x058	CLIP_LEFT_TOP	R/W	0x00000000

Registers not listed are reserved. Production software shall not probe reserved offsets. Use the detailed Chapter 5 pages for field definitions and ordering requirements.

Appendix A.2 - Register Summary

Offset	Register	Access	Reset
0x05C	CLIP_RIGHT_BOTTOM	R/W	0x00000000
0x060	ORIGIN	R/W	0x00000000
0x064	SCALE_X	R/W	0x00010000
0x068	SCALE_Y	R/W	0x00010000
0x06C	LINE_CTRL	R/W	0x00000000
0x070	FG_COLOR	R/W	0xFFFFFFFF
0x074	BG_COLOR	R/W	0x00000000
0x078	ROP	R/W	0x0000000C
0x07C	PATTERN	R/W	0xFFFFFFFF
0x080	BLT_SRC_BASE	R/W	0x00000000
0x084	BLT_DST_BASE	R/W	0x00000000
0x088	BLT_SRC_XY	R/W	0x00000000
0x08C	BLT_DST_XY	R/W	0x00000000
0x090	BLT_SIZE	R/W	0x00000000
0x094	BLT_CTRL	R/W	0x00000000
0x098	POLY_CTRL	R/W	0x00000000
0x09C	EDGE0	R/W	0x00000000
0x0A0	EDGE1	R/W	0x00000000
0x0A4	VERTEX_PORT	WO	undefined
0x0A8	VERTEX_COUNT	R/W	0x00000000
0x0AC	CURSOR_CTRL	R/W	0x00000000
0x0B0	CURSOR_POS	R/W	0x00000000
0x0B4	CURSOR_BASE	R/W	0x00000000

Registers not listed are reserved. Production software shall not probe reserved offsets. Use the detailed Chapter 5 pages for field definitions and ordering requirements.

Appendix A.3 - Register Summary

Offset	Register	Access	Reset
0x0B8	CURSOR_COLOR0	R/W	0x00FFFFFF
0x0BC	CURSOR_COLOR1	R/W	0x00000000
0x0C0	DISP_CTRL	R/W	0x00000000
0x0C4	H_TOTAL	R/W	0x00000000
0x0C8	H_SYNC	R/W	0x00000000
0x0CC	H_BLANK	R/W	0x00000000
0x0D0	V_TOTAL	R/W	0x00000000
0x0D4	V_SYNC	R/W	0x00000000
0x0D8	V_BLANK	R/W	0x00000000
0x0DC	START_ADDR	R/W	0x00000000
0x0E0	LINE_COMPARE	R/W	0x0000FFFF
0x0E4	CLOCK_CTRL	R/W	0x00000000
0x0E8	DAC_CTRL	R/W	0x00000000
0x0EC	LUT_ADDR	R/W	0x00000000
0x0F0	LUT_DATA	R/W	0x00000000
0x0F4	GPIO_CTRL	R/W	0x00000000
0x0F8	GPIO_DATA	R/W	0x00000000
0x0FC	DIAG_CTRL	R/W	0x00000000
0x100	PERF_COUNT0	RO	0x00000000
0x104	PERF_COUNT1	RO	0x00000000
0x108	SIGNATURE	RO	0x00000000
0x10C	TEST_STATUS	RO	revision dependent

Registers not listed are reserved. Production software shall not probe reserved offsets. Use the detailed Chapter 5 pages for field definitions and ordering requirements.

Appendix B - Command Summary

Appendix B.1 - Command Summary (1 of 2)

Opcode	Command	Words	Purpose
0x00	NOP	1	No operation; used for alignment and diagnostic submission tests.
0x01	SET_STATE	3	Update a selected drawing-state register through the ordered stream.
0x02	SET_CLIP	3	Set active clip rectangle.
0x03	SET_ORIGIN	2	Set signed drawing origin.
0x04	SET_COLOR	3	Set foreground and background colors.
0x10	LINE	5	Draw one line segment using current color/ROP/clip state.
0x11	POLYLINE	variable	Draw connected line segments.
0x12	POINTS	variable	Draw independent single-pixel points.
0x13	RECT_OUTLINE	3	Draw axis-aligned rectangle outline.
0x14	RECT_FILL	3	Fill axis-aligned rectangle.
0x15	TRIANGLE_FILL	7	Fill one triangle.
0x16	POLY_FILL	variable	Fill a convex polygon.

Opcode values not listed are reserved. Reserved opcodes may set CMD_FORMAT and require command-engine recovery.

Appendix B.2 - Command Summary (2 of 2)

Opcode	Command	Words	Purpose
0x20	BLIT	5	Copy a rectangle between framebuffer surfaces.
0x21	STRETCH_BLIT	7	Scale a source rectangle into destination rectangle.
0x22	HOST_TO_FB	variable	Transfer host-supplied pixel words into framebuffer.
0x23	FB_TO_HOST	4	Prepare framebuffer readback transfer.
0x24	PATTERN_FILL	4	Fill rectangle using repeating PATTERN bits and colors.
0x25	MONO_EXPAND	variable	Expand 1-bpp source data to foreground/background colors.
0x26	CLEAR_SURFACE	4	Clear defined surface region with constant color.
0x30	SET_CURSOR	3	Update cursor position/control in ordered stream.
0x31	LOAD_LUT	variable	Load consecutive palette/LUT entries.
0x40	WAIT_VBLANK	1	Wait for next vertical blank boundary.
0x41	WAIT_IDLE	1	Wait for drawing engines to become idle.
0x42	SYNC_MARKER	2	Generate ordered completion event/tag.

Variable-length packets must satisfy the minimum/maximum vertex or data counts documented in Chapter 7. Packet length always includes the header word.

Appendix C - Error Quick Reference

Appendix C - Error and Status Quick Reference

Condition	Primary register/bit	Recommended first action
Command parser fault	ERROR_STATUS.CMD_FORMAT	Stop submissions; capture trace; reset command engine.
FIFO overflow	ERROR_STATUS.FIFO_OVERFLOW	Audit FIFO accounting and host posting behavior.
Framebuffer bounds	ERROR_STATUS.FB_BOUNDS	Validate base/limit, pitch, format and command rectangle.
Display timing	ERROR_STATUS.DISPLAY_TIMING	Force blank/disable scanout; restore known-good timing.
Memory timeout	ERROR_STATUS.MEMORY_TIMEOUT	Check framebuffer memory/board diagnostics and traffic.
No completion	STATUS/FIFO/INT_STATUS	Verify host barrier, command acceptance and engine idle.
No vertical blank	STATUS.DISP_ACTIVE/VBLANK	Verify display enabled, timing valid and clock active.
Wrong colors	FB_FORMAT/LUT/DAC_CTRL	Separate stored pixel format from RAMDAC palette/output.

Always preserve raw state before reset when a repeatable field fault is under investigation.

Appendix D - Initialization Skeleton

Appendix D - Complete Initialization Skeleton

```
int vdp7_initialize(struct vdp7 *v, const struct mode *m)
{
    if (read32(DEVICE_ID) != 0x56445037) return -1;
    v->revision = read32(REVISION);

    write32(INT_ENABLE, 0);
    host_barrier();
    write32(RESET, RESET_CMD | RESET_DIAG);
    if (!wait_bits(STATUS, STATUS_CMD_IDLE, 100)) return -2;
    write32(INT_STATUS, 0xffffffff);
    write32(ERROR_STATUS, 0xffffffff);

    write32(FB_BASE, v->fb_base);
    write32(FB_LIMIT, v->fb_limit);
    write32(FB_PITCH, m->pitch);
    write32(FB_FORMAT, m->format);
    write32(HOST_CTRL, HOST_CMD_ENABLE | HOST_BOUNDS_ENABLE);

    write32(CLIP_LEFT_TOP, 0);
    write32(CLIP_RIGHT_BOTTOM, PACK_XY(m->width, m->height));
    write32(ORIGIN, 0);
    write32(SCALE_X, 0x00010000);
    write32(SCALE_Y, 0x00010000);
    write32(ROP, ROP_COPY);

    if (program_display_mode(m) != 0) return -3;
    write32(INT_ENABLE, INT_CMD_DONE | INT_VBLANK | INT_ERROR);
    return run_basic_render_test();
}
```

This skeleton illustrates ordering only. Board clock/RAMDAC programming, operating-system resource management, bridge setup, timeout implementation and cleanup paths remain platform responsibilities.

Appendix E - Glossary

Appendix E - Glossary and Support References

Term	Meaning
blit	Block transfer between framebuffer regions or between host and framebuffer.
clip window	Rectangle limiting generated destination pixels.
command FIFO	64-word buffer between host command writes and command decoder.
completion marker	Ordered command used to establish rendering completion.
framebuffer	External graphics memory containing visible and off-screen surfaces.
pitch	Byte distance between starts of adjacent surface rows.
ROP	Logical raster operation combining source/pattern/destination data.
scanout	Display fetch process that reads framebuffer pixels according to timing state.
surface	Software-defined rectangular image region in framebuffer memory.
vertical blank	Frame interval outside active displayed lines, used for synchronized changes.

Support References

Technical Support: 1-800-555-0142 / 408-555-0186

Documentation Fax: 408-555-0163

Software Downloads: VDP Development Toolkit 3.2 and Graphics Library 4.1 through the Sinneslöschen Software Download Service.

Document Identification

RM-VDP7-03 Revision C - May 19, 1998. Preserve this document number and revision when citing register or command behavior in OEM qualification records.