CL-GD5446 Extended Display Modes Summary

Mode No.	VESA® No.	Colors	Display Resolution	Chars.	Refresh (Hz)
58, 6A	102	16/256K	800 × 600	100 × 37	56, 60, 72, 75
5C	103	256/256K	800 × 600	100 × 37	56, 60, 72, 75, 85
5D	104	16/256K	1024 × 768	128 × 48	43i[a], 60, 70, 72, 75
5E	100	256/256K	640 × 400	80 × 25	70
5F	101	256/256K	640 × 480	80 × 30	60, 72, 75, 85
60	105	256/256K	1024 × 768	128 × 48	43i, 60, 70, 72, 75, 85
64	111	64K	640 × 480	–	60, 72, 75, 85
65	114	64K	800 × 600	–	56, 60, 72, 75, 85
66	110	32K	640 × 480	–	60, 72, 75, 85
67	113	32K	800 × 600	–	56, 60, 72, 75, 85
68	116	32K	1024 × 768	–	43i, 60, 70, 75, 85
69	119	32K	1280 × 1024	–	43i, 60
6C	106	16/256K	1280 × 1024	160 × 64	43i
6D	107	256/256K	1280 × 1024	160 × 64	43i, 60, 75
71	112	16M	640 × 480	–	60, 72, 75, 85
74	117	64K	1024 × 768	–	43i, 60, 70, 75, 85
75	11A	64K	1280 × 1024	–	43i
78	115	16M	800 × 600	–	56, 60, 72, 75, 85
79	118	16M	1024 × 768	–	43i, 60, 70, 75, 85
7B	–	256/256K	1600 × 1200	–	48i
7C	–	256/256K	1152 × 864	–	70, 75

a. ‘i’ indicates interlaced.

Refer to Table 3-2, “Cirrus Logic Extended Display Modes,” on page 3-25.

CL-GD5446

Technical Reference Manual

November 1996

Second Edition

CIRRUS LOGIC®

Notice

Cirrus Logic Inc. has made best efforts to ensure that the information contained in this document is accurate and reliable. However, the information is subject to change without notice. No responsibility is assumed by Cirrus Logic Inc. for the use of this information, nor for infringements of patents or other rights of third parties. Cirrus, Cirrus Logic, AccuPak, DIVA, FastPath, FasText, FeatureChips, Good Data, Laguna, Laguna3D, MediaDAC, MotionVideo, SimulSCAN, S/LA, SMASH, SofTarget, TextureJet, TVTap, UXART, VisualMedia, VPM, V-Port, Voyager, WavePort, and WebSet are trademarks of Cirrus Logic Inc., which may be registered in some jurisdictions. Other trademarks in this document belong to their respective companies. CRUS and Cirrus Logic International, Ltd. are trade names of Cirrus Logic Inc.

Copyright Notice

Contents

Revision History

The following are the differences between the December 1995 and November 1996 versions of this technical reference manual:

- Information pertaining to the Revision B device has been added

Table of Contents

APPENDIXES

INDEXES

Introduction

1. INTRODUCTION

1.1 Scope of Document

This manual provides a technical discussion of the CL-GD5446 VisualMedia™ accelerator. This manual includes descriptions of each major component integrated into the device, a data book, detailed information on each register, a BIOS description, and appendices intended to assist hardware and software designers.

1.2 Chip Types Covered

This manual documents the CL-GD5446. Table 1-1 shows the production versions covered.

Table 1-1. Production Versions Covered

Revision	A	B
CL-GD5446	✔	✔

The CL-GD5446 ID register — CR27, reads back a value specifying the CL-GD5446. Refer to Chapter 8, “Miscellaneous Extension Registers”, for further information. The ID is also in register PCI00.

1.3 Intended Audience

This manual is intended for a technically sophisticated audience. It is assumed that the reader is familiar with assembly language programming on the 8088/8086, 80286/80386/80486, Pentium®, or similar microprocessor, and understands the fundamentals of computer generated graphics display technology.

Hardware engineers should find Chapter 3, “Data Book”, useful. It contains the pinouts and pin summary. In addition, Chapter 10 contains the detailed pin descriptions and Chapter 11 contains the DC and AC specifications. Appendix B1 and Appendix B2 contain board design and layout information.

Software engineers should find Chapters 4–8 (register descriptions) useful for BIOS- and driver-level codes. All registers are described to the bit level. Chapter 9 contains programming notes.

1.4 Conventions

This section discusses conventions used throughout this document. Conventions include acronyms, abbreviations, and nomenclature usage. For a quick reference of acronyms see Table 1-2.

Bits

Bits are always listed in descending order, most-significant (highest number) to least-significant (lowest number). When discussing a bit field within a register or memory, the bit number of the most-significant bit is given on the left, followed by a colon (:) and then the bit number of the least-significant bit (for example, bits 7:0). A field consists of a set of adjoining bits with common functionality. Registers consist of fields of one or more bits.

Table 1-2. Acronym Quick Reference

Acronym	Definition
AC	alternating current
ALU	arithmetic logic unit
ATE	automatic test equipment
BIOS	basic input/output system
BitBLT, BLT	bit boundary block transfer
bpp	bits per pixel
CAD	computer-aided design
CAS	column address strobe
CGA	color graphics adapter
CLUT	color lookup table
CMOS	complementary metal-oxide semiconductor
CPU	central processing unit
CRT	cathode ray tube
CRTC	CRT controller
DAC	digital-to-analog converter
DC	direct current
DDA	digital differential algorithm
DDC	display data channel
DMI	desktop management signaling
DPMS	display power management signaling
DRAM	dynamic random access memory
dword	doubleword
EEPROM	electrically erasable/programmable read-only memory
EGA	enhanced graphics adapter
EPROM	electrically programmable read-only memory
EVAFC	extended VESA® advanced feature connector
FIFO	first in/first out
GPIO	general-purpose IO
GSC	graphics system controller
GUI	graphical user interface
HDR	Hidden DAC register
HRQ	host read queue
HSYNC/VSYNC	horizontal/vertical synchronization
HWQ	host write queue
IC	integrated circuit
I/O	input/output
LBI	local bus interface
LSB	least-significant bit

Acronym	Definition
LUT	lookup table
MA	memory arbiter
MC	memory controller
MCC	monochrome-to-color converter
MD	memory data
MMI/O	memory-mapped I/O
MSB	most-significant bit
OFU	operand fetch unit
OSU	operand storage unit
PCI	peripheral component interconnect
PFS	programmable frequency synthesizer
PLL	phase-locked loop
PQFP	plastic quad-flat pack
qword	two dwords
RAC	Rambus® access channel
RAM	random-access memory
RAS	row address strobe
RDRAM	Rambus® dynamic random-access memory
RGB	red, green, and blue
RIF	Rambus® interface
ROPs	raster operations
RSU	result storage unit
R/W	read/write
SC	serial clock
SG	signature generator
SGRAM	synchronous graphics RAM
SRAM	static random-access memory
TSR	terminate and stay resident
TTL	transistor-transistor logic
VBE	VESA BIOS extensions
VBI	vertical blanking interval
VDD	virtual device driver
VESA®	Video Electronics Standards Association
VGA	video graphics array
VL	VESA® local
VPM	video port manager
VRAM	video random-access memory
WE	transparency write enable

Acronyms

Throughout this manual, the first usage of all acronyms has the definition following in parentheses. Table 1-2 on page 1-3 lists most of the acronyms used in this manual. For further definitions, refer to Appendix E1, “Glossary and Bibliography”.

Abbreviations

The unit ‘Kbyte’ designates 1024 bytes. The unit ‘Mbyte’ designates 1,048,576 bytes (1024 squared). The unit ‘Gbyte’ designates 1,024 megabytes. The unit ‘Hz’ designates hertz. The unit ‘kHz’ designates 1,000 hertz. The unit ‘MHz’ designates 1,000 kilohertz. The unit ‘ns’ designates nanosecond. The unit ‘µs’ designates microsecond (1,000 nanoseconds). The unit ‘ms’ designates millisecond (1,000 microseconds). The unit ‘mA’ designates milliampere. The use of ‘tbd’ in tables indicates values that are ‘to be determined’. The unit ‘µF’ designates the capacitance measurement micro-farad (10^{-6} farad). N/A designates ‘not available’. The use of ‘n/c’ indicates the pin is a ‘no connect’.

Numeric Naming

Hexadecimal numbers are represented with all letters in upper case and a lower-case ‘h’ is appended to them (for example, ‘14h’, ‘3A7h’, and ‘C000h’ are hexadecimal numbers). Programming examples may use the C convention (prepend 0x to a hex number). Numbers not indicated by an ‘h’ are decimal. Octal numbers are not used in this manual.

Reserved

When a system memory or I/O address is referred to as ‘reserved’, it indicates that writing to that address is not permitted. Reserved bits *must* be written as ‘0’ to maintain upward compatibility.

Read-Only

The word ‘read-only’ is used to indicate registers and bits that can be read, but not written to.

Overview

2. OVERVIEW

The CL-GD5446 VisualMedia accelerator is a 64-bit DRAM based SVGA controller with hardware-accelerated BitBLT, video playback, and video capture to the frame buffer.

The CL-GD5446 combines the Cirrus Logic V-Port™ with a multi-format frame buffer for cost effective video playback. The V-Port captures real-time video into the frame buffer with optional data reduction. The video is typically displayed in the hardware video window with optional interpolated zooming. The video can be of a different format (for example, 16-bpp YUV 4:2:2) than the graphic format (for example, 8-bpp palettized VGA format).

The CL-GD5446 features a 64-bit GUI BitBLT engine with double-buffered, memory-mapped control registers. Transparency is supported with color expansion for all color depths, and supported without color expansion for 8- and 16-bpp graphics modes.

Highly integrated, the CL-GD5446 includes a programmable dual-frequency synthesizer and palette DAC, allowing a motherboard video playback solution with as few as three ICs plus the video decoder.

Production Revision B of the CL-GD5446 is specifically designed for compliance with PC97. The differences between Revision A and Revision B are detailed in Appendix A2, “Revision B Notes”.

2.1 Features

Table 2-1 presents a list of the major features of the CL-GD5446 VisualMedia accelerator.

Table 2-1. CL-GD5446 Features List

Features	CL-GD5446
GUI acceleration width (in bits)	64
Maximum dot clock	135 MHz
Maximum memory clock	80 MHz
Multimedia ready	✔
Integrated video playback support	✔
Video capture	✔
Video windowing	✔
Color key, chroma key occlusion support	✔
YUV and AccuPak™ video support	✔
Unique planar assist video support	✔
Multi-format frame buffer	✔
Color space conversion	✔
Interpolated zooming (independent for X and Y)	✔

Table 2-1. CL-GD5446 Features List *(cont.)*

Features	CL-GD5446
Transparent source BitBLT	✔
Active display line readback	✔
'Page flip' support	✔
I^2C support	✔
8- or 16-bit General-Purpose I/O bus	✔
DDC2B support	✔
'Green PC' power-saving features	✔
Direct PCI bus interface (2.1-compliant)	✔
VESA® pass-through feature connector	✔
Resolutions up to 1280 × 1024 (see inside front cover)	✔
Integrated triple 8-bit DAC	✔
Programmable dual-clock synthesizer	✔
64-bit DRAM display memory interface	✔
Memory size (Mbytes)	1, 2, 3, 4
4-, 16-bit-wide DRAMs	✔
EDO DRAM support	✔
128K × 16, 128K × 32 DRAM support	✔
16-bit Pixel bus	✔
CL-GD542X register- and software-compatible	✔
Low-power CMOS, 208-pin PQFP/HQFP package	✔
100% hardware- and BIOS-compatible with IBM® VGA display standards	✔
PC97 compliance	Revision B

2.2 Major Components

The CL-GD5446 incorporates all of the logic listed in Table 2-2 into a single integrated chip. These components are discussed in the following sections.

Table 2-2. CL-GD5446 Major Components

Logic Component	Section
VGA core: sequencer	2.3.1
VGA core: CRT controller	2.3.2
VGA core: graphics controller	2.3.3
VGA core: attribute controller	2.3.4
VGA core: programmable dual-frequency synthesizer	2.3.5
VGA core: palette DAC	2.3.6
PCI bus interface	2.4
BitBLT engine	2.5
Video capture	2.6
Video window and video display	2.7
General-purpose I/O	2.7.2
DDC2B interface	2.7.1

In describing the CL-GD5446, it is useful to retain the identity of the original major subsections found in the IBM EGA and VGA controllers. The architectures of these major subsections, as well as CL-GD5446 enhancements, are further described in the following sections.

NOTE: The diagrams in these sections are functional block diagrams of the components and are not intended to represent actual implementation.

2.3 VGA Core

2.3.1 Sequencer

The sequencer controls access to the display memory. It ensures that the necessary screen refresh and dynamic memory refresh cycles are executed, and that the remaining memory cycles are made available for CPU read/write operations, BitBLT read/write operations, and V-Port write operations.

The sequencer consists of a memory arbitrator and memory controller. It accepts requests from memory address counters associated with the CRTC, and address-transformation logic associated with the graphics controller. It uses the display FIFOs to deliver data to the display pipeline, and the write buffer to transfer data to the graphics controller. The Memory Sequencer registers are described in Chapter 4, “VGA Core Registers”.

The memory controller generates the signals and addresses necessary for accessing display memory. The memory controller is driven by a MCLK (memory clock) optimized for the speed of the DRAM used, independent of the VCLK (video clock). The memory controller can

generate optimized timing for EDO DRAMs and operate with an MCLK of up to 80 MHz. The memory arbitrator and host bus interface are also driven by the MCLK.

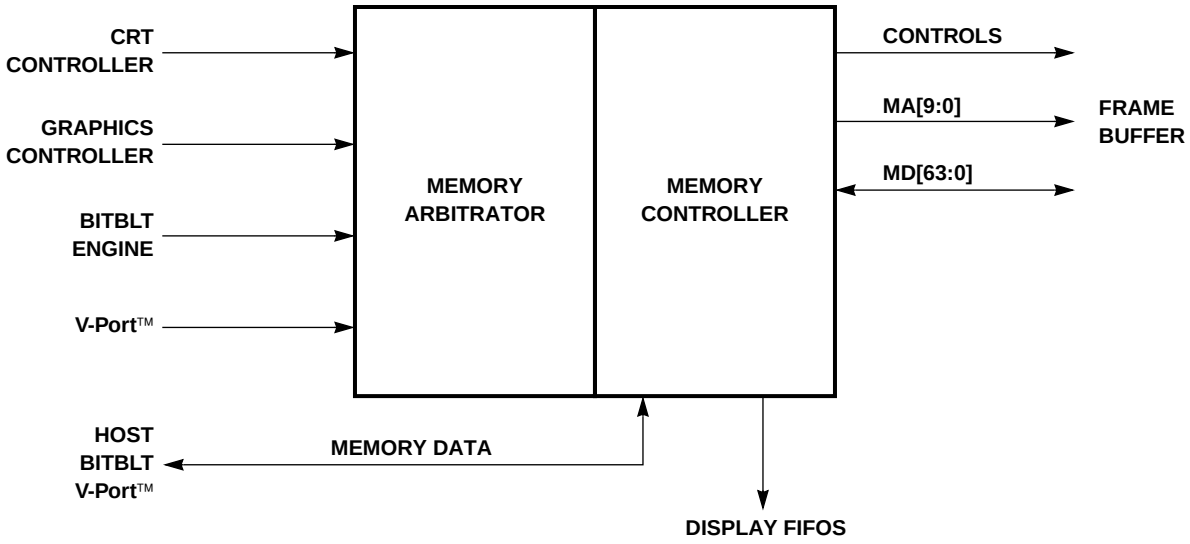


Figure 2-1. Sequencer Functional Block Diagram

2.3.2 CRT Controller

The CRTC (CRT controller) generates the horizontal and vertical synchronization signals for the CRT display. The CRTC allows configurable horizontal and vertical timing and polarity, cursor position, horizontal scanlines, and both horizontal and vertical GENLOCK. The CRTC registers are also described in Chapter 4.

The CRTC is software-compatible with IBM VGA hardware. The registers are expanded, as necessary, for high-resolution monitors. The CRTC also provides split-screen capability and smooth scrolling. A simplified functional diagram of the CRTC is shown in Figure 2-2.

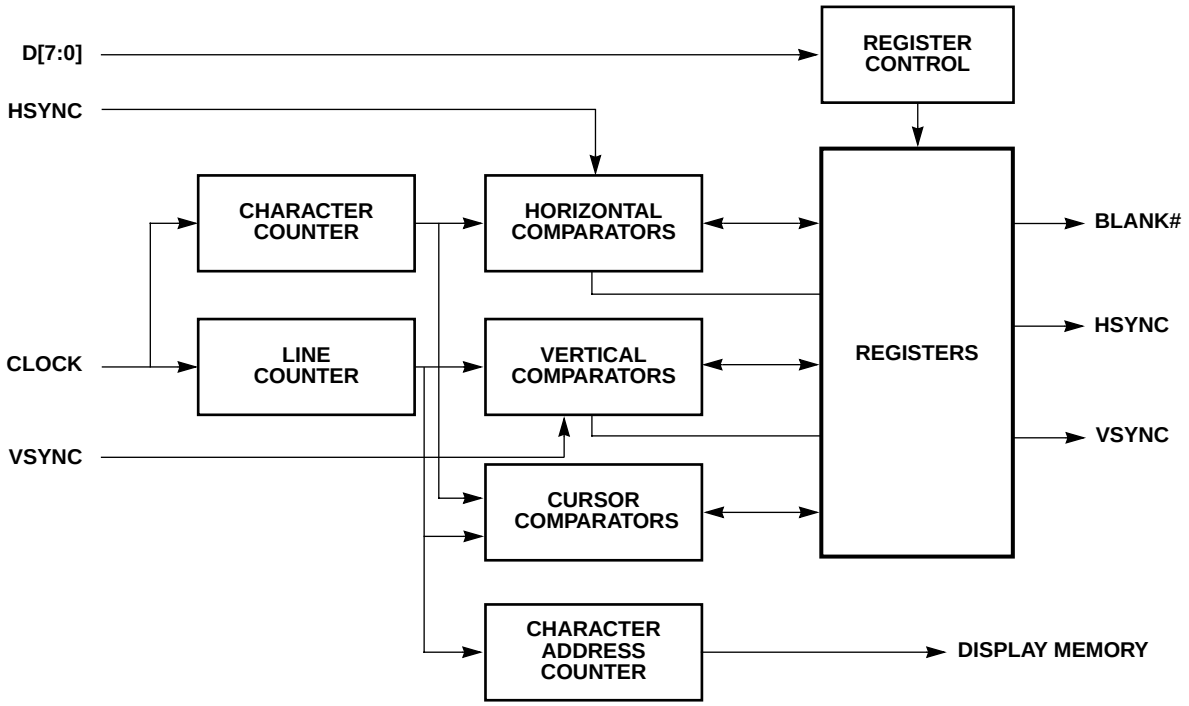


Figure 2-2. CRT Controller Functional Block Diagram

2.3.3 Graphics Controller

The graphics controller operates in either text or graphics modes and has the following major functions:

- Provides the host CPU with a read/write access path to display memory.
- Controls all four memory planes in planar modes (used for 16-color graphics).
- Allows data to be manipulated prior to being written to display memory.
- Formats data for use in various backward-compatibility modes.
- Provides color comparators for use in color painting modes.
- Reads/writes 32- or 64-bit words through the 32- or 64-bit display memory interface.
- Combines display memory data and attributes for output to the Pixel bus.

The graphics controller directs data from the display memory to the attribute controller and CPU. Figure 2-3 and Figure 2-4 illustrate typical write and read operations.

For a write operation, the data from the CPU bus are combined with the data from the Set/Reset logic, depending on the write and display modes. In addition, the data can be combined with the contents of the read latches, and some bits or planes may be masked (prevented from being changed). See the bit descriptions in Chapter 4, “VGA Core Registers”, for more information.

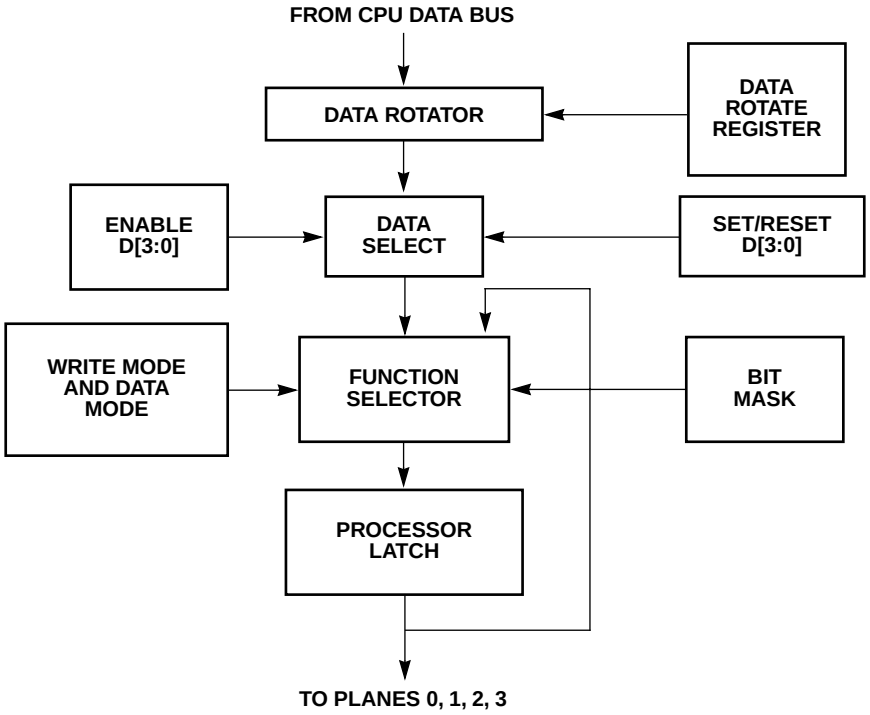


Figure 2-3. Graphics Controller Write Operation

The graphics controller is implemented when the CPU is reading data from display memory. Depending on the read mode, the data returned may be the actual contents of the display memory or reflect the outcome of comparisons with the color value in one of the Graphics Controller registers. See the descriptions in Chapter 4 for more information.

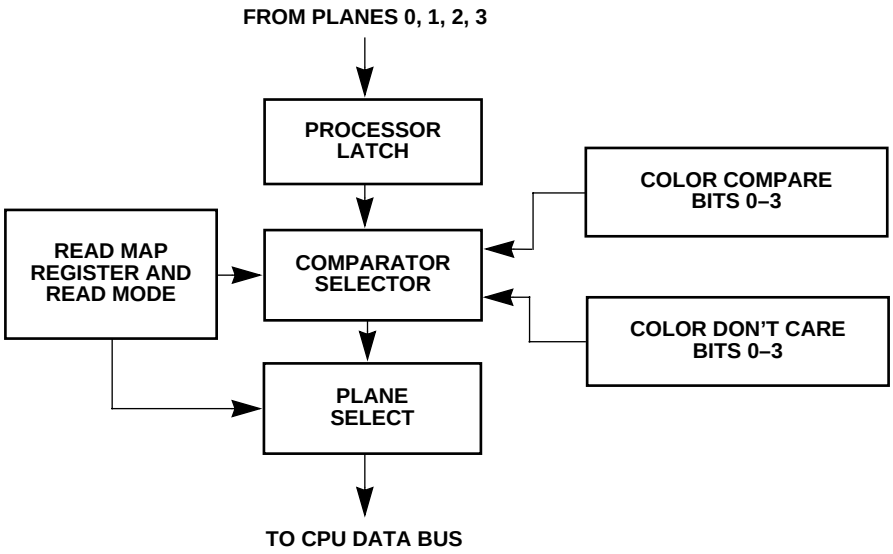


Figure 2-4. Graphics Controller Read Operation

2.3.4 Attribute Controller

The attribute controller controls the blinking and underline attributes in alphanumeric modes. It also provides horizontal pixel-panning capability in both alphanumeric and graphics modes. The attribute controller registers are described in Chapter 4. Figure 2-5 is a functional block diagram of the attribute controller.

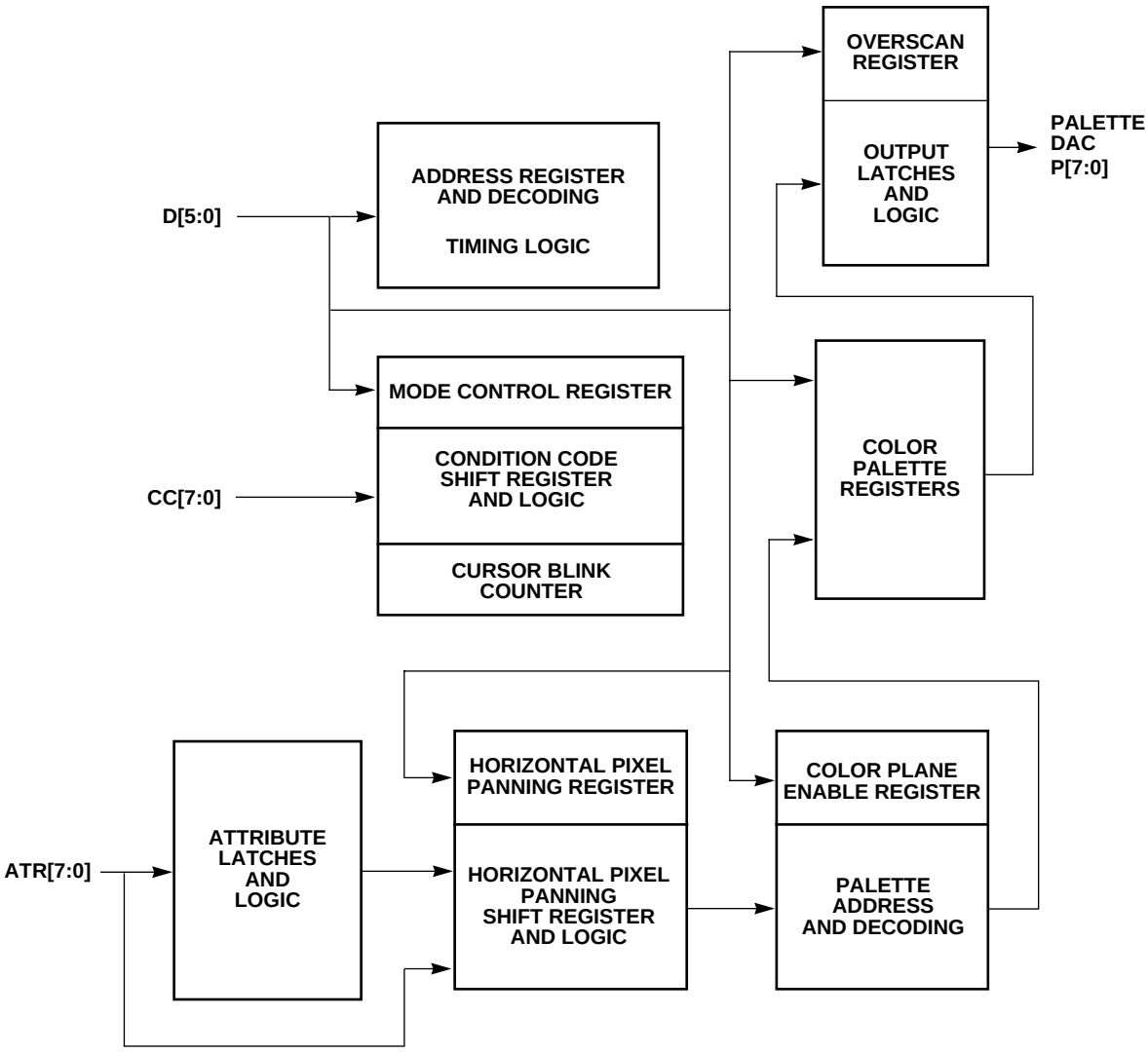


Figure 2-5. Attribute Controller

2.3.5 Programmable Dual-Frequency Synthesizer

The CL-GD5446 includes an integrated dual-frequency synthesizer that can be programmed to generate the VCLK for all supported screen formats, and the MCLK used by the sequencer. The VCLK synthesizer can support a pixel clock of up to 135 MHz. The MCLK synthesizer can be programmed for up to 80 MHz (for EDO DRAMS). The dual-frequency synthesizer includes an on-chip oscillator that requires an inexpensive, two-pin 14.31818-MHz crystal. Alternatively, the dual-frequency synthesizer can use a reference frequency of 14.31818 MHz from an external source. Figure 2-6 is a functional block diagram of the programmable dual-frequency synthesizer.

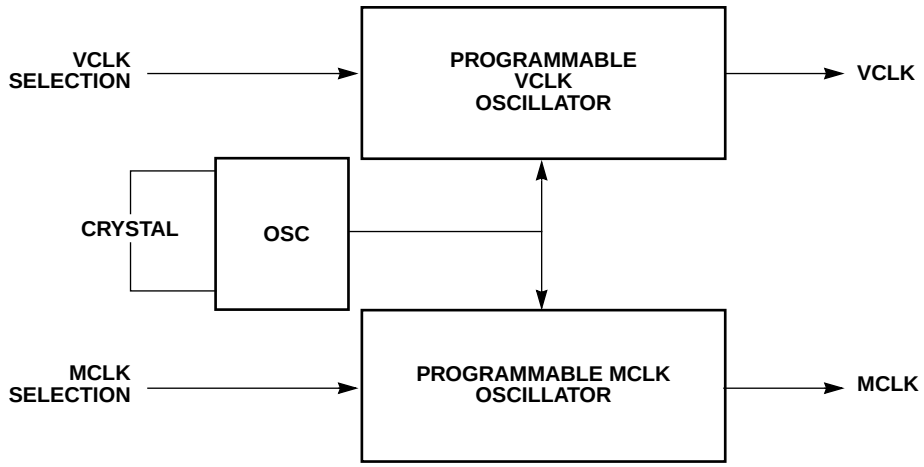


Figure 2-6. Programmable Dual-Frequency Synthesizer Functional Diagram

2.3.6 Palette DAC

The CL-GD5446 includes an integrated palette DAC that can interface to an analog monitor connector through the appropriate RFI filters. The palette DAC can be programmed for 256 simultaneous colors from a palette of 256K, or it can be programmed for Direct-color mode. In Direct-color mode, two, three, or four contiguous bytes from the display memory are combined for each pixel. This allows 32K, 64K, or 16.8 million simultaneous colors on the screen.

The CL-GD5446 supports color space conversion and can display YUV 4:2:2 or AccuPak video data as well as 8-bpp LUT and 16-bpp RGB in the hardware video window.

Figure 2-7 is a functional block diagram of the palette DAC.

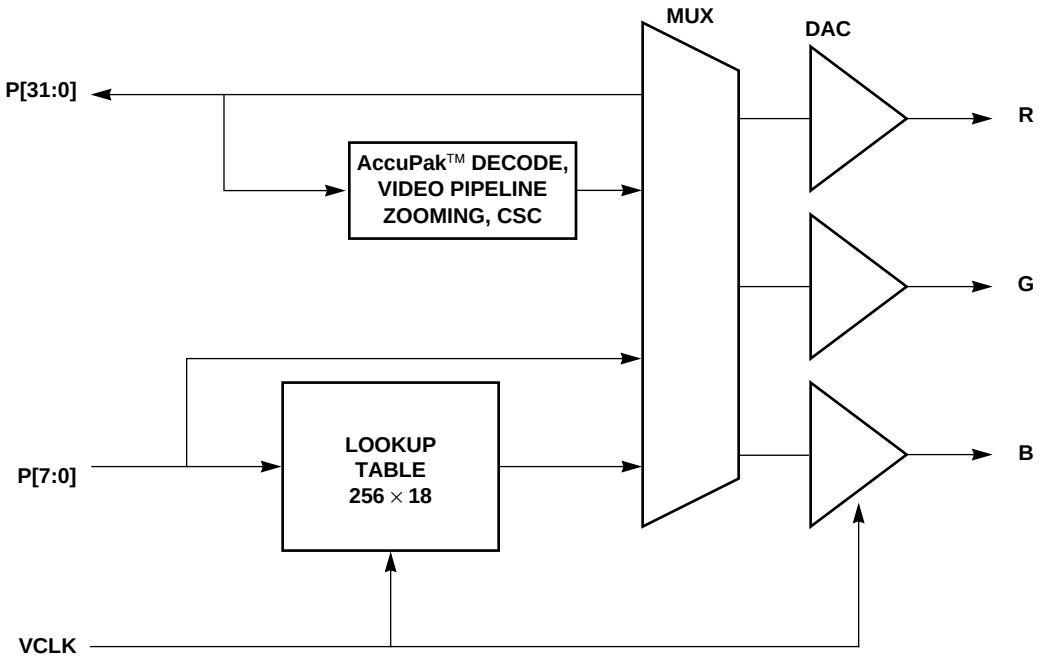


Figure 2-7. 256-Color/Direct-Color Palette DAC

2.4 PCI Bus Interface

The CL-GD5446 includes a glueless 32-bit PCI bus interface. This interface features full PCI compliance, including optimized PCI burst write, which supports PCI writes to the frame buffer at greater than 55 Mbytes per second.

The frame buffer is addressable through a 16-Mbyte window consisting of three 4-Mbyte byte-swapping apertures, and a special video aperture. The VGA control registers are relocateable anywhere in the 64-Kbyte space (allowing multiple devices in a single system).

The frame buffer in Revision B of the CL-GD5446 is addressable through two 16-Mbyte windows. One window is for direct accesses to the frame buffer; the second window is for system-to-screen BitBLTs. The VGA registers in Revision B of the CL-GD5446 are accessible anywhere in the memory address space. Revision B of the CL-GD5446 supports Subsystem Vendor ID in PCI2C.

2.5 BitBLT Engine

The CL-GD5446 includes a BitBLT engine for block transfers within display memory at full memory bandwidth. System-to-display transfers can also be effected with the BitBLT engine.

The CL-GD5446 BitBLT engine supports transparency with color expansion for all graphics formats and transparency without color expansion for 8- and 16-bpp graphics formats.

The BitBLT control registers are double-buffered and memory-mapped. Double-buffered registers, in conjunction with the autostart feature, allow concurrent operation of the host and the BitBLT engine. The host can prepare and load the parameters for operation n + 1 while the BitBLT engine is executing operation n. When the current operation completes, the BitBLT engine automatically loads and begins with the parameters for the next operation.

All 16 two-operand ROPs (raster operations) are implemented in hardware. Color expansion leverages host bandwidth by up to 32 times.

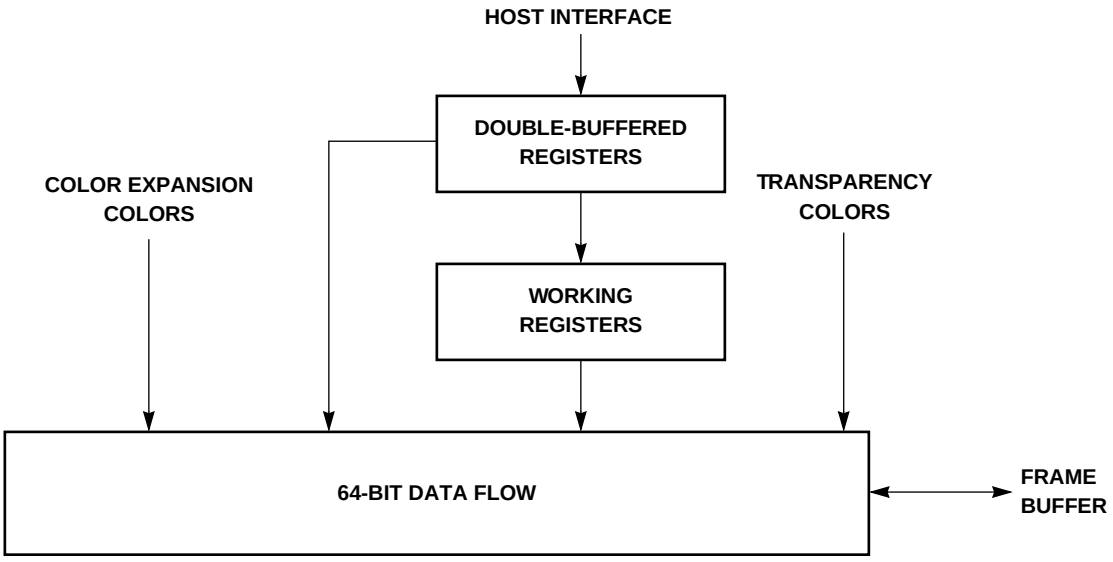


Figure 2-8. BitBLT Engine

2.6 Video Capture

The CL-GD5446 V-Port accepts video data from a realtime or recorded source and stores it into the frame buffer. V-Port can accept data in YUV 4:2:2, RGB16, or AccuPak formats. Figure 2-9 shows a V-Port functional block diagram.

Video data can be converted from YUV 4:2:2 to AccuPak as it is being stored, or it can be decimated as it is being stored. Decimation and AccuPak conversion cannot be used together. Horizontal and vertical decimation are independently specified. In addition, temporal decimation can be used.

The video capture address can come from either of two register sets, allowing automatic double buffering. When this is used, the video buffers being used for capture and display can be automatically swapped, ensuring that partial images are not displayed.

The CL-GD5446 has an independent video capture FIFO, allowing simultaneous video capture and occlusion or interpolated Y-zooming (subject to frame buffer bandwidth restrictions).

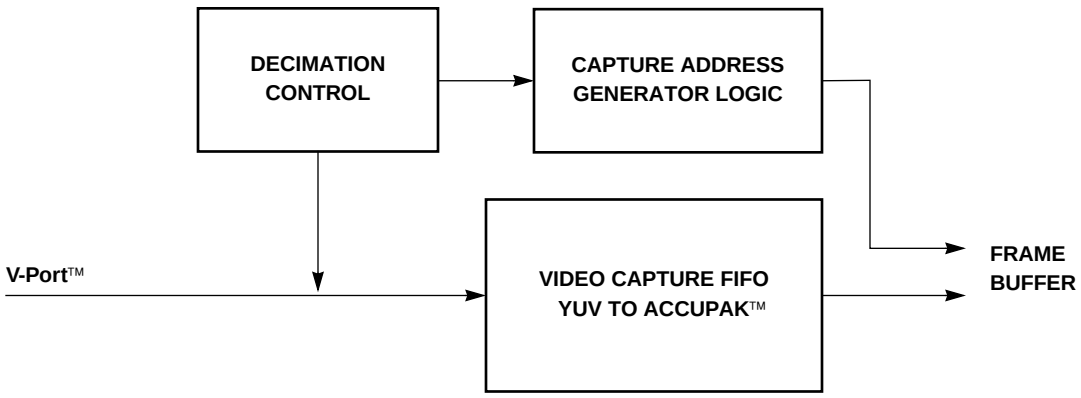


Figure 2-9. V-Port™ Functional Block Diagram

2.7 Video Window and Video Display

The CL-GD5446 has a video window timing generator that defines a rectangular area on the display. This area can display video data or mixed graphics and video data. Video data can (and typically does) have a format different from Graphics data. Typically, Video data also comes from a separate area in the frame buffer.

YUV (4:2:2 in CCIR601 encoding) data is color space converted to RGB in the video pipeline. AccuPak data is expanded to YUV 4:2:2 prior to color space conversion.

Video data can be zoomed for display in the video window. Zoom factors in the range of 1 through 4 are generally used. X-zooming is always done with interpolation (the intermediate pixels values are a weighted average of 'real' pixels). Y-zooming can be done with interpolation (2× or above) or line replication. Interpolation zooming produces superior results and should be used whenever bandwidth requirements permit.

The CL-GD5446 has support for occlusion in the video window. This permits Graphics data and Video data to be mixed on a pixel-by-pixel basis. Color keying the Graphics data or chroma keying the Video data can determine which pixels to replace corresponding pixels from the alternate data stream.

2.7.1 DDC2B/I²C Support

The CL-GD5446 supports a two-pin I²C interface used for DDC2B support. It can also control peripherals such as TV tuners with an I²C interface.

2.7.2 General-Purpose I/O Port

The CL-GD5446 provides address decoding and an 8- or 16-bit data bus for an additional peripheral device on the same adapter board as the GUI-X. The CL-GD5446 provides address and data buffering to comply with the PCI 'one-load' specification. The base address of the port is specified in a PCI configuration register. The port can be in I/O space or memory space.

2.8 Hardware/Software Compatibility

The CL-GD5446 is compatible with the IBM VGA standard.

2.9 Computer Display Subsystem Architecture

Figure 2-10 shows the main components required to implement a functional VGA subsystem using the CL-GD5446. The interfaces that must be implemented are the host CPU, the BIOS (for adapter board implementation only), the display memory, and the CRT. If video is required, the V-Port interface must be implemented.

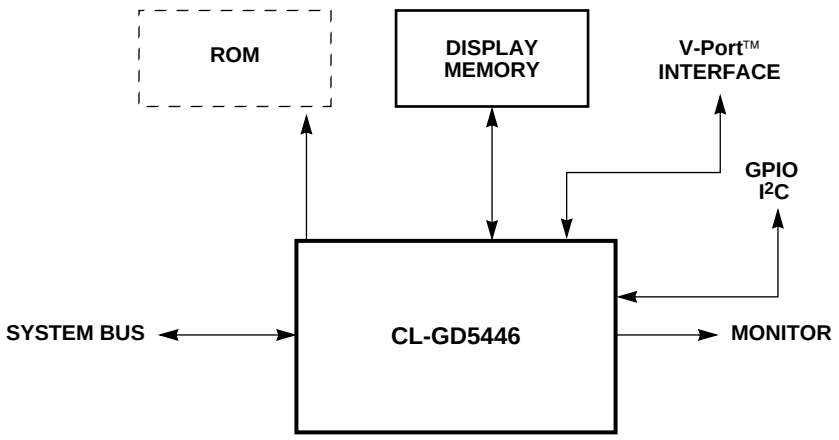


Figure 2-10. Computer Display Subsystem Architecture

Data Book

CL-GD5446

Preliminary Data Book

FEATURES

- **High-throughput PCI bus interface optimized for video playback**
  - — Large write buffer allows sustained zero-wait-state bursts
  - — Independent memory apertures for BitBLT and CPU/video allow concurrent operations for optimized video playback
  - — Byte-swapping for PowerPC™ support
  - — PCI v2.1-compliant
- **Advanced 64-bit BitBLT engine for Windows® 95**
  - — Transparent source data BitBLT for DirectDraw™
  - — Color expansion for all graphics modes
  - — Large data buffers for fast screen-to-screen BitBLTs
  - — Double-buffered, memory-mapped registers with AutoStart™
  - — Optimized color 8 × 8 PatCopy
  - — Accelerated Packed-24 modes
- **64-bit DRAM interface optimized for EDO DRAM**
  - — 80-MHz MCLK offers up to 320 Mbytes/sec. peak bandwidth
  - — Supports new 128K × 16, 128K × 32 DRAM
- **V-Port™, GPIO, I²C bus interfaces for video decoders**
  - — Video capture, closed-caption capture applications
  - — GPIO permits video decoders with single load on PCI bus
  - — Automatic double buffering prevents video 'tearing'
  - — Glueless interface to the CL-PX4072
  - — Interface to MPEG and other video decoders
- **Hardware window for video display**
  - — Multiformat frame buffer
  - — Supports YUV-16 true color video with 8-bit graphics
  - — YUV 4:2:2, AccuPak™, RGB-8, RGB-16 video formats
  - — Unique YUV planar assist mode
  - — Independent interpolated X and Y zooming
  - — Occlusion support with color- or chroma-key
- **PC97-compliant (Revision B)**

(cont.)

64-bit VisualMedia™ Accelerator

OVERVIEW

The CL-GD5446 delivers high-performance graphics and TV-quality, full-motion, full-screen video playback in an integrated, single-chip device. The CL-GD5446 VisualMedia™ accelerator, integrated into a cost-effective personal computer, plays CD-ROM video clips and disk-based video files (including MPEG titles), in full screen at up to 30 frames per second with fully synchronized sound. At the same time, the CL-GD5446 delivers exceptional system throughput with minimal impact to system operation. Transparent BitBLT and page-flipping features provide outstanding DirectDraw™ and games performance.

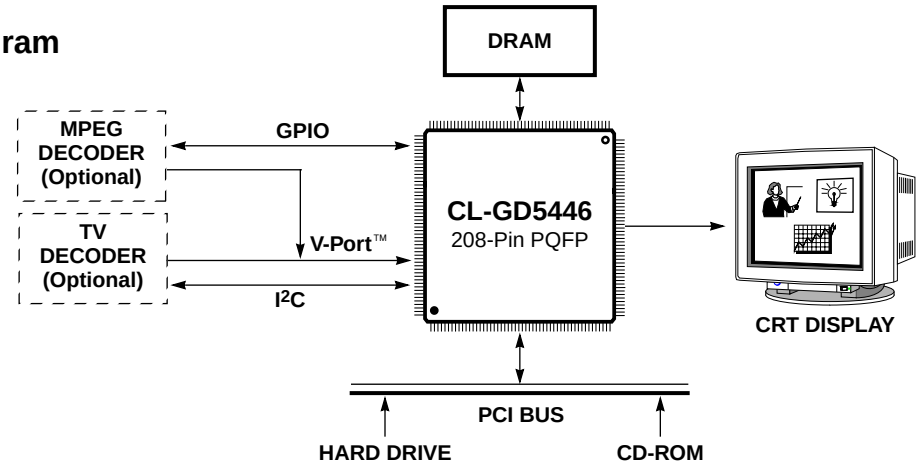
The CL-GD5446 provides a glueless connection to most of the popular video decoder devices from Cirrus Logic as well as other vendors. This provides broad flexibility to support live TV-in-a-window, closed captioning, hardware MPEG, and video conferencing, extending baseline system functions with enhanced features to meet the requirements of a wide range of applications.

The CL-GD5446 can support YUV 4:2:2 video playback in an arbitrarily sized window on 1024 × 768, 256-color graphics with a frame buffer of only 1 Mbyte. This capability can help place PCs using the CL-GD5446 at a very favorable price-performance point.

The CL-GD5446 supports pixel resolutions of up to 1280 × 1024, and 16.8 million colors at resolutions of up to 1024 × 768.

(cont.)

System Block Diagram



FEATURES (cont.)

- **Product differentiation for video-playback applications**
  - Home PC television tuner: 'TV-in-a-window'
  - Video clip capture
  - Home video editing
  - Video mail/video message
  - Personal video conferencing
  - MPEG-1, MPEG-2 applications
  - Closed-caption applications
- **Cirrus Logic provides enabling software drivers**
  - Windows® 95, Windows® 3.x, NT™, OS/2®, and AutoCAD®
  - DirectDraw™ and DCI
  - VPM™ (video port manager)

OVERVIEW (cont.)

The CL-GD5446 features an integrated dual-frequency synthesizer with on-chip oscillator and filters, as well as a triple 8-bit palette DAC with on-chip current reference. Green-PC power-management features help make systems based on the CL-GD5446 compliant with the Energy Star Program.

The CL-GD5446 is software- and pin-compatible with the industry-standard Alpine family of VGA controllers from Cirrus Logic. It comes with the same Cirrus Logic quality software, applications support, and documentation.

Revision B of the CL-GD5446 is PC97-compliant.

Cirrus Logic also provides TV decoder application software — TVTap™ — for the CL-GD5446/'PX407X designs.

ADVANTAGES

Unique Features	Benefits
Outstanding VisualMedia™ Acceleration	
■ High-throughput PCI bus interface	❒ Minimizes host bus bottleneck for VisualMedia™ playback.
■ Advanced 64-bit BitBLT engine with transparent BitBLT and page-flip support	❒ Supports Fast Windows® 95, DirectDraw™, and games.
■ Optimized EDO DRAM interface	❒ 64-bit and 80-MHz MCLK offer best performance for mainstream DRAMs.
■ 128K × 16/32-bit DRAM options	❒ Allows 3- and 1-Mbyte (64-bit) options.
Superior TV-Like-Quality Video Performance	
■ Hardware video window	❒ Allows independent graphics and video streams to be displayed on-screen.
■ X and Y linear interpolated scaling	❒ Minimizes aliasing and allows best video display regardless of screen size.
■ YUV planar assist, AccuPak™ encoding	❒ Technology to obtain best performance while minimizing video-quality degradation.
■ Multiformat frame buffer	❒ Allows true-color video with 256-color graphics.
■ Color key, chroma key	❒ Allows graphics over video in video playback and video capture modes.
Foundation for Differentiation	
■ Video capture V-Port™	❒ Allows video decoder interface and eliminates separate frame buffer for lower system cost.
■ General-purpose I/O bus	❒ Allows single load, glueless, generic I/O interfacing to industry-standard video decoders.
■ VPM™	❒ Video port API for Windows® v3.x and Windows® 95 easing peripheral application software development.
■ I^2C interface	❒ Allows low-cost control interface for applications such as TV decoders.
Compatibility	
■ Compatible with VGA and VESA® standards	❒ Compatible with installed base of systems and software.
■ Drivers supplied at various resolutions for Windows® 3.1, Windows NT™, AutoCAD®, OS/2®, and other popular applications	❒ Provides a 'ready-to-go' solution minimizing the need for additional driver development.

SOFTWARE SUPPORT

Cirrus Logic provides an extensive — and expanding — range of software drivers to enhance the resolution and performance of many popular software packages. Note that the CL-GD5446 VGA graphics portion of a system *does not* require software drivers to run applications in standard-resolution modes.

Cirrus Logic software drivers for the CL-GD5446 include:

Software Drivers	Resolution Supported[a]	No. of Colors
Microsoft® / Intel® DCI (display control interface), DirectDraw™, VPM™ Provider	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
Microsoft® Windows® v3.x Microsoft® Windows® 95	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
Microsoft® Windows NT™ v3.5, v3.51, v4.0	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	16 and 256
	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
OS/2® v2.11, v3.0	640 × 480, 800 × 600, 1024 × 768,1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
AutoCAD® v12.0, v13.0 Autoshade® v2.0 with Renderman 3D Studio™ v1.0, v2.0, v3.0, v4.0	640 × 480, 800 × 600, 1024 × 768, 1280 × 1024	16
	640 × 480, 800 × 600, 1024 × 768, 1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768	32,768
	640 × 480, 800 × 600, 1024 × 768	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million

[a] All monitor types do not support all resolutions; 640 × 480 drivers will run on PS/2®-type monitors. Extended resolutions are dependent upon monitor type and VGA system implementation.

BIOS SUPPORT

- **Fully IBM® VGA-compatible BIOS**
- **Relocatable, 32 Kbytes with PCI bus support**
- **VBE (VESA® BIOS Extensions) support in ROM**
- **Support for DPMS (display power management signaling) in ROM**
- **VESA® monitor timing-compliant**
- **DDC1/2B support**

UTILITIES

- **Graphics and video diagnostics test**
- **Windows® NT™ and DOS utilities**
- **Video mode configuration utility — CLMODE**
- **Set resolution in Windows® utility — WINMODE**
- **Configurable system integration for OEMs — OEMSI**

CONTENTS

Revision History

The following are the differences between the December 1995 and November 1996 versions of this data book:

- Information pertaining to the Revision B device has been added

CONVENTIONS

Abbreviations

Symbol	Units of measure
°C	degree Celsius
Hz	hertz (cycles per second)
Kbyte	kilobyte (1,024 bytes)
kHz	kilohertz
kΩ	kilohm
Mbyte	megabyte (1,048,576 bytes)
MHz	megahertz (1,000 kilohertz)
µF	microfarad
µs	microsecond (1,000 nanoseconds)
mA	milliampere
ms	millisecond (1,000 microseconds)
ns	nanosecond
pV	picovolt

The use of ‘tbd’ indicates values that are ‘to be determined’, ‘n/a’ designates ‘not available’, and ‘n/c’ indicates a pin that is a ‘no connect’.

Numeric Naming

Hexadecimal numbers are represented with all letters in upper case and a lower-case ‘h’ is appended to them (for example, ‘14h’, ‘3A7h’, and ‘C000h’ are hexadecimal numbers). Binary numbers are represented with a lower-case ‘b’ appended. Numbers not indicated by a ‘b’ or an ‘h’ are decimal.

Acronyms

Acronym	Definition
AC	alternating current
ALU	arithmetic logic unit
ATE	automatic test equipment
BIOS	basic input/output system
BitBLT, BLT	bit boundary block transfer
bpp	bits per pixel
CAD	computer-aided design
CAS	column address strobe
CGA	color graphics adapter

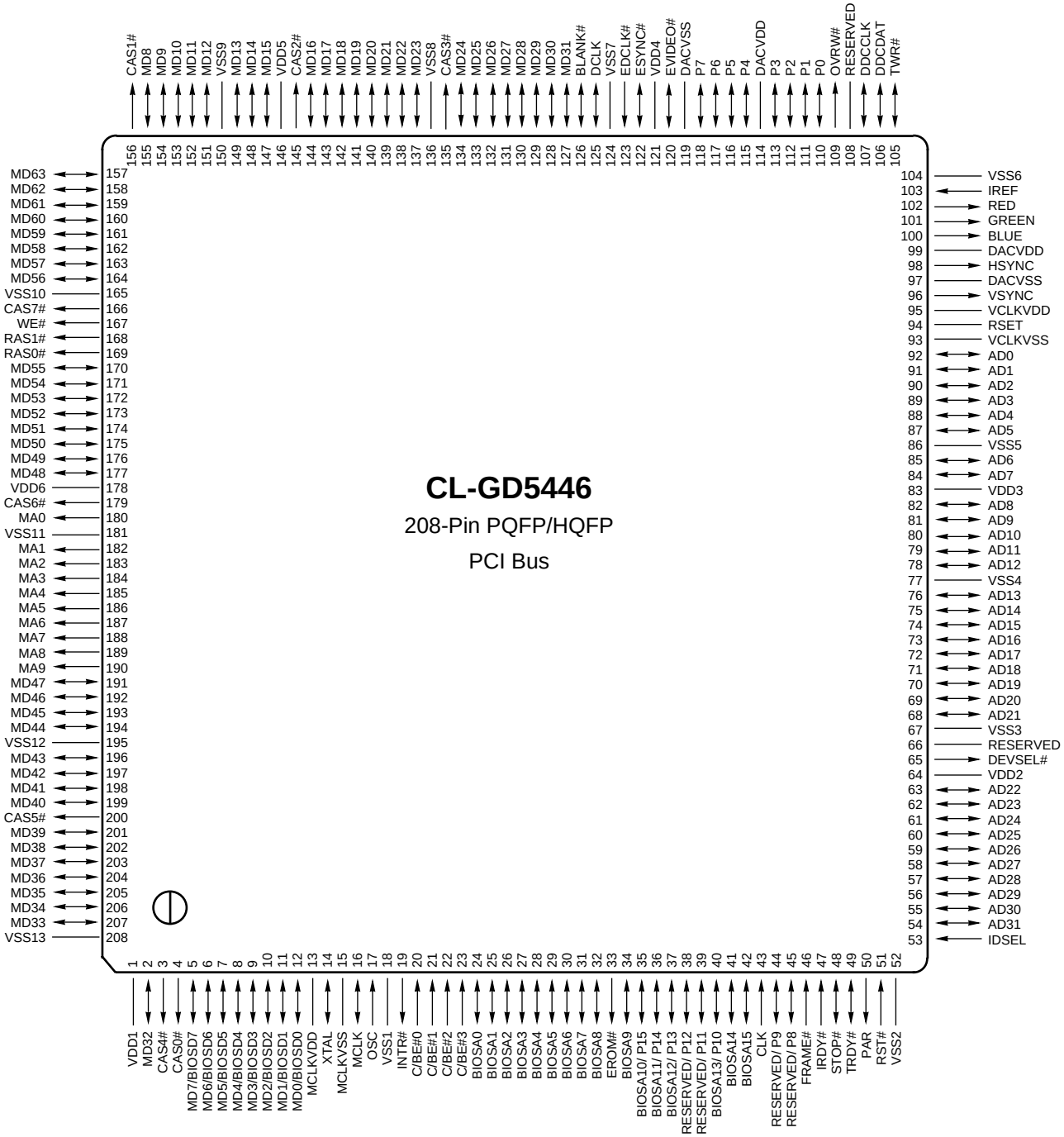
Acronym	Definition
CLUT	color lookup table
CMOS	complementary metal-oxide semiconductor
CPU	central processing unit
CRT	cathode ray tube
CRTC	CRT controller
DAC	digital-to-analog converter
DC	direct current
DDA	digital differential algorithm

Acronym	Definition
DDC	display data channel
DMI	desktop management signaling
DPMS	display power management signaling
DRAM	dynamic random access memory
dword	doubleword
EEPROM	electrically erasable/programmable read-only memory
EGA	enhanced graphics adapter
EPROM	electrically programmable read-only memory
EVAFC	extended VESA® advanced feature connector
FIFO	first in/first out
GPIO	general-purpose IO
GSC	graphics system controller
GUI	graphical user interface
HDR	Hidden DAC register
HRQ	host read queue
HSYNC/VSYNC	horizontal/vertical synchronization
HWQ	host write queue
IC	integrated circuit
I/O	input/output
LBI	local bus interface
LSB	least-significant bit
LUT	lookup table
MA	memory arbiter
MC	memory controller
MCC	monochrome-to-color converter
MD	memory data
MMI/O	memory-mapped I/O
MSB	most-significant bit
OFU	operand fetch unit
OSU	operand storage unit

Acronym	Definition
PCI	peripheral component interconnect
PFS	programmable frequency synthesizer
PLL	phase-locked loop
PQFP	plastic quad-flat pack
qword	two dwords
RAC	Rambus® access channel
RAM	random-access memory
RAS	row address strobe
RDRAM	Rambus® dynamic random-access memory
RGB	red, green, and blue
RIF	Rambus® interface
ROPs	raster operations
RSU	result storage unit
R/W	read/write
SC	serial clock
SG	signature generator
SGRAM	synchronous graphics RAM
SRAM	static random-access memory
TSR	terminate and stay resident
TTL	transistor-transistor logic
VBE	VESA BIOS extensions
VBI	vertical blanking interval
VDD	virtual device driver
VESA®	Video Electronics Standards Association
VGA	video graphics array
VL	VESA® local
VPM	video port manager
VRAM	video random-access memory
WE	transparency write enable

1. PIN INFORMATION

The CL-GD5446 VGA GUI controller is available in a 208-pin PQFP (plastic quad flat pack) or HQFP (high-performance quad flat pack) for the PCI bus only.



1.1 Pin Summary

The following abbreviations are used for pin types in the following tables: (I) indicates input; (O) indicates output; (O-Z) indicates tristate output; (OC) indicates open-collector output; (BIO) indicates bidirectional I/O; (I/O) indicates input or output depending on how the device is configured and programmed.

Table 1-1. Host Interface

Pin Number	Pin Type	Pull-up[a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	PCI	GPIO[b] Redefinition	Pixel Bus[c] Redefinition
53	I					IDSEL		
51	I					RST#		
46	I					FRAME#		
47	I					IRDY#		
43	I					CLK		
49	BIO		–3	8	240	TRDY#		
48	BIO		–3	8	240	STOP#		
50	O		–3	8	240	PAR		
65	O		–3	4	200	DEVSEL#		
19	OC		(OC)	24	200	INTR#		
54	BIO		–3	12	240	AD31		
55	BIO		–3	12	240	AD30		
56	BIO		–3	12	240	AD29		
57	BIO		–3	12	240	AD28		
58	BIO		–3	12	240	AD27		
59	BIO		–3	12	240	AD26		
60	BIO		–3	12	240	AD25		
61	BIO		–3	12	240	AD24		
62	BIO		–3	12	240	AD23		
63	BIO		–3	12	240	AD22		
68	BIO		–3	12	240	AD21		
69	BIO		–3	12	240	AD20		
70	BIO		–3	12	240	AD19		
71	BIO		–3	12	240	AD18		
72	BIO		–3	12	240	AD17		
73	BIO		–3	12	240	AD16		

Table 1-1. Host Interface *(cont.)*

Pin Number	Pin Type	Pull-up[a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	PCI	GPIO[b] Redefinition	Pixel Bus[c] Redefinition
74	BIO		–3	12	240	AD15		
75	BIO		–3	12	240	AD14		
76	BIO		–3	12	240	AD13		
78	BIO		–3	12	240	AD12		
79	BIO		–3	12	240	AD11		
80	BIO		–3	12	240	AD10		
81	BIO		–3	12	240	AD9		
82	BIO		–3	12	240	AD8		
84	BIO		–3	12	240	AD7		
85	BIO		–3	12	240	AD6		
87	BIO		–3	12	240	AD5		
88	BIO		–3	12	240	AD4		
89	BIO		–3	12	240	AD3		
90	BIO		–3	12	240	AD2		
91	BIO		–3	12	240	AD1		
92	BIO		–3	12	240	AD0		
23	I					C/BE#3		
22	I					C/BE#2		
21	I					C/BE#1		
20	I					C/BE#0		
42	I/O		–3	8	50	BIOSA15	GPA1	
41	I/O		–3	8	50	BIOSA14	GPA0	
40	I/O		–3	8	50	BIOSA13	GPD10	P10
37	I/O		–3	8	50	BIOSA12	GPD13	P13
36	I/O		–3	8	50	BIOSA11	GPD14	P14
35	I/O		–3	8	50	BIOSA10	GPD15	P15
34	I/O		–3	8	50	BIOSA9	GPIOWR#	
32	I/O		–3	8	50	BIOSA8	GPIORD#	
31	I/O		–3	8	50	BIOSA7	GPD7	
30	I/O		–3	8	50	BIOSA6	GPD6	

Table 1-1. Host Interface *(cont.)*

Pin Number	Pin Type	Pull-up[a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	PCI	GPIO[b] Redefinition	Pixel Bus[c] Redefinition
29	I/O		–3	8	50	BIOSA5	GPD5	
28	I/O		–3	8	50	BIOSA4	GPD4	
27	I/O		–3	8	50	BIOSA3	GPD3	
26	I/O		–3	8	50	BIOSA2	GPD2	
25	I/O		–3	8	50	BIOSA1	GPD1	
24	I/O		–3	8	50	BIOSA0	GPD0	
66	I/O		–3	8	50	Reserved	GPRDY/DT	
45	I/O		–3	8	50	Reserved	GPD8	P8
44	I/O		–3	8	50	Reserved	GPD9	P9
39	I/O		–3	8	50	Reserved	GPD11	P11
38	I/O		–3	8	50	Reserved	GPD12	P12

[a] Indicates nominal 250-kΩ pull-up resistor.

[b] See Appendix B11 for additional information on the general-purpose I/O port.

[c] See the definition of register GR18[6].

Table 1-2. Video Interface

Pin Number	Pin Type	Pull-up[a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name	V-Port™ Redefinition
96	O-Z		–12	24	50	VSYNC	
98	O-Z		–12	24	50	HSYNC	
126	I/O		–12	12	50	BLANK#	HREF Input
35	I/O		–3	8	50	P15[b]	PIXD15
36	I/O		–3	8	50	P14[b]	PIXD14
37	I/O		–3	8	50	P13[b]	PIXD13
38	I/O		–3	8	50	P12[b]	PIXD12
39	I/O		–3	8	50	P11[b]	PIXD11
40	I/O		–3	8	50	P10[b]	PIXD10
44	I/O		–3	8	50	P9[b]	PIXD9
45	I/O		–3	8	50	P8[b]	PIXD8
118	I/O		–12	12	50	P7	PIXD7
117	I/O		–12	12	50	P6	PIXD6

Table 1-2. Video Interface *(cont.)*

Pin Number	Pin Type	Pull-up [a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name	V-Port™ Redefinition
116	I/O		–12	12	50	P5	PIXD5
115	I/O		–12	12	50	P4	PIXD4
113	I/O		–12	12	50	P3	PIXD3
112	I/O		–12	12	50	P2	PIXD2
111	I/O		–12	12	50	P1	PIXD1
110	I/O		–12	12	50	P0	PIXD0
125	I/O		–12	12	50	DCLK	PIXCLK Input
122	I/O	●	–12	12		ESYNC#	(Prog. Output 1)
120	I/O	●	–12	12		EVIDEO#	VACT Input
123	I	●				EDCLK#	VREF Input
102	Analog Out					RED	
101	Analog Out					GREEN	
100	Analog Out					BLUE	
103	Analog In					IREF	
94	Analog In					RSET	

[a] ● indicates the presence of an internal 250-kΩ ±50% pull-up resistor.

[b] P[15:8] are redefined PCI pins. See the definition of register GR18[6].

Table 1-3. Display Memory Interface

Pin Number	Pin Type	Pull-up [a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name
168	O		–12	12	50	RAS1#
169	O		–12	12	50	RAS0#
166	O		–12	12	50	CAS7#
179	O		–12	12	50	CAS6#
200	O		–12	12	50	CAS5#
3	O		–12	12	50	CAS4#
135	O		–12	12	50	CAS3#
145	O		–12	12	50	CAS2#
156	O		–12	12	50	CAS1#
4	O		–12	12	50	CAS0#

Table 1-3. Display Memory Interface *(cont.)*

Pin Number	Pin Type	Pull-up [a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name
167	O		–12	12	150	WE#
190	O		–12	12	150	MA9
189	O		–12	12	150	MA8 [b]
188	O		–12	12	150	MA7
187	O		–12	12	150	MA6
186	O		–12	12	150	MA5
185	O		–12	12	150	MA4
184	O		–12	12	150	MA3
183	O		–12	12	150	MA2
182	O		–12	12	150	MA1
180	O		–12	12	150	MA0 [c]
157	I/O	●	–8	8	50	MD63
158	I/O	●	–8	8	50	MD62
159	I/O	●	–8	8	50	MD61
160	I/O	●	–8	8	50	MD60
161	I/O	●	–8	8	50	MD59
162	I/O	●	–8	8	50	MD58
163	I/O	●	–8	8	50	MD57
164	I/O	●	–8	8	50	MD56
170	I/O	●	–8	8	50	MD55
171	I/O	●	–8	8	50	MD54
172	I/O	●	–8	8	50	MD53
173	I/O	●	–8	8	50	MD52
174	I/O	●	–8	8	50	MD51
175	I/O	●	–8	8	50	MD50
176	I/O	●	–8	8	50	MD49
177	I/O	●	–8	8	50	MD48
191	I/O	●	–8	8	50	MD47
192	I/O	●	–8	8	50	MD46
193	I/O	●	–8	8	50	MD45
194	I/O	●	–8	8	50	MD44

Table 1-3. Display Memory Interface *(cont.)*

Pin Number	Pin Type	Pull-up [a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name
196	I/O	●	–8	8	50	MD43
197	I/O	●	–8	8	50	MD42
198	I/O	●	–8	8	50	MD41
199	I/O	●	–8	8	50	MD40
201	I/O	●	–8	8	50	MD39
202	I/O	●	–8	8	50	MD38
203	I/O	●	–8	8	50	MD37
204	I/O	●	–8	8	50	MD36
205	I/O	●	–8	8	50	MD35
206	I/O	●	–8	8	50	MD34
207	I/O	●	–8	8	50	MD33
2	I/O	●	–8	8	50	MD32
127	I/O	●	–8	8	50	MD31
128	I/O	●	–8	8	50	MD30
129	I/O	●	–8	8	50	MD29
130	I/O	●	–8	8	50	MD28
131	I/O	●	–8	8	50	MD27
132	I/O	●	–8	8	50	MD26
133	I/O	●	–8	8	50	MD25
134	I/O	●	–8	8	50	MD24
137	I/O	●	–8	8	50	MD23
138	I/O	●	–8	8	50	MD22
139	I/O	●	–8	8	50	MD21
140	I/O	●	–8	8	50	MD20
141	I/O	●	–8	8	50	MD19
142	I/O	●	–8	8	50	MD18
143	I/O	●	–8	8	50	MD17
144	I/O	●	–8	8	50	MD16
147	I/O	●	–8	8	50	MD15
148	I/O	●	–8	8	50	MD14
149	I/O	●	–8	8	50	MD13

Table 1-3. Display Memory Interface *(cont.)*

Pin Number	Pin Type	Pull-up [a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name
151	I/O	●	–8	8	50	MD12
152	I/O	●	–8	8	50	MD11
153	I/O	●	–8	8	50	MD10
154	I/O	●	–8	8	50	MD9
155	I/O	●	–8	8	50	MD8
5	I/O	●	–8	8	50	MD7/BIOSD7 [d]
6	I/O	●	–8	8	50	MD6/BIOSD6 [d]
7	I/O	●	–8	8	50	MD5/BIOSD5 [d]
8	I/O	●	–8	8	50	MD4/BIOSD4 [d]
9	I/O	●	–8	8	50	MD3/BIOSD3 [d]
10	I/O	●	–8	8	50	MD2/BIOSD2 [d]
11	I/O	●	–8	8	50	MD1/BIOSD1 [d]
12	I/O	●	–8	8	50	MD0/BIOSD0 [d]

[a] ● indicates the presence of an internal 250-kΩ ±50% pull-up resistor.

[b] MA8 is connected to Memory Address 0 for asymmetric DRAMs.

[c] MA0 is connected to Memory Address 8 for asymmetric DRAMs.

[d] MD[7:0] are also used as the BIOS Data Input pins.

Table 1-4. General-Purpose I/O [a] Port

Pin Number	Pin Type	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Pin Name	Redefined As
35	I/O	–3	8	50	BIOSA10	GPD15
36	I/O	–3	8	50	BIOSA11	GPD14
37	I/O	–3	8	50	BIOSA12	GPD13
38	I/O	–3	8	50	Reserved	GPD12
39	I/O	–3	8	50	Reserved	GPD11
40	I/O	–3	8	50	BIOSA13	GPD10
44	I/O	–3	8	50	Reserved	GPD9
45	I/O	–3	8	50	Reserved	GPD8
31	I/O	–3	8	50	BIOSA7	GPD7
30	I/O	–3	8	50	BIOSA6	GPD6
29	I/O	–3	8	50	BIOSA5	GPD5
28	I/O	–3	8	50	BIOSA4	GPD4

Table 1-4. General-Purpose I/O[a] Port

Pin Number	Pin Type	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Pin Name	Redefined As
27	I/O	–3	8	50	BIOSA3	GPD3
26	I/O	–3	8	50	BIOSA2	GPD2
25	I/O	–3	8	50	BIOSA1	GPD1
24	I/O	–3	8	50	BIOSA0	GPD0
14	I/O	–3	8	50	XTAL	GPA6
42	I/O	–3	8	50	BIOSA15	GPA5/GPA1
41	I/O	–3	8	50	BIOSA14	GPA4/GPA0
109	O	–3	8	50	OVRW#	GPA3
105	I/O	–3	8	50	TWR#	GPA2
108	O	–3	8	50	Reserved	GPCS#
32	I/O	–3	8	50	BIOSA8	GPIORD#
34	I/O	–3	8	50	BIOSA9	GPIOWR#
66	I/O	–3	8	50	Reserved	GPRDY/DT

[a] The pins in this table are redefined to be used for the General-Purpose I/O port. See Appendix B11.

Table 1-5. V-Port™[a]

Pin Number	Pin Type	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Pin Name	Redefined As
35	I/O	–3	8	50	BIOSA10	PIXD15
36	I/O	–3	8	50	BIOSA11	PIXD14
37	I/O	–3	8	50	BIOSA12	PIXD13
38	I/O	–3	8	50	Reserved	PIXD12
39	I/O	–3	8	50	Reserved	PIXD11
40	I/O	–3	8	50	BIOSA13	PIXD10
44	I/O	–3	8	50	Reserved	PIXD9
45	I/O	–3	8	50	Reserved	PIXD8
118	I/O	–12	12	50	P7	PIXD7
117	I/O	–12	12	50	P6	PIXD6
116	I/O	–12	12	50	P5	PIXD5
115	I/O	–12	12	50	P4	PIXD4
113	I/O	–12	12	50	P3	PIXD3
112	I/O	–12	12	50	P2	PIXD2

Table 1-5. V-Port™[a]

Pin Number	Pin Type	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Pin Name	Redefined As
111	I/O	–12	12	50	P1	PIXD1
110	I/O	–12	12	50	P0	PIXD0
120	I/O	–12	12	50	EVIDEO#	VACT
123	I				EDCLK#	VREF
125	I/O	–12	12	50	DCLK	PIXCLK
126	I/O	–12	12	50	BLANK#	HREF

[a] The pins in this table are redefined to be used for the V-Port.

Table 1-6. Miscellaneous Pins

Pin Number	Pin Type	Pull-up[a]	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name	GPIO[b] Redefinition
107	I/O		–12	12	35	DDCCLK	
106	I/O		–12	12	35	DDCDAT	
33	O		–12	12	35	EROM#	
109	O		–12	12	35	OVRW#	GPA3
105	I	●				TWR#	GPA2
108	–					Reserved	GPCS#

[a] ● indicates the presence of an internal 250-kΩ ±50% pull-up resistor.

[b] These pins are also used for the General-Purpose I/O port. See Appendix B11.

Table 1-7. Clock Synthesizer Interface

Pin Number	Pin Type	I_{OH} (mA)	I_{OL} (mA)	Load (pF)	Name
17	I				OSC
14	O				XTAL
16	I/O	–12	12	20	MCLK[a]

[a] Pin 16 is also used as Programmable Output 0.

Table 1-8. Power and Ground

Pin Number	Pin Type	Name	Note
178	Power	VDD6	Digital
146	Power	VDD5	Digital
121	Power	VDD4	Digital
83	Power	VDD3	Digital
64	Power	VDD2	Digital
1	Power	VDD1	Digital
208	Ground	VSS13	Digital
195	Ground	VSS12	Digital
181	Ground	VSS11	Digital
165	Ground	VSS10	Digital
150	Ground	VSS9	Digital
136	Ground	VSS8	Digital
124	Ground	VSS7	Digital
104	Ground	VSS6	Digital
86	Ground	VSS5	Digital
77	Ground	VSS4	Digital
67	Ground	VSS3	Digital
52	Ground	VSS2	Digital
18	Ground	VSS1	Digital
95	Power	VCLKVDD	VCLK
93	Ground	VCLKVSS	VCLK
13	Power	MCLKVDD	MCLK
15	Ground	MCLKVSS	MCLK
114	Power	DACVDD	DAC
99	Power	DACVDD	DAC
119	Ground	DACVSS	DAC
97	Ground	DACVSS	DAC

Table 1-9. Pins with Multiple Uses (Ordered by Pin Number)

Pin Number	PCI Name[a]	GPIO[b]	V-Port™[c]	VESA®	VMI[d] Interface	Memory Bus	Programmable I/O	Other
5	BIOSD7	–	–	–	–	MD7	–	–
6	BIOSD6	–	–	–	–	MD6	–	–
7	BIOSD5	–	–	–	–	MD5	–	–
8	BIOSD4	–	–	–	–	MD4	–	–
9	BIOSD3	–	–	–	–	MD3	–	–
10	BIOSD2	–	–	–	–	MD2	–	–
11	BIOSD1	–	–	–	–	MD1	–	–
12	BIOSD0	–	–	–	–	MD0	–	–
14	–	GPA6	–	–	–	–	–	XTAL
16	–	–	–	–	–	–	Prog. Out 0	MCLK
24	BIOSA0	GPD0	–	–	HD[0]	–	–	–
25	BIOSA1	GPD1	–	–	HD[1]	–	–	–
26	BIOSA2	GPD2	–	–	HD[2]	–	–	–
27	BIOSA3	GPD3	–	–	HD[3]	–	–	–
28	BIOSA4	GPD4	–	–	HD[4]	–	–	–
29	BIOSA5	GPD5	–	–	HD[5]	–	–	–
30	BIOSA6	GPD6	–	–	HD[6]	–	–	–
31	BIOSA7	GPD7	–	–	HD[7]	–	–	–
32	BIOSA8	GPIORD#	–	–	RD#	–	–	–
34	BIOSA9	GPIOWR#	–	–	WR#	–	–	–
35	BIOSA10	GPD15	PIXD15	P15	–	–	–	–
36	BIOSA11	GPD14	PIXD14	P14	–	–	–	–
37	BIOSA12	GPD13	PIXD13	P13	–	–	–	–
38	–	GPD12	PIXD12	P12	–	–	–	–
39	–	GPD11	PIXD11	P11	–	–	–	–
40	BIOSA13	GPD10	PIXD10	P10	–	–	–	–
41	BIOSA14	GPA0	–	–	HA[0]	–	–	–
42	BIOSA15	GPA1	–	–	HA[1]	–	–	–
44	–	GPD9	PIXD9	P9	–	–	–	–
45	–	GPD8	PIXD8	P8	–	–	–	–

Table 1-9. Pins with Multiple Uses (Ordered by Pin Number) *(cont.)*

Pin Number	PCI Name[a]	GPIO[b]	V-Port™[c]	VESA®	VMI[d] Interface	Memory Bus	Programmable I/O	Other
66	–	GPDRY/DT	–	–	GPRDY	–	–	–
105	–	GPA2	–	–	HA[2]	–	–	TWR#
108	–	GPCS#	–	–	CS#	–	–	–
109	–	GPA3	–	–	HA[3]	–	–	OVRW#
110	–	–	PIXD0	P0	–	–	–	–
111	–	–	PIXD1	P1	–	–	–	–
112	–	–	PIXD2	P2	–	–	–	–
113	–	–	PIXD3	P3	–	–	–	–
115	–	–	PIXD4	P4	–	–	–	–
116	–	–	PIXD5	P5	–	–	–	–
117	–	–	PIXD6	P6	–	–	–	–
118	–	–	PIXD7	P7	–	–	–	–
120	–	–	VACT	EVIDEO#	–	–	–	–
122	–	–	–	ESYNC#	–	–	Prog. Out 1	–
123	–	–	VREF	EDCLK#	–	–	–	–
125	–	–	PIXCLK	DCLK	–	–	–	–
126	–	–	HREF	BLANK#	–	–	–	–

[a] These functions are enabled when the BIOS is enabled in PCI30.

[b] GPIO is configured with CF8, CF4, CF3 (Revision A only).

[c] These pins are configured for V-Port in CR50[4] and CR50[1:0].

[d] These are the pin names on the VMI interface for reference only.

2. FUNCTIONAL DESCRIPTION

2.1 General

The CL-GD5446 offers a VGA solution that is totally compatible with the IBM VGA standard. The CL-GD5446 includes a VGA core, 64-bit BitBLT engine, video capture and display, and on-board frequency synthesizers and palette DAC. A complete VGA motherboard solution can be implemented by using two 256K × 16 DRAMs with the CL-GD5446.

Figure 2-1 presents a functional block diagram of the CL-GD5446, showing the connections to the host, display memory, V-Port, and monitor.

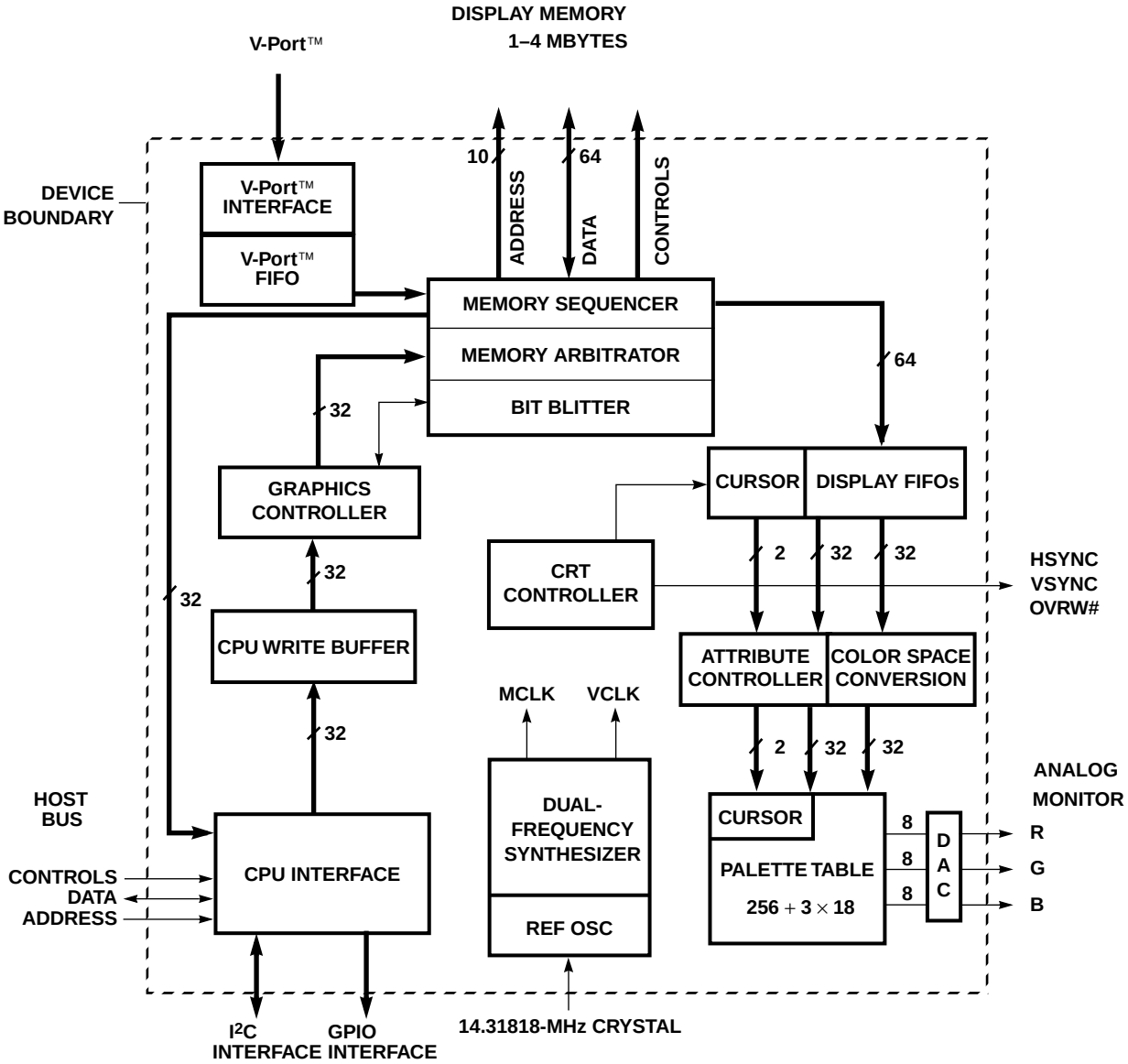


Figure 2-1. CL-GD5446 Functional Block Diagram

2.2 Functional Blocks

The following sections describe functional blocks that are integrated into the CL-GD5446.

2.2.1 CPU Interface

The CL-GD5446 connects directly to the PCI bus with no glue logic. The CL-GD5446 decodes the entire 32-bit address so that no address mirroring occurs. The CL-GD5446 interface executes 32-bit I/O and memory accesses at a speed of up to 33 MHz. The CL-GD5446 also supports memory burst cycles. The CL-GD5446 can support an additional peripheral device while remaining fully compliant with the PCI single-load specification. The CL-GD5446 is PCI 2.1-compliant.

Revision B of the CL-GD5446 has two 16-Mbyte windows into the frame buffer for compliance with PC97.

2.2.2 CPU Write Buffer

The CL-GD5446 has a multi-level 32-bit CPU write buffer which dramatically increases GUI acceleration and enhances CPU performance. The CPU write buffer contains a queue of CPU write accesses to display memory or the BitBLT engine that have not been executed because frame buffer bandwidth has not yet been available. Maintaining a queue allows the CL-GD5446 to generate TRDY# to complete the bus cycle as soon as it has recorded the address and data, and then to execute the operation when display memory cycles are available.

2.2.3 Graphics Controller

The graphics controller is located between the CPU interface and the memory sequencer. It performs text manipulation, data rotation, color mapping, and miscellaneous operations. These operations are typically performed in the graphics controller for VGA-compatible applications; newer applications take advantage of the BitBLT engine.

2.2.4 BitBLT Engine

The CL-GD5446 has a 64-bit BitBLT engine that supports color expansion with or without transparency for all graphics pixel sizes as well as transparency without color expansion for 8- and 16-bpp graphics formats.

The Control registers for the BitBLT engine are memory-mapped and double-buffered. Memory-mapping the Control registers allows the fastest possible parameter transfer. Double-Buffered Control registers and the AutoStart feature provide the greatest possible degree of parallelism between the host and the BitBLT engine.

2.2.5 Memory Arbitrator

The memory arbitrator allocates bandwidth to the four functions that compete for the frame buffer bandwidth: DRAM refresh, screen refresh, V-Port writes, and CPU and BitBLT access.

DRAM refresh is handled invisibly by allocating a selectable number of CAS#-before-RAS# refresh cycles at the beginning of each scanline. Screen refresh, V-Port writes, and CPU/BitBLT access are allocated cycles according to the FIFO control parameters. Priority is given to screen refresh and V-Port writes.

2.2.6 Memory Sequencer

The memory sequencer generates timing for display memory. The CL-GD5446 can be configured to generate timing optimized for EDO (extended data output) DRAMS with MCLK programmable up to 80 MHz. The control signals from the CL-GD5446 to the DRAM are RAS#, CAS#, WE#, and the multiplexed address bus. The sequencer generates CAS#-before-RAS# refresh cycles, random read and random early write cycles, Fast-Page mode read and early write cycles, and EDO read cycles. The memory sequencer can generate addresses for symmetric or asymmetric DRAMs.

2.2.7 CRT Controller

The CRT controller generates all the timing required by the monitor including HSYNC, HSYNC, and BLANK#. The sync signals have programmable polarity and can be forced static for monitor power management. The CL-GD5446 BIOS supports all standard VGA modes, as well as extended resolutions up to 1280 × 1024. The CL-GD5446 supports a hardware video window for video playback.

2.2.8 Display FIFOs

The display FIFOs allow data from the frame buffer to be fetched before it is actually needed for screen refresh. This allows the fetches to be executed as EDO Fast-Page mode read cycles rather than random read cycles, greatly increasing the available memory bandwidth. The CL-GD5446 has two display FIFOs, allowing information from two independent sources streams to be mixed together in the display pipeline. This is necessary for occlusion support and also for Y-interpolation.

2.2.9 Attribute Controller

The attribute controller formats the display for the screen (primarily text modes). Display color selection, text blinking, and underlining are performed by the attribute controller. Alternate font selection also occurs in the attribute controller.

2.2.10 V-Port™

The CL-GD5446 V-Port writes realtime or recorded video from a decoder to the frame buffer, typically for display in the video window. Video can be converted to AccuPak™ or can be decimated vertically and horizontally. When video is being captured for display in the window, the capture and display buffers can be automatically swapped as each frame is captured. This prevents the display of partial frames with a minimum of host intervention.

The CL-GD5446 has an independent capture FIFO. This allows video capture to occur at the same time interpolated Y-zooming or occlusion is being used.

Luminance-only capture is available for TeleText and closed caption with suitable software.

The V-Port hardware interface uses the same pins as the VGA pass-through connector. It can be configured for an 8- or 16-bit pixel bus and for either active sense of HREF.

2.2.11 Hardware Video Window

The CL-GD5446 features a programmable hardware window for the simultaneous display of graphics and video. The graphics and video formats can have different color spaces and even pixel sizes. The display of 8-bpp palettized graphics with YUV 4:2:2 graphics is a typical application.

The video can be independently zoomed in the horizontal and vertical directions up to 4×. Horizontal zooming is always done with interpolation of 'in-between' pixels. Vertical zooming can be done with scanline replication. Scanline interpolation can be used for vertical zooming at 2× or greater (subject to frame buffer bandwidth limitations).

Occlusion support allows the graphics and video streams to be mixed on a pixel-by-pixel basis. Color key matching of the graphics source or chroma key matching of the video source can be used to determine which pixels are replaced. Occlusion is supported for 8- and 16-bpp graphics. Occlusion and Y-zoom with interpolation are mutually exclusive.

2.2.12 Palette DAC

The palette DAC block contains the color palette and three 8-bit digital-to-analog converters. The color palette, with 256 18-bit entries, converts an 8-bit color code that specifies the color of a pixel into three 6-bit values, one each for red, green, and blue.

Alternatively, the CL-GD5446 can be configured for 8-, 15-, 16-, or 24-bit direct color RGB pixels. This allows 256, 32K, 64K, or 16M simultaneous colors to be displayed on the screen.

The CL-GD5446 also supports YUV 4:2:2 and AccuPak formats within the video window.

The palette DAC supports a Power-Down mode which temporarily turns off clocks to the palette and power to the DAC to conserve power.

2.2.13 Dual-Frequency Synthesizer

The dual-frequency synthesizer generates the memory sequencer and display clocks from a single reference frequency. The frequency of each clock is independently programmable. The maximum memory sequencer clock and display clock are 80 MHz and 135 MHz, respectively. The reference frequency of 14.31818 MHz can be generated on-chip using an inexpensive 2-pin crystal or it can be supplied from an external TTL source.

2.2.14 VESA®/VGA Pass-Through Connector

The CL-GD5446 can connect directly to a VESA connector for input or output. The device supports the three enable/disable inputs; the Pixel bus can drive the connector directly.

2.2.15 General-Purpose I/O Port

The CL-GD5446 can support an additional peripheral device on its adapter card. Address decoding and data buffering allow the additional device while maintaining the PCI 'single-load' specification.

2.2.16 I²C Interface

The CL-GD5446 has a built-in two pin interface that can be used to control peripheral devices such as TV tuners. This interface can also be used for DDC2B monitor identification.

2.3 Performance

The CL-GD5446 is designed with the following performance-enhancing features:

- 64-bit display memory data bus for faster access to display memory
- Memory-mapped, double-buffered BitBLT registers with autostart maximizes host/BLT overlap
- Transparent source BitBLT for increased BLT functionality
- DRAM timing configurable for EDO operations for faster access to display memory
- 80-MHz MCLK provides 320-Mbyte/second peak frame-buffer bandwidth
- Burst host bus performance and a CPU write buffer that allows faster CPU access for writes to display memory
- Increased throughput with PCI local bus interface with Burst mode
- 32-bit CPU interface to display memory for faster host access in all modes, including Planar mode
- 16- or 32-bit CPU interface to I/O registers for faster host access
- Multi-level, 32-bit system memory write cache
- 32-bit internal data inputs for internal DAC
- Two display FIFOs to minimize memory contention
- Video capture decimation to reduce the memory bandwidth requirements
- YUV planar assist and AccuPak™ reduce codec CPU processing, increasing host bus and memory bus transfer rates
- 32 × 32 and 64 × 64 hardware cursor to improve Microsoft® Windows® performance

2.4 Compatibility

The CL-GD5446 includes all registers and data paths required for VGA controllers, and is upward-compatible with the CL-GD542X family.

The CL-GD5446 supports extensions to VGA, including 1024 × 768 × 16M interlaced, 1024 × 768 × 64K interlaced and non-interlaced, and 1280 × 1024 × 256 interlaced and non-interlaced modes.

Production Revision B of the CL-GD5446 is compliant with PC97.

2.5 Board Testability

The CL-GD5446 device is testable, even when installed on a printed circuit board. By using PinScan testing, any IC signal pin not connected to the board or shorted to a neighboring pin or trace, is detected (see Appendix B7, "Pin Scan" in the *CL-GD5446 Technical Reference Manual*).The signature generator allows the entire system, including the display memory, to be tested at speed (see Appendix B6, "Signature Generator" in the *CL-GD5446 Technical Reference Manual*). The CL-GD5446 enhanced signature generator test allows the BitBLT engine, the V-Port, as well as the frame buffer to be tested.

3. CONFIGURATION TABLES

3.1 Graphics Modes

Table 3-1. IBM® Standard VGA Display Modes

Mode No.	VESA® No.	No. of Colors	Char. × Row	Char. Cell	Screen Format	Display Mode	Pixel Freq. MHz	Horiz. Freq. kHz	Vert. Freq. Hz
0, 1	0, 1	16/256K	40 × 25	9 × 16	360 × 400	Text	14	31.5	70
2, 3	2, 3	16/256K	80 × 25	9 × 16	720 × 400	Text	28	31.5	70
4, 5	4, 5	4/256K	40 × 25	8 × 8	320 × 200	Graphics	12.5	31.5	70
6	6	2/256K	80 × 25	8 × 8	640 × 200	Graphics	25	31.5	70
7	7	Monochrome	80 × 25	9 × 16	720 × 400	Text	28	31.5	70
D	D	16/256K	40 × 25	8 × 8	320 × 200	Graphics	12.5	31.5	70
E	E	16/256K	80 × 25	8 × 14	640 × 200	Graphics	25	31.5	70
F	F	Monochrome	80 × 25	8 × 14	640 × 350	Graphics	25	31.5	70
10	10	16/256K	80 × 25	8 × 14	640 × 350	Graphics	25	31.5	70
11	11	2/256K	80 × 30	8 × 16	640 × 480	Graphics	25	31.5	60
11+	11	2/256K	80 × 30	8 × 16	640 × 480	Graphics	31.5	37.9	72
11+	11	2/256K	80 × 30	8 × 16	640 × 480	Graphics	31.5	37.5	75
12	12	16/256K	80 × 30	8 × 16	640 × 480	Graphics	25	31.5	60
12+	12+	16/256K	80 × 30	8 × 16	640 × 480	Graphics	31.5	37.9	72
12+	12+	16/256K	80 × 30	8 × 16	640 × 480	Graphics	31.5	37.5	75
12+	12+	16/256K	80 × 30	8 × 16	640 × 480	Graphics	35.8	43.3	85
13	13	256/256K	40 × 25	8 × 8	320 × 200	Graphics	12.5	31.5	70

NOTE: The EGA-compatible text modes (which use an 8 × 14 font) and graphics modes 10 and F use a 16-dot high font, with the bottom two lines truncated, in the absence of TSRFONT (8 × 14 font TSR). This creates some errors when displaying characters with descenders, but does not restrict operation of programs using these modes. In text modes using the 8 × 14 font, the characters ‘g’, ‘j’, ‘p’, ‘q’, ‘y’, and ‘ÿ’ are truncated using a middle- and bottom-line algorithm to avoid truncation of descenders. For compatibility with some DOS applications using the 8 × 14 font, the TSRFONT utility should be used. Applications such as DOSSHELL, in Graphics 25 or 34 line display modes, require the TSRFONT utility to be loaded.

Table 3-2. Cirrus Logic Extended Display Modes

Mode No.	VESA® No.	No. of Colors	Char. × Row	Char. Cell	Screen Format	Display Mode	Pixel Freq. MHz	Horiz. Freq. kHz	Vert. Freq. Hz
58, 6A	102	16/256K	100 × 37	8 × 16	800 × 600	Graphics	36	35.2	56
58, 6A	102	16/256K	100 × 37	8 × 16	800 × 600	Graphics	40	37.8	60
58, 6A	102	16/256K	100 × 37	8 × 16	800 × 600	Graphics	50	48.1	72
58, 6A	102	16/256K	100 × 37	8 × 16	800 × 600	Graphics	49.5	46.9	75
5C	103	256/256K	100 × 37	8 × 16	800 × 600	Graphics	36	35.2	56
5C	103	256/256K	100 × 37	8 × 16	800 × 600	Graphics	40	37.9	60
5C	103	256/256K	100 × 37	8 × 16	800 × 600	Graphics	50	48.1	72
5C	103	256/256K	100 × 37	8 × 16	800 × 600	Graphics	49.5	46.9	75
5C	103	256/256K	100 × 37	8 × 16	800 × 600	Graphics	56.25	53.7	85
5D†	104	16/256K	128 × 48	8 × 16	1024 × 768	Graphics	44.9	35.5	43i†
5D	104	16/256K	128 × 48	8 × 16	1024 × 768	Graphics	65	48.3	60
5D	104	16/256K	128 × 48	8 × 16	1024 × 768	Graphics	75	56	70
5D	104	16/256K	128 × 48	8 × 16	1024 × 768	Graphics	77	58	72
5D	104	16/256K	128 × 48	8 × 16	1024 × 768	Graphics	78.7	60	75
5E	100	256/256K	80 × 25	8 × 16	640 × 400	Graphics	25	31.5	70
5F	101	256/256K	80 × 30	8 × 16	640 × 480	Graphics	25	31.5	60
5F	101	256/256K	80 × 30	8 × 16	640 × 480	Graphics	31.5	37.9	72
5F	101	256/256K	80 × 30	8 × 16	640 × 480	Graphics	31.5	37.5	75
5F	101	256/256K	80 × 30	8 × 16	640 × 480	Graphics	36	43.3	85
60†	105	256/256K	128 × 48	8 × 16	1024 × 768	Graphics	44.9	35.5	43i†
60	105	256/256K	128 × 48	8 × 16	1024 × 768	Graphics	65	48.3	60
60	105	256/256K	128 × 48	8 × 16	1024 × 768	Graphics	75	56	70
60	105	256/256K	128 × 48	8 × 16	1024 × 768	Graphics	77	58	72
60	105	256/256K	128 × 48	8 × 16	1024 × 768	Graphics	78.7	60	75
60	105	256/256K	128 × 48	8 × 16	1024 × 768	Graphics	94.5	68.3	85
64	111	64K	–	–	640 × 480	Graphics	25	31.5	60
64	111	64K	–	–	640 × 480	Graphics	31.5	37.9	72
64	111	64K	–	–	640 × 480	Graphics	31.5	37.5	75
64	111	64K	–	–	640 × 480	Graphics	36	43.3	85

Table 3-2. Cirrus Logic Extended Display Modes *(cont.)*

Mode No.	VESA® No.	No. of Colors	Char. × Row	Char. Cell	Screen Format	Display Mode	Pixel Freq. MHz	Horiz. Freq. kHz	Vert. Freq. Hz
65	114	64K	–	–	800 × 600	Graphics	36	35.2	56
65	114	64K	–	–	800 × 600	Graphics	40	37.8	60
65	114	64K	–	–	800 × 600	Graphics	50	48.1	72
65	114	64K	–	–	800 × 600	Graphics	49.5	46.9	75
65	114	64K	–	–	800 × 600	Graphics	56.25	53.7	85
66	110	32K‡	–	–	640 × 480	Graphics	25	31.5	60
66	110	32K‡	–	–	640 × 480	Graphics	31.5	37.9	72
66	110	32K‡	–	–	640 × 480	Graphics	31.5	37.5	75
66	110	32K‡	–	–	640 × 480	Graphics	36	43.3	85
67	113	32K‡	–	–	800 × 600	Graphics	36	35.2	56
67	113	32K‡	–	–	800 × 600	Graphics	40	37.8	60
67	113	32K‡	–	–	800 × 600	Graphics	50	48.1	72
67	113	32K‡	–	–	800 × 600	Graphics	49.5	46.9	75
67	113	32K‡	–	–	800 × 600	Graphics	56.25	53.7	85
68†	116	32K‡	–	–	1024 × 768	Graphics	44.9	35.5	43i†
68	116	32K‡	–	–	1024 × 768	Graphics	65	48.3	60
68	116	32K‡	–	–	1024 × 768	Graphics	75	56	70
68	116	32K‡	–	–	1024 × 768	Graphics	78.7	60	75
68	116	32K‡	–	–	1024 × 768	Graphics	94.5	68.3	85
6C†	106	16/256K	160 × 64	8 × 16	1280 × 1024	Graphics	75	48	43i†
6D†	107	256/256K	160 × 64	8 × 16	1280 × 1024	Graphics	75	48	43i†
6D	107	256/256K	160 × 64	8 × 16	1280 × 1024	Graphics	108	65	60
6D	107	256/256K	160 × 64	8 × 16	1280 × 1024	Graphics	135	80	75
71	112	16M	–	–	640 × 480	Graphics	25	31.5	60
71	112	16M	–	–	640 × 480	Graphics	31.5	37.9	72
71	112	16M	–	–	640 × 480	Graphics	31.5	37.5	75
71	112	16M	–	–	640 × 480	Graphics	36	43.3	85
74†	117	64K	–	–	1024 × 768	Graphics	44.9	35.5	43i†
74	117	64K	–	–	1024 × 768	Graphics	65	48.3	60

Table 3-2. Cirrus Logic Extended Display Modes *(cont.)*

Mode No.	VESA® No.	No. of Colors	Char. × Row	Char. Cell	Screen Format	Display Mode	Pixel Freq. MHz	Horiz. Freq. kHz	Vert. Freq. Hz
74	117	64K	–	–	1024 × 768	Graphics	75	56	70
74	117	64K	–	–	1024 × 768	Graphics	78.7	60	75
74	117	64K	–	–	1024 × 768	Graphics	94.5	68.3	85
75†	11A	64K	–	–	1280 × 1024	Graphics	75	48	43i†
78	115	16M	–	–	800 × 600	Graphics	36	35.2	56
78	115	16M	–	–	800 × 600	Graphics	40	37.8	60
78	115	16M	–	–	800 × 600	Graphics	50	48.1	72
78	115	16M	–	–	800 × 600	Graphics	49.5	46.9	75
78	115	16M	–	–	800 × 600	Graphics	56.25	53.7	85
79	118	16M	–	–	1024 × 768	Graphics	44.9	35.5	43i†
79	118	16M	–	–	1024 × 768	Graphics	65	48.3	60
79	118	16M	–	–	1024 × 768	Graphics	75	56	70
79	118	16M	–	–	1024 × 768	Graphics	78.7	60	75
79	118	16M	–	–	1024 × 768	Graphics	94.5	68.3	85
7B	–	256/256K	–	–	1600 × 1200	Graphics	135	62.5	48i†
7C	–	256/256K	144 × 54	8 × 16	1152 × 864	Graphics	94.5	63.9	70
7C	–	256/256K	144 × 54	8 × 16	1152 × 864	Graphics	108	67.5	75

NOTES:

1) '‡' character indicates 32K Direct-Color/256-Color Mixed mode.
2) †' character indicates Interlaced mode.
3) Some modes and some refresh rates are not supported by the CL-GD5446. Refer to the CL-GD5446 Software Release Kit for the list of display modes supported by the CL-GD5446 BIOS. Also see the inside front cover of this manual.
4) Some modes are not supported by all monitors. The fastest vertical refresh rate for the monitor type selected is automatically used.
5) The CL-GD5446 can support 132-column text modes, not included in the BIOS.

3.2 Configuration Register, CF

When RESET (system power-on reset) is active, the CL-GD5446 samples the levels on several of the Display Memory Data (MD[63:48]) pins. These levels are latched into a write-only Configuration register (CF1). This register controls some fundamental operating modes of the CL-GD5446.

The levels on the Memory Data bus default to a logic '1' during power-on reset because of internal 250-kΩ pull-up resistors. A logic '0' is achieved by installing an external 6.8-kΩ pull-down resistor on the memory data line corresponding to the appropriate bit in the Configuration register. Refer to Appendix B5, "Configuration Notes", in this manual. Table 3-3 summarizes the Configuration register.

Table 3-3. Configuration Register Bits

Memory Data Bit	Pin Number	CF Bits	Level	Description
MD63	157	15	0 1	Enable Pin-Scan test Disable Pin-Scan test
MD62	158	14	0 1	PCI3C[8] = 1 (interrupt claimed) PCI3C[8] = 0 (interrupt not claimed)
MD61	159	13	–	Reserved
MD60	160	12	–	Used with CF5 to define MCLK
MD59	161	11	0 1	Asymmetric DRAM (RAS*/CAS* addressing) Symmetric DRAM (RAS*/CAS* addressing)
MD58	162	10	–	Pull-down resistor required (CAS steering)
MD57	163	9	0 1	7-MCLK RAS* cycle 6-MCLK RAS* cycle
MD56	164	8	_	Used with CF4 to define GPIO, VGA register relocation Revision A only. See Appendix A2 for Revision B silicon.
MD55	170	7	–	Reserved
MD54	171	6	0 1	Feature Connector pins (P[7:0], BLANK#, DCLK) disabled Feature Connector pins normal operation
MD53	172	5	–	Used with CF12 to define MCLK
MD52	173	4	_	Used with CF8 to define GPIO, VGA register relocation Revision A only. See Appendix A2 for Revision B silicon.
MD51	174	3	0 1	Enable PCI14 for GPIO, VGA register relocation Disable PCI14 (no GPIO, VGA register relocation)
MD50, MD49, MD48	141, 142, 143	2, 1, 0	000 001 010 011 100 110 111	Reserved Reserved Reserved Reserved PCI bus Reserved (VESA VL-Bus, reference only) Reserved

4. VGA REGISTER PORT MAP

Table 4-1. VGA Register Port Map

Address	Port	VESA® VL-Bus™ Note[a]
94	POS 102 Access Control (3C3 sleep)	✔
102	POS102 register	✔
3B4	CRT Controller Index (R/W — monochrome)	
3B5	CRT Controller Data (R/W — monochrome)	
3BA	Feature Control (W), Input Status Register 1 (R — monochrome)	
3C0	Attribute Controller Index/Data (Write)	
3C1	Attribute Controller Index/Data (Read)	
3C2	Miscellaneous Output (W), Input Status Register 0 (R)	
3C3	Motherboard Sleep	
3C4	Sequencer Index (R/W)	
3C5	Sequencer Data (R/W)	
3C6	Video DAC Pixel Mask (R/W), Hidden DAC Register (R/W)	
3C7	Pixel Address Read Mode (W), DAC State (R)	
3C8	Pixel Mask Write Mode (R/W)	
3C9	Pixel Data (R/W)	
3CA	Feature Control Readback (R)	
3CC	Miscellaneous Output Readback (R)	
3CE	Graphics Controller Index (R/W)	
3CF	Graphics Controller Data (R/W)	
3D4	CRT Controller Index (R/W — color)	
3D5	CRT Controller Data (R/W — color)	
3DA	Feature Control (W), Input Status Register 1 (R — color)	
46E8	Adapter Sleep	✔

[a] These registers are available only when the CL-GD5446 is configured for VESA VL-Bus. The CL-GD5446 is not available for VESA VL-Bus.

5. REGISTER MAP

All CL-GD5446 registers are listed in Table 5-1. Page numbers in the Page column refer to the register description chapters later in this manual. The registers that have a (V) in the I/O port column are for the VESA VL-Bus and are listed for reference only. Registers at I/O port 3Dxh are at 3Bxh when the CL-GD5446 is programmed for Monochrome mode (MISC[0] = 0).

Table 5-1. CL-GD5446 Registers

Abbreviation	Register Name	I/O Port	Index	MMI/O	Page
MISC	Miscellaneous Output (write only)	3C2h	–	–	4-9
	Miscellaneous Output (read only)	3CCh	–	–	4-9
FC	Feature Control (write only)	3DAh	–	–	4-11
	Feature Control (read only)	3CAh	–	–	4-11
FEAT	Input Status Register 0	3C2h	–	–	4-12
STAT	Input Status Register 1	3DAh	–	–	4-13
–	Pixel Mask	3C6h	–	–	4-14
–	Palette Address (Read mode) (write only)	3C7h	–	–	4-15
–	DAC State (read only)	3C7h	–	–	4-16
–	Palette Address (Write mode)	3C8h	–	–	4-17
–	Palette Data	3C9h	–	–	4-18
HDR	Hidden DAC Register	3C6h	–	–	8-52
PCI00	PCI Device/Vendor ID	00h	–	–	7-3
PCI04	PCI Status/Command	04h	–	–	7-4
PCI08	PCI Class Code	08h	–	–	7-5
PCI10	PCI Display Memory Base Address	10h	–	–	7-6
PCI14	PCI Relocatable I/O and GPIO Base Address (Revision A)	14h	–	–	7-7
PCI14	PCI VGA/BitBLT Register Base Address (Revision B)	14h	–	–	7-8
PCI18	PCI GPIO Base Address (Revision B)	18h	–	–	7-9
PCI2C	PCI Subsystem/Subsystem Vendor ID (Revision B)	2Ch	–	–	7-10
PCI30	PCI Expansion ROM Base Address	30h	–	–	7-11
PCI3C	PCI Interrupt Line	3Ch	–	–	7-12
ARX	Attribute Controller Index	3C0h/3C1h	–	–	4-72
AR0–ARF	Attribute Controller Palette	3C0h/3C1h	00h–0Fh	–	4-73
AR10	Attribute Controller Mode	3C0h/3C1h	10h	–	4-74

Table 5-1. CL-GD5446 Registers *(cont.)*

Abbreviation	Register Name	I/O Port	Index	MMI/O	Page
AR11	Overscan (Border) Color	3C0h/3C1h	11h	–	4-76
AR12	Color Plane Enable	3C0h/3C1h	12h	–	4-77
AR13	Pixel Panning	3C0h/3C1h	13h	–	4-78
AR14	Color Select	3C0h/3C1h	14h	–	4-79
CRX	CRTC Index	3D4h	–	–	4-26
CR0	CRTC Horizontal Total	3D5h	00h	–	4-29
CR1	CRTC Horizontal Display End	3D5h	01h	–	4-30
CR2	CRTC Horizontal Blanking Start	3D5h	02h	–	4-31
CR3	CRTC Horizontal Blanking End	3D5h	03h	–	4-32
CR4	CRTC Horizontal Sync Start	3D5h	04h	–	4-34
CR5	CRTC Horizontal Sync End	3D5h	05h	–	4-35
CR6	CRTC Vertical Total	3D5h	06h	–	4-37
CR7	CRTC Overflow	3D5h	07h	–	4-38
CR8	CRTC Screen A Preset Row-Scan	3D5h	08h	–	4-39
CR9	CRTC Character Cell Height	3D5h	09h	–	4-40
CRA	CRTC Text Cursor Start	3D5h	0Ah	–	4-41
CRB	CRTC Text Cursor End	3D5h	0Bh	–	4-42
CRC	CRTC Screen Start Address High	3D5h	0Ch	–	4-43
CRD	CRTC Screen Start Address Low	3D5h	0Dh	–	4-44
CRE	CRTC Text Cursor Location High	3D5h	0Eh	–	4-45
CRF	CRTC Text Cursor Location Low	3D5h	0Fh	–	4-46
CR10	CRTC Vertical Sync Start	3D5h	10h	–	4-47
CR11	CRTC Vertical Sync End	3D5h	11h	–	4-48
CR12	CRTC Vertical Display End	3D5h	12h	–	4-50
CR13	CRTC Offset (Pitch)	3D5h	13h	–	4-51
CR14	CRTC Underline Row Scanline	3D5h	14h	–	4-52
CR15	CRTC Vertical Blank Start	3D5h	15h	–	4-53
CR16	CRTC Vertical Blank End	3D5h	16h	–	4-54
CR17	CRTC Mode Control	3D5h	17h	–	4-55
CR18	CRTC Line Compare	3D5h	18h	–	4-57
CR19	Interlace End	3D5h	19h	–	8-41
CR1A	Miscellaneous Control	3D5h	1Ah	–	8-42

Table 5-1. CL-GD5446 Registers *(cont.)*

Abbreviation	Register Name	I/O Port	Index	MMI/O	Page
CR1B	Extended Display Controls	3D5h	1Bh	–	8-44
CR1C	Sync Adjust and GENLOCK	3D5h	1Ch	–	8-46
CR1D	Overlay Extended Control	3D5h	1Dh	–	8-48
CR22	Graphics Data Latches Readback (read only)	3D5h	22h	–	4-58
CR24	Attribute Controller Toggle Readback (read only)	3D5h	24h	–	4-59
CR25	Part Status (read only)	3D5h	25h	–	8-50
CR26	Attribute Controller Index Readback (read only)	3D5h	26h	–	4-60
CR27	ID (read only)	3D5h	27h	–	8-51
CR31	Video Window Horizontal Zoom Control	3D5h	31h	–	6-4
CR32	Video Window Vertical Zoom Control	3D5h	32h	–	6-5
CR33	Video Window Horizontal Region 1 Size	3D5h	33h	–	6-6
CR34	Video Window Region 2 Width	3D5h	34h	–	6-7
CR35	Video Window Region 2 Source Data Size	3D5h	35h	–	6-8
CR36	Video Window Horizontal Overflow	3D5h	36h	–	6-9
CR37	Video Window Vertical Start	3D5h	37h	–	6-10
CR38	Video Window Vertical End	3D5h	38h	–	6-11
CR39	Video Window Vertical Overflow	3D5h	39h	–	6-12
CR3A	Video Buffer 1 Start Address Byte 0	3D5h	3Ah	–	6-13
CR3B	Video Buffer 1 Start Address Byte 1	3D5h	3Bh	–	6-13
CR3C	Video Buffer 1 Start Address Byte 2	3D5h	3Ch	–	6-14
CR3D	Video Buffer Address Offset	3D5h	3Dh	–	6-15
CR3E	Video Window Master Control	3D5h	3Eh	–	6-16
CR3F	Miscellaneous Video Control	3D5h	3Fh	–	6-18
CR50	Video Capture Control	3D5h	50h	–	6-20
CR51	Video Capture Data Format	3D5h	51h	–	6-22
CR52	Video Capture Horizontal Data Reduction	3D5h	52h	–	6-23
CR53	Video Capture Vertical Data Reduction	3D5h	53h	–	6-24
CR54	Video Capture Horizontal Delay	3D5h	54h	–	6-25
CR56	Video Capture Vertical Delay	3D5h	56h	–	6-26
CR57	Video Capture Maximum Height	3D5h	57h	–	6-27
CR58	Video Capture Miscellaneous Control	3D5h	58h	–	6-28
CR59	Video Buffer 2 Start Address Byte 0	3D5h	59h	–	6-29

Table 5-1. CL-GD5446 Registers *(cont.)*

Abbreviation	Register Name	I/O Port	Index	MMI/O	Page
CR5A	Video Buffer 2 Start Address Byte 1	3D5h	5Ah	–	6-29
CR5B	Video Window Brightness Adjust	3D5h	5Bh	–	6-30
CR5C	Luminance-Only Capture Control	3D5h	5Ch	–	6-31
CR5D	Video Window Pixel Alignment	3D5h	5Dh	–	6-32
CR5E	Double-Buffer Control	3D5h	5Eh	–	6-33
GRX	Graphics Controller Index	3CEh	–	–	4-61
GR0	Set/Reset / Background Color Byte 0	3CFh	00h	00	4-62
GR1	Set/Reset Enable / Foreground Color Byte 0	3CFh	01h	04	4-63
GR2	Graphics Controller Color Compare	3CFh	02h	–	4-64
GR3	Graphics Controller Data Rotate	3CFh	03h	–	4-65
GR4	Graphics Controller Read Map Select	3CFh	04h	–	4-66
GR5	Graphics Controller Mode	3CFh	05h	–	4-67
GR6	Graphics Controller Miscellaneous	3CFh	06h	–	4-69
GR7	Graphics Controller Color Don’t Care	3CFh	07h	–	4-70
GR8	Graphics Controller Bit Mask	3CFh	08h	–	4-71
GR9	Offset Register 0	3CFh	09h	–	8-26
GRA	Offset Register 1	3CFh	0Ah	–	8-28
GRB	Graphics Controller Mode Extensions	3CFh	0Bh	–	8-29
GRC	Color Key/Chroma Key Compare	3CFh	0Ch	–	8-31
GRD	Color Key/Mask/Chroma Key	3CFh	0Dh	–	8-32
GRE	Power Management	3CFh	0Eh	–	8-33
GR10	Background Color Byte 1	3CFh	10h	01	5-3
GR11	Foreground Color Byte 1	3CFh	11h	05	5-3
GR12	Background Color Byte 2	3CFh	12h	02	5-3
GR13	Foreground Color Byte 2	3CFh	13h	06	5-3
GR14	Background Color Byte 3	3CFh	14h	03	5-3
GR15	Foreground Color Byte 3	3CFh	15h	07	5-3
GR16	Active Display Line Readback Byte 0	3CFh	16h	–	8-35
GR17	Active Display Line Readback Byte 1	3CFh	17h	–	8-36
GR18	Extended DRAM Control	3CFh	18h	–	8-37
GR19	GPIO Port Configuration	3CFh	19h	–	8-39
GR1A	Scratch Pad 4	3CFh	1Ah	–	8-40

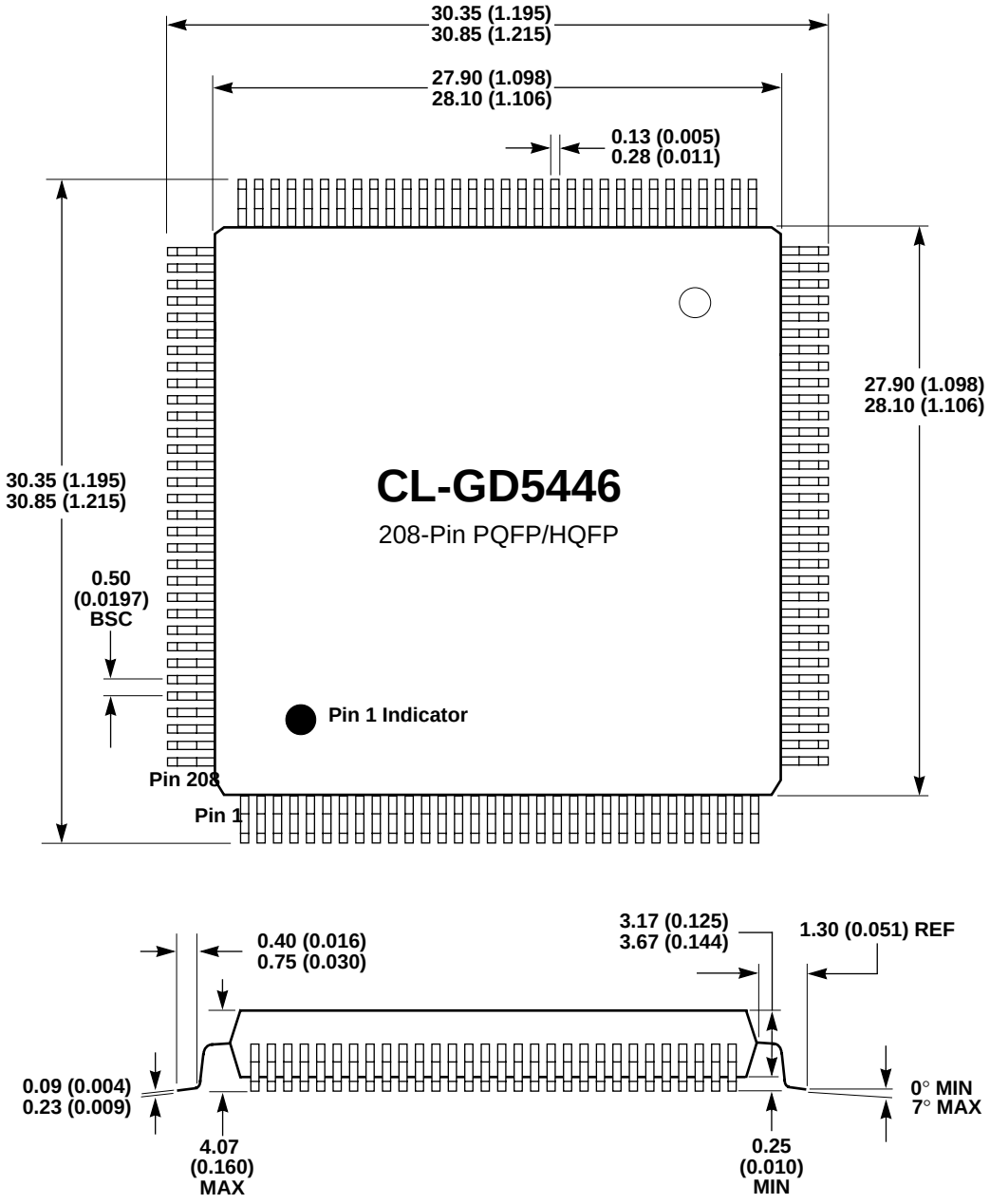
Table 5-1. CL-GD5446 Registers *(cont.)*

Abbreviation	Register Name	I/O Port	Index	MMI/O	Page
GR1B	Scratch Pad 5	3CFh	1Bh	–	8-40
GR1C	Chroma Key – U Minimum/Green Minimum	3CFh	1Ch	–	6-35
GR1D	Chroma Key – U Maximum/Green Maximum	3CFh	1Dh	–	6-35
GR1E	Chroma Key – V Minimum/Blue Minimum	3CFh	1Eh	–	6-35
GR1F	Chroma Key – V Maximum/Blue Maximum	3CFh	1Fh	–	6-35
GR20	BLT Width Byte 0	3CFh	20h	08	5-4
GR21	BLT Width Byte 1	3CFh	21h	09	5-4
GR22	BLT Height Byte 0	3CFh	22h	0A	5-5
GR23	BLT Height Byte 1	3CFh	23h	0B	5-5
GR24	BLT Destination Pitch Byte 0	3CFh	24h	0C	5-6
GR25	BLT Destination Pitch Byte 1	3CFh	25h	0D	5-6
GR26	BLT Source Pitch Byte 0	3CFh	26h	0E	5-7
GR27	BLT Source Pitch Byte 1	3CFh	27h	0F	5-7
GR28	BLT Destination Start Byte 0	3CFh	28h	10	5-8
GR29	BLT Destination Start Byte 1	3CFh	29h	11	5-8
GR2A	BLT Destination Start Byte 2	3CFh	2Ah	12	5-8
GR2C	BLT Source Start Byte 0	3CFh	2Ch	14	5-9
GR2D	BLT Source Start Byte 1	3CFh	2Dh	15	5-9
GR2E	BLT Source Start Byte 2	3CFh	2Eh	16	5-9
GR2F	BLT Destination Left-Side Clipping	3CFh	2Fh	17	5-10
GR30	BLT Mode	3CFh	30h	18	5-11
GR31	BLT Start/Status	3CFh	31h	40	5-13
GR32	BLT ROP (Raster Operation)	3CFh	32h	1A	5-15
GR33	BLT Mode Extensions	3CFh	33h	1B	5-17
GR34	Transparent BLT Key Color Byte 0	3CFh	34h	1C	5-18
GR35	Transparent BLT Key Color Byte 1	3CFh	35h	1D	5-18
SRX	Sequencer Index	3C4h	–	–	4-19
SR0	Sequencer Reset	3C5h	00h	–	4-20
SR1	Sequencer Clocking Mode	3C5h	01h	–	4-21
SR2	Sequencer Plane Mask	3C5h	02h	–	4-22
SR3	Sequencer Character Map Select	3C5h	03h	–	4-23
SR4	Sequencer Memory Mode	3C5h	04h	–	4-25

Table 5-1. CL-GD5446 Registers *(cont.)*

Abbreviation	Register Name	I/O Port	Index	MMI/O	Page
SR6	Key	3C5h	06h	–	8-4
SR7	Extended Sequencer Mode	3C5h	07h	–	8-5
SR8	DDC2B/EEPROM Control	3C5h	08h	–	8-7
SR9	Scratch Pad 0	3C5h	09h	–	8-9
SRA	Scratch Pad 1	3C5h	0Ah	–	8-9
SRB	VCLK0 Numerator	3C5h	0Bh	–	8-10
SRC	VCLK1 Numerator	3C5h	0Ch	–	8-10
SRD	VCLK2 Numerator	3C5h	0Dh	–	8-10
SRE	VCLK3 Numerator	3C5h	0Eh	–	8-10
SRF	DRAM Control	3C5h	0Fh	–	8-11
SR10	Graphics Cursor X Position	3C5h	10h	–	8-13
SR11	Graphics Cursor Y Position	3C5h	11h	–	8-14
SR12	Graphics Cursor Attributes	3C5h	12h	–	8-15
SR13	Graphics Cursor Pattern Address Offset	3C5h	13h	–	8-16
SR14	Scratch Pad 2	3C5h	14h	–	8-17
SR15	Scratch Pad 3	3C5h	15h	–	8-17
SR16	Display FIFO Threshold Control	3C5h	16h	–	8-18
SR17	Configuration Readback and Extended Control	3C5h	17h	–	8-19
SR18	Signature Generator Control	3C5h	18h	–	8-20
SR19	Signature Generator Result Low Byte	3C5h	19h	–	8-22
SR1A	Signature Generator Result High Byte	3C5h	1Ah	–	8-23
SR1B	VCLK0 Denominator and Post Scalar	3C5h	1Bh	–	8-24
SR1C	VCLK1 Denominator and Post Scalar	3C5h	1Ch	–	8-24
SR1D	VCLK2 Denominator and Post Scalar	3C5h	1Dh	–	8-24
SR1E	VCLK3 Denominator and Post Scalar	3C5h	1Eh	–	8-24
SR1F	MCLK Select	3C5h	1Fh	–	8-25
POS94	POS102 Access Control	94h (V)	–	–	4-5
POS102	POS102	102h (V)	–	–	4-6
VSSM	3C3 (Planar) Sleep Address	3C3h (V)	–	–	4-7
VSSM	46E8 (Adapter) Sleep Address	46E8h (V)	–	–	4-8

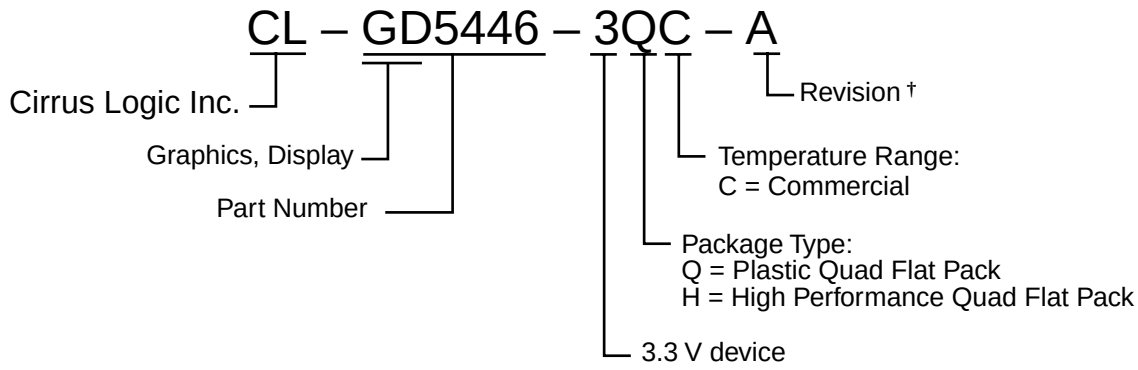
6. PACKAGE SPECIFICATIONS



NOTES:

1) Dimensions are in millimeters (inches), and controlling dimension is millimeter.
2) Drawing above does not reflect exact package pin count.
3) Before beginning any new design with this device, please contact Cirrus Logic for the latest package information.
4) HQFP is a high-performance QFP with an exposed or unexposed heat sink.

7. ORDERING INFORMATION EXAMPLE



† Contact Cirrus Logic. for up-to-date information on revisions.

VGA Core Registers

4. VGA CORE REGISTERS

The VGA core registers are summarized in Table 4-1. These are registers defined in the IBM VGA. The first four are not accessible when the CL-GD5446 is configured for PCI; these are indicated with a (V) in the I/O Port column.

Table 4-1. VGA Core Registers Quick Reference

Abbreviation	Register Name	I/O Port	Index	Page
POS94	POS102 Access Control	94h (V)	–	4-5
POS102	POS102	102h (V)	–	4-6
VSSM	3c3 (Planar) Sleep Address	3C3h (V)	–	4-7
	46E8 (Adapter) Sleep Address	46E8h (V)	–	4-8
MISC	Miscellaneous Output (write only)	3C2h	–	4-9
	Miscellaneous Output (read only)	3CCh	–	4-9
FC	Feature Control (write only)	3DAh	–	4-11
	Feature Control (read only)	3CAh	–	4-11
FEAT	Input Status Register 0 (read only)	3C2h	–	4-12
STAT	Input Status Register 1 (read only)	3DAh	–	4-13
	Pixel Mask	3C6h	–	4-14
	Palette Address (Read mode) (write only)	3C7h	–	4-15
	DAC State (read only)	3C7h	–	4-16
	Palette Address (Write mode)	3C8h	–	4-17
	Palette Data	3C9h	–	4-18
SRX	Sequencer Index	3C4h	–	4-19
SR0	Sequencer Reset	3C5h	00h	4-20
SR1	Sequencer Clocking Mode	3C5h	01h	4-21
SR2	Sequencer Plane Mask	3C5h	02h	4-22
SR3	Sequencer Character Map Select	3C5h	03h	4-23
SR4	Sequencer Memory Mode	3C5h	04h	4-25
CRX	CRTC Index	3D4h[a]	–	4-26
CR0	CRTC Horizontal Total	3D5h	00h	4-29
CR1	CRTC Horizontal Display End	3D5h	01h	4-30
CR2	CRTC Horizontal Blanking Start	3D5h	02h	4-31
CR3	CRTC Horizontal Blanking End	3D5h	03h	4-32

Table 4-1. VGA Core Registers Quick Reference *(cont.)*

Abbreviation	Register Name	I/O Port	Index	Page
CR4	CRTC Horizontal Sync Start	3D5h	04h	4-34
CR5	CRTC Horizontal Sync End	3D5h	05h	4-35
CR6	CRTC Vertical Total	3D5h	06h	4-37
CR7	CRTC Overflow	3D5h	07h	4-38
CR8	CRTC Screen A Preset Row-Scan	3D5h	08h	4-39
CR9	CRTC Character Cell Height	3D5h	09h	4-40
CRA	CRTC Text Cursor Start	3D5h	0Ah	4-41
CRB	CRTC Text Cursor End	3D5h	0Bh	4-42
CRC	CRTC Screen Start Address High	3D5h	0Ch	4-43
CRD	CRTC Screen Start Address Low	3D5h	0Dh	4-44
CRE	CRTC Text Cursor Location High	3D5h	0Eh	4-45
CRF	CRTC Text Cursor Location Low	3D5h	0Fh	4-46
CR10	CRTC Vertical Sync Start	3D5h	10h	4-47
CR11	CRTC Vertical Sync End	3D5h	11h	4-48
CR12	CRTC Vertical Display End	3D5h	12h	4-50
CR13	CRTC Offset (Pitch)	3D5h	13h	4-51
CR14	CRTC Underline Row Scanline	3D5h	14h	4-52
CR15	CRTC Vertical Blanking Start	3D5h	15h	4-53
CR16	CRTC Vertical Blanking End	3D5h	16h	4-54
CR17	CRTC Mode Control	3D5h	17h	4-55
CR18	CRTC Line Compare	3D5h	18h	4-57
CR22	Graphics Data Latches Readback (read only)	3D5h	22h	4-58
CR24	Attribute Controller Toggle Readback (read only)	3D5h	24h	4-59
CR26	Attribute Controller Index Readback (read only)	3D5h	26h	4-60
GRX	Graphics Controller Index	3CEh	–	4-61
GR0	Graphic Controller Set/Reset	3CFh	0h	4-62
GR1	Graphics Controller Set/Reset Enable	3CFh	1h	4-63
GR2	Graphics Controller Color Compare	3CFh	2h	4-64
GR3	Graphics Controller Data Rotate	3CFh	3h	4-65
GR4	Graphics Controller Read Map Select	3CFh	4h	4-66
GR5	Graphics Controller Mode	3CFh	5h	4-67

Table 4-1. VGA Core Registers Quick Reference *(cont.)*

Abbreviation	Register Name	I/O Port	Index	Page
GR6	Graphics Controller Miscellaneous	3CFh	6h	4-69
GR7	Graphics Controller Color Don't Care	3CFh	7h	4-70
GR8	Graphics Controller Bit Mask	3CFh	8h	4-71
ARX	Attribute Controller Index	3C0/3C1h	–	4-72
AR0–ARF	Attribute Controller Palette	3C0/3C1h	0h–Fh	4-73
AR10	Attribute Controller Mode	3C0/3C1h	10h	4-74
AR11	Overscan (Border) Color	3C0/3C1h	11h	4-76
AR12	Color Plane Enable	3C0/3C1h	12h	4-77
AR13	Pixel Panning	3C0/3C1h	13h	4-78
AR14	Color Select	3C0/3C1h	14h	4-79

[a] If the CL-GD5446 is programmed for Monochrome mode (MISC[0] = 0), registers at 3Dxh are at 3Bxh.

4.1 POS94: POS102 Access Control

I/O Port Address: (VESA VL-Bus): 94h
Index: –
Size (bits): 8
Access Type: Write only

Bit	Description	Reset State
7:6	Reserved	
5	POS102 Access	1
4:0	Reserved	

This register contains the enable bit for POS102. This was originally an IBM PS/2 planar register and is retained for software compatibility.

This register is accessible only if CL-GD5446 is configured for 3C3 (planar) sleep and the VESA VL-Bus. This port is not accessible when the device is configured for the PCI bus. This port is not readable. When CL-GD5446 is configured for the VESA VL-Bus, it responds to writes to this register by latching the data, but does not generate LDEV# or LRDY#.

NOTE: The CL-GD5446 is marketed as PCI-compatible only. VESA VL-Bus functionality is not guaranteed.

Bit	Description
7:6	**Reserved**
5	**POS102 Access:** If the CL-GD5446 is configured for 3C3 sleep and the VESA VL-Bus, this bit controls access to POS register 102. If this bit is '0', POS102 is accessible; if it is '1', POS102 is not accessible. In addition, if this bit is '0', the Display Subsystem Enable in 3C3 is overridden, and CL-GD5446 remains in Sleep mode.
4:0	**Reserved**

4.2 POS102: POS102

I/O Port Address: (VESA VL-Bus): 102h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description	Reset State
7:1	Reserved	
0	Display Subsystem Enable	0

This register contains a Display Subsystem Enable bit. This port is not accessible when the device is configured for the PCI bus. This register is accessible according to the following table and only if the CL-GD5446 is configured for the VESA VL-Bus.

NOTE: The CL-GD5446 is marketed as PCI-compatible only. VESA VL-Bus functionality is not guaranteed.

Sleep Address	102 Register Accessibility
46E8 (Adapter)	46E8 [4] = 1
3C3 (Motherboard)	POS94[5] = 0

Bit	Description
7:1	**Reserved**
0	**Display Subsystem Enable:** If this bit is '1', the CL-GD5446 is enabled and operates normally if the VSE bit in 46E8 or 3C3 is also '1'. If this bit is '0', the CL-GD5446 is disabled. It does not respond to any I/O accesses except those to POS94 and POS102 or to any memory accesses except those to the BIOS ROM. This bit has the same effect as 3C3[0] or 46E8[3], and is provided for compatibility with software written for certain models of IBM PS/2.

4.3 VSSM: 3C3 (Planar) Sleep Address

I/O Port Address: (VESA VL-Bus): 3C3h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description	Reset State
7:1	Reserved	
0	Display Subsystem Enable	

This is the Sleep register when CL-GD5446 is configured for 3C3 (planar) sleep. This port is not accessible when the device is configured for the PCI bus. This register is read/write.

NOTE: The CL-GD5446 is marketed as PCI-compatible only. VESA VL-Bus functionality is not guaranteed.

Bit	Description
7:1	**Reserved**
0	**Display Subsystem Enable:** If the CL-GD5446 is configured for 46E8 sleep, this register is not accessible and this bit is a don't care. If the CL-GD5446 is configured for 3C3 sleep and for the VESA VL-Bus, this register is always accessible. If this bit is '1', register POS102[0] is '1', and register 94[5] is '1' the CL-GD5446 is enabled and operates normally. If this bit is '0', the CL-GD5446 is disabled; it does not respond to any I/O accesses, except those addressed to 3C3 or 94. It does not respond to any accesses to display memory, but responds normally to BIOS accesses. The display continues (if enabled) regardless of the state of this bit.

4.4 VSSM: 46E8 (Adapter) Sleep Address

I/O Port Address: (VESA VL-Bus): 46E8h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description	Reset State
7:5	Reserved	
4	Setup	0
3	Display Subsystem Enable	0
2:0	Reserved	

This is the Sleep Address register for an adapter VGA and can be accessed only if the CL-GD5446 is configured for 46E8 (adapter) Sleep Address. This port is not accessible when the device is configured for the PCI bus.

NOTE: The CL-GD5446 is marketed as PCI-compatible only. VESA VL-Bus functionality is not guaranteed.

Bit	Description
7:5	**Reserved**
4	**Setup:** If this bit is '1', the CL-GD5446 is in Setup mode. In Setup mode, the resister at I/O Address 102 is accessible, and the register at 46E8 is accessible. The device responds normally to accesses to BIOS, but does not respond to accesses to display memory. If this bit is '0', the device is not in Setup mode and operates normally.
3	**Display Subsystem Enable:** If the CL-GD5446 is not configured for 46E8 Sleep Address, this bit cannot be accessed. If this bit is '1', the CL-GD5446 is enabled and operates normally. If this bit is '0', the CL-GD5446 is disabled; it does not respond to any I/O accesses except those addressed to 46E8 and 102. It does not respond to any accesses to display memory, but responds normally to BIOS accesses. The display continues (if enabled) regardless of the state of this bit.
2:0	**Reserved**

4.5 MISC: Miscellaneous Output

I/O Port Address: 3C2h (write); 3CCh (read)
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Sync Polarity
6	Horizontal Sync Polarity
5	Page Select
4	Reserved
3	Clock Select [1]
2	Clock Select [0]
1	Enable Display Memory
0	CRTC I/O Address

This is one of the standard VGA registers.

Bit **Description**

7 **Vertical Sync Polarity:** If this bit is '0', the Vertical Sync is a low signal, going high to indicate the beginning of sync time. If this bit is '1', the Vertical Sync is a high signal, going low to indicate the beginning of sync time.

See the description of register GRE for information regarding static sync signals.

6 **Horizontal Sync Polarity:** If this bit is '0', the Horizontal Sync is a low signal, going high to indicate the beginning of sync time. If this bit is '1', the Horizontal Sync is a high signal, going low to indicate the beginning of sync time.

See the description of register GRE for information regarding static sync signals.

For some monitors, the polarities of Vertical and Horizontal Sync indicates the number of scanlines per frame as summarized below:

MISC[7]	MISC[6]	Vertical Size
0 (+)	0 (+)	Reserved
0 (+)	1 (–)	400
1 (–)	0 (+)	350
1 (–)	1 (–)	480

4.5 MISC: Miscellaneous Output *(cont.)*

Bit **Description**

5 **Page Select:** This bit affects the meaning of the least-significant bit of the display memory address when in Even/Odd modes (SR4[2] = 1). If this bit is '0', only odd memory locations are selected. If this bit is '1', only even memory locations are selected.

NOTE: This bit is effective in modes 6, D, E, 11, and 12. This bit is ignored if Chain (GR6[1]) or Chain4 (SR4[3]) are enabled.

4 **Reserved**

3:2 **Clock Select [1:0]:** This 2-bit field selects one of the VCLK frequencies, as shown in the following table:

EDCLK#	Clock Select [1:0]	VCLK Source	Default Frequency
1	00	VCLK0	25.180 MHz
1	01	VCLK1	28.325 MHz
1	10	VCLK2	41.165 MHz
1	11	VCLK3	36.082 MHz
0	1X	DCLK pin (DAC and CRTC counters)	
0	0X	DCLK pin (DAC only)	

NOTE: Refer to Chapter 9, "Programming Notes", for programming VCLK frequencies other than those listed in the table above.

1 **Enable Display Memory:** If this bit is '0', the CL-GD5446 does not respond to any access to display memory. If this bit is '1', the CL-GD5446 responds normally to accesses to display memory.

0 **CRTC I/O Address:** This bit selects I/O addresses for either Monochrome or Color mode. The affected addresses are summarized in the table below:

MISC[0]	ISR/FC	CRTC Index	CRTC Data	Mode
0	3BA	3B4	3B5	Monochrome
1	3DA	3D4	3D5	Color

4.6 FC: Feature Control

I/O Port Address: 3DAh (write), 3CAh (read)
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description	Reset State
7:4	Reserved	
3	VSYNC Control	0
2:0	Reserved	

This is one of the original IBM VGA registers.

Bit	Description
7:4	**Reserved**
3	**VSYNC Control:** If this bit is '1', VSYNC is logically OR'ed with Display Enable (an internal signal) prior to going to the VSYNC pin. If this bit is '0', VSYNC is unchanged.
2:0	**Reserved**

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.7 FEAT: Input Status Register 0

I/O Port Address: 3C2h
Index: –
Size (bits): 8
Access Type: Read only

Bit	Description
7	VGA Interrupt Pending
6:5	Reserved
4	DAC Sensing
3:0	Reserved

This is one of the registers in the IBM VGA. This register is read only.

Bit	Description
7	**VGA Interrupt Pending:** If this bit is '1', an interrupt request is pending. If this bit is '0', no interrupt is pending. See the description of register CR11 for more information regarding the CL-GD5446 interrupt system. Additional information is in Chapter 9, "Programming Notes".
6:5	**Reserved**
4	**DAC Sensing:** This read-only bit is used by the Cirrus Logic BIOS to determine whether a monitor is connected and, if so, whether it is color or monochrome.
3:0	**Reserved**

4.8 STAT: Input Status Register 1

I/O Port Address: 3DAh
Index: –
Size (bits): 8
Access Type: Read only

Bit	Description
7:6	Reserved
5	Diagnostic [1]
4	Diagnostic [0]
3	Vertical Retrace
2:1	Reserved
0	Display Enable

This read-only register contains some VGA status bits.

Bit **Description**

7:6 **Reserved**

5:4 **Diagnostic [1:0]:** These bits follow two of eight outputs of the attribute controller. The selection is made according to AR12[5:4] (Color Plane Enable register) as indicated in the following table:

AR12[5]	AR12[4]	STAT[5]	STAT[4]
0	0	P[2]	P[0]
0	1	P[5]	P[4]
1	0	P[3]	P[1]
1	1	P[7]	P[6]

If CR1A[3:2] are programmed for overlay, the inputs on P[7:0] can be read on these bits.

3 **Vertical Retrace:** If this bit is ‘1’, a vertical retrace is in progress.

2:1 **Reserved**

0 **Display Enable:** If this bit is read as ‘0’, data is being serialized and displayed. If this bit is read as ‘1’, vertical or horizontal blanking is active.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.9 Pixel Mask

I/O Port Address: 3C6h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Pixel Mask [7]
6	Pixel Mask [6]
5	Pixel Mask [5]
4	Pixel Mask [4]
3	Pixel Mask [3]
2	Pixel Mask [2]
1	Pixel Mask [1]
0	Pixel Mask [0]

The bits in this register form the pixel mask for the palette DAC. All bits in this register are typically programmed to ‘1’ by the Cirrus Logic BIOS.

Bit	Description
7:0	**Pixel Mask [7:0]:** This field is the pixel mask for the palette DAC. If a bit in this field is ‘0’, the corresponding bit in the pixel data is ignored when looking up an entry in the LUT.

4.10 Palette Address (Read Mode, Write only)

I/O Port Address: 3C7h
Index: –
Size (bits): 8
Access Type: Write only

Bit	Description
7	Palette Address (Read Mode) [7]
6	Palette Address (Read Mode) [6]
5	Palette Address (Read Mode) [5]
4	Palette Address (Read Mode) [4]
3	Palette Address (Read Mode) [3]
2	Palette Address (Read Mode) [2]
1	Palette Address (Read Mode) [1]
0	Palette Address (Read Mode) [0]

The bits in this write-only register specify the address (Read mode) for the palette. This is used to specify the entry in the LUT to be read.

Bit	Description
7:0	**Palette Address (Read Mode) [7:0]:** This field is the address (Read mode) for the LUT. This address is incremented after every third read of the Pixel Data register.

4.11 DAC State (Read only)

I/O Port Address: 3C7h
Index: –
Size (bits): 8
Access Type: Read only

Bit	Description
7:2	Reserved
1	DAC State [1]
0	DAC State [0]

The bits in this read-only register indicate whether a read or a write occurred last to the LUT.

Bit	Description
7:2	**Reserved**
1:0	**DAC State [1:0]:** This field indicates whether the Palette Address (Read) register or the Palette Address (Write) register was accessed last. The two bits must always have the same value. When the state of these bits is ‘00’, a write operation is in progress. When the state of these bits is ‘11’, a read operation is in progress.

4.12 Palette Address (Write Mode)

I/O Port Address: 3C8h
Index: –
Size (bits): 8
Access Type: Write only

Bit	Description
7	Palette Address (Write Mode) [7]
6	Palette Address (Write Mode) [6]
5	Palette Address (Write Mode) [5]
4	Palette Address (Write Mode) [4]
3	Palette Address (Write Mode) [3]
2	Palette Address (Write Mode) [2]
1	Palette Address (Write Mode) [1]
0	Palette Address (Write Mode) [0]

The bits in this register form the address (Write mode) for the palette DAC. This specifies the entry in the LUT to be written.

Bit	Description
7:0	**Palette Address (Write Mode) [7:0]:** This field is the Palette Address (Write mode) for the LUT. This address is incremented after every third write to the Pixel Data register.

4.13 Palette Data

I/O Port Address: 3C9h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Pixel Data [7]
6	Pixel Data [6]
5	Pixel Data [5]
4	Pixel Data [4]
3	Pixel Data [3]
2	Pixel Data [2]
1	Pixel Data [1]
0	Pixel Data [0]

This is the Pixel Data register for the palette DAC.

Bit	Description
7:0	**Pixel Data [7:0]:** This field is the Pixel Data for the palette DAC. This is a read/write register. Prior to writing to this register, 3C8h is written with the first or only palette address. Then three values, corresponding to red, green, and blue are written to this address. Following the third write, the values are transferred to the LUT, and the Palette Address is incremented in case values for the next address that are to be written. Prior to reading from this register, 3C7h is written with the first or only palette address. Then three values, corresponding to red, green, and blue, can be read from this address. Following the third read, the Palette Address is incremented in case the values for the next address that are to be read.

4.14 SRX: Sequencer Index

I/O Port Address: 3C4h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7:5	Reserved
4	Sequencer Index [4]
3	Sequencer Index [3]
2	Sequencer Index [2]
1	Sequencer Index [1]
0	Sequencer Index [0]

This register specifies the register in the sequencer block to be accessed by the next I/O read or write to Address 3C5. Indices greater than five point to the registers that are defined in Chapter 8, “Miscellaneous Extension Registers”.

Bit	Description
7:5	**Reserved**
4:0	**Sequencer Index [4:0]:** This field selects the register to be accessed with the next I/O read or I/O write to 3C5h.

4.15 SR0: Sequencer Reset

I/O Port Address: 3C5h
Index: 00h
Size (bits): 8
Access Type: Read/write

Bit	Description	Reset State
7:2	Reserved	
1	Synchronous Reset	1
0	Asynchronous Reset	1

This register resets the sequencer. These bits are for compatibility only and never need to be used in the CL-GD5446.

Bit	Description
7:2	**Reserved**
1	**Synchronous Reset:** If this bit is '0', the sequencer clears and halts. This disables screen refresh and display memory refresh. If this bit is '1' and SR0[0] is '1', the sequencer operates normally.
0	**Asynchronous Reset:** If this bit is '0', the sequencer clears and halts and register SR3 is cleared. If this bit is '1' and SR0[1] is '1', the sequencer operates normally.

4.16 SR1: Sequencer Clocking Mode

I/O Port Address: 3C5h
Index: 01h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Full Bandwidth
4	Shift and Load 32
3	Dot Clock ÷ 2
2	Shift and Load 16
1	Reserved
0	8/9 Dot Clock

This register controls miscellaneous functions in the sequencer.

Bit	Description
7:6	**Reserved**
5	**Full Bandwidth:** If this bit is '1', screen refresh stops. This allows the CPU to use nearly 100% of the display memory bandwidth. HSYNC and VSYNC continue normally, and display memory refresh also continues. BLANK# goes active and stays active. If this bit is '0', the CL-GD5446 operates normally.
4	**Shift and Load 32:** This bit in conjunction with SR1[2], controls the Display Data Shifters in the graphics controller according to the following table:
3	**Dot Clock ÷ 2:** If this bit is '1', VCLK is divided by two to generate DCLK. This is for low-resolution display modes (such as, 0, 1, 4, 5, and D). If this bit is '0', the Master Clock is not divided by two.
2	**Shift and Load 16:** Refer to the description of SR1[4].
1	**Reserved**
0	**8/9 Dot Clock:** If this bit is '1', DCLK is divided by eight to generate the character clock. If this bit is '0', DCLK is divided by nine to generate the character clock. This is used for 720 × 350 and 720 × 400 resolution AN (alphanumeric) modes.

SR1[4]	SR1[2]	Data Shifters Loaded
0	0	Every character clock
0	1	Every second character clock
1	X	Every fourth character clock

4.17 SR2: Sequencer Plane Mask

I/O Port Address: 3C5h
Index: 02h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Map 3 Enable [3]
2	Map 2 Enable [2]
1	Map 1 Enable [1]
0	Map 0 Enable [0]

This register enables/disables writing to the four planes of display memory.

Bit	Description
7:4	**Reserved**
3:0	**Map Enable [3:0]:** These four bits individually control whether Bit Planes 3:0 are written with Write modes 0 through 3. If GRB[2] is set to '1', these bits can also control which bytes are written by the BitBLT engine.

4.18 SR3: Sequencer Character Map Select

I/O Port Address: 3C5h
Index: 03h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Secondary Map Select [0]
4	Primary Map Select [0]
3	Secondary Map Select [2]
2	Secondary Map Select [1]
1	Primary Map Select [2]
0	Primary Map Select [1]

This register specifies the primary and secondary character sets (fonts). This is used only for Text modes.

Bit	Description
7:6	**Reserved**
5, 3:2	**Secondary Map Select:** These three bits select the Secondary Character Map according to the following table:

SR3[5]	SR3[3]	SR3[2]	Map	Offset
0	0	0	0	0K
0	0	1	1	16K
0	1	0	2	32K
0	1	1	3	48K
1	0	0	4	8K
1	0	1	5	24K
1	1	0	6	40K
1	1	1	7	56K

4.18 SR3: Sequencer Character Map Select *(cont.)*

Bit **Description**

4, 1:0 **Primary Map Select:** These three bits select the Primary Character Map according to the following table:

SR3[4]	SR3[1]	SR3[0]	Map	Offset
0	0	0	0	0K
0	0	1	1	16K
0	1	0	2	32K
0	1	1	3	48K
1	0	0	4	8K
1	0	1	5	24K
1	1	0	6	40K
1	1	1	7	56K

NOTES:

1) In Text modes, the ASCII text character is stored in Plane 0, the attribute is stored in Plane 1, and the font is stored in Plane 2.

2) Bit 3 of the attribute byte normally controls the intensity of the foreground color. This bit may be redefined to be a switch between character sets, allowing 512 displayable characters. This switch is enabled whenever there is a difference between the values of the Primary Map Select and Secondary Map Select, and SR4[1] is '1'.

3) The format of the Plane 2 Font Address bits 15:0 is:

   F0 F1 F2 C7 C6 C5 C4 C3 C2 C1 C0 R4 R3 R2 R1 R0,

   where F[2:0] is the Character Map Select, C[7:0] is the ASCII character, and R[4:0] is the Character Row (scanline in the character cell).

4.19 SR4: Sequencer Memory Mode

I/O Port Address: 3C5h
Index: 04h
Size (bits): 8
Access Type: Read/Write

Bit	Description
7:4	Reserved
3	Chain-4
2	Odd/Even
1	Extended Memory
0	Reserved

This register controls miscellaneous functions in the sequencer.

Bit	Description
7:4	**Reserved**
3	**Chain-4:** If this bit is '1', A0 provides Plane Select bit 0, and A1 provides Plane Select bit 1. This has an similar effect to Odd/Even mode, except that both A1 and A0 are used. This bit takes priority over SR4[2] (Odd/Even) and GR5[4]. There is not a separate bit in the graphics controller to select Chain-4 addressing, as is the case with the Odd/Even bit. The Graphics Controller Read Map register (GR4) is ignored when this bit is '1'. This bit also modifies the meaning of SR7[0].
2	**Odd/Even:** If this bit is '0', the sequencer is placed in Odd/Even mode. Even CPU addresses access Planes 0 and 2; odd CPU addresses access Planes 1 and 3. This bit must be '0' for Text modes. The value of this bit must track GR5[4] (Odd/Even); the values are opposite. This bit also modifies the meaning of SR7[0].
1	**Extended Memory:** If this bit is '0', the effective memory size is 64K, regardless of the memory actually installed. EGA modes require this to be the case. If this bit is '1', the effective memory size is equal to the actual memory installed.
0	**Reserved**

4.20 CRX: CRTC Index

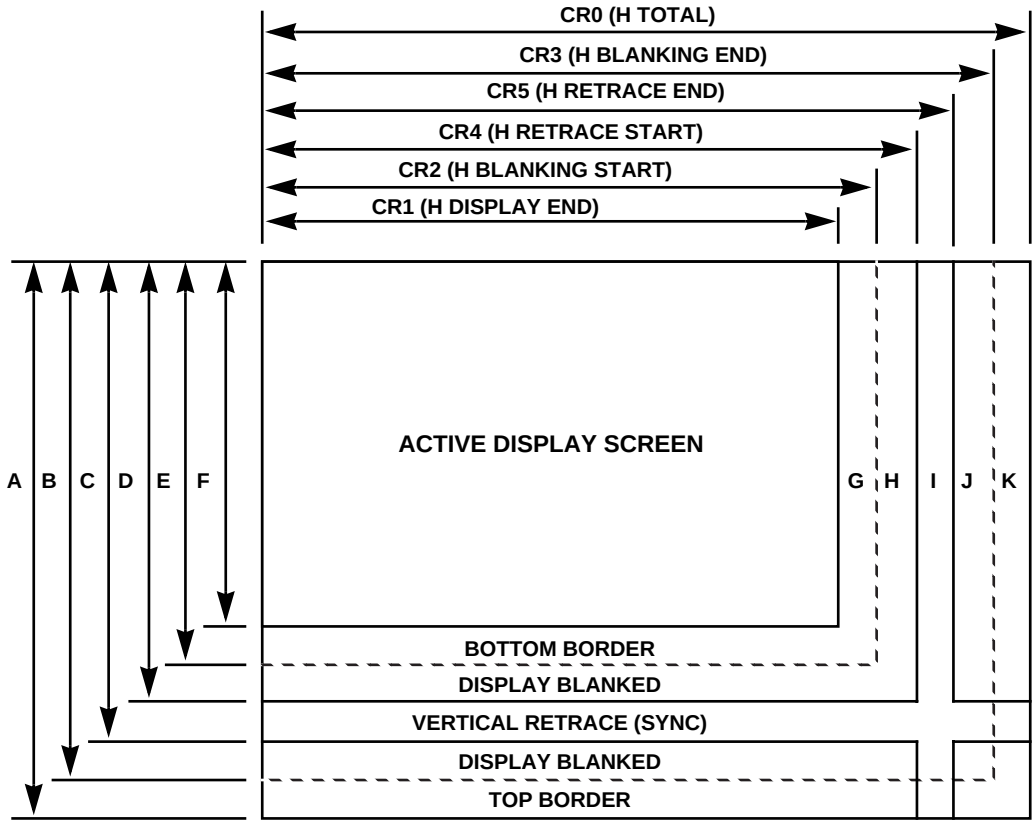
I/O Port Address: 3D4h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	CRTC Index [5]
4	CRTC Index [4]
3	CRTC Index [3]
2	CRTC Index [2]
1	CRTC Index [1]
0	CRTC Index [0]

This register specifies the register in the CRTC block accessed by the next I/O read or I/O write to Address 3D5.

Bit	Description
7:6	**Reserved**
5:0	**CRTC Index [5:0]:** This value points to the register to be accessed in the next I/O read or I/O write to Address 3D5. **NOTE:** Registers above 18 were never documented by IBM.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.



A - CR6 (V TOTAL)
B - CR16 (V BLANKING END)
C - CR11 (V RETRACE END)
D - CR10 (V RETRACE START)
E - CR15 (V BLANKING START)
F - CR12 (V DISPLAY END)
G - RIGHT BORDER
H - DISPLAY BLANKED
I - HORIZONTAL RETRACE (SYNC)
J - DISPLAY BLANKED
K - LEFT BORDER

Figure 4-1. CRTC Timing Registers

The Extension and Overflow bits are organized by parameter and bit position in Table 4-2.

Table 4-2. Summary of CRTC Timing Registers

Parameter	9	8	7	6	5	4:0
H Total			CR0[7]	CR0[6]	CR0[5]	CR0[4:0]
H Display End			CR1[7]	CR1[6]	CR1[5]	CR1[4:0]
H Blank Start			CR2[7]	CR2[6]	CR2[5]	CR2[4:0]
H Blank End			CR1A[5]	CR1A[4]	CR5[7]	CR3[4:0]
H Sync Start			CR4[7]	CR4[6]	CR4[5]	CR4[4:0]
H Sync End						CR5[4:0]
V Total	CR7[5]	CR7[0]	CR6[7]	CR6[6]	CR6[5]	CR6[4:0]
V Display End	CR7[6]	CR7[1]	CR12[7]	CR12[6]	CR12[5]	CR12[4:0]
V Blank Start	CR9[5]	CR7[3]	CR15[7]	CR15[6]	CR15[5]	CR15[4:0]
V Blank End	CR1A[7]	CR1A[6]	CR16[7]	CR16[6]	CR16[5]	CR16[4:0]
V Sync Start	CR7[7]	CR7[2]	CR10[7]	CR10[6]	CR10[5]	CR10[4:0]
V Sync End						CR11[3:0]
Line Compare	CR9[6]	CR7[4]	CR18[7]	CR18[6]	CR18[5]	CR18[4:0]
Offset		CR1B[4]	CR13[7]	CR13[6]	CR13[5]	CR13[4:0]

The Extension and Overflow bits for the Screen Start A value are shown in Table 4-3.

Table 4-3. Screen Start A Extensions

19	18:17	16	15:8	7:0
CR1D[7]	CR1B[3:2]	CR1B[0]	CRC[7:0]	CRD[7:0]

4.21 CR0: CRTC Horizontal Total

I/O Port Address: 3D5h
Index: 00h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Horizontal Total [7]
6	Horizontal Total [6]
5	Horizontal Total [5]
4	Horizontal Total [4]
3	Horizontal Total [3]
2	Horizontal Total [2]
1	Horizontal Total [1]
0	Horizontal Total [0]

This register specifies the total number of character clocks per horizontal period.

Bit	Description
7:0	**Horizontal Total [7:0]:** The character clock (derived from the VCLK according to the character width) is counted in the character counter. The value of the character counter is compared with the value in this register to provide the basic horizontal timing. All horizontal and vertical timing is eventually derived from this register. The value in the register is 'Total number of character times minus five'. Figure 4-1 on page 4-27 indicates the way the horizontal and vertical timing is defined. The horizontal timing is calculated in terms of character clock periods and the vertical timing is calculated in terms of horizontal periods. Table 4-2 indicates how the various timing registers are extended.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.22 CR1: CRTC Horizontal Display End

I/O Port Address: 3D5h
Index: 01h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Horizontal Display End [7]
6	Horizontal Display End [6]
5	Horizontal Display End [5]
4	Horizontal Display End [4]
3	Horizontal Display End [3]
2	Horizontal Display End [2]
1	Horizontal Display End [1]
0	Horizontal Display End [0]

This register specifies the number of character clocks during horizontal display time.

Bit	Description
7:0	**Horizontal Display End [7:0]:** For Text modes, the number of character clocks during horizontal display time is the number of characters; for graphics modes, the number of character clocks during horizontal display time is the number of pixels in each scanline divided by the number of pixels in each character clock. The number is usually eight, but can be 16 for modes that use clock doubling. The value in the field is the number of character clocks minus one. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.23 CR2: CRTC Horizontal Blanking Start

I/O Port Address: 3D5h
Index: 02h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Horizontal Blanking Start [7]
6	Horizontal Blanking Start [6]
5	Horizontal Blanking Start [5]
4	Horizontal Blanking Start [4]
3	Horizontal Blanking Start [3]
2	Horizontal Blanking Start [2]
1	Horizontal Blanking Start [1]
0	Horizontal Blanking Start [0]

This register specifies the character count where Horizontal Blanking starts.

Bit	Description
7:0	**Horizontal Blanking Start [7:0]:** The value programmed into this register and its extension must always be larger than the value programmed into register CR1 and its extension. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.24 CR3: CRTC Horizontal Blanking End

I/O Port Address: 3D5h
Index: 03h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Compatible Read
6	Display Enable Skew [1]
5	Display Enable Skew [0]
4	Horizontal Blanking End [4]
3	Horizontal Blanking End [3]
2	Horizontal Blanking End [2]
1	Horizontal Blanking End [1]
0	Horizontal Blanking End [0]

This register determines the width of the Horizontal Blanking Period and controls Display Enable Skew and access to registers CR10 and CR11.

Bit **Description**

7 **Compatible Read:** If this bit is '0', registers CR10 and CR11 are write-only. If this bit is '1', registers CR10 and CR11 are read/write.

6:5 **Display Enable Skew [1:0]:** This 2-bit field specifies the number of character clocks that the display enable is to be delayed from Horizontal Total. This is necessary to compensate for the accesses of the character code and attribute byte, and the accesses of the font, etc. The following table indicates the coding of CR3[6:5]:

Bit		Skew	Note
6	5		
0	0	0	
0	1	1	Typical setting
1	0	2	
1	1	3	

NOTE: If the skew is too low, the left-most character is repeated. If the skew is too high, one or more characters disappear at the left of each character row.

4.24 CR3: CRTC Horizontal Blanking End *(cont.)*

Bit	Description
4:0	**Horizontal Blanking End [4:0]:** This field determines the width of the horizontal blanking period. This field is extended with CR5[7] and CR1A[5:4] (if enabled). The least-significant 5–8 bits of the character counter are compared with the contents of this field. When a match occurs, the horizontal blanking period ends. Note that the horizontal blanking period is limited to 63 or 255 character-clock times. The value programmed into this register can be calculated by adding the desired blanking period to the value programmed into register CR2 (Horizontal Blanking Start). The blanking period must never be extend past the horizontal total. If CR1B[5] or CR1B[7] is '1', this field is extended with Extension register CR1A[5:4]. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.25 CR4: CRTC Horizontal Sync Start

I/O Port Address: 3D5h
Index: 04h
Size (bits): 8

Bit	Description
7	Horizontal Sync Start [7]
6	Horizontal Sync Start [6]
5	Horizontal Sync Start [5]
4	Horizontal Sync Start [4]
3	Horizontal Sync Start [3]
2	Horizontal Sync Start [2]
1	Horizontal Sync Start [1]
0	Horizontal Sync Start [0]

This register specifies the time where HSYNC becomes active.

Bit	Description
7:0	**Horizontal Sync Start [7:0]:** This field specifies the Character Count where HSYNC (Horizontal Sync) becomes active. This is extended with CR1A[4]. Adjusting the value in this field moves the display horizontally on the screen. The Horizontal Sync Start *must* be equal to or greater than Horizontal Display End. The time from Horizontal Sync Start to Horizontal Total *must* be equal to or greater than four character times. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.26 CR5: CRTC Horizontal Sync End

I/O Port Address: 3D5h
Index: 05h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Horizontal Blanking End [5]
6	Horizontal Sync Delay [1]
5	Horizontal Sync Delay [0]
4	Horizontal Sync End [4]
3	Horizontal Sync End [3]
2	Horizontal Sync End [2]
1	Horizontal Sync End [1]
0	Horizontal Sync End [0]

This register specifies the position where the Horizontal Sync pulse ends, effectively specifying the width of the pulse. In addition, this register contains an overflow bit and a skew field.

Bit **Description**

7 **Horizontal Blanking End [5]:** This bit extends the Horizontal Blanking End value by one bit. Refer to register CR3 for an explanation of the Horizontal Blanking End Value.

6:5 **Horizontal Sync Delay [1:0]:** This 2-bit field delays the external Horizontal Sync pulse from the position implied in register CR4. This is necessary in some modes to allow internal timing signals triggered from Horizontal Sync Start to begin prior to Display Enable. The following table summarizes the HSYNC delay:

CR[6]	CR5[5]	Skew In Character Clocks
0	0	0
0	1	1
1	0	2
1	1	3

4.26 CR5: CRTC Horizontal Sync End *(cont.)*

Bit	Description
4:0	**Horizontal Sync End [4:0]:** This field determines the width of the Horizontal Sync pulse. The least-significant five bits of the Character Counter are compared with the contents of this field. When a match occurs, the Horizontal Sync pulse is ended. Note the Horizontal Sync pulse is limited to 31 character-clock times. The value to be programmed into this register can be calculated by subtracting the desired Sync width from the value programmed into register CR4 (Horizontal Sync Start). The Sync pulse must never be extend past the Horizontal Total. In addition, HSYNC must always end during the Horizontal Blanking period. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.27 CR6: CRTC Vertical Total

I/O Port Address: 3D5h
Index: 06h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Total [7]
6	Vertical Total [6]
5	Vertical Total [5]
4	Vertical Total [4]
3	Vertical Total [3]
2	Vertical Total [2]
1	Vertical Total [1]
0	Vertical Total [0]

This register specifies the total number of scanlines per frame.

Bit	Description
7:0	**Vertical Total [7:0]:** This field is the least-significant 8 bits of a 10-bit field that defines the total number of scanlines per frame. This field is extended with CR7[0] and CR7[5]. The value programmed into the Vertical Total field is the total number of scanlines minus two. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.28 CR7: CRTC Overflow

I/O Port Address: 3D5h
Index: 07h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Retrace Start [9]
6	Vertical Display End [9]
5	Vertical Total [9]
4	Line Compare [8]
3	Vertical Blanking Start [8]
2	Vertical Retrace Start [8]
1	Vertical Display End [8]
0	Vertical Total [8]

This register contains bits that extend various vertical count fields. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers. Cirrus Logic extension bits also reside in this register.

Bit	Description
7	**Vertical Retrace Start [9]:** This bit extends the Vertical Retrace Start (CR10) field to 10 bits.
6	**Vertical Display End [9]:** This bit extends the Vertical Display End (CR12) field to 10 bits.
5	**Vertical Total [9]:** This bit extends the Vertical Total (CR6) field to 10 bits.
4	**Line Compare [8]:** This bit extends the Line Compare (CR18) field to 9 bits.
3	**Vertical Blanking Start [8]:** This bit extends the Vertical Blanking Start (CR15) field to 9 bits.
2	**Vertical Retrace Start [8]:** This bit extends the Vertical Retrace Start (CR10) field to 9 bits.
1	**Vertical Display End [8]:** This bit extends the Vertical Display End (CR12) field to 9 bits.
0	**Vertical Total [8]:** This bit extends the Vertical Total (CR6) field to 9 bits.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.29 CR8: CRTC Screen A Preset Row-Scan

I/O Port Address: 3D5h
Index: 08h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Reserved
6	Byte Pan [1]
5	Byte Pan [0]
4	Screen A Preset Row Scan [4]
3	Screen A Preset Row Scan [3]
2	Screen A Preset Row Scan [2]
1	Screen A Preset Row Scan [1]
0	Screen A Preset Row Scan [0]

This register specifies the row scanline where Screen A begins, allowing scrolling on a scanline basis (soft scroll). In addition, this register specifies the Byte Pan (coarse panning).

Bit **Description**

7 **Reserved**

6:5 **Byte Pan [1:0]:** This 10-bit field controls coarse panning. It can specify a pan of up to 24 pixels with a resolution of eight pixels. AR13 provides for panning on a pixel basis. The values programmed into CR8[6:5] are interpreted as indicated in the following table:

CR8[6]	CR8[5]	Bytes	Pixels
0	0	0	0
0	1	1	8
1	0	2	16
1	1	3	24

4:0 **Screen A Preset Row Scan [4:0]:** This field specifies the scanline where the first character row begins. This provides scrolling on a scanline basis (soft scrolling). The contents of this field should be changed only during Vertical Retrace time.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.30 CR9: CRTC Character Cell Height

I/O Port Address: 3D5h
Index: 09h
Size (bits): 8
Access Type: Read/Write

Bit	Description
7	CRTC Scan Double
6	Line Compare [9]
5	Vertical Blank Start [9]
4	Character Cell Height [4]
3	Character Cell Height [3]
2	Character Cell Height [2]
1	Character Cell Height [1]
0	Character Cell Height [0]

This register specifies the number of scanlines in the character cell. In addition, it contains two vertical overflow bits and one control bit.

Bit	Description
7	**CRTC Scan Double:** If this bit is '1', every scanline is displayed twice in succession. The scanline counter-based parameters (character height, cursor start and end, and underline location) double. This bit typically displays 200-line modes at 400 scanlines. This function is not available in Interlaced Display modes.
6	**Line Compare [9]:** This bit extends the Line Compare field (CR18) to 10 bits.
5	**Vertical Blank Start [9]:** This bit extends the Vertical Blank Start field (CR15) to 10 bits.
4:0	**Character Cell Height [4:0]:** This field specifies the vertical size of the character cell in terms of scanlines. The value programmed into this field is the actual size minus 1. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.31 CRA: CRTC Text Cursor Start

I/O Port Address: 3D5h
Index: 0Ah
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Disable Text Cursor
4	Text Cursor Start [4]
3	Text Cursor Start [3]
2	Text Cursor Start [2]
1	Text Cursor Start [1]
0	Text Cursor Start [0]

This register specifies the scanline where the text cursor is to begin. In addition, this register contains a bit that disables the text cursor.

Bit	Description
7:6	**Reserved**
5	**Disable Text Cursor:** If this bit is '1', the text cursor is disabled (that is, it is removed). If this bit is '0', the text cursor functions normally.
4:0	**Text Cursor Start [4:0]:** This field specifies the scanline within the Character Cell where the text cursor is to start. If the Text Cursor Start value is greater than the Text Cursor End value, no text cursor is displayed. If the Text Cursor Start value is equal to the Text Cursor End value, the text cursor occupies a single scanline.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.32 CRB: CRTC Text Cursor End

I/O Port Address: 3D5h
Index: 0Bh
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Reserved
6	Text Cursor Skew [1]
5	Text Cursor Skew [0]
4	Text Cursor End [4]
3	Text Cursor End [3]
2	Text Cursor End [2]
1	Text Cursor End [1]
0	Text Cursor End [0]

This register specifies the scanline within the character cell where the text cursor is to end. It also contains a field that allows the text cursor to be skewed from the location specified in registers CRE and CRF.

Bit	Description
7	**Reserved**
6:5	**Text Cursor Skew [1:0]:** This 2-bit field specifies a delay, in character clocks, from the text cursor location specified in registers CRE and CRF to the actual cursor.
4:0	**Text Cursor End [4:0]:** This field specifies the scanline within the character where the text cursor is to end. A value greater than the character cell height yields an effective ending value equal to the cell height.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.33 CRC: CRTC Screen Start Address High

I/O Port Address: 3D5h
Index: 0Ch
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Screen Start A Address [15]
6	Screen Start A Address [14]
5	Screen Start A Address [13]
4	Screen Start A Address [12]
3	Screen Start A Address [11]
2	Screen Start A Address [10]
1	Screen Start A Address [9]
0	Screen Start A Address [8]

This register, and register CRD along with the Cirrus Logic extensions, specifies the location in display memory where the data to be displayed on the screen begins.

Bit	Description
7:0	**Screen Start A Address [15:8]:** The Screen Start A field specifies the location in display memory where the screen begins. This register contains bits 15:8 of this value. Bits 7:0 are in register CRD, bits 18:16 are in register CR1B, and bit 19 is CR1D[7]. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.34 CRD: CRTC Screen Start Address Low

I/O Port Address: 3D5h
Index: 0Dh
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Screen Start A Address [7]
6	Screen Start A Address [6]
5	Screen Start A Address [5]
4	Screen Start A Address [4]
3	Screen Start A Address [3]
2	Screen Start A Address [2]
1	Screen Start A Address [1]
0	Screen Start A Address [0]

This register and register CRC specify the location in display memory where the data to be displayed on the screen begins.

Bit	Description
7:0	**Screen Start A Address [7:0]:** The Screen Start A field specifies the location in display memory where the screen begins. This register contains bits 7:0 of this value, bits 15:8 are in register CRC. Extension bits 18:16 are in register CR1B and bit 19 is CR1D[7]. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.35 CRE: CRTC Text Cursor Location High

I/O Port Address: 3D5h
Index: 0Eh
Size (bits): 8
Access Type: Read/Write

Bit	Description
7	Text Cursor Location [15]
6	Text Cursor Location [14]
5	Text Cursor Location [13]
4	Text Cursor Location [12]
3	Text Cursor Location [11]
2	Text Cursor Location [10]
1	Text Cursor Location [9]
0	Text Cursor Location [8]

This register, and register CRF specify the location in display memory where the text cursor is to be displayed.

Bit	Description
7:0	**Text Cursor Location [15:8]:** This 16-bit field specifies the location in display memory where the text cursor is to be displayed. This register contains bits 15:8 of this field. Register CRF contains bits 7:0. **NOTE:** The value contained in this field is an address in display memory, not an offset from the beginning of the screen. If the value of Screen A Start is changed without a compensating change in the Text Cursor Location field, the text cursor moves on the screen. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.36 CRF: CRTC Text Cursor Location Low

I/O Port Address: 3D5h
Index: 0Fh
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Text Cursor Location [7]
6	Text Cursor Location [6]
5	Text Cursor Location [5]
4	Text Cursor Location [4]
3	Text Cursor Location [3]
2	Text Cursor Location [2]
1	Text Cursor Location [1]
0	Text Cursor Location [0]

This register and register CRE specify the location in display memory where the text cursor is to be displayed.

Bit	Description
7:0	**Text Cursor Location [7:0]:** This 16-bit field specifies the location in display memory where the text cursor is to be displayed. This register contains bits 7:0 of this field. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.37 CR10: CRTC Vertical Sync Start

I/O Port Address: 3D5h
Index: 10h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Sync Start [7]
6	Vertical Sync Start [6]
5	Vertical Sync Start [5]
4	Vertical Sync Start [4]
3	Vertical Sync Start [3]
2	Vertical Sync Start [2]
1	Vertical Sync Start [1]
0	Vertical Sync Start [0]

The Vertical Sync Start field specifies the scanline where the VSYNC pulse becomes active. This register contains the least-significant eight bits of that field.

Bit	Description
7:0	**Vertical Sync Start [7:0]:** This register contains bits 7:0 of the Vertical Sync Start field. This register is extended by bits in register CR7. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.38 CR11: CRTC Vertical Sync End

I/O Port Address: 3D5h
Index: 11h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Write Protect CR0–CR7
6	Refresh Cycle Control
5	Disable Vertical Interrupt
4	Clear Vertical Interrupt
3	Vertical Sync End [3]
2	Vertical Sync End [2]
1	Vertical Sync End [1]
0	Vertical Sync End [0]

This register specifies the scanline where the VSYNC pulse becomes inactive, thereby specifying the VSYNC pulse width. In addition, this register contains controls for the standard VGA interrupt and two miscellaneous control bits.

Bit	Description
7	**Write Protect CR0–CR7:** If this bit is '1', registers CR0–CR7 cannot be written. Writes addressed to those registers are ignored. CR7[4] (Line Compare Extension bit) can always be written. If this bit is '0', registers CR0–CR7 can be written normally.
6	**Refresh Cycle Control:** If this bit is '1', five refresh cycles are executed for each scanline. If this bit is '0', three refresh cycles are executed for each scanline. If CR18[3] is '1', one refresh cycle is executed per scanline and this bit is ignored.
5	**Disable Vertical Interrupt:** If this bit is '1', the vertical interrupt is disabled. The INTR# pin never goes active. If this bit is '0', the vertical interrupt is enabled and functions normally.
4	**Clear Vertical Interrupt:** If this bit is '0', the Interrupt Pending bit (FEAT[7]) clears to '0' and the INTR# pin is forced inactive. Programming this bit to '1' allows the next occurrence of the interrupt. This can be done immediately after programming it to '0'.

4.38 CR11: CRTC Vertical Sync End *(cont.)*

Bit	Description
3:0	**Vertical Sync End [3:0]:** This field determines the width of the VSYNC pulse. The four least-significant bits of the scanline counter are compared with the contents of this field. When a match occurs, the VSYNC pulse ends. Note the VSYNC pulse is limited to 15 scanlines. The value to be programmed into this register can be calculated by subtracting the desired Sync width from the value programmed into the Vertical Sync Start field. The Sync pulse must never extend past the Vertical Total. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.39 CR12: CRTC Vertical Display End

I/O Port Address: 3D5h
Index: 12h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Display End [7]
6	Vertical Display End [6]
5	Vertical Display End [5]
4	Vertical Display End [4]
3	Vertical Display End [3]
2	Vertical Display End [2]
1	Vertical Display End [1]
0	Vertical Display End [0]

This register specifies the scanline where the display is to end.

Bit	Description
7:0	**Vertical Display End [7:0]:** This register contains the least-significant eight bits of the Vertical Display End field. An additional bit is in register CR7. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.40 CR13: CRTC Offset (Pitch)

I/O Port Address: 3D5h
Index: 13h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Offset [7]
6	Offset [6]
5	Offset [5]
4	Offset [4]
3	Offset [3]
2	Offset [2]
1	Offset [1]
0	Offset [0]

This register specifies the distance in display memory between the beginnings of adjacent character rows or scanlines. This is also referred to as display 'pitch'.

Bit	Description
7:0	**Offset [7:0]:** This register specifies the distance in display memory between the beginnings of adjacent character rows or scanlines. This field is extended to nine bits with CR1B[4]. At the beginning of each scanline (except the first), the address that data is to be fetched from is calculated by adding the contents of this register to the beginning address of the previous scanline or character row. The offset is left-shifted one or two bit positions depending on the state of CR17[6].

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.41 CR14: CRTC Underline Row Scanline

I/O Port Address: 3D5h
Index: 14h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Reserved
6	DoubleWord Mode
5	Count by Four
4	Underline Scanline [4]
3	Underline Scanline [3]
2	Underline Scanline [2]
1	Underline Scanline [1]
0	Underline Scanline [0]

This register specifies the scanline where the underline appears. This is for VGA text modes only.

Bit	Description
7	**Reserved**
6	**DoubleWord Mode:** When this bit is '1', double-word addresses are forced. The CRTC memory address counter is rotated left two bit positions so that Display Memory Address bits 1 and 0 are sourced from CRTC Address Counter bits 13 and 12, respectively. When this bit is '0', CR17[6] controls whether the device uses byte or word addresses.
5	**Count by Four:** This bit must be '1' when DoubleWord mode is enabled to clock the memory address counter with character clock divided by four. This bit must be '0' when Double-word mode is not enabled.
4:0	**Underline Scanline [4:0]:** This field specifies the scanline within the character cell where the underline occurs.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.42 CR15: CRTC Vertical Blank Start

I/O Port Address: 3D5h
Index: 15h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Blank Start [7]
6	Vertical Blank Start [6]
5	Vertical Blank Start [5]
4	Vertical Blank Start [4]
3	Vertical Blank Start [3]
2	Vertical Blank Start [2]
1	Vertical Blank Start [1]
0	Vertical Blank Start [0]

This register specifies the scanline where BLANK# becomes active.

Bit	Description
7:0	**Vertical Blank Start [7:0]:** The Vertical Blank Start field specifies the scanline where Vertical Blank is to begin. The least-significant eight bits of that field are in this register. Overflow bits are in registers CR7 and CR9. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.43 CR16: CRTC Vertical Blank End

I/O Port Address: 3D5h
Index: 16h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Vertical Blank End [7]
6	Vertical Blank End [6]
5	Vertical Blank End [5]
4	Vertical Blank End [4]
3	Vertical Blank End [3]
2	Vertical Blank End [2]
1	Vertical Blank End [1]
0	Vertical Blank End [0]

The Vertical Blank End field specifies the scanline where Vertical Blank ends.

Bit	Description
7:0	**Vertical Blank End [7:0]:** his register contains the least-significant eight bits of the Vertical Blank End field. If CR1B[5] is ‘1’, this field is extended with CR1A[7:6]. The contents of the Vertical Blank End field are compared to the scanline counter to determine when to terminate Vertical Blank. This limits the duration of Vertical Blank to 255 scanlines if CR1B[5] is ‘0’ or 1023 scanlines if CR1B[5] is ‘1’. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.44 CR17: CRTC Mode Control

I/O Port Address: 3D5h
Index: 17h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Timing Enable
6	Byte/Word Mode
5	Address Wrap
4	Reserved
3	Count by Two
2	Multiply Vertical Registers by Two
1	Select Row-Scan Counter
0	Compatibility Mode (CGA) Support

This register contains a number of miscellaneous control bits.

Bit	Description
7	**Timing Enable:** If this bit is '1', the CRTC timing logic is enabled and functions normally. If this bit is '0', the CRTC timing logic is disabled.
6	**Byte/Word Mode:** If this bit is '1', the contents of the CRTC address counter are sent to the display memory without being rotated. If this bit is '0', the contents of the CRTC address counter are rotated left one bit position before being sent to the display memory.
5	**Address Wrap:** If CR17[6] is '1', this bit is ignored. If CR17[6] is '0' and this bit is '1', then the left rotation described above involves 16 bits of the CRTC address counter. If CR17[6] is '0' and this bit is '0', then the left rotation described above involves 14 bits of the CRTC address counter.
4	**Reserved**
3	**Count by Two:** If this bit is '1', the CL-GD5446 clocks the memory address counter with character clock divided by two. If this bit is '0', then the CL-GD5446 clocks the memory address counter with character clock.
2	**Multiply Vertical Registers by Two:** If this bit is '1', the scanline counter is clocked with HSYNC divided by two. This allows the number of scanlines to be doubled to 2048. Note that all periods are even multiples of two scanlines. If this bit is '0', the scanline counter is clocked with HSYNC.

4.44 CR17: CRTC Mode Control *(cont.)*

Bit	Description
1	**Select Row-Scan Counter:** If this bit is '0', Row-scan Counter [1] is substituted for CRTC Address Counter [14]. This provides Hercules compatibility. **NOTE:** The Cirrus Logic BIOS does *not* support Hercules compatibility. If this bit is '1', the substitution described above does not occur.
0	**Compatibility Mode (CGA) Support:** If this bit is '0', Row-scan Counter [0] is substituted for CRTC Address Counter [14]. This provides for CGA compatibility. If this bit is '1', the substitution described above does not occur.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.45 CR18: CRTC Line Compare

I/O Port Address: 3D5h
Index: 18h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Line Compare [7]
6	Line Compare [6]
5	Line Compare [5]
4	Line Compare [4]
3	Line Compare [3]
2	Line Compare [2]
1	Line Compare [1]
0	Line Compare [0]

This register specifies where Screen A terminates and Screen B begins.

Bit	Description
7:0	**Line Compare [7:0]:** This register contains the eight least-significant bits of the Line Compare field. This is extended with one bit each in registers CR7 and CR9. The Line Compare field can implement a vertically split screen. The top portion of the screen is Screen A and can begin anywhere in display memory. Screen A can be panned and scrolled on a pixel basis. The bottom portion of the screen is Screen B. Screen B always begins at location 0 in display memory and cannot be panned or scrolled. Refer to Figure 4-1 on page 4-27 and Table 4-2 on page 4-28 for a summary of CRTC Timing registers.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.46 CR22: Graphics Data Latches Readback (Read only)

I/O Port Address: 3D5h
Index: 22h
Size (bits): 8
Access Type: Read only

Bit	Description
7	Graphics Data Latch n Readback [7]
6	Graphics Data Latch n Readback [6]
5	Graphics Data Latch n Readback [5]
4	Graphics Data Latch n Readback [4]
3	Graphics Data Latch n Readback [3]
2	Graphics Data Latch n Readback [2]
1	Graphics Data Latch n Readback [1]
0	Graphics Data Latch n Readback [0]

This register address reads the four graphics controller data latches.

Bit	Description
7:0	**Graphics Data Latch n Readback [7:0]:** This read-only register can read back one of the four graphics controller data latches. The latch is selected with GR4[1:0]. These latches are loaded whenever display memory is read by the CPU.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.47 CR24: Attribute Controller Toggle Readback (Read only)

I/O Port Address: 3D5h
Index: 24h
Size (bits): 8
Access Type: Read only

Bit	Description
7	Attribute Controller Toggle
6:0	Reserved

This read-only register provides access to the attribute controller toggle.

Bit	Description
7	**Attribute Controller Toggle:** If this bit is '1', the attribute controller reads or writes a data value on the next access. If this bit is '0', the attribute controller reads or writes an index value on the next access.
6:0	**Reserved**

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.48 CR26: Attribute Controller Index Readback (Read only)

I/O Port Address: 3D5h
Index: 26h
Size (bits): 8
Access Type: Read only

Bit	Description
7:6	Reserved
5	Display Enable
4	Attribute Controller Index [4]
3	Attribute Controller Index [3]
2	Attribute Controller Index [2]
1	Attribute Controller Index [1]
0	Attribute Controller Index [0]

This read-only register provides access to the current attribute controller index.

Bit	Description
7:6	**Reserved**
5	**Display Enable:** This bit follows the Display Enable bit in the Attribute Controller Index register (this bit is a read-only copy of ARX[5]).
4:0	**Attribute Controller Index [4:0]:** This field follows the index in the Attribute Controller Index register (this bit is a read-only copy of ARX[4:0]).

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

4.49 GRX: Graphics Controller Index

I/O Port Address: 3CEh
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Graphics Controller Index [5]
4	Graphics Controller Index [4]
3	Graphics Controller Index [3]
2	Graphics Controller Index [2]
1	Graphics Controller Index [1]
0	Graphics Controller Index [0]

This register specifies the register in the VGA graphics controller group or the Extension register accessed by the next I/O read or I/O write to Address 3CF.

Bit	Description
7:6	**Reserved**
5:0	**Graphics Controller Index [5:0]:** This field specifies the register in the VGA Graphics Controller group or the Extension register accessed by the next I/O read or I/O write to Address 3CF.

4.50 GR0: Set/Reset / Background Color Byte 0

I/O Port Address: 3CFh
Index: 00h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Set/Reset Plane 3
2	Set/Reset Plane 2
1	Set/Reset Plane 1
0	Set/Reset Plane 0

This register specifies the values to be written into the respective display memory planes when the processor executes a Write mode 0 or Write mode 3 operation.

This register (all eight bits) is also used for color expansion (see Section 5.1 on page 5-3).

Bit	Description
7:4	**Reserved**
3:0	**Set/Reset Plane [3:0]:** These bits control the values written into the respective display memory planes for Write modes 0 and 3. Refer to the description of register GR5 for an overview of the Write modes.

4.51 GR1: Set/Reset Enable / Foreground Color Byte 0

I/O Port Address: 3CFh
Index: 01h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Enable SR Plane 3
2	Enable SR Plane 2
1	Enable SR Plane 1
0	Enable SR Plane 0

This register and register GR0 determine the values to be written into the respective display memory planes when Write mode 0 is selected.

This register (all eight bits) is also used for color expansion (see Section 5.1 on page 5-3).

Bit	Description
7:4	**Reserved**
3:0	**Enable SR Plane [3:0]:** If a bit in this field is ‘1’, the corresponding value in register GR0 is written into the corresponding display memory plane. If a bit in this field is ‘0’, the corresponding value from the CPU Data bus is written into the corresponding display memory plane. Refer to the description of register GR5 for an overview of the Write modes.

4.52 GR2: Color Compare

I/O Port Address: 3CFh
Index: 02h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Color Compare Plane [3]
2	Color Compare Plane [2]
1	Color Compare Plane [1]
0	Color Compare Plane [0]

This register specifies the color compare value for Read mode 1.

Bit	Description
7:4	**Reserved**
3:0	**Color Compare Plane [3:0]:** When a Read mode 1 occurs, these four bits are compared with each of the eight bits from the corresponding display memory planes under the mask in register GR7. Refer to the description of register GR5 for an overview of the Read modes.

4.53 GR3: Data Rotate

I/O Port Address: 3CFh
Index: 03h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:5	Reserved
4	Function Select [1]
3	Function Select [0]
2	Rotate Count [2]
1	Rotate Count [1]
0	Rotate Count [0]

This register contains two fields that are used with Write modes 0 and 3.

Bit	Description
7:5	**Reserved**
4:3	**Function Select [1:0]:** This field controls the operation that occurs between the data in the latches and the data from the CPU or Set/Reset logic. The result of this operation is written into display memory. This field is for Write mode 0 only. The operations are summarized in the following table:
2:0	**Rotate Count [2:0]:** This field allows data from the CPU bus to be rotated by as many as seven bit positions prior to being altered by the Set/Reset logic. Refer to the description of register GR5 for an overview of the Write modes.

GR3[4]	GR3[3]	Operation
0	0	None: The data in the latches are ignored
0	1	Logical AND
1	0	Logical OR
1	1	Logical XOR

4.54 GR4: Read Map Select

I/O Port Address: 3CFh
Index: 04h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:2	Reserved
1	Plane Select [1]
0	Plane Select [0]

This register specifies the display memory plane for Read mode 0.

Bit	Description
7:2	**Reserved**
1:0	**Plane Select [1:0]:** This field specifies the display memory plane for Read mode 0. The values are shown in the following table:

Bit		Plane Selected
1	**0**	
0	0	0
0	1	1
1	0	2
1	1	3

4.55 GR5: Graphics Controller Mode

I/O Port Address: 3CFh
Index: 05h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Reserved
6	256-color Mode
5	Shift Register Mode
4	Odd/Even
3	Read Mode
2	Reserved
1	Write Mode [1]
0	Write Mode [0]

This register specifies the Read and Write modes and controls the configuration of the Data Shift registers.

Bit	Description
7	**Reserved**
6	**256-color Mode:** If this bit is '1', the display shift registers are configured for 256-color display modes; GR5[5] is ignored. If this bit is '0', the Display Shift registers are configured for 16-, 4-, or 2-color modes.
5	**Shift Register Mode:** If this bit is '1', the display Shift registers are configured for CGA compatibility. This is for Display modes 4 and 5. If this bit is '0', the Display Shift registers are configured for EGA compatibility.
4	**Odd/Even:** If this bit is '1', the graphics controller is configured for Odd/Even Addressing mode. This bit should always be the opposite value as set in SR4[2].

4.55 GR5: Graphics Controller Mode *(cont.)*

Bit	Description
3	**Read Mode:** This bit specifies whether the device is in Read mode 0 or Read mode 1. **Read Mode 0:** If this bit is '0', the CPU reads data directly from display memory. Each read returns eight adjacent bits of the display memory plane specified in GR4[1:0]. The color-match logic is not used in Read mode 0. Note that an I/O read of register CR22 forces a Read mode 0 for that operation. **Read Mode 1:** If this bit is '1', the CPU reads the results of the color compare logic. Read mode 1 allows eight adjacent pixels (in 16-color modes) to be compared to a specified color value in a single operation. Each of the eight bits returned to the processor indicates the result of a compare between the four bits of the Color Compare field (GR2[3:0]) and the bits from the four display memory planes. If the four bits of the Color Compare field match the four bits from the display memory planes, '1' is returned for the corresponding bit position. If any bits in the Color Don't Care field (GR7[3:0]) are '0', the corresponding plane comparison is forced to match.
2	**Reserved**
1:0	**Write Mode [1:0]:** These two bits specify the Write mode. Note that write modes 1, 2, and 3 cannot be used with packed pixel modes. In general, the BitBLT engine should be used to perform these functions. **Write Mode 0:** Each of the four display memory planes is written with the CPU data rotated by the number of counts in GR3[2:0]. If a bit in GR1[3:0] is '1', the corresponding plane is written with the contents of the corresponding bit in GR0[3:0]. If SR7[0] is '1', CPU data is written regardless of the contents of GR1[3:0]. The contents of the data latches can be combined with the data from the Set/Reset logic under control of GR3[4:3]. Bit planes are enabled with SR2[3:0]; bit positions are enabled with register GR8. **Write Mode 1:** Each of the four display memory planes is written with the data in the data latches. The data latches must be previously loaded from display memory with a previous read. Register GR8 is ignored in Write mode 1. **Write Mode 2:** Display memory planes 3:0 are written with the values of Data bits 3:0, respectively. The four bits are replicated eight times each to write up to eight adjacent pixels. Bit planes are enabled with SR2[3:0]; bit positions are enabled with register GR8. The data rotator, Set/Reset logic, and Function Select fields are ignored in Write mode 2. **Write Mode 3:** The data for each display memory plane comes from the corresponding bit of GR0[3:0]. The bit-position-enable field is formed with the logical AND of register GR8 and the rotated CPU data. The Set/Reset and Function Select fields are ignored in Write mode 3.

4.56 GR6: Miscellaneous

I/O Port Address: 3CFh
Index: 06h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Memory Map [1]
2	Memory Map [0]
1	Chain Odd Maps to Even
0	Graphics Mode

This register contains miscellaneous control bits.

Bit **Description**

7:4 **Reserved**

3:2 **Memory Map [1:0]:** This field specifies the beginning address and size of the display memory in the host address space. This is summarized in the following table:

GR6[3]	GR6[2]	Memory Map	Beginning Address	Length	Mode(s)
0	0	0	A000:0	128K	Extended
0	1	1	A000:0	64K	EGA/VGA
1	0	2	B000:0	32K	Hercules™
1	1	3	B800:0	32K	CGA

1 **Chain Odd Maps to Even:** When this bit is '1', CPU Address bit 0 is replaced with a higher-order address bit. This causes even host addresses to access Planes 0 and 2, and odd host addresses to access Planes 1 and 3. This mode is useful for MDA emulation. This bit also modifies the meaning of SR7[0].

0 **Graphics Mode:** If this bit is '1', the CL-GD5446 functions in APA (VGA Graphics) modes. If it is '0', the device functions in AN (VGA Text) modes.

4.57 GR7: Color Don't Care

I/O Port Address: 3CFh
Index: 07h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Color Don't Care Plane [3]
2	Color Don't Care Plane [2]
1	Color Don't Care Plane [1]
0	Color Don't Care Plane [0]

This register and register GR2 are for Read mode 1 accesses.

Bit	Description
7:4	**Reserved**
3:0	**Color Don't Care Plane [3:0]:** These four bits control whether the four planes are involved in color compares. If a bit is '1', the corresponding plane is involved; if a bit is '0', the corresponding plane is not involved. Refer to the description of register GR5 for an overview of the Read modes.

4.58 GR8: Bit Mask

I/O Port Address: 3CFh
Index: 08h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	Write Enable [7]
6	Write Enable [6]
5	Write Enable [5]
4	Write Enable [4]
3	Write Enable [3]
2	Write Enable [2]
1	Write Enable [1]
0	Write Enable [0]

This register controls writes to display memory on a bit basis in Write modes 0, 2, and 3.

Bit	Description
7:0	**Write Enable [7:0]:** Each bit in this register controls whether the corresponding bit in display memory is written in Write modes 0, 2, and 3. If a bit in this field is '1', the corresponding bit in display memory is written. If a bit is '0', the corresponding bit in display memory is not written. This write protection is orthogonal to that provided by register SR2.

4.59 ARX: Attribute Controller Index

I/O Port Address: 3C0h/3C1h
Index: –
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Display Enable
4	Attribute Controller Index [4]
3	Attribute Controller Index [3]
2	Attribute Controller Index [2]
1	Attribute Controller Index [1]
0	Attribute Controller Index [0]

This register specifies the register in the Attribute Controller block accessed with the next I/O read or I/O write to 3C1 or 3C0, respectively. Note that the same port addresses are used for the index and data for the Attribute Controller block, unlike the other blocks where the Index and Data registers are at different addresses. Alternate writes toggle between index and data. It is possible to read the toggle at register CR24, and the index value at register CR26. See Section 9.2 in Chapter 9, “Programming Notes”, for information about the mechanics of accessing the Attribute Controller registers.

Bit	Description
7:6	**Reserved**
5	**Display Enable:** When this bit is ‘0’, the screen displays the color indicated by the Overscan register (AR11). When this bit is ‘1’, normal graphics and video are displayed.
4:0	**Attribute Controller Index [4:0]:** This field is the index into the Data registers in the Attribute Controller block.

4.60 AR0–ARF: Attribute Controller Palette

I/O Port Address: 3C0h/3C1h
Index: 00h–0Fh
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Secondary Red
4	Secondary Green/Intensity
3	Secondary Blue/Monochrome
2	Red
1	Green
0	Blue

In 16-color Text and Graphics modes, these digital palette entries are selected by the four bits of pixel data, and point to VGA palette entries. The VGA palette entries are programmed so that the DAC outputs reflect these values. That is, VGA palette simulates standard EGA colors.

Bit	Description
7:6	**Reserved**
5:0	**Palette Entries**

4.61 AR10: Attribute Controller Mode

I/O Port Address: 3C0h/3C1h
Index: 10h
Size (bits): 8
Access Type: Read/write

Bit	Description
7	AR14 Graphics Source Enable
6	Pixel Double Clock Select
5	Pixel Panning Compatibility
4	Reserved
3	Blink Enable
2	Line Graphics Enable
1	Display Type
0	Graphics Mode

This register contains miscellaneous control bits for the attribute controller.

Bit	Description
7	**AR14 Graphics Source Enable:** If this bit is '1', AR14[1:0] are the source for the LUT address (Color Bits C [5:4]). This allows the rapid selection of four 16-color palettes. This bit is ignored for deeper color modes. If this bit is '0', the Palette registers AR0–ARF[5:4] are the source for the LUT address (Color Bits C [5:4]).
6	**Pixel Double Clock Select:** If this bit is '1', pixels are clocked every other clock cycle and the Palette registers, AR0–ARF, are bypassed. This is used with mode 13. The sequencer logic operates at twice the pixel rate. If this bit is '0', pixels are clocked every cycle.
5	**Pixel Panning Compatibility:** If this bit is '1', a line compare match in the CRTC forces the output of the Pixel Panning register to '0', until the next VSYNC occurs. This allows the panning of Screen A without Screen B. If this bit is '0', the two parts of a split screen pan together.
4	**Reserved**
3	**Blink Enable:** If this bit is '1', character blinking is enabled at the vertical refresh frequency divided by 32. If this bit is '0', character blinking is disabled.
2	**Line Graphics Enable:** If this bit is '1', the ninth bit of a nine-bit-wide character cell is made the same as the eighth bit for character codes in the range C0–DF. If this bit is '0', the ninth bit of a nine-bit-wide character cell is the same as the background.

4.61 AR10: Attribute Controller Mode *(cont.)*

Bit **Description**

1 **Display Type:** This bit is useful only if the CL-GD5446 is in AN mode. If this bit is '1', the contents of the attribute byte are considered MDA-compatible attributes. The following table shows examples of the monochrome attributes:

Blink Bit 7	Background Bit [6:4]	Intensity Bit 3	Foreground Bit [2:0]	Hex Code	Attribute
0	0	0	7	07	Normal
0	0	1	7	0F	Intense
0	0	0	1	01	Underline
0	0	1	1	09	Underline Intense
0	7	0	0	70	Reverse
1	7	0	0	F0	Blinking Reverse

If this bit is '0', the contents of the attribute byte are considered color attributes.

0 **Graphics Mode:** If this bit is '1', the attribute controller functions in APA mode. If this bit is '0', the attribute controller functions in AN modes.

4.62 AR11: Overscan (Border) Color

I/O Port Address: 3C0h/3C1h
Index: 11h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Secondary Red
4	Secondary Green
3	Secondary Blue
2	Red
1	Green
0	Blue

This register points to the entry in the LUT that defines the border color. Typically, LUT entries are programmed so that the as color defined, is the color that actually results. The border is defined as that portion of the raster between blanking and active display, on all four sides. Refer to Figure 4-1 on page 4-27.

Bit	Description
7:6	**Reserved**
5:0	**Border Color [5:0]:** Either four or six of these bits select the LUT entry for the border color in CGA and EGA modes.

4.63 AR12: Color Plane Enable

I/O Port Address: 3C0h/3C1h
Index: 12h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:6	Reserved
5	Video Status Mux [1]
4	Video Status Mux [0]
3	Enable Plane [3]
2	Enable Plane [2]
1	Enable Plane [1]
0	Enable Plane [0]

This register contains a field that enables the four planes into the Attribute Controller Palette registers. It also contains a field that selects the inputs for Diagnostic bits in STAT[5:4].

Bit **Description**

7:6 **Reserved**

5:4 **Video Status Mux [1:0]:** This field selects the inputs for the Diagnostic bits in STAT[5:4] as indicated in the following table:

AR12[5]	AR12[4]	STAT[5]	STAT[4]
0	0	P[2]	P[0]
0	1	P[5]	P[4]
1	0	P[3]	P[1]
1	1	P[7]	P[6]

3:0 **Enable Color Plane [3:0]:** If any bit in this field is ‘1’, the data from the corresponding display memory plane is enabled in the selection in the Attribute Controller Palette register. If any bit in this field is ‘0’, the data from the corresponding display memory plane is forced to ‘0’ in the selection in the Attribute Controller Palette register.

4.64 AR13: Pixel Panning

I/O Port Address: 3C0h/3C1h
Index: 13h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Pixel Panning [3]
2	Pixel Panning [2]
1	Pixel Panning [1]
0	Pixel Panning [0]

This register specifies the number of pixels the display data is shifted left. This field functions both in the APA (Graphics) and AN (Alphanumeric) modes.

Bit	Description
7:4	**Reserved**
3:0	**Pixel Panning [3:0]:** This field specifies the number of pixels the display data is shifted left. This field is interpreted as indicated in the following table:

AR13[3:0]	9-bit Characters	8-bit Characters	Mode 13
0	1 bit left	None	None
1	2 bits left	1 bit left	None
2	3 bits left	2 bits left	1 bit left
3	4 bits left	3 bits left	1 bit left
4	5 bits left	4 bits left	2 bits left
5	6 bits left	5 bits left	2 bits left
6	7 bits left	6 bits left	3 bits left
7	8 bits left	7 bits left	3 bits left
8–F	No shift	1 bit right	3 bits left

4.65 AR14: Color Select

I/O Port Address: 3C0h/3C1h
Index: 14h
Size (bits): 8
Access Type: Read/write

Bit	Description
7:4	Reserved
3	Color Bit C [7]
2	Color Bit C [6]
1	Color Bit C [5]
0	Color Bit C [4]

This register contains two fields involved in the selection of addresses into the LUT.

Bit	Description
7:4	**Reserved**
3:2	**Color Bit C [7:6]:** These two bits are concatenated with six bits from the Attribute Controller Palette register to form the address into the LUT and to drive P[7:6]. These bit are ignored in 8-, 16-, and 24-bit Pixel modes.
1:0	**Color Bits C [5:4]:** If AR10[7] is ‘1’, these two bits replace the corresponding two bits from the Attribute Controller Palette register to form the address into the LUT and to drive P[7:6]. If AR10[7] is ‘0’, these two bits are ignored. These bits are ignored in 8-, 16-, 24-, and 32-bit Pixel modes.

BitBLT Registers

5. BITBLT EXTENSION REGISTERS

The BitBLT Extension registers are summarized in Table 5-1.

Table 5-1. BitBLT Extension Registers Quick Reference

Abbreviation	Register Name	Port	Index	MMI/O	Page
GR0	Background Color Byte 0	3CFh	00h	00	4-62
GR1	Foreground Color Byte 0	3CFh	01h	04	4-63
GR10	Background Color Byte 1	3CFh	10h	01	5-3
GR11	Foreground Color Byte 1	3CFh	11h	05	5-3
GR12	Background Color Byte 2	3CFh	12h	02	5-3
GR13	Foreground Color Byte 2	3CFh	13h	06	5-3
GR14	Background Color Byte 3	3CFh	14h	03	5-3
GR15	Foreground Color Byte 3	3CFh	15h	07	5-3
GR20	BLT Width Byte 0	3CFh	20h	08	5-4
GR21	BLT Width Byte 1	3CFh	21h	09	5-4
GR22	BLT Height Byte 0	3CFh	22h	0A	5-5
GR23	BLT Height Byte 1	3CFh	23h	0B	5-5
GR24	BLT Destination Pitch Byte 0	3CFh	24h	0C	5-6
GR25	BLT Destination Pitch Byte 1	3CFh	25h	0D	5-6
GR26	BLT Source Pitch Byte 0	3CFh	26h	0E	5-7
GR27	BLT Source Pitch Byte 1	3CFh	27h	0F	5-7
GR28	BLT Destination Start Byte 0	3CFh	28h	10	5-8
GR29	BLT Destination Start Byte 1	3CFh	29h	11	5-8
GR2A	BLT Destination Start Byte 2	3CFh	2Ah	12	5-8
GR2C	BLT Source Start Byte 0	3CFh	2Ch	14	5-9
GR2D	BLT Source Start Byte 1	3CFh	2Dh	15	5-9
GR2E	BLT Source Start Byte 2	3CFh	2Eh	16	5-9
GR2F	BLT Destination Left-Side Clipping	3CFh	2Fh	17	5-10
GR30	BLT Mode	3CFh	30h	18	5-11
GR31	BLT Start/Status	3CFh	31h	40	5-13
GR32	BLT ROP (Raster Operation)	3CFh	32h	1A	5-15
GR33	BLT Mode Extensions	3CFh	33h	1B	5-17
GR34	Transparent BLT Key Color Byte 0	3CFh	34h	1C	5-18
GR35	Transparent BLT Key Color Byte 1	3CFh	35h	1D	5-18

5.1 GR10–GR15: Color Expansion Foreground/Background Colors

I/O Port Address: 3CFh
Index: 10h–15h
MM/IO Offset: 01–03, 05–07

Bit	Description
7	Color [7]
6	Color [6]
5	Color [5]
4	Color [4]
3	Color [3]
2	Color [2]
1	Color [1]
0	Color [0]

These six registers contain the foreground and background colors for color expansion. Refer to Chapter 9, “Programming Notes”, for detailed information regarding BitBLTs with color expansion.

Bit **Description**

7:0 **Color [7:0]:** These six registers, in combination with registers GR0 and GR1, contain the foreground and background colors used for color expansion and solid fill. The following table gives the MMI/O offset for each register and shows how it is used for the four possible expansion widths. A hyphen in the table indicates that register is not used for that expansion width.

		Expansion Width			
Register	**MMI/O**	**8-bpp**	**16-bpp**	**24-bpp**	**32-bpp**
GR1	4	FG[a] color	FG low byte	FG Blue	FG blue
GR11	5	–	FG high byte	FG Green	FG green
GR13	6	–	–	FG Red	FG red
GR15	7	–	–	–	FR Alpha
GR0	0	BG[b] color	BG low byte	24-bpp color expansion must use transparency	BG blue
GR10	1	–	BG high byte		BG green
GR12	2	–	–		BG red
GR14	3	–	–		BG Alpha

[a] FG indicates foreground.
[b] BG indicates background.

5.2 GR20–GR21: BLT Width Byte 0, 1

I/O Port Address: 3CFh
Index: 20h, 21h
MM/IO Offset: 08, 09

Bit	Description
7	BLT Width [7]
6	BLT Width [6]
5	BLT Width [5]
4	BLT Width [4] / [12]
3	BLT Width [3] / [11]
2	BLT Width [2] / [10]
1	BLT Width [1] / [9]
0	BLT Width [0] / [8]

This register pair contains the 13-bit value specifying the width – 1, in bytes, of the areas involved in a BitBLT. A 13-bit value allows BitBLT widths of up to 8192 bytes. Refer to Section 9.4, “BitBLT Engine”.

Bit	Description
12:0	**BLT Width [12:0]:** Bits 12:8 are in register GR21; bits 7:0 are in register GR20.

5.3 GR22–GR23: BLT Height Byte 0, 1

I/O Port Address: 3CFh
Index: 22h, 23h
MM/IO Offset: 0A, 0B

Bit	Description
7	BLT Height [7]
6	BLT Height [6]
5	BLT Height [5]
4	BLT Height [4]
3	BLT Height [3]
2	BLT Height [2] /[10]
1	BLT Height [1] / [9]
0	BLT Height [0] / [8]

This register pair contains the 11-bit value specifying height – 1, in scanlines, of the areas involved in a BitBLT. An 11-bit field allows BitBLT heights of up to 2048 scanlines.

Bit	Description
12:0	**BLT Height [10:0]:** Bits 10:8 are in register GR23; bits 7:0 are in register GR22.

5.4 GR24–GR25: BLT Destination Pitch Byte 0, 1

I/O Port Address: 3CFh
Index: 24h, 25h
MM/IO Offset: 0C, 0D

Bit	Description
7	BLT Destination Pitch [7]
6	BLT Destination Pitch [6]
5	BLT Destination Pitch [5]
4	BLT Destination Pitch [4] / [12]
3	BLT Destination Pitch [3] / [11]
2	BLT Destination Pitch [2] / [10]
1	BLT Destination Pitch [1] / [9]
0	BLT Destination Pitch [0] / [8]

This register pair contains the 13-bit value specifying the pitch (that is, the scanline-to-scanline byte address offset) of the destination area involved in a BitBLT.

Bit	Description
12:0	**BLT Destination Pitch [12:0]:** Bits 12:8 are in register GR25; bits 7:0 are in register GR24.

5.5 GR26–GR27: BLT Source Pitch Byte 0, 1

I/O Port Address: 3CFh
Index: 26h, 27h
MM/IO Offset: 0E, 0F

Bit	Description
7	BLT Source Pitch [7]
6	BLT Source Pitch [6]
5	BLT Source Pitch [5]
4	BLT Source Pitch [4] / [12]
3	BLT Source Pitch [3] / [11]
2	BLT Source Pitch [2] / [10]
1	BLT Source Pitch [1] / [9]
0	BLT Source Pitch [0] / [8]

This register pair contains the 13-bit value specifying the pitch (that is, the scanline-to-scanline byte address offset) of the source area involved in a BitBLT.

Bit	Description
12:0	**BLT Source Pitch [12:0]:** Bits 12:8 are in register GR27; bits 7:0 are in register GR26.

5.6 GR28–GR2A: BLT Destination Start Byte 0, 1, 2

I/O Port Address: 3CFh
Index: 28h, 29h, 2Ah
MM/IO Offset: 10, 11, 12

Bit	Description
7	BLT Destination Start [7] / [15]
6	BLT Destination Start [6] / [14]
5	BLT Destination Start [5] / [13] / [21]
4	BLT Destination Start [4] / [12] / [20]
3	BLT Destination Start [3] / [11] / [19]
2	BLT Destination Start [2] / [10] / [18]
1	BLT Destination Start [1] / [9] / [17]
0	BLT Destination Start [0] / [8] / [16]

This register triplet contains the 22-bit value specifying the byte address of the beginning destination pixel for a BitBLT. A 22-bit value allows an address of up to 4 Mbytes.

Bit	Description
21:0	**BLT Destination Start [21:0]:** Bits 21:16 are in register GR2A; bits 15:8 are in register GR29; bits 7:0 are in register GR28.

5.7 GR2C–GR2E: BLT Source Start Byte 0, 1, 2

I/O Port Address: 3CFh
Index: 2Ch, 2Dh, 2Eh
MM/IO Offset: 14, 15, 16

Bit	Description
7	BLT Source Start [7] / [15]
6	BLT Source Start [6] / [14]
5	BLT Source Start [5] / [13] / [21]
4	BLT Source Start [4] / [12] / [20]
3	BLT Source Start [3] / [11] / [19]
2	BLT Source Start [2] / [10] / [18]
1	BLT Source Start [1] / [9] / [17]
0	BLT Source Start [0] / [8] / [16]

This register triplet contains the 22-bit value specifying the byte address of the beginning source pixel for a BitBLT. A 22-bit value allows an address of up to 4 Mbytes.

Bit	Description
21:0	**BLT Source Start [21:0]:** Bits 21:16 are in GR2E; bits 15:8 are in register GR2D; and bits 7:0 are in register GR2C. When a pattern copy BitBLT occurs, the least-significant bits of register GR2C also specify the offset into the pattern. See Section 9.4.8.

5.8 GR2F: BLT Destination Left-Side Clipping

I/O Port Address: 3CFh
Index: 2Fh
MM/IO Offset: 17

Bit	Description
7	Reserved
6	System-to-Screen Dword [1]
5	System-to-Screen Dword [0]
4	Packed-24 Write Mask [1]
3	Packed-24 Write Mask [0]
2	Destination Write Mask [2]
1	Destination Write Mask [1]
0	Destination Write Mask [0]

This register contains a 3-bit field that suppresses writing to the first ‘n’ pixels (or bytes) of color-expanded data to an aligned destination area. This supports color expansion with left-edge clipping. This register also contains a 2-bit field that specifies the byte alignment of the first byte for each scanline.

For Packed-24 modes only, this register contains a 2-bit field that expands the write mask to 5 bits.

Bit	Description
7	**Reserved**
6:5	**System-to-Screen Dword Pointer [1:0]:** This 2-bit field specifies the position of the first byte within the first dword of each destination scanline. This allows data that is unaligned in system memory to be transferred without the overhead of unaligned bus cycles. If this field is programmed to a non-zero value for a color-expand BitBLT, GR33[0] must be programmed to ‘1’.
4:3	**Packed-24 Byte Write Mask [1:0]:** This 2-bit field expands the write mask contained in bit 2:0. This is for color expansion in Packed-24 modes only. The resulting 5-bit field is a byte mask.
2:0	**Destination Write Mask [2:0]:** This 3-bit field can prevent writing the first n pixels (up to seven) of each scanline for a color-expanded or pattern copy BitBLT. This field is expanded to 5 bits with bits 4:3 in Packed-24 modes. In these modes, the field is a count of bytes, not pixels. If GR33[3] is ‘1’, this field suppresses writing background pixels only for a color expand BitBLT.

5.9 GR30: BLT Mode

I/O Port Address: 3CFh
Index: 30h
MM/IO Offset: 18

Bit	Description
7	Enable Color Expand
6	Enable 8 × 8 Pattern Copy
5	Color Expand Width [1]
4	Color Expand Width [0]
3	Enable Transparency
2	BLT Source Display/System Memory
1	Reserved
0	BLT Direction

This register contains the bits that specify the details of the BitBLT, but not the ROP.

Bit **Description**

7 **Enable Color Expand:** If this bit is set to '1', the ROP source is the expanded result from the bit-mapped source.

The direction must be 'increment'. Registers GR0–GR1, and GR10–GR15 contain the foreground and background colors. All ROPs that use a source are available.

When the source data is expanded, the most-significant bit of the first source byte is expanded to the first pixel in the destination. When the source of color-expand data is display memory, the source pitch is ignored.

If this bit is set to '0', the ROP source is the pixel data read from the source.

6 **Enable 8 × 8 Pattern Copy:** If this bit is set to '1', the source pattern is repeatedly copied to the destination rectangular area. The source pattern must be aligned on a boundary equal to the size of the pattern. The source must be display memory and the ROP must use a source. The source is a linear string of bytes, in one of five arrangements as shown in the following table:

Case	Pattern	Pattern Alignment
Color expansion	8 bytes of bitmap for 64 pixels	Dword
8-bit pixels	64 bytes of color data for 64 pixels	64-byte boundary
16-bit pixels	128 bytes of color data for 64 pixels	128-byte boundary
24-bit pixels	8 × (24 bytes + 8 bytes filler per scanline)	256-byte boundary
32-bit pixels	256 bytes of color data for 64 pixels	256-byte boundary

5.9 GR30: BLT Mode *(cont.)*

Bit **Description**

5:4 **Color Expand Width [1:0]:** This 2-bit field controls the width of color expand BitBLTs (including solid color and pattern copy) according to the following table:

GR30[5]	GR30[4]	Color Expansion Width
0	0	8-bpp
0	1	16-bpp
1	0	24-bpp
1	1	32-bpp

3 **Enable Transparency:** If color expansion is enabled, this bit controls color expansion with transparency. If this bit is set to '1', then zeroes in the monochrome image being expanded result in the corresponding pixel not being written to; only the foreground is written.

If this bit is set to '0', then zeroes in the monochrome image being expanded result in the background color being written to the corresponding pixel. When expanding to 24-bpp, transparency *must* be enabled (this bit must be '1').

If color expansion is not enabled, this bit enables source transparency. The pixel about to be written is compared to the transparency color in registers GR34–GR35. If the pixel matches in all 8 or 16 bits, it is not written. This can only be used for 8- and 16-bpp Graphics modes.

2 **BLT Source Display/System Memory:** If this bit is set to '1', the BitBLT source is system memory rather than display memory. The CPU performs the system bus transfers; the CL-GD5446 ignores the address provided with these transfers. The CPU must use dword transfers. For system-to-screen BitBLTs without color expansion, up to three bytes of the last dword for each scanline is ignored. For system-to-screen BitBLTs with color expansion, bits or bytes left over at the end of a scanline are managed according to the state of register GR33[0]. The transfers must always be programmed as dword operations.

If this bit is set to '0', the BitBLT source is display memory. Regardless of the source, the destination is always display memory.

1 **Reserved:** This bit must be set to '0'.

0 **BLT Direction:** If this bit is set to '1', the source and destination addresses are decremented rather than incremented as the BitBLT proceeds. That is, the operation proceeds from right-to-left and bottom-to-top. The starting address is the highest addressed byte in each area. Neither color expansion, solid color fill, source transparency, or pattern copy can be used with decreasing addresses.

5.10 GR31: BLT Start/Status

I/O Port Address: 3CFh
Index: 31h
MM/IO Offset: 40

Bit	Description
7	Enable Autostart
6	System Source Location
5	Pause
4	Buffered Register Status (Read-only)
3	Reserved
2	BLT Reset
1	BLT Start
0	BLT Status (Read-only)

This register contains bits that control the BitBLT and some status bits.

Bit	Description
7	**Enable Autostart:** If this bit is set to '1', a BitBLT automatically starts whenever the engine is not busy and a set of parameters is available in the buffered registers. Programming this bit to '1' also write-enables register GR33. If this bit is set to '0', one BitBLT starts each time GR31[1] is set to '1'.
6	**System Source Location (Revision A):** If this bit is '1', the CL-GD546X responds to write accesses at 000BC000h–000BFFFFh for color-expand BitBLTs. This frees the linear address apertures for other, concurrent accesses. If this bit is '0', the CL-GD546X uses the linear aperture for BitBLTs (compatible with CL-GD543X/'4X). **System Source Location (Revision B):** If this bit is '1', system-to-screen BitBLTs use the second 16-Mbyte window specified in PCI10. This allows direct frame buffer accesses in the first window to be mixed with system-to-screen writes in the second window without restrictions. If a system-to-screen BitBLT requiring data is not active, writes to the second window complete in the minimum time and the data is discarded. Writes to the first window are ignored by the BitBLT engine (but are taken as direct writes to the frame buffer). If this bit is '0', system-to-screen BitBLTs use the first 16-Mbyte window (compatible with CL-GD543X/'4X).

5.10 GR31: BLT Start/Status *(cont.)*

Bit	Description
5	**Pause:** If this bit is set to '1', a system-to-screen BitBLT pauses. Writes to the display memory address range are considered ordinary display memory writes; the address supplied by the processor is the entire address. This can be used to change the hardware cursor in response to a mouse interrupt during a system-to-screen BitBLT. Reads are always permitted and return valid data. If this bit is set to '0', writes to the display memory address range are sent to the BitBLT engine (that is, if a system-to-screen BitBLT requires additional data to complete). When setting and resetting this bit, do not program data bit 1 to '1'.
4	**Buffered Register Status (Read-only):** This bit is set to '1' if the buffered registers are loaded and waiting for their BitBLT to start. This bit is reset to '0' if the buffered registers are available.
3	**Reserved**
2	**BLT Reset:** If this bit is set to '1', the entire BitBLT engine immediately resets and any operation in progress terminates. The operation *cannot* be restarted.
1	**BLT Start:** When this bit is set to '1', the BitBLT begins with the next available display memory cycle. This bit is cleared to '0' when the BitBLT is complete.
0	**BLT Status (Read-only):** If this bit is '1', the BitBLT is in progress. If this bit is '0', the BitBLT is complete or successfully suspended.

5.11 GR32: BLT ROP (Raster Operation)

I/O Port Address: 3CFh
Index: 32h
MM/IO Offset: 1A

Bit	Description
7	f [7]
6	f [6]
5	f [5]
4	f [4]
3	f [3]
2	f [2]
1	f [1]
0	f [0]

This register selects one of 16 two-operand ROPs. ROPs that do not use the source (such as ~D) must not be used when color expansion is selected.

Bit	Description
7:0	**f [7:0]:** This eight-bit value selects a two-operand ROP, as indicated in the following table. Note that the value programmed into register GR32 is identical for the cases of source/pattern where the actual logical operation is the same. This table is ordered by Microsoft ROP.

Z RPN	Z	ROP (hex)	Microsoft® Name	Microsoft® ROP
0	0	00	BLACKNESS	00000042
DPon	~P.~D	90	–	000500A9
DPna	~P.D	50	–	000A0329
Pn	~P	D0	–	000F0001
DSon	~S.~D	90	NOTSRCERASE	001100A6
DSna	~S.D	50	–	00220326
Sn	~S	D0	NOTSRCCOPY	00330008
SDna	S.~D	09	SRCERASE	00440328
PDna	P.~D	09	–	00500325
Dn	~D	0B	DSTINVERT	00550009
DPx	P~=D	59	PATINVERT	005A0049
DPan	~P+~D	DA	–	005F00E9
DSx	S~=D	59	SRCINVERT	00660046

5.11 GR32: BLT ROP (Raster Operation) *(cont.)*

Bit **Description**

7:0 *(cont.)*

Z RPN	Z	ROP (hex.)	Microsoft® Name	Microsoft® ROP
DSan	~S+~D	DA	–	007700E6
DSa	S.D	05	SRCAND	008800C6
DSxn	S=D	95	–	00990066
DPa	P.D	05	–	00A000C9
PDxn	P=D	95	–	00A50065
D	D	06	–	00AA0029
DPno	~P+D	D6	–	00AF0229
DSno	~S+D	D6	MERGEPAINT	00BB0226
S	S	0D	SRCCOPY	00CC0020
SDno	S+~D	AD	–	00DD0228
DSo	S+D	6D	SRCPAINT	00EE0086
P	P	0D	PATCOPY	00F00021
PDno	P+~D	AD	–	00F50225
DPo	P+D	6D	–	00FA0089
1	1	0E	WHITENESS	00FF0062

NOTE: In the first two columns, **D** denotes destination, **S** denotes source, and **P** denotes pattern.
The first column is RPN (reverse polish notation). **a** denotes 'and', **o** denotes 'or', **x** denotes 'exclusive or', and **n** denotes 'not'. For example, the second entry, DPon, would be interpreted as follows: Destination, (enter), Pattern, oR, nOT.
The second column is provided for those who prefer to avoid RPN. **~** denotes 'not',**.** denotes 'and', and **+** denotes 'or'. For example, the second entry would be interpreted as follows: NOT pattern AND NOT destination.

5.12 GR33: BLT Mode Extensions

I/O Port Address: 3CFh
Index: 33h
MM/IO Offset: 1B

Bit	Description
7:5	Reserved
4	Enable BLT Synchronous Display Switching
3	Enable Background-only Clipping
2	Enable Solid Color Fill
1	Invert Color Expand Source Sense
0	Source Data Granularity

This register contains some extended mode controls. This register is write-protected unless GR31[7] and CR5E[5] both = 1.

Bit	Description
7:5	**Reserved**
4	**Enable BLT Synchronous Display Switching:** If this bit is '1', the CL-GD5446 switches display buffers at the end of each BitBLT (if CR5E[2:0] is programmed to '111b'). The two display buffers begin at frame buffer location '0' and the address programmed into Screen Start A (CRA, CRB, and so on).
3	**Enabled Background-only Clipping:** If this bit is '1', the clipping specified in register GR2F only suppresses writing background pixels. This only applies to color-expanded BitBLTs.
2	**Enable Solid Color Fill:** If this bit is set to '1', the destination area (rectangle) is filled with the foreground color. GR30[7] and GR30[6] must both be set to '1'. GR30[3] must be set to '0' (no transparency). This function yields identical results as a color expanded, pattern-fill BitBLT with a pattern of all '1's, but is faster. If this bit is set to '0', this function is not enabled.
1	**Invert Color Expand Source Sense:** If this bit is set to '1', the sense of the CPU data for a color expanded BitBLT with transparency is inverted. A '1' causes the pixel to not be written; a '0' cause the foreground color to be written.
0	**Source Data Granularity:** If this bit is set to '1', dword granularity is enabled for color expanded system-to-screen BitBLTs. At the end of each scanline, unused source data is discarded to the end of the current dword. The next scanline begins with the next dword. Up to 31 bits can be discarded. If this bit is set to '0', unused source data to the end of the current byte are discarded at the end of each scanline. This affects color expanded system-to-screen Bit-BLTs only. Non-color expansion system-to-screen BitBLTs always discard to the end of the current dword. This bit must be set to '1' for any color expanded BitBLT where the GR2F[6:5] field is a non-zero value.

5.13 GR34–GR35: Transparent BLT Key Color Byte 0, 1

I/O Port Address: 3CFh
Index: 34h, 35h
MM/IO Offset: 1C, 1D

Bit	Description
7	Transparent BLT Key Color [7] / [15]
6	Transparent BLT Key Color [6] / [14]
5	Transparent BLT Key Color [5] / [13]
4	Transparent BLT Key Color [4] / [12]
3	Transparent BLT Key Color [3] / [11]
2	Transparent BLT Key Color [2] / [10]
1	Transparent BLT Key Color [1] / [9]
0	Transparent BLT Key Color [0] / [8]

These registers contain the Transparent BLT Key Color for 8- or 16-bpp BitBLTs with transparency and without color expansion.

Bit	Description
7:0	**Transparent BLT Key Color [15:0]:** If GR30[3] is ‘1’ and GR30[7] is ‘0’, the value of each pixel about to be written in the frame buffer is compared with the contents of these two registers. If the pixel matches in all 8 or 16 bits, it is not written. This mode can only be used for 8- or 16-bpp BitBLTs. If this mode is used for 8-bpp BitBLTs, registers GR34–GR35 must be programmed identically. If this mode is used for 16-bpp BitBLTs, register GR34 is the low byte and register GR35 is the high byte.

Video Capture and Playback Registers

6. VIDEO CAPTURE AND PLAYBACK REGISTERS

The CL-GD5446 Video Capture and Playback registers are summarized in Table 6-1.

Table 6-1. Video Capture and Playback Registers Quick Reference

Abbreviation	Register Name	Index	Port	Page
CR31	Video Window Horizontal Zoom Control	31h	3D5h	6-4
CR32	Video Window Vertical Zoom Control	32h	3D5h	6-5
CR33	Video Window Horizontal Region 1 Size	33h	3D5h	6-6
CR34	Video Window Region 2 Width	34h	3D5h	6-7
CR35	Video Window Region 2 Source Data Size	35h	3D5h	6-8
CR36	Video Window Horizontal Overflow	36h	3D5h	6-9
CR37	Video Window Vertical Start	37h	3D5h	6-10
CR38	Video Window Vertical End	38h	3D5h	6-11
CR39	Video Window Vertical Overflow	39h	3D5h	6-12
CR3A	Video Buffer 1 Start Address Byte 0	3Ah	3D5h	6-13
CR3B	Video Buffer 1 Start Address Byte 1	3Bh	3D5h	6-13
CR3C	Video Buffer 1 Start Address Byte 2	3Ch	3D5h	6-14
CR3D	Video Window Address Offset	3Dh	3D5h	6-15
CR3E	Video Window Master Control	3Eh	3D5h	6-16
CR3F	Miscellaneous Video Control	3Fh	3D5h	6-18
CR50	Video Capture Control	50h	3D5h	6-20
CR51	Video Capture Data Format	51h	3D5h	6-22
CR52	Video Capture Horizontal Data Reduction	52h	3D5h	6-23
CR53	Video Capture Vertical Data Reduction	53h	3D5h	6-24
CR54	Video Capture Horizontal Delay	54h	3D5h	6-25
CR56	Video Capture Vertical Delay	56h	3D5h	6-26
CR57	Video Capture Maximum Height	57h	3D5h	6-27
CR58	Video Capture Miscellaneous Control	58h	3D5h	6-28
CR59	Video Buffer 2 Start Address Byte 0	59h	3D5h	6-29
CR5A	Video Buffer 2 Start Address Byte 1	5Ah	3D5h	6-29
CR5B	Video Window Brightness Adjust	5Bh	3D5h	6-30
CR5C	Luminance-only Capture Control	5Ch	3D5h	6-31

Table 6-1. Video Capture and Playback Registers Quick Reference *(cont.)*

Abbreviation	Register Name	Index	Port	Page
CR5D	Video Window Pixel Alignment	5Dh	3D5h	6-32
CR5E	Double-Buffer Control	5Eh	3D5h	6-33
GR1C	Chroma Key: U/Green Minimum	1Ch	3DFh	6-35
GR1D	Chroma Key: U/Green Maximum	1Dh	3D5h	6-35
GR1E	Chroma Key: V/Blue Minimum	1Eh	3D5h	6-35
GR1F	Chroma Key: V/Blue Maximum	1Fh	3D5h	6-35

6.1 CR31: Video Window Horizontal Zoom Control

I/O Port Address: 3D5h
Index: 31h

Bit	Description
7	Horizontal Zoom Code [7]
6	Horizontal Zoom Code [6]
5	Horizontal Zoom Code [5]
4	Horizontal Zoom Code [4]
3	Horizontal Zoom Code [3]
2	Horizontal Zoom Code [2]
1	Horizontal Zoom Code [1]
0	Horizontal Zoom Code [0]

This register controls the horizontal zooming of the video window.

Bit	Description
7:0	**Horizontal Zoom Control [7:0]:** This field controls the horizontal zooming of the video window. Continuous horizontal zooming from 1× to 4× is possible, subject to the restrictions noted in Chapter 9, “Programming Notes”. The zoom factor is determined by Equation 6-1.

$$HZoomFactor = \frac{256}{HorZoomCode}$$ **Equation 6-1**

where,
values in the range 64 through 255 can be used for *HorZoomCode*.
This provides continuous zooming from 4 × (*HorZoomCode* = 64) to slightly more than 1 × (*HorZoomCode* = 255). A ‘0’ value sets 1×. A value of less than 64 results in a zoom factor of more than 4×, but the horizontal interpolation does not provide further benefit.

6.2 CR32: Video Window Vertical Zoom Control

I/O Port Address: 3D5h
Index: 32h

Bit	Description
7	Vertical Zoom Code [7]
6	Vertical Zoom Code [6]
5	Vertical Zoom Code [5]
4	Vertical Zoom Code [4]
3	Vertical Zoom Code [3]
2	Vertical Zoom Code [2]
1	Vertical Zoom Code [0]
0	Vertical Zoom Code [0]

This register controls the vertical zooming in the video window.

Bit	Description
7:0	**Vertical Zoom Control [7:0]:** This field controls the vertical zooming of the video window. Continuous vertical zooming from 1× to 4× is possible. The zoom factor is determined by Equation 6-2.

$$VZoomFactor = \frac{256}{VertZoomCode}$$ **Equation 6-2**

where,
values in the range 64 through 255 can be used for *VertZoomCode*.
This provides continuous zooming from 4 × (*VertZoomCode* = 64) to slightly more than 1 × (*VertZoomCode* = 255). A '0' value sets 1×.

If a vertical zoom of greater than 1× is selected, some scanlines are generated by interpolation of source image scanlines (or by replication). The physical height of the window on the screen is fixed by the Vertical Start and Vertical End values.

Vertical zooming with interpolation requires twice the normal video bandwidth. This places restrictions on when zooming by interpolation can be used. See Chapter 9, "Programming Notes", for further details.

6.3 CR33: Video Window Horizontal Region 1 Size

I/O Port Address: 3D5h
Index: 33h

Bit	Description
7	R1SZ [7]
6	R1SZ [6]
5	R1SZ [5]
4	R1SZ [4]
3	R1SZ [3]
2	R1SZ [2]
1	R1SZ [1]
0	R1SZ [0]

This register contains the least-significant bits of the Video Window Horizontal Region 1 Size.

Bit	Description
7:0	**R1SZ [7:0]:** The Horizontal Region 1 Size specifies the size of the background to the left of the video window. This register is extended to 10 bits with CR36[1:0]. The units are dwords of Graphics data transferred to the CRT FIFO from the start of a scanline to the left boundary of the video window (that is, the last pixel before the window begins). Use Equation 6-3 to determine the location of the first pixel of the video window.

$$Region1Size = \left(\frac{32}{GraphicsBPP} \bullet R1SZ\right) + \left(\frac{R1Adjust \bullet 8}{GraphicsBPP}\right) \quad \textbf{Equation 6-3}$$

Use Equation 6-4 to calculate the value that should be programmed into the field, given the desired left edge.

$$R1SZ = \frac{Region1SizeinPixels}{GraphicsPixelsPerDWORD} \quad \textbf{Equation 6-4}$$

Refer to Chapter 9, “Programming Notes”, for detailed information.

6.4 CR34: Video Window Region 2 Width

I/O Port Address: 3D5h
Index: 34h

Bit	Description
7	R2SZ [7]
6	R2SZ [6]
5	R2SZ [5]
4	R2SZ [4]
3	R2SZ [3]
2	R2SZ [2]
1	R2SZ [1]
0	R2SZ [0]

This register contains the least-significant bits of the Video Window Horizontal Region 2 width.

Bit	Description
7:0	**R2SZ [7:0]:** The Horizontal Region 2 Width is the size of the overlaid region in the background that is skipped for each scanline. This field is extended to 10 bits with CR36[3:2]. The units are the dwords that would have been transferred to the FIFO, but are not because the corresponding region on the screen displayed the window. Use Equation 6-5 to calculate the actual size in pixels.

$$Region2Size = \left(\frac{32}{GraphicsBPP} \bullet R2SZ\right) + \left(\frac{R2Adjust \bullet 8}{GraphicsBPP}\right) \quad \textbf{Equation 6-5}$$

Use Equation 6-6 to calculate the value to be programmed given the desired window width.

$$R2SZ = \frac{WindowWidthInPixels}{GraphicsPixelsPerDWORD} \quad \textbf{Equation 6-6}$$

See Chapter 9, “Programming Notes”, for details.

6.5 CR35: Video Window Region 2 Source Data Size

I/O Port Address: 3D5h
Index: 35h

Bit	Description
7	R2SDZ [7]
6	R2SDZ [6]
5	R2SDZ [5]
4	R2SDZ [4]
3	R2SDZ [3]
2	R2SDZ [2]
1	R2SDZ [1]
0	R2SDZ [0]

This register contains the least-significant bits of the Video Window Region 2 Source Data Size.

Bit	Description
7:0	**HR2ASZ [7:0]:** The Horizontal Region 2 Source Data Size specifies the number of dwords of graphics data that must be fetched for the video window. This is extended to 10 bits with CR36[5:4]. This value can be calculated by evaluating Equation 6-7.

$$Region2SDSize = \frac{Region2SizeinPixels}{\frac{32}{VideoBPP} \bullet HorizontalZoomFactor}$$ **Equation 6-7**

See Chapter 9, “Programming Notes”, for details.

6.6 CR36: Video Window Horizontal Overflow

I/O Port Address: 3D5h
Index: 36h

Bit	Description
7:6	Reserved
5	R2SDZ [9]
4	R2SDZ [8]
3	R2SZ [9]
2	R2SZ [8]
1	R1SZ [9]
0	R1SZ [8]

This register contains bits that extend each of the three horizontal fields to 10 bits.

Bit	Description
7:6	**Reserved**
5:4	**R2ASDZ [9:8]:** These two bits extend the Region 2 Source Data Size to 10 bits. The least-significant eight bits are contained in register CR35.
3:2	**R2SZ [9:8]:** These two bits extend the Region 2 Size to 10 bits. The least-significant eight bits are contained in register CR34.
1:0	**R1SZ [9:8]:** These two bits extend the Region 1 Size to 10 bits. The least-significant eight bits are contained in register CR33.

6.7 CR37: Video Window Vertical Start

I/O Port Address: 3D5h
Index: 37h

Bit	Description
7	WVS [7]
6	WVS [6]
5	WVS [5]
4	WVS [4]
3	WVS [3]
2	WVS [2]
1	WVS [1]
0	WVS [0]

This register contains the least-significant bits of the Video Window Vertical Start.

Bit	Description
7:0	**WVS [7:0]:** The Window Vertical Start specifies the first scanline that contains the window. This is extended to 10 bits with CR39[1:0]. Combined with the Window Vertical End value, this defines the vertical extent of the window.

6.8 CR38: Video Window Vertical End

I/O Port Address: 3D5h
Index: 38h

Bit	Description
7	WVE [7]
6	WVE [7]
5	WVE [7]
4	WVE [7]
3	WVE [7]
2	WVE [7]
1	WVE [7]
0	WVE [7]

This register contains the least-significant bits of the Video Window Vertical End.

Bit	Description
7:0	**WVS [7:0]:** The Window Vertical End specifies the last scanline that contains the window. This field is extended to 10 bits with CR39[3:2]. Combined with the Window Vertical Start value, this defines the vertical extent of the window.

6.9 CR39: Video Window Vertical Overflow

I/O Port Address: 3D5h
Index: 39h

Bit	Description
7:4	Reserved
3	WVE [9]
2	WVE [8]
1	WVS [9]
0	WVS [8]

This register contains bits that extend each of two vertical fields to 10 bits.

Bit	Description
7:4	**Reserved**
3:2	**WVE [9:8]:** These two bits extend the Window Vertical End to 10 bits. The least-significant eight bits are contained in register CR38.
1:0	**WVS [9:8]:** These two bits extend the Window Vertical Start to 10 bits. The least-significant eight bits are contained in register CR37.

6.10 CR3A–CR3B: Video Buffer 1 Start Address Byte 0, 1

I/O Port Address: 3D5h
Index: 3Ah, 3Bh

Bit	Description
7	Video Buffer 1 Start Address [9] / [17]
6	Video Buffer 1 Start Address [8] / [16]
5	Video Buffer 1 Start Address [7] / [15]
4	Video Buffer 1 Start Address [6] / [14]
3	Video Buffer 1 Start Address [5] / [13]
2	Video Buffer 1 Start Address [4] / [12]
1	Video Buffer 1 Start Address [3] / [11]
0	Video Buffer 1 Start Address [2] / [10]

These two registers contain the least-significant bits of the Video Buffer 1 Start Address.

Bit **Description**

15:0 **Video Buffer 1 Address [17:2]:** This field contains bits [17:2] of the Video Buffer 1 Start Address. This is extended to 22 bits with CR5D[3:2] and CR3C[3:0] as shown in the following table:

Address Bits Correspond to 4-Mbyte Linear Address	21:18	17:10	9:2	1:0
Video Buffer 1 (default display buffer)	CR3C[3:0]	CR3B[7:0]	CR3A[7:0]	CR5D[3:2]
Video Buffer 2 (default capture buffer)	CR58[3:0]	CR5A[7:0]	CR59[7:0]	CR5D[3:2]

This buffer is the default video display buffer. If video double buffering is being used, this buffer is alternately used for video capture and video display. See Chapter 9 for details.

The start addresses of both video buffers are defined in terms of bytes. For video capture, data is always written on dword or qword boundaries (CR5D[1:0] has no effect for capture writes).

See Chapter 9, “Programming Notes”, for details.

6.11 CR3C: Video Buffer 1 Start Address Byte 2

I/O Port Address: 3D5h
Index: 3Ch

Bit	Description
7:6	Reserved
5	Video Address Offset [11]
4	Reserved
3	Video Buffer 1 Start Address [21]
2	Video Buffer 1 Start Address [20]
1	Video Buffer 1 Start Address [19]
0	Video Buffer 1 Start Address [18]

This register contains the three most-significant bits of the Video Buffer 1 Start Address field and an upper bit of the video address offset.

Bit	Description
7:6	**Reserved**
5	**Video Address Offset [11]:** This bit extends the Video Address Offset to 10 bits. The least-significant bits are contained in register CR3D.
4	**Reserved**
3:0	**Video Buffer 1 Address [21:18]:** These four bits extend the Video Window Start Address to 22 bits. The least-significant bits are contained in registers CR3A, CR3B, and CR5D. See Section 6.10 to see the complete 22-bit address.

6.12 CR3D: Video Buffer Address Offset

I/O Port Address: 3D5h
Index: 3Dh

Bit	Description
7	Video Buffer Address Offset [10]
6	Video Buffer Address Offset [9]
5	Video Buffer Address Offset [8]
4	Video Buffer Address Offset [7]
3	Video Buffer Address Offset [6]
2	Video Buffer Address Offset [5]
1	Video Buffer Address Offset [4]
0	Video Buffer Address Offset [3]

This register contains eight bits of the Video Buffer Address Offset.

Bit	Description
7:0	**Video Address Offset [10:3]:** This value is for both the video window and video capture. The three least-significant bits are always considered to be ‘0’. This field is extended with CR3C[5]. This 12-bit value specifies the distance in display memory (in bytes) between vertically adjacent pixels. This value is added to the video window address at the end of each scanline within the window to obtain the beginning address of the next scanline. When interleave addressing is used for video capture, this value is multiplied by two. When CR5C[7] is ‘1’, the value 1024 is the capture buffer offset, regardless of the programming of this field.

Address Bits Correspond to Linear Address	11	10:3	2:0
Register	CR3C[5]	CR3D[7:0]	0

6.13 CR3E: Video Window Master Control

I/O Port Address: 3D5h
Index: 3Eh

Bit	Description
7	Occlusion Enable
6	Reserved
5	Error Diffusion Enable
4	Vertical Zoom Mode
3	Video Display Format [2]
2	Video Display Format [1]
1	Video Display Format [0]
0	Video Window Master Enable

This register configures the video window for the various modes of operation.

Bit	Description
7	**Occlusion Enable:** If this bit is set to '1', information in the video window can be overlaid on a pixel-by-pixel basis. This can place video on graphics or graphics on video. See Chapter 9, "Programming Notes", for details.
6	**Reserved**
5	**Error Diffusion Enable:** If this bit is set to '1', error diffusion is enabled for Accu-Pak-to-YCrCb 4:4:2 conversion. This reduces the contouring that results from the truncated luminance value by inserting random values in place of the missing LSBs. If this bit is set to '0', error diffusion is not enabled.
4	**Vertical Zoom Mode:** If this bit is set to '1', line replication is used for vertical zooming. If this bit is set to '0', interpolation is used for vertical zooming. Vertical interpolation and occlusion are mutually exclusive.

6.13 CR3E: Video Window Master Control *(cont.)*

Bit **Description**

3:1 **Video Display Format [2:0]:** This 3-bit field specifies the format of the video displayed in the video window. The following table shows the encoding.

CR3E[3:1]	Format	Note
000	YUV 4:2:2	Excess 128 Cr, Cb
001	AccuPak™	Encoded from YCrCb, Excess 128 Cr, Cb
010	8-bit LUT	Graphics format must be RGB 16
011	Reserved	
100	RGB 5:5:5	Also decimate if CR3F[4] = 1
101	RGB 5:6:5	Also decimate if CR3F[4] = 1
110	Reserved	
111	Reserved	

0 **Video Window Master Enable:** If this bit is set to '1', the video window is enabled and displays as configured. If this bit is set to '0', the video window is not enabled. When this bit is set to '1', the video window is enabled at the next leading edge of VSYNC. This avoids the requirement for the software to synchronize with VSYNC. When this bit is set to '0', the video window is immediately disabled.

6.14 CR3F: Miscellaneous Video Control

I/O Port Address: 3D5h
Index: 3Fh

Bit	Description
7	VREF State (R/O)
6	Field ID (R/O)
5	INTR# Pin Source Select
4	Auto-Decimation/YUV12 Support Enable
3	Reserved
2	Current Capture Buffer
1	Interrupt Control
0	Auto-Decimation Memory Page Bit

This register contains bits for control/status of miscellaneous video functions.

Bit	Description
7	**VREF State (R/O):** This read-only bit follows the Video Port VREF signal (EDCLK#).
6	**Field ID (R/O):** This read-only bit returns the ID of the current field (odd/even). This bit is updated on the active edge of VREF.
5	**INTR# Pin Source Select:** This bit controls the internal interrupt request. If this bit is '0', the VGA VSYNC is the interrupt request. If this bit is '1', the video port VREF is the interrupt request.
4	**Auto-Decimation Enable:** When this bit is '1' and CR3E[3:1] is a non-zero value, auto-decimation is enabled when the video window display mode is RGB. For writes to the frame buffer video aperture (the fourth aperture), every high word (enabled by C/BE[3:2]) is ignored. The address offset from the beginning of the aperture is divided by two. Byte writes are not supported; word and dword writes are supported. This allows reduced video data and reduced bandwidth. Program the horizontal zoom factor to two. This bit also enables YUV12 planar assist, if CR3E[3:1] is '000'.
3	**Reserved:** This bit must always be programmed to '0'.

6.14 CR3F: Miscellaneous Video Control *(cont.)*

Bit **Description**

2 **Current Capture Buffer:** This read-only bit indicates which video buffer is being used as the capture buffer at the last valid falling edge of VREF.

CR3F[2]	Capture Buffer	Buffer Pointers
0	Video buffer 2	CR59/CR5A
1	Video buffer 1	CR3A/CR3B

1 **Interrupt Control:** If this bit is '1', the V-Port interrupt is enabled. If this bit is programmed to '0', the interrupt pending bit in GR17[4] is cleared and the interrupt is forced inactive. After programming this bit to '0', the application must program it to '1' if the next VREF is to generate an interrupt. The interrupt request is generated on the active edge of VREF.

0 **Auto-Decimation Memory Page Bit:** This bit replaces internal memory address A21 to select the upper- or lower-2 Mbytes of frame buffer for the 2:1 horizontally decimated RGB data from the video aperture.

6.15 CR50: Video Capture Control

I/O Port Address: 3D5h
Index: 50h

Bit	Description
7	Capture Alternate Fields
6	Interlaced Video Capture Mode
5	Invert Capture Clock
4	Enable 16-bit Capture Port
3	Enable Double Edge Capture Clock
2	Capture Alternate Fields
1	Video Pins Configuration [1]
0	Video Pins Configuration [0]

This register controls the video capture port.

Bit	Description
7	**Capture Alternate Fields:** This bit, in conjunction with CR50[2], specifies which video fields to capture.
6	**Interlaced Video Capture Mode:** If this bit is '1', the interlaced capture addresses is generated. Odd fields are captured beginning at the capture buffer address plus the capture offset. Scanlines are separated by twice the capture offset.
5	**Invert Capture Clock:** If this bit is '1', the capture clock input on the DCLK pin is inverted internally. This can help with some data timing problems.
4	**Enable 16-bit Capture Port:** If this bit is '1', the capture port is configured for 16 bits.
3	**Enable Double Edge Capture Clock:** If this bit is '1', the V-Port is configured to capture data on both edges of DCLK. This provides for capturing 16-bit data using an 8-bit port.

6.15 CR50: Video Capture Control *(cont.)*

Bit **Description**

2 **Capture Alternate Fields:** This bit, with CR50[7], specifies which video fields to capture.

CR51[3]	CR50[7]	CR50[2]	Capture
0	X	X	Capture disabled
1	0	0	Capture every field
1	0	1	Capture alternate fields
1	1	0	Capture odd (or alternate) fields
1	1	1	Capture alternate odd (every fourth) field

1:0 **Video Pins Configuration:** This field controls the configuration of the video pins. These pins are P[15:0], ESYNC#, EVIDEO#, EDCLK#, DCLK, and BLANK#.

CR50[1:0]	Configuration
00	Standard feature connector
01	Reserved
10	Video capture: rising edge on HREF starts line
11	Video capture: falling edge on HREF starts line

6.16 CR51: Video Capture Data Format

I/O Port Address: 3D5h
Index: 51h

Bit	Description
7:4	Reserved
3	Enable Video Capture
2	Video Capture Format [2]
1	Video Capture Format [1]
0	Video Capture Format [0]

This register controls the video capture port.

Bit	Description
7:4	**Reserved**
3	**Enable Video Capture:** When this bit is '1', video capture is enabled. The CL-GD5446 captures fields of video as specified in CR50[7], CR50[2], and CR58[6]. Video capture does not start in the middle of a field; this bit is examined on VREF falling edges.
2:0	**Video Capture Format:** This field specifies the video capture format, as defined in the following table:

CR51[2:0]	Format	Note
000	YUV16	
001	RGB16	5:5:5 or 5:6:5
010	AccuPak™	Decimation not supported
011	YUV-to-AccuPak™	Decimation not supported
100	Reserved	
101	Reserved	
110	Reserved	
111	'Y' data capture only	Used for luminance-only capture without video display. Requires YUV 4:2:2 data input

6.17 CR52: Video Capture Horizontal Data Reduction

I/O Port Address: 3D5h
Index: 52h

Bit	Description
7:3	Reserved
2	Horizontal Data Reduction Control [2]
1	Horizontal Data Reduction Control [1]
0	Horizontal Data Reduction Control [0]

This register controls data reduction in the horizontal dimension.

Bit	Description
7:3	**Reserved**
2:0	**Horizontal Data Reduction Control [2:0]:** When video is being captured, pixels are kept as shown in the following table; all other pixels are discarded. This is only used when the capture format is RGB and YUV.

CR52[2:0]	Data Reduction
000	Disable horizontal data reduction (keep all active pixels)
001	Keep every second pixel
010	Keep every fourth pixel
011	Keep every 8th pixel
100	Keep every 16th pixel
101	Keep every 32nd pixel
110	Reserved
111	Reserved

6.18 CR53: Video Capture Vertical Data Reduction

I/O Port Address: 3D5h
Index: 53h

Bit	Description
7:3	Reserved
2	Vertical Data Reduction Control [2]
1	Vertical Data Reduction Control [1]
0	Vertical Data Reduction Control [0]

This register controls data reduction in the vertical dimension.

Bit	Description
7:3	**Reserved**
2:0	**Vertical Data Reduction Control [2:0]:** When video is being captured, scanlines are kept as shown in the following table; others are discarded. This can only be used when the capture format is RGB and YUV.

CR52[2:0]	Data Reduction
000	Disable vertical data reduction (keep all active scanlines)
001	Keep every second scanline
010	Keep every fourth scanline
011	Keep every 8th scanline
100	Keep every 16th scanline
101	Keep every 32nd scanline
110	Reserved
111	Reserved

6.19 CR54: Video Capture Horizontal Delay

I/O Port Address: 3D5h
Index: 54h

Bit	Description
7:5	Reserved
4	Video Capture Horizontal Delay [4]
3	Video Capture Horizontal Delay [3]
2	Video Capture Horizontal Delay [2]
1	Video Capture Horizontal Delay [1]
0	Video Capture Horizontal Delay [0]

This register controls left-edge clipping of captured data.

Bit	Description
7:5	**Reserved**
4:0	**Video Capture Horizontal Delay [4:0]:** These values set the number of valid PIXCLKS to count from HREF to the first captured data. This allows data on the left edge of each captured frame to be clipped (discarded).

6.20 CR56: Video Capture Vertical Delay

I/O Port Address: 3D5h
Index: 56h

Bit	Description
7:5	Reserved
4	Video Capture Vertical Delay [4]
3	Video Capture Vertical Delay [3]
2	Video Capture Vertical Delay [2]
1	Video Capture Vertical Delay [1]
0	Video Capture Vertical Delay [0]

This register controls clipping on the top of captured data, used in conjunction with luminance-only capture.

Bit	Description
7:5	**Reserved**
4:0	**Video Capture Vertical Delay [4:0]:** If luminance-only capture is enabled, this field specifies the number of scanlines of Y data to capture before switching to decimation or AccuPak conversion, or the scanline in the odd field where the luminance (Y) data is captured. If luminance-only capture is not enabled, this field specifies the number of scanlines skipped before video capture begins. These cases are summarized in the following table.

CR5C[6]	CR5C[5]	Case	CR56 Interpretation
X	0	Luminance-only capture disabled	Vertical Delay (top-edge clipping)
0	1	Luminance-only capture enabled	One scanline at register CR56 value
1	1	Luminance-only capture enabled	Y data to register CR56 value

6.21 CR57: Video Capture Maximum Height

I/O Port Address: 3D5h
Index: 57h

Bit	Description
7	Video Capture Maximum Height [7]
6	Video Capture Maximum Height [6]
5	Video Capture Maximum Height [5]
4	Video Capture Maximum Height [4]
3	Video Capture Maximum Height [3]
2	Video Capture Maximum Height [2]
1	Video Capture Maximum Height [1]
0	Video Capture Maximum Height [0]

This register specifies the maximum number of scanlines of video data to be captured (after vertical data reduction). This field is extended to nine bits with CR58[5] for a maximum of 511 scanlines of captured data.

Bit	Description
7:0	**Video Capture Maximum Height [4:0]:** This field specifies the maximum number of scanlines of video data per field. This field is extended to nine bits with CR58[5]. This is a count of actually captured scanlines and does not include lines clipped at the top, lines discarded due to vertical data reduction, or the scanlines skipped due to vertical data reduction.

6.22 CR58: Video Capture Miscellaneous Control

I/O Port Address: 3D5h
Index: 58h

Bit	Description
7	Reserved
6	Reverse Odd/Even Field Decode Sense
5	Video Capture Maximum Height [8]
4	Reserved
3	Video Buffer 2 Start Address [21]
2	Video Buffer 2 Start Address [20]
1	Video Buffer 2 Start Address [19]
0	Video Buffer 2 Start Address [18]

This register contains the four most-significant bits of the Video Buffer 2 start address.

Bit	Description
7	**Reserved**
6	**Reverse Odd/Even Field Decode Sense:** If this bit is '1', field decoder sense is inverted. See the timing diagram for Even/Odd Field timing in Chapter 11, "Electrical Specifications".
5	**Video Capture Maximum Height [8]:** This bit extends the field in register CR57 to nine bits.
4	**Reserved**
3:0	**Video Buffer 2 Start Address [21:18]:** This field extends the starting address of video buffer 2 to 22 bits. This is the default capture buffer. See registers CR59–CR5A.

6.23 CR59–CR5A: Video Buffer 2 Start Address Byte 0, 1

I/O Port Address: 3D5h
Index: 59h, 5Ah

Bit	Description
7	Video Buffer 2 Start Address [9] / [17]
6	Video Buffer 2 Start Address [9] / [16]
5	Video Buffer 2 Start Address [7] / [15]
4	Video Buffer 2 Start Address [6] / [14]
3	Video Buffer 2 Start Address [5] / [13]
2	Video Buffer 2 Start Address [4] / [12]
1	Video Buffer 2 Start Address [3] / [11]
0	Video Buffer 2 Start Address [2] / [10]

This register pair contains 16 bits of the Video Buffer 2 Start Address. This is the default video capture buffer.

Bit **Description**

7:0 **Video Buffer 2 Start Address [17:2]:** These two registers contain 16 bits of the Video Buffer 2 Start Address. The following table indicates how the bits are assigned.

Address Bits Correspond to 4-Mbyte Linear Address	21:18	17:10	9:3	2	1:0
Capture Buffer Case	CR58[3:0]	CR5A[7:0]	CR59[7:1]	0	0
Display Buffer Case	CR58[3:0]	CR5A[7:0]	CR59[7:1]	CR59[0]	CR5D[3:2]

If Video Buffer 2 is used as the capture buffer, CR59[0] is ignored and the capture buffer starts on a qword boundary. If Video Buffer 2 is used as the display buffer, CR59[0] is used and corresponds to linear address bit 2.

6.24 CR5B: Video Window Brightness Adjust

I/O Port Address: 3D5h
Index: 5Bh

Bit	Description
7:5	Reserved
4	Video Window Brightness Adjust [4]
3	Video Window Brightness Adjust [3]
2	Video Window Brightness Adjust [2]
1	Video Window Brightness Adjust [1]
0	Video Window Brightness Adjust [0]

The value in this register increases the luminance (brightness) of all YUV pixels in the video window.

Bit	Description
7:5	**Reserved**
4:0	**Video Window Brightness Adjust [4:0]:** This value is added to the Y data (luminance) for all pixels in the video window. If the result of the addition is greater than 255, it is forced to 255. This function is used when the Video Display Format (Section 6.13) is YUV. This feature allows a modest increase in brightness for MPEG playback.

6.25 CR5C: Luminance-Only Capture Control

I/O Port Address: 3D5h
Index: 5Ch

Bit	Description
7	Force Capture Buffer Address Offset
6	Luminance-only Capture [1]
5	Luminance-only Capture [0]
4	Enable Alternate FIFO Threshold Setting
3	Alternate FIFO Threshold Setting [3]
2	Alternate FIFO Threshold Setting [2]
1	Alternate FIFO Threshold Setting [1]
0	Alternate FIFO Threshold Setting [0]

This register controls luminance-only capture and provides an alternate FIFO threshold control.

Bit **Description**

7 **Force Capture Buffer Address Offset:** If this bit is '1', the capture buffer address offset is 1024. The value in register CR3D is not used for the capture buffer address offset, although it continues to be used for the display buffer address offset.

6:5 **Luminance-only Capture [1:0]:** This field, in conjunction with the Video Capture Vertical Delay field in register CR56, controls luminance-only capture.

CR5C[6]	CR5C[5]	Case	CR56 Interpretation
X	0	Luminance-only capture disabled	Vertical delay (top-edge clipping)
0	1	Luminance-only capture enabled	One scanline at register CR56 value
1	1	Luminance-only capture enabled	Y data to register CR56 value

4 **Enable Alternate FIFO Threshold Setting:** If this bit is '1', the alternate FIFO threshold setting in CR5C[3:0] is used for scanlines containing an active video window in place of SR16[3:0].

3:0 **Alternate FIFO Threshold Setting [3:0]:** This field is the alternate FIFO threshold setting. This field is typically set to a value higher than the primary threshold setting in SR16[3:0].

6.26 CR5D: Video Window Pixel Alignment

I/O Port Address: 3D5h
Index: 5Dh

Bit	Description
7:6	Reserved
5	R2Adjust [1]
4	R2Adjust [0]
3	Window Display Start Address [1]
2	Window Display Start Address [0]
1	R1Adjust [1]
0	R1Adjust [0]

This register contains fields that can adjust the start and size of the video window to any pixel boundary.

Bit	Description
7:6	**Reserved**
5:4	**R2Adjust [1:0]:** This field specifies the number of bytes to add to Region 2 (the video window itself). This field can adjust the width of the video window to any pixel width for 8- or 16-bpp graphics formats only.
3:2	**Window Display Start Address [1:0]:** This is the least-significant two bits of the video window display start address. These bits can be included in the address of video buffer 1 or video buffer 2, but only when the respective buffer is the display buffer.
1:0	**R1Adjust [1:0]:** This field specifies the number of bytes to add to Region 1 (the graphics area to the left of the video window). This field can position the video window at any pixel for 8- or 16-bpp graphics formats only.

6.27 CR5E: Double-Buffer Control

I/O Port Address: 3D5h
Index: 5Eh

Bit	Description
7:6	Reserved
5	Video Window Double Buffer Control [1]
4	Video Window Double Buffer Control [0]
3	Reserved
2	Graphics Double Buffer Control [2]
1	Graphics Double Buffer Control [1]
0	Graphics Double Buffer Control [0]

This register contains fields that control video double buffering and graphics double buffering.

Bit **Description**

7:6 **Reserved**

5:4 **Video Window Double Buffer Control [1:0]:** This field controls the two buffers used for video capture and video display. Refer to Section 6.23 for the definition of the two buffers.

CR5E[5:4]	Definition	Note
00	Video buffer 1 is display, video buffer 2 is capture	–
01	Video buffer 1 is capture, video buffer 2 is display	–
10	Auto-switch buffers on VREF	CR3F[2] indicates current capture buffer
11	Auto-switch buffer on BitBLT completion	GR33[4] = 1

3 **Reserved**

6.27 CR5E: Double-Buffer Control *(cont.)*

Bit **Description**

2:0 **Graphics Double Buffer Control [2:0]:** This field controls the two buffers used for graphics display. Graphics buffer 1 always starts at the beginning of the frame buffer; graphics buffer 2 starts at the location specified in the Screen Start A registers (CRC, CRD, and so on).

CR5E[2:0]	Function	Note
000	Compatible VGA display start address controls	–
001	VSYNC switching	–
010	Forces graphics buffer 1 as display	Beginning of frame buffer
011	Forces graphics buffer 2 as display	CRC, CRD, and so on
100	A18 controls switching (0 chooses buffer 2)	A[21:19] = 000
101	A19 controls switching (0 chooses buffer 2)	A[21:20] = 00
110	Reserved	–
111	BitBLT switches graphics buffer	GR33[4] = 1

6.28 GR1C–GR1F: Chroma Key

I/O Port Address: 3CFh
Index: 1Ch, 1Dh, 1Eh, 1Fh

Bit	Description
7	Chroma Key [7]
6	Chroma Key [6]
5	Chroma Key [5]
4	Chroma Key [4]
3	Chroma Key [3]
2	Chroma Key [2]
1	Chroma Key [1]
0	Chroma Key [0]

These four registers contain 4 bytes of the chroma key, which are used in conjunction with registers GRC and GRD for occlusion keying on the video stream

Bit	Description
7:0	**Chroma Key [7:0]:** Each register contains 1 byte of the 6 byte chroma key. This is used for occlusion support when keying on the video stream (CR1D[5] =1). See Section 9.5.8 on page 9-39 for additional information.

Register	YUV Video	RGB Video
GRC	Y minimum	Red minimum
GRD	Y maximum	Red maximum
GR1C	U minimum	Green minimum
GR1D	U maximum	Green maximum
GR1E	V minimum	Blue minimum
GR1F	V maximum	Blue maximum

PCI Configuration Registers

7. PCI CONFIGURATION REGISTERS

The PCI Configuration registers in the CL-GD5446 are summarized in Table 7-1. The PCI Configuration registers are accessible when the IDSEL pin is active.

Table 7-1. PCI Configuration Registers Quick Reference

Abbreviation	Register Name	Port	Page
PCI00	PCI Device/Vendor ID	00h	7-3
PCI04	PCI Status/Command	04h	7-4
PCI08	PCI Class Code	08h	7-5
PCI10	PCI Display Memory Base Address	10h	7-6
PCI14	PCI Relocatable I/O / GPIO Base Address (Revision A)	14h	7-7
PCI14	PCI VGA/BitBLT Register Base Address (Revision B)	14h	7-8
PCI18	PCI GPIO Base Address (Revision B)	18h	7-9
PCI2C	PCI Subsystem/Subsystem Vendor ID (Revision B)	2Ch	7-10
PCI30	PCI Expansion ROM Base Address Enable	30h	7-11
PCI3C	PCI Interrupt	3Ch	7-12

7.1 PCI00: PCI Device/Vendor ID

PCI Configuration Address: 00h

Bit	Description	Reset State
31	Device ID [15]	0
30	Device ID [14]	0
29	Device ID [13]	0
28	Device ID [12]	0
27	Device ID [11]	0
26	Device ID [10]	0
25	Device ID [9]	0
24	Device ID [8]	0
23	Device ID [7]	1
22	Device ID [6]	0
21	Device ID [5]	1
20	Device ID [4]	1
19	Device ID [3]	1
18	Device ID [2]	0
17	Device ID [1]	0
16	Device ID [0]	0
15	Vendor ID [15]	0
14	Vendor ID [14]	0
13	Vendor ID [13]	0
12	Vendor ID [12]	1
11	Vendor ID [11]	0
10	Vendor ID [10]	0
9	Vendor ID [9]	0
8	Vendor ID [8]	0
7	Vendor ID [7]	0
6	Vendor ID [6]	0
5	Vendor ID [5]	0
4	Vendor ID [4]	1
3	Vendor ID [3]	0
2	Vendor ID [2]	0
1	Vendor ID [1]	1
0	Vendor ID [0]	1

This is the Device/Vendor ID required for PCI compliance.

Bit	Description
31:16	**Device ID [15:0]:** This read-only field contains the device identifier assigned by Cirrus Logic. This field always returns the value 00B8h for the CL-GD5446.
15:0	**Vendor ID [15:0]:** This read-only field contains the Vendor ID assigned to Cirrus Logic. The value returned is 1013h.

7.2 PCI04: PCI Status/Command

PCI Configuration Address: 04h

Bit	Description	Reset State
31:27	Reserved	
26	DEVSEL Timing [1]	0
25	DEVSEL Timing [0]	1
24:6	Reserved	
5	Enable DAC Shadowing	
4	Reserved	
3	Reserved]	
2	Reserved	
1	Enable Memory Accesses	
0	Enable I/O Accesses	

This is the PCI Status and Command register.

Bit	Description
31:27	**Reserved**
26:25	**DEVSEL# Timing [1:0]:** This read-only field always returns the value '01' to indicate medium DEVSEL# timing.
24:6	**Reserved:** These bits must be programmed to '0'.
5	**Enable DAC Shadowing:** If this bit is programmed to '1', PCI DAC Shadowing is enabled. Write accesses to the CL-GD5446 are executed in the sense that the data are latched in the appropriate register or the palette. However, CL-GD5446 does not acknowledge the access. Read accesses are executed normally.
4:2	**Reserved:** These bits *must* be programmed to '0'.
1	**Enable Memory Accesses:** If this bit is programmed to '1', memory accesses are enabled on the CL-GD5446. If this bit is programmed to '0', memory accesses are not enabled on the CL-GD5446. Memory-mapped I/O access is controlled with this bit.
0	**Enable I/O Accesses:** If this bit is programmed to '1', I/O accesses are enabled on the CL-GD5446. If this bit is programmed to '0', I/O accesses are not enabled on the CL-GD5446. I/O accesses to PCI Configuration registers are always enabled.

7.3 PCI08: PCI Class Code

PCI Configuration Address: 08h

Bit	Description	Reset State
31	Class Code [23]	0
30	Class Code [22]	0
29	Class Code [21]	0
28	Class Code [20]	0
27	Class Code [19]	0
26	Class Code [18]	0
25	Class Code [17]	1
24	Class Code [16]	1
23	Class Code [15]	0
22	Class Code [14]	0
21	Class Code [13]	0
20	Class Code [12]	0
19	Class Code [11]	0
18	Class Code [10]	0
17	Class Code [9]	0
16	Class Code [8]	0
15	Class Code [7]	0
14	Class Code [6]	0
13	Class Code [5]	0
12	Class Code [4]	0
11	Class Code [3]	0
10	Class Code [2]	0
9	Class Code [1]	0
8	Class Code [0]	0
7	Revision ID [7]	X
6	Revision ID [6]	X
5	Revision ID [5]	X
4	Revision ID [4]	X
3	Revision ID [3]	X
2	Revision ID [2]	X
1	Revision ID [1]	X
0	Revision ID [0]	X

This register contains the Class Code required for PCI 2.1 compliance.

Bit	Description
31:8	**Class Code [23:0]:** This read-only field contains the device identifier assigned to a VGA compatible controller. In particular, the Base Class is 03, the Sub-Class is 00, and the Programming interface is 00.
7:0	**Revision ID [7:0]:** This read-only field contains a revision ID assigned by Cirrus Logic. No application program should ever take any action based on the contents of this field.

7.4 PCI10: PCI Display Memory Base Address

PCI Configuration Address: 10h
Index: –

Bit	Description	Reset State
31	Display Memory Base Address	
30	Display Memory Base Address	
29	Display Memory Base Address	
28	Display Memory Base Address	
27	Display Memory Base Address	
26	Display Memory Base Address	
25	Display Memory Base Address	
24	Display Memory Base Address (Revision A only)	
24:1	Reserved (Revision B only)	
23:1	Reserved	0
0	Memory/IO Indicator (Read-only)	0

This 32-bit register contains the Base Address of display memory.

Bit	Description
31:24	**Display Memory Base Address [31:24]:** This 8-bit field contains the base address of the contiguous 16-Mbyte memory block reserved for CL-GD5446 Revision A. The memory is addressable as four byte-swapping apertures. For Revision B of the CL-GD5446, this register claims a 32-Mbyte addressing range. Bit 24 is reserved and always returns '0'. The first 16 Mbytes are three byte-swapping apertures for direct access to the frame buffer and the YUV assist aperture. The second 16 Mbytes are three byte-swapping apertures for system-to-screen BitBLTs and one unused aperture.
23:1	**Reserved:** These read-only bits always returns '0'. For Revision B, this is bit field 24:1.
0	**Memory/IO Indicator:** This read-only field indicates the type of address space requested. A '0' value indicates memory.

7.5 PCI14: PCI Relocatable I/O / GPIO Base Address (Revision A)

PCI Configuration Address: 14h
Index: –

Bit	Description	Reset State
31:5	Base Address	
4:1	Reserved	0
0	Memory I/O	

This 32-bit register contains the Base Address of the VGA I/O and the General-Purpose I/O. This description is for Revision A of the CL-GD5446. This is configured by registers CF8, CF4, and CF3.

Bit **Description**

31:16 **Reserved**

31:5 **Base Address:** This field is programmed to the base address of the relocatable I/O and GPIO. These bits specify an address range configured by registers CF8, CF4, and CF3.

When the CL-GD5446 is configured so that PCI14 is an I/O address; bits above bit 15 are not decoded.

CF8	CF4	CF3	PCI14	VGA I/O	GPIO
X	X	1	All ‘0’s	Disabled	Disabled
0	0	0	15:7 specify I/O address	Disabled	128 bytes I/O
1	0	0	15:5 specify I/O address	Disabled	32 bytes I/O
0	1	0	31:12 specify memory address (4096 bytes claimed)	32 bytes memory	32 bytes memory
1	1	0	15:5 specify I/O address	32 bytes I/O	Disabled

4:1 **Reserved**

0 **Memory I/O:** ‘0’ indicates that memory space is requested; ‘1’ indicates that I/O space is requested.

7.6 PCI14: PCI VGA/BitBLT Register Base Address (Revision B)

PCI Configuration Address: 14h
Index: –

Bit	Description	Reset State
31:12	Base Address	
11:1	Reserved	0
0	Memory I/O	

This 32-bit register contains the Base Address of the VGA I/O and the BitBLT control registers (memory-mapped I/O). This is always enabled on Revision B.

Bit	Description
31:12	**Base Address:** This field is programmed to the base address of the VGA I/O and BitBLT control registers. The first 100 hex addresses are used for VGA; the second 100 hex addresses are used for BitBLT register access. This is always enabled, regardless of configuration resistors.
11:1	**Reserved**
0	**Memory I/O:** Read-only ‘0’ indicates that memory space is requested.

7.7 PCI18: PCI GPIO Base Address (Revision B)

PCI Configuration Address: 18h
Index: –

Bit	Description	Reset State
31:5	Base Address	
4:1	Reserved	0
0	Memory I/O	

This 32-bit register contains the Base Address of GPIO on Revision B. This is configured by CF8 and CF4.

Bit **Description**

31:5 **Base Address:** This field is programmed to the base address of the relocatable GPIO. These bits specify an address range configured by registers CF8 and CF4.

CF8	CF4	PCI18	GPIO
0	0	Reserved, do not use	–
0	1	31:12 specify memory address (4096 bytes claimed)	32 bytes memory
1	0	31:5 specify I/O address	32 bytes I/O
1	1	Read-only all ‘0’s	Disabled

4:1 **Reserved**

0 **Memory I/O:** ‘0’ indicates that memory space is requested; ‘1’ indicates that I/O space is requested.

7.8 PCI2C: PCI Subsystem/Subsystem Vendor ID (Revision B)

PCI Configuration Address: 2Ch
Index: –

Bit	Description	Reset State
31:24	Subsystem ID [15:8]	P[7:0]
23:16	Subsystem ID [7:0]	ROM 7FFE [7:0]
15:8	Subsystem Vendor ID [15:8]	ROM 7FFD [7:0]
7:0	Subsystem Vendor ID [7:0]	ROM 7FFC [7:0]

This 32-bit register uniquely identifies the adapter card containing the CL-GD5446 Revision B. For adapter card applications, this register is loaded from the BIOS ROM and pixel bus and is read-only. For motherboard (PC97) applications, this register can be write-enabled by setting SR17[3]. Cirrus Logic provides a utility called CONFIG.EXE that allows these values to be inserted into the BIOS binary file.

Bit	Description
31:24	**Subsystem ID [15:8]:** If GPIO is enabled, this read-only field is loaded from the Pixel bus on the rising edge of RST#. If GPIO is not enabled, this read-only field returns ‘0’s.
23:16	**Subsystem ID [7:0]:** This read-only field is loaded with the contents of ROM location 0x7FFE.
15:8	**Subsystem Vendor ID {15:8]:** This read-only field is loaded with the contents of ROM location 0x7FFD.
7:0	**Subsystem Vendor ID [7:0]:** This read-only field is loaded with the contents of ROM location 0x7FFC.

7.9 PCI30: PCI Expansion ROM Base Address Enable

PCI Configuration Address: 30h
Index: –

Bit	Description	Reset State
31:14	Expansion ROM Base Address [31:14]	
13:1	Reserved	0
0	EROM Enable	

This 32-bit register contains the base address of EPROM (BIOS) memory.

Bit	Description
31:14	**Expansion ROM Base Address [31:14]:** This field contains the base address of the contiguous 32-Kbyte memory block reserved for the CL-GD5446 BIOS during POST.
13:1	**Reserved**
0	**EROM Enable:** When this bit is programmed to '1', the VGA BIOS at C000:0 is enabled. Display memory is disabled by forcing CAS0# high to allow the MD[7:0] pins to serve as BIOS data pins. When this bit is programmed to '0', the VGA BIOS at C000:0 is disabled.

7.10 PCI3C: PCI Interrupt

PCI Configuration Address: 3Ch
Index: –

Bit	Description	Reset State
15	PCI Interrupt Pin [7]	0
14	PCI Interrupt Pin [6]	0
13	PCI Interrupt Pin [5]	0
12	PCI Interrupt Pin [4]	0
11	PCI Interrupt Pin [3]	0
10	PCI Interrupt Pin [2]	0
9	PCI Interrupt Pin [1]	0
8	PCI Interrupt Pin [0]	$\overline{\text{CF14}}$
7	PCI Interrupt Line [7]	
6	PCI Interrupt Line [6]	
5	PCI Interrupt Line [5]	
4	PCI Interrupt Line [4]	
3	PCI Interrupt Line [3]	
2	PCI Interrupt Line [2]	
1	PCI Interrupt Line [1]	
0	PCI Interrupt Line [0]	

This register denotes the interrupt pin that the CL-GD5446 is connected to (if any) and communicates interrupt line routing information.

Bit	Description
15:8	**PCI Interrupt Pin [7:0]:** If a pull-down resistor is installed on MD62, this read-only field contains the value ‘01’. This is an indication that the CL-GD5446 is claiming an interrupt on INTR#. If no pull-down resistor is installed on MD62, this read-only field contains the value ‘00’.
7:0	**PCI Interrupt Line [7:0]:** This 8-bit field has no direct effect on the CL-GD5446.

Miscellaneous Extension Registers

8. MISCELLANEOUS EXTENSION REGISTERS

The Miscellaneous Extension registers are summarized in Table 8-1.

Table 8-1. Miscellaneous Extension Registers Quick Reference

Abbreviation	Register Name	Index	Port	Page
SR6	Key	06h	3C5h	8-4
SR7	Extended Sequencer Mode	07h	3C5h	8-5
SR8	DDC2B/EEPROM Control	08h	3C5h	8-7
SR9	Scratch Pad 0	09h	3C5h	8-9
SRA	Scratch Pad 1	0Ah	3C5h	8-9
SRB	VCLK0 Numerator	0Bh	3C5h	8-10
SRC	VCLK1 Numerator	0Ch	3C5h	8-10
SRD	VCLK2 Numerator	0Dh	3C5h	8-10
SRE	VCLK3 Numerator	0Eh	3C5h	8-10
SRF	DRAM Control	0Fh	3C5h	8-11
SR10	Graphics Cursor X Position	10h	3C5h	8-13
SR11	Graphics Cursor Y Position	11h	3C5h	8-14
SR12	Graphics Cursor Attributes	12h	3C5h	8-15
SR13	Graphics Cursor Pattern Address Offset	13h	3C5h	8-16
SR14	Scratch Pad 2	14h	3C5h	8-17
SR15	Scratch Pad 3	15h	3C5h	8-17
SR16	Display FIFO Threshold Control	16h	3C5h	8-18
SR17	Configuration Readback and Extended Control	17h	3C5h	8-19
SR18	Signature Generator Control	18h	3C5h	8-20
SR19	Signature Generator Result Low-Byte	19h	3C5h	8-22
SR1A	Signature Generator Result High-Byte	1Ah	3C5h	8-23
SR1B	VCLK0 Denominator and Post-Scalar	1Bh	3C5h	8-24
SR1C	VCLK1 Denominator and Post-Scalar	1Ch	3C5h	8-24
SR1D	VCLK2 Denominator and Post-Scalar	1Dh	3C5h	8-24
SR1E	VCLK3 Denominator and Post-Scalar	1Eh	3C5h	8-24
SR1F	MCLK Select	1Fh	3C5h	8-25
GR9	Offset Register 0	09h	3CFh	8-26

Table 8-1. Miscellaneous Extension Registers Quick Reference *(cont.)*

Abbreviation	Register Name	Index	Port	Page
GRA	Offset Register 1	0Ah	3CFh	8-28
GRB	Graphics Controller Mode Extensions	0Bh	3CFh	8-29
GRC	Color Key/Chroma Key Compare	0Ch	3CFh	8-31
GRD	Color Key/Mask/Chroma Key	0Dh	3CFh	8-32
GRE	Power Management	0Eh	3CFh	8-33
GR16	Active Display Line Readback Byte 0	16h	3CFh	8-35
GR17	Active Display Line Readback Byte 1	17h	3CFh	8-36
GR18	Extended DRAM Controls	18h	3CFh	8-37
GR19	GPIO Configuration	19h	3CFh	8-39
GR1A	Scratch Pad 4	1Ah	3CFh	8-40
GR1B	Scratch Pad 5	1Bh	3CFh	8-40
CR19	Interlace End	19h	3?5h	8-41
CR1A	Miscellaneous Control	1Ah	3?5h	8-42
CR1B	Extended Display Controls	1Bh	3?5h	8-44
CR1C	Sync Adjust and GENLOCK	1Ch	3?5h	8-46
CR1D	Overlay Extended Control	1Dh	3?5h	8-48
CR25	Part Status (read only)	25h	3?5h	8-50
CR27	ID (read only)	27h	3?5h	8-51
HDR	Hidden DAC Register	–	3C6h	8-52

8.1 SR6: Key

I/O Port Address: 3C5h
Index: 06h

Bit	Description	Reset State
7	Don't Care	0
6	Don't Care	0
5	Don't Care	0
4	Unlock	0
3	Don't Care	1
2	Unlock	1
1	Unlock	1
0	Unlock	1

This register is provided for compatibility with earlier Cirrus Logic desktop products. This register does not control access to the Extension registers on the CL-GD5446.

Bit	Description
7:0	**Key:** If this register is loaded with 12h, when read it returns the value '12h'. If this register is loaded with any other value, when read it returns the value 'FFh'.

8.2 SR7: Extended Sequencer Mode

I/O Port Address: 3C5h
Index: 07h

Bit	Description	Reset State
7	Linear Frame Buffer Enable [3]	0
6	Linear Frame Buffer Enable [2]	0
5	Linear Frame Buffer Enable [1]	0
4	Linear Frame Buffer Enable [0]	0
3	Sequencer and CRTC Clocking Control [2]	
2	Sequencer and CRTC Clocking Control [1]	
1	Sequencer and CRTC Clocking Control [0]	
0	Enable High-resolution Packed-Pixel Modes	

This register has several purposes, as described in the following bit descriptions.

Bit	Description
7:4	**Linear Frame Buffer Enable [3:0]:** If this field is programmed to any value other than '0000', linear addressing of the frame buffer is enabled. The CL-GD5446 requests a 16-Mbyte space starting at the value programmed in PCI10.
3:1	**Sequencer and CRTC Clocking Control [2:0]:** This 3-bit field selects the sequencer data path width. All extended graphics modes should be programmed to use a parallel data path the same width as the pixel. Program VCLK to the desired pixel frequency (except for clock-doubled 8-bpp modes).

SR7[3:1]	Mode	Note
000	8-bpp	
001	Reserved	
010	24-bpp	Packed-24
011	16-bpp	Also used for clock-doubled 8-bpp modes
100	32-bpp	
101	Reserved	
110	Reserved	
111	Reserved	

8.2 SR7: Extended Sequencer Mode *(cont.)*

Bit	Description
0	**Enable High-Resolution Packed-Pixel Modes:** If this bit is '0', VGA operation is enabled. The frame buffer uses 32-bit operation and only the first 1 Mbyte is displayed or accessible to the host regardless of the amount actually configured (any additional memory that is configured is refreshed). The CRTC FIFO is set to eight levels. All standard VGA display modes are supported, including text mode and Planar Graphics. Standard VGA data manipulation is available including: • GR0/GR1 Set/Reset function • GR3 data rotate and read latch functionality • Write Modes 0, 1, 2, and 3 • Read Modes 0 and 1 If this bit is '1', extended display modes are enabled, including 8-, 16-, 24-, and 32-bpp modes. The frame buffer uses 32- or 64-bit operations and all configured memory is accessible to the host and for display. The CRTC FIFO uses all available levels. Text and Planar Graphics modes are not supported for display. If SR4[3] is '1', CPU reads and writes use extended functionality. The following VGA data manipulation is disabled • GR0/GR1 Set/Reset function (GR0/GR1 are color registers) • GR3 data rotate and read latch functionality • Write Modes 2 and 3 • Read Mode 1 If SR4[3] is '0', SR4[2] is '1', and GR6[1] is '0', an extended, unchained addressing mode is selected. This is a planar, unchained addressing mode that can access all memory. Standard VGA data manipulation is available including: • GR0/GR1 Set/Reset function • GR3 data rotate and read latch functionality • Write Modes 0, 1, 2, and 3 • Read Modes 0 and 1

8.3 SR8: DDC2B/EEPROM Control

I/O Port Address: 3C5h
Index: 08h

Bit	Description	Reset State
7	DDCDAT Readback/EEDI Readback	
6	DDC2B Configuration	0
5	Reserved/Latch ESYNC and EVIDEO#	
4	Reserved/Enable EEPROM Data and SK	
3	Reserved/Data to EEPROM	
2	DDCCLK Readback/SK to EEPROM	
1	DDC Data Out/EEPROM Data In	
0	DDC Clock Out/EECS Output	

This register controls the DDC (Display Data Channel) control. See Appendix B8, "DDC2B/I2C Support", for information regarding DDC. In the bit descriptions that follow, each bit (except bit 6) is described first for DDC2B configuration, then for EEPROM configuration.

Bit	Description
7	**DDCDATA Readback (DDC2B Configuration):** This read-only bit reflects the state of the DDCDATA pin (pin 106). **EEDI Readback (EEPROM Configuration):** This read-only bit reflects the state of the EEDI pin (pin 106) if SR8[1] is '1'. If SR8[1] is '0', the bit is always read as '0'.
6	**DDC2B Support:** If this bit is set to '0', the CL-GD5446 is configured for EEPROM and the remaining bits in this register have their EEPROM meanings. If this bit is set to '1', the CL-GD5446 is configured for DDC2B support and the remaining bits have their DDC2B meanings.
5	**Reserved (DDC2B Configuration)** **Latch ESYNC# and EVIDEO# (EEPROM Configuration):** When this bit is set to '0', the ESYNC and EVIDEO# pins are inputs and normally control the HSYNC, VSYNC, BLANK#, and P[7:0] drivers. When this bit is set to '1', the input levels on ESYNC# and EVIDEO# are latched internally and these latched levels control the HSYNC, VSYNC, BLANK#, and P[7:0] drivers. This frees the ESYNC# and EVIDEO# pins to control the EEPROM. Set this bit to '1' prior to setting SR8[4] and set to '0' only after clearing SR8[4].

8.3 SR8: DDC2B/EEPROM Control *(cont.)*

Bit	Description
4	**Reserved (DDC2B Configuration)** **Enable EEPROM Data and SK (EEPROM Configuration):** When this bit is set to '1', ESYNC# and EVIDEO# become outputs and reflect the values in SR8[2] and SR8[3], respectively. When this bit is set to '0', ESYNC and EVIDEO# are inputs.
3	**Reserved (DDC2B Configuration)** **Data-to-EEPROM (EEPROM Configuration):** When SR8[4] is set to '1', the level on EVIDEO# reflects the value of this bit. This bit typically controls the DI pin of EEPROM.
2	**DDCCLK Readback (DDC2B Configuration):** The read-only bit reflects the state of the DDCCLK pin (pin 107). **SK-to-EEPROM (EEPROM Configuration):** When SR8[4] is set to '1', the level on ESYNC reflects the value of this bit. This bit typically controls the SK pin of EEPROM.
1	**DDCDAT Output (DDC2B Configuration):** If this bit is set to '0', the DDCDAT pin (pin 106) is driven low. If this bit is set to '1', the DDCDAT pin is high-impedance (a nominal 1-kΩ resistor pulls the pin high). **Enable EEPROM Data In (EEPROM Configuration):** When this bit is set to '1', the level on GPIO0 (EEDI) is reflected on SR8[7].
0	**DDCCLK Output (DDC2B Configuration):** If this bit is set to '0', the DDCCLK pin (pin 107) is driven low. If this bit is set to '1', the DDCCLK pin is high-impedance (a nominal 1-kΩ resistor pulls the pin high). **CS Out to EEPROM (EEPROM Configuration):** The level on GPIO1 (EECS) reflects the value of this bit.

8.4 SR9–SRA: Scratch Pad 0, 1

I/O Port Address: 3C5h
Index: 09h, 0Ah

Bit	Description	Reset State
7	R/W Data [7]	0
6	R/W Data [6]	0
5	R/W Data [5]	0
4	R/W Data [4]	0
3	R/W Data [3]	0
2	R/W Data [2]	0
1	R/W Data [1]	0
0	R/W Data [0]	0

IMPORTANT: These two registers are reserved for the exclusive use of the CL-GD5446 BIOS and must never be written to by an application program.

This register description is included for completeness only.

Bit	Description
7:0	These bits are reserved for the Cirrus Logic BIOS.

8.5 SRB–SRE: VCLK0–VCLK3 Numerator

I/O Port Address: 3C5h
Index: 0Bh, 0Ch, 0Dh, 0Eh

Bit	Description
7	Reserved
6	VCLK Numerator [6]
5	VCLK Numerator [5]
4	VCLK Numerator [4]
3	VCLK Numerator [3]
2	VCLK Numerator [2]
1	VCLK Numerator [1]
0	VCLK Numerator [0]

These registers, in conjunction with SR1B–SR1E, determine the frequency of display clock. Refer to Chapter 9, “Programming Notes”, for complete programming information for the synthesizers.

Bit	Description
7	**Reserved**
6:0	**VCLK Numerator [6:0]:** The following table shows the values these registers contain at RESET:

Clock	Freq. (MHz)	N	D	P	Numerator	Denominator/ Post-Scalar
VCLK0	25.180	102	29	1	66h	3Bh
VCLK1	28.325	91	23	1	5Bh	2Fh
VCLK2	41.165	69	24	0	45h	30h
VCLK3	36.082	126	25	1	7Eh	33h

8.6 SRF: DRAM Control

I/O Port Address: 3C5h
Index: 0Fh

Bit	Description	Reset State
7	DRAM Bank Switch Control	
6	CPU Write Buffer Control	0
5	Reserved	
4	DRAM Data Bus Width [1]	0
3	DRAM Data Bus Width [0]	0
2	RAS Timing: MD[57] (Read-only)	CF[9]
1:0	Reserved	

This register controls the display memory.

Bit **Description**

7 **DRAM Bank Switch Control:** When this bit is set to '0', bank switching is disabled and the RAS0# output is inactive. When this bit is set to '1', bank switching is enabled. RAS1# is forced active for the first bank; RAS0# is forced active for the second bank.

6 **CPU Write Buffer Control:** When this bit is set to '0', Fast-Page Detection is enabled. CPU writes that can take place as Fast Page mode writes do. If this bit is set to '1' Fast-Page Detection is disabled. CPU writes never take place as Fast Page mode writes.

This bit must be set to '1' when loading font data for the 'page mode' text (132-column and CR1B[6] is set to '1'). In all other circumstances, this bit is set to '0'.

5 **Reserved**

4:3 **DRAM Data Bus Width [1:0]:** This 2-bit field specifies the Display Memory Data bus width as shown in the following table.

SRF[4]	SRF[3]	Data Bus Width	Memory Size
0	0	Reserved	n/a
0	1	Reserved	n/a
1	0	32 bits	1Mbyte
1	1	64 bits	1Mbyte / 2Mbytes / 4Mbytes

These bits have one level of buffering. At the end of each horizontal scanline refresh interval (that is, when horizontal blanking begins), these bits are transferred to the timing logic.

8.6 SRF: DRAM Control *(cont.)*

Bit **Description**

2 **RAS# Timing MD[57]:** This read-only bit indicates the RAS# timing as selected in CF[9]. This is summarized in the following table:

SRF[2] CF[9]	Pull-down on MD[57]?	RAS# High	RAS# Low	Note
0	Yes	3 MCLK	4 MCLK	Extended RAS#
1	No	2.5 MCLK	3.5 MCLK	Standard RAS#

1:0 **Reserved**

8.7 SR10: Graphics Cursor X Position

I/O Port Address: 3C5h
Index: 10h, 30h, 50h, 70h, 90h, B0h, D0h, F0h

Bit	Description	Reset State
7	Cursor X [10]	0
6	Cursor X [9]	0
5	Cursor X [8]	0
4	Cursor X [7]	0
3	Cursor X [6]	0
2	Cursor X [5]	0
1	Cursor X [4]	0
0	Cursor X [3]	0

This register, and bits 7:5 of the index that access it, define the horizontal (X) pixel offset of the graphics cursor. Refer to Section 9.8 on page 9-53, for more information, including code examples, regarding the graphics cursor.

The data forms the eight upper bits of the 11-bit position; bits 7:5 of the index form the three lower bits of the 11-bit position. This allows the entire 11-bit cursor offset to be written in a single 16-bit I/O write. The offset must be placed in AX[15:5], AX[4:0] must be '10000', and DX must be '03C4'.

The three bits of stored cursor position can be read by executing a byte write to 3C4 using the data values 10, 30, 50.F0. A read of 3C4 then returns the previously stored three bits of cursor position.

Bit	Description
7:0	**Cursor X [10:3]:** This 8-bit field forms the upper eight bits of the 11-bit horizontal offset of the graphics cursor. The index that accesses this register forms the least-significant three bits of the 11-bit offset.

8.8 SR11: Graphics Cursor Y Position

I/O Port Address: 3C5h
Index: 11h, 31h, 51h, 71h, 91h, B1h, D1h, F1h

Bit	Description	Reset State
7	Cursor Y [10]	0
6	Cursor Y [9]	0
5	Cursor Y [8]	0
4	Cursor Y [7]	0
3	Cursor Y [6]	0
2	Cursor Y [5]	0
1	Cursor Y [4]	0
0	Cursor Y [3]	0

This register, and bits 7:5 of the index that access it, define the vertical (Y) scanline offset of the graphics cursor. Refer to Section 9.8 on page 9-53, for more information, including code examples, regarding the graphics cursor.

The data forms the eight upper bits of the 11-bit position; bits 7:5 of the index form the three lower bits of the 11-bit position. This allows the entire 11-bit cursor offset to be written in a single 16-bit I/O write. The offset must be placed in AX[15:5], AX[4:0] must be '10001', and DX must be '03C4'.

The three bits of stored cursor position can be read by executing a byte write to 3C4 using the data values 11, 31, 51.F1. A read of 3C4 then returns the previously stored three bits of cursor position.

Bit	Description
7:0	**Cursor Y [10:3]:** This 8-bit field forms the upper eight bits of the 11-bit vertical offset of the graphics cursor. The index that accesses this register forms the least-significant three bits of the 11-bit offset.

8.9 SR12: Graphics Cursor Attribute

I/O Port Address: 3C5h
Index: 12h

Bit	Description
7	Overscan Color Protect
6:3	Reserved
2	Cursor Size Select
1	Allow Access to DAC Extended Colors
0	Graphics Cursor Enable

This register enables/disables the graphics cursor, as well sets the cursor size and enables the palette DAC table entries used to define the colors. Refer to Section 9.8 on page 9-53 for a complete programming guide for the graphics cursor.

Bit	Description
7	**Overscan Color Protect:** If this bit is set to '1', the border color is DAC LUT entry 258. Note that entry 258 can only be accessed if SR12[1] is set to '1'. If this bit is set to '0', the border color is normal (that is, the contents of the palette pointed to by register AR11). Register AR11 normally contains '0', and palette entry 0 normally contains values corresponding to black.
6:3	**Reserved**
2	**Cursor Size Select:** If this bit is set to '0', the graphics cursor is 32 × 32 pixels. If this bit is set to '1', the graphics cursor is 64 × 64 pixels.
1	**Allow Access to DAC Extended Colors:** If this bit is set to '1', three extra DAC LUT entries are addressable. The entry addressable as x0h is the cursor background. The entry addressable as xFh is the cursor foreground. This allows a a cursor completely independent of the display data colors. The entry addressable as X2h provides a selected overscan (border) color. If this bit is set to '0', the DAC LUT is VGA-compatible.
0	**Graphics Cursor Enable:** If this bit is set to '1', the graphics cursor is enabled and appears on the screen. If it is set to '0', the graphics cursor is disabled and does not appear on the screen.

8.10 SR13: Graphics Cursor Pattern Address Offset

I/O Port Address: 3C5h
Index: 13h

Bit	Description
7:6	Reserved
5	Cursor Select [5]
4	Cursor Select [4]
3	Cursor Select [3]
2	Cursor Select [2]
1	Cursor Select [1]
0	Cursor Select [0]

This register selects one of 64 cursor patterns (32 × 32 cursor) or one of 16 cursor patterns (64 × 64 cursor). Refer to Refer to Section 9.8 on page 9-53, for a complete programming guide for the graphics cursor.

Bit	Description
7:6	**Reserved**
5:0	**Cursor Select (32 × 32 Cursor) [5:0]:** This 6-bit field selects one of 64 possible cursor patterns stored at the top (highest addressed 16 Kbytes) of display memory. This definition is valid only if SR12[2] is set to ‘0’.
5:2	**Cursor Select (64 × 64 Cursor) [5:2]:** This 4-bit field selects one of 16 possible cursor patterns stored at the top (highest addressed 16 Kbytes) of display memory. This definition is valid only if SR12[2] is set to ‘1’. In this case, SR13[1:0] are ignored.

8.11 SR14–SR15: Scratch Pad 2, 3

I/O Port Address: 3C5h
Index: 14h, 15h

Bit	Description	Reset State
7	R/W Data [7]	0
6	R/W Data [6]	0
5	R/W Data [5]	0
4	R/W Data [4]	0
3	R/W Data [3]	0
2	R/W Data [2]	0
1	R/W Data [1]	0
0	R/W Data [0]	0

These two registers are reserved for the exclusive use of the CL-GD5446 BIOS. These registers are included for completeness only.

NOTE: These registers must never be written to by an application program.

Bit	Description
7:0	**R/W Data [7:0]:** These bits are reserved for the Cirrus Logic BIOS.

8.12 SR16: Display FIFO Threshold Control

I/O Port Address: 3C5h
Index: 16h

Bit	Description	Reset State
7:4	Reserved	
3	Display FIFO Threshold [3]	0
2	Display FIFO Threshold [2]	0
1	Display FIFO Threshold [1]	0
0	Display FIFO Threshold [0]	0

This register controls the threshold that the CRT FIFO is refilled. This register must never be written by an application program.

Bit	Description
7:4	**Reserved**
3:0	**Display FIFO Threshold [3:0]:** The value written to this field selects the level that the sequencer begins display memory cycles to refill the CRT FIFO (and thereby hold off CPU and BitBLT cycles). For each display mode and MCLK frequency, there is an optimum value that most efficiently uses the DRAM. This value can be overridden for scanlines that fall within the video window with the secondary FIFO threshold value in register CR5C (See Chapter 6, “Video Capture and Playback Registers”).

8.13 SR17: Configuration Readback and Extended Control

I/O Port Address: 3C5h
Index: 17h

Bit	Description
7	DRAM Bank Size Select
6	Memory-Mapped I/O Address
5:4	Reserved
3	Write Enable PCI2C (Revision B only)
2	Enable Memory-Mapped I/O
1	Enable DRAM Bank Swap
0	Reserved

Bit	Description
7	**DRAM Bank Size:** If this bit is '1', the DRAM bank size is 1 Mbyte. This is used in conjunction with SR17[1] and SRF[7] to specify the number and size of DRAM banks. Appendix B3, "Memory Configurations and Timing" contains a complete listing of DRAM configurations and upgrade paths.
6	**Memory-Mapped I/O Address:** If memory-mapped I/O or linear addressing is not enabled, this bit is ignored. If linear addressing and memory-mapped I/O are both enabled, this bit has the following meaning: If this bit is set to '0', the address space for memory-mapped I/O is 256 bytes, beginning at B800:0. If this bit is set to '1' the address space for memory-mapped I/O is the last 256 bytes of the populated linear address space.
5:4	**Reserved**
3	**Write Enable PCI2C (Revision B only):** If this bit is '1', PCI2C can be written. This is intended for compliance with PC97 for motherboard applications.
2	**Enable Memory-mapped I/O:** If this bit is set to '0', the CL-GD5446 operates normally in that the memory-mapped I/O is not enabled. If this bit is set to '1', the BitBLT registers (GR0, GR1, GR10–GR15 and GR20–GR3F) are addressable as a 36-byte block of memory, allowing faster access. When the CL-GD5446 is configured for linear addressing, SR17[6] specifies the location of the block reserved for memory-mapped I/O.
1	**Enable DRAM Bank Swap:** If this bit is '1', the decodes for RAS0# and RAS1# are swapped. This allows a first bank of 1 Mbyte and second upgrade bank of 2 Mbytes. When this bit is '1', the bank size must be set to 2 Mbytes (see SR17[7]).
0	**Reserved:** This bit must always be programmed to '0'.

8.14 SR18: Signature Generator Control

I/O Port Address: 3C5h
Index: 18h

Bit	Description
7	Signature Type [1]
6	Signature Type [0]
5	Enable Data Generator
4	Pixel Bus Select [2]
3	Pixel Bus Select [1]
2	Pixel Bus Select [0]
1	Reset Signature Generator
0	Signal Generator Enable/Status

This register controls and monitors the status of the SG (signature generator). The CL-GD5446 contains an upgraded SG that can test the BitBLT engine and V-Port logic, as well as the frame buffer and display data paths. Refer to Appendix B6, “Signature Generator”, for a complete description including sample code.

Bit	Description
7:6	**Signature Type [1:0]:** This field selects the functional block where the signature is being captured.
5	**Enable Data Generator:** If this bit is set to ‘1’, pseudo-random data is placed on the memory data bus. This is used in conjunction with the SG. This mode is intended for factory testing only.

8.14 SR18: Signature Generator Control *(cont.)*

Bit **Description**

4:2 **Pixel Bus Select [2:0]:** This field selects the bit of the pixel bus used as the input for the SG according to the following table.

SR18[4]	SR18[3]	SR18[2]	P-Bus Bit
0	0	0	P[0]
0	0	1	P[1]
0	1	0	P[2]
0	1	1	P[3]
1	0	0	P[4]
1	0	1	P[5]
1	1	0	P[6]
1	1	1	P[7]

1 **Reset Signature Generator:** When this bit is set to ‘1’, the SG is reset to an initially-defined condition. When this bit is set to ‘0’, the SG is allowed to run under the control of SR18[0].

0 **Signature Generator Enable/Status:** When this bit is set to ‘1’, the SG begins operation on the next VSYNC. It accumulates a signature from the pixel bus bit selected in SR18[4:2] for one display frame and then stops, forcing this bit to ‘0’. The application can determine when the signature is complete by monitoring this bit.

8.15 SR19: Signature Generator Result Low-Byte

I/O Port Address: 3C5h
Index: 19h

Bit	Description	Reset State
7	Signature Generator Result [7]	0
6	Signature Generator Result [6]	0
5	Signature Generator Result [5]	0
4	Signature Generator Result [4]	0
3	Signature Generator Result [3]	0
2	Signature Generator Result [2]	0
1	Signature Generator Result/Byte Select [1]	0
0	Signature Generator Result/Byte Select [0]	0

This register reads the least-significant byte of the SG result. The CL-GD5446 SG is for board-level testing of the display subsystem. Refer to Appendix B6, “Signature Generator”, for a complete description.

Bit	Description
7:0	**Signature Generator Result [7:0]**
1:0	**Byte Select [1:0]**

8.16 SR1A: Signature Generator Result High-Byte

I/O Port Address: 3C5h
Index: 1Ah

Bit	Description	Reset State
7	Signature Generator Result [15]	0
6	Signature Generator Result [14]	0
5	Signature Generator Result [13]	0
4	Signature Generator Result [12]	0
3	Signature Generator Result [11]	0
2	Signature Generator Result [10]	0
1	Signature Generator Result [9]	0
0	Signature Generator Result [8]	0

This register reads the most-significant byte of the SG result. The CL-GD5446 SG is for board-level testing of the display subsystem. Refer to Appendix B6, “Signature Generator”, for a complete description.

Bit	Description
7:0	**Signature Generator Result [15:8]**

8.17 SR1B–SR1E: VCLK0–VCLK3 Denominator and Post-Scalar

I/O Port Address: 3C5h
Index: 1Bh, 1Ch, 1Dh, 1Eh

Bit	Description
7	VCLK Denominator [6]
6	VCLK Denominator [5]
5	VCLK Denominator [4]
4	VCLK Denominator [3]
3	VCLK Denominator [2]
2	VCLK Denominator [1]
1	VCLK Denominator [0]
0	VCLK Post-Scalar (÷2)

These registers, in conjunction with registers SRB–SRE, program the frequency of display clocks 0 (VCLK0) through 3 (VCLK3). Refer to Chapter 9, “Programming Notes”, for complete programming information for the synthesizers. The reset values for these four registers are shown in the VCLK numerator table in Section 8.5 on page 8-10.

Bit	Description
7:1	**VCLK Denominator [6:0]**
0	**VCLK Post-Scalar (÷2)**

8.18 SR1F: MCLK Select

I/O Port Address: 3C5h
Index: 1Fh

Bit	Description	Reset State
7	Reserved	0
6	Use MCLK as VCLK	X
5	MCLK Frequency [5]	0
4	MCLK Frequency [4]	(Refer to MCLK table below)
3	MCLK Frequency [3]	(Refer to MCLK table below)
2	MCLK Frequency [2]	(Refer to MCLK table below)
1	MCLK Frequency [1]	(Refer to MCLK table below)
0	MCLK Frequency [0]	(Refer to MCLK table below)

This register allows the MCLK frequency to be programmed directly. This register should never be programmed by an applications program and is listed for completeness only.

Bit **Description**

7 **Reserved:** This bit must always be set to '0'.

6 **Use MCLK as VCLK:** If this bit is set to '0', the VCLK synthesizer operates normally. If this bit is set to '1', VCLK is derived from MCLK as follows:

SR1F[6]	SR1E[0]	VCLK Source
0	X	VCLK (normal operation)
1	0	VCLK = MCLK
1	1	VCLK = MCLK ÷ 2

5:0 **MCLK Frequency [5:0]:** This field directly programs the MCLK frequency as indicated in Equation 8-1.

$$MCLK = SR1F \bullet \frac{Reference}{8} \cong SR1F \bullet 1.8MHz$$ **Equation 8-1**

Refer to Appendix B3, "Memory Configurations and Timing", for information regarding the DRAM specification requirements for various MCLK frequencies.

8.19 GR9: Offset Register 0

I/O Port Address: 3CFh
Index: 09h

Bit	Description	Reset State
7	Offset 0 [7]	0
6	Offset 0 [6]	0
5	Offset 0 [5]	0
4	Offset 0 [4]	0
3	Offset 0 [3]	0
2	Offset 0 [2]	0
1	Offset 0 [1]	0
0	Offset 0 [0]	0

NOTE: Offset addressing is retained for compatibility with previous Cirrus Logic products. All new software should be written to use linear addressing.

This register provides access to up to 1 Mbyte of display memory with 4-Kbyte granularity or 4 Mbytes of display memory with 16-Kbyte granularity. The contents of this register are added to A[19:12] when Extension register GRB[0] is set to '0', or when GRB[0] is set to '1' and SA15 = 0. If GRB[5] is set to '1', bits 6:0 of this register are added to A[20:14] to provide access to up to 4 Mbytes of display memory with 16-Kbyte granularity.

The display memory address, prior to being modified by address wrap controls, is called XMA, the sum of XA and an Offset register. XA is the address on the bus with bits 16 and 15 possibly forced to '0' as indicated in the following table.

Configuration	XA[16]	XA[15]	XA[14:0]
64K Memory: GR6[3:2] = 0,1 and Offset 1 Disabled: GRB[0] = 0	0	SA[15]	SA[14:0]
64K Memory: GR6[3:2] = 0,1 or Offset 1 Enabled: GRB[0] = 1	0	0	SA[14:0]

The XA address is summed with the contents of an Offset register, with one of three relative alignments according to the configuration. These are indicated in the following tables.

Table 8-2. 1 Mbyte Memory with 4-Kbyte Granularity: VGA Mapping

0	0	0	XA[16]	XA[15]	SA[14]	SA[13]	SA[12]
+OFF[7]	OFF[6]	OFF[5]	OFF[4]	OFF[3]	OFF[2]	OFF[1]	OFF[0]
XMA[19]	XMA[18]	XMA[17]	XMA[16]	XMA[15]	XMA[14]	XMA[13]	XMA[12]

8.19 GR9: Offset Register 0 *(cont.)*

Table 8-3. 4 Mbytes Memory with 16-Kbyte Granularity: VGA Mapping

0	0	0	0	0	XA[16]	XA[15]	SA[14]	SA[13]	SA[12]
+OFF[7]	+OFF[6]	OFF[5]	OFF[4]	OFF[3]	OFF[2]	OFF[1]	OFF[0]	0	0
XMA[21]	XMA[20]	XMA[19]	XMA[18]	XMA[17]	XMA[16]	XMA[15]	XMA[14]	XMA[13]	XMA[12]

Bit	Description
7:0	**Offset 0 [7:0]:** This value is added to A[19:12] to provide the address into display memory. This Offset register is selected when GRB[0] is set to '0', or when GRB[0] = 1 and SA15 = 0.

8.20 GRA: Offset Register 1

I/O Port Address: 3CFh
Index: 0Ah

Bit	Description	Reset State
7	Offset 1 [7]	0
6	Offset 1 [6]	0
5	Offset 1 [5]	0
4	Offset 1 [4]	0
3	Offset 1 [3]	0
2	Offset 1 [2]	0
1	Offset 1 [1]	0
0	Offset 1 [0]	0

This register provides access to up to 1 Mbyte of display memory with 4-Kbyte granularity. The contents of this register are added to A[19:12] when Extension register GRB[0] is set to '1' and SA15 = 1. If GRB[5] is set to '1', bits 6:0 of this register are added to A[20:14] to provide access to up to 2 Mbytes of display memory with 16-Kbyte granularity.

This provides an additional 32K window into 1 Mbyte of display memory with 4- or 16-Kbyte granularity.

Bit	Description
7:0	**Offset 1 [7:0]:** This value is added to A[19:12] to provide the address into display memory. This Offset register is selected when GRB[0] is set to '1' and SA15 = 1. If GRB[0] is set to '0', this register is unused.

8.21 GRB: Graphics Controller Mode Extensions

I/O Port Address: 3CFh
Index: 0Bh

Bit	Description
7:6	Reserved
5	Offset Granularity
4	Enable Enhanced Writes for 16-bit Pixels
3	Enable Eight Byte Data Latches
2	Enable Extended Write Modes
1	Enable BY8 Addressing
0	Enable Offset Register 1

This register enables/disables Extended Write modes. These Extended Write modes provide compatibility with earlier Cirrus Logic non-BitBLT controllers.

Bit	Description
7:6	**Reserved**
5	**Offset Granularity:** If this bit is set to '1', the Offset registers are redefined as containing bits [6:0] which are added to Address bits [21:14] to provide access to 4 Mbytes of display memory with 16-Kbyte granularity. SR7[4] (least-significant bit of 1-Mbyte address page) becomes a don't care.
4	**Enable Enhanced Writes for 16-bit pixels:** When this bit and GRB[2] are both set to '1', the CL-GD5446 executes Enhanced Write modes 4 and 5 writes. In particular: • **BY16 Addressing Enabled:** The system address is shifted by four relative to true packed-pixel addressing so that each system byte address points to a different 8-pixel (16-byte) block in display memory. • **16-Byte Transfer Enabled:** Up to 16 bytes (8 pixels) can be written into display memory for each CPU byte transfer. • **GR10 and GR11 Enabled:** Registers GR10 and GR11 are enabled as foreground and background color extensions. • **SR2 Doubling Enabled:** Each bit of register SR2 is a pixel write mask for 2 bytes (1 pixel).
3	**Enable Eight-Byte Data Latches:** If this bit is set to '1', the display memory latches are 8-bytes wide rather than the normal four.

8.21 GRB: Graphics Controller Mode Extensions *(cont.)*

Bit	Description
2	**Enable Extended Write Modes:** If this bit is set to '1', the CL-GD5446 executes an Extended mode write. In particular: • **8-Byte Transfer Enabled:** Up to 8 bytes (8 pixels) can be written into display memory for each CPU byte transferred. If GRB[4] is set to '1', up to 16 bytes can be written for color expansion. • **GR5[2] Enabled:** Extended Write modes 4 and 5 can be enabled. • **GR0 Extended:** register GR0 is extended from 4 to 8 bits. • **GR1 Extended:** register GR1 is extended from 4 to 8 bits. • **SR2 Extended:** register SR2 is extended from 4 to 8 bits. • **GRB[4] Enabled:** GRB[2] must be set to '1' to enable GRB[4]. A side effect of programming this bit from '1' to '0' is to clear GR0[7:4] and GR1[7:4] to '0'.
1	**Enable BY8 Addressing:** The system address is shifted by three relative to true packed-pixel addressing so that each system byte address points to a different 8-pixel (8-byte) block in display memory. This bit must be set to '0' if GRB[4] is set to '1' (that is, BY8 and BY16 addressing must not be selected simultaneously).
0	**Enable Offset Register 1:** If this bit is set to '1', then register SA15 selects between Offset registers 0 and 1. If this bit is set to '0', then Offset register '0' is always selected regardless of the value of register SA15. This bit must always be set to '0' for 1 Mbyte of linear addressing.

8.22 GRC: Color Key/Chroma Key Compare

I/O Port Address: 3CFh
Index: 0Ch

Bit	Description	Reset State
7	Color/Chroma Key Compare [7]	1
6	Color/Chroma Key Compare [6]	1
5	Color/Chroma Key Compare [5]	1
4	Color/Chroma Key Compare [4]	1
3	Color/Chroma Key Compare [3]	1
2	Color/Chroma Key Compare [2]	1
1	Color/Chroma Key Compare [1]	1
0	Color/Chroma Key Compare [0]	1

This register contains the graphics color key value or one byte of the video chroma key value.

Bit	Description
7:0	**Color/Chroma Key Compare [7:0]:** The meaning of this field depends on CR1D[5:3] as summarized in the following table. Refer to Section 9.5.8 on page 9-39 for additional information. Color key and chroma key occlusion is supported for 8- and 16-bpp modes.

CR1D[5]	CR1D[4:3]	GRC	CRD	Case
0	00	Color key	Mask	8-bit compare under mask
0	01	Low byte	High byte	16-bit compare (16-bpp graphics)
0	10	–	–	Reserved
0	11	Unused	bit 7	See Chapter 9
1	XX	Y MIN	Y MAX	Chroma key (video)

8.23 GRD: Color Key/Mask/Chroma Key

I/O Port Address: 3CFh
Index: 0Dh

Bit	Description	Reset State
7	Color Key / Mask / Chroma Key [7]	0
6	Color Key / Mask / Chroma Key [6]	0
5	Color Key / Mask / Chroma Key [5]	0
4	Color Key / Mask / Chroma Key [4]	0
3	Color Key / Mask / Chroma Key [3]	0
2	Color Key / Mask / Chroma Key [2]	0
1	Color Key / Mask / Chroma Key [1]	0
0	Color Key / Mask / Chroma Key [0]	0

This register contains the 8-bit color key mask, the most-significant byte of the color key, or one byte of the chroma key.

Bit	Description
7:0	**Color Key / Mask / Chroma Key [7:0]:** Depending on how the fields in CR1D[5:3] are programmed, this field can serve as the most-significant byte of the color key (graphics), the mask for 8-bpp color key (graphics), or the Y maximum for chroma key (video). See Chapter 9, “Programming Notes”, and Section 8.22.

8.24 GRE: Power Management

I/O Port Address: 3CFh
Index: 0Eh

Bit	Description
7:6	Reserved
5	Enable Writes to GR33
4	System Level Power Management
3	Static Clock Mode
2	Static VSYNC
1	Static HSYNC
0	DCLK Output ÷2

This register contains bits that are used for power management.

Bit	Description
7:6	**Reserved**
5	Enable Writes to GR33: If this bit is '1' or GR31[7] is '1', then GR33 can be written. If this bit is '0' and GR31[7] is '0', then register GR33 is write protected. The CL-GD5446 takes the bus cycle and returns TRDY#, but the data directed to register GR33 is discarded.
4	**System Level Power Management:** If this bit is set to '1', host access to the display memory is disabled (see MISC[1]) and screen refresh is disabled (see SR1[5]). The MCLK and VCLK VCOs continue to operate at their programmed frequencies, but the device consumes very little power. DRAM refresh continues. The following procedures must be used when programming power-down and power-up. ***Power Down*** • Set GRE[4] to '1'; GRE[3] must be '0'. • Ensure interrupts are disabled (if necessary, execute a CLI). • Wait for two '0'-to-'1' transitions of 3DA[0] (the Status Port Display Enable bit) • Interrupts can be restored at this point • Set GRE[3] to '1'. ***Power Up*** • Clear GRE[3] to '0'; GRE[4] must be '1'. • Clear GRE[4] to '0'. • Ensure interrupts are disabled (if necessary, execute a CLI). • Wait for two '0'-to-'1' transitions of 3DA[0] (the Status Port Display Enable bit) • Interrupts can be restored at this point

8.24 GRE: Power Management *(cont.)*

Bit	Description
3	**Static Clock Mode:** If this bit is set to '1', the CL-GD5446 is placed in Static Clock mode. VCLK and MCLK are gated off so the device does not dissipate any dynamic power. DRAM refresh continues. When the device is in Static Clock mode, memory reads and writes are not valid. I/O accesses other than Palette registers are valid.
2	**Static VSYNC:** If this bit is set to '1', the VSYNC output is static. The level is as programmed into MISC[7]. In addition, the DAC is placed in the power-down state. The palette can still be accessed if the device is not powered down with GRE[4:3]. If this bit is set to '0', the VSYNC pin functions as programmed by the CRTC registers.
1	**Static HSYNC:** If this bit is set to '1', the HSYNC output is static. The level is as programmed into MISC[6]. In addition, the DAC is placed in the power-down state. The palette can still be accessed if the device is not powered down with GRE[4:3]. If this bit is set to '0', the HSYNC pin functions as programmed by the CRTC registers.
0	**DCLK Output ÷2:** If this bit is set to '0', the CL-GD5446 operates normally. If this bit is set to '1', the CL-GD5446 simulates external DAC Clocking mode 1. The rising edge of DCLK can clock the low byte of 16-bit data; the falling edge of DCLK can clock the high byte of 16-bit data. If the V-Port is enabled (CR50[1:0]), '1' also inverts the input on DCLK.

8.25 GR16: Active Display Line Readback Byte 0

I/O Port Address: 3CFh
Index: 16h

Bit	Description
7	Active Display Line [7]
6	Active Display Line [6]
5	Active Display Line [5]
4	Active Display Line [4]
3	Active Display Line [3]
2	Active Display Line [2]
1	Active Display Line [1]
0	Active Display Line [0]

This read-only register contains the least-significant 8 bits of the current active display line.

Bit	Description
7:0	**Active Display Line [7:0]:** This register can determine the current active display line. The least-significant 8 bits are returned in this read-only register. The 2 most-significant bits are returned in GR17[1:0]. **Programming Note:** GR17[1:0] and GR16[7:0] provide a method of determining the current scanline. When either of these registers are read, the appropriate bits of the vertical line counter are returned. Since these registers are not accessible with memory-mapped I/O, it is impossible to read them at the same instant. The programmer must consider the consequences of the counter having been incremented in the interval between the reads. For example, if the least-significant field increments from FFh to 00h in this interval, the results are either 255 too high or 256 too low, depending on the order the fields were read in. To resolve this, first read register GR17, then register GR16. If register GR16 is not equal to zero when it is read, the programmer can be certain that register GR17 did not change (either as a result of a carry from register GR16 or as a result of the counter being cleared) after it was read and that the results are self-consistent.

8.26 GR17: Active Display Line Readback Byte 1

I/O Port Address: 3CFh
Index: 17h

Bit	Description	Reset State
7	Reserved	
6	Programmable Output1	0
5	Programmable Output0	0
4	V-Port Interrupt Status	
3	Feature Connector Input	$\overline{CF6}$
2	INTR# Disable	0
1	Active Display Line [9]	
0	Active Display Line [8]	

This register contains the two most-significant bits of the active display line counter.

Bit **Description**

7 **Reserved**

6:5 **Multimedia Support:** When the V-Port is enabled, these two bits are for programmable output for multimedia support.

GR17 Bit	Description	CL-GD5446 Pin	Direction
6	Programmable Output 1	ESYNC# (pin 122)	O
5	Programmable Output 0	MCLK (pin 16)	O

4 **V-Port Interrupt Status:** This read-only bit returns the status of the V-Port interrupt if enabled in CR3F[1].

3 **Feature Connector Input:** When this bit is ‘1’, the feature connector is configured for input. The output drivers on the following pins are disabled: P[7:0], DCLK, BLANK#, and EVIDEO#. If a pull-down resistor is installed on MD54, this bit is set to ‘1’ at RESET. When this bit is ‘0’, these drivers are normally controlled. HSYNC and VSYNC are not affected by this bit.

2 **Disable INTR#:** This bit is used only if a pull-down resistor is installed on MD62 to claim a PCI interrupt. If so and this bit is ‘0’, an internal interrupt condition is prevented from driving the IRQ# pin low. If this bit is ‘1’ and the interrupt is claimed, an internal interrupt condition causes IRQ# to be driven low. See CR3F[2] to the determine source of the interrupt request.

1:0 **Vertical Line Counter [9:8]:** This field is the two most-significant bits of the vertical line counter. See Section 8.25.

8.27 GR18: Extended DRAM Controls

I/O Port Address: 3CFh
Index: 18h

Bit	Description	Reset State
7	Reserved	
6	Enable 16-bit Pixel Bus	
5	Enable CD and Overlay Interpolation	
4	Tristate DRAM Interface	0
3	Single Refresh Cycle	0
2	Enable 8-MCLK EDO Timing	0
1	Decreased Write Following Read Timing	0
0	Decreased WE# Active Delay	0

This register contains bits for DRAM timing controls. This register is reset to '0'.

Bit	Description
7	**Reserved**
6	**Enable 16-bit Pixel Bus:** If this bit is set to '1', the 16-bit pixel bus extension is enabled. This requires that the BIOS ROM is disabled by programming PCI30[0] to '0'. In addition, GPIO (see Section 8.28) must not be enabled (programmed for 16-bit data).
5	**Enable Clock Doubling and Overlay Interpolation:** If this bit is set to '1', 16-bits from pixel bus can be used to overlay 8-bit LUT data at pixel rates up to 80 MHz. The 16-bit data is clocked in at 1/2 the pixel rate and interpolation creates the extra pixels. This mode must be used with the 16-bit pixel bus extension.
4	**Tristate DRAM Interface:** If this bit is set to '1', the DRAM interface is forced to the high-impedance state. The pins are MD[63:0], MA[9:0], CAS#[7:0], RAS#[1:0], and WE#. Since this is not synchronized with the DRAM state machine, it is necessary to suppress DRAM activity prior to setting this mode. See Section 8.24. The lines must be driven externally to prevent them from floating to the threshold.

8.27 GR18: Extended DRAM Controls *(cont.)*

Bit	Description
3	**Single Refresh Cycle:** If this bit is set to '1', the CL-GD5446 generates a single CAS#-before-RAS# refresh cycle per scanline, allowing more MCLK cycles for CPU access. The value programmed into CR11[6] is ignored. This mode should be enabled only when the horizontal frequency is high enough to guarantee the DRAM refresh period specification is met. If this bit is set to '0', the CL-GD5446 generates either three or five refresh cycles per scanline according to the value in CR11[6]
2	**Enable 8-MCLK EDO Timing:** If this bit is set to '1', the CL-GD5446 generates EDO DRAM timing. The CL-GD5446 must have been configured for Extended RAS# timing (pull-down resistor installed on MD57). See Appendix B3, "Memory Configurations and Timing", for a detailed discussion of EDO support. In addition, programming this bit to '1' inserts an extra MCLK cycle between read and write CAS# of BitBLT operations that modify the destination using extended page cycles (such as, DST INV).
1	**Decreased Write Following Read Timing:** If this bit is set to '1', the timing for a write CAS# immediately following a read CAS# is reduced by one MCLK as shown in the following table.
0	**Decreased WE# Active Delay:** If this bit is set to '1', the timing to make WE# for a cycle immediately following a read cycle is reduced by one MCLK, as shown in the following table.

GR18[1]	GR18[2] = 0	GR18[2] = 1
0	3	4
1	2	3

GR18[1]	GR18[2] = 0	GR18[2] = 1
0	2	3
1	1	2

8.28 GR19: GPIO Port Configuration

I/O Port Address: 3CFh
Index: 19h

Bit	Description
7:6	Reserved
5	Minimum RW Strobe Active
4	CS# to RW Strobe Delay
3	RW Strobe to RDY Sample
2	Address to CS# Delay
1	Cycle Termination Selection
0	GPIO Bus Width

This register contains controls for the GPIO port. GPIO must be enabled with configuration bits CF3, CF4, and CF8. See Appendix B11, “General-Purpose I/O”, for detailed information on GPIO.

Bit	Description
7:6	**Reserved**
5	**Minimum RW Strobe Active:** If this bit is ‘0’, the IORD# or IOWR# strobe is a minimum of two MCLKs. If this bit is ‘1’, the IORD# or IOWR# strobe is a minimum of four MCLKs.
4	**CS# to RW Strobe Delay:** If this bit is ‘0’, there is a one CLK delay from CS# active to IORD# or IOWR# active. If this bit is ‘1’, there is a two CLK delay from CS# active to IORD# or IOWR# active.
3	**RW Strobe to RDY Sample:** If this bit is ‘0’, there is a one CLK delay from IORD# or IOWR# active until RDY# is sampled for the first time. If this bit is ‘1’, there is a two CLK delay from IORD# or IOWR# active until RDY# is sampled for the first time.
2	**Address to CS# Delay:** If this bit is ‘0’, there is a one CLK delay from address valid to CS# active. If this bit is ‘1’, there is a two CLK delay from address valid to CS# active.
1	**Cycle Termination Selection:** If this bit is ‘0’, the cycle terminates when RDY is sampled high. If this bit is ‘1’, the cycle terminates with the rising edge of DTACK. See Appendix B11, “General-Purpose I/O”.
0	**GPIO Bus Width:** If this bit is ‘0’, the GPIO is configured for an 8-bit data bus. If this bit is ‘1’, the GPIO is configured for a 16-bit bus.

8.29 GR1A–GR1B: Scratch Pad 4, 5

I/O Port Address: 3CFh
Index: 1Ah, 1Bh

Bit	Description	Reset State
7	R/W Data [7]	0
6	R/W Data [6]	0
5	R/W Data [5]	0
4	R/W Data [4]	0
3	R/W Data [3]	0
2	R/W Data [2]	0
1	R/W Data [1]	0
0	R/W Data [0]	0

IMPORTANT: These two registers are reserved for the exclusive use of the CL-GD5446 BIOS, and must never be written to by an application program.

This register description is included for completeness only.

Bit	Description
7:0	These bits are reserved for the Cirrus Logic BIOS.

8.30 CR19: Interlace End

I/O Port Address: 3D5h
Index: 19h

Bit	Description
7	Interlace End [7]
6	Interlace End [6]
5	Interlace End [5]
4	Interlace End [4]
3	Interlace End [3]
2	Interlace End [2]
1	Interlace End [1]
0	Interlace End [0]

This register holds the ending horizontal character count for the odd field VSYNC.

Bit	Description
7:0	**Interlace End:** This value is the number of characters in the last scanline of the odd field in interlaced timing. This can be adjusted to center the scanlines in the odd field halfway between scanlines in the even field. This register is typically set to approximately half the horizontal total.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.31 CR1A: Miscellaneous Control

I/O Port Address: 3D5h
Index: 1Ah

Bit	Description
7	Vertical Blank End Overflow [9]
6	Vertical Blank End Overflow [8]
5	Horizontal Blank End Overflow [7]
4	Horizontal Blank End Overflow [6]
3	Overlay/DAC Mode Switching Control [1]
2	Overlay/DAC Mode Switching Control [0]
1	Enable Double Buffered Display Start Address
0	Enable Interlaced

This register contains timing overflow bits as well as miscellaneous control bits.

Bit **Description**

7:6 **Vertical Blank End Overflow [9:8]:** This 2-bit field extends the vertical blank end value to 10 bits. Refer to Table 4-2 on page 4-28 for the timing value bits. These bits are enabled only if CR1B[5] is set to '1' or if CR1B[7] is set to '1'.

These two bits are intended for use when CR1B[5] is set to '1' and the blank counters control OVRW#.

5:4 **Horizontal Blank End Overflow [7:6]:** This 2-bit field extends the horizontal blanking end value to 8 bits. Refer to Table 4-2 on page 4-28 for the timing value bits. These bits are enabled only if CR1B[5] is set to '1', or if CR1B[7] is set to '1'.

These two bits are intended for use when CR1B[5] is set to '1' and the blank counters control OVRW#.

3:2 **Overlay/DAC Mode Switch Control [1:0]:** This field selects the Overlay and DAC Mode Switch term. The values are summarized in the following table. Refer to Section 9.7 on page 9-47 for further information.

CR1A[3]	CR1A[2]	Switch Enabled With	P[7:0]
0	0	Normal operation (switch disabled)	Outputs
0	1	Timing: (EVIDEO# or OVRW#)	Inputs
1	0	Timing AND'ed with color key	Inputs
1	1	Color key only	Inputs

8.31 CR1A: Miscellaneous Control *(cont.)*

Bit	Description
1	**Enable Double Buffered Display Start Address:** If this bit is set to '1', the display start address updates on the VSYNC following a write to start address low. This provides control of display frame switching without the need to explicitly monitor VSYNC. Setting this bit to '1' also enables the double-buffering of the Video Window position and width in registers CR31–CR39.
0	**Enable Interlaced:** If this bit is set to '1', interlaced timing is enabled. Interlaced timing means interlaced sync in Text mode, and interlaced sync and display data in Graphics mode. In addition, IRQ# requests are only generated at the end of odd fields (that is, at the end of a frame). For interlaced sync and data in Graphics mode, the CRTC Scan Double bit (CR9[7]) must be set to '0'. Graphics modes 4 and 6 must always be non-interlaced.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.32 CR1B: Extended Display Controls

I/O Port Address: 3D5h
Index: 1Bh

Bit	Description
7	Enable Blank End Extensions
6	Enable Text Mode Fast-page
5	Blanking Control
4	Offset Register Overflow [8]
3	Screen Start A Address [18]
2	Screen Start A Address [17]
1	Enable Extended Address Wrap
0	Extended Display Start Address [16]

This register contains bits that control extended display functions.

Bit	Description
7	**Enable Blank End Extensions:** If this bit is set to '0', the Vertical and Horizontal Blank End Extension bits in register CR1A are disabled if CR1B[5] is also '0'. If this bit is set to '1', the Vertical and Horizontal Blank End Extension bits in CR1A are enabled, regardless of the programming of CR1B[5].
6	**Enable Text Mode Fast-page:** If this bit is set to '0', all font fetch cycles occur as random read cycles. This bit must be set to '0' for standard VGA dual-font operations. If this bit is set to '1', Fast-page mode cycles fetch font data. This allows for Text modes with a VCLK greater than 30 MHz, as is required for 132-column modes. **NOTE:** 132-column modes are not supported in the Cirrus Logic BIOS.
5	**Blanking Control:** If this bit is set to '0', the DAC blanking is controlled by the Blanking Signal generated by the CRTC. In this case, the border can be used (Refer to Section 4.62 for a description of AR11). If this bit is set to '1', the DAC Blanking is controlled by display enable. The DAC is blanked during the time when the border is normally displayed. In addition, the OVRW# pin follows the blanking signal generated by in the CRTC. This signal can be directed to the feature connector or used to control an external overlay circuit. Finally, programming this bit to '1' enables the Vertical and Horizontal Blank End Extension bits in register CR1A.

8.32 CR1B: Extended Display Controls *(cont.)*

Bit	Description
4	**Offset Register Overflow [8]:** This bit extends the CRTC Offset register (CR13) by one bit. Refer to Table 4-2 on page 4-28 for a summary of CRTC Timing register.
3:2	**Screen Start A Address [18:17]:** These two bits extend the Screen Start A Address.
1	**Enable Extended Address Wrap:** If this bit is set to '0', the Display Memory Address wraps at 64K maps (256K total memory) providing VGA compatibility. If this bit is set to '1', the Display Memory Address wraps at the total available memory size. In particular, this bit provides the following functions: If this bit is set to '1', and Chain-4 addressing is selected (SR4[3] = 1), then DRAM Addresses A0 and A1 are supplied from Addresses XMA[16] and XMA[17]. XMA[18:12] Addresses are the sum of XA[16:12] and either Offset register 0 or 1. If this bit is set to '1' and CRTC Doubleword Addressing is selected (CR14[6] = 1), then DRAM addresses A0 and A1 are supplied from CRTC Addresses CR[14] and CR[15]. This provides four displayable pages in display mode 13h. Character counter addresses CA[16] and CA[18] provide up to 256 Kbytes in each bit plane, or 1 Mbyte of packed-pixel memory. If this bit is set to '0', the CRTC character address counter is 16-bits wide, providing VGA compatibility. If this bit is set to '1', the CRTC character address counter is 19-bits wide.
0	**Extended Display Start Address [16]:** This is bit 16 of the Extended Display Start Address.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.33 CR1C: Sync Adjust and GENLOCK

I/O Port Address: 3D5h
Index: 1Ch

Bit	Description
7	Enable VSYNC GENLOCK
6	Enable HSYNC GENLOCK
5	Horizontal Total Adjust [2]
4	Horizontal Total Adjust [1]
3	Horizontal Total Adjust [0]
2	Horizontal Sync Start Adjust [2]
1	Horizontal Sync Start Adjust [1]
0	Horizontal Sync Start Adjust [0]

This register enables GENLOCK for horizontal timing adjustments.

Bit	Description
7	**Enable VSYNC GENLOCK:** If this bit is set to '1', VSYNC GENLOCK is enabled. The VSYNC pin becomes an input. See Appendix B9, "GENLOCK Support". If the CL-GD5446 is programmed for interlaced operation, the first transition sampled after this bit is set to '1' also clears the interlace field state to 'even'.
6	**Enable HSYNC GENLOCK:** If this bit is set to '1', HSYNC GENLOCK is enabled and the HSYNC pin becomes an input. The value programmed into the Horizontal Total register must be so that the external HSYNC occurs before the programmed value is reached. This function is independent of VSYNC GENLOCK. Either VSYNC GENLOCK or HSYNC GENLOCK is used with an external VCLK derived externally from the video source and supplied to DCLK (EDCLK is Low). MISC register 3C2[3:2] must be set to '1X' so that the externally generated DCLK drives the CRT Controller, as well as the DAC. The external master supplies HSYNC and VSYNC to the display, as well as to the CL-GD5446.

8.33 CR1C: Sync Adjust and GENLOCK *(cont.)*

Bit	Description
5:3	**Horizontal Total Adjust [2:0]:** This field allows for a –3/+4 VCLK adjustment of the programmed Horizontal Total. The length of the character, which occurs two character clocks after the horizontal counter has reached the value programmed into horizontal total, is adjusted according to this field. The following table shows this adjustment:

CR1C[5:3]	Character Clock Adjustment
000	0 (Normal)
001	–3 VCLKs
010	–2 VCLKs
011	–1 VCLKs
100	+1 VCLKs
101	+2 VCLKs
110	+3 VCLKs
111	+4 VCLKs

2:0	**Horizontal Sync Start Adjust [2:0]:** This field allows a 0–7 VCLK adjustment of the position of Horizontal Sync Start (relative to BLANK#). HSYNC is delayed the additional number of VCLKs programmed in this field. The HSYNC width is still adjustable in character clock increments.

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.34 CR1D: Overlay Extended Control

I/O Port Address: 3D5h
Index: 1Dh

Bit	Description
7	Screen Start A Address [19]
6	Overlay Timing Select
5	Color Key / Chroma Key Select
4	Enable Color Key Tag
3	Color Key Compare Width
2	DAC Mode Switching Control [1]
1	DAC Mode Switching Control [0]
0	Reserved

This register contains a number of bits that extend the overlay functions.

Bit	Description
7	**Screen Start A Address [19]:** This is bit 19 of the Display Start Address.
6	**Overlay Timing Select:** If this bit is '0', the EVIDEO# input is the overlay timing source. If this bit is '1', the internal OVRW# source is overlay timing source.
5	**Color Key / Chroma Key Select:** If this bit is '0', graphics data is compared to the color key in registers GRC/GRD. If this bit is '1', video data is compared to the chroma key in registers GRC, GRD, and GR1C–GR1F. See Section 9.5.8 on page 9-39.
4	**Enable Color Key Tag:** If 16-bit graphics color key compare is selected, setting this bit to '1' enables tag bit color key. If GRD[7] is equal to bit 15 of a 16-bit pixel, this constitutes a color match and the graphics pixel is replaced. The table in Section 8.22 presents a summary of the various color/chroma key modes.
3	**Color Compare Width:** If this bit is set to '1', all 16 bits of each pixel are the color key compare. The low byte of the graphics data is compared to the value in register GRC; the high byte of the graphics data is compared to the value in register GRD. Since register GRD is the high-byte comparand, no mask is available. This feature is for 16-bpp graphics modes only. If this bit and CR1D[4] are both '1', color key tag compare is enabled.

8.34 CR1D: Overlay Extended Control *(cont.)*

Bit	Description
2:1	**DAC Mode Switching Control [1:0]:** This 2-bit field controls DAC mode switching. This controls only the DAC mode switching; the enabling of video overlay is still done as programmed in CR1A[3:2].

CR1D[2:1]	DAC Mode Switching
00	Follows Switch as programmed in register CR1A.
01	Opposite Switch as programmed in register CR1A.
1X	Disable DAC Mode Switching.

0	**Reserved**

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.35 CR25: Part Status (Read only)

I/O Port Address: 3D5h
Index: 25h

Bit	Description	Reset State
7	PSR [7]	0
6	PSR [6]	1
5	PSR [5]	
4	PSR [4]	
3	PSR [3]	
2	PSR [2]	
1	PSR [1]	
0	PSR [0]	

This read-only register is for factory testing and internal tracking only. This register description is included for completeness only.

NOTE: Application programs need never read this register.

Bit	Description
7:0	**Part Status Value [7:0]**

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.36 CR27: ID (Read only)

I/O Port Address: 3D5h
Index: 27h

Bit	Description
7	Device ID [5]
6	Device ID [4]
5	Device ID [3]
4	Device ID [2]
3	Device ID [1]
2	Device ID [0]
1:0	Reserved

This read-only register returns an identifying value. Applications programs need not read this register if a Cirrus Logic BIOS is available. Instead, applications should use the Inquire VGA Type INT10 call. See Appendix C3, “BIOS Extensions”.

Bit	Description
7:2	**Device ID [5:0]:** This 6-bit field contains a unique identifier, as shown in the following table:

Product	CR27[7:2]
CL-GD5446	101110

1:0	**Reserved**

NOTE: If the CL-GD5446 is programmed for Monochrome mode, the registers at 3Dxh are at 3Bxh.

8.37 HDR: Hidden DAC Register

I/O Port Address: 3C6h
Index: –

Bit	Description	Reset State
7	Enable 5:5:5 Mode	0
6	Enable All Extended Modes	0
5	Clocking Mode	0
4	32K Color Control	0
3	Extended Mode Select [3]	0
2	Extended Mode Select [2]	0
1	Extended Mode Select [1]	0
0	Extended Mode Select [0]	0

This register selects the Extended Color modes, including 15-, 16-, and 24-bpp modes. This register is cleared to '0' at reset, placing the CL-GD5446 in VGA-Compatibility mode.

This register is accessed four times in succession by a read of the Pixel Mask register at 03C6h; the next write or read at 03C6h accesses the HDR. Subsequent accesses require four accesses to the Pixel Mask register.

Bit	Description
7	**Enable 5:5:5 Mode:** If this bit is set to '0', the Extended Color modes are disabled and the palette DAC is VGA-compatible. If this bit is set to '1', Extended Color modes are enabled, as selected in bit 6 and bits 3:0 of this register.
6	**Enable All Extended Modes:** If this bit is set to '0' and bit 7 is set to '1', the palette DAC is in 5:5:5 Sierra™ mode, regardless of the values set in bits 3:0. If this bit is set to '1' and bit 7 is set to '1', the Palette DAC mode is selected by the values set in bits 2:0 of this register.
5	**Clocking Mode:** If this bit is set to '0', Clocking mode 1 is selected. In Clocking mode 1, 16-bpp modes use both edges of DCLK to latch data. The rising edge of DCLK latch the least-significant byte; the falling edge of DCLK latch the most-significant byte. If this bit is set to '1', Clocking mode 2 is selected. In Clocking mode 2, 16-bpp modes use only the rising edge of DCLK to latch data. The DCLK must be supplied at twice the pixel rate. The least-significant byte is latched on the first rising edge and the most-significant byte is latched on the second rising edge. This mode is not used when running video capture applications.

8.37 HDR: Hidden DAC Register *(cont.)*

Bit	Description
4	**32K Color Control:** If this bit is set to '0', 5:5:5 operation occurs normally. If this bit is set to '1', Pixel Data bit 15 selects between palette operation and 5:5:5 color. This allows 5:5:5 data to overlay 256-color images on a pixel-by-pixel basis. If Pixel Data bit 15 is '1', then bits 7:0 select a palette entry and bits 14:8 are ignored. If pixel bit 15 is '0', then 5:5:5 operation is selected.
3:0	**Extended Mode Select [3:0]:** If bits 7 and 6 are both set to '1', then this 4-bit field selects the Extended Color mode according to the following table.

Bit 7	Bit 6	Bit 3	Bit 2	Bit 1	Bit 0	Function
0	X	X	X	X	X	VGA compatibility (Palette mode)
0	1	1	0	1	0	Palette mode > 85 MHz [a]
1	0	X	X	X	X	5:5:5 Sierra™
1	1	0	0	0	0	5:5:5 Sierra™ [b]
1	1	0	0	0	1	5:6:5 XGA™
1	1	0	0	1	X	Reserved
1	1	0	1	0	0	Reserved
1	1	0	1	0	1	8:8:8 16M colors (24- or 32-bit)
1	1	0	1	1	X	Power-down DAC [c]
1	1	1	0	0	0	8-bit grayscale
1	1	1	0	0	1	3:3:2 8-bit RGB
1	1	1	0	1	X	Reserved
1	1	1	1	X	X	Reserved

[a] This mode is used for 8-bit-per-pixel modes where VCLK is > ~85 MHz. The CRTC is clocked at 1/2 the programmed VCLK rate. SR7[3:0] must be set to '0111' (16-bpp and high-resolution). The 16-bit data is sent through the LUT one byte at a time at the doubled VCLK rate.

[b] HDR[5] selects the clocking mode for the 15- and 16-bit modes.

[c] The result of programming this pattern is identical to the side effect of programming GRE[2] or GRE[1] to '1'; IREF is turned off, greatly reducing the power consumed in the DACs; VCLK to the palette is gated off, placing it in low-power static operation.

Programming Notes

9. PROGRAMMING NOTES

9.1 Introduction

This chapter contains information on programming the CL-GD5446. Topics discussed are:

- Formats and organization of pixels for each pixel format in the frame buffer
- The BitBLT engine with text and comprehensive examples
- The hardware video window and video capture
- The hardware cursor
- Synthesizer programming
- Power management
- CRTC programming
- Chip identification
- Code fragments
- Programming examples

This chapter is intended to be used in conjunction with the rest of this manual, especially the register chapters. This chapter (or at least the sections of it that apply to the programming exercises) should be read in its entirety.

Cirrus Logic provides a comprehensive BIOS for the CL-GD5446. This should be used whenever possible. If an application needs to directly program registers (except perhaps the BitBLT and V-Port registers), this is an indication that the BIOS is not being properly utilized. The BIOS is covered in Appendix C2, “VGA BIOS”, and Appendix C3, “BIOS Extensions”.

9.2 Resource Addressing

The CL-GD5446 has three general classes of resource accessible: Registers, frame buffer, and BIOS ROM. Accessing these resources is discussed in the following sections.

9.2.1 VGA Registers

9.2.1.1 Standard I/O Addresses

All registers on the CL-GD5446 except the PCI configuration registers can be accessed for read or write in the standard I/O space. Table 9-1 lists the I/O addresses used by the CL-GD5446.

Table 9-1. Standard I/O Addresses

Address (hex)	Use Registers	Index/Data Pair	Monochrome/ Color
3B4–3B5	CRTC	✔	Monochrome
3BA	Feature Control/Status1		Monochrome
3C0–3C1	Attribute Controller	✔	Both
3C2	MISC Out/Status 0		Both
3C4–3C5	Sequencer Registers	✔	Both
3C6–3C9	DAC Palette		Both
3CA	FEAT Readback		Both
3CC	MISC Out Readback		Both
3CE–3CF	Graphics Controller Registers	✔	Both
3D4–3D5	CRTC Registers	✔	Color
3DA	Feature Control/Status 1		Color

The CRTC Controller registers are at 3B4–3B5 or 3D4–3D5 when the CL-GD5446 is programmed for monochrome or color, respectively. This manual uses registers 3D4–3D5.

All of these registers are 8 bits (some bits are reserved on some registers).

Many of the addresses are for a number of registers; an index specifies the register at a given address is to be written or read. For example, the Graphics Controller Index and Data addresses are 3CEh and 3CFh, respectively. The index is written and then the data is written or read. This is illustrated in the following code fragment:

```
            mov         dx,3ceh                 ;point to graphics controller
index
            mov         al,5h                   ;will access GR5 (mode
register)
            out         dx,al                   ;set index
            inc         dx                      ;point to data address
            in          al,dx                   ;and read the register
```

The index and data can be written in a single cycle; this is often done to increase throughput. The index and data are concatenated to form a single 16-bit operand, written to the address of the index (the even address). The index occupies the least-significant byte (AL) and the data occupies the most-significant byte (AH). For example, to write the value 21h to the MCLK Select register (SR1F), the following code fragment could be used.

```
mov          ax,211fh    ;index is 1f, data is 21
mov          dx,3c4h     ;sequencer port
out          dx,ax       ;write the index and the data
```

Addressing the Attribute Index and Data ports is slightly different. The distinction between the index and data ports is not made by means of the address, but rather by an internal flip-flop that toggles between index and data. There are still two addresses: 3C0 writes index or data and 3C1 is used for reads. The internal toggle can be read at register CR24, but it is easier to force it to 'index' by reading the STAT register at 3DA (3BA for monochrome). When the index is written, bit 5 must be '0', but then must be programmed to '1' to leave the display on. The following code fragment can be used set the border color in register AR11:

```
mov          dx,3dah     ;point to STAT (color mode)
in           al,dx       ;force the toggle to index
mov          dx,3c0h     ;point to attribute controller write port
mov          al,11h      ;AR11 is Overscan (border) color
                         ;leaves toggle pointing at data port
out          dx,al       ;write the index, bit five (20h) must be 0
mov          al,2h       ;green is my favorite color
out          dx,al       ;use 3C0 for writes, 3C1 for reads
                         ;leaves toggle pointing at index port
mov          al,20h      ;will now turn display back on
out          dx,al       ;leaves toggle pointing at data port
```

9.2.1.2 VGA Register Relocation (Revision A)

The VGA I/O registers can be relocated. This is part of the capabilities necessary to allow multiple controllers in a single system. PCI14 is used in conjunction with three Configuration bits (CF8, CF4, and CF3) to control VGA register relocation and general-purpose I/O (see Section 9.2.4). Table 9-2 summarizes these controls. The CF (Configuration) bits are described in detail in Appendix B5, "Configuration Notes".

Table 9-2. VGA Register Relocation and GPIO Controls

CF8 MD56	CF4 MD52	CF3 MD51	VGA Register Relocation	GPIO	PCI14	Address Space
X	X	1	Disabled	Disabled	Returns all '0's	–
0	0	0	Disabled	128 bytes I/O	15:7 specify address bit 0 = 1	128 bytes
1	0	0	Disabled	16 bytes I/O	15:5 specify address bit 0 = 1	32 bytes
0	1	0	32 bytes memory	32 bytes memory	31:12 specify address bit 0 = 0	4 Kbytes
1	1	0	32 bytes I/O	Disabled	15:5 specify address bit 0 = 1	32 bytes

When VGA register relocation is enabled, the VGA registers occupy a 32-byte block of address space at the address specified in PCI14. Depending on how CF8 and CF4 are set up, the block can be in either I/O or memory space. When PCI14 is configured for I/O, the address bits above bit 15 are not decoded. The offsets from the beginning of the 32-byte block are given in Table 9-3.

Table 9-3. VGA Registers Relocation Offset

VGA Address (hex)	Use Register	Offset (hex)
3C0/3C1	Attribute Controller	00/01
3C2	MISC Out/Status 0	02
3C4/3C5	Sequencer	04/05
3C6–3C9	DAC Palette	06–09
3CA	Feat Readback	0A
3CC	MISC Out Readback	0C
3CE/3CF	Graphics Controller	0E/0F
3D4/3D5	CRTC (MISC[0] = 1)	14/15
3DA	Feature Control/Status 1	1A

9.2.1.3 VGA Register Relocation (Revision B)

On Revision B of the CL-GD5446, the VGA registers are always accessible in the memory space claimed in PCI14, regardless of configuration resistors. The VGA registers are accessible in the first 32 bytes of this space. The VGA registers must always be accessed with byte or word reads or writes. The offsets of the registers are given in Table 9-3.

9.2.2 BitBLT Registers: Memory-Mapped I/O (Revision A)

The registers that control the BitBLT engine are addressable in memory space as well as I/O space. When memory-mapped I/O is used, up to 4 bytes can be written with a single operation. This increases throughput significantly, especially since the addresses are carefully allocated so that registers with a similar function are grouped into a common dword. In addition, PCI bursts can occur.

Memory-mapped I/O is enabled by setting SR17[2] to ‘1’. This reserves 256 bytes of memory space. If SR17[6] is ‘0’, the space is at 000B8000h (actually, the 256-byte block is aliased every 256 bytes from 000B8000h–000BBF00h). If linear addressing is enabled and SR17[6] is ‘1’, the space will be the upper 256 bytes of the populated memory in each 4-Mbyte aperture. In this case, the upper 256 bytes of frame buffer is not accessible.

The registers can be written with byte, word, or dword accesses. They are write-only with memory-mapped I/O except for GR31, which is read/write. If write-only registers are read with memory-mapped I/O, the CL-GD5446 takes the cycle and returns TRDY#, but the data is indeterminate. When memory-mapped I/O is enabled, ordinary I/O can still be used, including read cycles.

Table 9-4 lists the registers accessible with memory-mapped I/O. Entries in Table 9-4 are separated into dword groups by a thick line.

Table 9-4. Memory-Mapped I/O Registers

Offset (hex) Revision A	Offset (hex) Revision B	Register	Description
00	100	GR0	Background Color Byte 0
01	101	GR10	Background Color Byte 1
02	102	GR12	Background Color Byte 2
03	103	GR14	Background Color Byte 3
04	104	GR1	Foreground Color Byte 0
05	105	GR11	Foreground Color Byte 1
06	106	GR13	Foreground Color Byte 2
07	107	GR15	Foreground Color Byte 3
08	108	GR20	BLT Width Byte 0
09	109	GR21	BLT Width Byte 1
0A	10A	GR22	BLT Height Byte 0
0B	10B	GR23	BLT Height Byte 1
0C	10C	GR24	BLT Dest Pitch Byte 0
0D	10D	GR25	BLT Dest Pitch Byte 1
0E	10E	GR26	BLT Source Pitch Byte 0
0F	10F	GR27	BLT Source Pitch Byte 1
10	110	GR28	BLT Dest Address Byte 0
11	111	GR29	BLT Dest Address Byte 1
12	112	GR2A	BLT Dest Address Byte 2
13	113	GR2B	Reserved
14	114	GR2C	BLT Source Address Byte 0
15	115	GR2D	BLT Source Address Byte 1
16	116	GR2E	BLT Source Address Byte 2
17	117	GR2F	Destination Write Mask
18	118	GR30	BLT Mode
19	119	–	Reserved
1A	11A	GR32	BLT Raster OP
1B	11B	GR33	BLT Mode Extensions

Table 9-4. Memory-Mapped I/O Registers *(cont.)*

Offset (hex) Revision A	Offset (hex) Revision B	Register	Description
1C	11C	GR34	Transparency Color Byte 0
1D	11D	GR35	Transparency Color Byte 1
1E	11E	–	Reserved
1F	11F	–	Reserved
20–3F	120–13F	–	Reserved
40	140	GR31	BLT Control (r/w)
41–FF	141–1FF	–	Reserved

9.2.3 BitBLT Register: Memory-Mapped I/O (Revision B)

On Revision B of the CL-GD5446 the BitBLT control register are always writeable at the memory address space claimed in PCI14. This address space is accessible regardless of how the is configured. Byte, word, and dword write accesses can be used. The BitBLT registers are addressed beginning at the base address plus 100 hex. The actual offsets are given in Table 9-4.

9.2.4 General-Purpose I/O (Revision A)

When the CL-GD5446 is installed on a PCI adapter card, it is often useful to be able to support an additional device (such as the CL-PX4072 MPEG-1 decoder) while still meeting the PCI single-load specification. The CL-GD5446 can be configured to support an 8- or 16-bit I/O port, providing address decoding and data buffering. GPIO is covered in detail in Appendix B11, “General-Purpose I/O”.

Table 9-2 on page 9-4 shows how GPIO is enabled (on Revision A) for either 32 or 128 bytes of I/O space or 32 bytes of memory space. When GPIO is enabled for memory space on Revision A (GPIO and VGA relocation enabled), GPIO begins 32 bytes above the location specified in PCI14 (the VGA registers come first.

When GPIO is configured for eight bits, byte, word, or dword accesses can be used. When GPIO is configured for 16 bits, word or dword accesses can be used; byte accesses cannot.

9.2.5 General-Purpose I/O (Revision B)

On Revision B of the CL-GD5446, GPIO is configured with CF8 and CF4. CF3 is unused. On Revision B, the address space for GPIO is claimed in PCI18. 32 bytes of memory or 32 bytes of I/O space can be claimed for GPIO. See Appendix B11, “General-Purpose I/O”.

9.2.6 PCI Configuration Registers

The CL-GD5446 supports the standard PCI configuration registers. See Section 9.14.3 on page 9-74 for an example of programming these registers. Table 9-5 lists the PCI configuration

registers. The PCI configuration registers are described in detail in Chapter 7, “PCI Configuration Registers”.

Table 9-5. PCI Configuration Registers

Offset (hex)	Name	Description	Access	Value
00	Vendor ID	Number assigned to Cirrus Logic	Read only	1013h
02	Device ID	Number assigned by Cirrus Logic	Read only	00B8h
04	Status/Command	Controls device ability to respond to PCI cycles and provides status information.	Read/write	
08	Class Code	Provides generic function of device.	Read only	030000xx
0C	Reserved		Read only	0000000h
10	Base Adrs 1	Frame Buffer Base Address	Read/write	
14	Base Adrs 2	VGA register/GPIO Base Address (Revision A)	Read/write	See CF3
14	Base Adrs 2	VGA/MMIO Base Address (Revision B)	Read/write	
18	Base Adrs 3	GPIO Base Address (Revision B)	Read/write	
18–2B	Reserved		Read only	00000000h
2C–2F	Reserved	(Revision A)	Read only	00000000h
2C–2F	Subsystem ID	Subsystem/Subsystem Vendor ID	See SR17	
30	ROM Base Adrs	Expansion ROM Base Address	Read/write	32K
34–3B	Reserved		Read only	00000000h
3C	Interrupt	Indicates that interrupt is assigned, if any.	Read/write	See CF14

9.2.7 Frame Buffer

9.2.7.1 VGA Compatibility

The CL-GD5446 supports standard VGA addressing of the first 32K or 64K of the frame buffer at 000A0000h or 000B8000h, according to the programming of GR6[3:2]. This is for the standard VGA modes.

9.2.7.2 CL-GD543X/’4X Compatibility

The first Mbyte of the frame buffer can be accessed through a single 64K window or two 32K windows at 000A000h. This is provided for backward compatibility with the Cirrus Logic CL-GD542X and CL-GD543X/’4X. Readers interested in this topic should consult the technical reference manual for the respective controller. All new software should be written to take advantage of linear addressing, as described in Section 9.2.7.3.

9.2.7.3 Linear Addressing

If SR7[7:4] is set to any value other than '0', the entire frame buffer is accessible as a linear string of bytes beginning at the address specified in PCI10. This addressing method should be used for any new software that use any Extended Graphics modes.

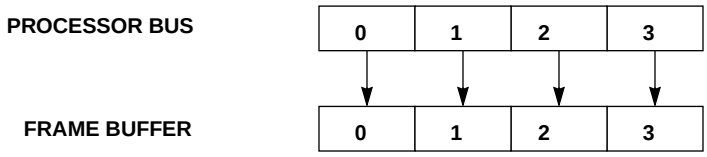
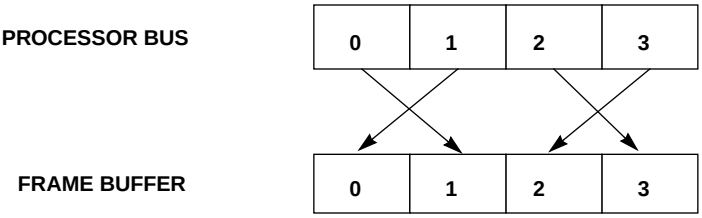
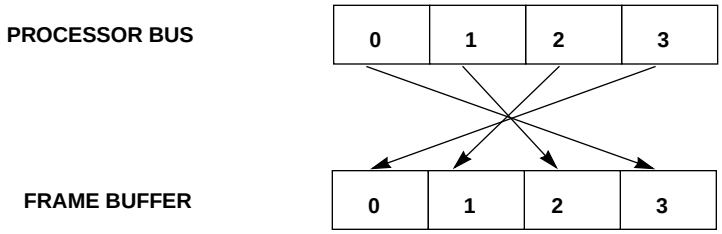
Revision A

PCI10 claims a 16-Mbyte segment, divided into three byte-swapping apertures of 4 Mbytes each, plus a video aperture. Each aperture can access all 4 Mbytes of frame buffer (except the last 256 bytes if MMI/O occupies them). According to the aperture used, the bytes within a word or dword are swapped as shown in Table 9-6 on page 9-9.

Revision B

PCI10 claims a 32-Mbyte segment. The first 16 Mbytes is divided into three byte-swapping apertures of 4 Mbytes each, plus a video aperture. These are used for direct access to the frame buffer. The second 16 Mbytes is divided into three byte-swapping apertures of 4 Mbytes each and an unused aperture. These are used for system-to-screen BitBLTs.

Table 9-6. Byte Swapping for Bi-Endian Support

Aperture	Swap	Diagram
0	No Swap	
1	Word Swap	
2	Dword Swap	
3	Video Aperture	See Section 9.2.7.4.

9.2.7.4 Video Aperture — YUV12 Planar Assist

MPEG or Indeo decoders keep Y (luminance) and U and V (chrominance) data in separate areas in memory. Packing these values from different buffers into YUYV dwords for the frame buffer is a compute-intensive process that the CL-GD5446 seeks to mitigate.

The fourth 4-Mbyte aperture can rearrange data from the YYYY, UUUU, and VVVV format generated by the decoder into YYVU pixels as required by the display pipeline. The CL-GD5446 must be configured for linear addressing, CR3E[3:1] must be '000' (YUV video data format) and CR3F must be '1'.

Each 1-Mbyte area in the fourth aperture accesses the same actual frame buffer memory as the corresponding Mbyte area in any other aperture. Each of the four 1-Mbyte areas are divided into three regions according to the offset into the area, as shown in Table 9-7.

Table 9-7. Frame Buffer Video Aperture

Offset Range	Data steered to:
0–512K	Y plane from system
512K–768K	V plane from system
768K–1024K	U plane from system

Figure 9-1 illustrates how each dword of source data at the PCI bus is transferred to the frame buffer as pixels of displayable data. Note that the format in the frame buffer is slightly different from YUYV; the display pipeline sorts this out.

An example of how this feature is used in presented in Section 9.14.1 on page 9-62.

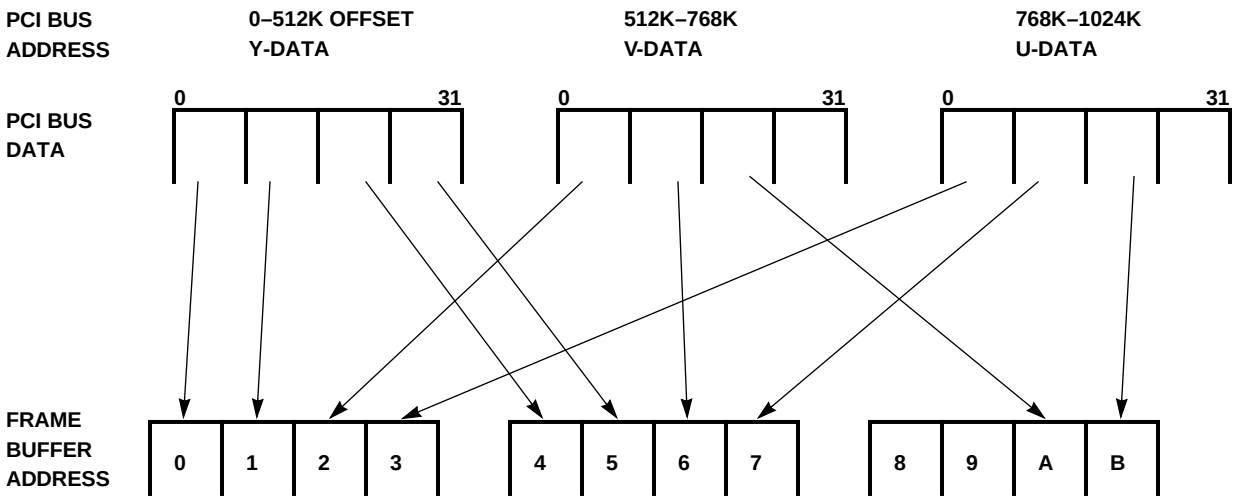


Figure 9-1. Frame Buffer Video Aperture

9.2.7.5 Alternative BitBLT System Aperture (Revision A)

If GR31[6] is '1', color-expand system-to-screen BitBLTs uses a 16-Kbyte block of addresses beginning at 000BC000h. This allows a process, other than the BitBLT driver, write accesses to the entire frame buffer at the PCI10 address even when a system-to-screen BitBLT is taking place.

9.2.8 BIOS PROM

Access to the BIOS ROM is controlled using PCI30. When PCI30[0] is '1', the BIOS ROM is readable at the address specified in PCI30. Typically, the BIOS is read from the ROM at POST time, and the PCI30[0] is set to '0'. The BIOS ROM is always 32 Kbytes.

9.2.9 HDR (Hidden DAC Register)

The HDR is accessed by reading the Pixel Mask register (3C6h) four times in succession. The next write or read accesses the HDR.

```
mov         dx,3c6h                 ;point to pixel mask register
in          al,dx
in          al,dx
in          al,dx
in          al,dx
mov         al,c6h                  ;power down DAC
out         dx,al                   ;write to HDR
```

9.3 Pixel Addressing and Formats

9.3.1 Pixel Addressing

The frame buffer contains a description of each dot (pixel) on the screen. For nearly all extended modes, the pixel organization is said to be 'packed-pixel', a term that simply means each pixel occupies contiguous bytes in memory and contiguous pixels occupy contiguous bytes in memory. 16-color modes use a planar organization.

Figure 9-2 on page 9-12 shows the relationship between the contents of the frame buffer and pixels on the screen. Pixels with higher addresses are displayed to the right of and beneath pixels with lower addresses. Pixels that are horizontally adjacent on the screen are adjacent in the frame buffer. Pixels that are vertically adjacent on the screen are separated in the frame buffer by a distance called the 'pitch'.

The pitch can be determined with the appropriate VESA function. Typically, but not always, this is equal to the pixel size times the number of pixels per scanline (the right-most pixel on scanline n is typically adjacent to the left-most pixel on scanline n + 1). Finally, the upper-left pixel (the first pixel) on the screen is typically, but not always, at the beginning of the frame buffer.

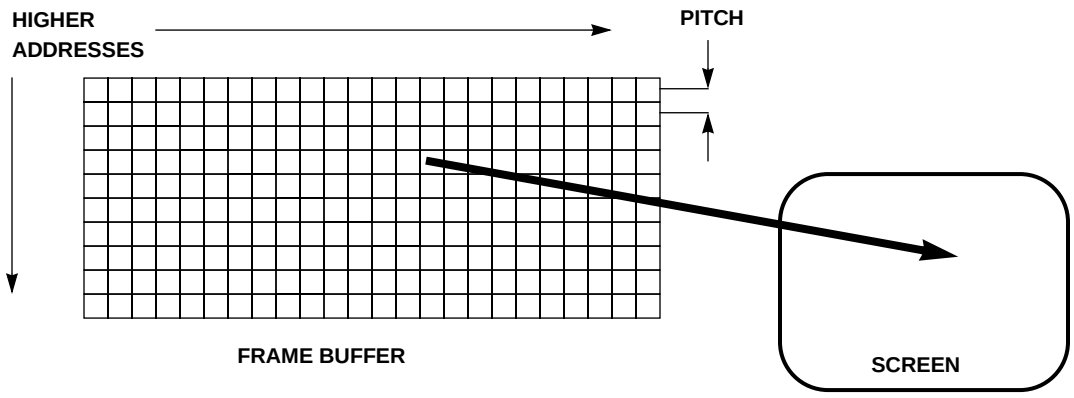


Figure 9-2. Pixel Organization

Use Equation 9-1 to compute the address of (the first byte of) any pixel, given the (zero-based) X and Y position, the pixel size, the pitch, and screen starting address.

$$PhysicalAddress = (X \bullet PixelSize) + (Y \bullet Pitch) + ScreenStartAddress + PCI10$$ **Equation 9-1**

The first element accounts for the displacement of the pixel within the containing scanline. *Pixel-Size* is in terms of bytes per pixel. The second element accounts for the vertical distance from the beginning of the screen to the beginning of the containing scanline. *Pitch* is also called the scan-line offset. The third element accounts for the possibility that the upper-left pixel is not at the beginning of the frame buffer. The fourth element is the beginning of the frame buffer in the host address space. For a given application, the last element and usually the next to last element are constants.

9.3.2 Pixel Formats

The CL-GD5446 supports a number of pixel formats. These formats are discussed in the following sections. Table 9-8 indicates how the BIOS programs the CL-GD5446 registers for these pixel formats (reference only). In general, use the BIOS to set extended modes.

Table 9-8. Pixel Formats

Pixel Format	Section	HDR	SR7[3:0]	CR3E[3:1]	Note
VGA compatibility (LUT)	9.3.2.1	00h	1h	010b	
VGA compatibility (>85 MHz)	9.3.2.1	CAh	7h	010b	Clock doubling
8-bpp grayscale	9.3.2.2	C8h	1h	–	Graphics only
8-bpp direct color	9.3.2.3	C9h	01h	–	Graphics only
8-bpp AccuPak™	9.3.2.4	–	–	001b	Video window only
5:5:5 mode with 32K colors	9.3.2.5	C0h	7h	–	
5:5:5 with 256-Color Mix mode	9.3.2.6	D0h	7h	100b	Graphics only

Table 9-8. Pixel Formats *(cont.)*

Pixel Format	Section	HDR	SR7[3:0]	CR3E[3:1]	Note
5:6:5 mode with 64K colors	9.3.2.7	C1h	7h	101b	
16-bpp YUV 4:2:2 mode	9.3.2.8	–	–	000b	Video window only
Planar assist YVU	9.3.2.9	–	–	000b	CR3F[4] = 1
8:8:8 mode with 16.8M colors	9.3.2.10	C5h	5h	–	Graphics only
8:8:8:8 mode with 16.8M colors and alpha	9.3.2.11	C5h	9h	–	Graphics only

9.3.2.1 VGA Compatibility (8-bpp Palettized)

This mode supports the industry-standard 8-bit 256 color palette mode. Each pixel occupies one byte in display memory. The value is an address into the color palette. The three 6-bit color values (one each for Red, Green, and Blue) from the corresponding location in the color palette are passed to the three DACs.

When the pixel rate required is above approximately 85 MHz, VCLK clock doubling must be used. This involves programming the VCLK synthesizer to one-half the required pixel rate and transferring pixels to the LUT two at a time, using an internal 16-bit data path. The LUT-DAC module generates a local clock at twice the programmed frequency, unpacks the pixel pairs, and performs the look-up and conversion at the local 2× clock. Clock doubling is enabled in the programming of the HDR. An example of a CL-GD5446 graphics mode using clock doubling is 1280 × 1024, 256 colors at 75 Hz refresh. The VCLK is programmed to 67.5 MHz, which the DAC module doubles to 135 MHz.

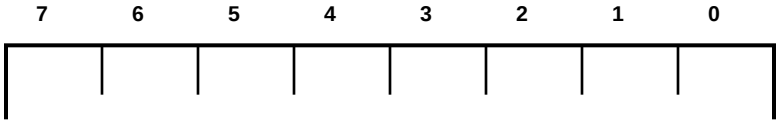


Figure 9-3. 8-bpp Palettized

9.3.2.2 8-bpp Grayscale

Each pixel occupies one byte in display memory. The byte is sent in parallel to all three DACs. The result on the screen is a gray pixel whose brightness corresponds to the value of the byte. The CLUT (palette) is not used in this mode.

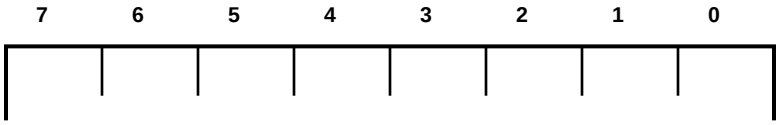


Figure 9-4. 8-bpp Grayscale

9.3.2.3 8-bpp Direct Color

Each pixel occupies 1 byte in display memory. The 8 bits are allocated to the three colors as indicated in Figure 9-5. This provides access to 256 fixed colors. The CLUT is not used in this mode.

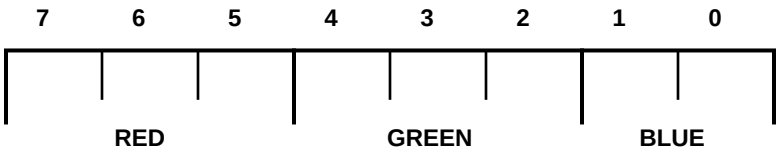


Figure 9-5. 8-bpp Direct Color

9.3.2.4 8-bpp AccuPak™ (YUV 4:1:1)

Each group of four adjacent pixels (packet) occupies four adjacent bytes in display memory. The packet format is shown in Figure 9-6. The Y (luminance) of each pixel is specified by 5 bits; the U and V (chrominance) values are specified by 6 bits each. Four pixels share a common chrominance. For display, each packet is converted to four pixels of RGB values. This involves interpolation and color space conversion.

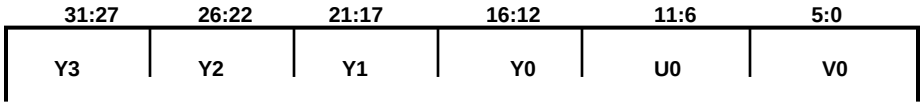


Figure 9-6. 8-bpp AccuPak™

9.3.2.5 5:5:5 Mode with 32K Colors

This format supports the RGB 5:5:5 mode with 32,768 colors. Each pixel is represented by 15 bits containing five bits each of Red, Green, and Blue color information. The CLUT is not used in this mode.

Each pixel occupies two contiguous bytes on a 2-byte boundary. When pixels are written into display memory using 16- or 32-bit write operations, they can be treated just as shown in Figure 9-7 on page 9-15. If partial pixels are being stored one byte at a time in a little-endian machine (such as an 80486), the byte containing the Blue bits is stored at the first address and the byte containing the Red bits is stored at the next higher address.

This DAC mode is no longer used by the Cirrus Logic BIOS. 32K Color modes now use 5:5:5 with 256-Color Mix mode. For compatibility, bit 15 of each pixel must be programmed to '0'.

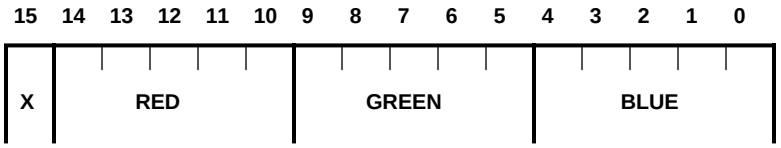


Figure 9-7. 5:5:5 Mode with 32K Colors

9.3.2.6 5:5:5 with 256-Color Mix Mode

This format allows the mixing of industry-standard 5:5:5 RGB mode pixels and palette DAC pixels. The interpretation of each pixel is based on bit 15 of the pixel itself.

If bit 15 contains '0', the remaining 15 bits are interpreted as 5:5:5 data. The CLUT is not used in this mode.

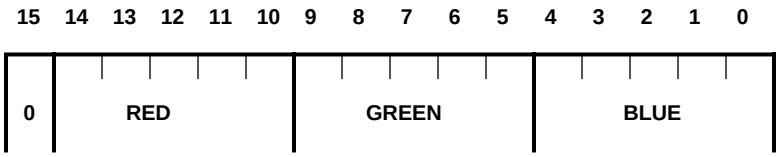


Figure 9-8. 5:5:5 with 256-Color Mix Mode: 5:5:5 Example

If bit 15 contains '1', bits 14:8 are ignored. Bits 7:0 select an entry in the palette whose contents are directed to the DACs.

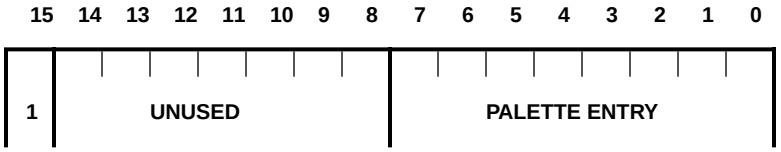


Figure 9-9. 5:5:5 with 256-Color Mix Mode: Palette Example

Each pixel occupies two contiguous bytes on a 2-byte boundary. When pixels are written into display memory using 16- or 32-bit write operations, they can be treated as shown in Figure 9-9. If partial pixels are being stored one byte at a time in a little-endian machine (such as an 80486), the byte containing the Blue bits (or the palette entry) are stored at the first address and the byte containing the Red bits (or bit 15) are stored at the next higher address.

9.3.2.7 XGA™ 5:6:5 Mode with 64K Colors.

This format supports the XGA 5:6:5 RGB mode with 65,536 colors. Each pixel is represented by 16 bits containing 5 bits of Red, 6 bits of Green, and 5 bits of Blue color information. The CLUT is not used in this mode.

Each pixel occupies two contiguous bytes on a 2-byte boundary. When pixels are written into display memory using 16- or 32-bit write operations, they can be treated as shown in Figure 9-10. If partial pixels are being stored one byte at a time in a little-endian machine (such as an 80486), the byte containing the blue bits is stored at the first address and the byte containing the red bits is stored at the next higher address.

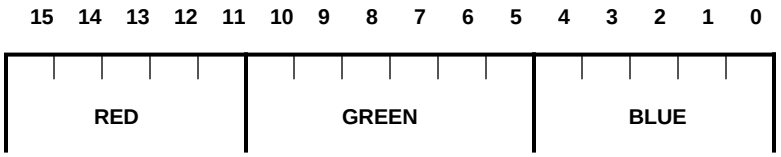


Figure 9-10. XGA™ 5:6:5 Mode with 64K Colors

9.3.2.8 16-bpp YUV 4:2:2 Mode

YUV is defined in the CCIR (International Radio Consultive Committee) recommendation CCIR601. The CL-GD5446 supports the YUV 4:2:2 format. Each two-pixel unit is stored as one dword containing luminance for each of the two pixels, and chrominance for alternate pixels. When YUV is displayed, the missing chrominance values are generated by interpolation.

YUV video can only be displayed within the video window.

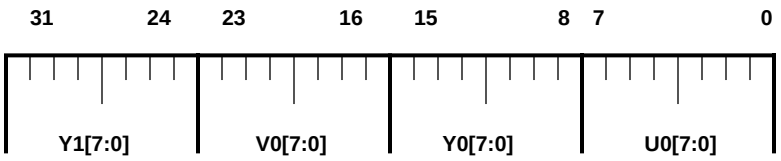


Figure 9-11. 16-bpp YUV 4:2:2 Mode

9.3.2.9 16-bpp YUV Planar Assist

YUV stored with planar assist contains the same information as YUV 4:2:2, but the bytes within each dword are ordered differently.

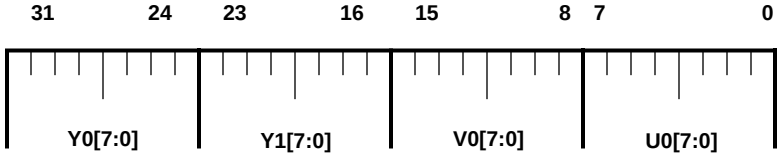


Figure 9-12. 16-bpp YUV Planar Assist

9.3.2.10 8:8:8 Mode with 16.8 Million Colors

This format supports the industry-standard 8:8:8 RGB mode with 16,777,216 colors. Each pixel occupies three adjacent bytes, one each of Red, Green, and Blue color information. The Blue value is stored in the lowest-addressed byte, the Green value is stored in the next higher-addressed byte, and the Red value is stored in the next higher-addressed byte. For the 640×480 24-bit-per-pixel mode, the Cirrus Logic BIOS sets the Offset (pitch) to 2048. A small amount of memory is unused, but the address calculations are simplified somewhat. The CLUT is not used in this mode.

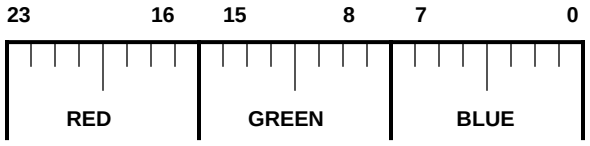


Figure 9-13. 8:8:8 Mode with 16.8 Million Colors

9.3.2.11 8:8:8:8 Mode with 16.8 Million Colors and Alpha

This mode supports the industry-standard 8:8:8:8 ARGB mode with 16,777,216 colors plus the Alpha Channel. Each pixel occupies four bytes, one each of Alpha, Red, Green, and Blue color information. The Blue value is stored in the lowest-addressed byte, followed by one byte each of Green, Red, and Alpha, in that order.

This mode is distinguished from the 3-bpp mode (see Section 9.3.2.10) by programming SR7[3:1]. If this field is programmed to '100', 32-bpp mode is selected. The Alpha byte is output on P[7:0], but has no internal use. The Cirrus Logic BIOS preferentially sets a 24-bpp mode, rather than a 32-bpp mode. This reduces the frame buffer bandwidth required to refresh the display, thus increasing performance.

If this mode must be programmed, the corresponding 24-bpp mode can be selected with an INT10 call, then modified by changing register SR7 and the offset.

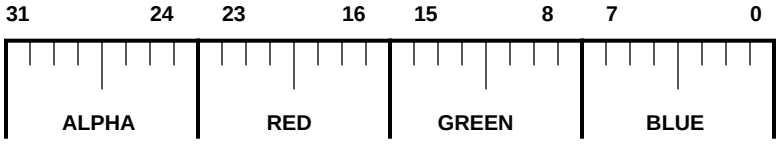


Figure 9-14. 8:8:8:8 Mode with 16.8 Million Colors and Alpha

9.4 BitBLT Engine

The CL-GD5446 incorporates a 64-bit BitBLT engine capable of color expansion, pattern fills, ROPs, and transparency support. The programming of this engine is discussed in the following sections. Basically, the BitBLT engine can move data around in the frame buffer autonomously while performing operations on it.

9.4.1 Definitions

This section establishes a common vocabulary for discussing the BitBLT engine. Figure 9-15 on page 9-19 illustrates these definitions, as well as the example in Section 9.4.2 on page 9-20. In this example, the displayable area (the screen) begins at the beginning of the frame buffer. The figure shows a BitBLT where the source is on-screen and the destination is off-screen. The destination pitch is set equal to the width so that the destination is as compact as possible.

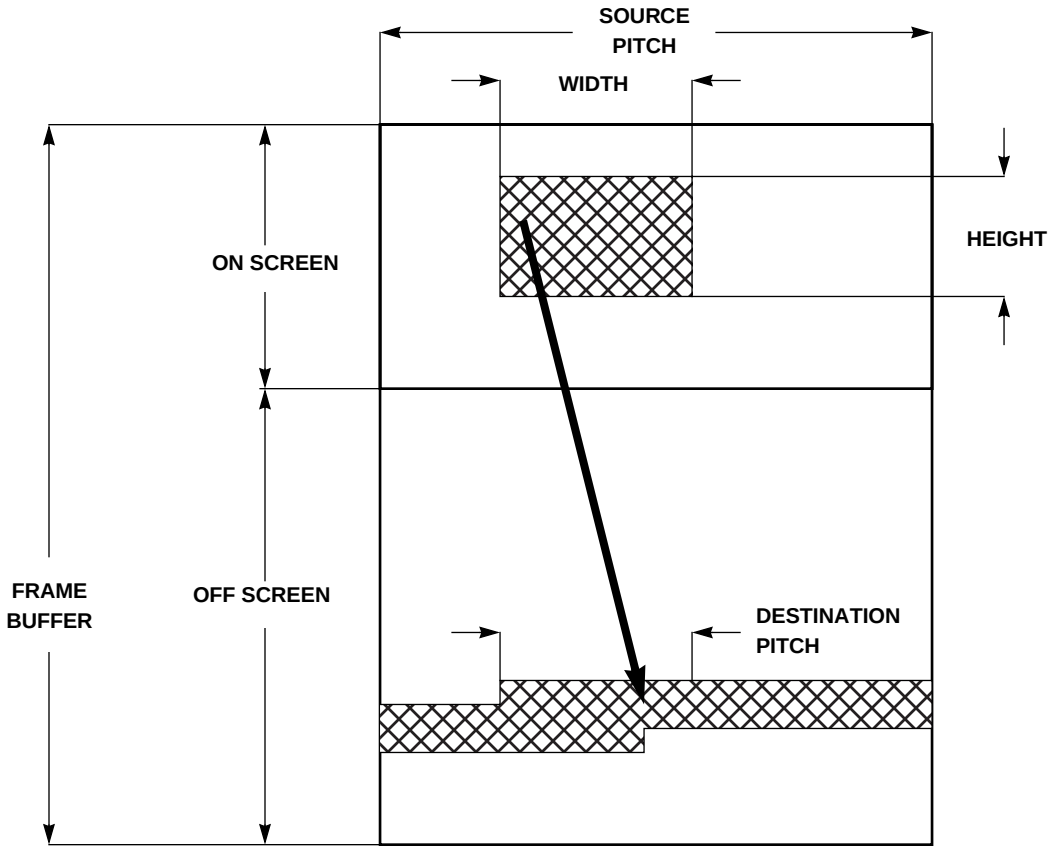


Figure 9-15. BitBLT Example

Source: The source for a BitBLT is the area the data is copied from. The source can be in the frame buffer or system memory. The source data can be a monochrome image expanded into color or it can be a full-color image. The source data can be a single 8×8 pixel pattern (in Windows, this is called a brush) replicated to fill a larger area. The source area is never written by the BitBLT engine, except in special cases (for example, where it overlaps the destination area in the frame buffer). The beginning of the source area, if it is in the frame buffer, is specified in the register triplet GR2C–GR2E.

Destination: The destination for a BitBLT is the area where the data are written. The destination area must always be in the frame buffer. The beginning of the destination is specified in register triplet GR28–GR2A.

Width: The width for a BitBLT is the number of bytes (not necessarily pixels) of destination that are processed before adding the pitch values to the address values. If the destination is actually or potentially on the screen (that is, if it is a rasterized area that might be displayed on the screen), the width is the number of bytes to be written into each scanline. If the source is a rasterized area, the width is the number of bytes per scanline of source. This example is shown in Figure 9-15.

If neither the source or the destination is a rasterized area, the width is the number of bytes of destination processed before the pitch values are added to the address values, and has no other

special meaning. Width is specified in register pair GR20–GR21. The number actually written into this register pair is one less than the actual desired width in bytes.

Height: The height for a BitBLT is the number of times the pitch values are added to the address values. If the destination or source is actually or potentially on the screen, the height is the number of scanlines in that area. If not, height is the number of times the pitch values are added to the address values, and has no other special meaning. Height is specified in register pair GR22–GR23. The number actually written into this register pair is one less than the actual desired height in scanlines.

Pitch: The destination and source pitch values are added to the respective addresses after each *width* bytes of destination are processed. The two pitches are specified separately. When an area is a rasterized image, the pitch is programmed to the distance between vertically adjacent pixels, usually the same as the display pitch (see Section 9.3.1 on page 9-11). Unless entire scanlines are being moved, the display pitch is greater than the BitBLT width. The source pitch is equal to the display pitch in Figure 9-15 on page 9-19.

When an image is in off-screen memory, it is often stored occupying a contiguous area (that is, so that the last pixel of scanline n is adjacent to the first pixel of scanline n + 1). This minimizes fragmentation. In this case, the pitch is set equal to the width (+1). In Figure 9-15 on page 9-19, the destination pitch is set equal to the width (+1).

When the source for a BitBLT is the system bus, the source pitch is a don't care. When the BitBLT uses color expansion or pattern copy, the source is considered a linear string of bytes, and the source pitch is a don't care. The destination pitch is specified in register pair GR24–GR25. The source pitch is specified in register pair GR26–GR27.

9.4.2 Basic BitBLT Example: Screen-to-Screen

Table 9-9 shows how the BitBLT registers are set up for a very basic operation. This copies a 64 pixel by 64 scanline area from an on-screen area to an area off the screen. It is assumed that the CL-GD5446 is programmed for Extended mode 74h (1024 × 768 at 64K colors). Figure 9-15 on page 9-19 illustrates this operation.

Table 9-9. Basic BitBLT Example

Register(s)	Value (hex)	Field	Notes
GR0, GR10, GR12, GR14	Don't care	Background Color	No color expansion
GR1, GR11, GR13, GR15	Don't care	Foreground Color	No color expansion
GR20–GR21	7Fh	Width	(64 × 2) –1 bytes
GR22–GR23	3Fh	Height	64 – 1 scanlines
GR24–GR25	80h	Destination Pitch	Same as width, destination area is compact
GR26–GR27	800h	Source Pitch	1024 × 2
GR28, GR29, GR2A	200000h	Destination Address	Beginning of third Mbyte
GR2C, GR2D, GR2E	5096h	Source Address	Scanline 10, pixel 75
GR2F	Don't care	Write Mask	No color expansion

Table 9-9. Basic BitBLT Example *(cont.)*

Register(s)	Value (hex)	Field	Notes
GR30	00h	Mode	Plain BitBLT
GR32	0Dh	ROP	SRCCOPY
GR33	0	Extended Mode	Unused
GR31	2	Start/Status	Start the operation

9.4.3 Controls

9.4.3.1 Start/RESET

Register GR31 contains bits to start a BitBLT, to determine the current status of the BitBLT engine, and to reset the BitBLT engine. Additionally, register GR31 contains the controls described in Section 9.4.3.2 and Section 9.4.4 on page 9-23.

GR31[1] starts a BitBLT operation. When this bit is set to '1', the operation starts with the next available memory cycle. When the operation is complete, the engine sets this bit to '0'. This provides the best method of synchronizing the host to the BitBLT engine; the host must wait until this bit is '0', it then programs all the BitBLT control registers for the next operation and sets this bit to '1'. When this bit is again '0', the operation is complete and the next operation can be programmed. See Section 9.4.3.2 for details on a more efficient approach for advanced applications.

GR31[0] can determine when the BitBLT engine is idle. If buffered BitBLTs are not being programmed, this bit is a read-only copy of GR31[1]. If buffered BitBLTs are being programmed, this bit indicates when all buffered BitBLTs are complete and the engine is idle.

GR31[2] can be programmed to '1' to reset the BitBLT engine. GR31[1:0] clears and the operation stops after the next write cycle. This can result in a partial BitBLT being completed, but does not otherwise corrupt the frame buffer.

9.4.3.2 AutoStart and Buffered Registers

The method of controlling the BitBLT engine described in this section is very straight forward, but does not provide maximum parallelism between the host and the engine. The use of buffered registers and the autostart capability makes it possible for the host and the BitBLT engine to function in parallel, keeping the BitBLT engine busy to the maximum possible extent.

When GR31[7] is '1', the AutoStart function is enabled. While BitBLT operation *n* is being executed, the parameters for operation n + 1 can be loaded into a set of buffered registers and operation n + 1 begins as soon as operation *n* completes. When GR31[7] is '0', the AutoStart function is disabled.

When AutoStart is enabled, GR31[4] indicates when the buffered registers are available. The host uses this bit to synchronize the loading of the registers. Whenever GR31[4] is '0', the parameter set for the next BitBLT can be loaded. This is illustrated in Figure 9-16 on page 9-22. Most of the time, the host is one parameter set ahead of the engine.

As soon as the engine starts BitBLT *n*, GR31[4] clears and the host loads the parameters for BitBLT n + 1. This happens almost immediately. For subsequent operations, the host may have to wait until the engine completes the previous operation.

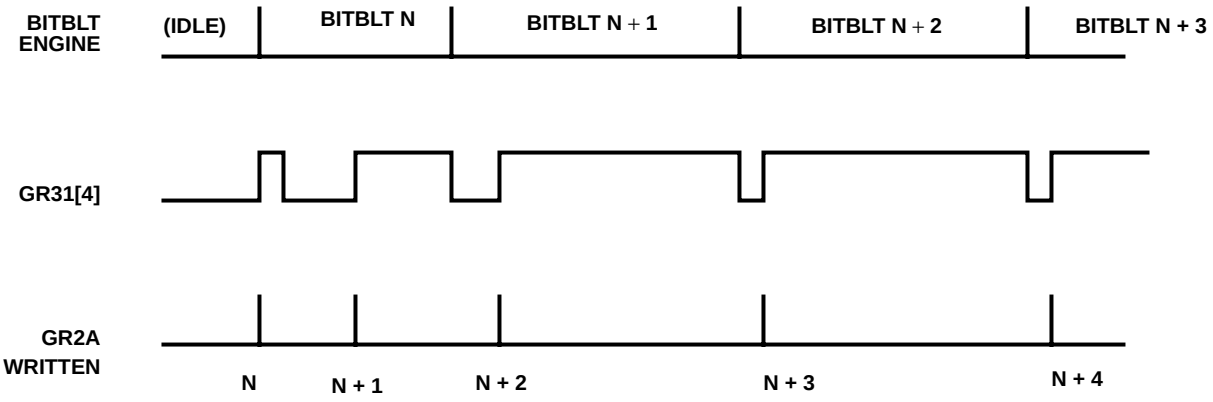


Figure 9-16. AutoStart Time Line

9.4.3.3 Using PCI Retry to Avoid Polling

The host can avoid status polling (reading GR31[4]) by using the PCI retry capability. An attempt to write the BitBLT registers when GR31[4] is '1' results in the CL-GD5446 executing a Disconnect C (Retry). It forces STOP# active without TRDY#. This is an indication to the host that the data was not transferred and that it must attempt the transfer again. To use this method, the host must use memory-mapped I/O with AutoStart enabled, write to a memory-mapped offset lower than 10h first, and write the Destination Address (offset 10h) last. These restrictions do not apply if the application uses status polling.

9.4.3.4 Memory-Mapped I/O

When SR17[2] is '1', the BitBLT registers are accessible for writes in memory space, as well as I/O space, allowing dword access and burst transfers. Memory-mapped I/O must be enabled to use AutoStart or the PCI Retry method to avoid status polling.

If SR17[6] is '0', the registers occupy a 256-byte block beginning at 000B8000h. If SR17[6] is '1' and linear addressing is enabled, the registers occupy the upper 256 bytes of the occupied portion of each linear aperture. As shown in Figure 9-17 on page 9-23, if the actual memory is less than 4 Mbytes, the Memory-mapped I/O is at the top of the occupied area, not necessarily the top of the 4-Mbyte aperture.

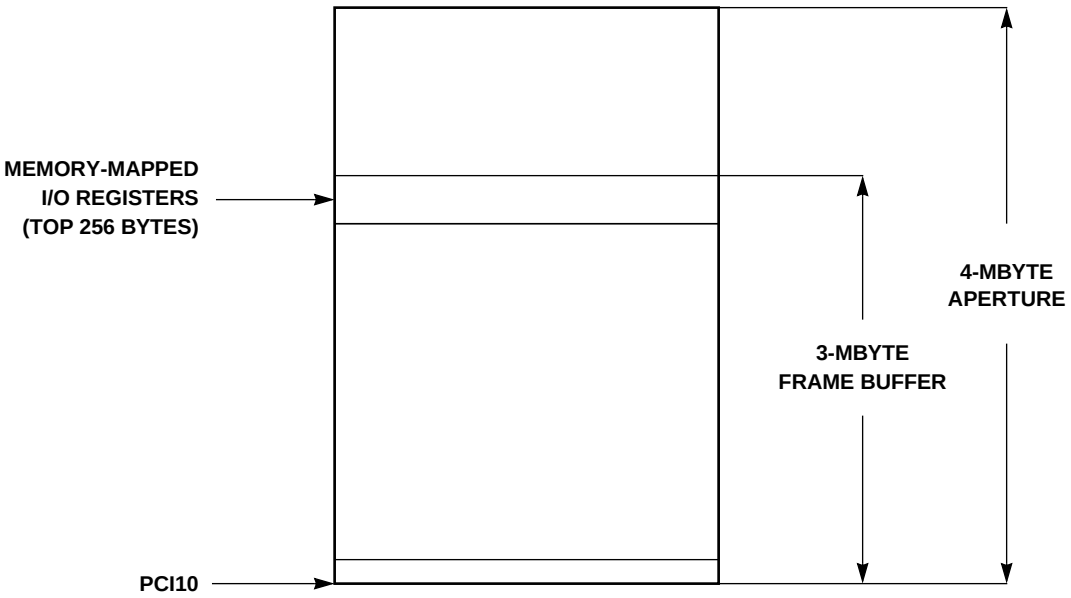


Figure 9-17. Memory-Mapped I/O in Linear Aperture

Table 9-4 on page 9-6 lists registers accessible with Memory-mapped I/O. The addresses (offsets within the block) are carefully allocated so that registers with a similar function are groups into dwords. The bold lines in the table separate dword blocks.

9.4.4 Host Access to Display Memory During BitBLTs

It is sometimes necessary to allow the CPU to access the frame buffer independently of the BitBLT. For example, this occurs during luminance-only captures when the cursor needs to be changed in response to a mouse interrupt, or for Bus Master writes (such as, MPEG data).

9.4.4.1 Memory Read During BitBLT

The CL-GD5446 responds to a memory read by temporarily preempting any BitBLT in progress (either a system-to-screen BitBLT or a screen-to-screen BitBLT). Once the data has been transferred to the host, the BitBLT resumes. The CL-GD5446 manages this invisibly; the application does not have to perform any special tasks.

9.4.4.2 Memory Write During Screen-to-Screen BitBLTs

The CL-GD5446 responds to a memory write while executing a screen-to-screen BitBLT by storing the address and data into the system write buffer. When the write buffer is at least half full, it preempts the BitBLT to execute writes to the frame buffer. The CL-GD5446 manages this invisibly; the application does not have to perform any special tasks.

9.4.4.3 Memory Write During System-to-Screen BitBLTs

If color-expand system-to-screen BitBLTs are to be mixed with normal memory writes, the CL-GD5446 must be able to distinguish between BitBLT data and other data. The host can enable a second address aperture by programming GR31[6] to '1'. When this aperture is enabled, writes to the linear frame buffer at the address specified in PCI10 are considered normal memory writes. Writes to the aperture at 000BC000h–000BFFFFh are considered writes to the BitBLT engine. This method allows autonomous processes (such as, an MPEG decoder and a GUI driver) to share the host interface without being aware of each other.

The second aperture can only be used with color-expanded BitBLTs. The host software must wait for the BitBLT engine to be idle (GR31[0]) before it changes GR31[6]. Since the BitBLT engine does not have access to the write buffer, performance suffers slightly.

The other method of sharing the frame buffer for writes requires that the processes know about each other. To temporarily interrupt any system-to-screen BitBLT to write to directly to the frame buffer, the host can execute the following sequence:

1) Write to memory-mapped I/O location 3Fh; the data is don't care. This has the effect of waiting for any pending BitBLT to become active.

2) Program GR31[5] to '1'. When this bit is set, the BitBLT engine ignores memory writes. The actual address on the bus determines where the data is to be written.

3) Program GR31[5] '0' when writes to the frame buffer are complete. The BitBLT engine again accepts data written to the frame buffer.

This method of interrupting system-to-screen BitBLTs can be used regardless of whether color expansion is enabled.

9.4.5 Data Source

The source data for a BitBLT can be the frame buffer or the system bus. These two cases are referred to as screen-to-screen and system-to-screen BitBLTs, respectively. When the source is the system bus, monochrome data can be expanded to color or the color data itself can be transferred across the bus. When pattern copy is being used, the source cannot be the system bus.

If GR30[2] is '1', the source is the system bus. The host must transfer the data (monochrome or color) to the CL-GD5446. The CL-GD5446 is never a bus master. The host must always transfer data with dword cycles.

System-to-Screen BitBLTs

For a system-to-screen BitBLT without color expansion, the CL-GD5446 discards information to the end of a dword at the end of each scanline (that is, each *width* of bytes). The amount of data discarded can be 0–3 bytes, depending on the *width* value.

For a system-to-screen BitBLT with color expansion, GR33[0] defines how the data are discarded at the end of each scanline. If GR33[0] is '0', byte granularity is used. When the end of each scanline is reached, the remaining 0–7 bits of the current byte are discarded. Any bytes remaining in the current dword are used at the beginning of the next scanline. The source data must be byte-packed (that is, scanline n + 1 must begin on the byte immediately following the end of scanline n).

If GR33[0] is '1', dword granularity is used. When the end of each scanline is reached, the remaining 0–7 bits of the current byte and the remaining 0–3 bytes of the current dword are discarded. The source can be transferred using dwords, without having to be byte-packed.

To allow unaligned source fields to be transferred without the overhead of unaligned bus cycles, program GR2F[6:5] to skip bytes of the source field for system-to-screen BitBLTs. The field selects the initial byte of the first dword to be used for each scanline. If this field is programmed to a non-zero value for color-expanded system-to-screen BitBLTs, GR33[0] *must* be programmed to '1'. When GR2F[6:5] is used, data is not written at the destination start address, but is instead clipped. This means the destination start address must be adjusted to point to the *left* of where the data is actually to begin. In addition, the width value must be increased to compensate for the source data skipped.

9.4.6 ROPs (Raster Operations)

The CL-GD5446 supports all 16 possible operations on two variables. The operation to be performed is programmed into register GR32. The ROPs are listed in Table 9-10. Note that the value actually programmed into register GR32 is independent of whether a source or pattern is to be used; this distinction is made in GR30[6].

Do not use ROPs that do not use the source (such as, DSTINVERT) when color expansion or pattern copy is enabled.

Table 9-10. Raster Operations

Source Operation	Microsoft® Name/ROP	Pattern Operation	Microsoft® Name/ROP	Regsiter GR32 (hex)
0	BLACKNESS 00000042	0	BLACKNESS 00000042	00
DSon	NOTSRCERASE 001100A6	DPon	– 000500A9	90
DSna	– 00220326	DPna	– 000A0329	50
Sn	NOTSRCCOPY 00330008	Pn	– 000F0001	D0
SDna	SRCERASE 00440328	PDna	– 00500325	09
Dn	DSTINVERT 00550009	Dn	DSTINVERT 00550009	0B
DSx	SRCINVERT 00660046	DPx	PATINVERT 005A0049	59
DSan	– 007700E6	DPan	– 005F00E9	DA
DSa	SRCAND 008800C6	DPa	– 00A000C9	05
DSxn	– 00990066	PDxn	– 00A50065	95
D	– 00AA0029	D	– 00AA0029	06
DSno	MERGEPAINT 00BB0226	DPno	– 00AF0229	D6

Table 9-10. Raster Operations *(cont.)*

Source Operation	Microsoft® Name/ROP	Pattern Operation	Microsoft® Name/ROP	Regsiter GR32 (hex)
S	SRCCOPY 00CC0020	P	PATCOPY 00F00021	0D
SDno	– 00DD0228	PDno	– 00F50225	AD
DSo	SRCPAINT 00EE0086	DPo	– 00FA0089	6D
1	WHITENESS 00FF0062	1	WHITENESS 00FF0062	0E

There are exactly 16 ways that two operands can be logically combined; these are enumerated in Table 9-11. This table is provided for readers interested in the underlying logical functions. The first four columns show the logical result of the four possible input combinations for each of the 16 cases.

Table 9-11. Sixteen Logical Operations

S = 1, D = 1	S = 1, D = 0	S = 0, D = 1	S = 0, D = 0	Description	Register GR32 (hex)
0	0	0	0	Zero	00
0	0	0	1	NOT Source AND NOT Destination	90
0	0	1	0	NOT Source AND Destination	50
0	0	1	1	NOT Source	D0
0	1	0	0	Source AND NOT Destination	09
0	1	0	1	NOT Destination	0B
0	1	1	0	Source NOT EQUAL to Destination (XOR)	59
0	1	1	1	NOT Source OR NOT Destination	DA
1	0	0	0	Source AND Destination	05
1	0	0	1	Source EQUAL to Destination (XNOR)	95
1	0	1	0	Destination	06
1	0	1	1	NOT Source OR Destination	D6
1	1	0	0	Source	0D
1	1	0	1	Source OR NOT Destination	AD
1	1	1	0	Source OR Destination	6D
1	1	1	1	One	0E

9.4.7 Color Expansion

Color expansion is enabled by setting GR30[7] to '1'. The source data is not the actual data from the source or pattern, but is a monochrome image. Each bit of the source is replaced with an entire *pixel* that can have either of two colors. If color expansion with transparency is enabled, each bit written is replaced with a pixel of the a single color or the corresponding pixel is not written at all. Color expansion cannot be used with ROPs that do not have a source such as, Whiteness, Blackness, Destination, and Destination Invert.

The monochrome image can come from display memory or can be transferred from system memory. Since each source bit represents an entire destination pixel, substantial performance benefits can be obtained, especially if the image is being expanded to 16 or more bits per pixel.

The number of bytes each bit of monochrome data is expanded into is programmed in GR31[5:4]. Table 9-12 and Table 9-12 enumerate the widths and specify the register each byte of color information comes from. Note that when 24-bpp color expansion is programmed, transparency *must* be enabled (only one color is written).

Table 9-12. Color Expansion: Foreground ('1' in source)

GR30[5:4]	Width	GR1	GR11	GR13	GR15
00	8-bits	Color	–	–	–
01	16-bits	Low byte	High byte	–	–
10	24-bits	Blue	Green	Red	–
11	32-bits	Blue	Green	Red	Alpha

Table 9-13. Color Expansion: Background ('0' in source)

GR30[5:4]	Width	GR0	GR10	GR12	GR14
00	8-bits	Color	–	–	–
01	16-bits	Low byte	High byte	–	–
10	24-bits	24-bpp color expansion *must* use transparency			
11	32-bits	Blue	Green	Red	Alpha

The source data polarity can be reversed by setting GR33[1] to '1'. When this is done, '1' in the source invokes the background color, and '0' in the source invokes the foreground color.

If color expansion is used with transparency (GR30[7] = GR30[3] = 1) background pixels are not written. This allows text to be applied onto an arbitrary background. Transparency with color expansion can be used for any color depth. Transparency is also supported without color expansion for 8- and 16-bpp graphics modes. See Section 9.4.9 on page 9-28.

9.4.8 Pattern Fills

The CL-GD5446 supports an 8 × 8 pattern fill. When GR30[6] is '1', the source is an array of 8 pixels by 8 scanlines. The source for pattern fill must be stored in the display memory; sys-

tem-to-screen pattern fills are not permitted. The array is repeatedly copied to the destination area with or without color expansion. For any scanline of destination, the same 8 pixels of source data are repeatedly used. Pattern fill cannot be used with ROPs that do not have a source such as, Whiteness, Blackness, Destination, and Destination Invert. The number of bytes in the source pattern is a function of the pixel format, as shown in Table 9-14.

Table 9-14. Source for Pattern Fills

Pixel Format	Pattern	Starting Address Boundary
Color Expansion	8 bytes of monochrome data for 64 pixels.	8 bytes
8-bpp	64 bytes of color data for 64 pixels.	64 bytes
16-bpp	128 bytes of color data for 64 pixels	128 bytes
24-bpp	24 bytes of color data, plus 8 bytes of padding for each scanline, repeated 8 times.	256 bytes
32-bpp	256 bytes of color/alpha data for 64 pixels.	256 bytes

Offsets into the pattern can be programmed as follows:

The patterns must be stored beginning on a boundary equal to the size of the pattern itself.

The vertical offset into the pattern is then programmed into the three low-order bits of the Source Address (register GR2C).

The horizontal offset must be specified by programming a clip value into GR2F. GR2F[2:0] can be programmed to ‘000h’ through ‘111h’ to disable the writing of the first n pixels of each destination scanline. This field is programmed in terms of pixels for all pixel formats, except Packed-24. For Packed-24 mode, this field is expanded to 5 bits (GR2F[4:0]) and programmed in terms of bytes.

9.4.9 Transparency

The CL-GD5446 supports transparency both for color-expansion and for color source images. Transparency is enabled by setting GR30[3] to ‘1’.

When color expansion is being used, pixels that would otherwise be written with the background color are not written. This can be used with any pixel depth and *must* be used for 24-bpp.

When color expansion is not being used, transparency can be used for 8- and 16-bpp formats. If GR30[3] is ‘1’, the value of each pixel is compared to registers GR34–GR35 before being written. If the pixel matches in all 8 or 16 bits, it is not written.

For 8-bpp formats, registers GR34 and GR35 must be identically programmed. For 16-bpp formats, register GR34 is the low byte and register GR35 is the high byte. Transparency without color expansion can only be used with a ROP of source copy. No other ROP can be used.

9.4.10 BitBLT Direction

If the source and destination areas overlap in display memory, the application program must ensure that the move progresses so that the source area is not overwritten prior to being used.

Consider Figure 9-18. If the operation began with the upper-left corner of the source and destination, the contents of the overlapped area would be overwritten before being used.

If GR30[0] is '1', the direction that the operation progresses is reversed (that is, the bytes are processed right-to-left and bottom-to-top). In this case (the overlay as shown in Figure 9-18), bytes are prevented from being overwritten before being copied. Note that the start addresses in this case are the highest in the areas, not the lowest. BitBLTs using color expansion, pattern fill, system-to-screen, or transparency cannot be programmed for reverse direction.

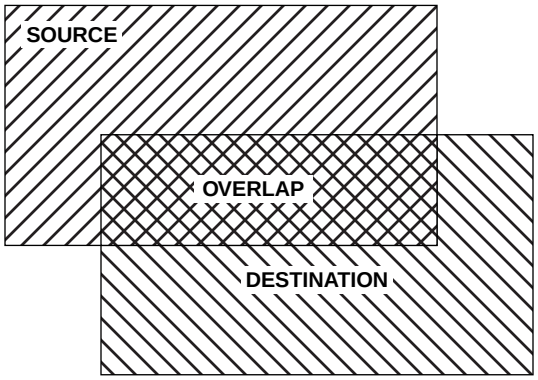


Figure 9-18. Overlapping BitBLT

9.4.11 Miscellaneous BitBLT Topics

9.4.11.1 Solid Color Fill

The CL-GD5446 can be programmed to perform a solid color fill by programming GR33[2] to '1'. GR30[7] and GR30[6] must both be '1' (enable color expansion and pattern copy). GR30[3] and GR30[2] must both be '0' (no transparency or system-to-screen). The contents of the foreground register(s) are written to the destination rectangle. Any expansion width can be used. This functions precisely as a color expanded pattern copy with the pattern of all ones, except the pattern reads are skipped.

9.4.11.2 Patterned Polygon Fills

The CL-GD5446 can support patterned polygon fills with color expansion. The software decomposes the polygon to be filled into a series of single scanlines, each of which is filled with a single BitBLT. The first operation reads all 8 bytes of the monochrome pattern. Then, as long as there are no writes to the Source Start Address registers or the BLT Mode register, subsequent BitBLT operations use the previously loaded source data (skipping the read cycle). In addition, the Y offset (initially set to the three low-order bits of the source start address in GR2C[2:0]) increments modulo eight at the end of each operation. The result is each scanline starting one byte (that is, one scanline) further into the pattern than the previous (immediately above) scanline. The polygon can be filled with a series of single scanline fills that change only the destination start address, left-edge clipping, and width. The operation should proceed from top-to-bottom.

9.4.11.3 Protecting Bytes

For 32-bpp graphics formats, it may be desirable to avoid modifying the alpha bytes of each pixel during BitBLTs or ordinary memory writes. An easy way to do this is to set GRB[2] to '1', this makes register SR2 is a byte-wise write protect for the entire 8-byte data path. That is, it protects the bytes of each qword destination write. This is also useful for replicating the chrominance value for the video aperture. See Section 9.14.1.

9.4.11.4 Frame Switching

CR5E[2:0] can be programmed so that the active graphics buffer (the displayed graphics buffer) is automatically switched at the conclusion of a BitBLT. The two buffers begin at frame buffer offset 0 and at the Screen Start A registers (CRC, CRD, and so on). GR33[4] must be '1' for frame buffer switching to occur.

9.4.11.5 Active Display Line Readback

Register GR16 and GR17[1:0] contain the active display line. GR17[1:0] are the high-order bits. These registers can be read at any time to determine what scanline is currently displayed. See the programming note in the description of register GR16 in Section 8.25 on page 8-35.

9.4.12 Text Expansion BitBLT Example

When using color expansion, a text string is copied from system memory. The monochrome image of the string is arranged in system memory by scanline. The destination area is 150 pixels by 25 scanlines (8-bpp). The destination pitch is 1024 bytes. The registers must be loaded as indicated in Table 9-15. If the background pixels are not to be written (transparency), set GR30[3] to '1'.

Table 9-15. Text Expansion BitBLT Example

Register(s)	Value (hex)	Field	Notes
GR0, GR10, GR12, GR14	FFFFFFFFh	Background Color	As desired (white).
GR1, GR11, GR13, GR15	00000000h	Foreground Color	As desired (black).
GR20–GR21	95h	Width	150 – 1 bytes per scanline.
GR22–GR23	18h	Height	25 – 1 scanlines.
GR24–GR25	400h	Destination Pitch	
GR26–GR27	n/a	Source Pitch	System memory.
GR28, GR29, GR2A	X	Destination Address	Wherever.
GR2C, GR2D, GR2E	X	Source Address	System memory.
GR2F	0	Write Mask	No clipping.
GR30	84h	Mode	Color expansion, 8-bpp, system memory source.
GR32	0Dh	Raster OP	SRCCOPY
GR33	0	Extended Mode	Discard partial bytes.
GR31	2	Start	Set bit 1.

After the registers are loaded the source bitmap must be transferred. The first dword write transfers the image for pixels 0–31; the second write for pixels 32–63; the third write for pixels 64–95; the fourth for pixels 96–127.

The fifth dword write transfer is more interesting. The data for this transfer is shown in Figure 9-19. Higher-numbered bits within each byte of source controls pixels further to the left on the screen.

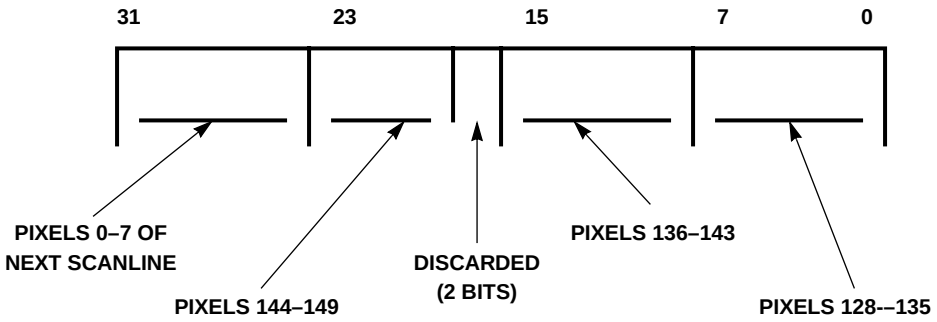


Figure 9-19. Text Expansion Example

9.5 Video Window

The CL-GD5446 supports a single rectangular video window. An example video window is shown in Figure 9-20 on page 9-32. The information displayed in the video window is taken from an area in the frame buffer other than the normal graphics data and is typically displayed using a different color space conversion. The graphics data is written into the frame buffer by the host (perhaps using the BitBLT engine) in the normal manner. The video data can be written into the frame buffer by a video decoder process running in the host or written by the video capture port (V-Port).

The term ‘video’ is used for pixels, modes, and so on, in the window. The term ‘graphics’ is used for pixels, modes, and so on, not in the window.

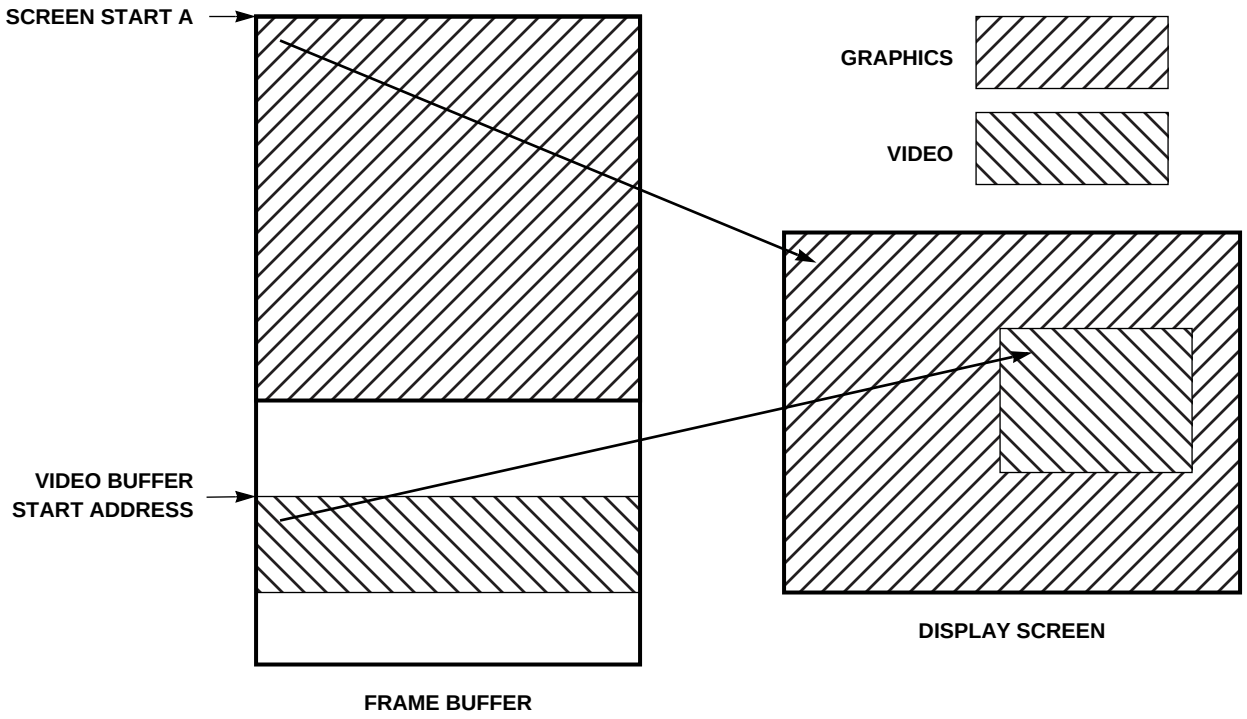


Figure 9-20. Video Window

9.5.1 Positioning the Hardware Window on the Screen

The size and position of the hardware window is specified in a number of registers, as shown in Figure 9-21 and Table 9-16. Figure 9-21 illustrates the parameters, while Table 9-16 gives actual register numbers. Refer to Chapter 6, “Video Capture and Playback Registers”, for detailed register descriptions.

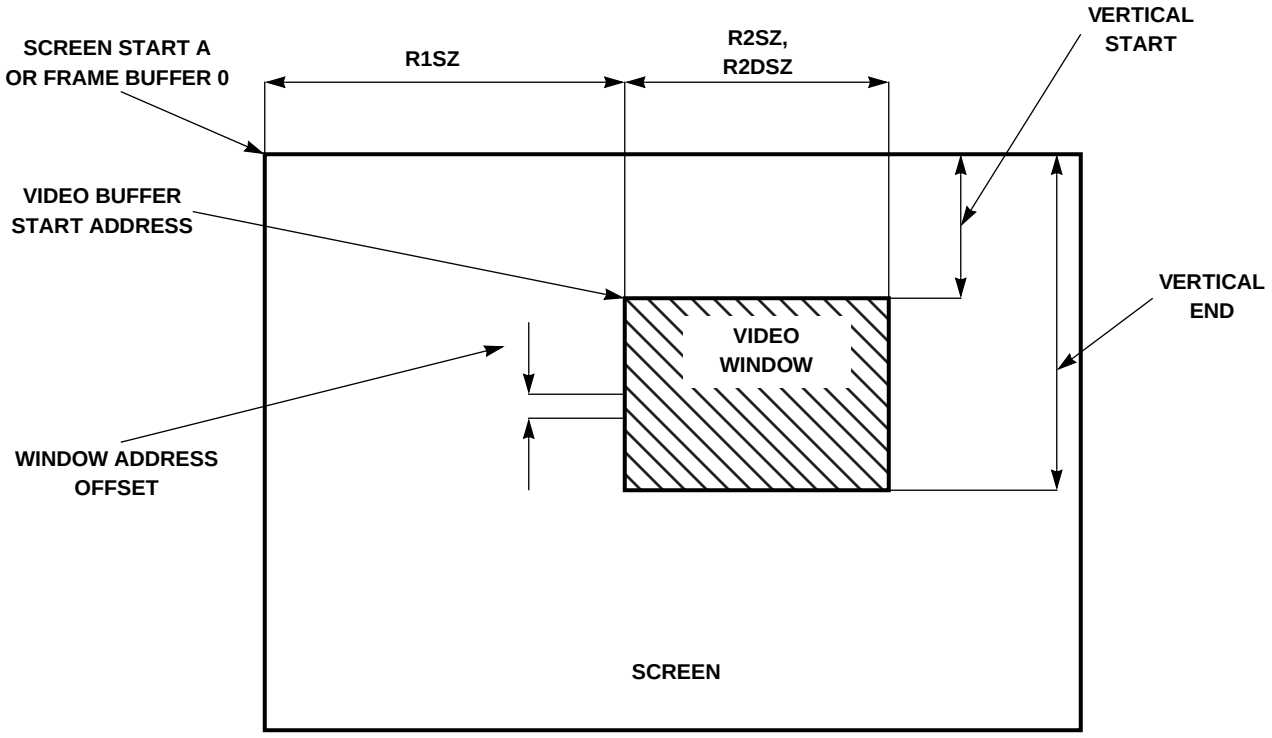


Figure 9-21. Positioning of the Video Window

Table 9-16. Size and Position of the Video Window

Parameter	Mnemonic	Register(s)	Function
Region 1 Size	R1SZ	CR33/CR36	Start of video window in dwords of graphics data plus byte adjustment for pixel granularity.
Region 1 Size Adjust	R1Adjust	CR5D[1:0]	
Region 2 Width	R2SZ	CR34/CR36	Width of video window in dwords of graphics data skipped, plus byte adjustment for pixel resolution.
Region 2 Size Adjust	R2 Adjust	CR5D[5:4]	
Region 2 Source Data Size	R2DSZ	CR35/CR36	Dwords of video data to fetch for window.
Vertical Start	WVS	CR37/CR39	First scanline of video window.
Vertical End	WVE	CR38/39	Last scanline of video window.
Video Buffer 1 Start	WSA	CR3ACR3B/CR3C	Address of top-left pixel in video window.
Window Address Offset	WAO	CR3C/CR3D	Video window pitch.

9.5.2 Window Position

Vertical

The vertical start and end of the video window are programmed in absolute scanlines. Vertical Start is programmed to the first scanline to be displayed in the window. A '0' value places the window at the top of the screen. Vertical End is programmed to the last scanline to be displayed in the window. Each of these values is 10 bits (an 8-bit register plus two overflow bits). Vertical End must always be greater than Vertical Start (that is, the window must be vertically contiguous).

Horizontal

The horizontal position and size of the video window are programmed in dwords of graphics data. Figure 9-22 shows a single scanline within the video window. Each scanline consists of up to three regions.

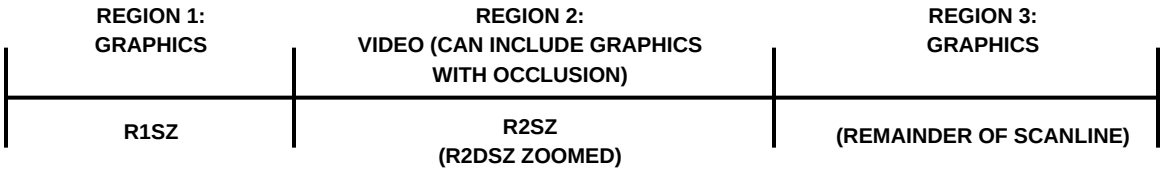


Figure 9-22. Horizontal Regions in the Video Window

Horizontal Region 1

Region 1 begins with the first pixel in the scanline and consists of graphics pixels to the left of (before) the window. A number of dwords of graphical data equal to *R1SZ* (plus a zero-to-three byte count in *R1Adjust*) are fetched and displayed. The number of pixels in Region 1 can be calculated with Equation 9-2.

$$Region1Size = \left(\frac{32}{GraphicsBPP} \bullet R1SZ\right) + \left(\frac{R1Adjust \bullet 8}{GraphicsBPP}\right) \qquad \textbf{Equation 9-2}$$

For software compatibility with the CL-GD5440, hardware is programmed in dwords rather than pixels. To allow the window to begin on any pixel boundary, R1 Adjust can move the boundary within a dword (except for 24-bpp graphics data)

Region 1 can be programmed for a size of zero. In this case, the video window begins at the left edge of the screen.

Horizontal Region 2

Region 2 is the window itself. It begins with the first pixel after the end of Region 1. The size of this region is also defined in dwords of graphics data. The width of the video window (in pixels) can be calculated with Equation 9-3.

$$Region2Size = \left(\frac{32}{GraphicsBPP} \bullet R2SZ\right) + \left(\frac{R2Adjust \bullet 8}{GraphicsBPP}\right)$$ **Equation 9-3**

The same considerations as those for Region 1 apply. Since the hardware is first counting dwords rather than pixels, the window size can be adjusted to any pixel using *R2Adjust*.

If occlusion support is not enabled (that is, if the window is a rectangular region containing only video data), the graphics data 'under' the window is not actually fetched from the frame buffer. This lowers the bandwidth required and can offer better performance.

Video Source Size

For each scanline contained in the window, R2SDSZ dwords of video data are fetched and displayed during Region 2 (the video window) with any programmed zoom factor. This value must be programmed so that sufficient video data is fetched to fill (after zooming) the window. Use Equation 9-4 to calculate this value. The first element under the line in the equation considers each dword of source data contains two or four pixels. The *HorizontalZoomFactor* is included to account for the fact that pixels are 'generated' for zooming. If the evaluation of this equation does not yield an integer, it must be rounded up. Programming *R2SDSZ* to a value too small results in 'trash' at the right side of the window. Programming *R2SDSZ* to a value too large results in trash at the start of Region 3. Programming *R2SDSZ* just right obtains the best results. *R2SDSize* is a 10-bit field, programmed into a byte register with two overflow bits.

$$Region2SDSize = \frac{Region2SizeinPixels}{\frac{32}{VideoBPP} \bullet HorizontalZoomFactor}$$ **Equation 9-4**

Horizontal Region 3

Region 3 is the area to the right of (after) the window. Graphic data is displayed there, beginning with the first pixel after those skipped as specified in R2SZ. If Regions 1 and 2 are programmed so that the window extends to the right edge of the screen, there is no Region 3.

9.5.3 Enabling the Window

CR3E[0] is the master enable bit for the video window. When this bit is '1', the display of data in the video window is enabled. This bit takes effect on the next leading edge of the display (VGA) VSYNC.

9.5.4 Video Source Address and Offset

Video can be displayed from anywhere in the frame buffer, although typically it comes from a region separate from the graphics. The CL-GD5446 provides support for double-buffering the video data. Table 9-17 shows the source of the address bits for the beginning of each of the two buffers.

Table 9-17. Video Source Beginning Address Bits

Address Bits (Correspond to a 4-Mbyte Linear Address)	21:18	17:10	9:2	1:0
Video Buffer 1 (default display buffer)	CR3C[3:0]	CR3B[7:0]	CR3A[7:0]	CR5D[3:2]
Video Buffer 2 (default capture buffer)	CR58[3:0]	CR5A[7:0]	CR59[7:0]	CR5D[3:2]

At the end of each scanline contained in the window, the video buffer offset is added to the video window address to obtain the beginning address of the video for the next scanline. This is a 12-bit byte offset; the low-order three bits are forced to '0'.

Table 9-18. Video Buffer Line Address Offset

Address Bits (Correspond to a 4-Mbyte Linear Address)	11	10:3	2:0
Register:	CR3C[5]	CR3D[7:0]	0

9.5.5 Video Display Double Buffering

The CL-GD5446 supports double buffering of the video display. The video source beginning address is contained in one of two register sets. (referred to as Video Buffer 1 and Video Buffer 2 as indicated in Table 9-17). CR5E[4:3] controls the selection of video buffers, as shown in Table 9-19.

Table 9-19. Video Buffer Switching Control: CR5E[4:3]

CR5E[5:4]	Function	Note
00	Video Buffer 1 is display; Video buffer 2 is capture.	These cases are for software capture, where the process can explicitly control how the buffers are being used.
01	Video Buffer 1 is capture; Video buffer 2 is display.	
10	Auto-Switch with each VREF (or alternate VREF for interlace).	See Section 9.6.4 on page 9-43.
11	Switch at completion of BitBLT with GR33[4] = 1	

The switches described in Table 9-19 are considered triggers. The actual buffer used for any given VGA refresh period is selected just as the screen refresh reaches the top of the video window.

9.5.6 Video Display Formats

The pixel format of the video displayed in the window can be YUV16, 16-bpp RGB, 8-bpp LUT, or AccuPak, as specified in CR3E[3:1].

Table 9-20. Video Display Format

CR3E[3:1]	Format	Note
000	YUV 4:2:2	CCIR YUYV, or selects YUV planar assist if CR3F[4] = 1.
001	AccuPak™	Compressed YUV 4:1:1 format.
010	8-bit LUT	Graphics format must be RGB-16; the 256 × 18 VGA palette is used.
011	Reserved	
100	RGB 5-5-5	Also selects PCI bus writes; RGB decimate if CR3F[4] = 1 using the video aperture.
101	RGB 5-6-5	
110	Reserved	
111		

There are some limitations, depending on the graphics mode and the DRAM size (effectively determining the frame buffer bandwidth). These limitations are noted in Table 9-21 and the subsequent notes.

Table 9-21. Video Window Display Features and Limitations

Graphics Format	Video Format	DRAM Size	Y-Interpolation?	Occlusion Support
8-bit LUT, < 90 MHz	YUV16, RGB16	1 Mbyte	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: 2× MIN horizontal zoom. 1024 × 768: 2× MIN horizontal zoom AND 80 MHz MCLK.
		2, 3, 4 Mbytes	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: Any horizontal zoom factor. 1024 × 768: Any horizontal zoom factor.
	AccuPak™	1 Mbyte	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: Any horizontal zoom factor. 1024 × 768: 2× MIN horizontal zoom OR 80 MHz MCLK.
		2, 3, 4 Mbytes	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: Any horizontal zoom factor. 1024 × 768: Any horizontal zoom factor.
8-bit LUT, > 90 MHz VCLK ×2, (1280 × 1024 only)	YUV16, RGB16	2, 3, 4 Mbytes	No	1280 × 1024: Color-key granularity of two graphics pixels. MCLK: 66 MHz for 60 Hz refresh; 80 MHz for 70 Hz refresh. Window has 2-pixel position and width granularity. The zoomed video data is clocked at VCLK rate and pixel replicated by clock doubler; this implies the programmed horizontal zoom factor is multiplied by two.

Table 9-21. Video Window Display Features and Limitations *(cont.)*

Graphics Format	Video Format	DRAM Size	Y-Interpolation?	Occlusion Support
RGB16	YUV16, RGB16	1 Mbyte	Yes	640 × 480: 2× MIN horizontal zoom OR 80 MHz MCLK. 800 × 600: 2× MIN horizontal zoom AND 80 MHz MCLK; AccuPak or 8-bpp LUT recommended.
		2, 3, 4 Mbytes	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: Any horizontal zoom factor. 1024 × 768: 2× MIN horizontal zoom OR 80 MHz MCLK.
	AccuPak, 8-bit LUT	1 Mbyte	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: 2× MIN horizontal zoom OR 80 MHz MCLK
		2, 3, 4 Mbytes	Yes	640 × 480: Any horizontal zoom factor. 800 × 600: Any horizontal zoom factor. 1024 × 768: Any horizontal zoom factor.
24-bit RGB (Packed-24)	YUV16, RGB16	1 Mbyte, 2 Mbytes	No	Not supported. In addition, the window position and width have a granularity of four graphics pixels.
32-bit ARGB	(n/a)	(n/a)	(n/a)	Video window not supported.

NOTES:

1) Y-interpolation and occlusion are mutually exclusive. They cannot be used together.
2) Y-interpolation is disallowed by the software unless Horizontal Zoom is 2× or greater.
3) When 2× horizontal zoom must be used, it can be compensated for by using video capture decimation or system bus aperture auto-decimate (RGB 16 only).
4) A 1-Mbyte frame buffer based on 128K × 16 DRAMs uses all 64 MD bits, providing the same bandwidth as a 2-Mbyte frame buffer.

9.5.7 Zooming the Video in the Hardware Window

The source video can be zoomed (expanded) as it is displayed. The zoom factors for X (horizontal) and Y (vertical) are specified independently. X zoom is specified in register CR31; Y zoom is specified in register CR32. The zoom code programmed into each register is interpreted according to Equation 9-5.

$$ZoomFactor = \frac{256}{ZoomCode}$$ **Equation 9-5**

A zoom code of '0' produces a 1:1 zoom factor (zooming is turned off in the corresponding dimension). Practical values for the Zoom Code are 64–255 (producing zoom factors of 4:1 to slightly more than 1:1).

Zooming in the X (horizontal) dimension is always done with interpolation. Pixels inserted between source pixels are calculated by taking a weighted average of the adjacent real pixels. The weighting is not continuously variable; there are five levels including both end cases. As shown in

Table 9-21, there are restrictions on the minimum zoom factor for some combinations of graphic format/resolution and video format. These are imposed by frame buffer bandwidth limitations.

Zooming in the Y (vertical) dimension can be done either with interpolation or with line replication. Interpolation produces better results and should be used at 2× or more whenever it can (based on available memory bandwidth). Line replication mode is selected when CR3E[4] is '1'. Line replication must be selected when occlusion is enabled, and for some combinations of graphic format/resolution and video format. These limitations are noted in Table 9-21.

9.5.8 Occlusion Support

When occlusion support is enabled, graphics and video can be mixed within the video window, on a pixel-by-pixel basis, as an overlay of video on graphics or graphics on video. This function is enabled by setting CR3E[7] to '1'. There are two quite distinct methods of selecting the pixel to be displayed, depending on the programming of CR1D[5].

The area of overlay is limited to the video window (which can be programmed to be the entire screen). When memory bandwidth permits, use this for live-video display in windowing environments.

9.5.8.1 Occlusion Support Using Color Key (Graphics Stream)

This mode is enabled by setting CR1D[5] to '0'. Each pixel of graphics data is compared to the contents of the color key registers, as shown in Table 9-22. If the comparison is true, the graphics pixel is replaced with the corresponding video pixel. If the comparison is false, the graphics pixel is displayed.

Table 9-22. Color Key Registers: CR1D[5] = 0

CR1D[4]	CR1D[3]	GRC	GRD	Case
0	0	8-bit color key	8-bit mask	8-bit compare under mask. '1' in mask causes corresponding bit position in color key to be ignored.
0	1	Low byte	High byte	16-bit compare (intended for 16-bpp graphics)
1	0	–	–	Reserved
1	1	Unused	Bit 7	GRD[7] compared to pixel[15] (intended for 1:5:5:5 RGB graphics)

This method of pixel selection uses the pixels in the graphics stream to select where to display the video pixels. This works well when other windows partially cover a video window, to put text in the video, or to mix two screens (foreground/background for video game applications). One graphics color value can be dedicated to allow the video to show through.

9.5.8.2 Occlusion Support Using Chroma Key (Video Stream)

This mode is enabled by setting CR1D[5] to '1'. Each pixel from the video stream is compared to the contents of three register pairs, shown in Table 9-23. This is also called 'self-keying video'.

This comparison occurs before horizontal zooming. If the video format is 8-bit LUT, the comparison occurs after the palette lookup. If the video format is AccuPak, the comparison occurs after the conversion to YUV.

Table 9-23. Chroma Key Registers: CR1D[5] = 1

Register	YUV Video	RGB Video
GRC	Y MIN	Red MIN
GRD	Y MAX	Red MAX
GR1C	U MIN	Green MIN
GR1D	U MAX	Green MAX
GR1E	V MIN	Blue MIN
GR1F	V MAX	Blue MAX

If all three components of the video source are within their respective ranges, the pixel is replaced with the pixel of graphics source displayed if window not been enabled. The comparison is inclusive. If a component of the video source is equal to its respective minimum or maximum, it is considered within the range.

This is typically for special live-video effects such as ‘blue screen’ (for example, where a weather reporter stands in front of a blue background, and is overlaid onto a different scene, such as a weather map). In this case, the chroma key is set to a range of blue hues.

9.5.9 Display Brightness Adjust

Register CR5B contains a 5-bit value added to the Y data (luminance) for each YUV pixel in the window. This allows a modest increase in brightness for MPEG playback. Experiments indicate that a value between 16 and 25 works well.

9.5.10 AccuPak™

AccuPak is a Cirrus Logic-proprietary method of compressing YUV into 1 byte per pixel (actually, four pixels are compressed into a 4-byte packet). This reduces the amount of storage required, as well as the display memory bandwidth requirements. The format of the AccuPak packet is shown in Figure 9-6 on page 9-14. The CL-GD5446 can compress YUV to AccuPak during the video capture process, then decompress the AccuPak back to YUV in the display process.

If CR51[2:0] is programmed to ‘011’, YUV 4:2:2 data is compressed to AccuPak as it is captured (refer to Table 9-25).

AccuPak video can only be displayed in the video window. If CR3E[3:1] is ‘001b’, the source data expands to YUV 4:2:2 prior to the color space conversion to RGB.

9.6 Video Capture

The CL-GD5446 can accept video from the V-Port and transfer it to the frame buffer. Typically, video captured this way is displayed in the video window. Since frame-rate conversion occurs in the CL-GD5446 frame buffer, it is not necessary to synchronize the video source with the graphics screen refresh. The data can be decimated or converted to AccuPak as it is captured.

9.6.1 Enabling Video Capture

Video capture is enabled when CR51[3] is '1' and the CR50[1:0] field is programmed for the V-Port. When this bit is initially programmed to '1', video capture begins on the next VREF (see Section 9.6.7 on page 9-45). So long as this bit remains '1', every field, every odd or even field, or every fourth field is captured. Table 9-24 shows how CR50[7], CR50[2], and CR58[6] control the captured fields.

Capturing alternate fields is called 'temporal' decimation. This is often used when realtime capture data is being written to a disk that may not have sufficient bandwidth to save every field.

Table 9-24. Video Capture Enable

CR51[3]	CR50[7]	CR50[2]	CR58[6]	Capture
0	X	X	X	Capture disabled
1	0	0	0	Capture every field
1	1	0	0	Capture odd fields
1	1	0	1	Capture even fields
1	1	1	0	Capture alternate odd fields
1	1	1	1	Capture alternate even fields

9.6.2 Data Format

CR51[2:0] specifies the format of the captured data. The encoding of this field is shown in Table 9-25. The capture and display formats are specified in separate registers. If the data is being captured for immediate display, for the best results, program these fields to compatible values.

Table 9-25. Video Capture Data Format

CR51[2:0]	Format	Note
000	YUV16	
001	RGB16	5:5:5 or 5:6:5
010	AccuPak	Decimation not supported.
011	YUV16 to AccuPak compress	Decimation not supported.
100	Reserved	
101		
110		
111	Y-data capture only	Can be used for luminance-only capture without video display. Requires YUV 4:2:2 data input.

9.6.3 Video Capture Buffer — Beginning Address and Offset

There are two sets of pointers that contain the video capture address. The beginning addresses of the two buffers are specified in Table 9-26. The low-order three address bits are '0', placing the capture buffers on qwords (8-byte boundaries). These same registers contain the Video Source Beginning addresses; when used for that function all 22 bits can be programmed.

Table 9-26. Video Capture Buffer Start Address

Address Bits (Correspond to 4-Mbyte Linear Address)	21:18	17:10	9:3	2:0
Video Buffer 1 (default display buffer)	CR3C[3:0]	CR3B[7:0]	CR3A[7:1]	0
Video Buffer 2 (default capture buffer)	CR58[3:0]	CR5A[7:0]	CR59[7:1]	0

9.6.3.1 Capture Buffer Addressing

Non-interlaced Video Capture

At the start of each video field, as indicated by the falling edge on VREF (see Section 9.6.7 on page 9-45) the starting address is copied from the appropriate buffer pointer register set into a start-of-line address register. At the start of each active line, as indicated by the first assertion of VACT after a leading edge of HREF, the start-of-line address register is copied into the video address counter. This is the register that actually addresses memory for stores. The start-of-line address then increments by the video window offset, as shown in Table 9-18 on page 9-36, for the next scanline of data. The video address counter increments by one (corresponding to address bit three) following each qword transfer from the video input FIFO to the frame buffer.

If CR5C[7] is '1', the value 1024 is the capture buffer offset rather than the video window offset.

Interlaced Video Capture

This mode is enabled by programming CR50[6] to '1'. At the start of each video field, as indicated by the leading edge of VREF (see Section 9.6.7 on page 9-45), the starting address is copied from the appropriate buffer pointer register set into a start-of-line address register. For odd fields, this register increments by the video buffer offset to point to the second scanline in the video window. At the start of each active line, as indicated by the first assertion of VACT after a leading edge of HREF, the start-of-line address register is copied into the video address counter. This is the register that actually addresses memory for stores. The start-of-line address then increments by twice the video buffer offset to point to the next scanline but one. The video data is converted from interlaced to non-interlaced as it is captured, by being stored into alternate scanlines and offsetting the odd field by one scanline.

If CR5C[7] is '1', the value 1024 is the capture buffer offset rather than the video buffer offset.

The odd field is indicated by HREF being high at the falling edge of VREF. See the timing diagram Even/Odd Field Timing in Chapter 11. If CR58[6] is '1', the sense of the field decode is inverted.

Luminance-Only Capture

Since the video capture and video display start addresses are independent, it is easy to capture any number of lines that are not displayed (if auto-switch double buffering is not being used). See Section 9.6.5 on page 9-43 for more information regarding luminance-only capture.

9.6.4 Double Buffering

The CL-GD5446 has support for double buffering of the video display and video capture buffers. This simplifies the 'tear-free' display of animation, video, video games, and other continuously updated screens. Double-buffering control is based on the frame-rate of the input video always being lower than the refresh rate of the display.

Two sets of buffer pointers are used, as shown in Table 9-17 on page 9-36 and Table 9-26 on page 9-42. While one buffer is being filled by the capture mechanism, the other is being displayed in the video window. When auto-switch is enabled (by programming CR5E[5:4] to '10'), the CL-GD5446 automatically switches between buffers each time a frame is captured.

When a frame capture is complete (as indicated by VREF), the CL-GD5446 switches capture buffers immediately (switching to the buffer being used for video display) and switches display buffers at the beginning of the video window in the next screen refresh cycle (switching to the buffer containing the frame just captured). The same buffer is used for capture and display until the end of the video window in the current display frame, but is not a problem since the capture is behind the display and cannot possibly catch up.

When auto-switch is enabled, CR3F[2] can be read to determine which buffer is currently the capture buffer.

9.6.5 Luminance-Only Capture

Luminance-only capture is enabled by two bits in register CR5C, as shown in Table 9-27. When counting lines to CR56[4:0], only lines with at least one valid PIXCLK are counted (that is, at least one PIXCLK accompanied with VACT high).

Table 9-27. Luminance-Only Capture Control

CR5C[5]	CR5C[6]	Function	Note
0	X	Luminance-only capture disabled	Count lines until CR56[4:0], then capture video.
1	0	Luminance-only capture enabled	Count lines until CR56[4:0], capture Y-only for one scanline, then capture video.
1	1	Luminance-only capture enabled	Capture Y data from VREF to CR56[4:0], then capture video.

Closed caption and TeleText data uses only the Y (luminance) portion of the YUV color space, so only the Y portion need be captured. If the Capture Data Format (Table 9-25) is programmed to '111', the CL-GD5446 captures and stores only Y bytes of the input YUV input during luminance-only scanlines, and then switches to normal video capture.

Closed caption data is transmitted during a single scanline of each odd field (usually line 21). The CL-GD5446 can be programmed to skip over the first n scanlines (programmed in CR56[4:0]), capture one scanline of Y (luminance) bytes, then switch to normal video capture.

TeleText data is transmitted during the vertical front porch period and can be present in all blanked lines. The CL-GD5446 can be programmed to capture only Y bytes during the first n scanlines (programmed in CR56[4:0]) and then switch to normal video capture.

Luminance-only data is stored at the beginning of the capture buffer. When CR5C[6] is '0', one extra offset is added to the Start address for the display.

9.6.6 Data Reduction and Clipping

The amount of data stored by video capture can be reduced by regularly discarding pixels and scanlines, as well as clipping the left edge and the top and bottom of the frame. These are programmed in the registers shown in Table 9-28.

Table 9-28. Data Reduction and Clipping

Function	Register	Note
Horizontal data reduction	CR52[2:0]	See Table 9-29. RGB and YUV capture format only.
Vertical data reduction	CR53[2:0]	See Table 9-29.
Horizontal capture delay	CR54[4:0]	Left-edge clipping. Up to 32 PIXCLKs of data.
Vertical capture delay	CR56[4:0]	Top clipping. Not available with TeleText capture
Maximum capture height	CR57	MAX scanlines. Extended with CR58[5].

The encoding of the values for data reduction is shown in Table 9-29. Data reduction is independent for the two dimensions. Register CR52 controls horizontal data reduction; register CR53 controls vertical reduction.

Table 9-29. Data Reduction Encoding

CR52[2:0] CR53[2:0]	Keep Every nth Pixel/Scanline	Scale Factor
000	All	Disable data reduction
001	Every 2nd	1/2 scale
010	Every 4th	1/4 scale
011	Every 8th	1/8 scale
100	Every 16th	1/16 scale
101	Every 32nd	1/32 scale
110	Reserved	–
111	Reserved	–

9.6.7 Pin Redefinition

Table 9-30 lists the pins that are redefined (CR50[1:0]) when the V-Port is enabled. These pins are all inputs when the V-Port is enabled.

Table 9-30. V-Port™ Pin Definition

Pin Name	V-Port™ Name	Description	Note
DCLK	PIXCLK	Video Pixel clock	Can be programmed as single or double edge.
EDCLK#	VREF	Video Vertical Sync	Rising edge indicates beginning of each field.
BLANK#	HREF	Video Horizontal Sync	Indicates start of each video line. Sense is programmable.
EVIDEO#	VACT	Video Active or $\overline{\text{BLANK}}$	When high, indicates valid video samples on PIXD.
P[7:0]	PIXD[7:0]	Video pixel input	Low byte in 16-bit configuration.
Reserved, BIOSA[13:10]	PIXD [15:8]	Video pixel input	Option high byte in 16-bit configuration. See CR50[4].

9.6.8 Transfer Protocol

9.6.8.1 Vertical Timing

The falling edge (high-to-low transition) of VREF defines the end of one frame and the beginning of the next. Several things occur on this edge:

1) The state of HREF and CR58[6] determine whether the next field is even or odd.

2) The CL-GD5446 determines whether to skip the current frame, based on whether it is odd or even and the state of CR50[7] and CR50[2].

If the field is to be captured:

3) The capture line counter is reset to '0'.

4) The internal capture address register is loaded with the beginning address of Video Buffer 1 or Video Buffer 2, depending on CR5E[5:4]. This is the capture address of the next field of video.

Depending on the state of CR5C[6:5], the CL-GD5446 counts or captures scanlines until CR56[4:0], as shown in Table 9-27. Once normal video capture commences, it continues until the next falling edge of VREF or until the line count equals the video capture maximum height (register CR57). In the latter case, the V-Port waits until the VREF falling edge. When counting lines to video capture maximum height, only lines with at least one PIXCLK with VACT high are counted. In addition, lines that are skipped due to data reduction are not counted. Only lines where pixels are actually stored are counted.

The falling edge of VREF defines the end of the field and the beginning of the next field.

9.6.8.2 Horizontal Timing

A scanline ends with the active edge of HREF, as controlled by CR50[1:0] (see Table 9-31). The following occurs with this edge:

1) The capture FIFO is flushed. Any data from the previous line is written into the frame buffer. A few microseconds is required for this, and must be provided before VACT is allowed to go active.

2) On the first PIXCLK accompanied by VACT high, the new line start address is loaded by adding the start address offset (or two times the start address offset for interlaced capture) to the previous start address.

PIXCLKs accompanied by VACT high are counted until CR54[4:0], but the data are not stored. This is *left-edge clipping*. When CR54[4:0] data pixels are skipped, pixels are captured with horizontal data reduction, until the next HREF active edge.

The active edge of HREF defines the end of the scanline.

9.6.9 Miscellaneous Controls

The register bits described in Table 9-31 control the details of how the captured video is clocked into the V-Port.

Table 9-31. Video Capture Protocol: Miscellaneous Controls

Register Bit(s)	Function	Sense
CR50[5]	Invert capture clock	1 = invert clock
CR50[4]	V-Port width	0 = 8 bits; 1 = 16 bits
CR50[3]	Capture clock input mode	0 = rising edge only; 1 = use both edges Only for 16-bit data through 8-bit V-Port with a PIXCLK at 16-bit rate.
CR50[1:0]	V-Port timing mode	00 = standard feature connector 01 = reserved 10 = rising edge HREF ends capture line 11 = falling edge HREF ends capture line

If CR5C[4] is '1', the alternate FIFO threshold in CR5C[3:0] is used instead of the default threshold in SR16[3:0]. The alternate threshold is only for scanlines included in the video window. To prevent data underflow during the video window, the alternate threshold is typically programmed to a value higher than SR16[3:0]. The CL-GD5446 uses the default threshold for scanlines not in the window, switching to the alternate threshold during the video window. This maximizes the available frame buffer bandwidth while concurrently avoiding overruns.

9.7 Overlay: CL-GD543X/'4X Compatibility

The CL-GD5446 supports the hardware overlay capabilities of the CL-GD543X/'4X family of controllers. The overlay logic requires that the video source be synchronous with the display. For this reason, many applications use the CL-GD5446 video capture and window features that synchronize the video to the display.

9.7.1 SWITCH: Which Pixels are Selected

The SWITCH signal determines which pixels are selected to overlay or display in an alternate DAC mode. This takes place on a pixel-by-pixel basis. The source of the SWITCH signal is controlled by register bits CR1A[3:2] and CR1D[6]. There are three cases.

Timing

The SWITCH signal can come from the EVIDEO# pin or the internally generated OVRW# signal. This selection method is called *timing*. This method selects pixels according to their position on the screen.

Key

The SWITCH signal can be generated with the Color Key Compare logic. This selection method is called *key*. This method selects pixels according to their color without regard to their position on the screen. The color key is always compared to the graphics pixels.

Timing AND Key

The SWITCH signal can be generated with the timing signal (either EVIDEO# or OVRW#) AND'ed with the output of the Color Compare logic. This method allows both the color and position of a pixel to be considered.

Table 9-32 enumerates the methods of generating the SWITCH signal.

Table 9-32. SWITCH Selection

CR1A[3]	CR1A[2]	CR1D[6]	Source of SWITCH	EVIDEO#
0	0	X	VGA-compatible operation: SWITCH disabled	Input
0	1	0	EVIDEO# pin	Input
0	1	1	OVRW# signal	Input
1	0	0	EVIDEO# pin AND'ed with Color Key Compare	Input
1	0	1	OVRW# signal AND'ed with Color Key Compare	Input
1	1	X	Color Key Compare	Output

9.7.2 Video Overlay

Anytime the CR1A[3:2] field is programmed to a non-zero value, the Pixel bus drivers are turned off and the bus becomes an input. Pixels that have SWITCH active are replaced on the screen with pixels from the Pixel bus. SWITCH is controlled by register bits CR1A[3:2] and CR1D[6] as shown in Table 9-32.

If SR18[6] is programmed to ‘1’, data from the frame buffer is used even if the SWITCH signal is active. This allows DAC Mode Switching without overlay, useful for displaying a window of data from the frame buffer in a mode other than VGA palettized.

9.7.3 DAC Mode Switching

The RAMDAC mode can be dynamically changed between standard VGA 8-bpp through the palette and another mode (such as 8- or 16-RGB). DAC mode switching is controlled by the SWITCH signal and can occur on a pixel basis.

In every case where DAC mode Switching is used, one mode is the standard VGA 8-bit mode through the LUT. The second mode is as programmed into the HDR. If DAC mode switching is disabled (see Table 9-33), any desired DAC mode can be used.

The CL-GD5446 can be programmed so that DAC mode switching occurs when SWITCH is active, when SWITCH is not active, or not at all. This is shown in Table 9-33.

Table 9-33. SWITCH and DAC Mode Switching

CR1D[2:1]	DAC Mode Switching
00	Select Extended DAC mode on SWITCH true
01	Select Extended DAC mode on SWITCH false
1X	DAC mode switching disabled

9.7.4 Generating SWITCH

The various methods of generating the SWITCH signal are discussed in the following sections. The programming of registers CR1A and CR1D are discussed and a block diagram presents how to configure a system to use that mode of generating SWITCH.

Whenever the pixel bus is switched to inputs, the external video generator must always drive the pixel bus to valid CMOS levels. This is true even for the portion of the frame where the CL-GD5446 is not actually using the data from the bus. This prevents the bus pins from floating to CMOS threshold and oscillating.

9.7.4.1 SWITCH with EVIDEO#

SWITCH with EVIDEO# is selected when the CR1A[3,2] field is programmed to ‘01’. CR1D[6] must be programmed to ‘0’. The block diagram in Figure 9-23 on page 9-49 indicates how a system can be configured to use this mode.

The external video source determines which pixels are selected. The CL-GD5446 simply displays data from the display memory or data from the Pixel bus according to whether EVIDEO# is active (but refer to SR18[6]). For every pixel that is to be overlaid, the external video source must drive EVIDEO# active and provide either 8- or 16-bit video at the Pixel bus. Setup and hold time requirements are specified in the timing diagrams in Chapter 11, “Electrical Specifications”.

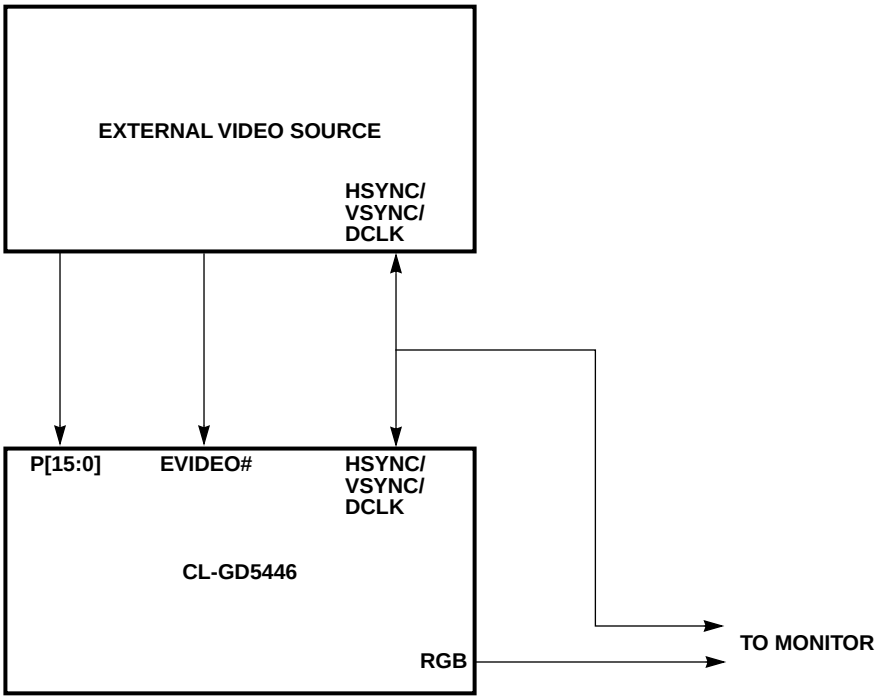


Figure 9-23. SWITCH with EVIDEO#

The system designer must guarantee that the external video source and the CL-GD5446 are precisely synchronized. Otherwise the overlaid video appears at randomly on the screen. This is why the SYNC and DCLK signals *must* be available at both blocks. There are two basic approaches. Either the external video source must synchronize to the CL-GD5446 or the CL-GD5446 must synchronize to the external video source. In the last example, refer to Appendix B9, "GENLOCK Support".

9.7.4.2 SWITCH with Color Key

SWITCH with color key is enabled when the CR1A[3:2] field is programmed to '11'. In this mode, the selection of pixels to overlay is strictly controlled by the contents of display memory. If a graphics pixel compares with the contents of the Color Key register, SWITCH is active. In this mode, EVIDEO# is a normally low output, going high one VCLK period before any pixel is replaced. If horizontally contiguous pixels are to be replaced, EVIDEO# remains appropriately high.

The matching of the pixel with the color key is done under a mask. Register GRC contains the color key; register GRD contains the mask. The key is compared with the pixel from display memory only for those bits where the mask is a '0'. If values '0xFC'–'0xFF' are used for the key, register GRC can be loaded with any value in the range '0xFC'–'0xFF', and register GRD is loaded with the value '0x03'.

In cases where the CL-GD5446 is programmed for other than 8-bpp, the most-significant byte of the pixel is compared; other(s) are ignored. This is shown in Table 9-34.

Table 9-34. Color Key Bytes

Mode	Byte Compared
8-bpp	Every byte
16-bpp	High byte
24-bpp	Red byte

The CL-GD5446 can be programmed for a 16-bit color key compare. If the 16-bit parallel transfer mode is selected by programming the SR7[2:1] field to '11', and CR1D[3] is programmed to '1', all 16 bits of each pixel are compared. The low byte of VGA data is compared to the value in register GRC and the high byte of VGA data is compared to the value in register CRD. No mask is available. This is only for 16-bpp graphics modes.

9.7.4.3 SWITCH with Color Key AND'ed with EVIDEO#

SWITCH with color key AND'ed with EVIDEO# is selected when the CR1A[3:2] field is programmed to '10'. The SWITCH signal is the logical AND of the color key comparison and EVIDEO# is low. EVIDEO# is an input in this mode. This mode is intended to restrict the color key effect to a specified area.

This mode can also mechanize a dynamic window with a horizontal resolution finer than an 8-pixel character clock. The coarse horizontal timing is generated with the on-chip window timing generator as previously described, and the fine (pixel resolution) is controlled by changing either the contents of display memory or the color key mask.

9.7.4.4 SWITCH with Color Key AND'ed with OVRW#

SWITCH with color key AND'ed with OVRW# is selected when the CR1A[3:2] field is programmed to '10'. CR1D[6] must be programmed to '1'. The SWITCH signal is the logical AND of the color key comparison and the internal OVRW# signal is active. EVIDEO# is an input in this mode, but is not used. This mode is intended to restrict the color key effect to a specified area.

9.7.4.5 SWITCH with OVRW#

SWITCH with OVRW# is selected when the CR1A[3,2] field is programmed to '01' and CR1D[6] is programmed to '1'. This mode allows the CL-GD5446 to generate its own window. Do not confuse this with the video window generator described in Section 9.5 on page 9-31.

The external video source must monitor OVRW# to determine which portion of the screen to overlay with video. For every pixel to be overlaid, either 8- or 16-bit video at the Pixel bus must be provided.

9.7.4.6 OVRW# — On-chip Window Timing Generator

The CL-GD5446 contains logic to generate timing for a single rectangular window. If this function is enabled, the window timing comes out on OVRW#, which can be fed back into EVIDEO#; the device specifies its own window. The internal signal is also directly available as the SWITCH signal.

This mode is selected by programming CR1B[5] to '1'. This allows the blanking term to the palette DAC to come from the display enable, and there is no border. This makes the blank generator logic available, which is used as a window generator. The timing diagram in Figure 9-24 shows how the

Blank Start and Blank End registers specify either the horizontal or vertical component of the window; the other component is similar. The Horizontal Blank End field is extended to 8 bits and the Vertical Blank End is extended to 10 bits.

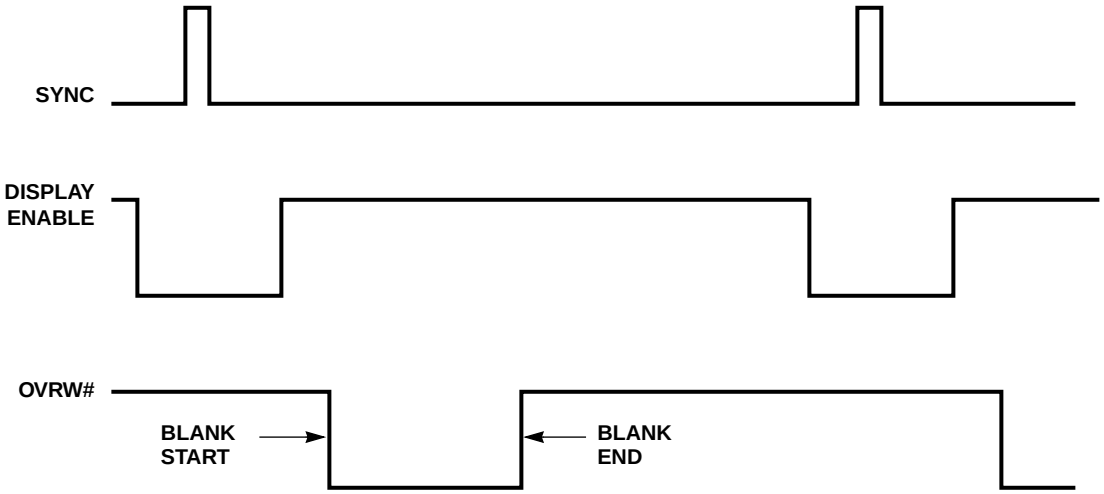


Figure 9-24. OVRW# Generation

When the vertical and horizontal timings generated coincide, OVRW# is driven low as an indicator to the external video generator. The term is available internally to specify the window timing to the CL-GD5446.

This method of generating timing has four restrictions that require consideration:

1) Since the blank timing generation logic is for window generation, there can be no border. This is inconsequential since VESA timing specifications do not include a border.

2) Since the horizontal counters operate on an 8-pixel character clock, the resolution of the horizontal component of the window is 8 pixels. This can be partially overcome by using EVIDEO# and color key overlay, as previously discussed.

3) There can only be a single window and it must be rectangular.

4) The first, third, or subsequent scanline (or group of 8 pixels) can be selected as a window boundary, not the second scanline (or group of 8 pixels).

When programming horizontal and vertical blanking, nine registers must be programmed to set the four blank parameters. Five of these registers contain bits related to other functions.

Table 9-35. Programming of Horizontal and Vertical Blanking

Register	Bits	Parameter	Other Bits
CR2	7:0	Horizontal Blank Start [7:0]	–
CR3	4:0	Horizontal Blank End [4:0]	7:5
CR5	7	Horizontal Blank End [5]	6:0

Table 9-35. Programming of Horizontal and Vertical Blanking *(cont.)*

Register	Bits	Parameter	Other Bits
CR7	3	Vertical Blank Start [8]	7:4, 2:0
CR9	5	Vertical Blank Start [9]	7:6, 4:0
CR15	[7:0]	Vertical Blank Start [7:0]	–
CR16	[7:0]	Vertical Blank End [7:0]	–
CR1A	[7:6]	Vertical Blank End [9:8]	3:0
CR1A	[5:4]	Horizontal Blank End [7:6]	3:0

9.7.5 16-Bit Pixel Bus

When the CL-GD5446 is configured for the PCI bus, it can be programmed for a 16-bit input pixel bus. Table 9-36 shows the pins used for the upper 8 bits.

Table 9-36. Pixel Bus Extension (Input only)

P-Bit	Pin	Was
P8	45	Reserved
P9	44	Reserved
P10	40	BIOSA13
P11	39	Reserved
P12	38	Reserved
P13	37	BIOSA12
P14	36	BIOSA11
P15	35	BIOSA10

To enable the Pixel bus extension, the CL-GD5446 must be configured for PCI host bus, the BIOS ROM must be disabled by programming PCI30[0] to ‘0’, and GR18[6] must be programmed to ‘1’. This makes the following three overlay modes available.

Table 9-37. Overlay Modes with Pixel Bus Extension

VGA Data	MAX Pixel Clock	(Modes)	Overlay Data	Overlay Clocking
8-bit LUT	50 MHz	5Ch at 75 Hz	16-bit RGB	Pixel clock
8-bit LUT	40–80 MHz	60h at 75 Hz	16-bit RGB	Pixel clock ÷ 2 with interpolation
16- bit RGB	50 MHz	60h at 75 Hz	16-bit RGB	Pixel clock

9.7.6 Clock Doubling with Interpolation

When the 16-bit Pixel bus extension is enabled, clock doubling and overlay interpolation can be enabled by programming GR81[5] to '1'. This allows 16-bit RGB to be overlaid onto 8-bit VGA palettized data with a pixel clock of up to 80 MHz.

SR7[3:1] are programmed to '011' so that data is provided to the RAMDAC 16 bits in parallel. VCLK is programmed to 1/2 the actual pixel clock. The RAMDAC provides a clock doubling function and each 16-bit word is unpacked and converted at twice the VCLK rate.

The 16-bit RGB overlay data is input at the VCLK frequency (that is, for every other pixel). The pixels in between are generated in the CL-GD5446 by averaging the two adjacent actual pixels. Since there is no following pixel for interpolation, the last pixel to be overlaid is replicated.

9.7.7 Using the Internal Clock with DCLK as Input

The overlay modes described in this section are intended to be used in a configuration where the video clock is provided by the external video source. If the clock were to be provided by the CL-GD5446, the result is clocking data into a device where the device itself is providing the clock.

The VCLK VCO can be sourced onto the MCLK pin and an external buffer can be supplied to drive MCLK (which is the VCLK) onto the DCLK pin. This makes both DCLK and the Pixel bus inputs. This is a configuration option.The block diagram in Figure 9-25 on page 9-53 indicates how a system can be configured to use the buffer to drive MCLK.

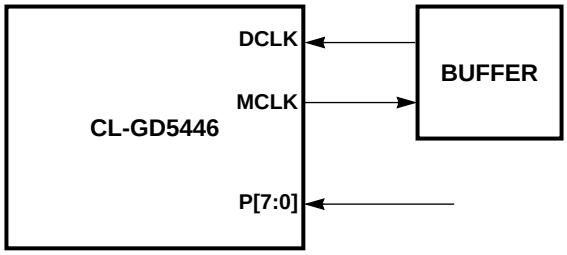


Figure 9-25. Using MCLK as DCLK

9.7.8 Static Overlay

The CL-GD5446 supports the standard VESA pass-through function, during which EVIDEO# is statically driven low and video is driven onto the P-bus. The entire frame is overlaid, and the contents of display memory are ignored. EDCLK# is typically driven low so that DCLK, as well as the video, are supplied externally.

9.8 Hardware Cursor

The CL-GD5446 supports a hardware cursor that is compatible with the CL-GD543X/'4X family. The cursor is 32×32 or 64×64 and available in 16-color planar and all packed-pixel modes at all supported resolutions and refresh rates.

The hardware cursor replaces the software mouse pointer commonly used by GUI applications. The hardware cursor eliminates the need for applications software to save and restore the screen

data as the cursor position changes. The application initializes the cursor pattern(s) once, then only needs to update the cursor position to move the cursor on the screen. Multiple cursor images can be loaded into the frame buffer and selected as necessary.

Programming examples are in Section 9.14.2 on page 9-64.

9.9 Frequency Synthesizer Programming

The CL-GD5446 contains two built-in frequency synthesizers. One synthesizer is VCLK, which drives the pixel clock and display synchronization timing. The other synthesizer is MCLK, which drives the display memory control state machine and the host interface state machine.

9.9.1 VCLK

VCLK generates the pixel clock, which in turn generates the horizontal and vertical timing for the CRT. This section describes how to program the VCLK. CRTC programming is discussed in Section 9.11 on page 9-58.

9.9.1.1 VCLK Sources

The VCLK source is determined by a number of factors, as indicated in Table 9-38.

Table 9-38. VCLK Source

EDCLK Pin	SR1F[6]	SR1E[0]	Misc. Register 3C2[3:2]	DAC Clock Pixel Timing	CRTC Clock Monitor Timing
1	0	X	00	VCLK0	VCLK0
1	0	X	01	VCLK1	VCLK1
1	0	X	10	VCLK2	VCLK2
1	0	X	11	VCLK3	VCLK3
1	1	0	X	MCLK	MCLK
1	1	1	X	MCLK ÷ 2	MCLK ÷ 2
0	0	X	00	VCLK0	DCLK pin
1	1	X	01	VCLK1	DCLK pin
0	1	0	0X	MCLK	DCLK pin
0	1	1	0X	MCLK ÷ 2	DCLK pin
0	x	x	1X	DCLK pin	DCLK pin

9.9.1.2 VCLK Programming

The VCLK synthesizer is controlled by one of four register pairs, selected by MISC[3:2]. Table 9-39 shows these registers.

Table 9-39. VCLK Programming

MISC[3:2]	VCLK	Numerator	Denominator/ Post Scalar	Usage Note
00	VCLK0	SRB	SR1B	Nominally 25 MHz VGA Graphics modes
01	VCLK1	SRC	SR1C	Nominally 28 MHz VGA Text modes
10	VCLK2	SRD	SR1D	Unused
11	VCLK3	SRE	SR1E	Extended modes

The Numerator is a 7-bit integer, right justified in the corresponding register. The Denominator is a 7-bit integer, scaled at bit position 1 in the corresponding register. The PostScalar is a single bit at bit position 0 in the corresponding register. The values programmed into the active register pair controls the VCLK synthesizer, according to the appropriate formula. Reference is nominally 14.31818 MHz.

$$VCLK = Reference \bullet \left(\frac{Numerator}{Denominator}\right)$$ Postscalar = 0 **Equation 9-6**

$$VCLK = Reference \bullet \left(\frac{Numerator}{Denominator \bullet 2}\right)$$ Postscalar = 1 **Equation 9-7**

The Cirrus Logic BIOS programs VCLK when a graphics mode is set. If the Cirrus Logic BIOS cannot be used or if a mode not available in the BIOS must be programmed, the following approach can be used for selecting a numerator/denominator combination.

A spread sheet is available from Cirrus Logic that evaluates every combination of numerator and denominator. This can be a starting point. Typically, a large number of numerator/denominator combinations evaluate to the desired frequency. Best results are obtained by selecting values near the center of their respective ranges and if the postscalar is programmed to ‘1’, especially at lower frequencies. After these criteria have been used to sort through potential combinations, the final selection is made empirically.

For pixel frequencies above ~85 MHz, program VCLK to one-half the desired pixel frequency. The clock is then multiplied by two in the color palette (the HDR is programmed to ‘0xCAh’ for this). This is only for 8-bpp LUT mode.

9.9.1.3 Using MCLK as VCLK

If the MCLK and VCLK are programmed to frequencies close to each other (within ~1%) or to frequencies that are nearly multiples of each other, they can interfere with each other. This can show up as screen 'jitter'. The solution is to shut down the VCLK synthesizer and use MCLK (or MCLK ÷ 2) as VCLK. This is selected by programming SR1F[6] to '1'. This function is currently not used by the Cirrus Logic BIOS.

9.9.2 MCLK Programming

MCLK is directly programmed by writing the value into SR1F[5:0] that most nearly corresponds to the desired frequency.

$$MCLK = SR1F \bullet \left(\frac{Reference}{8}\right)$$ **Equation 9-8**

Table 9-40 shows representative examples, assuming a reference frequency of 14.31818 MHz. Refer to Appendix B3, "Memory Configurations and Timing", to see how MCLK relates to DRAM speed.

Table 9-40. MCLK Programming

SR1F[5:0] (decimal)	SR1F[5:0] (hex)	MCLK Frequency	SR1F[5:0] (decimal)	SR1F[5:0] (hex)	MCLK Frequency
28	1C	50.1 MHz	37	25	66.2 MHz
29	1D	51.9 MHz	38	26	68.0 MHz
30	1E	53.7 MHz	39	27	69.8 MHz
31	1F	55.5 MHz	40	28	71.6 MHz
32	20	57.3 MHz	41	29	73.4 MHz
33	21	59.1 MHz	42	2A	75.2 MHz
34	22	60.9 MHz	43	2B	77.0 MHz
35	23	62.6 MHz	44	2C	78.7 MHz
36	24	64.4 MHz	45	2D	80.5 MHz

9.10 Power Management

The CL-GD5446 features comprehensive PC power-management functions that support compliance with the United States Environmental Protection Agency's Energy Star Computer Program. The Energy Star Program is a voluntary program promoting energy efficient technology for desktop computers. Compliance with the Energy Star Program is a qualification for federal government purchases of computers, monitors, and printers.

The best way to program the power-management functions is to use the VESA VBE/PM BIOS functions. These functions are used by the screen saver provided with the Cirrus Logic Windows drivers. Register programming should be used only if the Cirrus Logic BIOS is not available.

9.10.1 Register Bits

Four bits in register GRE are for power management. These are shown in Table 9-41. Refer to Section 8.24 on page 8-33. Note that none of these bits turn off display memory refresh.

Table 9-41. GRE: Power Management Bits

GRE Bit	Function	Note
GRE[4]	System-level power management	Used with GRE[3]
GRE[3]	Static Clock Mode	
GRE[2]	Static VSYNC	Powers down DAC
GRE[1]	Static HSYNC	Powers down DAC

9.11 CRTC Programming

The following section discusses programming the registers in the CRTC. These registers control video timing.

9.11.1 VESA® Timing Specifications

The video display for most modes is defined in VESA Monitor Timing specifications. Wherever possible, Cirrus Logic devices use VESA timing. Table 9-42 indicates the VESA resolutions and refresh rates as of May 1995.

Table 9-42. VESA® Monitor Timing Specifications[a]

Refresh × Resolution	Interlaced	56	60	65	70	72	75	80	85
640 × 350			IS[b]		IS				VP[c]
640 × 400			IS		IS				VP
640 × 480			IS			VO[d]	VS[e]		VP
800 × 600		VS	VS			VO	VS		VP
1024 × 768	IS		VS		VS		VS	VP	VP
1152 × 864			VS		VP		VP		VP
1280 × 960	VP		VP				VP		VP
1280 × 1024	VP		VP				VS		VP
1600 × 1200	VP		VP	VP	VP		VP	VP	VP

[a] The notations used in this table do not necessarily indicate that the Cirrus Logic BIOS supports the corresponding specification. This table is for reference only.

[b] IS indicates industry standard.

[c] VP indicates VESA proposal.

[d] VO indicates obsolete VESA standard.

[e] VS indicates current VESA standard.

9.11.2 CRTC Timing

Figure 9-26 can be aid to understanding how CRTC timing is generated. There are two sets of timing, horizontal and vertical, combined produce an orthogonal raster. The timings are generated in a similar manner to each other.

Horizontal timing is generated by counting character clocks (this term is left over from character graphics; for packed pixel graphics modes, it means the pixel clock divided by eight). Vertical timing is generated by counting scanlines.

Each counter counts from zero to its maximum value and then resets to zero and begins again. The zero value of the horizontal counter corresponds to the first (left-most) pixel of active video. The zero value of the vertical counter corresponds to the top scanline on the screen.

Refer to Figure 4-1 on page 4-27 for a detailed diagram of the CRTC Timing registers.

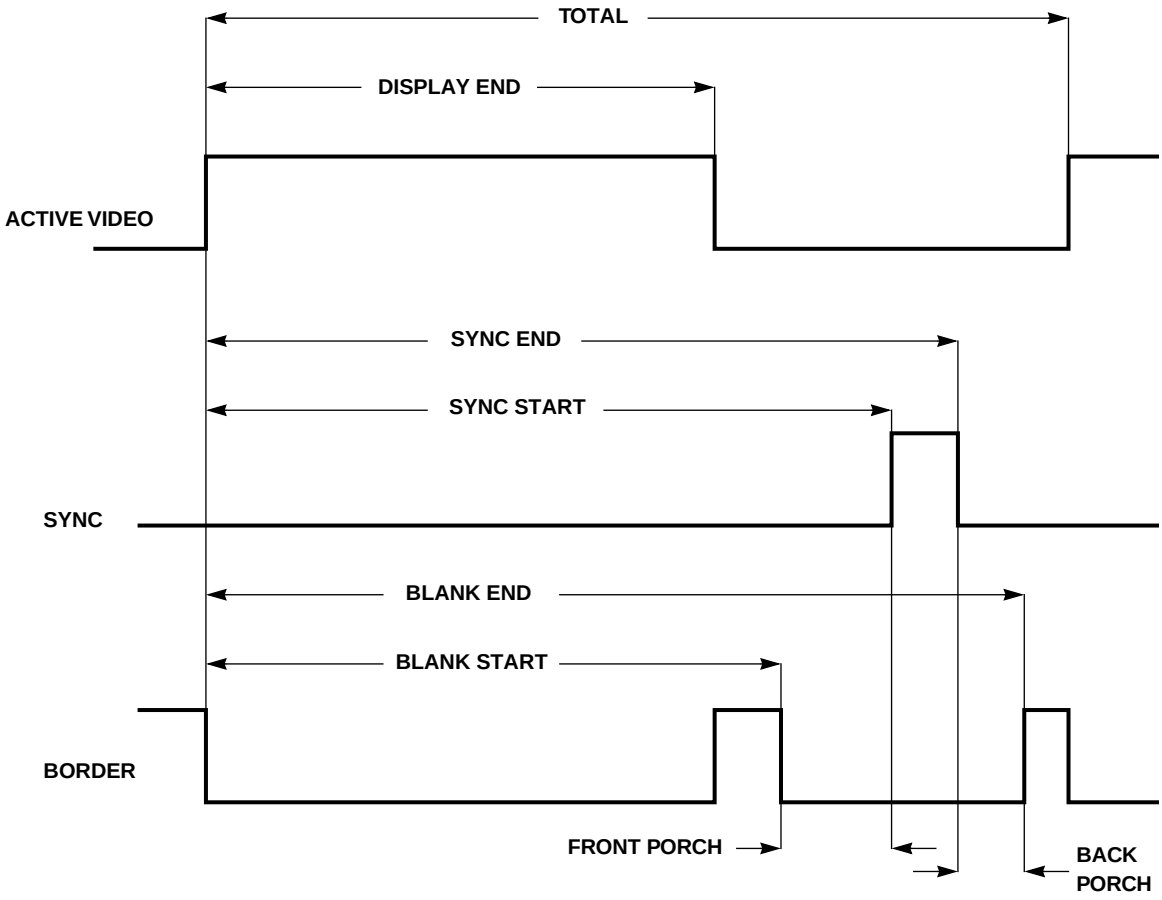


Figure 9-26. CRTC Timing

As either counter increases from zero, counting either pixels or scanlines, it arrives at the values programmed for the various events in the raster. These are shown in table 9-43. Many of the values do not fit into a single 8-bit register. The extensions, assigned either as part of the VGA specification or by Cirrus Logic, are all presented in Table 4-3 on page 4-28, in Chapter 4, "VGA Core Registers". For Sync End and Blank End, in both horizontal and vertical, the compare is for fewer than the total number of bits in the respective counter. This results in limitations on the width of the sync pulses and the blanking periods.

Table 9-43. CRTC Registers

Event	Horizontal	Vertical	Next Period
Display End	CR1	CR12	Border or front porch
Blank Start	CR2	CR15	Front porch
Sync Start	CR4	CR10	Sync

Table 9-43. CRTC Registers *(cont.)*

Event	Horizontal	Vertical	Next Period
Sync End	CR5	CR11	Back porch
Blank End	CR3	CR16	Border or active video
Total	CR0	CR6	Active video (next scanline or frame)

Figure 9-26 shows the border timing. If no border is used, Blank starts and ends with active video.

9.11.3 Non-Standard Timing Parameters

The VESA standards, referenced in Section 9.11.1, provide precise timing details for standard monitors using standard resolutions. There are two situations where the standard values cannot be used. It may be necessary to program a non-standard resolution or refresh rate, or program a standard resolution for a non-standard monitor.

The programmer can use the following general approach. First, decide on the horizontal timing. If the monitor manufacturer cannot provide guidelines for the horizontal total and blank timing, opt for another monitor.

$$Horizontal\ Active\ Time = (Horizontal\ Total\ -\ Horizontal\ Blanking) \qquad \textbf{Equation 9-9}$$

The horizontal active time can be calculated by subtracting the horizontal blank from the horizontal total. Dividing horizontal active by the number of pixels per scanline yields the pixel time. The pixel frequency is the reciprocal of the pixel time.

$$Pixel\ Time = \frac{Horizontal\ Active}{Pixels/Scanline} \qquad \textbf{Equation 9-10}$$

$$VCLK = \frac{1}{Pixel\ Time} \qquad \textbf{Equation 9-11}$$

The VCLK synthesizer can be programmed using one of the equations in Section 9.9.1.2 on page 9-55. For most frequencies, a number of solutions yields the desired frequency. In general, the best results are obtained by selecting a solution with both the numerator and denominator in the middle of their respective ranges. Use the postscalar, if possible, especially for frequencies below ~30 MHz. If the pixel clock is above ~85 MHz, clock doubling must be used.

The horizontal counter counts with a resolution of 8 pixels. All horizontal timing is in terms of this counter. The horizontal display end and horizontal total values is determined by the pixels-per-scanline and horizontal period, respectively.

If no horizontal border is used, program horizontal blank start to correspond to horizontal display end; program horizontal blank end to correspond to horizontal total. If a horizontal border is used, horizontal blank start follows horizontal display end and horizontal blank end occurs before horizontal total. The differences are the desired border width.

Next, program the horizontal sync pulse. The horizontal sync pulse generally begins near the start of horizontal blanking (horizontal blank start). The polarity of the sync pulse is programmed in the

MISC register. Typically, the sync width is not important; most monitors require only the leading edge. The position of the sync pulse can be adjusted to position the active video on the monitor. Having the sync pulse start earlier moves the video to the left.

The vertical parameters are selected in the same way, except they are in terms of scanlines.

9.12 Chip Identification

If a Cirrus Logic BIOS is available, the correct way to identify the CL-GD5446 is to execute the extended BIOS call, Inquire VGA Type (See Appendix C3, "BIOS Extensions"). This call returns the value 39h for the CL-GD5446. If a Cirrus Logic BIOS is not available, use the VENDOR/DEVICE ID in PCI00. The CL-GD5446 then returns the value 101300B8h.

The CL-GD5446 also implements registers SR6 and CR27 exactly as all Cirrus Logic desktop controllers (for compatibility).

9.13 CL-GD5446 Interrupt System

9.13.1 External Interrupt Pin

If register CF14 is '1' (no pull-down resistor installed on MD62), the INTR# pin is never driven low and no interrupt is claimed in register PCI3C. This completely and permanently disables the CL-GD5446 from making an interrupt request. If register CF14 is '0' (pull-down resistor installed on MD62), INTR# is driven low — if enabled with GR17[2] and an internal interrupt request is active. If CF14 is '0', an interrupt is claimed in register PCI3C.

If GR17[2] is '0', the INTR# pin is prevented from being driven low. This is the default state for GR17[2]; the bit must be set to '1' to enable the interrupt.

If register CF14 is '0' and GR17[2] is '1', an internal interrupt request causes INTR# to be driven low, signalling an interrupt request.

9.13.2 Internal Interrupt Request

In addition to the standard end-of-frame interrupt supported by VGA, the CL-GD5446 can generate an interrupt from the falling edge of V-Port VREF. This interrupt source is selected if CR3F[5] is '1'.

Table 9-44. VGA/V-Port™ Interrupt

Function	VGA	V-Port™
CR3F[5]	0	1
Pending Status	FEAT[7]	GR17[4]
Enable Interrupt	CR11[4]	CR3F[1]
Clear Interrupt	CR11[5]	CR3F[1]

9.14 Programming Examples

9.14.1 Using YUV Planar (Video Aperture)

The following general procedure writes data into the frame buffer using the video aperture (the fourth 4-Mbyte aperture in the linear address space). See Section 9.2.7.4 on page 9-10 for a description of how the data is disassembled and then packed into pixels. Table 9-45 shows assumptions used.

Table 9-45. YUV Planar Example

Parameter	Value	Note	Step
DRAM Size	1 Mbyte	1,048,576 bytes	Given
Display	1024 × 768, 256 colors	786,432 bytes	Given
Frame Size	352 × 240 pixels	84,480 pixels	Given
Y (Luminance) data	352 × 240 pixels	84,480 bytes	Given
U, V (Chrominance) data	176 × 120	21,120 bytes each	Given
Window Line Offset	704 bytes	2 bytes/pixel	Section 9.14.1.1
Video Buffer Size	704 × 240	168,960 bytes	Section 9.14.1.1
Window Start Address	786,432		Section 9.14.1.2
Luminance data address			Section 9.14.1.3
Chrominance data address			Section 9.14.1.4, Section 9.14.1.5

The data generated by the MPEG or Indeo decoder has one U and V value for every four Y values (2 × 2 square).This means that the same U and V data must be duplicated in the frame buffer. This duplication is the final step in the process.

9.14.1.1 Select Window Scanline Size (Offset)

Select the Window Scanline Offset (see Section 9.6.3 on page 9-42), which can be any value equal to or greater than the horizontal frame size (and must be on a qword boundary). For this example, select 704 bytes. Since the window is 240 scanlines, the buffer size is 168,960 bytes. This fits in the 1-Mbyte frame buffer above the mode 60 display, leaving room for the cursors. Note that while the window scanline offset is 704, the host offset into the aperture is one-half that (352 bytes).

9.14.1.2 Select Window Start Address

There is no reason to place the video window anywhere but directly above the frame buffer. In a system with more than a single Mbyte, select the window start address so that the window does not overlap the Mbyte boundary in the aperture (see Section 9.5.4 on page 9-35).

9.14.1.3 Write Y Data

The beginning address to write the Y (luminance) data for each scanline can be calculated with Equation 9-12.

$$YAddress = PCI10 + MbyteOffset + \frac{ScanLineBase}{2}$$ **Equation 9-12**

In this example, the MbyteOffset is 12M (the first Mbyte of the fourth aperture). The ScanLineBase is 768K for the first scanline with an offset of 352 bytes for subsequent scanlines. The program should write 352 bytes of Y data for each scanline. The data for each scanline is (or should be) contiguous in the decoder's buffer and can be written to contiguous addresses in the frame buffer. The addresses to the beginning of each scanline are divided by two. 240 scanline of Y data must be written.

9.14.1.4 Write V Data

The beginning address to write the V (first chrominance) data for each scanline can be calculated using Equation 9-13.

$$VAddress = PCI10 + MbyteOffset + \frac{ScanLineBase}{4} + 512K$$ **Equation 9-13**

In this example, the MbyteOffset 12M (the first Mbyte of the fourth aperture). The ScanLineBase is 768K for the first scanline with an offset of 352 bytes for subsequent scanlines. The program should write 176 bytes of V data for each scanline. The data for each scanline is (or should be) contiguous in the decoder's buffer and can be written to contiguous addresses in the frame buffer. The addresses to the beginning of each scanline are divided by four. The 512K correction gets the addresses into the range of 512K–768K. The program writes a total of 120 scanlines of V data; it goes to the even numbered scanlines only.

9.14.1.5 Write U Data

The beginning address to write the U (second chrominance) data for each scanline can be calculated using Equation 9-14.

$$VAddress = PCI10 + MbyteOffset + \frac{ScanLineBase}{4} + 768K$$ **Equation 9-14**

In this example, the MbyteOffset is 12M (the first Mbyte of the fourth aperture). The ScanLineBase is 768K for the first scanline with an offset of 352 bytes for subsequent scanlines. The program should write 176 bytes of U data for each scanline. The data for each scanline is (or should be) contiguous in the decoder's buffer and can be written to contiguous addresses in the frame buffer. The addresses to the beginning of each scanline are divided by four. The 768K correction gets the addresses into the range 768K–1024K. 120 scanlines of U data are written.

9.14.1.6 Copy the Chrominance Data to Odd Scanlines

When the procedure above is complete, the chrominance data is written to every other scanline (the even scanlines) and must be replicated into the odd scanlines. This can be done by writing each scanline twice or it can be done with a BitBLT. If a BitBLT is selected, use the parameters in Table 9-46 on page 9-64.

Table 9-46. Replicating Chrominance Data with BitBLT

Function	Value	Note
SR2[3:0]	0011b	Protect Y bytes
Destination Address	Window Start Address plus offset	Second scanline
Source Address	Window Start Address	
Width	Window Width	
Height	Window Height ÷ 2	Every other scanline
Destination Pitch	Window Offset × 2	Every other scanline
Source Pitch	Window Offset × 2	Every other scanline

9.14.2 HW Cursor Example

```
// ***************************************************************************
// HWC_XMPL.CPP
//
// THIS FILE CONTAINS THE PATTERN FOR THE HARDWARE CURSOR TESTS.
// THE CURSOR PATTERN IS THE CIRRUS LOGIC LOGO.
//
// DATA:
//      _64x64_hw_cursor[]  -- USER-MODIFIABLE TABLE OF CURSOR
//      _64x64_planes_0_1[] -- DISPLAY MEMORY-READY DATA FOR CURSOR PLANE 0
//
// THE DATA IN _64x64_hw_cursor[] IS CODED THE FOLLOWING INTEGERS
//      0 == TRANSPARENT    DISPLAY BACKGROUND IS NOT CHANGED
//      1 == INVERTED       DISPLAY BACKGROUND IS INVERTED
//      2 == BLACK          INTERNAL PALETTE DAC EXTRA LUT 00H
//      3 == BLUE           INTERNAL PALETTE DAC EXTRA LUT FFH
// ***************************************************************************

#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <conio.h>

unsigned char _64x64_planes_0_1[1024];
unsigned char _32x32_planes_0_1[256];

char _32x32_hw_cursor[1024] =
{
//                1         2         3
```



```
//       01234567890123456789012345678901
        "22222222222222222222222222222222" // 0
        "22222222222222222222222222222222"
        "22000000000000000000000000000022" // 2
        "22000000000000000000000000000022"
        "22003333333333333333333333330022" // 4
        "22003333333333333333333333330022"
        "22003300000000000000000000330022" //  6
        "22003300000000000000000000330022"
        "22003300222222222222222200330022" //  8
        "22003300222222222222222200330022"
        "22003300220000000000002200330022" //  10
        "22003300220000000000002200330022"
        "22003300220033333333002200330022" //  12
        "22003300220033333333002200330022"
        "22003300220033111133002200330022" //  14
        "22003300220033111133002200330022"
        "22003300220033111133002200330022" //  16
        "22003300220033111133002200330022"
        "22003300220033333333002200330022" //  18
        "22003300220033333333002200330022"
        "22003300220000000000002200330022" //  20
        "22003300220000000000002200330022"
        "22003300222222222222222200330022" //  22
        "22003300222222222222222200330022"
        "22003300000000000000000000330022" //  24
        "22003300000000000000000000330022"
        "22003333333333333333333333330022" //  26
        "22003333333333333333333333330022"
        "22000000000000000000000000000022" //  28
        "22000000000000000000000000000022"
        "22222222222222222222222222222222" //  30
        "22222222222222222222222222222222"
};      // _32x32_hw_cursor[]

char _64x64_hw_cursor[4096] =
{
//                1         2         3         4         5         6
//       0123456789012345678901234567890123456789012345678901234567890123
        "2222222222222222222222222222222222222222222222222222222222222222"
        "2222222222222222222222222222222222222222222222222222222222222222"
        "2200000000000000000000000000000000000000000000000000000000000022"
        "2200000000000000000000000000000000000000000000000000000000000022"
```

```
"2200333333333333333333333333333333333333333333333333333330022"
"2200333333333333333333333333333333333333333333333333333330022"
"2200330000000000000000000000000000000000000000000000000330022"
"2200330000000000000000000000000000000000000000000000000330022"
"2200330022222222222222222222222222222222222222222222200330022"
"2200330022222222222222222222222222222222222222222222200330022"
"2200330022000000000000000000000000000000000000000002200330022"
"2200330022000000000000000000000000000000000000000002200330022"
"2200330022003333333333333333333333333333333333333002200330022"
"2200330022003333333333333333333333333333333333333002200330022"
"2200330022003300000000000000000000000000000000033002200330022"
"2200330022003300000000000000000000000000000000033002200330022"
"2200330022003300222222222222222222222222222222003300220033002"
"2200330022003300222222222222222222222222222222003300220033002"
"2200330022003300220000000000000000000000000002200330022003300"
"2200330022003300220033333333333333333333333002200330022003300"
"2200330022003300220033333333333333333333333002200330022003300"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033002222222222222222003300220033002200330"
"2200330022003300220033002222222222222222003300220033002200330"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033333333333333333333333002200330022003300"
"2200330022003300220033333333333333333333333002200330022003300"
"2200330022003300220000000000000000000000002200330022003300"
"2200330022003300220000000000000000000000002200330022003300"
"2200330022003300220033333333002200330022003300220033002200330"
"2200330022003300220033333333002200330022003300220033002200330"
"2200330022003300220033111133002200330022003300220033002200330"
"2200330022003300220033111133002200330022003300220033002200330"
"2200330022003300220033111133002200330022003300220033002200330"
"2200330022003300220033333333002200330022003300220033002200330"
"2200330022003300220033333333002200330022003300220033002200330"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033002222222222222222003300220033002200330"
"2200330022003300220033002222222222222222003300220033002200330"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033000000000000000000033002200330022003300"
"2200330022003300220033333333333333333333333002200330022003300"
"2200330022003300220033333333333333333333333002200330022003300"
"2200330022003300220000000000000000000000002200330022003300"
"2200330022003300220000000000000000000000002200330022003300"
```



```
        "2200330022003300222222222222222222222222222222220033002200330022"
        "2200330022003300222222222222222222222222222222220033002200330022"
       "2200330022003300000000000000000000000000000000000033002200330022"
       "2200330022003300000000000000000000000000000000000033002200330022"
       "2200330022003333333333333333333333333333333333333333002200330022"
       "2200330022003333333333333333333333333333333333333333002200330022"
       "2200330022000000000000000000000000000000000000000000002200330022"
       "2200330022000000000000000000000000000000000000000000002200330022"
       "2200330022222222222222222222222222222222222222222222222200330022"
       "2200330022222222222222222222222222222222222222222222222200330022"
       "2200330000000000000000000000000000000000000000000000000000330022"
       "2200330000000000000000000000000000000000000000000000000000330022"
       "2200333333333333333333333333333333333333333333333333333333330022"
       "2200333333333333333333333333333333333333333333333333333333330022"
       "2200000000000000000000000000000000000000000000000000000000000022"
       "2200000000000000000000000000000000000000000000000000000000000022"
        "2222222222222222222222222222222222222222222222222222222222222222"
        "2222222222222222222222222222222222222222222222222222222222222222"
};      // _64x64_hw_cursor[]

// ************************************************************************
// INITIALIZE SINGLE PAGE 16K GRANULARITY MODE.
// ************************************************************************
void SP_16K_Init( void )
{
        unsigned char grb;

        outportb( 0x3ce, 0x0b );
        grb = inportb( 0x3cf );
        grb |= 0x20;              // GRB[5] <-- 1
        grb &= ~0x01;             // GRB[0] <-- 0
        outportb( 0x3cf, grb );
};      // SP_16K_Init()

// ************************************************************************
// SET GR9 OFFSET REGISTER IN SINGLE PAGE 16K GRANULARITY MODE
// ************************************************************************
unsigned int SP_16K_Set_Offset_Reg( unsigned long address )
{
        unsigned char gr9_value;
        unsigned int ret_val;
```

```
        ret_val = (unsigned int) (address & 0x0000ffffUL);

        address &= 0xffff0000UL;
        address >>= 16;
        gr9_value = (unsigned char) address;
        gr9_value <<= 2;
        outportb( 0x3ce, 0x09 );
        outportb( 0x3cf, gr9_value );
        return ret_val;
};      // SP_16K_Set()

// ************************************************************************
// INITIALIZE CURSOR PLANES 0 AND 1.
// ************************************************************************
void Cursor_Plane_Init( char* map, unsigned char* plane, int number_of_bytes )
{
        unsigned char   mask;
        int             bit, byte;
        int             i = 0;
        unsigned char*  plane_0 = &plane[0];
        unsigned char*  plane_1 = &plane[number_of_bytes];

        for ( byte = 0; byte < number_of_bytes; byte++ )
        {
          plane_0[byte] = 0;
          plane_1[byte] = 0;

          mask = 0x80;

          for ( bit = 0; bit < 8; bit++, mask >>= 1, i++ )
          {
            if      ( map[i]  == '1' )
            {
                plane_0[byte] |= mask;
            }
            else if ( map[i]  == '2' )
            {
                plane_1[byte] |= mask;
            }
            else if ( map[i]  == '3' )
            {
                plane_0[byte] |= mask;
```



```
                plane_1[byte] |= mask;
              }
            }
          }
};      // Cursor_Plane_Init()

// ***************************************************************************
// SET X, Y COORDINATE OF HARDWARE CURSOR.
// ***************************************************************************
void Cursor_Set_XY( int x, int y )
{
        x <<= 5;
        x |= 0x0010;
        outport( 0x3c4, x );

        y <<= 5;
        y |= 0x0011;
        outport( 0x3c4, y );
};      // Cursor_Set_XY()

// ***************************************************************************
// SET EXTENDED DAC COLORS FOR HARDWARE CURSOR.
// ***************************************************************************
void Cursor_Set_Color( unsigned long background, unsigned long foreground )
{
        unsigned char sr12;
        unsigned char b_red, b_green, b_blue;
        unsigned char f_red, f_green, f_blue;

        b_red   = (unsigned char) (background & 0x000000ffUL);
        b_green = (unsigned char) ((background & 0x0000ff00UL) >> 8);
        b_blue  = (unsigned char) ((background & 0x00ff0000UL) >> 16);
        f_red   = (unsigned char) (foreground & 0x000000ffUL);
        f_green = (unsigned char) ((foreground & 0x0000ff00UL) >> 8);
        f_blue  = (unsigned char) ((foreground & 0x00ff0000UL) >> 16);

        // SR12[1] : ENABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) | 0x82;   // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

        // WRITE BACKGROUND
```

```
        outportb( 0x3c8, 0x00 );
        outportb( 0x3c9, b_red );
        outportb( 0x3c9, b_green );
        outportb( 0x3c9, b_blue );

        // WRITE FOREGROUND
        outportb( 0x3c8, 0x0f );
        outportb( 0x3c9, f_red );
        outportb( 0x3c9, f_green );
        outportb( 0x3c9, f_blue );

        // SR12[1] : DISABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) & ~0x82;  // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE
}       // Cursor_Set_Color()

// ***************************************************************************
// ENABLE 64 X 64 HARDWARE CURSOR.
// ***************************************************************************
void _64x64_Cursor_Enable( void )
{
        unsigned char sr12;

        outportb( 0x3c4, 0x12 );
        sr12 = inportb( 0x3c5 ) | 0x05;
        outportb( 0x3c5, sr12 );

};      // _64x64_Cursor_Enable()

// ***************************************************************************
// DISABLE 64 X 64 HARDWARE CURSOR
// ***************************************************************************
void _64x64_Cursor_Disable( void )
{
        unsigned char sr12;

        outportb( 0x3c4, 0x12 );
        sr12 = inportb( 0x3c5 ) | 0x04;
        sr12 &= 0xfe;
        outportb( 0x3c5, sr12 );
```



```
};      // _64x64_Cursor_Disable()

// ************************************************************************
// SET PREVIOUSLY LOADED PATTERN FOR 64 X 64 HARDWARE CURSOR.
// ************************************************************************
void _64x64_Cursor_Set_Pattern( int pattern_number )
{
        unsigned char sr13 = (unsigned char) pattern_number << 2;

        outportb( 0x3c4, 0x13 );
        outportb( 0x3c5, sr13 );
};      // _64x64_Cursor_Set_Pattern()

// ************************************************************************
// LOAD 64X64 HARDWARE CURSOR PATTERN 0 IN 8 BPP MODE.
// LOAD 8 BYTES FROM PLANE 0 TO TO DISPLAY MEMORY,
// FOLLOWED BY 8 BYTES FROM PLANE 1 UNTIL THE PATTERN IS COPIED.
// ************************************************************************
void _64x64_Patterns_Load( unsigned char far* pattern )
{
        unsigned int  di, l, b;
      unsigned long pattern_address = 0x200000UL - 0x4000UL;  // 2 MB DISPLAY MEM
       unsigned char far* mem_ptr    = (unsigned char far*) MK_FP( 0xa000, 0 );
        unsigned char far* plane1_ptr = pattern;
        unsigned char far* plane2_ptr = pattern + 512;

        di = SP_16K_Set_Offset_Reg( pattern_address );

        for ( l = 0; l < 64; l++ )
        {
          for ( b = 0; b < 8; b++ ) mem_ptr[di++] = *plane1_ptr++;
          for ( b = 0; b < 8; b++ ) mem_ptr[di++] = *plane2_ptr++;
        }

};      // _64x64_Patterns_Load()

// ************************************************************************
// ENABLE 32 X 32 HARDWARE CURSOR.
// ************************************************************************
void _32x32_Cursor_Enable( void )
{
        unsigned char sr12;
```

```
        outportb( 0x3c4, 0x12 );
        sr12 = inportb( 0x3c5 ) | 0x01;
        sr12 &= ~0x04;
        outportb( 0x3c5, sr12 );

};      // _32x32_Cursor_Enable()

// ***************************************************************************
// DISABLE 32 X 32 HARDWARE CURSOR.
// ***************************************************************************
void _32x32_Cursor_Disable( void )
{
        unsigned char sr12;

        outportb( 0x3c4, 0x12 );
        sr12 = inportb( 0x3c5 );
        sr12 &= ~0x01;
        outportb( 0x3c5, sr12 );

};      // _32x32_Cursor_Disable()

// ***************************************************************************
// LOAD 32X32 HARDWARE CURSOR PATTERN 0 IN 8 BPP MODE.
// LOAD 8 BYTES FROM PLANE 0 TO TO DISPLAY MEMORY,
// FOLLOWED BY 8 BYTES FROM PLANE 1 UNTIL THE PATTERN IS COPIED.
// ***************************************************************************
void _32x32_Patterns_Load( unsigned char far* pattern )
{
        unsigned int  di, l;
      unsigned long pattern_address = 0x200000UL - 0x4000UL;  // 2 MB DISPLAY MEM
       unsigned char far* mem_ptr    = (unsigned char far*) MK_FP( 0xa000, 0 );
        unsigned char far* plane_ptr  = pattern;

        di = SP_16K_Set_Offset_Reg( pattern_address );

        for ( l = 0; l < 256; l++ ) mem_ptr[di++] = *plane_ptr++;

};      // _32x32_Patterns_Load()

// ***************************************************************************
// SET PREVIOUSLY LOADED PATTERN FOR 32 X 32 HARDWARE CURSOR.
```



```
// ***************************************************************************
void _32x32_Cursor_Set_Pattern( int pattern_number )
{
        unsigned char sr13 = (unsigned char) pattern_number;

        outportb( 0x3c4, 0x13 );
        outportb( 0x3c5, sr13 );
};      // _32x32_Cursor_Set_Pattern()

// ***************************************************************************
// INITIALIZE, LOAD, AND DISPLAY 32 X 32 AND 64 X 64 HARDWARE CURSORS.
// CHANGE COLORS AND LOCATION.
// ***************************************************************************
void main( void )
{
        asm   mov   ax, 0x005f
        asm   int   0x10            // SET MODE 5F -- 640X480X8 GRAPHICS MODE

        Cursor_Plane_Init( _64x64_hw_cursor, _64x64_planes_0_1, 512 );
        Cursor_Plane_Init( _32x32_hw_cursor, _32x32_planes_0_1, 128 );

        SP_16K_Init();                    // INITIALIZE GRAPHICS MEMORY MODE

        // INITIALIZE AND DISPLAY 64 X 64 HARDWARE CURSOR
        _64x64_Patterns_Load( _64x64_planes_0_1 );
        Cursor_Set_XY( 0, 0 );
        _64x64_Cursor_Enable();
        delay( 1000 );

        // CHANGE COLOR OF 64 X 64 HARDWARE CURSOR
        Cursor_Set_Color( 0x0000003fUL, 0x00003f00UL );
        delay( 1000 );
        Cursor_Set_Color( 0x00003f00UL, 0x003f0000UL );
        delay( 1000 );
        Cursor_Set_Color( 0x0000003fUL, 0x003f0000UL );
        delay( 1000 );

        // INITIALIZE AND DISPLAY 32 X 32 HARDWARE CURSOR
        _32x32_Patterns_Load( _32x32_planes_0_1 );
        Cursor_Set_XY( 0, 0 );
        _32x32_Cursor_Enable();
        delay( 1000 );
```

```
        // CHANGE LOCATION OF 32 X 32 HARDWARE CURSOR
        for ( int y = 0; y < 480; y++ )
        {
          Cursor_Set_XY( y, y );
          delay( 10 );
        }


        asm   mov   ax, 0x0003
        asm   int   0x10            // SET MODE 3 -- 80X25 TEXT MODE
};      // main()
```

9.14.3 PCI Configuration Register Access

```
// ************************************************************************
// PCI_XMPL.CPP
//
// THIS SAMPLE PROGRAM DEMONSTRATES THE MS-DOS INT 1AH INTERFACE TO
// THE PCI BUS CONFIGURATION REGISTERS.
//
// USE OF THE INT 1AH INTERFACE, WHICH IS PART OF THE PCI CHIPSET BIOS, IS
// HIGHLY RECOMMENDED, DUE TO POSSIBLE ALTERNATE IMPLEMENTATIONS
// IN PCI CHIPSETS.
// ************************************************************************
#pragma  option -4                // COMPILE WITH 486 EXTENSIONS
#pragma inline                    // PRODUCE ASSEMBLY MODULE BEFORE COMPILE
#include <stdio.h>

// GLOBAL VARIABLES
unsigned char bus_number;
unsigned char function_device;
unsigned char byte_config_regs[256];
unsigned int  word_config_regs[128];
unsigned long dword_config_regs[64];

// ************************************************************************
// THIS IS THE FIRST PCI BIOS SUBFUNCTION THAT SHOULD BE CALLED.
// IT DETERMINES WHETHER THE PCI BIOS IS PRESENT AND PROVIDES INFORMATION
// ABOUT THE REVISION OF THE PCI BIOS.
// ************************************************************************
void PCI_Get_BIOS_Present_Status( void )
```



```
{
        unsigned char ret_val, hw_mech, last_pci_bus;
        unsigned char major_version, minor_version;
        char p, c, i, q;
        unsigned int  present       = 1;
        unsigned int  bios_support  = 0;

        asm{
          mov   ah, 0b1h
          mov   al, 01h
          int   01ah                    // GET PCI BIOS PRESENT STATUS
          jnc   all
          mov   word ptr [present], 0
        }
all:
        asm{
          mov   [ret_val], ah           // SAVE PARAMETERS FOR PRINTING
          mov   [hw_mech], al
          mov   [major_version], bh
          mov   [minor_version], bl
          mov   [last_pci_bus],  cl

          mov   [p], dl                 // STORE EDX IN P, C, I, Q
          mov   [c], dh
          shr   edx, 16
          mov   [i], dl
          mov   [q], dh

        }

        printf( “Return Code       : %d\n”, int( ret_val ) );
        printf( “Hardware Mechanism: %x\n”, int( hw_mech ) );
        printf( “Major Version #   : %x\n”, int( major_version ) );
        printf( “Minor Version #   : %x\n”, int( minor_version ) );
        printf( “Last PCI bus      : %d\n”, int( last_pci_bus ) );
        printf( “Signature         : %c%c%c%c\n”, p, c, i, q );
        printf( “Services Present  : %s\n”, (present == 1) ? “Not Present” :
“Present” );

        // SEE IF EDX RETURNED ‘P’, ‘C’, & ‘I’
        if ( p == ‘P’ && c == ‘C’ && i == ‘I’ )
        {
```

```
          if ( ret_val == 0 || present == 0 ) bios_support = 1;
          else                                bios_support = 0;
        }
        printf( "PCI BIOS         : %s\n", (bios_support) ? "Suported" : "Not
Supported" );

};      // PCI_Get_BIOS_Present_Status()

// ************************************************************************
// THIS SUBFUNCTION RETURNS THE NEEDED PCI BIOS PARAMETERS:
//      DEVICE_NUMBER, BUS_NUMBER, FUNCTION_DEVICE, AND FUNCTION_NUMBER
// ************************************************************************
void PCI_Find_Device( void )
{
        unsigned char device_number, function_number, ret_val;
        unsigned int  vga_controller, pci_device_id, error = 0;

        // USE CIRRUS LOGIC EXTENDED VGA BIOS TO DETERMINE VGA CONTROLLER
        asm{
          mov   ah, 12h                // FUNCTION
          mov   bl, 80h                // SUBFUNCTION
          int   10h                    // VGA BIOS
          mov   [vga_controller], ax   // SAVE VGA_CONTROLLER ID
        }

        // ASSIGN PCI_DEVICE_ID, BASED ON EXTENDED VGA BIOS CALL
        switch ( vga_controller )
        {
          case 0x32 : pci_device_id = 0x00a0; break;
          case 0x31 :
          case 0x33 : pci_device_id = 0x00a8; break;
          case 0x36 : pci_device_id = 0x00ac; break;
          case 0x35 : pci_device_id = 0x00a0; break;
          default   : pci_device_id = 0x00a0; break;
        }

        // CALL PCI CHIPSET BIOS TO FIND DEVICE
        asm{
          mov   ah, 0xb1            // PCI BIOS FUNCTION
          mov   al, 0x02            // PCI BIOS SUBFUNCTION
          mov   cx, [pci_device_id] // PCI DEVICE ID
          mov   dx, 0x1013          // PCI VENDOR ID -- 01013H = CIRRUS LOGIC
```



```
          mov   si, 0
          int   0x1a
          jnc   FPD_all
          mov   word ptr [error], 1
        }
FPD_all:
        asm{
          mov   [ret_val], ah
          mov   [bus_number], bh      // SAVE BUS_NUMBER
          mov   cl, bl
          mov   [function_device], bl // THIS IS THE ONE TO SAVE
          and   bl, 0x07
          mov   [function_number], bl // BREAK DOWN THE FUNCTION/DEVICE FIELDS
          shr   cl, 3
          mov   [device_number], cl
        }

        // PRINT OUT RESULTS
        printf( "Find PCI Device   : " );
        if ( error == 1 )
        {
          printf( "Could not find device\n" );
        }
        else
        {
          switch ( ret_val )
          {
            case 0x00 : printf( "Successful\n" );       break;
            case 0x83 : printf( "Bad Vendor ID\n" );    break;
            case 0x86 : printf( "Device not found\n" ); break;
          }
          printf( "Bus number        : %x\n", int( bus_number ));
          printf( "Function Number   : %x\n", int( function_number ));
          printf( "Device Number     : %x\n", int( device_number ) );
        }
};      // PCI_Find_Device()

// ************************************************************************
// PERFORM 32 BIT READ OF PCI CONFIGURATION REGISTER.
// REGISTER MUST BE A MULTIPLE OF 4. ( 0, 4, 8, C, E, 10, 14 ... )
// ************************************************************************
unsigned long PCI_DWord_Read_Config_Reg( int reg )
```

```
{
        unsigned long ret_val;

        asm{
          mov   ah, 0xb1                  // BIOS FUNCTION : PCI
          mov   al, 0x0a                  // BIOS SUBFUNCTION : READ DWORD
          mov   bh, [bus_number]          // PCI BUS NUMBER
          mov   bl, [function_device]     // DEVICE[7:3] FUNCTION[2:0]
          mov   di, [reg]                 // REGISTER TO READ
          int   0x1a                      // VALUE COMES BACK IN ECX
          mov   [ret_val], ecx            // SAVE RETURN VALUE
        }

        return ret_val;
};      // PCI_DWord_Read_Config_Reg()

// ************************************************************************
// PERFORM 32 BIT WRITE OF PCI CONFIGURATION REGISTER.
// REGISTER MUST BE A MULTIPLE OF 4. ( 0, 4, 8, C, 10, 14 ... )
// ************************************************************************
void PCI_DWord_Write_Config_Reg( int reg, unsigned long value )
{
        asm{
          mov   ah, 0xb1                // BIOS FUNCTION : PCI
          mov   al, 0x0d                // BIOS SUBFUNCTION : WRITE DWORD
          mov   bh, [bus_number]        // PCI BUS NUMBER
          mov   bl, [function_device]   // DEVICE[7:3] FUNCTION[2:0]
          mov   di, [reg]               // REGISTER TO WRITE
          mov   ecx, [value]
          int   0x1a
        }
};      // PCI_DWord_Write_Config_Reg()

// ************************************************************************
// PERFORM 16 BIT READ OF PCI CONFIGURATION REGISTER.
// REGISTER MUST BE A MULTIPLE OF 2. ( 0, 2, 4, 6, 8, A, C, E, 10, 12 ... )
// ************************************************************************
unsigned int PCI_Word_Read_Config_Reg( int reg )
{
        unsigned int ret_val;

        asm{
```



```
          mov   ah, 0xb1                  // BIOS FUNCTION
          mov   al, 0x09                  // BIOS SUBFUNCTION READ WORD
          mov   bh, [bus_number]          // PCI BUS NUMBER
          mov   bl, [function_device]     // DEVICE[7:3] FUNCTION[2:0]
          mov   di, [reg]                 // REGISTER TO READ
          int   0x1a                      // VALUE COMES BACK IN ECX
          mov   [ret_val], cx            // SAVE RETURN VALUE
        }

        return ret_val;
};      // PCI_Word_Read_Config_Reg()

// ***************************************************************************
// PERFORM 16 BIT WRITE OF PCI CONFIGURATION REGISTER.
// REGISTER MUST BE A MULTIPLE OF 2. ( 0, 2, 4, 6, 8, A, C, E, 10, 12 ... )
// ***************************************************************************
void PCI_Word_Write_Config_Reg( int reg, unsigned int value )
{
        asm{
          mov   ah, 0xb1                // BIOS FUNCTION
          mov   al, 0x0c                // BIOS SUBFUNCTION
          mov   bh, [bus_number]        // PCI BUS NUMBER
          mov   bl, [function_device]   // DEVICE[7:3] FUNCTION[2:0]
          mov   di, [reg]               // REGISTER TO WRITE
          mov   cx, [value]
          int   0x1a
        }
};      // PCI_Word_Write_Config_Reg()

// ***************************************************************************
// PERFORM 8 BIT READ OF PCI CONFIGURATION REGISTER.
// ***************************************************************************
unsigned char PCI_Byte_Read_Config_Reg( int reg )
{
        unsigned char ret_val;

        asm{
          mov   ah, 0xb1                  // BIOS FUNCTION
          mov   al, 0x08                  // BIOS SUBFUNCTION
          mov   bh, [bus_number]          // PCI BUS NUMBER
          mov   bl, [function_device]     // DEVICE[7:3] FUNCTION[2:0]
          mov   di, [reg]                 // REGISTER TO READ
```

```
          int   0x1a                      // VALUE COMES BACK IN ECX
          mov   [ret_val], cl             // SAVE RETURN VALUE
        }

        return ret_val;
};      // PCI_Byte_Read_Config_Reg()

// ***************************************************************************
// PERFORM 8 BIT WRITE OF PCI CONFIGURATION REGISTER.
// ***************************************************************************
void PCI_Byte_Write_Config_Reg( int reg, unsigned char value )
{
        asm{
          mov   ah, 0xb1                // BIOS FUNCTION
          mov   al, 0x0b                // BIOS SUBFUNCTION
          mov   bh, [bus_number]        // PCI BUS NUMBER
          mov   bl, [function_device]   // DEVICE[7:3] FUNCTION[2:0]
          mov   di, [reg]               // REGISTER TO WRITE
          mov   cl, [value]
          int   0x1a
        }
};      // PCI_Byte_Write_Config_Reg()

// ***************************************************************************
// COMPARE ALL CONFIGURATION REGISTER READ ARRAYS WITH EACH OTHER AS DWORDS.
// IF THERE IS AN ERROR, THE ENTIRE DWORD FOR EACH EXCEPTION IS PRINTED OUT
// AS CHAR'S.
// ***************************************************************************
int Compare_DWord_Word_Byte_Arrays( void )
{
        int reg;
        unsigned long* dword_ptr = (unsigned long*) &dword_config_regs[0];
        unsigned long* word_ptr  = (unsigned long*) &word_config_regs[0];
        unsigned long* byte_ptr  = (unsigned long*) &byte_config_regs[0];

        // USE 64 DWORD COMPARES TO CHECK ALL ARRAYS
        for ( reg = 0; reg < 64; reg++ )
        {
          if (    dword_ptr[reg] != word_ptr[reg]
               || dword_ptr[reg] != byte_ptr[reg]
               || word_ptr[reg]  != byte_ptr[reg] )
          {
```



```
            unsigned char* p1;
            unsigned char* p2;
            unsigned char* p3;
            unsigned char* p4;

            printf( "ERROR: Values read do not match.\n" );

            // PRINT 1 DWORD AS BYTES
            p1 = (unsigned char*) &dword_ptr[reg];
            p2 = p1 + 1;
            p3 = p2 + 1;
            p4 = p3 + 1;
            printf( "\tDWORD[%03d]: %02x %02x %02x %02x\n", reg,
                    (unsigned int) *p1, (unsigned int) *p2,
                    (unsigned int) *p3, (unsigned int) *p4 );

            // PRINT 2 WORDS AS BYTES
            p1 = (unsigned char*) &word_ptr[reg*2];
            p2 = p1 + 1;
            p3 = p2 + 1;
            p4 = p3 + 1;
            printf( "\tWORD[%03d] : %02x %02x %02x %02x\n", reg*2,
                    (unsigned int) *p1, (unsigned int) *p2,
                    (unsigned int) *p3, (unsigned int) *p4 );

            // PRINT 4 BYTES AS BYTES
            p1 = (unsigned char*) &byte_ptr[reg*4];
            p2 = p1 + 1;
            p3 = p2 + 1;
            p4 = p3 + 1;
            printf( "\tBYTE[%03d] : %02x %02x %02x %02x\n", reg*4,
                    (unsigned int) *p1, (unsigned int) *p2,
                    (unsigned int) *p3, (unsigned int) *p4 );
          return 1;
          }
        }
        printf( "DWord, Word, and Byte reads compare equal\n" );
        return 0;
};      // Compare_DWord_Word_Byte_Arrays()

// ***************************************************************************
// ***************************************************************************
```

```
void main( void )
{
        int x;

        // INITIALIZE SOFTWARE INTERFACE TO PCI SERVICES
        PCI_Get_BIOS_Present_Status();
        PCI_Find_Device();

        // READ ALL REGISTERS AS DWORDS, WORDS, BYTES AND COMPARE
        for ( x = 0; x < 64; x++ )
        {
          dword_config_regs[x] = PCI_DWord_Read_Config_Reg( x*4 );
        }
        for ( x = 0; x < 128; x++ )
        {
          word_config_regs[x] = PCI_Word_Read_Config_Reg( x*2 );
        }
        for ( x = 0; x < 256; x++ )
        {
          byte_config_regs[x] = PCI_Byte_Read_Config_Reg( x );
        }

        Compare_DWord_Word_Byte_Arrays();

};      // main
```

9.14.4 BitBLT Examples

```
// ***************************************************************************
// BLT_XMPL.CPP
//
// EXAMPLE BITBLT ENGINE CODE.
// ***************************************************************************
#pragma option -4
#include <conio.h>
#include <stdlib.h>
#include <dos.h>

unsigned char far* status;
unsigned char rand_data[2048];
unsigned char ce_data[16] =
{
        0x38, 0x7c, 0x64, 0xc0, 0xc0, 0xc0, 0xc0, 0xc0,
```



```
        0xc0, 0xc0, 0xc0, 0x64, 0x7c, 0x38, 0x00, 0x00
};

unsigned int  blt_sys_2_scr_1[] =
{ 0, 0, 0, 0, 39, 39, 640, 640, 0, 0, 0, 0, 0x04, 0x0d, 0 };
unsigned int  blt_sys_2_scr_2[] =
{ 0, 0, 0, 0, 15, 0, 640, 640, 32000, 0, 0, 0, 0x04, 0x0d, 0 };

unsigned int  blt_scr_2_scr_1[] =
{ 0, 0, 0, 0, 39, 39, 640, 640, 100, 0, 0, 0, 0, 0x0d, 0 };
unsigned int  blt_scr_2_scr_2[] =
{ 0x55, 0, 0xaa, 0, 7, 15, 640, 640, 32100, 0, 32000, 0, 0x80, 0x0d, 0 };
unsigned int  blt_scr_2_scr_3[] =
{ 0x55, 0, 0xaa, 0, 7, 15, 640, 640, 32108, 0, 32000, 0, 0x80, 0x0d, 0 };
unsigned int  blt_scr_2_scr_4[] =
{ 0x55, 0, 0xaa, 0, 7, 15, 640, 640, 32116, 0, 32000, 0, 0x80, 0x0d, 0 };

unsigned int  io_blt_scr_2_scr[28] =
{
        0x0000, 0x0010, 0x0012, 0x0014, 0x0001, 0x0011, 0x0013, 0x0015,
        0x2720, 0x0021, 0x2722, 0x0023, 0x8024, 0x0225, 0x8026, 0x0227,
        0x0028, 0x0129, 0x002a, 0x002c, 0x002d, 0x002e, 0x002f, 0x0030,
        0x0d32, 0x0033, 0x0034, 0x0035
};

// ************************************************************************
// WAIT FOR BLT TO FINISH.
// ************************************************************************
void BLT_Wait( void )
{
        while(1)
        {
          // CHECK BITBLT STATUS
          if ( (*status & 0x01 ) == 0x00 ) break;
        }
};      // BLT_Wait()

// ************************************************************************
// WAIT FOR BLT TO FINISH USING IO.
// ************************************************************************
void IO_BLT_Wait( void )
{
```

```
        outport( 0x3ce, 0x31 );

        while(1)
        {
          // CHECK BITBLT STATUS
          if ( ( inport( 0x3cf ) & 0x01 ) == 0x00 ) break;
        }
};      // IO_BLT_Wait()

// ************************************************************************
// SYSTEM TO SCREEN BLT USING MEMIO.
// ************************************************************************
void BLT_System_To_Screen(  unsigned long* blt_data,
                            unsigned long* scr_data, unsigned long count )
{
        unsigned int        x;
        unsigned long far*  bptr  = (unsigned long far*) MK_FP( 0xb800, 0 );
        unsigned long far*  aptr  = (unsigned long far*) MK_FP( 0xa000, 0 );

        BLT_Wait();

        // WRITE BLT PARAMETERS
        for ( x = 0; x < 8; x++ ) *bptr++ = *blt_data++;

        // START BITBLT
        *status = 0x02;

        // WRITE SYSTEM DATA INTO FRAME BUFFER
        for ( x = 0; x < count; x++ ) *aptr++ = *scr_data++;

};      // BLT_System_To_Screen

// ************************************************************************
// SCREEN TO SCREEN BLT USING MEMIO.
// ************************************************************************
void BLT_Screen_To_Screen(  unsigned long* blt_data )
{
        unsigned int        x;
        unsigned long far*  bptr  = (unsigned long far*) MK_FP( 0xb800, 0 );
       unsigned char far*  start = (unsigned char far*) MK_FP( 0xb800, 0x40 );

        BLT_Wait();
```



```
        // WRITE BLT PARAMETERS
        for ( x = 0; x < 8; x++ ) *bptr++ = *blt_data++;

        // START BITBLT
        *start = 0x02;

};      // BLT_Screen_To_Screen()

// ***************************************************************************
// SCREEN TO SCREEN BLT USING IO.
// ***************************************************************************
void IO_BLT_Screen_To_Screen(  unsigned int* blt_data )
{
        IO_BLT_Wait();

        // WRITE BLT PARAMETERS
        for ( int x = 0; x < 28; x++ ) outport( 0x3ce, blt_data[x] );

        // START BITBLT
        outport( 0x3ce, 0x0231 );

};      // IO_BLT_Screen_To_Screen()

// ***************************************************************************
// DO SEVERAL BLTS.
// ***************************************************************************
void main( void )
{
        unsigned char sr17, grb;
        int x;

        // SET VIDEO MODE 5F 640X480X8 GRAPHICS
        asm   mov   ax, 0x5f
        asm   int   0x10

        status = (unsigned char far*) MK_FP( 0xb800, 0x40 );

        outportb( 0x3c4, 0x17 );
        sr17 = inportb( 0x3c5 ) | 0x04;   // TURN ON SR17[2] MEMIO
        sr17 &= ~0x40;                    // TURN OFF SR17[6] MEMIO ADDRESS
        outportb( 0x3c5, sr17 );
```

```
        outportb( 0x3ce, 0x0b );
        grb = inportb( 0x3cf );
        grb |= 0x20;              // GRB[5] <-- 1
        grb &= ~0x01;             // GRB[0] <-- 0
        outportb( 0x3cf, grb );

        for ( x = 0; x < 2048; x++ ) rand_data[x] = random( 0xff );

        // SYSTEM TO SCREEN BLT WITH RANDOM COLOR DATA
        BLT_System_To_Screen( (unsigned long*) blt_sys_2_scr_1,
                              (unsigned long*) rand_data, 400 );

        // SCREEN TO SCREEN BLT WITH SOURCE EQUAL TO FIRST BLT
        BLT_Screen_To_Screen( (unsigned long*) blt_scr_2_scr_1 );

        // SCREEN TO SCREEN IO BLT WITH SOURCE EQUAL TO FIRST BLT
        IO_BLT_Screen_To_Screen( io_blt_scr_2_scr );

        // SYSTEM TO SCREEN BLT WITH COLOR EXPAND DATA
        BLT_System_To_Screen( (unsigned long*) blt_sys_2_scr_2,
                              (unsigned long*) ce_data, 4 );

        // SCREEN TO SCREEN COLOR EXPANDED BLT
        BLT_Screen_To_Screen( (unsigned long*) blt_scr_2_scr_2 );
        BLT_Screen_To_Screen( (unsigned long*) blt_scr_2_scr_3 );
        BLT_Screen_To_Screen( (unsigned long*) blt_scr_2_scr_4 );

        getch();

        // SET VIDEO MODE 3 80X25 TEXT
        asm   mov   ax, 0x03
        asm   int   0x10
};      // main()
```

9.14.5 Setting Mode 3

```
// ***************************************************************************
// MS3_XMPL.CPP
//
// EXAMPLE PROGRAM TO DEMONSTRATE SETTING VGA CONTROLLER FOR TEXT MODE.
// ***************************************************************************
#include <dos.h>
```



```
#include <mem.h>

extern char *font;                  // POINTER TO 8X16 FONT

struct VIDEO_REGS
{
   char seq[8];                             // SEQUENCER REGS
   char crtc[26];                // CRTC REGS
   char grph[9];                 // GRAPHICS REGS
   char attr[21];                // ATTRIBUTE REGS
};

struct VIDEO_REGS mode3 =
{
             { 0x03, 0x00, 0x03, 0x00, 0x02, 0xfa, 0x01, 0xfa      }, // SR REGS
             { 0x5f, 0x4f, 0x50, 0x82, 0x55, 0x81, 0xbf, 0x1f,        // CR REGS
               0x00, 0x4f, 0x0d, 0x0e, 0x00, 0x00, 0x01, 0xe0,
               0x9c, 0xae, 0x8f, 0x28, 0x1f, 0x96, 0xb9, 0xa3,
               0xff, 0x19                                            },
             { 0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x0e, 0x00, 0xff  }, // GR
REGS
             { 0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x14,             // AR REGS
               0x07, 0x38, 0x39, 0x3a, 0x3b, 0x3c, 0x3d,
               0x3e, 0x3f, 0x0c, 0x00, 0x0f, 0x08, 0x00              },
};

char attrchar[2] = { 0x20, 0x07 };  // FOR CLEARING THE VIDEO BUFFER
int  pattern[4]  = { 0xaaaa, 0x5555, 0xffff, 0x0000 }; // TEST PATTERN
char savearea[28*80*2];             // FOR SAVING CHARACTER TO BE TESTED

char ramdac_tbl[]=
{
0x00,0x00,0x00,0x00,0x00,0x2a,0x00,0x2a,0x00,0x00,0x2a,0x2a,0x2a,0x00,0x00,
0x2a,0x00,0x2a,0x2a,0x2a,0x00,0x2a,0x2a,0x2a,0x00,0x00,0x15,0x00,0x00,0x3f,
0x00,0x2a,0x15,0x00,0x2a,0x3f,0x2a,0x00,0x15,0x2a,0x00,0x3f,0x2a,0x2a,0x15,
0x2a,0x2a,0x3f,0x00,0x15,0x00,0x00,0x15,0x2a,0x00,0x3f,0x00,0x00,0x3f,0x2a,
0x2a,0x15,0x00,0x2a,0x15,0x2a,0x2a,0x3f,0x00,0x2a,0x3f,0x2a,0x00,0x15,0x15,
0x00,0x15,0x3f,0x00,0x3f,0x15,0x00,0x3f,0x3f,0x2a,0x15,0x15,0x2a,0x15,0x3f,
0x2a,0x3f,0x15,0x2a,0x3f,0x3f,0x15,0x00,0x00,0x15,0x00,0x2a,0x15,0x2a,0x00,
0x15,0x2a,0x2a,0x3f,0x00,0x00,0x3f,0x00,0x2a,0x3f,0x2a,0x00,0x3f,0x2a,0x2a,
0x15,0x00,0x15,0x15,0x00,0x3f,0x15,0x2a,0x15,0x15,0x2a,0x3f,0x3f,0x00,0x15,
0x3f,0x00,0x3f,0x3f,0x2a,0x15,0x3f,0x2a,0x3f,0x15,0x15,0x00,0x15,0x15,0x2a,
```

```
0x15,0x3f,0x00,0x15,0x3f,0x2a,0x3f,0x15,0x00,0x3f,0x15,0x2a,0x3f,0x3f,0x00,
0x3f,0x3f,0x2a,0x15,0x15,0x15,0x15,0x15,0x3f,0x15,0x3f,0x15,0x15,0x3f,0x3f,
0x3f,0x15,0x15,0x3f,0x15,0x3f,0x3f,0x3f,0x15,0x3f,0x3f,0x3f,0x39,0x0c,0x05,
0x15,0x2c,0x0f,0x26,0x10,0x3d,0x29,0x29,0x38,0x04,0x1a,0x0e,0x02,0x1e,0x3a,
0x3c,0x25,0x33,0x3c,0x0c,0x2c,0x3f,0x03,0x2b,0x1c,0x09,0x13,0x25,0x2a,0x35,
0x1e,0x0a,0x38,0x24,0x08,0x03,0x03,0x0e,0x36,0x0c,0x06,0x2a,0x26,0x03,0x32,
0x05,0x2f,0x33,0x3c,0x35,0x2f,0x2d,0x26,0x3e,0x0d,0x0a,0x10,0x25,0x3c,0x11,
0x0d,0x04,0x2e,0x05,0x19,0x3e,0x0c,0x13,0x34,0x2b,0x06,0x24,0x04,0x03,0x0d,
0x2f,0x3c,0x0c,0x2a,0x37,0x1f,0x0f,0x12,0x38,0x38,0x0e,0x2a,0x12,0x2f,0x19,
0x29,0x2e,0x31,0x25,0x13,0x3e,0x33,0x3e,0x33,0x1d,0x2c,0x25,0x15,0x15,0x05,
0x32,0x25,0x39,0x1a,0x07,0x1f,0x13,0x0e,0x1d,0x36,0x17,0x34,0x0f,0x15,0x23,
0x02,0x35,0x0d,0x15,0x3f,0x0c,0x14,0x2f,0x0f,0x19,0x21,0x3e,0x27,0x11,0x2f,
0x38,0x3f,0x3c,0x36,0x2d,0x15,0x16,0x17,0x02,0x01,0x0a,0x3d,0x1b,0x11,0x3f,
0x21,0x3c,0x0d,0x1a,0x39,0x3d,0x08,0x0e,0x0e,0x22,0x21,0x23,0x1e,0x30,0x05,
0x1f,0x22,0x3d,0x1e,0x2f,0x0a,0x00,0x1c,0x0e,0x00,0x1c,0x15,0x00,0x1c,0x1c,
0x00,0x15,0x1c,0x00,0x0e,0x1c,0x00,0x07,0x1c,0x0e,0x0e,0x1c,0x11,0x0e,0x1c,
0x15,0x0e,0x1c,0x18,0x0e,0x1c,0x1c,0x0e,0x1c,0x1c,0x0e,0x18,0x1c,0x0e,0x15,
0x1c,0x0e,0x11,0x1c,0x0e,0x0e,0x1c,0x11,0x0e,0x1c,0x15,0x0e,0x1c,0x18,0x0e,
0x1c,0x1c,0x0e,0x18,0x1c,0x0e,0x15,0x1c,0x0e,0x11,0x1c,0x0e,0x0e,0x1c,0x0e,
0x0e,0x1c,0x11,0x0e,0x1c,0x15,0x0e,0x1c,0x18,0x0e,0x1c,0x1c,0x0e,0x18,0x1c,
0x0e,0x15,0x1c,0x0e,0x11,0x1c,0x14,0x14,0x1c,0x16,0x14,0x1c,0x18,0x14,0x1c,
0x1a,0x14,0x1c,0x1c,0x14,0x1c,0x1c,0x14,0x1a,0x1c,0x14,0x18,0x1c,0x14,0x16,
0x1c,0x14,0x14,0x1c,0x16,0x14,0x1c,0x18,0x14,0x1c,0x1a,0x14,0x1c,0x1c,0x14,
0x1a,0x1c,0x14,0x18,0x1c,0x14,0x16,0x1c,0x14,0x14,0x1c,0x14,0x14,0x1c,0x16,
0x14,0x1c,0x18,0x14,0x1c,0x1a,0x14,0x1c,0x1c,0x14,0x1a,0x1c,0x14,0x18,0x1c,
0x14,0x16,0x1c,0x00,0x00,0x10,0x04,0x00,0x10,0x08,0x00,0x10,0x0c,0x00,0x10,
0x10,0x00,0x10,0x10,0x00,0x0c,0x10,0x00,0x08,0x10,0x00,0x04,0x10,0x00,0x00,
0x10,0x04,0x00,0x10,0x08,0x00,0x10,0x0c,0x00,0x10,0x10,0x00,0x0c,0x10,0x00,
0x08,0x10,0x00,0x04,0x10,0x00,0x00,0x10,0x00,0x00,0x10,0x04,0x00,0x10,0x08,
0x00,0x10,0x0c,0x00,0x10,0x10,0x00,0x0c,0x10,0x00,0x08,0x10,0x00,0x04,0x10,
0x08,0x08,0x10,0x0a,0x08,0x10,0x0c,0x08,0x10,0x0e,0x08,0x10,0x10,0x08,0x10,
0x10,0x08,0x0e,0x10,0x08,0x0c,0x10,0x08,0x0a,0x10,0x08,0x08,0x10,0x0a,0x08,
0x10,0x0c,0x08,0x10,0x0e,0x08,0x10,0x10,0x08,0x0e,0x10,0x08,0x0c,0x10,0x08,
0x0a,0x10,0x08,0x08,0x10,0x08,0x08,0x10,0x0a,0x08,0x10,0x0c,0x08,0x10,0x0e,
0x08,0x10,0x10,0x08,0x0e,0x10,0x08,0x0c,0x10,0x08,0x0a,0x10,0x0b,0x0b,0x10,
0x0c,0x0b,0x10,0x0d,0x0b,0x10,0x0f,0x0b,0x10,0x10,0x0b,0x10,0x10,0x0b,0x0f,
0x10,0x0b,0x0d,0x10,0x0b,0x0c,0x10,0x0b,0x0b,0x10,0x0c,0x0b,0x10,0x0d,0x0b,
0x10,0x0f,0x0b,0x10,0x10,0x0b,0x0f,0x10,0x0b,0x0d,0x10,0x0b,0x0c,0x10,0x0b,
0x0b,0x10,0x0b,0x0b,0x10,0x0c,0x0b,0x10,0x0d,0x0b,0x10,0x0f,0x0b,0x10,0x10,
0x0b,0x0f,0x10,0x0b,0x0d,0x10,0x0b,0x0c,0x10,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00
```



```
};      // ramdac_tbl[]

// ************************************************************************
// WRITE INDIRECTLY TO I/O PORT
// ************************************************************************
void setreg( unsigned int port, int index, char data )
{
             outportb( port, index);
             outportb( port+1, data );
};      // setreg()

// ************************************************************************
// READ I/O PORT
// ************************************************************************
int getreg(unsigned int port, int index)
{
             outportb( port, index );
             return inportb( port+1 ) & 0x00ff;
};      // getreg()

// ************************************************************************
// ************************************************************************
void updattr( int rg, char val )
{
             inportb( 0x3da );    // POINTS TO INDEX REGISTER FOR COLOR ADAPTER
             inportb( 0x3ba );     // POINTS TO INDEX REGISTER FOR MONO
             outportb( 0x3c0, rg );
             outportb( 0x3c0, val );
             outportb( 0x3c0, rg | 0x20 );
};      // updattr()

// ************************************************************************
// SET MOST OF VGA REGISTERS
// ************************************************************************
void setvidregs( struct VIDEO_REGS* d)
{

             int i;

             // SET ATTRIBUTE CONTROLLER
             for ( i = 0; i < 0x15; i++ ) updattr( i, d->attr[i] );
```

```
        // SET CRTC REGISTERS
             for ( i = 0; i < 0x19; i++ ) setreg( 0x3d4, i, d->crtc[i] );

             // SET GRAPHICS REGISTERS
             for ( i = 0x00;i < 0x09; i++ ) setreg( 0x3ce, i, d->grph[i] );

             // SET SEQUENCER REGISTERS
             setreg(0x3c4,0x00,0x03);      // RESTART SEQUENCER
             for ( i = 0; i < 5; i++ )     setreg( 0x3c4, i, d->seq[i] );

};      // setvidregs()

// ***************************************************************************
// ***************************************************************************
int ramtest( void )
{
        int   stat;
        int   p, ix;
        char  l, i;
             void far *address;
             unsigned bufseg, bufoff;

        // PUT CHIP INTO THE PLANAR GRAPHICS MODE FOR TESTING
        // DISABLE VIDEO AND ENABLE ALL TO CPU TO ENABLE MAXIMUM  VIDEO
        // MEMORY ACCESS

             i = getreg( 0x3c4, 0x01 ) | 0x20;// CLOCKING MODE REGISTER - */
             setreg( 0x3c4, 0x01, i );      // DISABLE VIDEO

             i = ( getreg( 0x3c4, 0x04 ) | 0x04 ) & 0x07; // MEMORY MODE REG
             setreg( 0x3c4, 0x04, i );      // DISABLE ODD/EVEN AND CHAIN 4

             i = getreg( 0x3ce, 0x05) & 0xef;// GRAPHICS CONTROLLER
             setreg( 0x3ce, 0x05, i);      // DISABLE ODD/EVEN

             setreg( 0x3ce, 0x06, 0x05 );     // MEMORY MAP - SET IT TO A000

                         // AND GRAPHICS MODE

             setreg( 0x3c4, 2, 0x0f );      // ENABLE WRITE PLANES 0-3

        // DO 4 DIFFERENT PATTERNS
```



```
                for( p = 0; p < 4; p++ )
                {
                  // FILL ALL 4 PLANES WITH TEST PATTERN
                  address = (void far *)( pattern + p );
                  bufseg  = FP_SEG( address );
                  bufoff  = FP_OFF( address );
                  for ( ix = 0; ix < 25*80*2; ix += 2 )
              {
                movedata( bufseg, bufoff, 0xa000, ix, 2);
              }

                  // NOW TEST ALL 4 PLANES FOR THE PATTERN
                  for( l = 0; l < 4; l++)
                  {
                    setreg( 0x3ce, 4, l );              // SELECT PLANE TO READ

                    address  = (void far *)savearea;
                    bufseg   = FP_SEG( address );
                    bufoff   = FP_OFF( address );
                    movedata( 0xa000, 0, bufseg, bufoff, 25*80*2 );

                    stat  = 0;
              ix    = 0;
                    while ( ( stat == 0 ) && ( ix <= 25*80 ) )
              {
                stat = memcmp( &(char)pattern[p], &savearea[ix++], 2 );
              }
              if( stat ) return stat;               // RETURN AN ERROR
                 }
                }
          return stat;
};      // ramtest()

// ***************************************************************************
// ***************************************************************************
int main( void )
{
              char      i;
              int       ram_status, ix, j;
              int       font_ptr;
              void far* address;
              unsigned  bufseg, bufoff;
```

```
    // ASSERT SYNCHRONEOUS RESET WHILE SETTING THE CLOCK MODE
    setreg( 0x3c4, 0, 1);               // ASSERT SYNCHRONOUS RESET
    outportb( 0x3c2, 0x67 );            // SELECT CLOCK
    setreg( 0x3c4, 0, 3);              // DE-ASSERT SYNCHRONOUS RESET

    setvidregs( &mode3 );            // SET "STANDARD" VIDEO REGISTERS

    ram_status = ramtest();             // DO THE VIDEO MEMORY TEST

    setvidregs( &mode3 );            // SET "STANDARD" VIDEO REGISTERS

    // LOAD 8X16 FONT INTO PLANE 2
    i = ( getreg( 0x3c4, 0x04 ) | 0x04 ) & 0x07;// MEMORY MODE REG
    setreg( 0x3c4, 0x04, i );        // DISABLE ODD/EVEN AND CHAIN 4

    i = getreg( 0x3c4, 0x01 ) | 0x20;  // CLOCKING MODE REGISTER
    setreg( 0x3c4, 0x01, i);        // DISABLE VIDEO

    i = getreg( 0x3ce, 0x05 ) & 0xef;  // GRAPHICS CONTROLLER
    setreg( 0x3ce, 0x05, i );        // DISABLE ODD/EVEN

    setreg( 0x3ce, 0x06, 0x05);       // MEMORY MAP - SET IT TO A000

                  // AND GRAPHICS MODE

    setreg( 0x3c4, 2, 4 );            // ENABLE WRITE PLANE 2

    // FILL PLANE 2 WITH 8X16 FONT
    address   = (void far *)&font;
    bufseg    = FP_SEG( address );
    bufoff    = FP_OFF( address );
    font_ptr  = 0;
    for ( ix = 0; ix <= 255; ix++ )
    {
// MOVE 1ST 16 BYTES
      movedata( bufseg, bufoff, 0xA000, font_ptr, 16 );
      font_ptr += 32;
      bufoff   += 16;
    }

    // LOAD RAMDAC
```



```
                for ( ix = 0, j = 0; j <= 0x0FF ; ix = ix+3, j++ )
                {
                  outportb( 0x3c8, (unsigned char) j );     // WRITE RAMDAC INDEX
                  outportb( 0x3c9, ramdac_tbl[ix]    );     // WRITE RED
                  outportb( 0x3c9, ramdac_tbl[ix+1]  );     // WRITE GREEN
                  outportb( 0x3c9, ramdac_tbl[ix+2]  );     // WRITE BLUE
                };

                // SET “STANDARD” VIDEO REGISTERS
                setvidregs( &mode3 );

                // FILL PLANE 0 AND 1 WITH 0X20 AND 0X07
                address = (void far *) attrchar;
                bufseg  = FP_SEG( address );
                bufoff  = FP_OFF( address );
                for ( ix = 0;ix < 25*80*2; ix += 2 )
        {
                  movedata( bufseg, bufoff, 0xb800, ix, sizeof( attrchar ) );
        }

                // PROGRAM THE PALLETTE MASK REGISTER
                outportb( 0x3c6, 0xff );

                // GET MONITOR TYPE
        return ram_status;

};      // main()



;
; FONT 8x16
;
                PUBLIC C font
DGROUP          GROUP       _DATA
                ASSUME DS:DGROUP
_DATA segment word public ‘DATA’

font            label byte
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;0
DB 000H,000H,07EH,081H,0A5H,081H,081H,0BDH
```

```
DB 099H,081H,081H,07EH,000H,000H,000H,000H ;1
DB 000H,000H,07EH,0FFH,0DBH,0FFH,0FFH,0C3H
DB 0E7H,0FFH,0FFH,07EH,000H,000H,000H,000H ;2
DB 000H,000H,000H,000H,06CH,0FEH,0FEH,0FEH
DB 0FEH,07CH,038H,010H,000H,000H,000H,000H ;3
DB 000H,000H,000H,000H,010H,038H,07CH,0FEH
DB 07CH,038H,010H,000H,000H,000H,000H,000H ;4
DB 000H,000H,000H,018H,03CH,03CH,0E7H,0E7H
DB 0E7H,099H,018H,03CH,000H,000H,000H,000H ;5
DB 000H,000H,000H,018H,03CH,07EH,0FFH,0FFH
DB 07EH,018H,018H,03CH,000H,000H,000H,000H ;6
DB 000H,000H,000H,000H,000H,000H,018H,03CH
DB 03CH,018H,000H,000H,000H,000H,000H,000H ;7
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0E7H,0C3H
DB 0C3H,0E7H,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH ;8
DB 000H,000H,000H,000H,000H,03CH,066H,042H
DB 042H,066H,03CH,000H,000H,000H,000H,000H ;9
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0C3H,099H,0BDH
DB 0BDH,099H,0C3H,0FFH,0FFH,0FFH,0FFH,0FFH ;10
DB 000H,000H,01EH,00EH,01AH,032H,078H,0CCH
DB 0CCH,0CCH,0CCH,078H,000H,000H,000H,000H ;11
DB 000H,000H,03CH,066H,066H,066H,066H,03CH
DB 018H,07EH,018H,018H,000H,000H,000H,000H ;12
DB 000H,000H,03FH,033H,03FH,030H,030H,030H
DB 030H,070H,0F0H,0E0H,000H,000H,000H,000H ;13
DB 000H,000H,07FH,063H,07FH,063H,063H,063H
DB 063H,067H,0E7H,0E6H,0C0H,000H,000H,000H ;14
DB 000H,000H,000H,018H,018H,0DBH,03CH,0E7H
DB 03CH,0DBH,018H,018H,000H,000H,000H,000H ;15
DB 000H,080H,0C0H,0E0H,0F0H,0F8H,0FEH,0F8H
DB 0F0H,0E0H,0C0H,080H,000H,000H,000H,000H ;16
DB 000H,002H,006H,00EH,01EH,03EH,0FEH,03EH
DB 01EH,00EH,006H,002H,000H,000H,000H,000H ;17
DB 000H,000H,018H,03CH,07EH,018H,018H,018H
DB 018H,07EH,03CH,018H,000H,000H,000H,000H ;18
DB 000H,000H,066H,066H,066H,066H,066H,066H
DB 066H,000H,066H,066H,000H,000H,000H,000H ;19
DB 000H,000H,07FH,0DBH,0DBH,0DBH,07BH,01BH
DB 01BH,01BH,01BH,01BH,000H,000H,000H,000H ;20
DB 000H,07CH,0C6H,060H,038H,06CH,0C6H,0C6H
DB 06CH,038H,00CH,0C6H,07CH,000H,000H,000H ;21
DB 000H,000H,000H,000H,000H,000H,000H,000H
```



```
DB 0FEH,0FEH,0FEH,0FEH,000H,000H,000H,000H ;22
DB 000H,000H,018H,03CH,07EH,018H,018H,018H
DB 018H,07EH,03CH,018H,07EH,000H,000H,000H ;23
DB 000H,000H,018H,03CH,07EH,018H,018H,018H
DB 018H,018H,018H,018H,000H,000H,000H,000H ;24
DB 000H,000H,018H,018H,018H,018H,018H,018H
DB 018H,07EH,03CH,018H,000H,000H,000H,000H ;25
DB 000H,000H,000H,000H,000H,018H,00CH,0FEH
DB 00CH,018H,000H,000H,000H,000H,000H,000H ;26
DB 000H,000H,000H,000H,000H,030H,060H,0FEH
DB 060H,030H,000H,000H,000H,000H,000H,000H ;27
DB 000H,000H,000H,000H,000H,0C0H,0C0H,0C0H
DB 0C0H,0FEH,000H,000H,000H,000H,000H,000H ;28
DB 000H,000H,000H,000H,000H,028H,06CH,0FEH
DB 06CH,028H,000H,000H,000H,000H,000H,000H ;29
DB 000H,000H,000H,000H,010H,038H,038H,07CH
DB 07CH,0FEH,0FEH,000H,000H,000H,000H,000H ;30
DB 000H,000H,000H,000H,0FEH,0FEH,07CH,07CH
DB 038H,038H,010H,000H,000H,000H,000H,000H ;31
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;32
DB 000H,000H,018H,03CH,03CH,03CH,018H,018H
DB 018H,000H,018H,018H,000H,000H,000H,000H ;33
DB 000H,066H,066H,066H,024H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;34
DB 000H,000H,000H,06CH,06CH,0FEH,06CH,06CH
DB 06CH,0FEH,06CH,06CH,000H,000H,000H,000H ;35
DB 018H,018H,07CH,0C6H,0C2H,0C0H,07CH,006H
DB 086H,0C6H,07CH,018H,018H,000H,000H,000H ;36
DB 000H,000H,000H,000H,0C2H,0C6H,00CH,018H
DB 030H,060H,0C6H,086H,000H,000H,000H,000H ;37
DB 000H,000H,038H,06CH,06CH,038H,076H,0DCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;38
DB 000H,030H,030H,030H,060H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;39
DB 000H,000H,00CH,018H,030H,030H,030H,030H
DB 030H,030H,018H,00CH,000H,000H,000H,000H ;40
DB 000H,000H,030H,018H,00CH,00CH,00CH,00CH
DB 00CH,00CH,018H,030H,000H,000H,000H,000H ;41
DB 000H,000H,000H,000H,000H,066H,03CH,0FFH
DB 03CH,066H,000H,000H,000H,000H,000H,000H ;42
DB 000H,000H,000H,000H,000H,018H,018H,07EH
```

```
DB 018H,018H,000H,000H,000H,000H,000H,000H ;43
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,018H,018H,018H,030H,000H,000H,000H ;44
DB 000H,000H,000H,000H,000H,000H,000H,0FEH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;45
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,018H,018H,000H,000H,000H,000H ;46
DB 000H,000H,000H,000H,002H,006H,00CH,018H
DB 030H,060H,0C0H,080H,000H,000H,000H,000H ;47
DB 000H,000H,03CH,066H,0C3H,0C3H,0DBH,0DBH
DB 0C3H,0C3H,066H,03CH,000H,000H,000H,000H ;48
DB 000H,000H,018H,038H,078H,018H,018H,018H
DB 018H,018H,018H,07EH,000H,000H,000H,000H ;49
DB 000H,000H,07CH,0C6H,006H,00CH,018H,030H
DB 060H,0C0H,0C6H,0FEH,000H,000H,000H,000H ;50
DB 000H,000H,07CH,0C6H,006H,006H,03CH,006H
DB 006H,006H,0C6H,07CH,000H,000H,000H,000H ;51
DB 000H,000H,00CH,01CH,03CH,06CH,0CCH,0FEH
DB 00CH,00CH,00CH,01EH,000H,000H,000H,000H ;52
DB 000H,000H,0FEH,0C0H,0C0H,0C0H,0FCH,006H
DB 006H,006H,0C6H,07CH,000H,000H,000H,000H ;53
DB 000H,000H,038H,060H,0C0H,0C0H,0FCH,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;54
DB 000H,000H,0FEH,0C6H,006H,006H,00CH,018H
DB 030H,030H,030H,030H,000H,000H,000H,000H ;55
DB 000H,000H,07CH,0C6H,0C6H,0C6H,07CH,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;56
DB 000H,000H,07CH,0C6H,0C6H,0C6H,07EH,006H
DB 006H,006H,00CH,078H,000H,000H,000H,000H ;57
DB 000H,000H,000H,000H,018H,018H,000H,000H
DB 000H,018H,018H,000H,000H,000H,000H,000H ;58
DB 000H,000H,000H,000H,018H,018H,000H,000H
DB 000H,018H,018H,030H,000H,000H,000H,000H ;59
DB 000H,000H,000H,006H,00CH,018H,030H,060H
DB 030H,018H,00CH,006H,000H,000H,000H,000H ;60
DB 000H,000H,000H,000H,000H,000H,0FEH,000H
DB 000H,0FEH,000H,000H,000H,000H,000H,000H ;61
DB 000H,000H,000H,060H,030H,018H,00CH,006H
DB 00CH,018H,030H,060H,000H,000H,000H,000H ;62
DB 000H,000H,07CH,0C6H,0C6H,00CH,018H,018H
DB 018H,000H,018H,018H,000H,000H,000H,000H ;63
DB 000H,000H,000H,07CH,0C6H,0C6H,0DEH,0DEH
```



```
DB 0DEH,0DCH,0C0H,07CH,000H,000H,000H,000H ;64
DB 000H,000H,010H,038H,06CH,0C6H,0C6H,0FEH
DB 0C6H,0C6H,0C6H,0C6H,000H,000H,000H,000H ;65
DB 000H,000H,0FCH,066H,066H,066H,07CH,066H
DB 066H,066H,066H,0FCH,000H,000H,000H,000H ;66
DB 000H,000H,03CH,066H,0C2H,0C0H,0C0H,0C0H
DB 0C0H,0C2H,066H,03CH,000H,000H,000H,000H ;67
DB 000H,000H,0F8H,06CH,066H,066H,066H,066H
DB 066H,066H,06CH,0F8H,000H,000H,000H,000H ;68
DB 000H,000H,0FEH,066H,062H,068H,078H,068H
DB 060H,062H,066H,0FEH,000H,000H,000H,000H ;69
DB 000H,000H,0FEH,066H,062H,068H,078H,068H
DB 060H,060H,060H,0F0H,000H,000H,000H,000H ;70
DB 000H,000H,03CH,066H,0C2H,0C0H,0C0H,0DEH
DB 0C6H,0C6H,066H,03AH,000H,000H,000H,000H ;71
DB 000H,000H,0C6H,0C6H,0C6H,0C6H,0FEH,0C6H
DB 0C6H,0C6H,0C6H,0C6H,000H,000H,000H,000H ;72
DB 000H,000H,03CH,018H,018H,018H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;73
DB 000H,000H,01EH,00CH,00CH,00CH,00CH,00CH
DB 0CCH,0CCH,0CCH,078H,000H,000H,000H,000H ;74
DB 000H,000H,0E6H,066H,06CH,06CH,078H,078H
DB 06CH,066H,066H,0E6H,000H,000H,000H,000H ;75
DB 000H,000H,0F0H,060H,060H,060H,060H,060H
DB 060H,062H,066H,0FEH,000H,000H,000H,000H ;76
DB 000H,000H,0C6H,0EEH,0FEH,0FEH,0D6H,0C6H
DB 0C6H,0C6H,0C6H,0C6H,000H,000H,000H,000H ;77
DB 000H,000H,0C6H,0E6H,0F6H,0FEH,0DEH,0CEH
DB 0C6H,0C6H,0C6H,0C6H,000H,000H,000H,000H ;78
DB 000H,000H,07CH,0C6H,0C6H,0C6H,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;79
DB 000H,000H,0FCH,066H,066H,066H,07CH,060H
DB 060H,060H,060H,0F0H,000H,000H,000H,000H ;80
DB 000H,000H,07CH,0C6H,0C6H,0C6H,0C6H,0C6H
DB 0C6H,0D6H,0DEH,07CH,00CH,00EH,000H,000H ;81
DB 000H,000H,0FCH,066H,066H,066H,07CH,06CH
DB 066H,066H,066H,0E6H,000H,000H,000H,000H ;82
DB 000H,000H,07CH,0C6H,0C6H,060H,038H,00CH
DB 006H,0C6H,0C6H,07CH,000H,000H,000H,000H ;83
DB 000H,000H,07EH,07EH,05AH,018H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;84
DB 000H,000H,0C6H,0C6H,0C6H,0C6H,0C6H,0C6H
```

```
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;85
DB 000H,000H,0C6H,0C6H,0C6H,0C6H,0C6H,0C6H
DB 0C6H,06CH,038H,010H,000H,000H,000H,000H ;86
DB 000H,000H,0C6H,0C6H,0C6H,0C6H,0C6H,0D6H
DB 0D6H,0FEH,06CH,06CH,000H,000H,000H,000H ;87
DB 000H,000H,0C6H,0C6H,06CH,06CH,038H,038H
DB 06CH,06CH,0C6H,0C6H,000H,000H,000H,000H ;88
DB 000H,000H,066H,066H,066H,066H,03CH,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;89
DB 000H,000H,0FEH,0C6H,086H,00CH,018H,030H
DB 060H,0C2H,0C6H,0FEH,000H,000H,000H,000H ;90
DB 000H,000H,03CH,030H,030H,030H,030H,030H
DB 030H,030H,030H,03CH,000H,000H,000H,000H ;91
DB 000H,000H,000H,080H,0C0H,0E0H,070H,038H
DB 01CH,00EH,006H,002H,000H,000H,000H,000H ;92
DB 000H,000H,03CH,00CH,00CH,00CH,00CH,00CH
DB 00CH,00CH,00CH,03CH,000H,000H,000H,000H ;93
DB 010H,038H,06CH,0C6H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;94
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,0FFH,000H,000H ;95
DB 030H,030H,018H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;96
DB 000H,000H,000H,000H,000H,078H,00CH,07CH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;97
DB 000H,000H,0E0H,060H,060H,078H,06CH,066H
DB 066H,066H,066H,0DCH,000H,000H,000H,000H ;98
DB 000H,000H,000H,000H,000H,07CH,0C6H,0C0H
DB 0C0H,0C0H,0C6H,07CH,000H,000H,000H,000H ;99
DB 000H,000H,01CH,00CH,00CH,03CH,06CH,0CCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;100
DB 000H,000H,000H,000H,000H,07CH,0C6H,0FEH
DB 0C0H,0C0H,0C6H,07CH,000H,000H,000H,000H ;101
DB 000H,000H,038H,06CH,064H,060H,0F0H,060H
DB 060H,060H,060H,0F0H,000H,000H,000H,000H ;102
DB 000H,000H,000H,000H,000H,076H,0CCH,0CCH
DB 0CCH,0CCH,0CCH,07CH,00CH,0CCH,078H,000H ;103
DB 000H,000H,0E0H,060H,060H,06CH,076H,066H
DB 066H,066H,066H,0E6H,000H,000H,000H,000H ;104
DB 000H,000H,018H,018H,000H,038H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;105
DB 000H,000H,006H,006H,000H,00EH,006H,006H
```



```
DB 006H,006H,006H,006H,066H,066H,03CH,000H ;106
DB 000H,000H,0E0H,060H,060H,066H,06CH,078H
DB 078H,06CH,066H,0E6H,000H,000H,000H,000H ;107
DB 000H,000H,038H,018H,018H,018H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;108
DB 000H,000H,000H,000H,000H,0ECH,0FEH,0D6H
DB 0D6H,0D6H,0D6H,0D6H,000H,000H,000H,000H ;109
DB 000H,000H,000H,000H,000H,0DCH,066H,066H
DB 066H,066H,066H,066H,000H,000H,000H,000H ;110
DB 000H,000H,000H,000H,000H,07CH,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;111
DB 000H,000H,000H,000H,000H,0DCH,066H,066H
DB 066H,066H,066H,07CH,060H,060H,0F0H,000H ;112
DB 000H,000H,000H,000H,000H,076H,0CCH,0CCH
DB 0CCH,0CCH,0CCH,07CH,00CH,00CH,01EH,000H ;113
DB 000H,000H,000H,000H,000H,0DCH,076H,062H
DB 060H,060H,060H,0F0H,000H,000H,000H,000H ;114
DB 000H,000H,000H,000H,000H,07CH,0C6H,060H
DB 038H,00CH,0C6H,07CH,000H,000H,000H,000H ;115
DB 000H,000H,010H,030H,030H,0FCH,030H,030H
DB 030H,030H,036H,01CH,000H,000H,000H,000H ;116
DB 000H,000H,000H,000H,000H,0CCH,0CCH,0CCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;117
DB 000H,000H,000H,000H,000H,066H,066H,066H
DB 066H,066H,03CH,018H,000H,000H,000H,000H ;118
DB 000H,000H,000H,000H,000H,0C6H,0C6H,0C6H
DB 0D6H,0D6H,0FEH,06CH,000H,000H,000H,000H ;119
DB 000H,000H,000H,000H,000H,0C6H,06CH,038H
DB 038H,038H,06CH,0C6H,000H,000H,000H,000H ;120
DB 000H,000H,000H,000H,000H,0C6H,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07EH,006H,00CH,0F8H,000H ;121
DB 000H,000H,000H,000H,000H,0FEH,0CCH,018H
DB 030H,060H,0C6H,0FEH,000H,000H,000H,000H ;122
DB 000H,000H,00EH,018H,018H,018H,070H,018H
DB 018H,018H,018H,00EH,000H,000H,000H,000H ;123
DB 000H,000H,018H,018H,018H,018H,000H,018H
DB 018H,018H,018H,018H,000H,000H,000H,000H ;124
DB 000H,000H,070H,018H,018H,018H,00EH,018H
DB 018H,018H,018H,070H,000H,000H,000H,000H ;125
DB 000H,000H,076H,0DCH,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;126
DB 000H,000H,000H,000H,010H,038H,06CH,0C6H
```

```
DB 0C6H,0C6H,0FEH,000H,000H,000H,000H,000H ;127
DB 000H,000H,03CH,066H,0C2H,0C0H,0C0H,0C0H
DB 0C2H,066H,03CH,00CH,006H,07CH,000H,000H ;128
DB 000H,000H,0CCH,000H,000H,0CCH,0CCH,0CCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;129
DB 000H,00CH,018H,030H,000H,07CH,0C6H,0FEH
DB 0C0H,0C0H,0C6H,07CH,000H,000H,000H,000H ;130
DB 000H,010H,038H,06CH,000H,078H,00CH,07CH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;131
DB 000H,000H,0CCH,000H,000H,078H,00CH,07CH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;132
DB 000H,060H,030H,018H,000H,078H,00CH,07CH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;133
DB 000H,038H,06CH,038H,000H,078H,00CH,07CH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;134
DB 000H,000H,000H,000H,03CH,066H,060H,060H
DB 066H,03CH,00CH,006H,03CH,000H,000H,000H ;135
DB 000H,010H,038H,06CH,000H,07CH,0C6H,0FEH
DB 0C0H,0C0H,0C6H,07CH,000H,000H,000H,000H ;136
DB 000H,000H,0C6H,000H,000H,07CH,0C6H,0FEH
DB 0C0H,0C0H,0C6H,07CH,000H,000H,000H,000H ;137
DB 000H,060H,030H,018H,000H,07CH,0C6H,0FEH
DB 0C0H,0C0H,0C6H,07CH,000H,000H,000H,000H ;138
DB 000H,000H,066H,000H,000H,038H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;139
DB 000H,018H,03CH,066H,000H,038H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;140
DB 000H,060H,030H,018H,000H,038H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;141
DB 000H,0C6H,000H,010H,038H,06CH,0C6H,0C6H
DB 0FEH,0C6H,0C6H,0C6H,000H,000H,000H,000H ;142
DB 038H,06CH,038H,000H,038H,06CH,0C6H,0C6H
DB 0FEH,0C6H,0C6H,0C6H,000H,000H,000H,000H ;143
DB 018H,030H,060H,000H,0FEH,066H,060H,07CH
DB 060H,060H,066H,0FEH,000H,000H,000H,000H ;144
DB 000H,000H,000H,000H,000H,0CCH,076H,036H
DB 07EH,0D8H,0D8H,06EH,000H,000H,000H,000H ;145
DB 000H,000H,03EH,06CH,0CCH,0CCH,0FEH,0CCH
DB 0CCH,0CCH,0CCH,0CEH,000H,000H,000H,000H ;146
DB 000H,010H,038H,06CH,000H,07CH,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;147
DB 000H,000H,0C6H,000H,000H,07CH,0C6H,0C6H
```



```
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;148
DB 000H,060H,030H,018H,000H,07CH,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;149
DB 000H,030H,078H,0CCH,000H,0CCH,0CCH,0CCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;150
DB 000H,060H,030H,018H,000H,0CCH,0CCH,0CCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;151
DB 000H,000H,0C6H,000H,000H,0C6H,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07EH,006H,00CH,078H,000H ;152
DB 000H,0C6H,000H,07CH,0C6H,0C6H,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;153
DB 000H,0C6H,000H,0C6H,0C6H,0C6H,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;154
DB 000H,018H,018H,03CH,066H,060H,060H,060H
DB 066H,03CH,018H,018H,000H,000H,000H,000H ;155
DB 000H,038H,06CH,064H,060H,0F0H,060H,060H
DB 060H,060H,0E6H,0FCH,000H,000H,000H,000H ;156
DB 000H,000H,066H,066H,03CH,018H,07EH,018H
DB 07EH,018H,018H,018H,000H,000H,000H,000H ;157
DB 000H,0F8H,0CCH,0CCH,0F8H,0C4H,0CCH,0DEH
DB 0CCH,0CCH,0CCH,0C6H,000H,000H,000H,000H ;158
DB 000H,00EH,01BH,018H,018H,018H,07EH,018H
DB 018H,018H,018H,018H,0D8H,070H,000H,000H ;159
DB 000H,018H,030H,060H,000H,078H,00CH,07CH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;160
DB 000H,00CH,018H,030H,000H,038H,018H,018H
DB 018H,018H,018H,03CH,000H,000H,000H,000H ;161
DB 000H,018H,030H,060H,000H,07CH,0C6H,0C6H
DB 0C6H,0C6H,0C6H,07CH,000H,000H,000H,000H ;162
DB 000H,018H,030H,060H,000H,0CCH,0CCH,0CCH
DB 0CCH,0CCH,0CCH,076H,000H,000H,000H,000H ;163
DB 000H,000H,076H,0DCH,000H,0DCH,066H,066H
DB 066H,066H,066H,066H,000H,000H,000H,000H ;164
DB 076H,0DCH,000H,0C6H,0E6H,0F6H,0FEH,0DEH
DB 0CEH,0C6H,0C6H,0C6H,000H,000H,000H,000H ;165
DB 000H,03CH,06CH,06CH,03EH,000H,07EH,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;166
DB 000H,038H,06CH,06CH,038H,000H,07CH,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;167
DB 000H,000H,030H,030H,000H,030H,030H,060H
DB 0C0H,0C6H,0C6H,07CH,000H,000H,000H,000H ;168
DB 000H,000H,000H,000H,000H,000H,0FEH,0C0H
```

```
DB 0C0H,0C0H,0C0H,000H,000H,000H,000H,000H ;169
DB 000H,000H,000H,000H,000H,000H,0FEH,006H
DB 006H,006H,006H,000H,000H,000H,000H,000H ;170
DB 000H,0C0H,0C0H,0C2H,0C6H,0CCH,018H,030H
DB 060H,0CEH,093H,006H,00CH,01FH,000H,000H ;171
DB 000H,0C0H,0C0H,0C2H,0C6H,0CCH,018H,030H
DB 066H,0CEH,09AH,03FH,006H,00FH,000H,000H ;172
DB 000H,000H,018H,018H,000H,018H,018H,018H
DB 03CH,03CH,03CH,018H,000H,000H,000H,000H ;173
DB 000H,000H,000H,000H,000H,033H,066H,0CCH
DB 066H,033H,000H,000H,000H,000H,000H,000H ;174
DB 000H,000H,000H,000H,000H,0CCH,066H,033H
DB 066H,0CCH,000H,000H,000H,000H,000H,000H ;175
DB 011H,044H,011H,044H,011H,044H,011H,044H
DB 011H,044H,011H,044H,011H,044H,011H,044H ;176
DB 055H,0AAH,055H,0AAH,055H,0AAH,055H,0AAH
DB 055H,0AAH,055H,0AAH,055H,0AAH,055H,0AAH ;177
DB 0DDH,077H,0DDH,077H,0DDH,077H,0DDH,077H
DB 0DDH,077H,0DDH,077H,0DDH,077H,0DDH,077H ;178
DB 018H,018H,018H,018H,018H,018H,018H,018H
DB 018H,018H,018H,018H,018H,018H,018H,018H ;179
DB 018H,018H,018H,018H,018H,018H,018H,0F8H
DB 018H,018H,018H,018H,018H,018H,018H,018H ;180
DB 018H,018H,018H,018H,018H,0F8H,018H,0F8H
DB 018H,018H,018H,018H,018H,018H,018H,018H ;181
DB 036H,036H,036H,036H,036H,036H,036H,0F6H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;182
DB 000H,000H,000H,000H,000H,000H,000H,0FEH
DB 036H,036H,036H,036H,036H,036H,036H,036H ;183
DB 000H,000H,000H,000H,000H,0F8H,018H,0F8H
DB 018H,018H,018H,018H,018H,018H,018H,018H ;184
DB 036H,036H,036H,036H,036H,0F6H,006H,0F6H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;185
DB 036H,036H,036H,036H,036H,036H,036H,036H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;186
DB 000H,000H,000H,000H,000H,0FEH,006H,0F6H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;187
DB 036H,036H,036H,036H,036H,0F6H,006H,0FEH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;188
DB 036H,036H,036H,036H,036H,036H,036H,0FEH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;189
DB 018H,018H,018H,018H,018H,0F8H,018H,0F8H
```



```
DB 000H,000H,000H,000H,000H,000H,000H,000H ;190
DB 000H,000H,000H,000H,000H,000H,000H,0F8H
DB 018H,018H,018H,018H,018H,018H,018H,018H ;191
DB 018H,018H,018H,018H,018H,018H,018H,01FH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;192
DB 018H,018H,018H,018H,018H,018H,018H,0FFH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;193
DB 000H,000H,000H,000H,000H,000H,000H,0FFH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;194
DB 018H,018H,018H,018H,018H,018H,018H,01FH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;195
DB 000H,000H,000H,000H,000H,000H,000H,0FFH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;196
DB 018H,018H,018H,018H,018H,018H,018H,0FFH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;197
DB 018H,018H,018H,018H,018H,01FH,018H,01FH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;198
DB 036H,036H,036H,036H,036H,036H,036H,037H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;199
DB 036H,036H,036H,036H,036H,037H,030H,03FH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;200
DB 000H,000H,000H,000H,000H,03FH,030H,037H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;201
DB 036H,036H,036H,036H,036H,0F7H,000H,0FFH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;202
DB 000H,000H,000H,000H,000H,0FFH,000H,0F7H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;203
DB 036H,036H,036H,036H,036H,037H,030H,037H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;204
DB 000H,000H,000H,000H,000H,0FFH,000H,0FFH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;205
DB 036H,036H,036H,036H,036H,0F7H,000H,0F7H
DB 036H,036H,036H,036H,036H,036H,036H,036H ;206
DB 018H,018H,018H,018H,018H,0FFH,000H,0FFH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;207
DB 036H,036H,036H,036H,036H,036H,036H,0FFH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;208
DB 000H,000H,000H,000H,000H,0FFH,000H,0FFH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;209
DB 000H,000H,000H,000H,000H,000H,000H,0FFH
DB 036H,036H,036H,036H,036H,036H,036H,036H ;210
DB 036H,036H,036H,036H,036H,036H,036H,03FH
```

```
DB 000H,000H,000H,000H,000H,000H,000H,000H ;211
DB 018H,018H,018H,018H,018H,01FH,018H,01FH
DB 000H,000H,000H,000H,000H,000H,000H,000H ;212
DB 000H,000H,000H,000H,000H,01FH,018H,01FH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;213
DB 000H,000H,000H,000H,000H,000H,000H,03FH
DB 036H,036H,036H,036H,036H,036H,036H,036H ;214
DB 036H,036H,036H,036H,036H,036H,036H,0FFH
DB 036H,036H,036H,036H,036H,036H,036H,036H ;215
DB 018H,018H,018H,018H,018H,0FFH,018H,0FFH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;216
DB 018H,018H,018H,018H,018H,018H,018H,0F8H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;217
DB 000H,000H,000H,000H,000H,000H,000H,01FH
DB 018H,018H,018H,018H,018H,018H,018H,018H ;218
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH ;219
DB 000H,000H,000H,000H,000H,000H,000H,0FFH
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH ;220
DB 0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H
DB 0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H,0F0H ;221
DB 00FH,00FH,00FH,00FH,00FH,00FH,00FH,00FH
DB 00FH,00FH,00FH,00FH,00FH,00FH,00FH,00FH ;222
DB 0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;223
DB 000H,000H,000H,000H,000H,076H,0DCH,0D8H
DB 0D8H,0D8H,0DCH,076H,000H,000H,000H,000H ;224
DB 000H,000H,078H,0CCH,0CCH,0CCH,0D8H,0CCH
DB 0C6H,0C6H,0C6H,0CCH,000H,000H,000H,000H ;225
DB 000H,000H,0FEH,0C6H,0C6H,0C0H,0C0H,0C0H
DB 0C0H,0C0H,0C0H,0C0H,000H,000H,000H,000H ;226
DB 000H,000H,000H,000H,0FEH,06CH,06CH,06CH
DB 06CH,06CH,06CH,06CH,000H,000H,000H,000H ;227
DB 000H,000H,000H,0FEH,0C6H,060H,030H,018H
DB 030H,060H,0C6H,0FEH,000H,000H,000H,000H ;228
DB 000H,000H,000H,000H,000H,07EH,0D8H,0D8H
DB 0D8H,0D8H,0D8H,070H,000H,000H,000H,000H ;229
DB 000H,000H,000H,000H,066H,066H,066H,066H
DB 066H,07CH,060H,060H,0C0H,000H,000H,000H ;230
DB 000H,000H,000H,000H,076H,0DCH,018H,018H
DB 018H,018H,018H,018H,000H,000H,000H,000H ;231
DB 000H,000H,000H,07EH,018H,03CH,066H,066H
```



```
DB 066H,03CH,018H,07EH,000H,000H,000H,000H ;232
DB 000H,000H,000H,038H,06CH,0C6H,0C6H,0FEH
DB 0C6H,0C6H,06CH,038H,000H,000H,000H,000H ;233
DB 000H,000H,038H,06CH,0C6H,0C6H,0C6H,06CH
DB 06CH,06CH,06CH,0EEH,000H,000H,000H,000H ;234
DB 000H,000H,01EH,030H,018H,00CH,03EH,066H
DB 066H,066H,066H,03CH,000H,000H,000H,000H ;235
DB 000H,000H,000H,000H,000H,07EH,0DBH,0DBH
DB 0DBH,07EH,000H,000H,000H,000H,000H,000H ;236
DB 000H,000H,000H,003H,006H,07EH,0CFH,0DBH
DB 0F3H,07EH,060H,0C0H,000H,000H,000H,000H ;237
DB 000H,000H,01CH,030H,060H,060H,07CH,060H
DB 060H,060H,030H,01CH,000H,000H,000H,000H ;238
DB 000H,000H,000H,07CH,0C6H,0C6H,0C6H,0C6H
DB 0C6H,0C6H,0C6H,0C6H,000H,000H,000H,000H ;239
DB 000H,000H,000H,000H,0FEH,000H,000H,0FEH
DB 000H,000H,0FEH,000H,000H,000H,000H,000H ;240
DB 000H,000H,000H,000H,018H,018H,07EH,018H
DB 018H,000H,000H,0FFH,000H,000H,000H,000H ;241
DB 000H,000H,000H,030H,018H,00CH,006H,00CH
DB 018H,030H,000H,07EH,000H,000H,000H,000H ;242
DB 000H,000H,000H,00CH,018H,030H,060H,030H
DB 018H,00CH,000H,07EH,000H,000H,000H,000H ;243
DB 000H,000H,00EH,01BH,01BH,018H,018H,018H
DB 018H,018H,018H,018H,018H,018H,018H,018H ;244
DB 018H,018H,018H,018H,018H,018H,018H,018H
DB 0D8H,0D8H,0D8H,070H,000H,000H,000H,000H ;245
DB 000H,000H,000H,000H,018H,018H,000H,07EH
DB 000H,018H,018H,000H,000H,000H,000H,000H ;246
DB 000H,000H,000H,000H,000H,076H,0DCH,000H
DB 076H,0DCH,000H,000H,000H,000H,000H,000H ;247
DB 000H,038H,06CH,06CH,038H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;248
DB 000H,000H,000H,000H,000H,000H,000H,018H
DB 018H,000H,000H,000H,000H,000H,000H,000H ;249
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 018H,000H,000H,000H,000H,000H,000H,000H ;250
DB 000H,00FH,00CH,00CH,00CH,00CH,00CH,0ECH
DB 06CH,06CH,03CH,01CH,000H,000H,000H,000H ;251
DB 000H,0D8H,06CH,06CH,06CH,06CH,06CH,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;252
DB 000H,070H,098H,030H,060H,0C8H,0F8H,000H
```

```
DB 000H,000H,000H,000H,000H,000H,000H,000H ;253
DB 000H,000H,000H,000H,07CH,07CH,07CH,07CH
DB 07CH,07CH,07CH,000H,000H,000H,000H,000H ;254
DB 000H,000H,000H,000H,000H,000H,000H,000H
DB 000H,000H,000H,000H,000H,000H,000H,000H ;255
_DATA         ENDS
              end
```

9.14.6 Loading and Reading the Palette DAC Table

```
// ************************************************************************
// PAL_XMPL.CPP
//
// THIS PROGRAM DEMONSTRATES THE DIRECT METHOD OF WRITING TO THE PALETTE
// DAC TABLE, BYPASSING THE PC VGA BIOS CALLS.
//
// THE PALETTE DAC TABLE CONSISTS OF 256 ENTRIES, PLUS 3 EXTENDED ENTRIES
// ADDRESSE WITH SR12[3].
//
// THE FOLLOWING IO REGISTERS CONTROL THE PALETTE:
//      0X3C6 PIXEL MASK REGISTER
//      0X3C7 PALETTE READ INDEX (WRITE ONLY)/DAC STATE REGISTER(READ ONLY)
//      0X3C8 PALETTE WRITE INDEX (WRITE ONLY)
//      0X3C9 PALETTE DATA (READ/WRITE)
//
// THE INDEX REGISTERS 0X3C7/0X3C8 AUTOMATICALLY INCREMENT AFTER EVERY THREE
// READS OR WRITES TO OR FROM 0X3C9.
//
// PALETTE DATA CONSISTS OF 1 BYTE EACH OF RED, GREEN AND BLUE.
// IN EACH BYTE, HOWEVER, ONLY 6 BITS ARE USED, GIVING 18 BITS FOR EACH
// PALETTE ENTRY.
//
// COMPILED AND TESTED USING BORLAND C++ 4.5
// ************************************************************************

#include <dos.h>

// ************************************************************************
//      Overscan_Cycle_Colors()
// ************************************************************************
void Overscan_Cycle_Colors( void )
{
        unsigned char sr12;
```



```
        // SR12[1] : ENABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) | 0x82;   // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

        // WRITE RED
        outportb( 0x3c8, 0x02 );          // ADDRESSABLE AS X2H
        outportb( 0x3c9, 0x3f );
        outportb( 0x3c9, 0 );
        outportb( 0x3c9, 0 );
        delay( 500 );

        // WRITE ALL GREEN
        outportb( 0x3c8, 0x02 );          // ADDRESSABLE AS X2H
        outportb( 0x3c9, 0 );
        outportb( 0x3c9, 0x3f );
        outportb( 0x3c9, 0 );
        delay( 500 );

        // WRITE ALL BLUE
        outportb( 0x3c8, 0x02 );          // ADDRESSABLE AS X2H
        outportb( 0x3c9, 0 );
        outportb( 0x3c9, 0 );
        outportb( 0x3c9, 0x3f );
        delay( 500 );

        // SR12[1] : DISABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) & ~0x82;  // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

};      // Overscan_Cycle_Colors()

// ************************************************************************
//      Palette_Read()
// THIS FUNCTION READS THE ENTIRE PALETTE AND STORES IT IN <PALETTE>
// ************************************************************************
void Palette_Read( unsigned char* palette )
{
        unsigned char sr12;
        int x, p;
```

```
        outportb( 0x3c7, 0x00 );    // SET PALETTE READ INDEX : START WITH 0

        for ( x = 0, p = 0; x < 256; x++ )
        {
          palette[p++] = inportb( 0x3c9 );        // READ RED
          palette[p++] = inportb( 0x3c9 );        // READ GREEN
          palette[p++] = inportb( 0x3c9 );        // READ BLUE
        }

        // SR12[1] : ENABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) | 0x02;   // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

        // READ DAC LUT ENTRY 256 -- HW CURSOR BACKGROUND
        outportb( 0x3c7, 0x00 );          // ADDRESSABLE AS X0H
        palette[p++] = inportb( 0x3c9 );  // READ RED
        palette[p++] = inportb( 0x3c9 );  // READ GREEN
        palette[p++] = inportb( 0x3c9 );  // READ BLUE

        // READ DAC LUT ENTRY 257 -- HW CURSOR FOREGROUND
        outportb( 0x3c7, 0x0f );          // ADDRESSABLE AS XFH
        palette[p++] = inportb( 0x3c9 );  // READ RED
        palette[p++] = inportb( 0x3c9 );  // READ GREEN
        palette[p++] = inportb( 0x3c9 );  // READ BLUE

        // READ DAC LUT ENTRY 256 -- OVERSCAN BORDER
        outportb( 0x3c7, 0x02 );          // ADDRESSABLE AS X2H
        palette[p++] = inportb( 0x3c9 );  // READ RED
        palette[p++] = inportb( 0x3c9 );  // READ GREEN
        palette[p++] = inportb( 0x3c9 );  // READ BLUE

        // SR12[1] : DISABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) & ~0x02;  // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

};      // Palette_Read()

// ***************************************************************************
//      Palette_Write()
```



```
// THIS FUNCTION WRITES THE ENTIRE PALETTE AND STORES IT IN <PALETTE>
// ***************************************************************************
void Palette_Write( unsigned char* palette )
{
        unsigned char sr12;
        int x, p;

        outportb( 0x3c8, 0x00 );    // SET PALETTE READ INDEX : START WITH 0

        for ( x = 0, p = 0; x < 256; x++ )
        {
          outportb( 0x3c9, palette[p++] );        // WRITE RED
          outportb( 0x3c9, palette[p++] );        // WRITE GREEN
          outportb( 0x3c9, palette[p++] );        // WRITE BLUE
        }

        // SR12[1] : ENABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) | 0x02;   // READ AND SET SR12[1]
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

        // READ DAC LUT ENTRY 256 -- HW CURSOR BACKGROUND
        outportb( 0x3c8, 0x00 );          // ADDRESSABLE AS X0H
        outportb( 0x3c9, palette[p++] );  // WRITE RED
        outportb( 0x3c9, palette[p++] );  // WRITE GREEN
        outportb( 0x3c9, palette[p++] );  // WRITE BLUE

        // READ DAC LUT ENTRY 257 -- HW CURSOR FOREGROUND
        outportb( 0x3c8, 0x0F );          // ADDRESSABLE AS XFH
        outportb( 0x3c9, palette[p++] );  // WRITE RED
        outportb( 0x3c9, palette[p++] );  // WRITE GREEN
        outportb( 0x3c9, palette[p++] );  // WRITE BLUE

        // READ DAC LUT ENTRY 258 -- OVERSCAN BORDER
        outportb( 0x3c8, 0x02 );          // ADDRESSABLE AS X2H
        outportb( 0x3c9, palette[p++] );  // WRITE RED
        outportb( 0x3c9, palette[p++] );  // WRITE GREEN
        outportb( 0x3c9, palette[p++] );  // WRITE BLUE

        // SR12[1] : DISABLE ACCESS TO DAC EXTENDED COLORS
        outportb( 0x3c4, 0x12 );          // SR12
        sr12 = inportb( 0x3c5 ) & ~0x02;  // READ AND SET SR12[1]
```

```
        outportb( 0x3c5, sr12 );          // WRITE NEW VALUE

};      // Palette_Write()

// ************************************************************************
//      Palette_Entry_Read()
// ************************************************************************
void Palette_Entry_Read( unsigned int index, unsigned char temp[3] )
{
        outportb( 0x3c7,  index );
        temp[0] = inportb( 0x3c9 );       // READ RED
        temp[1] = inportb( 0x3c9 );       // READ GREEN
        temp[2] = inportb( 0x3c9 );       // READ BLUE
};      // Palette_Entry_Read()

// ************************************************************************
//      Palette_Entry_Write()
// ************************************************************************
void Palette_Entry_Write( unsigned int index, unsigned char temp[3] )
{
        outportb( 0x3c8, index );
        outportb( 0x3c9, temp[0] );        // WRITE RED
        outportb( 0x3c9, temp[1] );        // WRITE GREEN
        outportb( 0x3c9, temp[2] );        // WRITE BLUE
};      // Palette_Entry_Write()

// ************************************************************************
//      Palette_Cycle_Up()
// LOWER PALETTE ENTRIES MIGRATE UP THE PALETTE, WRAPPING AROUND AT 255.
// PALETTE.RED_GREEN_BLUE[N + 1] <-- PALETTE.RED_GREEN_BLUE[N]
// ************************************************************************
void Palette_Cycle_Up( void )
{
        int index;
        unsigned char temp[3];

        // SET PALETTE READ INDEX : START WITH FF
        Palette_Entry_Read( 0x00ff, temp );

        for ( index = 0xfe; index >= 0; index-- )
        {
          unsigned char temp[3];
```



```
          Palette_Entry_Read( index, temp );
          Palette_Entry_Write( index + 1, temp );
        }

        Palette_Entry_Write( 0x0000, temp );
};      // Palette_Cycle_Up()

// ***************************************************************************
//      Palette_Cycle_Down()
// HIGHER PALETTE ENTRIES MIGRATE DOWN THE PALETTE, WRAPPING AROUND AT 0.
// PALETTE.RED_GREEN_BLUE[N] <-- PALETTE.RED_GREEN_BLUE[N + 1]
// ***************************************************************************
void Palette_Cycle_Down( void )
{
        int index;
        unsigned char temp[3];

        Palette_Entry_Read( 0x0000, temp );

        for ( index = 0x00; index < 0xff; index++ )
        {
          unsigned char temp[3];

          Palette_Entry_Read( index + 1, temp );
          Palette_Entry_Write( index, temp );
        }

        Palette_Entry_Write( 0x00ff, temp );
};      // Palette_Cycle_Down()

// ***************************************************************************
//      main()
// ***************************************************************************
void main( void )
{
        int x;
        unsigned char old_pal[777], black_pal[777], white_pal[777];

        for ( x = 0; x < 768; x++ )
        {
          black_pal[x] = 0;           // WRITE ZERO TO EVERY BYTE
```

```
        white_pal[x] = 0x3f;        // TURN ON ALL 6 BITS OF EVERY BYTE
      }

      outportb( 0x3c6, 0xff );      // ENABLE ALL PIXEL DATA

      Overscan_Cycle_Colors();      // CYCLE BORDER COLOR RED, GREEN, BLUE

      Palette_Read( old_pal );      // READ ORIGINAL PALETTE

      // TURN SCREEN BLACK
      Palette_Write( black_pal );

      // TURN SCREEN WHITE
      Palette_Write( white_pal );

      // SCREEN FLICKERS WITH DIFFERENT COLOR BACKGROUND AND FOREGROUND
      Palette_Write( old_pal );             // RESTORE ORIGINAL PALETTE
      for ( x = 0; x < 256; x++ ) Palette_Cycle_Up();

      // SCREEN FLICKERS WITH DIFFERENT COLOR BACKGROUND AND FOREGROUND
      Palette_Write( old_pal );             // RESTORE ORIGINAL PALETTE
      for ( x = 0; x < 256; x++ )  Palette_Cycle_Down();

      // RESTORE ORIGINAL PALETTE
      Palette_Write( old_pal );

};    // main()
```


Detailed Pin Descriptions

10. DETAILED PIN DESCRIPTIONS

Refer to Section 1 of Chapter 3, “Data Book” for the pin diagram. The following abbreviations are used for pin types in the following sections:

- (I) indicates inputs
- (O) indicates output
- (O-Z) indicates tristate output
- (OC) indicates open collector output
- (BIO) indicates bidirectional I/O
- (I/O) indicates input or output depending on how the device is configured and programmed.

10.1 Host Interface: PCI Bus

Name	Type	Description
IDSEL	I	**INITIALIZATION DEVICE SELECT:** This input is a chip select in lieu of the upper 24 address lines during configuration read and write cycles.
FRAME#	I	**FRAME#:** This active-low input indicates the beginning and duration of a (burst) access. FRAME# is asserted to indicate the beginning of a bus transaction. While asserted, data transfers continue. When the transaction is in its final data phase, FRAME# goes not active.
IRDY#	I	**INITIATOR READY#:** This input establishes a handshake between the CL-GD5446 and the PCI bus informing the CL-GD5446 when the cycle has ended. Wait states are inserted until both IRDY# and TRDY# are asserted together.
RST#	I	**RESET#:** This active-low input initializes the CL-GD5446 to a known state. The trailing (rising) edge of this input loads the Configuration register CF[15:0] with the levels on MD[63:48]. These levels are determined by internal pull-up resistors and optional external pull-down resistors.
CLK	I	**CLOCK:** This is the bus timing reference for the CL-GD5446. All PCI timing is with reference to this clock.
TRDY#	BIO	**TARGET READY#:** This active-low output terminates a CL-GD5446 bus cycle. This pin is an input for ‘snooped’ palette cycles. This output is a sustained tristate as defined in the PCI specification.
STOP#	BIO	**STOP#:** This active-low output indicates a request to the PCI Bus Master to stop the current transaction. This pin is an input for ‘snooped’ palette cycles. This output is a sustained tristate as defined in the PCI specification.
PAR	BIO	**PARITY:** This signal provides even parity across AD[31:0]. The CL-GD5446 asserts correct parity when it drives the bus for reads. The CL-GD5446 does not check bus parity.
DEVSEL#	O-Z	**DEVICE SELECT#:** This output is driven low to indicate that the CL-GD5446 will respond to the current cycle. This output is a sustained tristate as defined in the PCI specification. When palette snooping is enabled, DEVSEL# is not made active for palette cycles.
INTR#	OC	**INTERRUPT#:** This open-collector output is driven low when the CL-GD5446 is claiming an interrupt (CF14), GR17[2] is ‘1’, and an interrupt request is pending.

10.1 Host Interface: PCI Bus *(cont.)*

Name	Type	Description
AD[31:0]	BIO	**ADDRESS AND DATA [31:0]:** These multiplexed, bidirectional pins transfer system address and data during any memory or I/O operation on the PCI bus. These pins connect directly to AD[31:0] on the PCI bus. During the first clock of a transaction, these pins contain a 32-bit physical byte address. During subsequent clocks, they contain data.
CBE[3:0]#	I	**COMMAND AND BYTE ENABLE[3:0]#:** These multiplexed pins transfer bus commands and byte enables during any memory or I/O operation on the PCI bus. These pins connect directly to CBE[3:0]# on the PCI bus. During the address phase of the operation, CBE[3:0]# define the bus command (refer to the following table). During the data phase(s), they are the Byte Enable outputs. CBE0# applies to byte 0 and CBE3# applies to byte 3.
BIOSA[15:0]	O	**BIOS ADDRESS [15:0]:** These outputs are latched from the AD[31:0] bus and address the display system BIOS. These signals are connected directly to the address inputs of an 8-bit ROM. This provides compliance with the PCI 'one-load' specification.
BIOSD[7:0]	I	**BIOS DATA [7:0]:** These inputs transfer data during a display system BIOS operation. These pins are connected directly to the data outputs of an 8-bit ROM. These pins are multiplexed with MD[7:0].

CBE3#[a]	CBE2#	CBE1#	CBE0#	Command Type
0	0	1	0	I/O Read
0	0	1	1	I/O Write
0	1	1	0	Memory Read
0	1	1	1	Memory Write
1	0	1	0	Configuration Read
1	0	1	1	Configuration Write

[a] CBE values not in this table are not used by the CL-GD5446.

10.2 Video Interface

Name	Type	Description
VSYNC	O-Z	**VERTICAL SYNC:** This output supplies the vertical synchronization pulse to the monitor. The polarity of this output is programmable. This pin is put into high-impedance when ESYNC# is low.
HSYNC	O-Z	**HORIZONTAL SYNC:** This output supplies the horizontal synchronization pulse to the monitor. The polarity of this output is programmable. This pin is put into high-impedance when ESYNC# is low.
BLANK#	I/O	**BLANK#:** When the video pins are configured for the feature connector, ESYNC# controls this bidirectional pin. If ESYNC# is high, BLANK# is an output. As an output, BLANK# supplies a blanking signal to the feature connector. The same term is used internally as the DAC blanking term. If ESYNC# is low, BLANK# is an input. When BLANK# is active-low, the RED, GREEN, and BLUE outputs are forced to '0' current. When the video pins are configured for video capture (CR50[1:0]), this pin is the HREF Input. When a pull-down resistor is installed on MD54 (CF6), this pin powers up in the high-impedance state.
P[15:8]	I/O	**PIXEL BUS [15:8]:** These pins extend the Pixel bus for inputs only. GR18[6] must = 1. These pins are redefined BIOSA pins. These pins can also be the V-Port PIXD[15:8] inputs.
P[7:0]	I/O	**PIXEL BUS [7:0]:** When the video pins are configured for the feature connector, these bidirectional pins are controlled by EVIDEO#. If EVIDEO# is high and overlay is not enabled, these pins are outputs and reflect the address into the palette DAC. If EVIDEO# is low, these pins are inputs and can be used to drive pixel values into the palette DAC or drive the DACs directly in 8-bpp modes. If any overlay mode is selected by programming CR1A[3:2] to any value other than '00', these pins are inputs. When the video pins are configured for video capture (CR50[1:0]), these pins are the PIXD[7:0] inputs. When a pull-down resistor is installed on MD54 (CF6), these pins power up in the high-impedance state.
DCLK	I/O	**DOT CLOCK:** When the video pins are configured for the feature connector, this bidirectional pin is controlled by EDCLK#. If EDCLK# is high, this is an output and externally latches the data on the Pixel bus. If EDCLK# is low, this is an input and can clock data on the Pixel bus into the CL-GD5446. When the video pins are configured for video capture (CR50[1:0], this pin is the PIXCLK input. When a pull-down resistor is installed on MD54 (CF6), this pin powers up in the high-impedance state.
ESYNC#	I/O	**ENABLE SYNC AND BLANK#:** When the video pins are configured for the feature connector, this input controls the buffers on HSYNC, VSYNC, and BLANK#. If ESYNC# is high, these three pins are outputs. If ESYNC# is low, BLANK# is an input, and HSYNC and VSYNC are not driven by the CL-GD5446. This pin is connected to the corresponding pin on the feature connector with a 1-kΩ series resistor. When the video pins are configured for video capture (CR50[1:0]), or if a pull-down resistor is installed on MD54 [CF6), this pin does not control BLANK#.

10.2 Video Interface *(cont.)*

Name	Type	Description
EVIDEO#	I/O	**ENABLE VIDEO#:** When the video pins are configured for the feature connector, this input controls the buffers on P[7:0]. If EVIDEO# is high and overlay is not selected, P[7:0] are outputs. If EVIDEO# is low, P[7:0] are inputs. This pin is connected to the corresponding pin on the feature connector with a 1-kΩ series resistor. This pin is also an output when color key switching is enabled. When the video pins are configured for video capture (CR50[1:0]), this pin is an input used for VACT. When EVIDEO# is high, it indicates valid video samples.
EDCLK#	I	**ENABLE DOT CLOCK#:** When the video pins are configured for the feature connector, this input controls the buffer on DCLK. If EDCLK# is high, DCLK is an output. If EDCLK# is low, DCLK is an input. This pin is connected to the corresponding pin on the feature connector with a 1-kΩ series resistor. When the video pins are configured for video capture (CR50[1:0]), this pin is the VREF input.
RED	O	**RED VIDEO:** This analog output supplies current corresponding to the red value of the pixel being displayed. Each of the three DACs consists of 255 summed current sources. For each pixel, either the 6-bit value from the LUT or an appropriately sized true-color value is applied to each DAC input to determine the number of current sources to be summed. Each DAC output is typically terminated to monitor ground with a 75-Ω, 2% resistor. This resistor, in parallel with the 75-Ω resistor in the monitor, yields a 37.5-Ω impedance to ground. The full-scale current is determined by the resistor connected to REXT, typically 135 Ω. Equations relating to the value of this resistor are given in Appendix B4, "Current Reference".
GREEN	O	**GREEN VIDEO:** This analog output supplies current corresponding to the green value of the pixel being displayed. See the description of RED VIDEO for information regarding the termination of this pin.
BLUE	O	**BLUE VIDEO:** This analog output supplies current corresponding to the blue value of the pixel being displayed. See the description of RED VIDEO for information regarding the termination of this pin.
IREF	Analog I	**CURRENT REFERENCE:** This pin is connected to capacitors returned for DACVDD. This capacitance stabilizes the DAC current reference. Refer to Appendix B2, "PCI Bus Reference Design", for information relating to the capacitor values.
RSET	Analog I	**FULL-SCALE CURRENT REFERENCE:** This pin is connected to a resistor that sets the full-scale DAC current. This resistor is returned to DACVSS. The typical value for this resistor is 135 Ω.

10.3 Display Memory Interface

Name	Type	Description
RAS1#	O	**ROW ADDRESS STROBE#:** This active-low output latches the row address from MA[8:0] into the first bank of display memory. See Appendix B3, "Memory Configurations and Timing", for detailed information regarding display memory bank definition. This pad, and those for the other DRAM controls, are matched for one to four loads. If eight or more DRAMs are connected to any lines, damping resistors may be required to control edge rates and undershoot.
RAS0#	O	**ROW ADDRESS STROBE#:** This active-low output latches the row address from MA[8:0] into the second bank of display memory. See Appendix B3.
CAS[7:0]#	O	**COLUMN ADDRESS STROBE[7:0]#:** These active-low outputs latch the Column Address from MA[8:0] into the DRAMs. These pins must be connected to the CAS# pins of all the DRAMs in the display memory array. See Appendix B3.
WE#	O	**WRITE ENABLE#:** This active-low output controls the Write Enable inputs of the DRAMs. This pin must be connected to the WE# pins of the DRAMs. See Appendix B3.
MA[9:0]	O	**MEMORY ADDRESS [9:0]:** These pins supply the multiplexed address to the DRAMs. These pins must be connected to the address pins of the DRAMs. See Appendix B3.
MD[63:0]	BIO	**MEMORY DATA [63:0]:** These bidirectional pins transfer data between the CL-GD5446 and display memory. These pins must be connected to the data pins of the DRAMs. See Appendix B3. MD[63:48] are forced into high-impedance when RST# is active; this allows the configuration pull-down resistors to override the weak pull-ups and load into the Configuration register (CF). MD[7:0] are also the BIOSD[7:0] data inputs.

10.4 General-Purpose I/O

Name	Type	Description
GPD[15:0]	I/O	**GENERAL PURPOSE DATA [15:0]:** This bidirectional bus transfers data between an additional peripheral device and the CL-GD5446. These pins are redefined from BIOSA pins. GPD[15:8] are also shared with PIXD[15:8].
GPA[3:0]	I/O	**GENERAL PURPOSE ADDRESS [3:0]:** This is the general-purpose I/O address bus. These pins are defined as GPA[5:2] or GPA[3:0] depending on how GPI/O is configured. See Appendix B11, "General-Purpose I/O". These pins are redefined from a number of pins.
GPCS#	O	**GENERAL PURPOSE CHIP SELECT#:** This pin is the active-low peripheral chip select for the general-purpose I/O port. See Appendix B11 for detailed timing information.
GPIORD#	I/O	**GENERAL PURPOSE IO READ#:** This pin is the active-low read command to the peripheral chip on the general-purpose I/O port. This pin is redefined from BIOSA[8].
GPIOWR#	I/O	**GENERAL PURPOSE IO WRITE#:** This pin is the active-low write command to the peripheral chip on the general-purpose I/O port. This pin is redefined from BIOSA[9].
GPRDY/DT	I/O	**GENERAL PURPOSE READY/DTACK:** This pin is the output from the peripheral chips that indicate a cycle on the general-purpose I/O port is complete. This pin is redefined from TWR#. The peripheral device must not pull this pin low when RST# is active.

10.5 V-Port™

Name	Type	Description
PIXD[15:8]	I/O	**PIXEL DATA [15:8]:** This is the high-order byte of the pixel data input bus. These pins are used only if the V-Port is configured for 16 bits (CR50[4]). If these pins are for V-Port data, GPIO cannot be configured for 16-bit data.
PIXD[7:0]	I/O	**PIXEL DATA [7:0]:** This is the low-order byte of the pixel data input bus.
VACT	I/O	**VIDEO ACTIVE:** This input indicates that PIXCLK is accompanied by valid data. This pin is redefined from EVIDEO#.
VREF	I	**VERTICAL REFERENCE:** A falling edge on this input indicates the end of the current capture field and the beginning of the next capture field. The sense of HREF at this edge indicates whether the current capture field is defined as odd or even. This pin is redefined from EDCLK#.
PIXCLK	I/O	**PIXEL CLOCK:** This clock is the reference for data on the PIXD bus. This clock can be programmed to capture data on either or both edges. This pin is redefined from DCLK.
HREF	I/O	**HORIZONTAL REFERENCE:** An active edge of this input indicates the end of the current capture scanline and the beginning of the next. The active edge is programmable. This pin is redefined from BLANK#.

10.6 Miscellaneous Pins

Name	Type	Description
DDCCLK	OC/I	**DDC CLOCK:** This pin is the DDC clock pin. The CL-GD5446 can drive this pin low when SR8[0] is '0'. The level on this pin can be sensed in SR8[2].
DDCDAT	OC/I	**DDC DATA:** This pin is the DDC data pin. The CL-GD5446 can drive this pin low when SR8[1] is '0'. The level on this pin can be sensed in SR8[7].
EROM#	O	**ENABLE ROM#:** This pin is driven low when PCI30[0] is '1' and the address of the EPROM specified in PCI30 is on the bus. This allows the EPROM to drive data into MD[7:0], and thence to the PCI data bus.
OVRW#	0	**OVERLAY WINDOW#:** This signal provides timing for CL-GD543X/'4X–compatible overlay functions. This pin is also one of the general-purpose I/O address bus.
TWR#	I/O	**TWR#:** This pin must be pulled up.

10.7 Clock Synthesizer

Name	Type	Description
OSC	I	**OSCILLATOR:** This is the reference input for the frequency synthesizers. This pin can be used with XTAL for a two-pin crystal controlling the internal oscillator or it can be driven with an external reference. The nominal frequency is 14.31818 MHz.
XTAL	O	**CRYSTAL:** This pin can drive the reference crystal when the on-chip oscillator is being used. This pin is also for the general-purpose I/O port address bus (bit 6).
MCLK	I/O	**MCLK:** This pin can be configured as the MCLK VCO output, the VCLK VCO output, Programmable Output 0, or the MCLK input (factory testing only).

10.8 Power and Ground

Name	Type	Description
VDD[6:1]	Power	**POWER (Logic):** These six pins supply +5 volts to the digital logic of the CL-GD5446. Each pin must be connected to the power cutout in the power plane and described in Appendix B2, "PCI Bus Reference Design". Each pin must be bypassed to ground with a 0.1-µF capacitor, with proper high-frequency characteristics, placed as close to the pin as possible.
VSS[13:1]	Ground	**GROUND (Logic):** These 13 pins supply the ground reference to the digital logic in the CL-GD5446. Each pin must be connected to the ground plane.
VCLKVDD	Power	**POWER (VCLK):** This pin supplies +5 volts to the VCLK synthesizer. The supply to this pin must be isolated with an RC filter as described in Appendix B2. The filter capacitors on this pin must be returned to the VCLKVSS cutout.
VCLKVSS	Ground	**GROUND (VCLK):** This pin supplies the ground reference to the VCLK synthesizer. This pin must be connected to VCLKVSS cutout described in Appendix B2.
MCLKVDD	Power	**POWER (MCLK):** This pin supplies +5 volts to the MCLK synthesizer. The supply to this pin must be isolated with an RC filter as described in Appendix B2. The filter capacitors on this pin must be returned to the MCLKVSS cutout.
MCLKVSS	Ground	**GROUND (MCLK):** This pin supplies the ground reference to the MCLK synthesizer. This pin must be connected to MCLKVSS cutout described in Appendix B2.
DACVDD	Power	**POWER (DAC):** These two pins supply +5 volts to the palette DAC. These pins must be connected directly to the power plane and bypassed to DACVSS with a 0.1-µF capacitor.
DACVSS	Ground	**GROUND (DAC):** These two pins supply the ground reference to the palette DAC. These pins must be connected to the DACVSS cutout described in Appendix B2.

Electrical Specifications

11. ELECTRICAL SPECIFICATIONS

11.1 Absolute Maximum Ratings

Ambient temperature under bias..... 0°C to 70°C

Storage temperature.....–65°C to 150°C

Voltage on any pin ..... V_{SS} –0.5 V to V_{CC} + 0.5 V

Operating power dissipation ..... 2.0 W

Power supply voltage ..... 7 V

Injection current (latch-up testing)..... 100 mA

CAUTION: Stresses above those listed may cause permanent damage to system components. These are stress ratings only. Functional operation at these or any conditions above those indicated in the operational ratings of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect system reliability.

11.2 DC Specifications (Digital)

(V_{CC} = 5 V ± 5%, T_A = 0°C to 70°C, unless otherwise specified)

Symbol	Parameter	MIN	MAX	Units	Test Conditions
V_{CC}	Power supply voltage	4.75	5.25	V	Normal operation
V_{IL}	Input low voltage	0	0.8	V	
V_{IH}	Input high voltage	2.0	VCC + 0.5	V	
V_{OL}	Output low voltage	–	0.5	V	I_{OL} = 4 mA[a]
V_{OH}	Output high voltage	2.4	–	V	I_{OH} = 400 µA[b]
I_{CC}	Supply current	–	tbd	A	V_{CC} nominal[c]
I_{IH}	Input high current	–	10	µA	$V_{IN} = V_{CC}$
I_{IL}	Input low current	–10	–	µA	V_{CC} = 5.25; V_{IN} = 0
I_{IHP}	Input high current (pull-up)	–10	10	µA	$V_{IN} = V_{CC}$
I_{ILP}	Input low current (pull-up)	–45	12	µA	V_{CC} = 5.25; V_{IN} = 0
I_{OZ}	Input leakage	–10	10	µA	$0 < V_{IN} < V_{CC}$
C_{IN}	Input capacitance	–	10	pF[d]	
C_{OUT}	Output capacitance	–	10	pF[d]	
IREF	DAC current reference	–3	–10	mA	Nominal IREF is –6.67 mA

[a] I_{OL} is specified for a standard buffer. See Chapter 10, “Detailed Pin Descriptions”, for further information.

[b] I_{OH} is specified for a standard buffer. See Chapter 10 for further information.

[c] I_{CC} is measured with VCLK at 135 MHz and MCLK at 80 MHz.

[d] This is not 100% tested, but is periodically sampled.

11.3 DAC Characteristics

Symbol	Parameter	MAX	Units	Test Conditions	Notes
R	Resolution	8	bits		
IO	Output current	30	mA	VO < 1V	
TR	Analog output rise/fall time	3	ns	10% to 90% full scale	1, 2
TS	Analog output settling time	15	ns	50% FS to remaining within 2%	1, 2
TSK	Analog output skew	tbd	ns		1, 2, 3, 4
FDT	DAC-to-DAC correlation	2.5	%		1, 2, 3, 4
GI	Glitch impulse	tbd	pV/sec		2
IL	Integral linearity	1.5	LSB		
DL	Differential linearity	1.5	LSB		2

NOTES:

1) Load is 50 Ω and 30 pF per analog output.
2) IREF = –6.67 mA.
3) Outputs loaded identically.
4) About the mid-point of the distribution of the three DACs measured at full-scale output.

11.4 AC Specifications

Table 11-1. CLK Timing (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	Rise time (CLK) PCI bus	–	4	ns
t_2	Fall time (CLK) PCI bus	–	4	ns
t_3	High period (CLK) PCI bus	40	60	% t_5
t_4	Low period (CLK) PCI bus	40	60	% t_5
t_5	Period (CLK) PCI bus	30	–	ns

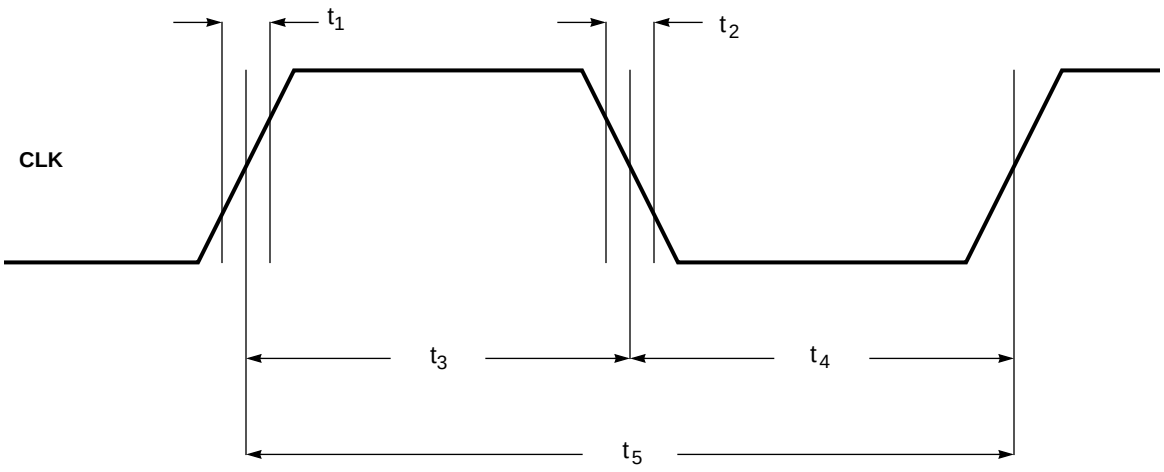


Figure 11-1. CLK Timing (PCI Bus)

Table 11-2. FRAME#, DEVSEL#, AD[31:0], and C/BE[3:0]# (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	FRAME# setup to CLK	7	–	ns
t_2	AD[31:0] (Address) setup to CLK	7	–	ns
t_3	AD[31:0] (Address) hold from CLK	0	–	ns
t_4	AD[31:0] (Data) setup to CLK	7	–	ns
t_5	AD[31:0] (Data) hold from CLK	0	–	ns
t_6	AD[31:0], C/BE[3:0]# high-impedance from CLK	0	28	ns
t_7	C/BE[3:0]# (bus CMD) setup to CLK	7	–	ns
t_8	C/BE[3:0]# (bus CMD) hold from CLK	0	–	ns
t_{8a}	C/BE[3:0]# (byte enable) setup to CLK	7	–	ns
t_9	DEVSEL# delay from CLK	2	11	ns
t_{10}	DEVSEL# high before high-impedance	1	–	CLK

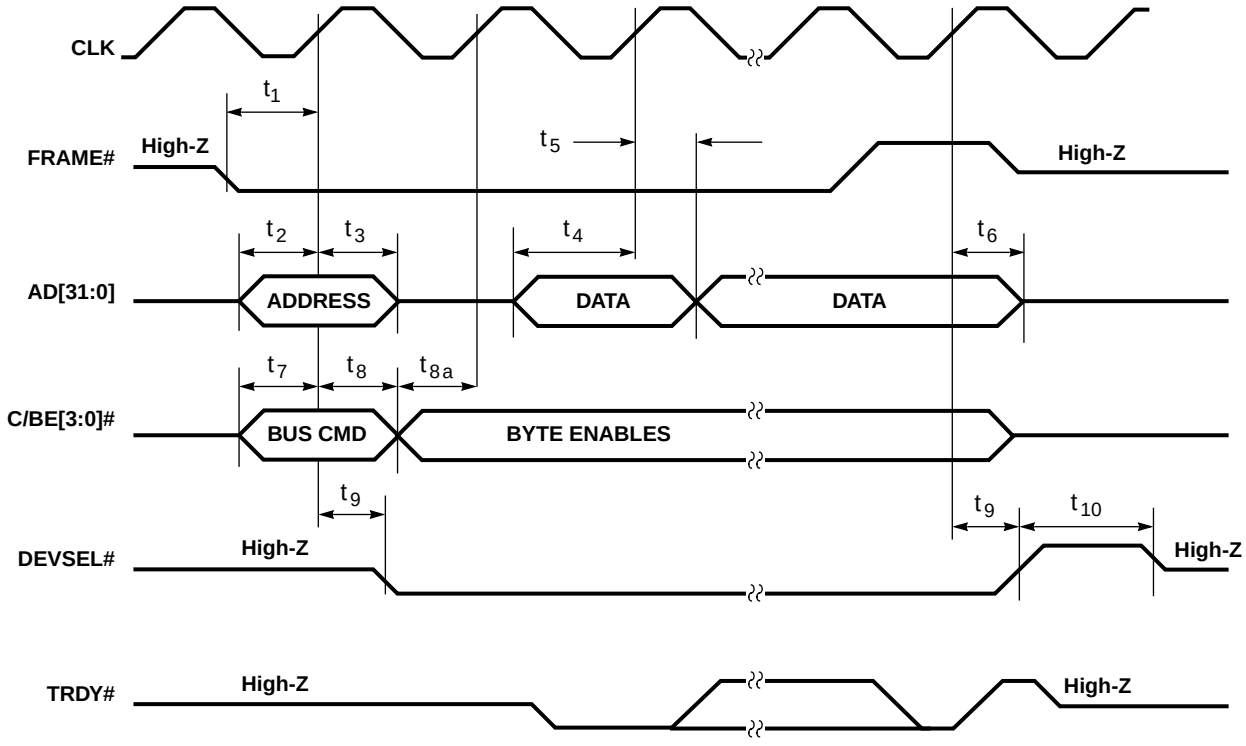
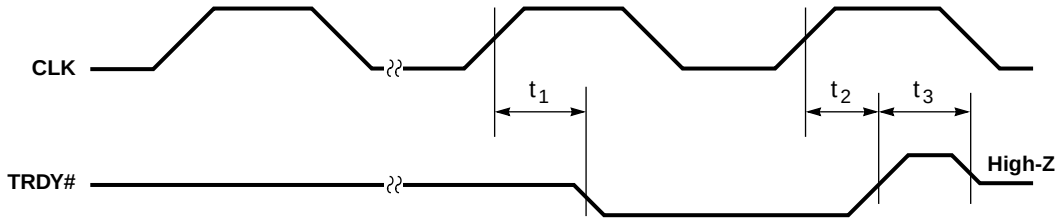


Figure 11-2. FRAME#, DEVSEL#, AD[31:0], and C/BE[3:0]# (PCI Bus)

Table 11-3. TRDY# Delay (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	TRDY# active delay from CLK	2	11	ns
t_2	TRDY# inactive delay from CLK	2	11	ns
t_3	TRDY# high before high-impedance	1	–	CLK



High-Z = High-impedance

Figure 11-3. TRDY# Delay (PCI Bus)

Table 11-4. Ready Data/ TRDY# (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	Read data setup to CLK while TRDY# active	7	–	ns
t_2	Read data hold from CLK while TRDY# active	0	–	ns
t_3	IRDY# setup to CLK	7	–	ns
t_4	IRDY# hold from CLK	0	–	ns

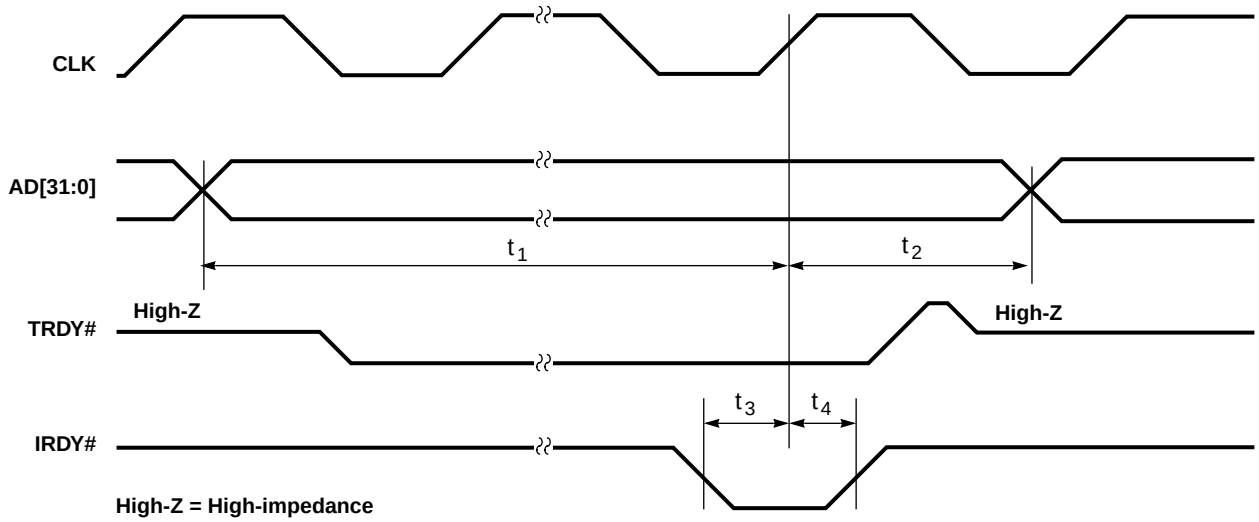
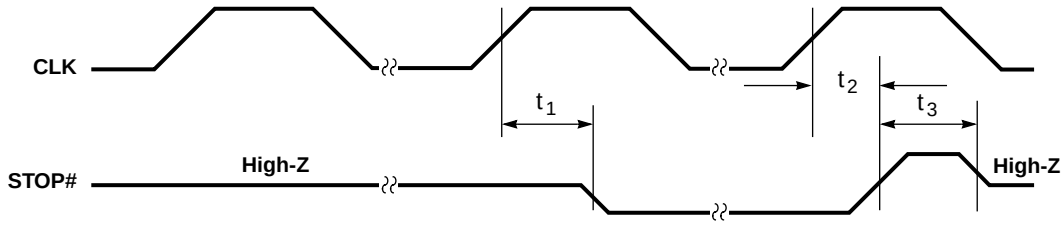


Figure 11-4. Ready Data/ TRDY# (PCI Bus)

Table 11-5. STOP# Delay (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	STOP# active delay from CLK	2	11	ns
t_2	STOP# inactive delay from CLK	2	11	ns
t_3	STOP# high before high-impedance	1	–	CLK

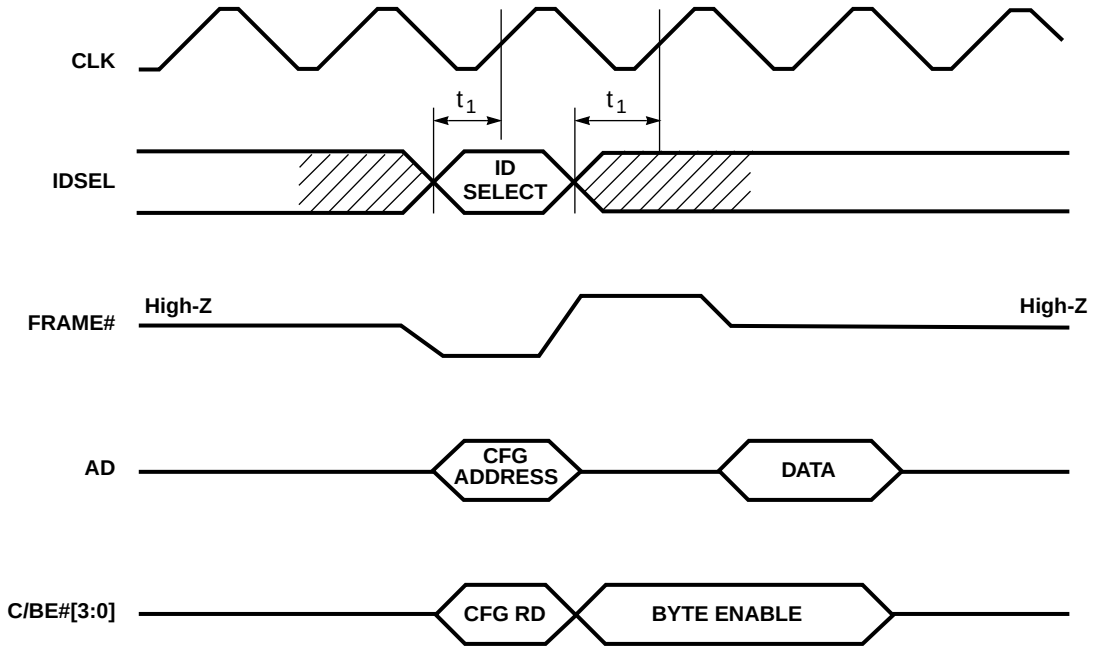


High-Z = High-impedance

Figure 11-5. STOP# Delay (PCI Bus)

Table 11-6. IDSEL Timing (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	IDSEL setup to CLK	–	15	ns

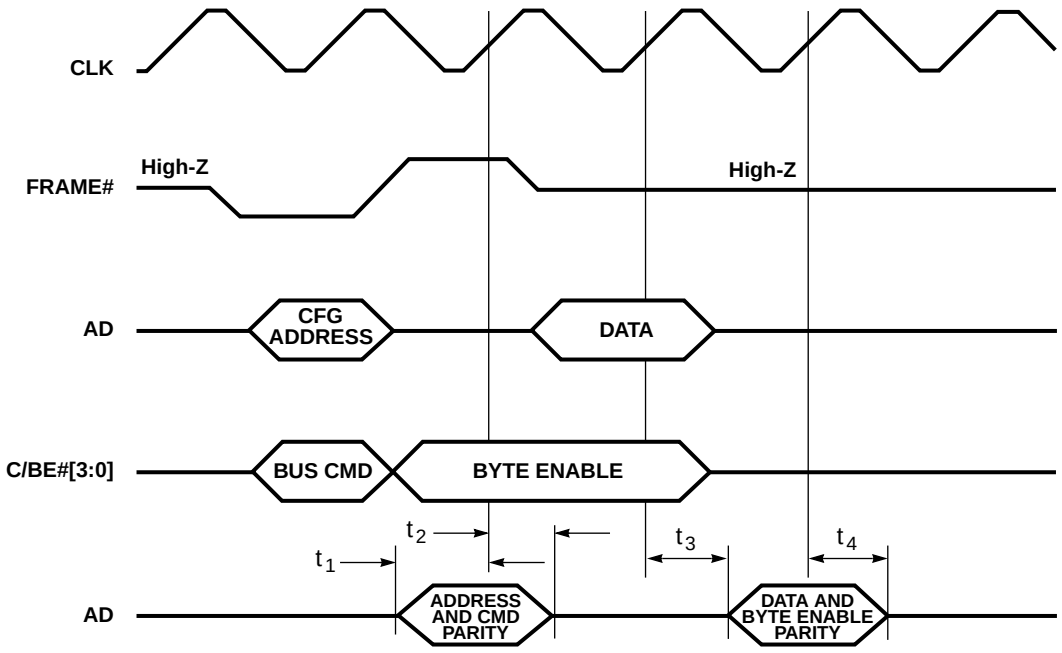


High-Z = High-impedance

Figure 11-6. IDSEL Timing (PCI Bus)

Table 11-7. PAR Timing (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	PAR setup from CLK (input to CL-GD5446)	7	–	ns
t_2	PAR hold from CLK (input to CL-GD5446)	0	–	ns
t_3	PAR delay from CLK (output from CL-GD5446)	2	11	ns
t_4	PAR hold from CLK (output from CL-GD5446)	0	–	ns

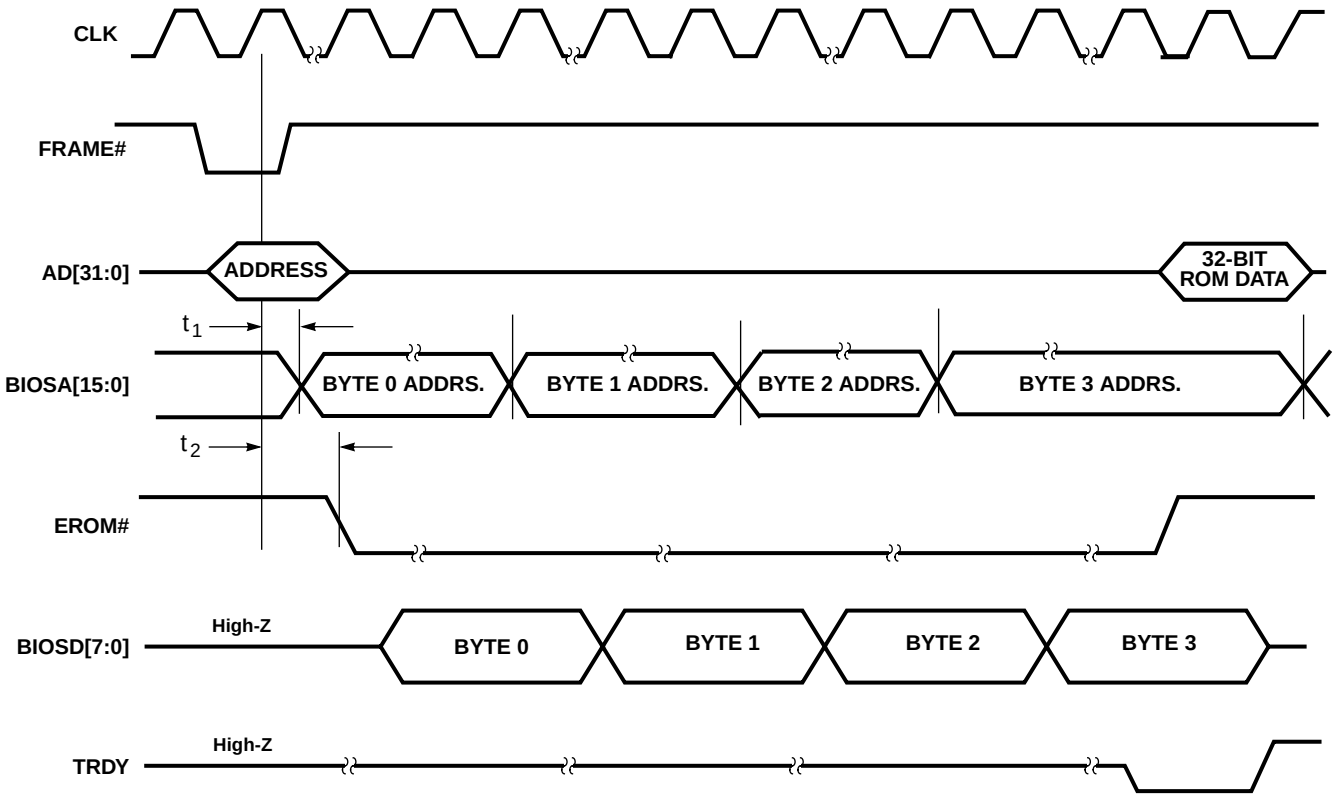


High-Z = High-impedance

Figure 11-7. PAR Timing (PCI Bus)

Table 11-8. EROM#, BIOSA[15:0] Timing (PCI Bus)

Symbol	Parameter	MIN	MAX	Units
t_1	BIOSA[15:0] delay from CLK	–	15	ns
t_2	EROM# delay from CLK	–	15	ns



High-Z = High-impedance

Figure 11-8. EROM#, BIOSA[15:0] Timing (PCI Bus)

Table 11-9. Display Memory Bus: CAS#-before-RAS# Refresh Timing [a]

Symbol	Parameter	MIN	MAX
t_1	t_{CSR}: CAS# active setup to RAS# active	1 m [b]	–
t_2	t_{RAS}: RAS# low pulse width	4 m	–
t_3	t_{RP}: RAS# high pulse width	3 m	–

[a] There are either one, three, or five RAS# pulses while CAS# remains low.

[b] m = MCLK

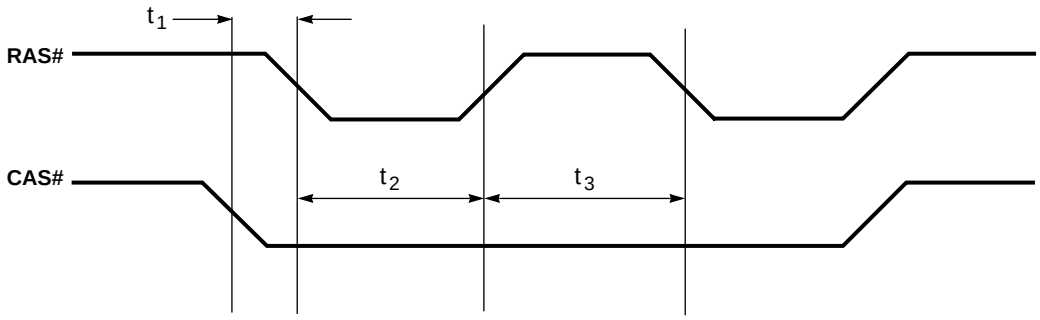


Figure 11-9. Display Memory Bus: CAS#-before-RAS# Refresh Timing

Table 11-10. Display Memory Bus: Common Parameters (No EDO)

Symbol	Parameter	MIN	MAX
t_1	t_{ASR}: Address setup to RAS# active	1.5 m – 9 ns [a]	–
t_2	t_{RAH}: Row Address hold from RAS# active	1.5 m – 5 ns	–
t_3	t_{ASC}: Address Setup to CAS# active	1 m – 3 ns	–
t_4	t_{CAH}: Column Address hold from CAS# active	1 m	–
t_5	t_{RCD}: RAS# active to CAS# active delay (standard RAS)	2.5 m – 7.5 ns	–
t_5	t_{RCD}: RAS# active to CAS# active delay (extended RAS)	3 m	–
t_6	t_{RAS}: RAS# pulse width low (standard RAS)	3.5 m	–
t_6	t_{RAS}: RAS# pulse width low (extended RAS)	4 m – 1 ns	–
t_7	t_{RP}: RAS# precharge (RAS# pulse width high — standard RAS)	2.5 m – 2 ns	–
t_7	t_{RP}: RAS# precharge (RAS# pulse width high — extended RAS)	3 m – 1.5 ns	–
t_8	t_{CAS}: CAS# pulse width low	1 m + 3 ns	1 m + 6 ns
t_9	t_{CP}: CAS# precharge (CAS# pulse width high)	1 m – 6 ns	1 m – 3 ns
t_{10}	t_{RC}: Random cycle (standard RAS)	6 m	–
t_{10}	t_{RC}: Random cycle (extended RAS)	7 m	–
t_{11}	t_{PC}: Page mode cycle	2 m	–

[a] m = MCLK

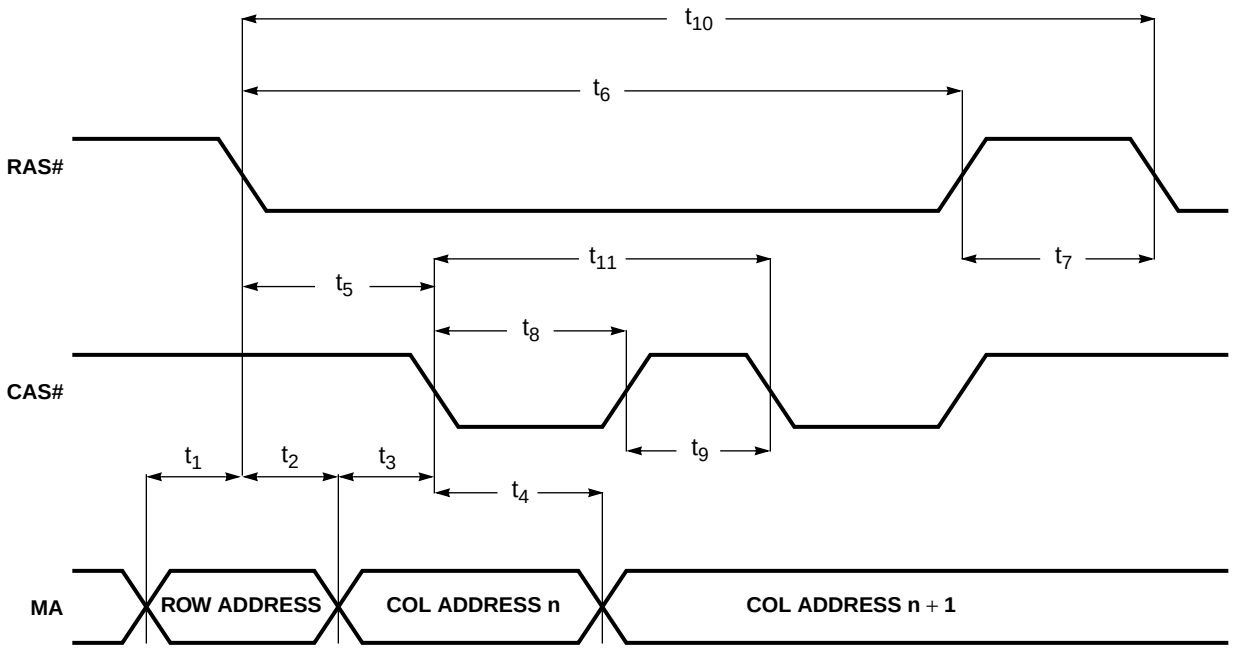


Figure 11-10. Display Memory Bus: Common Parameters (No EDO)

Table 11-11. Display Memory Bus: Read Cycles

Symbol	Parameter	MIN	MAX
t_1	Read data setup to CAS# rising edge	0	–
t_2	Read data hold from CAS# high	10 ns	–

Only parameters t_1 and t_2 are defined for the CL-GD5446. The remaining parameters in this table are calculated from parameters from this table and the previous table. These parameters are provided so that system designers can easily determine DRAM requirements.

t_3	DRAM access time from RAS# (standard RAS)	–	3.5 m – 1 ns
t_3	DRAM access time from RAS# (extended RAS)	–	4 m – 1 ns
t_4	DRAM access time from Column Address	–	2 m
t_5	DRAM access time from CAS# active	–	1 m + 3 ns
t_6	DRAM access time from CAS# precharge	–	2 m

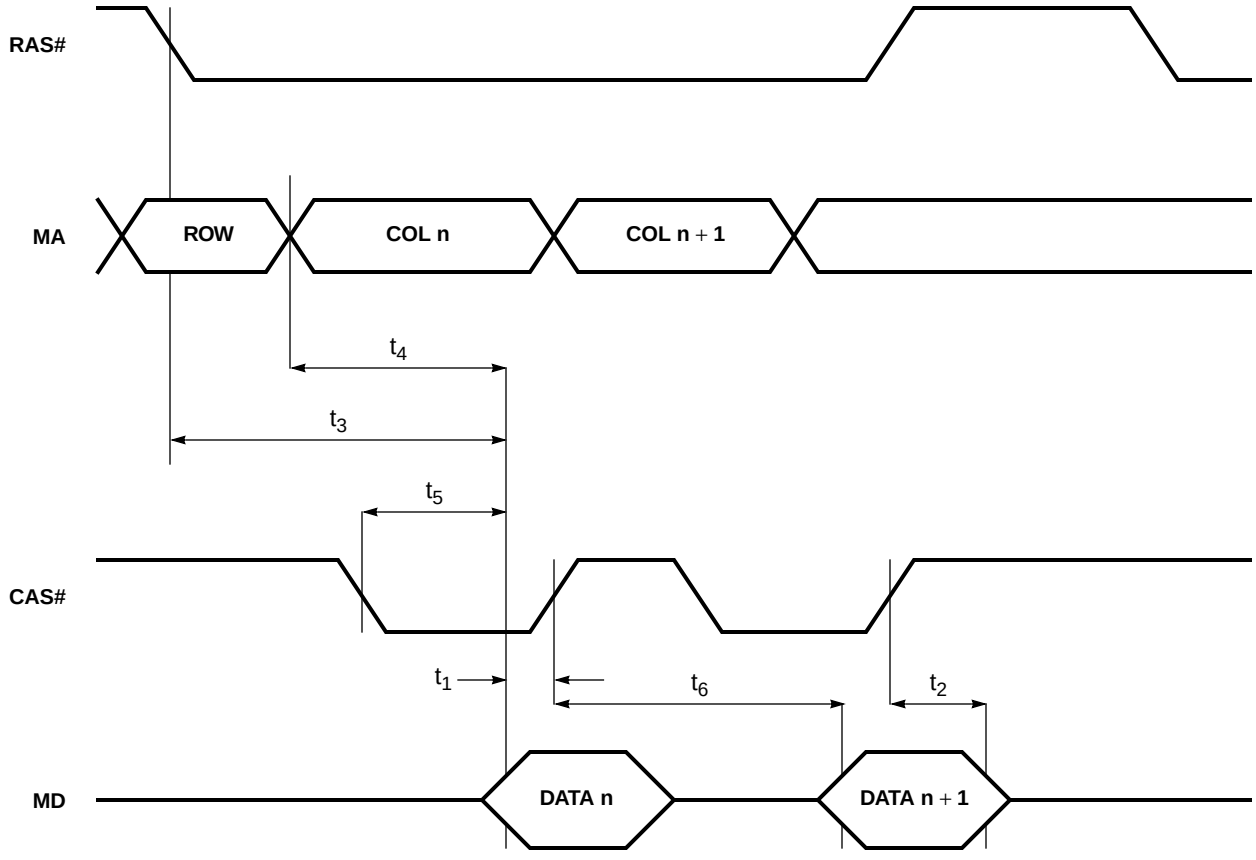


Figure 11-11. Display Memory Bus: Read Cycles

Table 11-12. Display Memory Bus: Write Cycles

Symbol	Parameter	MIN	MAX
t_1	t_{CWL}: WE# active setup to CAS# active	1 m + 0.5 ns [a]	–
t_2	t_{DS}: Write data setup to CAS# active	1 m – 2 ns	1 m + 2 ns
t_3	t_{DH}: Write data hold from CAS# active	1 m + 1 ns	–
t4	t_{WCH}: WE active hold from CAS# active	1.5 m – 2 ns	–

[a] m = MCLK

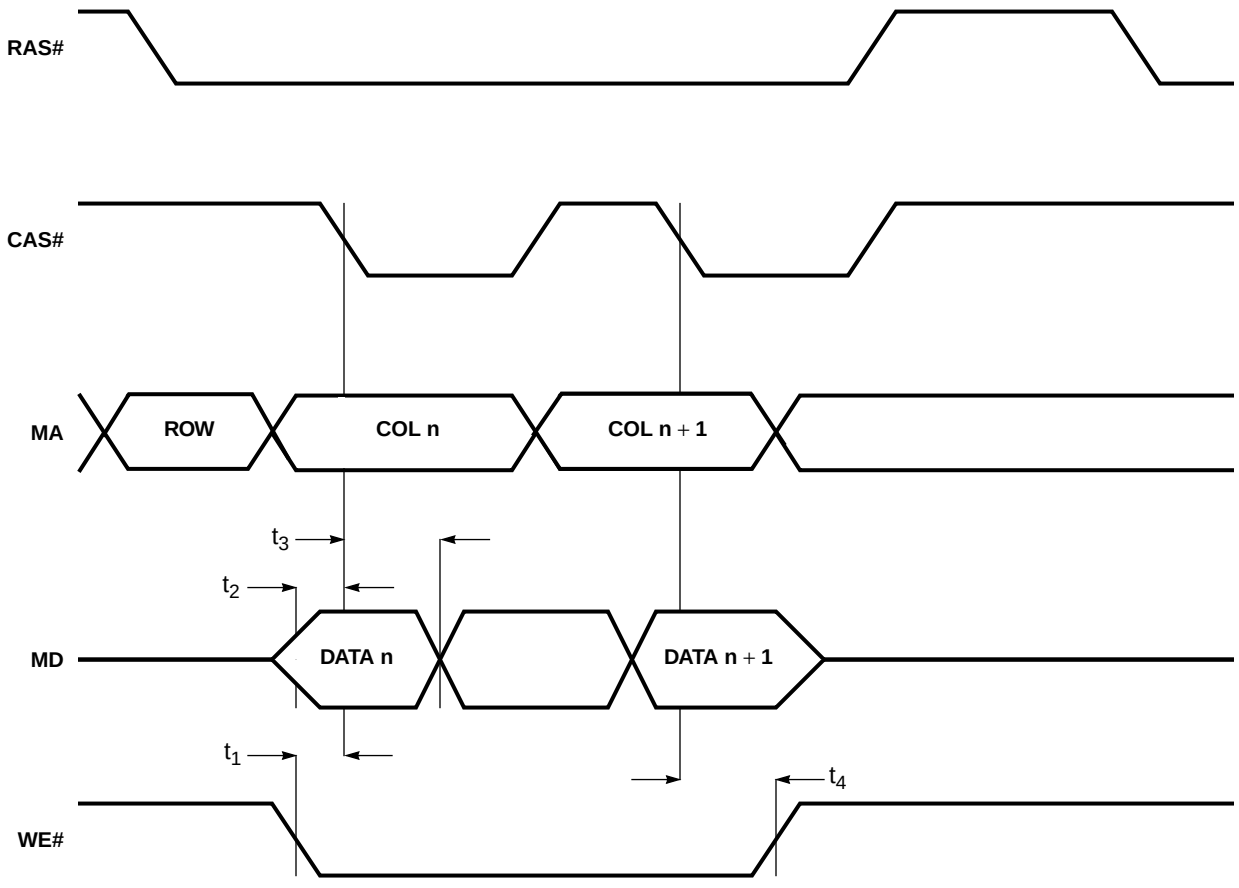


Figure 11-12. Display Memory Bus: Write Cycles

Table 11-13. Display Memory Bus: Common Parameters (EDO Timing)

Symbol	Parameter	MIN	MAX
t_1	t_{ASR}: Address setup to RAS# active	1.5 m – 9 ns [a]	–
t_2	t_{RAH}: Row Address hold from RAS# active	1.5 m – 5 ns	–
t_3	t_{ASC}: Address setup to CAS# active	1 m – 3 ns	–
t_4	t_{CAH}: Column Address hold from CAS# active	1 m	–
t_5	t_{RCD}: RAS# active to CAS# active delay (EDO timing)	4 m	–
t_6	t_{RAS}: RAS# pulse width low (EDO timing)	5 m – 1 ns	–
t_7	t_{RP}: RAS# precharge (RAS# pulse width high) (EDO timing)	3 m + 1.5 ns	–
t_8	t_{CAS}: CAS# pulse width low	1 m	–
t_9	t_{CP}: CAS# precharge (CAS# pulse width high)	1 m – 4.5 ns	–
t_{10}	t_{RC}: Random cycle (EDO timing)	8 m	–
t_{11}	t_{PC}: Page mode cycle	2 m	–
t_{12}	t_{CAS}: CAS# pulse width low (last CAS# of Page mode read burst)	3 m	–
t_{13}	t_{CRP}: CAS# to RAS# precharge	1 m	–

[a] m = MCLK

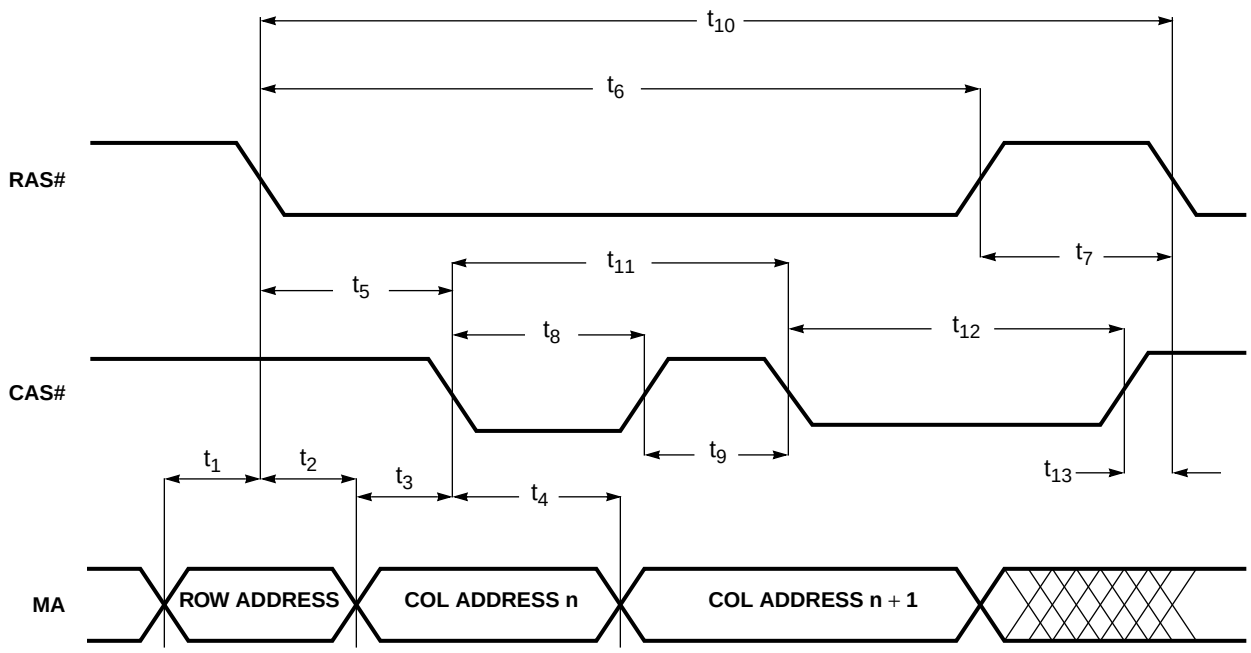


Figure 11-13. Display Memory Bus: Common Parameters (EDO Timing)

Table 11-14. Display Memory Bus: Read Cycles (EDO Timing)

Symbol	Parameter	MIN	MAX
t_1	Read data setup to CAS# falling edge	1 ns	–
t_2	Read data hold from CAS# falling edge	5 ns	–

The timing shown in this table and figure are for EDO mode DRAM timing. See the description of register GR18[2]. This timing requires the display memory be populated with EDO DRAMs. These devices hold their read data valid past the rising edge of CAS#.

Depending on the cycle being executed, the read data setup and hold times may be defined with respect to CAS# falling edge (Page mode cycle other than last), WE# falling edge (read followed by write), or a timing term that is available only internally (Last Page mode cycle). If the DRAM can meet the timing requirements of the falling CAS# edge case, it meets the other two cases.

Only parameters t_1 and t_2 are defined for the CL-GD5446. The remaining parameters in this table are calculated from parameters from this table and the previous table. These parameters are provided so that system designers can easily determine DRAM requirements.

t_3	DRAM access time from RAS# (EDO timing)	–	5 m
t_4	DRAM access time from column address (EDO timing)	–	3 m
t_5	DRAM access time from CAS# active (EDO timing)	–	2 m
t_6	DRAM access time from CAS# precharge (EDO timing)	–	3 m

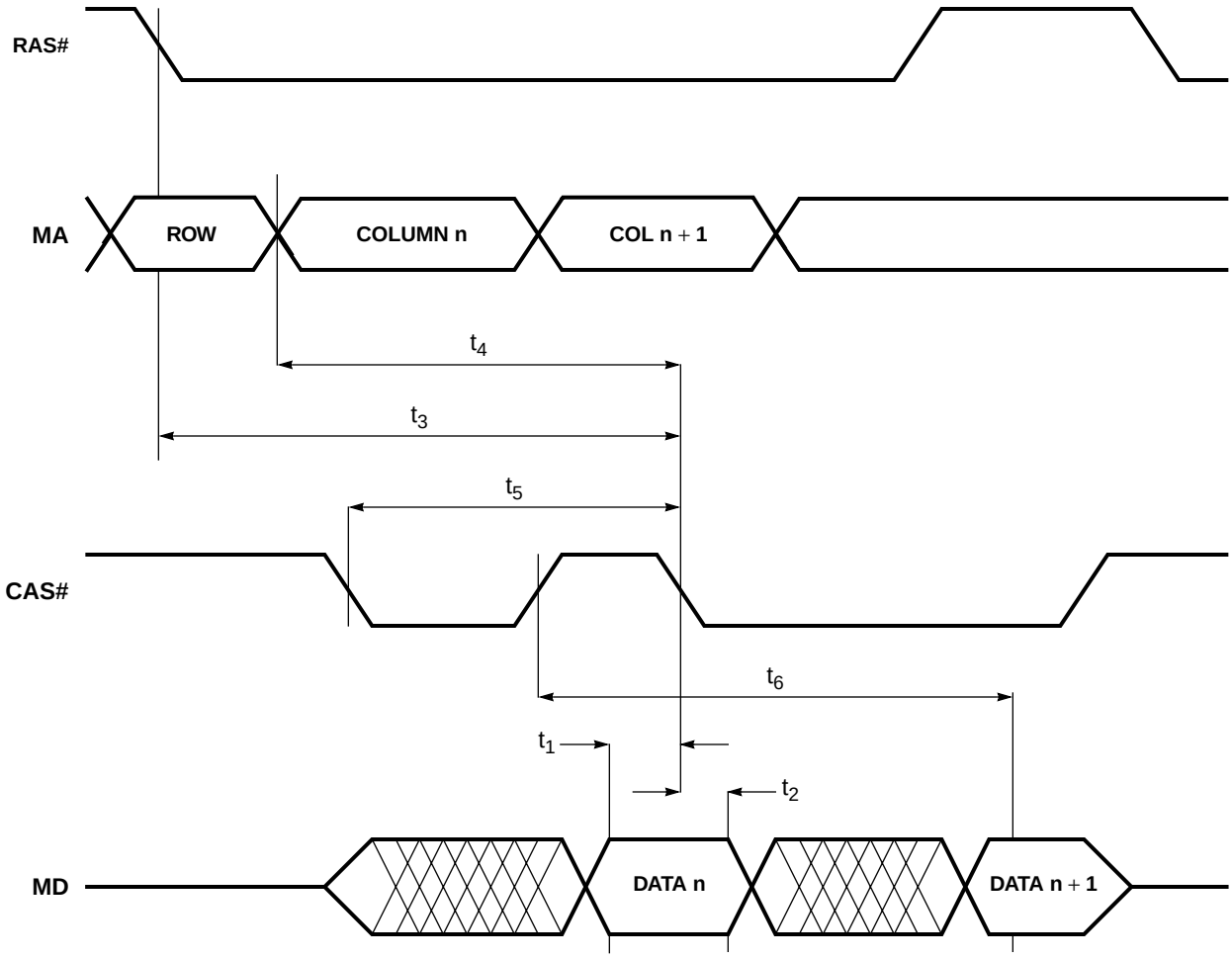


Figure 11-14. Display Memory Bus: Read Cycles (EDO Timing)

Table 11-15. Display Memory Bus: BitBLT Read/Write Cycle

Symbol	Parameter	Nominal	
		Non-EDO GR18[2] = 0	EDO GR18[2] = 1
t_1	Write data active delay from CAS# inactive (GR18[0] = 0)	1 m [a]	1 m
t_1	Write data active delay from CAS# inactive (GR18[0] = 1)	0.5 m	0.5 m
t_2	t_{CP}: read CAS# to write CAS# delay (GR18[1] = 0)	3 m	4 m
t_2	t_{CP}: read CAS# to write CAS# delay (GR18[1] = 1)	2 m	3 m
t_3	WE# active delay from CAS# inactive	1 m	1 m

[a] m = MCLK

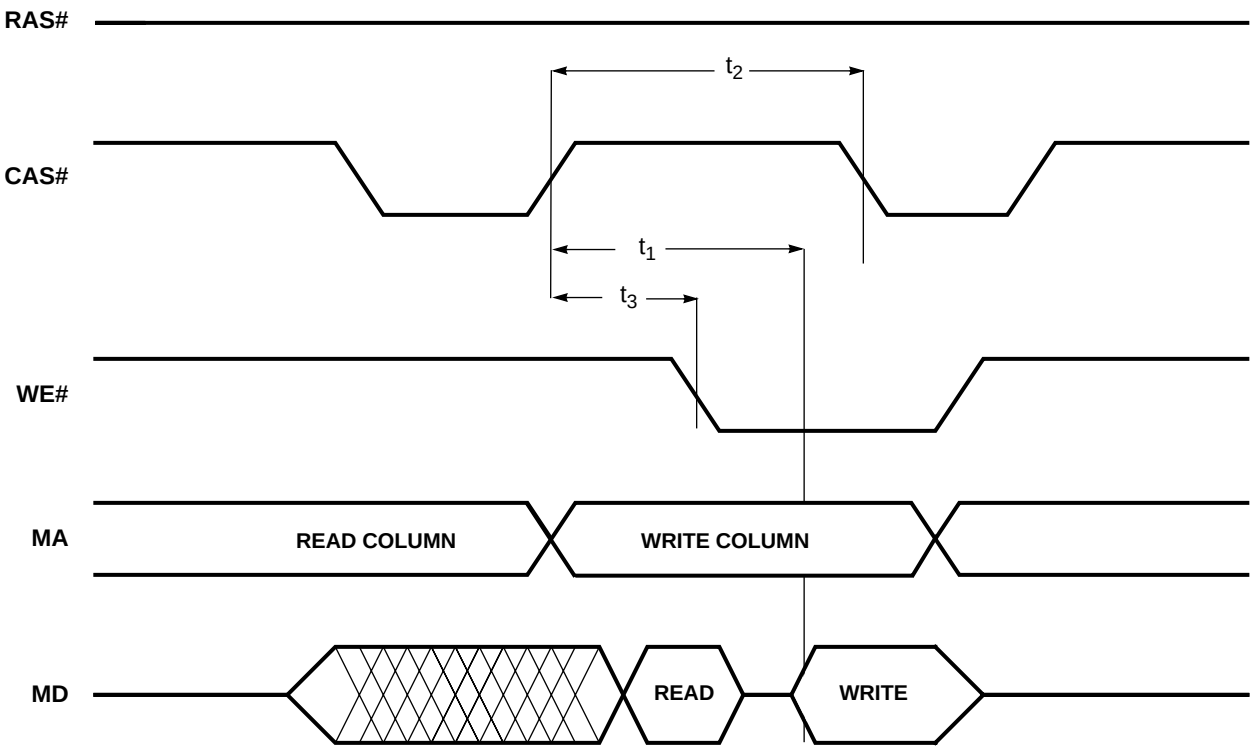


Figure 11-15. Display Memory Bus: BitBLT Read/Write Cycle

Table 11-16. P-Bus as Inputs: 8-Bit Mode (External DCLK)

Symbol	Parameter	MIN	MAX	Units
t_1	P[7:0], BLANK# setup to DCLK	0	–	ns
t_2	P[7:0], BLANK# hold from DCLK	6	–	ns

NOTE: The CL-GD5446 RAMDAC is driven externally. For CL-GD543X/'4X Overlay modes, BLANK# is an output.

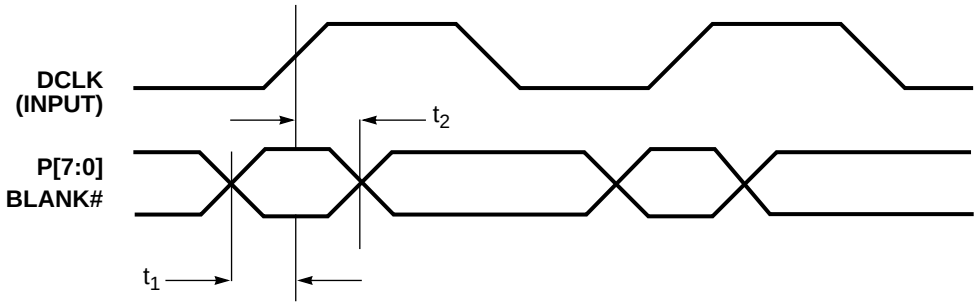


Figure 11-16. P-Bus as Inputs: 8-Bit Mode (External DCLK)

Table 11-17. Feature Bus Timing: 8-bit Mode, Output (Internal DCLK)

Symbol	Parameter	MIN	MAX	Units
t_1	DCLK to BLANK# delay	–1	1	ns
t_2	DCLK to HSYNC, VSYNC delay	1	3	ns
t_3	DCLK to P[7:0] delay	–2	0	ns
t_4	DCLK to OVRW# delay	–1	1	ns

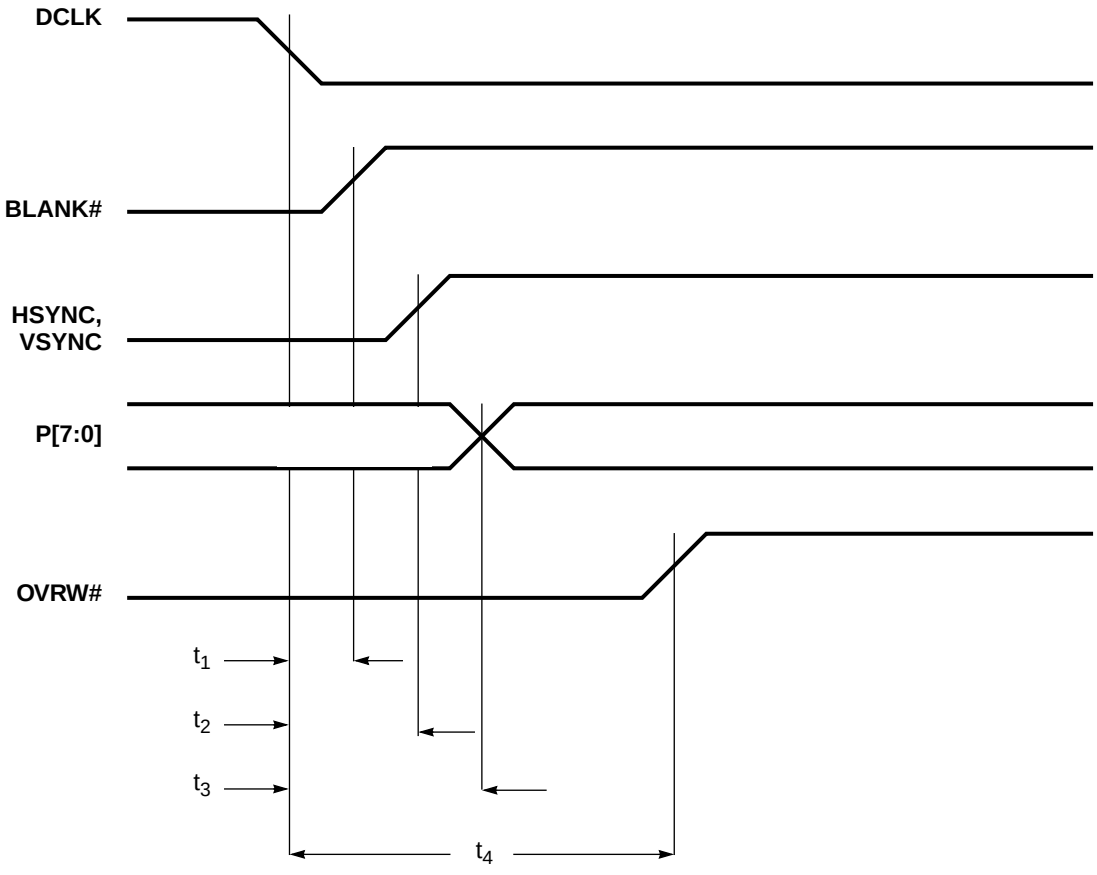


Figure 11-17. Feature Bus Timing: 8-bit Mode, Output (Internal DCLK)

Table 11-18. V-Port™: PIXCLK Timing

Symbol	Parameter	MIN	MAX	Units
t_1	PIXD[15:0], VACK setup to PIXCLK edge[a]	tbd	–	ns
t_2	PIXD[15:0], VACK hold from PIXCLK edge	tbd	–	ns

[a] PIXD and VACT can be clocked on either or both PIXCLK edges, according to the programming of CR50[5, 3].

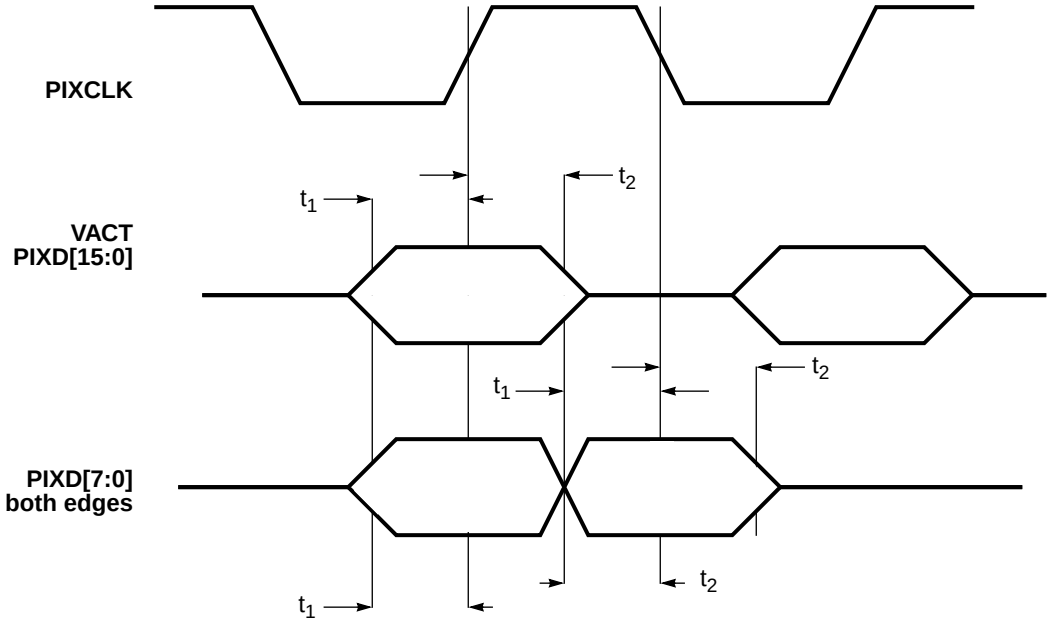


Figure 11-18. V-Port™: PIXCLK Timing

Table 11-19. V-Port™ Timing: VREF[a]

[a] The terms odd and even are reference only. The sense can be inverted with CR58[6].

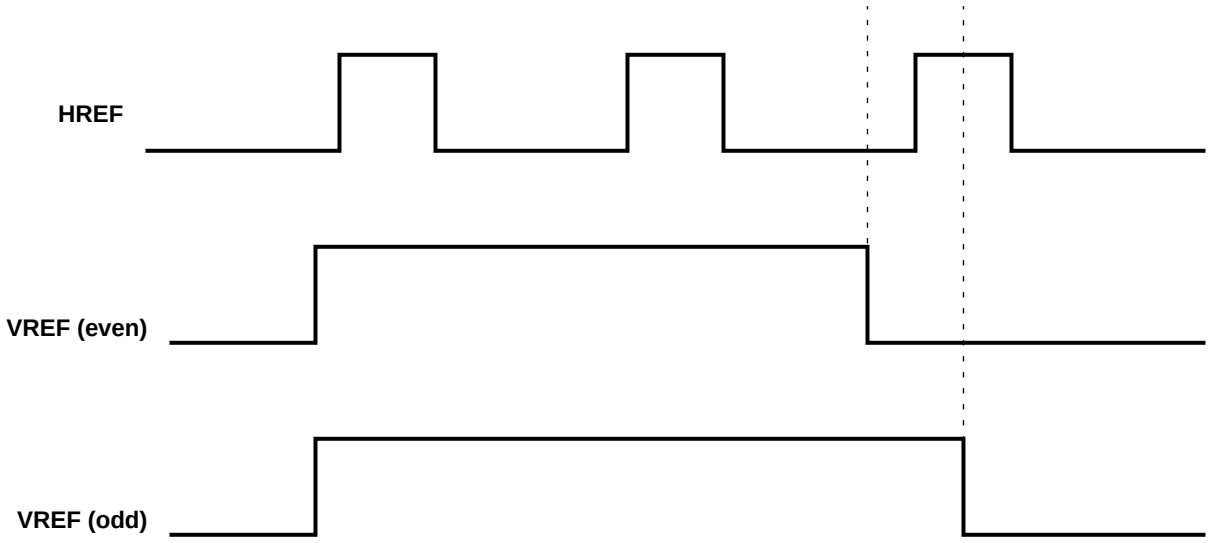


Figure 11-19. V-Port™ Timing: VREF

Table 11-20. Reset TimingReset Timing

Symbol	Parameter	MIN	MAX	Units
t_1	RST# pulse width	12	–	MCLK
t_2	MD[63:48] setup to RST# rising edge	2	–	ns
t_3	MD[63:48] hold from RST# rising edge	25	–	ns
t_4	RST# inactive to first command	12	–	MCLK

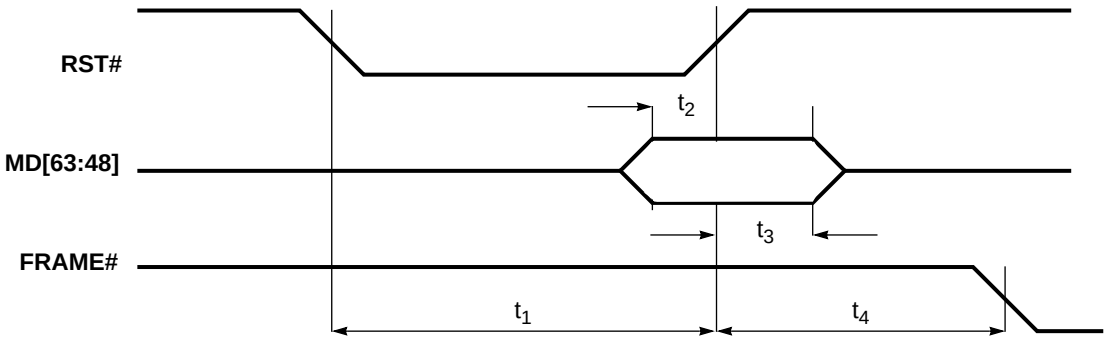


Figure 11-20. Reset Timing

Appendix A1

Connector Pinouts

CONNECTOR PINOUTS

Table A1-1. VGA DB15

Pin Number	Standard VGA	DDC1	DDC2B
1	Analog RED	Analog RED	Analog RED
2	Analog GREEN	Analog GREEN	Analog GREEN
3	Analog BLUE	Analog BLUE	Analog BLUE
4	Monitor ID 2	Monitor ID 2	Monitor ID 2
5	n/c	DDC Return	DDC Return
6	Analog RED Return	Analog RED Return	Analog RED Return
7	Analog GREEN Return	Analog GREEN Return	Analog GREEN Return
8	Analog BLUE Return	Analog BLUE Return	Analog BLUE Return
9	n/c	V_{CC} supply (optional)	V_{CC} supply (optional)
10	Digital Ground	Digital Ground	Digital Ground
11	Monitor ID 0	Monitor ID 0	Monitor ID 0
12	Monitor ID 1	Data from Display	Data: SDA
13	HSYNC	HSYNC	HSYNC
14	VSYNC	VSYNC (VCLK)	VSYNC
15	n/c	n/c	Clock: SCL

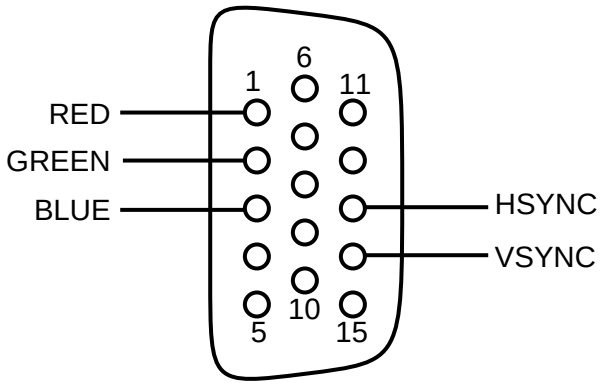
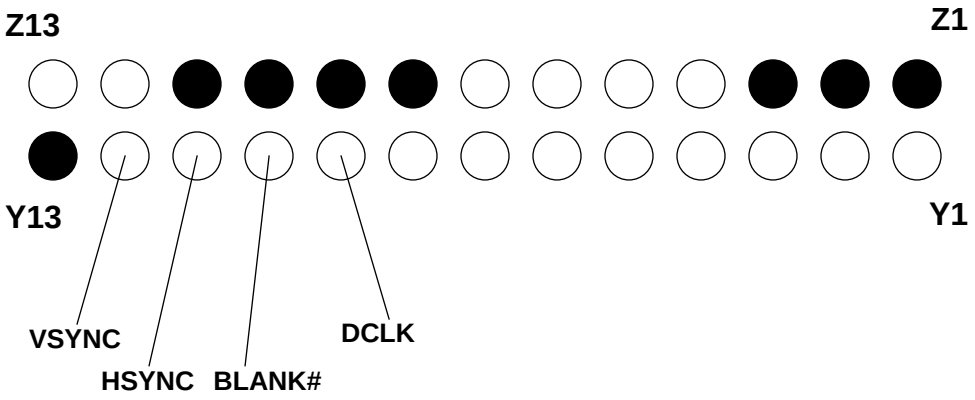


Table A1-2. VESA® Pass-Through Connector

Number	Z	Y
1	Ground	P[0]
2	Ground	P[1]
3	Ground	P[2]
4	EVIDEO#	P[3]
5	ESYNC#	P[4]
6	EDCLK#	P[5]
7	DDCCLK[a]	P[6]
8	Ground	P[7]
9	Ground	DCLK
10	Ground	BLANK#
11	Ground	HSYNC
12	MCLK[a]	VSYNC
13	DDCDAT[a]	Ground

[a] These connections are assigned by Cirrus Logic for compatibility with VMI.



Pins shown as black are ground pins.

Figure A1-1. View From Component Side

Table A1-3. VESA® Advanced Feature Connector

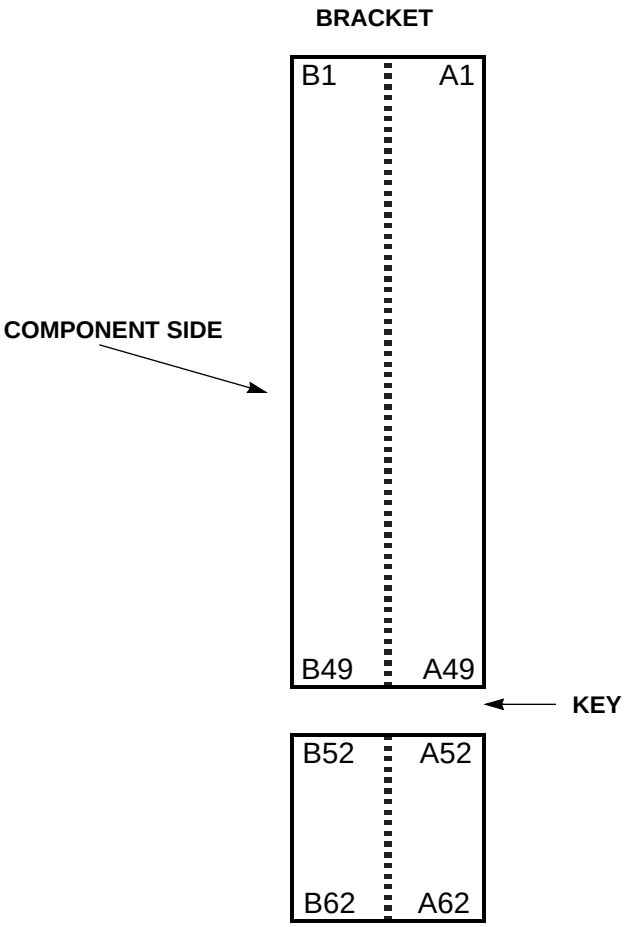
Pin	Name	Pin	Name
1	RSRV0	41	GND
2	RSRV1	42	GND
3	GENCLK	43	GND
4	OFFSET0	44	GND
5	OFFSET1	45	GND
6	FSTAT	46	GND
7	VRDY	47	GND
8	GRDY	48	GND
9	BLANK	49	GND
10	VSYNC	50	GND
11	HSYNC	51	GND
12	EGEN#	52	GND
13	VCLK	53	GND
14	RSRV2	54	GND
15	DCLK	55	GND
16	EVIDEO#	56	GND
17	P0	57	P1
18	GND	58	P2
19	P3	59	GND
20	P4	60	P5
21	GND	61	P6
22	P7	62	GND
23	P8	63	P9
24	GND	64	P10
25	P11	65	GND
26	P12	66	P13
27	GND	67	P14
28	P15	68	GND
29	P16	69	P17
30	GND	70	P18

Table A1-3. VESA® Advanced Feature Connector *(cont.)*

Pin	Name	Pin	Name
31	P19	71	GND
32	P20	72	P21
33	GND	73	P22
34	P23	74	GND
35	P24	75	P25
36	GND	76	P26
37	P27	77	GND
38	P28	78	P29
39	GND	79	P30
40	P31	80	GND

Table A1-4. PCI Bus

Pin	Side B	Side A	Pin	Side B	Side A
1	–12V (not used)	TRKST# (not used)	32	AD[17]	AD[16]
2	TCLK (not used)	+12 V (not used)	33	C/BE[2]#	+3.3 V (not used)
3	Ground	TMS (not used)	34	Ground	FRAME#
4	TDO	TDI (not used)	35	IRDY#	Ground
5	+5 V	+5 V	36	+3.3 V (not used)	TRDY#
6	+5 V	INTA#	37	DEVSEL#	Ground
7	INTB# (not used)	INTC# (not used)	38	Ground	STOP#
8	INTD# (not used)	+5 V	39	LOCK#	+3.3 V (not used)
9	PRSNT1#	Reserved	40	PERR# (not used)	SDONE (not used)
10	Reserved	+5 V (I/O)	41	+3.3 V (not used)	SBO# (not used)
11	PRSNT2#	Reserved	42	SERR# (not used)	Ground
12	Ground	Ground	43	+3.3 V (not used)	PAR
13	Ground	Ground	44	C/BE[1]#	AD[15]
14	Reserved	Reserved	45	AD[14]	+3.3 V (not used)
15	Ground	RST#	46	Ground	AD[13]
16	CLK	+5 V (I/O)	47	AD[12]	AD[11]
17	Ground	GNT# (not used)	48	AD[10]	Ground
18	REQ# (not used)	Ground	49	Ground	AD[09]
19	+5 V (I/O)	Reserved	50	Connector key	
20	AD[31]	AD[30]	51		
21	AD[29]	+3.3 V (not used)	52	AD[08]	C/BE[0]#
22	Ground	AD[28]	53	AD[07]	+3.3V (not used)
23	AD[27]	AD[26]	54	+3.3 V (not used)	AD[06]
24	AD[25]	Ground	55	AD[05]	AD[04]
25	+3.3 V (not used)	AD[24]	56	AD[03]	Ground
26	C/BE[3]#	IDSEL	57	Ground	AD[02]
27	AD[23]	+3.3 V (not used)	58	AD[01]	AD[00]
26	Ground	AD[22]	59	+5 V (I/O)	+5 V (I/O)
29	AD[21]	AD[20]	60	ACK64# (not used)	REQ64# (not used)
30	AD[19]	Ground	61	+5 V	+5 V
31	+3.3 V (not used)	AD[18]	62	+5 V	+5 V



PCI

Appendix A2

Revision B Notes

REVISION B NOTES

1. INTRODUCTION

Revision B of the CL-GD5446 incorporates a number of changes for compliance with PC97. These changes are summarized below. The detailed descriptions are in the appropriate places in this technical reference manual.

2. DESIGN CHANGES

2.1 PCI10 Claims 32 Mbytes

PCI10 claims 32 Mbytes rather than 16 Mbytes, as is the case with Revision A. This allows a set of byte-swapping apertures for direct access to the frame buffer, and a set of byte-swapping apertures for system-to-screen BitBLTs with no interference or restrictions.

The meaning of GR31 changes to accommodate this. In Revision A devices, GR31[6] enables system-to-screen BitBLT data transfers at BC000h. In Revision B devices, GR31[6] enables system-to-screen BitBLT data transfers at the second set of apertures beginning at PCI10 plus 16 Mbytes.

2.2 PCI14 Supports VGA and BitBLT Registers

PCI14 is always enabled for 4096 bytes of memory space regardless of the configuration. The first 100 hex-bytes are used for VGA register; the second 100 hex bytes are used for BitBLT registers (MMIO). CF3 is not used on Revision B. The addresses for these registers are given in Table 9-3 and Table 9-4 in Chapter 9, “Programming Notes”.

2.3 PCI18 Supports GPIO

GPIO has moved to PCI18. GPIO can be in either memory space or I/O (or can be disabled), according to CF8 and CF4.

2.4 PCI2C Supports Subsystem and Subsystem Vendor ID

PCI2C is loaded from the EPROM and the Pixel bus at RST# time as shown in Table A2-1. If SR17[3] is ‘1’, PCI2C is write-enabled.

Table A2-1. PCI Source

PCI	Source	Note
PCI2C	ROM 7FFC	
PCI2D	ROM 7FFD	
PCI2E	ROM 7FFE	
PCI2E	Pixel Bus	If GPIO enabled

2.5 General-Purpose I/O Configuration

Since VGA relocation is from GPIO, the configuration is slightly different.

CF8 (MD56) GPIO Configuration (Revision B)

This pin is used in conjunction with MD52 (CF4) to configure GPIO. The VGA registers are always accessible through PCI14 on the Revision B device. See Appendix B11, "General-Purpose I/O", for information regarding GPIO. Table A2-2 summarizes the configurations.

Table A2-2. Relocated I/O, GPI/O Configuration: Revision B

CF8 MD56	CF4 MD52	GPIO	PCI18
0	0	Reserved	Do not configure
0	1	32 bytes I/O	31:5 specify base, bit 0 = 1
1	0	32 bytes memory	31:12 specify base, bit 0 = 0 (4096 bytes claimed)
1	1	Disabled	Read-only, '0'

Appendix B1

Layout Guidelines

LAYOUT GUIDELINES

1. INTRODUCTION

The CL-GD5446 is a highly integrated, mixed-signal circuit with high operating frequencies. These devices are designed into display subsystems with very high-bandwidth buses. Boards based on these controllers will provide a reliable, compact circuit if designed with care.

This appendix distills into a single document the experiences of Cirrus Logic in completing reference designs and problem solving.

2. PARTS PLACEMENT AND ADAPTER CARDS

The first consideration is parts placement. This section covers the placement of the Cirrus Logic device. Subsequent sections discuss how to place passive devices around the main device.

2.1 PCI Bus Adapter Card

The requirements of the PCI specification leave little latitude in the placement of the CL-GD5446. The reference design places the device near the center of the board with the DRAM array in the upper right portion. The PCI board is much smaller that the VESA VL-Bus board, allowing the device to be placed close to the VGA connector. This is shown in Figure B1-1.

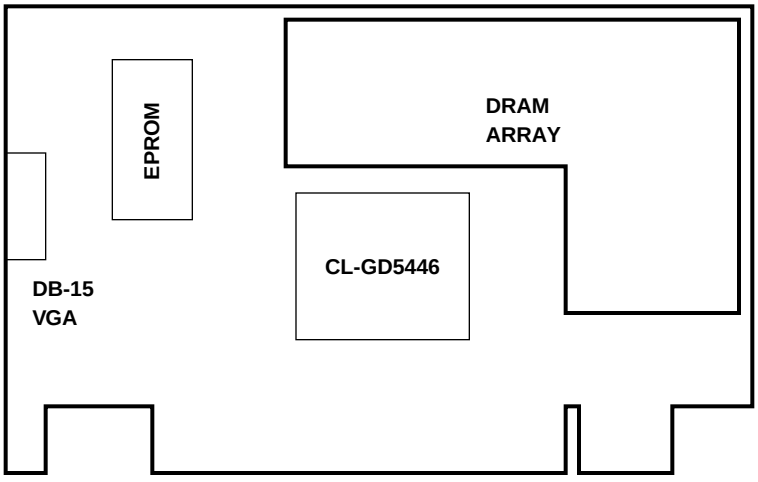


Figure B1-1. PCI Bus Adapter Card Layout

Questions regarding the PCI specification or membership in the PCI Special Interest Group can be forwarded to:

PCI Special Interest Group
M/S HF3-15A
5200 N.E. Elam Young Parkway
Hillsboro, OR 97124-6497
(503) 696-2000

2.2 Motherboard

Parts placement is as important in a motherboard design as in an adapter card. The Cirrus Logic controller must be close to the CPU and core logic, its DRAM array, and the VGA DB15 connector. Also, it must be positioned away from components on the motherboard that could induce noise, such as the main memory, keyboard controller and other peripherals, and the adapter slots.

3. POWER

Cirrus Logic recommends the use of multilayer boards for its components, especially in high-performance systems. As frequencies get higher, it becomes less likely that one can obtain acceptable results with a two-layer board. One plane must be dedicated exclusively to the distribution of power and one plane must be dedicated exclusively to ground.

There must be cuts in the power plane to completely isolate the three power rails to the CL-GD5446 from the VCC on the board (and from each other). Figure B1-2 on page B1-4 shows how the cuts are made on a typical board.

As can be seen in Figure B1-2 — and in the schematics for the Cirrus Logic reference designs — a 1⁄2-Ω resistor is placed in series between the board VCC and the digital VDD pins of the Cirrus Logic chip (the 1⁄2-Ω resistor is actually mechanized as two 1-Ω resistors in parallel to each other). This resistor serves as part of an RC filter to isolate the Cirrus Logic device from noise on the VCC rail, and to provide additional latch-up protection.

Two areas of the power plane must be further isolated. One of these is for VCLKVDD (VCLK synthesizer) and one is for MCLKVDD (MCLK synthesizer). As shown in the reference designs, these areas are individually further isolated with 33-Ω series resistors that serve as RC-filter components.

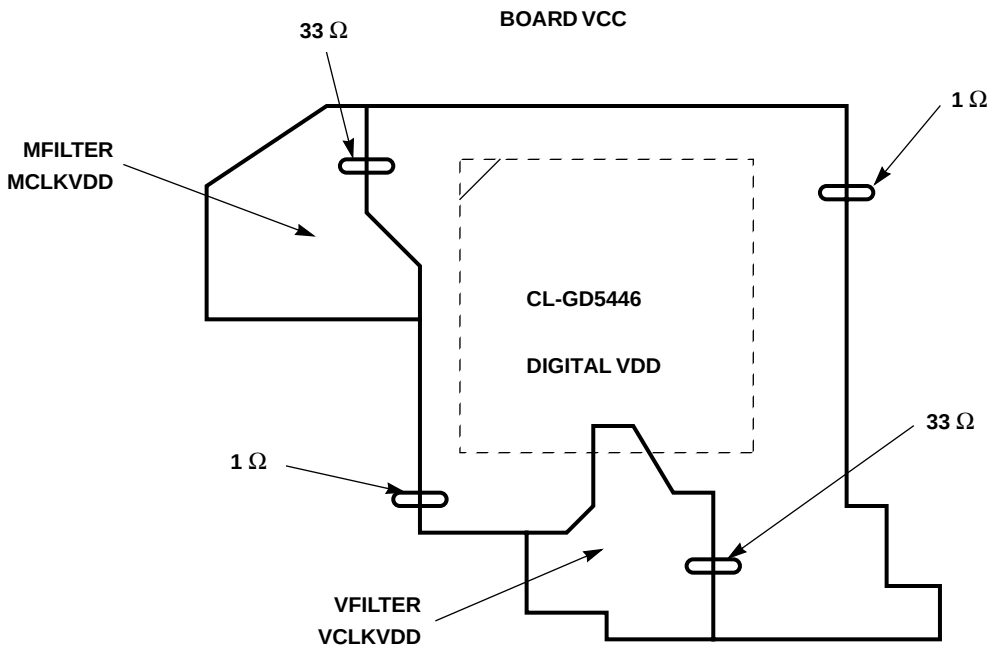


Figure B1-2. Power Plane Layout

4. GROUND

One plane on the board must be dedicated to ground. The ground must have cuts that suppress currents between the various areas (but that do not provide complete isolation). These cuts are shown in Figure B1-3 for a typical reference design.

There is a certain amount of art involved in the exact positioning and size of the cuts in the ground plane and the power plane. Some experimentation may be required to obtain satisfactory results.

The power plane and ground plane cuts must follow each other. It is critical that an isolated ground or power plane not overlay a noisy digital power or ground plane. If such an overlay were allowed, the result would be a capacitor composed of the overlay conductors separated by the relatively thin dielectric between the two pieces of epoxy that make up a four-layer board. Noisy buses (such as data or address) must not be allowed to cross any isolated area.

The ground cuts must not interfere in any way with the return currents between the controller and the DRAM array. Any ground differential between the controller and the DRAM array will directly subtract from the TTL margins.

Cirrus Logic can provide reference designs of adapter cards for various adapter cards which yield satisfactory results and pass FCC Class B emission tests.

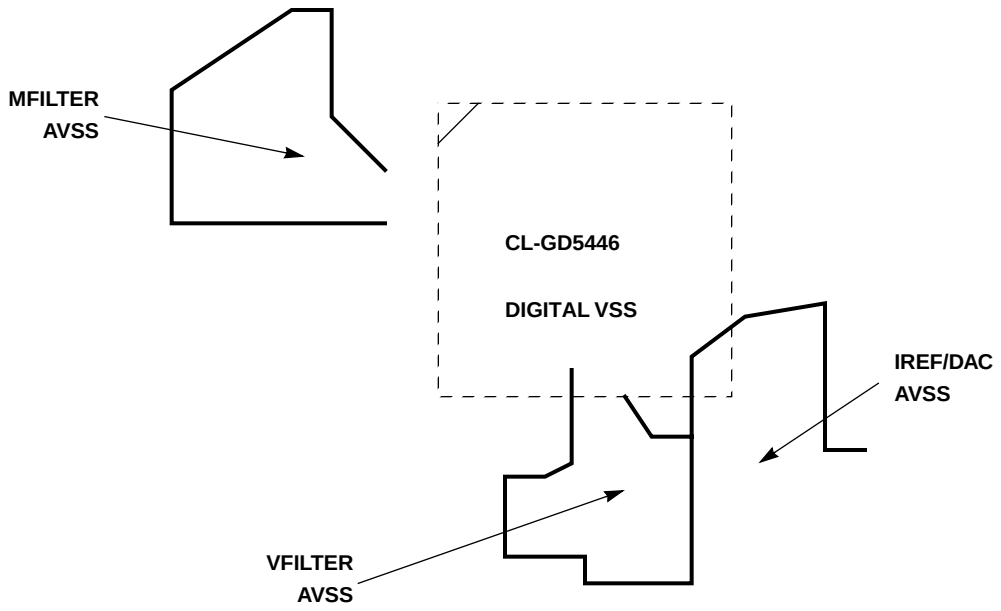


Figure B1-3. Ground Plane Layout

Designers with prior experience using discrete RAMDACs and clock sources may have found that such care with power distribution and isolation was not necessary, especially at relatively low frequencies. The integrated solution available from Cirrus Logic, operating at high frequencies, necessitates these precautions.

5. DECOUPLING CAPACITORS

The CL-GD5446 operates at high frequencies (up to 135 MHz). Adequate power decoupling is absolutely crucial to a successful design. Each power pin on the device must have a 0.1-µF capacitor returned to the local ground. These capacitors must be placed as close to the respective power pins as possible. These capacitors must have excellent high-frequency characteristics; Cirrus Logic has found the surface-mount ceramic chip capacitors perform adequately.

The high-frequency capacitors for MCLKVDD and VCLKVDD must be on the power pin side of the respective 33-Ω resistors, must be as close as possible to the power pin, and must be returned to the appropriate local ground.

The board design must include adequate bulk bypassing. Tantalum capacitors will serve this function. The high-frequency characteristics of the bulk bypass capacitors is not as critical as that of the high-frequency capacitors.

6. SYNTHESIZER

The CL-GD5446 has on-chip synthesizer filters. External synthesizer filters are not required. These chips also include an on-chip oscillator and so can use a 14.3-MHz crystal as the reference. This crystal is connected between OSC and XTAL. The connections to the crystal should be short and direct. Each side of the crystal requires a 27-pF capacitor to ground.

7. IREF CIRCUITRY

The CL-GD5446 has an on-chip current reference generator. The current level is set by a resistor from RSET to DACVSS. The board layout should be designed to support two parallel capacitors from IREF to VCC (one tantalum and one ceramic). Upon receiving silicon, Cirrus Logic will provide board population guidelines. If desired, an externally generated reference current can be injected on IREF.

8. RGB LINES

The rise and fall times on the RGB traces are going to be 2–4 ns, causing the traces to behave as transmission lines. This means that the characteristic impedance must be controlled and must be close to the nominal monitor termination value of 75 Ω.

There must be π-LC filters on each of the RGB lines, as shown in the reference designs. The recommended component values are 10 pF for the capacitors. The inductor is a ferrite bead, with 10–20 Ω impedance at 100 MHz.

There is a trade-off involved in the selection of these component values. Obtaining crisp pixels on screen requires that the rise and fall times be as short as possible. However, to obtain acceptable emissions testing results, relatively slow rise and fall times are preferred. As the pixel rates get higher, there will be less margin between these two conflicting requirements. The component values recommended above represent the Cirrus logic recommendation as of this writing. Filter components must be placed as close as possible to the VGA DB15 connector.

A 75-Ω resistor to DACVSS is specified for each of the RGB lines. These resistors must be placed as close to the CL-GD5446 as possible.

9. DRAM ARRAY

The DRAMs in display memory typically operate as fast as, or faster than, those in the system memory. The layout of this array must be given as much consideration as that of the system memory. The following general rules apply.

The devices should be placed close to the CL-GD5446. In addition, they must be organized so that each individual device is close to the respective MD pins on the controller. Pin organization of the controller was carefully optimized to allow this.

The control lines are fast, heavily loaded lines and must be treated as such. Relatively wide traces must be used (8–10 mm is typical) and they must be adequately spaced. Placing the traces on 25 mm centers would be ideal. Avoid long parallel runs.

Appendix B2

PCI Bus Reference Design

PCI BUS REFERENCE DESIGN

1. INTRODUCTION

This appendix covers the CL-GD5446 reference design. This is an adapter board for the PCI bus. The board can be populated with 1 or 2 Mbytes of display memory. The board will also support a CL-GD5440. This appendix illustrates the update path.

The schematic was captured with OrCAD® SDT. This schematic and the associated Gerber files are available to Cirrus Logic customers through the Company BBS (see Appendix D1, "Cirrus Logic Bulletin Board Service, FTP, and WWW").

2. PCI BUS INTERFACE

2.1 Bus Connections

The CL-GD5446 is designed for a glueless interface to the PCI bus. Pins on the CL-GD5446 are connected directly to similarly named pins on the PCI bus. This is summarized in Table B2-1.

Table B2-1. PCI Bus Connections

Pin Names	Function	Note
AD[31:0]	Address/Data bus	
PAR	Parity	
STOP#	Control	
C/BE[3:0]#	Control	
FRAME#	Control	
IRDY#	Control	
TRDY#	Control	
IDSEL	Control	
CLK	Clock	
RST#	Control	
DEVSEL#	Control	
INTR#	Interrupt Request	Connected with jumper
PRSNT#2	System Control	Grounded
TDI/TDO	System Control	Tied together

The CL-GD5446 is placed within an inch of the PCI connector and approximately centered on the connector. The pin assignments on the CL-GD5446 are carefully optimized to allow short and direct connections between the bus pins and the CL-GD5446 pins. The CLK trace is laid out first and made wider than others.

2.2 INTR# Pin

The INTR# pin is connected from CL-GD5446 with a jumper.

For the CL-GD5446 and the CL-GD5440, PCI3C[8] can be configured in the chip. If a pull-down resistor is installed on MD62, PCI3C[8] is '1' and the interrupt claimed. The board is designed with a pull-down resistor on MD62 that is automatically grounded if a VMI board is plugged in, grounding pin Z19 of the VMI connector.

2.3 VGA BIOS

The CL-GD5446 is designed to comply with the PCI requirement that a single load appear on each signal. This requirement precludes connecting the BIOS EPROM directly to the bus. Rather, the EPROM is connected only to the CL-GD5446. The EPROM address inputs are driven with dedicated pins on the CL-GD5446 (BIOSA[14:0]). The EPROM data pins are connected to MD[7:0]; these pins are multiplexed. The EPROM enables are both driven with the CL-GD5446 EROM#.

A 27C256 EPROM contains the 32K VGA BIOS. The address pins are connected directly pin-to-pin; no address bit swapping is used.

3. DISPLAY MEMORY INTERFACE

3.1 Memory Configurations

The display memory is made up of two or four pieces of 256K $\times$ 16 dual-CAS# DRAMs or four pieces of 128K $\times$ 16 dual-CAS# DRAMs. Table B2-2 indicates the memory configurations available with this design.

Table B2-2. Display Memory Configurations

Capacity	Devices	Note
1 Mbyte	One, two	256K devices
2 Mbyte	One, two, three, four	256K devices
1 Mbyte	One, two, three, four	128K devices (CL-GD5446 only)

3.2 Damping Resistors

The MA lines and all memory control lines from the CL-GD5446 into the display memory array have series resistors to damp reflections from the array and to control edge rates. These resistors are shown as 10 Ω. The value can be adjusted based on the number of devices actually populated. Ideally the lines should be nearly critically damped.

4. MONITOR INTERFACE

4.1 RGB Lines

The RGB lines are terminated in 75 Ω to DACVSS. This provides half of the nominal 37.5-Ω DC load; the other half is in the monitor.

The π-shaped filters on each RGB line control edge rates and reduce RFI (radio frequency interference) to an acceptable level. The component values in these filters represent an engineering trade-off. For crisp pixels, especially at higher frequencies, the cutoff frequency should be as high as possible. On the other hand, for reduced emissions the cutoff frequency should be fairly low. The values shown on the schematics represent the best engineering judgement at the time the schematics were captured.

The resistors are located as close as possible to the CL-GD5446. The filters are located very close to the DB-15 connector. The traces between the CL-GD5446 and the π filters are direct with an absolute minimum of vias and no sharp corners. These traces must be designed with a characteristic impedance as close as possible to 75 Ω. The edge rates, especially before the filter, are fast enough that a trace as short as a few inches begins to behave as a transmission line.

4.2 Sync Lines

HSYNC and VSYNC are isolated with RC filters of 33 Ω and 220 pF. The filter outputs connect directly to the DB-15 and VESA connectors.

4.3 Monitor ID

The BIOS requires information regarding the capability of the connected monitor so that it can program the appropriate refresh rates for the various display modes. In some cases, the monitor capability may not support higher resolution modes (that is, these modes cannot be programmed at all).

4.3.1 Legacy Monitors

End users with monitors that are not DDC-compliant should use CLMODE or an equivalent utility to specify to the BIOS the monitor type.

4.3.2 DDC2B Support (I^2C)

DDC2B is a bidirectional data channel based on the I^2C bus. The data clock is on MID[3] (DB15 pin 15) and the bidirectional data is on MID[1] (DB15 pin 12). Table B2-3 shows the connections for DDC2B and I^2C. The connections from both chip pins are made by zero-Ω resistors. The connections to the DB15 connectors are made with 1-kΩ resistors.

Table B2-3. DDC2B, I^2C Connections

CL-GD5446 Pin Name	Alternate Pin Name	MID (DB15 Pin)	Feature Connector	Function	Read Port	Write Port
DDCDAT (pin 106)	EEDI	MID1 (pin 12)	Z13 (I2CDAT)	Data	SR8[7]	SR8[1]
DDCCLK (pin 107)	EECS	MID3 (pin 15)	Z7 (I2CCLK)	Clock	SR8[2]	SR8[0]

5. FEATURE CONNECTOR

The VESA connector pins are tied to the corresponding pins on the CL-GD5446 either directly or through an appropriate resistor. The Video Port uses the same connector. Table B2-4 shows the pins on the VESA connector.

Table B2-4. VESA® Connector/ Video Port

Pin	Function	Video Port	Pin	Function	Video Port
Z1	GND	GND	Y1	P0	PIXD[0]
Z2	GND	GND	Y2	P1	PIXD[1]
Z3	GND	GND	Y3	P2	PIXD[2]
Z4	EVIDEO#	VACT	Y4	P3	PIXD[3]
Z5	ESYNC#	EN2	Y5	P4	PIXD[4]
Z6	EDCLK#	VREF	Y6	P5	PIXD[5]
Z7	n/c[a]	I2CCLK	Y7	P6	PIXD[6]
Z8	GND	GND	Y8	P7	PIXD[7]
Z9	GND	GND	Y9	DCLK	PIXCLK
Z10	GND	GND	Y10	BLANK#	HREF
Z11	GND	GND	Y11	HYSNC	n/c
Z12	n/c	EN1	Y12	VSYNC	n/c
Z13	n/c	I2CDAT	Y13	GND	GND

[a] n/c indicates ‘no connect’.

6. VMI HOST PORT

VMI is a standards proposal for connecting a video module, such as an MPEG decoder module, to a 'video ready' GUI device. For further information regarding VMI, refer to the VMI 1.4 document available upon request from Cirrus Logic. Table B2-5 and Table B2-6 show the connections to the VMI Host Port Connector.

Table B2-5. VMI Host Port Connections

VMI Port Pin Number	VMI Port Pin Name	Node Name	CL-GD5446 Pin Name	CL-GD5446 Pin Number
Z1	+12V	POWER	–	–
Z2	HD[1]	BIOSA1	GPD1	25
Z3	GND	–	–	–
Z4	HD[3]	BIOSA3	GPD3	27
Z5	+5V	–	–	–
Z6	HD[6]	BIOSA6	GPD6	30
Z7	OSC	BOSC	OSC	17
Z8	HA[1]	BIOSA15	GPA1	42
Z9	HA[3]	OVRW#	OVRW#	109
Z10	GND	–	–	–
Z11	CS#	GPCS#	Reserved	108
Z12	RD#	BIOSA8	GPIORD#	32
Z13	+3.3V	+3.3VOLTS	–	–
Z14	SCLK	–	–	–
Z15	LRCK	–	–	–
Z16	+5V	–	–	–
Z17	AVIDGND	AVIDGND	–	–
Z18	AVIDC	AVIDC	–	–
Z19	GPDRY	GPDRY	GPRDY	66
Z20	AUDGND	AUDGND	–	–
Y1	HD[0]	BIOSA0	GPD0	24
Y2	GND	–	–	–
Y3	HD[2]	BIOSA2	GPD2	26
Y4	HD[4]	BIOSA4	GPD4	28

Table B2-5. VMI Host Port Connections *(cont.)*

VMI Port Pin Number	VMI Port Pin Name	Node Name	CL-GD5446 Pin Name	CL-GD5446 Pin Number
Y5	HD[5]	BIOSA5	GPD5	29
Y6	HD[7]	BIOSA7	GPD7	31
Y7	HA[0]	BIOSA14	GPA0	41
Y8	HA[2]	TWR#	TWR#	105
Y9	+5V	–	–	–
Y10	RESET#	RST#	RST#	51
Y11	GND	–	–	–
Y12	WR#	BIOSA9	GPIOWR#	34
Y13	READY#	GPRDY	GPRDY	66
Y14	INTREQ#	INTR#	INTR#	19
Y15	PCMDATA	PCMDATA	–	–
Y16	+3.3V	+3.3VOLTS	–	–
Y17	AVIDY	–	–	–
Y18	Key	–	–	–
Y19	AUDIOL	–	–	–
Y20	AUDIOR	–	–	–

To interface to MPEG modules that require a 16-bit interface, either for the host interface or for the video interface, the most-significant PIXD/GPD bits have been brought out to a connector.

Table B2-6. 16-bit Host/Video Interface

Connector Pin Number	Connector Pin Name	Node Name	GPIO Pin Name	V-Port™ Pin Name	CL-GD5446 Pin Number
Z1	HD[8]	GPD8	GPD8	PIXD8	45
Z2	HD[10]	BIOSA13	GPD10	PIXD10	40
Z3	HD[12]	GPD12	GPD12	PIXD12	38
Z4	HD[14]	BIOSA11	GPD14	PIXD14	36
Y1	HD[9]	GPD9	GPD9	PIXD9	44
Y2	HD[11]	GPD11	GPD11	PIXD11	39
Y3	HD[13]	BIOSA12	GPD13	PIXD13	37
Y4	HD[15]	BIOSA10	GPD15	PIXD15	35

7. POWER DISTRIBUTION AND CONDITIONING

7.1 Introduction

By far the most common reason for unsatisfactory performance of a display subsystem is failure on the part of the board designer to properly manage power distribution and conditioning. Dedicated power and ground planes are very strongly recommended for boards based on CL-GD5446 devices.

7.2 Dedicated Ground Plane

A dedicated ground plane minimizes differential ground offsets and more nearly approximates the ideal notion of 'ground'. Additionally, a ground plane is necessary to predict and control the characteristic impedance of those traces that must be treated as transmission lines.

The ground plane has cuts to partially isolate the critical analog VSS sections from the relatively noisy digital VSS associated with the DRAM array and the bus interface. These cuts can be studied in the Gerber plots. On the schematic diagram, there are four ground nodes. The digital ground (used by the DRAMs) is designated with a standard ground symbol. The three isolated grounds are designated DACVSS, MCLKVSS, and VCLKVSS.

7.3 Dedicated Power Plane

A dedicated power plane allows low-impedance distribution of VCC, minimizing noise and coupling. A dedicated power plane also behaves as an AC ground, making it possible to predict and control the characteristic impedance of traces above it.

The power plane has regions completely isolated from the digital portion of the plane. The power plane beneath the CL-GD5446 is isolated and connected to the main section of the power plane with two parallel zero-Ω resistors. This isolated area is designated VDD on the schematic diagram.

Two areas for the synthesizer power conditioning are further isolated. The power plane can be studied in the Gerber prints.

7.4 Power Bypassing

Bypass capacitors minimize power sags caused by current spikes and to reduce the power distribution impedance. Bulk bypassing is present in the area where power comes onto the board, around the DRAM array, and near the EPROM.

High-frequency bypass capacitors are distributed as needed on the board. Every digital VCC pin on the CL-GD5446 has a bypass capacitor located as close to the pin as possible. Each pin is connected to its capacitor — and the isolated VDD section of the power plane — with a short, thick, direct lead. The ground connection of each capacitor is made with a via directly to the ground plane. Each DRAM has a high-frequency bypass capacitor located very close to pin 20 (the VCC pin). The VCC pin is connected with a short, thick, direct lead. The ground connection of the capacitor is made with a via directly to the ground plane.

7.5 Analog Power Conditioning

Two areas on the power plane are further isolated within the VDD section. One is designated MCLKVDD and the other is designated VCLKVDD. Each is connected to VDD through an RC filter consisting of a 33-Ω resistor and a 10-µF capacitor in parallel with a 0.1-µF capacitor. Each of the capacitors in each filter is returned to its respective cutout on the ground plane.

Power for the DAC (DACVDD) is taken from the VDD section of the power plane through a zero-Ω resistor. It is bypassed with a 10-µF capacitor in parallel with a 0.1-µF capacitor. Each capacitor is returned to the DAC cutout of the ground plane.

8. CONFIGURATION RESISTORS

There are two separate sets of configuration resistors for this board. The resistors listed in Table B2-7 are for bus and memory configurations. They should all be populated, except for MD56, which should never have a pull-down resistor for the CL-GD5446.

Table B2-7. Configuration Resistors

MD Pin	CF	CL-GD5446	CL-GD5440
MD58	CF10	Dual-CAS# DRAMs	Dual-CAS# DRAMs
MD57	CF9	Extended RAS# timing	Extended RAS# timing
MD56	CF8	128-byte GPIO (Do not populate)	50 MHz MCLK (default)
MD[48:47]	CF1, CF0	PCI bus	PCI bus

The resistors listed in Table B2-8 configure the V-Port. They are only used when the board is stuffed with a CL-GD5446. Do not populate the resistors when the board is stuffed with a CL-GD5440. These resistors are not connected directly to ground; rather they are connected to ground when a VMI board is plugged in. If no VMI board will ever be plugged in, these do not have to be populated.

It may be necessary to ground one of these configuration resistors when a VMI board is not installed. An example would be to claim an interrupt on a CL-GD5446. The zero-Ω resistor at the common node of the configuration resistors, and the appropriate configuration resistor(s) can be populated.

Table B2-8. V-Port™ Configuration Resistors

MD Pin	CF	CL-GD5446
MD51	CF3	Enable PCI14
MD52	CF4	16-byte GPIO; Disable VGA relocation (CF8 = 1)
MD54	CF6	Disable feature connector drivers on RST#
MD62	CF14	Claim interrupt

9. DUAL-FREQUENCY SYNTHESIZERS SUPPORT

9.1 Synthesizer Reference

A 14.31818-MHz crystal and two 27-pF capacitors can be populated to provide the synthesizer reference. This will take advantage of the on-chip oscillator. As an alternative, a 14.31818 oscillator can be populated to inject the reference directly into the OSC pin.

9.2 Synthesizer Filters

The CL-GD5446 synthesizers do not require external filters.

10. DAC CURRENT REFERENCE

The CL-GD5446 features an on-chip reference current generator that requires only a resistor to set the full scale current. The resistor value can be calculated with the following equation where *Load* is the DC load in ohms, and *VFullScale* is the desired full-scale voltage. The derivation of Equation B2-1 is given in Appendix B5.

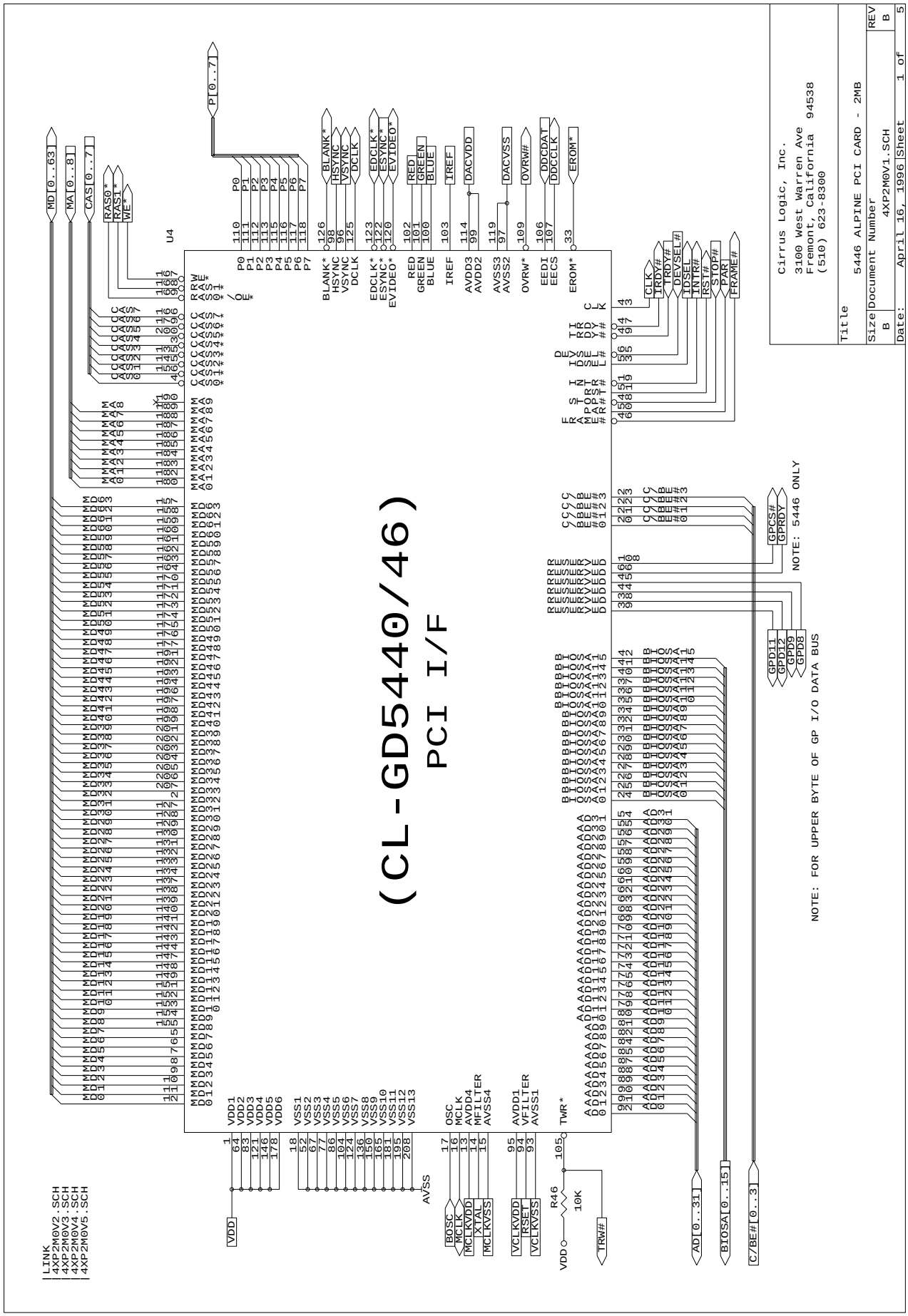
$$RSet = \frac{2.52V \bullet Load}{VFullScale}$$

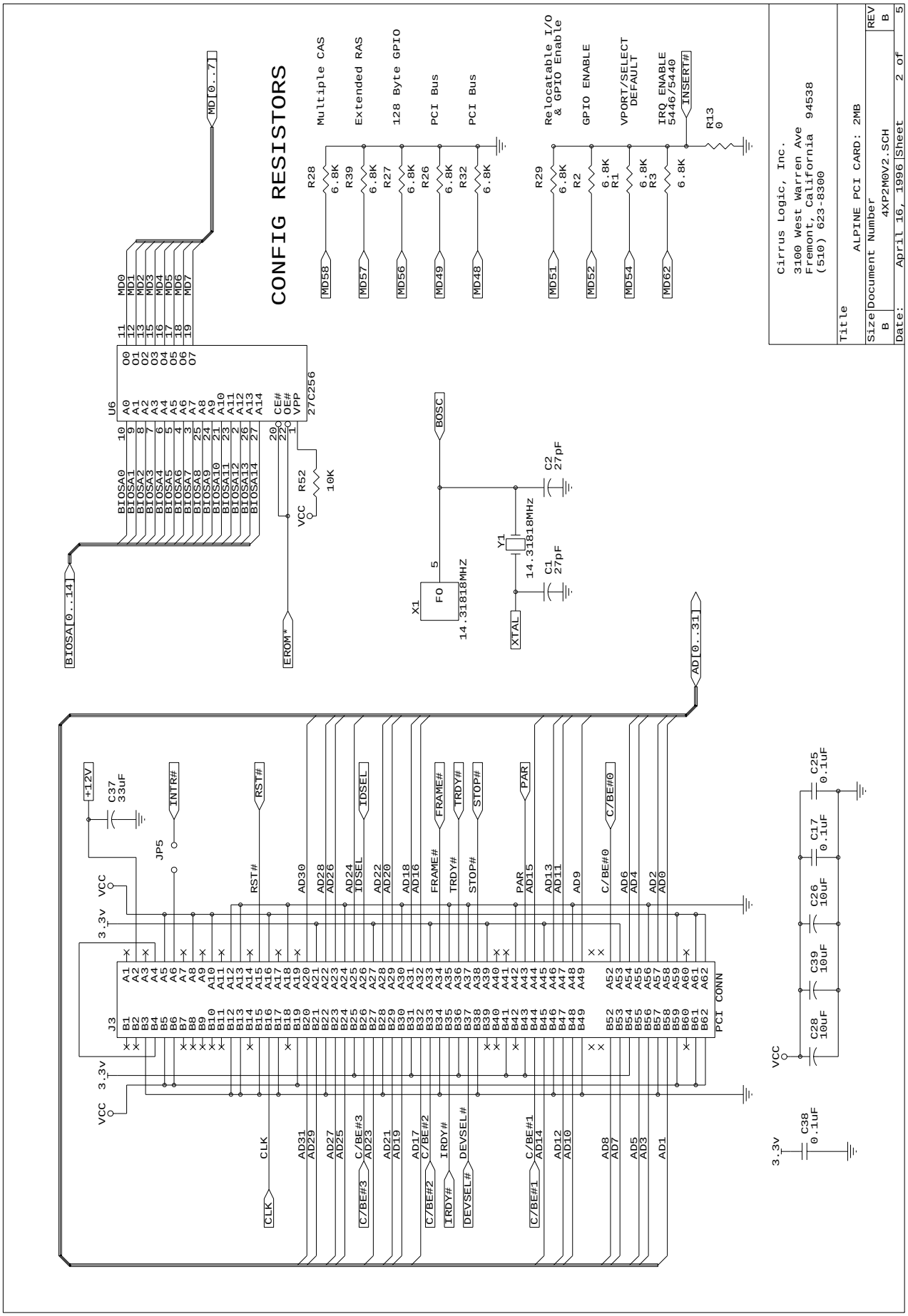
Equation B2-1.

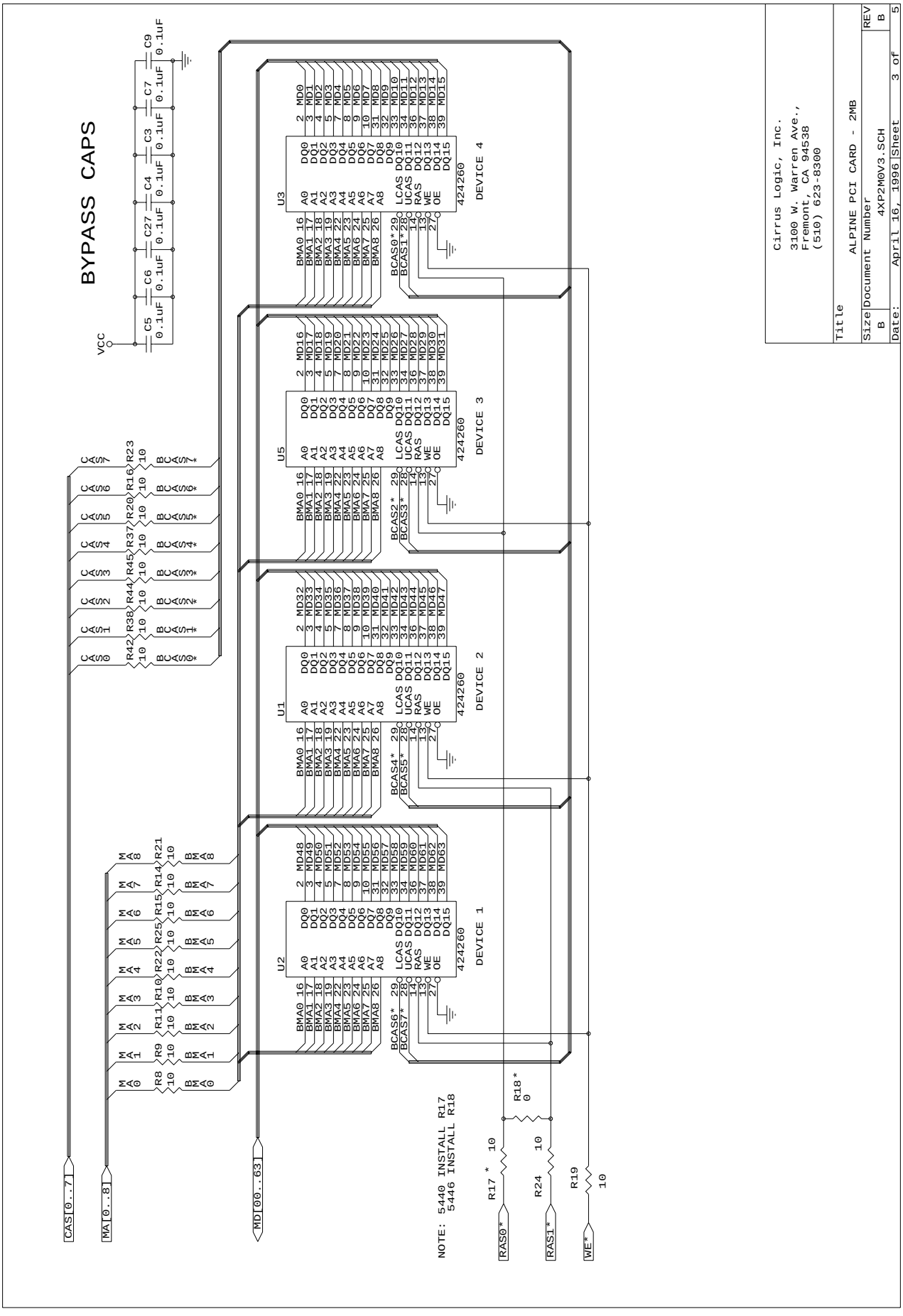
The board is designed with pads for two capacitors between the IREF pin and DACVDD. Which of these is used, if either, is determined when the design is evaluated with real silicon.

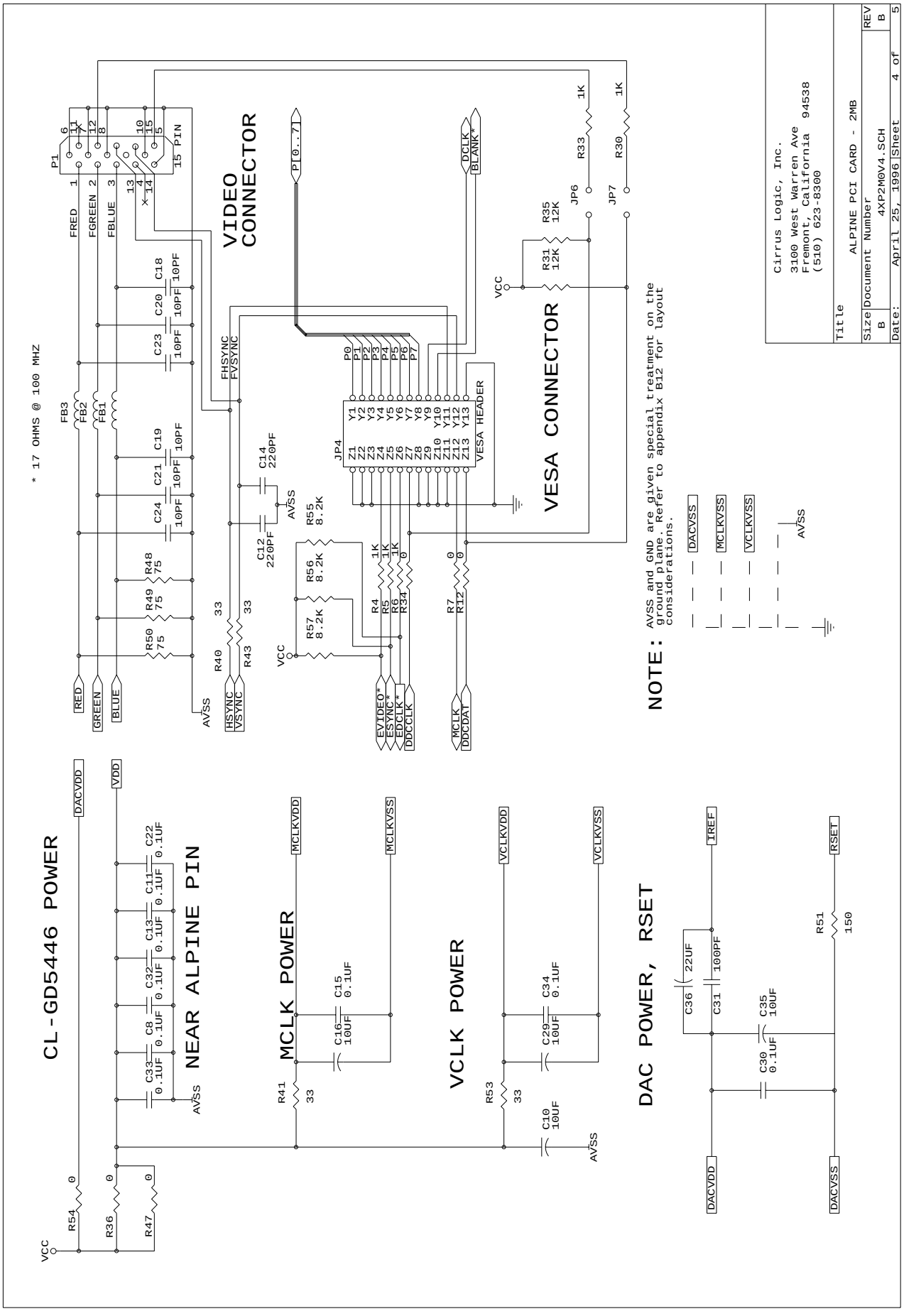
11. PCI BUS SCHEMATICS

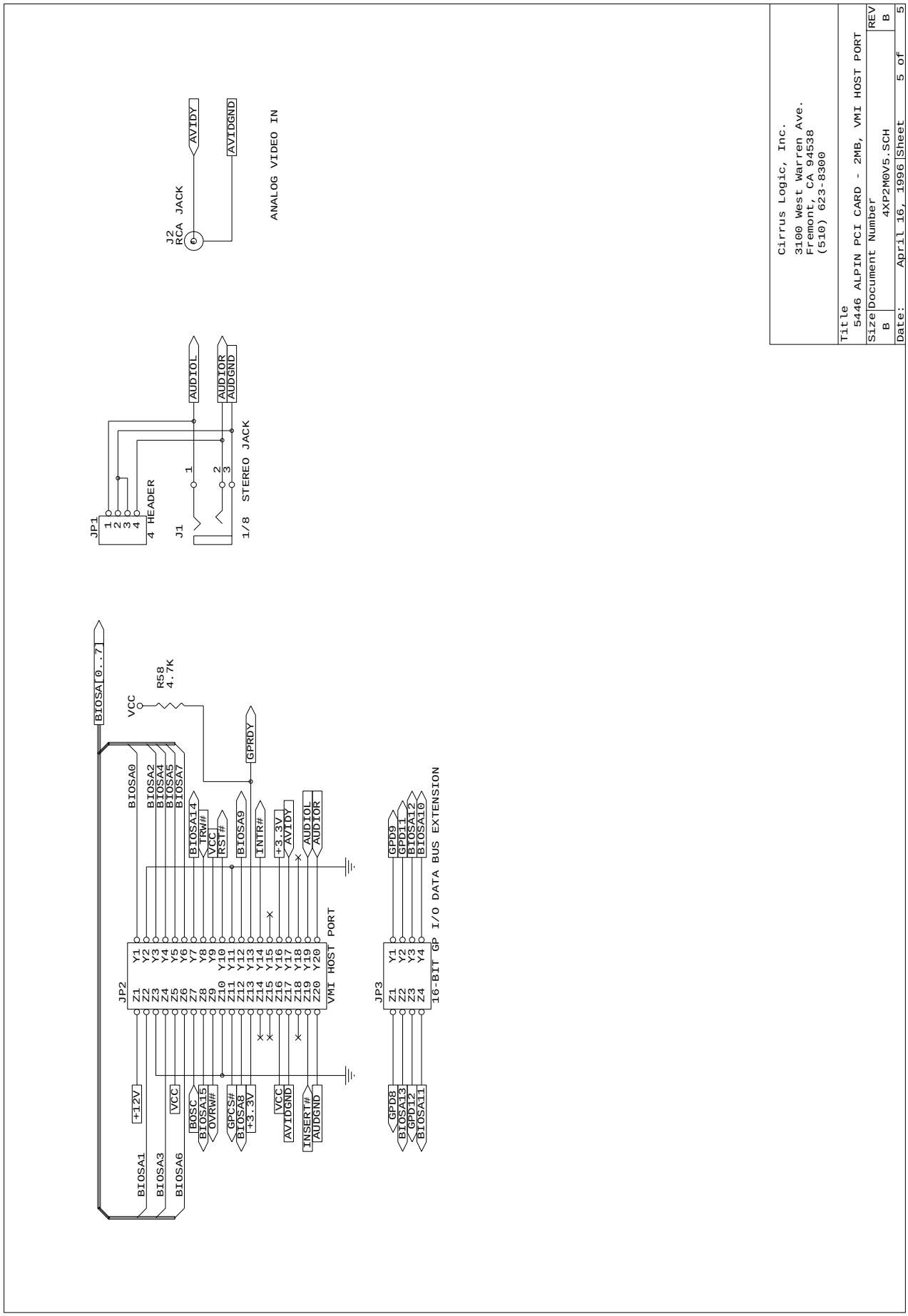
The following pages present the PCI bus schematics.


(CL-GD5440/46)
PCI I/F
U4
LINK
4XP2M0V2.SCH
4XP2M0V3.SCH
4XP2M0V4.SCH
4XP2M0V5.SCH
MD[0..63]
MA[0..8]
CAS[0..7]
RAS0*
RAS1*
WE*
P[0..7]
BLANK*
HSYNC
VSYNC
DCLK
EDCLK*
ESYNC*
EVIDEO*
RED
GREEN
BLUE
IREF
TREF
AVDD3
AVDD2
DACVDD
AVSS3
AVSS2
DACVSS
OVRW*
OVRW#
EEDI
EECS
DDCDAT
DDCCLK
EROM*
CLK
TRDY#
DEVSEL#
TDSEL
INTR#
RST#
PERR#
PAR
FRAME#
GPCS#
GPRDY
NOTE: 5446 ONLY
GPD11
GPD12
GPD9
GPD8
NOTE: FOR UPPER BYTE OF GP I/O DATA BUS
VDD
AVSS
BOSC
MCLK
MCLKVDD
XTAL
MCLKVSS
VCLKVDD
VCLKSS
VCLKVSS
R46
10K
TRW*
TRW#
ADI[0..31]
BIOSA[0..15]
C/BE#[0..3]
Cirrus Logic, Inc.
3100 West Warren Ave
Fremont, California 94538
(510) 623-8300
Title
5446 ALPINE PCI CARD - 2MB
Size
B
Document Number
4XP2M0V1.SCH
REV
B
Date: April 16, 1996
Sheet 1 of 5




BYPASS CAPS
VCC
C5 0.1uF
C6 0.1uF
C27 0.1uF
C4 0.1uF
C3 0.1uF
C7 0.1uF
C9 0.1uF
U1
U2
U3
U5
424260
DEVICE 1
DEVICE 2
DEVICE 3
DEVICE 4
CAS[0..7]
MA[0..8]
MD[00..63]
NOTE: 5440 INSTALL R17
5446 INSTALL R18
R17* 10
R18* 0
R24 10
R19 10
RAS0*
RAS1*
WE*
Cirrus Logic, Inc.
3100 W. Warren Ave.,
Fremont, CA 94538
(510) 623-8300
Title ALPINE PCI CARD - 2MB
Size B
Document Number 4XP2M0V3.SCH
REV B
Date: April 16, 1996 Sheet 3 of 5




Cirrus Logic, Inc.
3100 West Warren Ave.
Fremont, CA 94538
(510) 623-8300
Title
5446 ALPIN PCI CARD - 2MB, VMI HOST PORT
Size B
Document Number 4XP2M0V5.SCH
REV B
Date: April 16, 1996 Sheet 5 of 5
ANALOG VIDEO IN
J2
RCA JACK
AVIDY
AVIDGND
JP1
4 HEADER
J1
1/8 STEREO JACK
AUDIOL
AUDIOR
AUDGND
BIOSA[0..7]
VCC
R58
4.7K
GPRDY
JP2
VMI HOST PORT
JP3
16-BIT GP I/O DATA BUS EXTENSION
+12V
VCC
BOSC
GPCS#
+3.3V
AVIDGND
INSERT#
AUDGND
TRW#
RST#
INTR#
AUDIOL
AUDIOR
GPD8
GPD9
GPD11
GPD12
BIOSA10
BIOSA11
BIOSA12
BIOSA13

Appendix B3

Memory Configurations and Timing

MEMORY CONFIGURATIONS AND TIMING

1. INTRODUCTION

This appendix covers three topics related with the selection of DRAMs for the CL-GD5446: DRAM configuration and connections, DRAM timing requirements, and MCLK requirements for display modes.

2. DRAM CONFIGURATIONS

The CL-GD5446 supports a number of display memory configurations, from 1 to 4 Mbytes, using DRAMs that are organized as:

- 128K × 16
- 128K × 32 (each device replaces two pieces of 128K × 16)
- 256K × 16 — dual-CAS#
- 256K × 4 (four devices replace one 256K × 16)
- 256K × 8 (two devices replace one 256K × 16)
- 256K × 32 (each device replaces two pieces of 256K × 16)

Table B3-1 lists the memory configurations that the CL-GD5446 supports. The CL-GD5440 supports 256K devices in configurations up to 2 Mbytes.

Table B3-1. CL-GD5446 Memory Configurations

SRF[7] Second Bank	SR17[7] 1-Mbyte Banks	SR17[1] Bank Swap	Bus Width	RAS1#	RAS0#	Capacity	Table	Upgrade Path(s)
0	0	0	32	Two 256K × 16	–	1 Mbyte	B3-3	2M (B3-4); 4M (B3-5)
0	0	0	64	Four 256K × 16	–	2 Mbytes	B3-4	4M (B3-5)
0	0	1		–	–	(n/a)	–	–
0	1	0	64	Four 128K × 16	–	1 Mbyte	B3-6	2M (B3-7); 3M (B3-8)
0	1	1	–	–	–	Illegal	–	–
1	0	0	32	–	–	(n/a)	–	–
1	0	0	64	Four 256K × 16	Four 256K × 16	4 Mbytes	B3-5	–
1	0	1	64	Four 128K × 16	Four 256K × 16	3 Mbytes	B3-8	–
1	1	0	64	Four 128K × 16	Four 128K × 16	2 Mbytes	B3-7	–
1	1	1	–	–	–	Illegal		–

3. CL-GD5440/'46 DUAL LAYOUT CONSIDERATIONS

If building a common PC board for both the CL-GD5440 and the CL-GD5446, the RAS# connection to the second Mbyte of frame buffer requires special execution. This is because the CL-GD5440 uses RAS0# for the second Mbyte while the CL-GD5446 uses RAS1# for the second Mbyte. An obvious way to treat this is to source RAS# to the second Mbyte from both RAS0# and RAS1# through zero-Ω (or damping) resistors. At manufacturing time, only the appropriate resistor is actually populated. See Appendix B2 for information about how this is handled on the reference design.

NOTE: The CL-GD5440 does not support 128K devices or configurations above 2 Mbytes.

4. DRAM CONNECTION TABLES

Table B3-3 through Table B3-8 show the detailed memory interface connections for every DRAM configuration supported by CL-GD5446. These tables are based specifically on 16-bit-wide devices with two CAS# inputs.

Two 8-bit-wide or four 4-bit-wide devices can be substituted for each 16-bit-wide device to a maximum of 12 to 16 devices total. The relationship among the MD lines and the CAS# lines is shown in Table B3-2. Beyond eight devices total, the damping resistors on the address and control lines may require adjusting.

Alternatively, one 32-bit-wide device with four CAS# inputs may be substituted for two 16-bit-wide devices.

Table B3-2. MD and CAS# Relationship

MD Lines	Associated CAS#
MD[63:56]	CAS7#
MD[55:48]	CAS6#
MD[47:40]	CAS5#
MD[39:32]	CAS4#
MD[31:24]	CAS3#
MD[23:16]	CAS2#
MD[15:8]	CAS1#
MD[7:0]	CAS0#

Table B3-3. 1-Mbyte Display Memory: Two 256K × 16[a]

Memory Device	One	Two
OE#	GND	GND
WE#	WE#	WE#
RAS#	RAS1#	RAS1#
UCAS#	CAS7#	CAS5#
LCAS#	CAS6#	CAS4#
ADDR	MA[8:0]	MA[8:0]
DATA[16:8]	MD[63:56]	MD[47:40]
DATA[8:1]	MD[55:48]	MD[39:32]

[a] This configuration has a 32-bit frame buffer interface on the CL-GD5446 or CL-GD5440.

Table B3-4. 2-Mbyte Display Memory: Four 256K × 16[a]

Memory Device	One	Two	Three[b]	Four
OE#	GND	GND	GND	GND
WE#	WE#	WE#	WE#	WE#
RAS#	RAS1#	RAS1#	RAS1#	RAS1#
UCAS#	CAS7#	CAS5#	CAS3#	CAS1#
LCAS#	CAS6#	CAS4#	CAS2#	CAS0#
ADDR	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]
DATA[16:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]
DATA[8:1]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]

[a] For compatibility with the CL-GD5440, make provisions to source RAS# to the second Mbyte (devices three and four) from RAS0#.

[b] When the CL-GD5440 is configured with 2 Mbytes, the effective frame buffer bus width remains at 32 bits, even though there are 64 MD lines.

Table B3-5. 4-Mbyte Display Memory: Eight 256K × 16[a]

Device Pin	One	Two	Three	Four	Five	Six	Seven	Eight
OE#	GND	GND	GND	GND	GND	GND	GND	GND
WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#
RAS#	RAS1#	RAS1#	RAS1#	RAS1#	RAS0#	RAS0#	RAS0#	RAS0#
UCAS#	CAS7#	CAS5#	CAS3#	CAS1#	CAS7#	CAS5#	CAS3#	CAS1#
LCAS#	CAS6#	CAS4#	CAS2#	CAS0#	CAS6#	CAS4#	CAS2#	CAS0#
ADDR	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]
D[15:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]
D[7:0]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]

[a] CL-GD5446 only.

Table B3-6. 1-Mbyte Display Memory: Four 128K × 16[a]

Memory Device	One	Two	Three	Four
OE#	GND	GND	GND	GND
WE#	WE#	WE#	WE#	WE#
RAS#	RAS1#	RAS1#	RAS1#	RAS1#
UCAS#	CAS7#	CAS5#	CAS3#	CAS1#
LCAS#	CAS6#	CAS4#	CAS2#	CAS0#
ADDR	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]
DATA[16:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]
DATA[8:1]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]

[a] This configuration has a full 64-bit frame buffer with 1 Mbyte. The CL-GD5440 does not support this configuration.

Table B3-7. 2-Mbyte Display Memory: Eight 128K × 16[a]

Device Pin	One	Two	Three	Four	Five	Six	Seven	Eight
OE#	GND	GND	GND	GND	GND	GND	GND	GND
WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#
RAS#	RAS1#	RAS1#	RAS1#	RAS1#	RAS0#	RAS0#	RAS0#	RAS0#
UCAS#	CAS7#	CAS5#	CAS3#	CAS1#	CAS7#	CAS5#	CAS3#	CAS1#
LCAS#	CAS6#	CAS4#	CAS2#	CAS0#	CAS6#	CAS4#	CAS2#	CAS0#
ADDR	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]
D[15:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]
D[7:0]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]

[a] CL-GD5446 only.

Table B3-8. 3-Mbyte Display Memory: Four 128K × 16, Four 256K × 16[a]

Device Pin	One	Two	Three	Four	Five	Six	Seven	Eight
Device	128K	128K	128K	128K	256K	256K	256K	256K
OE#	GND	GND	GND	GND	GND	GND	GND	GND
WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#	WE#
RAS#	RAS1#	RAS1#	RAS1#	RAS1#	RAS0#	RAS0#	RAS0#	RAS0#
UCAS#	CAS7#	CAS5#	CAS3#	CAS1#	CAS7#	CAS5#	CAS3#	CAS1#
LCAS#	CAS6#	CAS4#	CAS2#	CAS0#	CAS6#	CAS4#	CAS2#	CAS0#
ADDR	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]	MA[8:0]
D[15:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]	MD[63:56]	MD[47:40]	MD[31:24]	MD[15:8]
D[7:0]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]	MD[55:48]	MD[39:32]	MD[23:16]	MD[7:0]

[a] SR17[1] is '1', so devices five through eight (RAS0#) are the first 2 Mbytes. This configuration is not supported by the CL-GD5440.

5. DRAM TIMING REQUIREMENTS

The CL-GD5446 is a state machine that generates the DRAM timing that operates from an internally-generated signal called *MCLK*. As the MCLK frequency is increased, the MCLK period is decreased and the DRAMs must be faster to keep pace. Increasing MCLK frequency increases performance, at the cost of more expensive DRAM devices.

Not all DRAMs of a given speed grade are the same. There are marked differences in the values of several parameters, including t_{RP} and t_{CAP}. This means it is dificult to ensure that a DRAM from *any* vendor of a particular speed grade can be used with an MCLK of *X* MHz. A system designer must carefully compare the proposed DRAM data sheet and the timing provided in the CL-GD5446 documentation to determine the maximum permissible MCLK.

The timing flexibility of the CL-GD5446 must also be taken into consideration. There are two timing options: Extended RAS# timing and EDO timing.

5.1 Extended RAS# Timing

Standard RAS# timing random read or write cycles require six MCLK periods; a Fast-page Cycle requires two MCLK periods. If a pull-down resistor is installed on MD57, the DRAM timing is extended so that a random read or write cycle requires seven MCLK periods. In particular, 1⁄2 additional MCLK is inserted between RAS# and CAS# (increasing the available RAS# access time), and 1⁄2 additional MCLK is inserted in RAS# precharge.

Some DRAMs have Column Address access and CAS# access times faster than their RAS# access times. By using Extended RAS# timing with these devices, the MCLK frequency can be increased without decreasing the RAS# Access and RAS# Precharge times. This may result in better performance, depending on how much the MCLK can be increased. The only way to verify this is to run the critical benchmarks with both configurations.

IMPORTANT: It should be clear that switching to Extended RAS# without increasing MCLK results in *decreased* performance.

5.2 EDO Timing

The CL-GD5446 provides EDO support. When GR18[2] is programmed to ‘1’, the timing changes so that a RAS# cycle is eight MCLKs. The device must have been configured for extended RAS# timing (see register CF9 in Appendix B5, “Configuration Notes”). Table B3-9 summarizes the timing equations for the CL-GD5446.

NOTE: In Table B3-9, Table B3-10, and Table B3-10 the ‘Table’ column refers to the table number in Chapter 11, “Electrical Specifications”; the ‘Symbol’ column refers to the timing parameter as shown in the timing diagrams in Chapter 11.

Table B3-9. EDO Timing Equations (CL-GD5446)

Table	Symbol	Description	Normal	Extended	EDO
11-13	t_5	t_{RCD}	2.5 m[a] – 7.5 ns	3 m	4 m
11-13	t_6	t_{RAS}	3.5 m	4 m – 1 ns	5 m – 1ns
11-13	t_7	t_{RP}	2.5 m –2 ns	3 m – 1.5 ns	3 m – 1.5 ns
11-14	t_3	DRAM access time from RAS#	3.5 m	4 m – 1 ns	5 m
11-14	t_4	DRAM access time from Column Address	2 m	2 m	3 m
11-14	t_5	DRAM access time from CAS#	1 m + 3 ns	1 m + 3 ns	2 m
11-14	t_6	DRAM access time from CAS# precharge	2 m	2 m	3 m

[a] m = MCLK

Extended RAS# timing must be configured to program EDO timing. This means there are three cases: Standard RAS# timing, Extended RAS# timing, and EDO timing. This is reflected in the timing diagrams (Chapter 11) and in Table B3-10 and Table B3-11.

5.3 DRAM Timing Equations Evaluated

Refer to the timing diagrams for the display memory interface in Chapter 11, “Electrical Specifications”. Most of the timing numbers are provided in terms of an MCLK period. For example, ‘t_1 — t_{ASR}: Address Setup to RAS#’ is 1.5m – 9 ns’ (‘m’ is the MCLK period). In Table B3-10 and Table B3-11, each of the equations in the timing diagram is evaluated for a number of MCLK frequencies. The parameters are rounded to integers. The numbers given are minimums.

NOTE: The appearance of any frequency in these tables is not a guarantee that any Cirrus Logic device will operate at that frequency or that DRAMs are available that will operate with the resultant timings.

IMPORTANT: To determine which MCLK can be programmed for a given DRAM, compare the numbers in the appropriate table to the corresponding numbers from the DRAM manufacturer’s data sheet. The numbers from the DRAM data sheet must be equal to or smaller than the numbers in the table.

Table B3-10. DRAM Timing Parameter Equations Evaluated (Not EDO Timing)

Table	Symbol	Description	DRAM Timing					
–	–	SR1F[5:0] (hex)	19	1B	1C	1E	20	22
–	–	MCLK frequency (MHz)	44.7	48.3	50.1	53.7	57.3	60.9
–	–	MCLK Period (ns)	22.3	20.7	20.0	18.6	17.5	16.4
11-10	t_1	t_{ASR}: Address setup to RAS#	25	22	21	19	17	16
11-10	t_2	t_{RAH}: Row Address hold	29	26	25	23	21	20
11-10	t_3	t_{ASC}: Address setup to CAS#	19	18	17	16	15	13
11-10	t_4	t_{CAH}: Column Address hold	22	21	20	19	18	16
11-10	t_5	t_{RCD}: RAS# to CAS# (standard RAS#)	48	44	42	39	36	34
11-10	t_5	t_{RCD}: RAS# to CAS# (extended RAS#)	67	62	60	56	52	49
11-10	t_6	t_{RAS}: RAS# pulse width (Standard RAS#)	78	72	70	65	61	58
11-10	t_6	t_{RAS}: RAS# pulse width (extended RAS#)	88	82	79	74	69	65
11-10	t_7	t_{RP}: RAS# precharge (standard RAS#)	54	50	48	45	42	39
11-10	t_7	t_{RP}: RAS# precharge (extended RAS#)	66	61	58	54	51	48
11-10	t_8	t_{CAS}: CAS# pulse width	25	24	23	22	21	19
11-10	t_9	t_{CP}: CAS# precharge	16	15	14	13	12	10
11-10	t_{10}	t_{RC}: Random cycle (standard RAS#) [a]	134	124	120	112	105	99
11-10	t_{10}	t_{RC}: Random cycle (extended RAS#)	156	145	140	130	122	115
11-10	t_{11}	t_{PC}: Page-mode cycle	45	41	40	37	35	33
11-11	t_1	Read data setup to CAS# high	0	0	0	0	0	0
11-11	t_2	Read data hold from CAS# high	10	10	10	10	10	10
11-11	t_3	RAS# active to data valid (standard RAS#)	88	82	79	74	69	65
11-11	t_3	RAS# active to data valid (extended RAS#)	100	92	89	83	78	73
11-11	t_4	Column Address valid to data valid	45	41	40	37	35	33
11-11	t_5	CAS# active to data valid	25	24	23	22	21	19
11-11	t_6	CAS# precharge to data valid	45	41	40	37	35	33
11-12	t_1	t_{CWL}: WE# setup to CAS#	23	21	21	19	18	17
11-12	t_2	t_{DS}: Write data setup to CAS#	20	19	18	17	16	14
11-12	t_3	t_{DH}: Write data hold from CAS#	23	22	21	20	19	17
11-12	t_4	t_{WCH}: WE# active hold	32	29	28	26	24	23
_		t_T: Transition	2	2	2	2	2	2

[a] t_{RC} and t_{PC} are provided for reference only.

Table B3-11. DRAM Timing Evaluated (EDO Timing)

Table	Symbol	Description	EDO DRAM	
–		SR1F[5:0] (hex)	25	2D
–		MCLK frequency (MHz)	66.2	80.5
–		MCLK period (ns)	15.1	12.4
11-13	t_1	t_{ASR}: Address setup to RAS#	14	10
11-13	t_2	t_{RAH}: Row Address hold	18	14
11-13	t_3	t_{ASC}: Address setup to CAS#	12	9
11-13	t_4	t_{CAH}: Column Address hold	15	12
11-13	t_5	t_{RCD}: RAS# to CAS# (EDO timing)	60	50
11-13	t_6	t_{RAS}: RAS# pulse width (EDO timing)	75	61
11-13	t_7	t_{RP}: RAS# precharge (EDO timing)	47	39
11-13	t_8	t_{CAS}: CAS# pulse width	18	15
11-13	t_9	t_{CP}: CAS# precharge	9	6
11-13	t_{10}	t_{RC}: Random cycle (EDO timing)[a]	121	99
11-13	t_{11}	t_{PC}: Page-mode cycle	30	25
11-13	t_{12}	t_{CAS}: (last CAS# of Page mode read burst)	45	37
11-14	t_1	Read data setup to CAS# high	1	1
11-14	t_2	Read data hold from CAS# high	5	5
11-14	t_3	RAS# active to data valid (EDO timing)	76	62
11-14	t_4	Column Address valid to data valid (EDO timing)	45	37
11-14	t_5	CAS# active to data valid (EDO timing)	30	25
11-14	t_6	CAS# precharge to data valid (EDO timing)	45	37

[a] t_{RC} and t_{PC} are provided for reference only.

6. MCLK vs. DISPLAY MODE REQUIREMENTS

The analysis that follows ignores the requirement for CPU and BitBLT accesses. It assumes that every bit of display memory bandwidth is available for the exclusive purpose of refreshing the screen. Of course, that leaves only blanking time for screen updates; this could result in unacceptable performance.

The above disclaimer aside, the relationship between MCLK and display modes can be simply stated in a single sentence:

Data must be put into the CRT FIFO faster than it is taken out.

If this condition is not met, FIFO underflow occurs and the screen does not refresh properly.

The maximum rate that data (in terms of bytes/sec.) is put into the FIFO is the width of the display memory expressed in bytes multiplied by the MCLK divided by two. This is because a Fast-page cycle fetches one memory width worth of bytes and requires two MCLK cycles.

The rate that data is removed from the FIFO is the effective number of bytes per pixel multiplied by the pixel rate (pixel rate is the rate that pixels are put onto the screen, which may not be the same as the value programmed into the synthesizer). The following inequality in Equation B3-1 expresses this. *W* is the *width* of the display memory *data path* in bytes; *B* is the effective number of bytes per pixel; *PCLK* is the Pixel clock.

$$\frac{W \times MCLK}{2} > B \times PCLK$$

Equation B3-1.

Note that the effective number of bytes per pixel is divided by two when AccuPak data is being expanded into YUV. When interpolation is used for vertical zoom or occlusion in the video window is used, the effective number of bytes per pixel is multiplied by two.

Rearranging terms to isolate MCLK yields:

$$MCLK > \frac{B \times PCLK \times 2}{W}$$

Equation B3-2.

For ‘acceptable’ performance, MCLK should be at least 1.1 times its calculated minimum. MCLK of at least 1.2 times its calculated minimum results in ‘good’ performance.

This information can be expressed in tabular form as in Table B3-12 and Table B3-13. These tables are calculated minimums; the actual values should be adjusted with Equation B3-1 and Equation B3-2.

Table B3-12. Minimum MCLK for Text Modes

Display Memory Width	Extended RAS# Timing?	Standard 8-Pixel	Standard 9-Pixel	Page Mode 8-Pixel	Page Mode 9-Pixel
32	Yes	.60 VCLK	.54 VCLK	.33 VCLK	.29 VCLK
32	No	.53 VCLK	.48 VCLK	.31 VCLK	.28 VCLK
64	Yes	.30 VCLK	.27 VCLK	.16 VCLK	.15 VCLK
32	No	.27 VCLK	.24 VCLK	.16 VCLK	.14 VCLK

Table B3-13. Minimum MCLK for Graphics Modes

Display Memory Width	4 bpp	8 bpp	16 bpp	24 bpp	32 bpp
32	.25 VCLK	.50 VCLK	1.0 VCLK	1.5 VCLK	2.0 VCLK
64	.12 VCLK	.25 VCLK	.50 VCLK	.75 VCLK	1.0 VCLK

7. BANDWIDTH NOTE FOR CL-GD5440 WITH 64-BIT INTERFACE

Although the CL-GD5440 appears to use a 64-bit display memory interface when configured for 2 Mbytes, the bandwidth calculations must be based on a 32-bit interface. This is because the CL-GD5440 has a 32-bit internal bus.

Appendix B4

Current Reference

CURRENT REFERENCE

1. INTRODUCTION

IREF (pin 103) requires an external resistor to set the full-scale current. The equations in the following section calculate the resistor value.

2. RSet: FULL-SCALE CURRENT SET RESISTOR VALUE

RSet (pin 94) has a resistor to the AVSS[3:2] node that sets the full-scale DAC output current. This resistor should have a 0.1-µF capacitor in parallel. Select the resistor value with the following equations.

Equation B4-1 is part of the chip specification. The resistor value (*RSet*) is in terms of current reference (*IREF*).

$$\mathrm{RSet} = 1.2\ \mathrm{V} \div \mathrm{IREF}$$ **Equation B4-1.**

Equation B4-2 is also part of the chip specification. The current reference (*IREF*) is in terms of desired full-scale current out (*IFullScale*).

$$\mathrm{IREF} = \mathrm{IFullScale} \div (63 \div 30)$$ **Equation B4-2.**

Equation B4-3 derives from Ohm's Law. Full-scale current out (*IFullScale*) is in terms of desired full-scale voltage (*VFullScale*) and the load the DACs look into (*Load*).

$$\mathrm{IFullScale} = \mathrm{VFullScale} \div \mathrm{Load}$$ **Equation B4-3.**

These three equations are combined (Equation B4-4) to yield a single equation that the resistor value (*RSet*), in terms of desired full-scale voltage (*VFullScale*), and the load the DACs look into (*Load*).

$$\mathrm{RSet} = (2.52\ \mathrm{V} \times \mathrm{Load}) \div \mathrm{VFullScale}$$ **Equation B4-4.**

For a full-scale voltage of 700 mV and a load of 37.5 Ω, the calculated value of *RSet* is 135 Ω. If another value is selected, the output voltage scales linearly. If the load is other than 37.5 Ω, the equation has to be evaluated for the actual load.

Appendix B5

Configuration Notes

CONFIGURATION NOTES

1. INTRODUCTION

When RST# is active, the CL-GD5446 loads the levels on MD[63:48] in 16 internal latches. These latches control some fundamental properties of the device, such as the host bus interface. These configuration latches are named CF[15:0].

2. CONFIGURATION SUMMARY

Each of the memory data lines MD[63:48] has an internal pull-up resistor (nominally 250 kΩ). The default (if no pull-down resistor is installed) is '1'. If '0' is to be loaded into the latch associated with a given MD line, an external pull-down resistor (typically 6.8 kΩ) must be installed. Table B5-1 provides an overview of the Configuration bits. Table B5-1 indicates the configuration if a pull-down resistor is installed.

Table B5-1. CL-GD5446 Configuration Bits

MD Bit	CF Bit	Pull-Down Resistor Installed	Readable At:
63	15	Enable pin scan	–
62	14	Claim PCI interrupt	PCI3C[8]
61	13	Reserved	–
60	12	MCLK pin configuration	–
59	11	Asymmetric DRAM addressing	–
58	10	Multiple CAS# DRAMs	–
57	9	Extended DRAM timing	SRF[2]
56	8	Relocated I/O, GPIO	–
55	7	Reserved	–
54	6	Feature connector output disable	GR17[3]
53	5	MCLK pin configuration (see register CF12)	–
52	4	Relocated I/O, GPIO (see register CF8)	–
51	3	PCI14 enable (see register CF8) (Revision A)	–
50	2	System Bus Select 2	SR17[5]
49	1	System Bus Select 1	SR17[4]
48	0	System Bus Select 0	SR17[3]

3. CONFIGURATION DETAILS

CF15 (MD63) Enable Pin Scan

If no pull-down resistor is installed, the CL-GD5446 is not put into Pin-Scan mode and functions normally. If a pull-down resistor is installed (or if MD63 is driven low by a tester), the CL-GD5446 is placed into Pin-Scan mode (see Appendix B7, "Pin Scan").

CF14 (MD62) PCI Interrupt Request Control

If no pull-down resistor is installed, PCI3C[8] is '0' and the PCI interrupt is not claimed. If a pull-down resistor is installed, PCI3C[8] is '1' and the PCI interrupt is claimed.

CF13 (MD61) Reserved

The reserved bits are for future expansion. Do not install pull-down resistors on these lines.

CF12 (MD60) MCLK Pin Configuration

This pin is used in conjunction with MD53 (CF5) to configure the MCLK pin (pin 16). This is detailed in Table B5-2. '0' indicates that a pull-down resistor is installed on the corresponding pin.

Table B5-2. MCLK Configuration: CF12, CF5

MD60 CF12	MD53 CF5	MCLK Pin	Note
1	1	Out 0	Follows GR17[5]
1	0	MCLK out	
0	1	VCLK VCO out	Prior to post-scalar
0	0	MCLK in	Factory testing only

CF11 (MD59) Asymmetric DRAM Addressing

If no pull-down resistor is installed, the DRAM Row and Column addresses are on MA[8:0]. If a pull-down resistor is installed, the DRAM Row address is on MA[9:0] and the DRAM Column address is on MA[8:1].

CF10 (MD58) Multiple-CAS# DRAM Interface

A pull-down must be installed on this pin. The CL-GD5446 does not support WE# steering.

CF9 (MD57) Extended-RAS# Timing

If no pull-down is installed, 6-cycle DRAM timing is generated. A Random cycle requires six MCLK periods; a Fast-page Mode cycle requires two MCLK periods. If a pull-down resistor is installed, extended DRAM timing is used. A Random cycle requires seven MCLK periods; a Fast-page Mode cycle requires two MCLK periods. See Appendix B3, "Memory Configurations and Timing", for detailed information on DRAM configuration and timing. Extended-RAS# timing must be configured to enable EDO timing.

CF8 (MD56) Relocated I/O, GPI/O Configuration (Revision A)

This pin is used in conjunction with MD52 (CF4) and MD51 (CF3) to configure VGA register relocation and GPI/O. See Chapter 9, "Programming Notes", for the addresses used for VGA register relocation. See Appendix B11, "General-Purpose I/O", for information regarding GPI/O. Table B5-3 summarizes the configurations. '0' indicates that a pull-down resistor is installed.

Table B5-3. Relocated I/O, GPI/O Configuration: Revision A

CF3 MD51	CF8 MD56	CF4 MD52	GPIO	VGA Register Relocation	PCI14
1	d/c	d/c	Disabled	Disabled	Returns all '0's
0	0	0	128 bytes, I/O	Disabled	31:7 specify base, bit 0 = 1
0	0	1	32 bytes, memory	32-bytes memory	31:12 specify base, bit 0 = 0 (4096 bytes claimed)
0	1	0	32 bytes, I/O	Disabled	31:5 specify base, bit 0 = 1
0	1	1	Disabled	32-bytes, I/O	31:12 specify base, bit 0 = 1

CF8 (MD56) GPIO Configuration (Revision B)

This pin is used in conjunction with MD52 (CF4) to configure GPIO. The VGA registers are always accessible through PCI14 on Revision B. See Appendix B11, "General-Purpose I/O", for information regarding GPIO. Table B5-4 summarizes the configurations.

Table B5-4. Relocated I/O, GPI/O Configuration: Revision B

CF8 MD56	CF4 MD52	GPIO	PCI18
0	0	Reserved	Do not configure
0	1	32 bytes, memory	31:12 specify base, bit 0 = 0 (4096 bytes claimed)
1	0	32 bytes, I/O	31:5 specify base, bit 0 = 1
1	1	Disabled	Read-only – '0'

CF7 (MD55) Reserved

The reserved bits are for future expansion. Do not install pull-down resistors on these lines.

CF6 (MD54) Feature Connector Output Disable

If no pull-down resistor is installed, GR17[3] powers up as '0', and the feature connector output drivers are normally controlled by ESYNC#, EDCLK#, and EVIDEO#. If a pull-down resistor is installed, GR17[3] powers up as '1', and the feature connector drivers are disabled. This configuration bit can be overridden by programming GR17[3].

CF5 (MD53) MCLK Pin Configuration

See register CF12.

CF4 (MD52) Relocated I/O, GPIO Configuration

See register CF8.

CF3 (MD51) Relocated I/O, GPIO Configuration

See register CF8.

CF2:0 (MD[50:48] System Bus Select [2:0]

These three bits specify the host bus connected to CL-GD5446.

Table B5-5. System Bus Select

CF2 MD50	CF1 MD49	CF0 MD48	System Bus	Note
0	0	0	Reserved	
0	0	1	Reserved	
0	1	0	Reserved	
0	1	1	Reserved	
1	0	0	PCI	
1	0	1	Reserved	
1	1	0	VESA VL-Bus	Reference only, not supported
1	1	1	Reserved	

Appendix B6

Signature Generator

SIGNATURE GENERATOR

1. INTRODUCTION

The CL-GD5446 has SG (signature generator) logic to test the frame buffer and video logic, the BitBLT engine, and the V-Port, all at full speed.

2. TESTING

SG testing involves sending a known good bitstream, generated by a known good device, into a feedback shift register, capturing a 16-bit value (the signature). Testing programs can reproduce the conditions that generated the bitstream and capture the signature generated by the device or system under test.

If the signature generated by the test device matches the known-good signature, there is excellent reason to believe the test device is behaving identically to the known good device. This allows automated testing that is both faster and less error prone than manual testing. The SG is used extensively in the manufacturing test.

Appendix B7

Pin Scan

PIN SCAN

1. INTRODUCTION

Pin-Scan testing is a technique for verifying that an IC has been properly soldered to a circuit board. Any IC signal pin that is not connected to the board, or that is shorted to any neighboring pin or trace, can be detected using this technique. The advantage of Pin-Scan testing is that the test patterns to verify full board connectivity are much simpler than would otherwise be possible. The pins are connected sequentially around the IC in a single chain, so that the value on each output pin depends only on the values applied to other pins, rather than the internal state of the VGA processor. In addition, the Pin-Scan logic is strictly combinatorial, so no clock pulses are required.

The first pin in the chain is an input pin; the last pin is an output pin. Each input signal is XOR'ed (exclusive-OR'ed) with the scan data from its lower-numbered neighboring input or output pin. The result of this XOR is passed to its higher-numbered neighbor. Each output pin will be driven with the value passed from its lower-numbered neighbor; that value is inverted and passed to its higher-numbered neighbor.

2. TEST METHOD

In Pin-Scan mode, the test program begins by driving all the input pins to '1', and verifying that the output pins match the values shown in the table that follows. On subsequent cycles, the program drives each input pin, one at a time, to '0' and verifies that all the 'down-stream' outputs match the values shown. In each case, the output is inverted from the value for the all-zeroes case.

If the value applied to an input pin is changed and the 'down-stream' output pins do not change, then that input is shorted or not soldered. If any single output is wrong, then it is either shorted or not soldered.

2.1 Entering Pin-Scan Mode

The CL-GD5446 is placed into Pin-Scan mode by making RST# low for at least 20 ns while MD63 is low, then making RST# high.

2.2 Exiting Pin-Scan Mode

The CL-GD5446 is removed from Pin-Scan mode by making RST# low with MD63 high.

3. PIN SCAN ORDER

Table B7-1 indicates the pins that are outputs, and indicates the level to be expected for the two cases of 'all inputs = 1' and 'one upstream input = 0'.

Table B7-1. Pin Scan Order

Pin Name	Pin Number	In/Out	All Inputs = 1	1 Input = 0	Alternate Pin Name(s)
C/BE0#	20	In			
C/BE1#	21	In			
C/BE2#	22	In			
C/BE3#	23	In			
BIOSA0	24	In			GPD0, Prog In
BIOSA1	25	In			GPD1
BIOSA2	26	In			GPD2
BIOSA3	27	In			GPD3
BIOSA4	28	In			GPD4
BIOSA5	29	In			GPD5
BIOSA6	30	In			GPD6
BIOSA7	31	In			GPD7
BIOSA8	32	In			GPIORD#
EROM#	33	Out	0	1	
BIOSA9	34	In			GPIOWR#
BIOSA10/P15	35	In			GPD15, PIXD15
BIOSA11/P14	36	In			GPD14, PIXD14
BIOSA12/P13	37	In			GPD13, PIXD13
P12	38	In			GPD12, PIXD12
P11	39	In			GPD11, PIXD11
BIOSA13/P10	40	In			GPD10, PIXD10
BIOSA14	41	In			GPA0
BIOSA15	42	In			GPA1
CLK/P15	43	In			GPD15, PIXD15
P9	44	In			GPD9, PIXD9
P8	45	In			GPD8, PIXD8
FRAME#	46	In			
IRDY#	47	In			
STOP#	48	In			
TRDY#	49	In			

Table B7-1. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 1	1 Input = 0	Alternate Pin Name(s)
PAR	50	In			
RST#	51	In			
IDSEL#	53	In			
AD31	54	In			
AD30	55	In			
AD29	56	In			
AD28	57	In			
AD27	58	In			
AD26	59	In			
AD25	60	In			
AD24	61	In			
AD23	62	In			
AD22	63	In			
DEVSEL#	65	Out	1	0	
	66	In			GPRDY/DT
AD21	68	In			
AD20	69	In			
AD19	70	In			
AD18	71	In			
AD17	72	In			
AD16	73	In			
AD15	74	In			
AD14	75	In			
AD13	76	In			
AD12	78	In			
AD11	79	In			
AD10	80	In			
AD9	81	In			
AD8	82	In			
AD7	84	In			

Table B7-1. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 1	1 Input = 0	Alternate Pin Name(s)
AD6	85	In			
AD5	87	In			
AD4	88	In			
AD3	89	In			
AD2	90	In			
AD1	91	In			
AD0	92	In			
VSYNC	96	In			
HSYNC	98	In			
TWR#	105	In			GPA2
DDCDAT	106	In			
DDCCLK	107	Out	0	1	
OVRW#	109	Out	1	0	GPA3
P0	110	In			PIXD0
P1	111	In			PIXD1
P2	112	In			PIXD2
P3	113	In			PIXD3
P4	115	In			PIXD4
P5	116	In			PIXD5
P6	117	In			PIXD6
P7	118	In			PIXD7
EVIDEO#	120	In			VACT
ESYNC#	122	In			Prog Out 1
EDCLK#	123	In			VREF
DCLK	125	In			PIXCLK
BLANK#	126	In			HREF
MD31	127	In			
MD30	128	In			
MD29	129	In			
MD28	130	In			

Table B7-1. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 1	1 Input = 0	Alternate Pin Name(s)
MD27	131	In			
MD26	132	In			
MD25	133	In			
MD24	134	In			
CAS3#	135	In			
MD23	137	In			
MD22	138	In			
MD21	139	In			
MD20	140	In			
MD19	141	In			
MD18	142	In			
MD17	143	In			
MD16	144	In			
CAS2#	145	In			
MD15	147	In			
MD14	148	In			
MD13	149	In			
MD12	151	In			
MD11	152	In			
MD10	153	In			
MD9	154	In			
MD8	155	In			
CAS1#	156	In			
MD63	157	In			
MD62	158	In			
MD61	159	In			
MD60	160	In			
MD59	161	In			
MD58	162	In			
MD57	163	In			

Table B7-1. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 1	1 Input = 0	Alternate Pin Name(s)
MD56	164	In			
CAS7#	166	In			
WE#	167	In			
RAS1#	168	Out	0	1	
RAS0#	169	Out	1	0	
MD55	170	In			
MD54	171	In			
MD53	172	In			
MD52	173	In			
MD51	174	In			
MD50	175	In			
MD49	176	In			
MD48	177	In			
CAS6#	179	In			
MA0	180	Out	1	0	
MA1	182	Out	0	1	
MA2	183	Out	1	0	
MA3	184	Out	0	1	
MA4	185	Out	1	0	
MA5	186	Out	0	1	
MA6	187	Out	1	0	
MA7	188	Out	0	1	
MA8	189	Out	1	0	
MA9	190	Out	0	1	
MD47	191	In			
MD46	192	In			
MD45	193	In			
MD44	194	In			
MD43	196	In			
MD42	197	In			

Table B7-1. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 1	1 Input = 0	Alternate Pin Name(s)
MD41	198	In			
MD40	199	In			
CAS5#	200	In			
MD39	201	In			
MD38	202	In			
MD37	203	In			
MD36	204	In			
MD35	205	In			
MD34	206	In			
MD33	207	In			
M32	2	In			
CAS4#	3	In			
CAS0#	4	In			
MD7	5	In			
MD6	6	In			
MD5	7	In			
MD4	8	In			
MD3	9	In			
MD2	10	In			
MD1	11	In			
MD0	12	In			
MCLK	16	In			
INTR#	19	Out	1	0	

Table B7-2. Pin Scan Order

Pin Name	Pin Number	In/Out	All Inputs = 0	1 Input = 1	ISA Note	PCI Note
BE0#	20	In			RERESH	C/BE#0
BE1#	21	In			SBHE*	C/BE#1
BE2#	22	In			SA0	C/BE#2
BE3#	23	In			SA1	C/BE#3
A2	24	In			SA2	BIOSA0
A3	25	In			SA3	BIOSA1
A4	26	In			SA4	BIOSA2
A5	27	In			SA5	BIOSA3
A6	28	In			SA6	BIOSA4
A7	29	In			SA7	BIOSA5
A8	30	In			SA8	BIOSA6
A9	31	In			SA9	BIOSA7
A10	32	In			SA10	BIOSA8
EROM*	33	Out	1	0		
A11	34	In			SA11	BIOSA9
A12	35	In			SA12	BIOSA10
A13	36	In			SA13	BIOSA11
A14	37	In			SA14	BIOSA12
A15	38	In			SA15	BIOSA13
A16	39	In			LA16	BIOSA14
A17	40	In			LA17	BIOSA15
A18	41	In			LA18	n/c[a]
A19	42	In			LA19	n/c
LCLK	43	In			IOW*	CLK
A20	44	In			LA20	n/c
A21	45	In			LA21	n/c
ADS#	46	In			BALE	FRAME#
RDYR#	47	In			AEN	IRDY#
LOWMEM	48	In			LA22	STOP#

Table B7-2. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 0	1 Input = 1	ISA Note	PCI Note
RDY#	49	In			IOCHRDY	TRDY#
HIMEM	50	In			LA23	PAR
RESET	51	In				RST
W/R#	53	In			IOR*	IDSEL#
D31	54	In			n/c	AD31
D30	55	In			n/c	AD30
D29	56	In			n/c	AD29
D28	57	In			n/c	AD28
D27	58	In			n/c	AD27
D26	59	In			n/c	AD26
D25	60	In			n/c	AD25
D24	61	In			n/c	AD24
D23	62	In			n/c	AD23
D22	63	In			n/c	AD22
LDEV#	65	Out	1	0	MCS16*	DEVSEL#
M/IO#	66	In			MEMR*	LOCK#
D21	68	In			n/c	AD21
D20	69	In			n/c	AD20
D19	70	In			n/c	AD19
D18	71	In			MEMW*	AD18
D17	72	In			IOCS16*	AD17
D16	73	In			IRQ	AD16
D15	74	In			SD15	AD15
D14	75	In			SD14	AD14
D13	76	In			SD13	AD13
D12	78	In			SD12	AD12
D11	79	In			SD11	AD11
D10	80	In			SD10	AD10
D9	81	In			SD9	AD9

Table B7-2. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 0	1 Input = 1	ISA Note	PCI Note
D8	82	In			SD8	AD8
D7	84	In			SD7	AD7
D6	85	In			SD6	AD6
D5	87	In			SD5	AD5
D4	88	In			SD4	AD4
D3	89	In			SD3	AD3
D2	90	In			SD2	AD2
D1	91	In			SD1	AD1
D0	92	In			SD0	AD0
VSYNC	96	In				
HSYNC	98	In				
TWR*	105	In				
OVRW*	106	Out	1	0		
EEDI	107	In				
EECS	108	Out	0	1		
P0	110	In				
P1	111	In				
P2	112	In				
P3	113	In				
P4	115	In				
P5	116	In				
P6	117	In				
P7	118	In				
EVIDEO*	120	In				
ESYNC*	122	In				
EDCLK*	123	In				
DCLK	125	In				
BLANK*	126	In				
MD31	127	In				

Table B7-2. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 0	1 Input = 1	ISA Note	PCI Note
MD30	128	In				
MD29	129	In				
MD28	130	In				
MD27	131	In				
MD26	132	In				
MD25	133	In				
MD24	134	In				
CAS3*	135	In				
MD63	137	In				
MD62	138	In				
MD61	139	In				
MD60	140	In				
MD59	141	In				
MD58	142	In				
MD57	143	In				
MD56	144	In				
CAS7*	145	In				
MD55	147	In				
MD54	148	In				
MD53	149	In				
MD52	151	In				
MD51	152	In				
MD50	153	In				
MD49	154	In				
MD48	155	In				
CAS6*	156	In				
MD47	157	In				
MD46	158	In				
MD45	159	In				

Table B7-2. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 0	1 Input = 1	ISA Note	PCI Note
MD44	160	In				
MD43	161	In				
MD42	162	In				
MD41	163	In				
MD40	164	In				
CAS5*	166	In				
WE*	167	In				
RAS1*	168	Out	1	0		
RAS0*/OE*	169	Out	0	1		
MD39	170	In				
MD38	171	In				
MD37	172	In				
MD36	173	In				
MD35	174	In				
MD34	175	In				
MD33	176	In				
MD32	177	In				
CAS4*	179	In				
MA0	180	Out	1	0		
MA1	182	Out	0	1		
MA2	183	Out	1	0		
MA3	184	Out	0	1		
MA4	185	Out	1	0		
MA5	186	Out	0	1		
MA6	187	Out	1	0		
MA7	188	Out	0	1		
MA8	189	Out	1	0		
MA9	190	Out	0	1		
MD23	191	In				

Table B7-2. Pin Scan Order *(cont.)*

Pin Name	Pin Number	In/Out	All Inputs = 0	1 Input = 1	ISA Note	PCI Note
MD22	192	In				
MD21	193	In				
MD20	194	In				
MD19	196	In				
MD18	197	In				
MD17	198	In				
MD16	199	In				
CAS2*	200	In				
MD15	201	In				
MD14	202	In				
MD13	203	In				
MD12	204	In				
MD11	205	In				
MD10	206	In				
CAS1*	207	In				
MD9	2	In				
MD8	3	In				
CAS0*	4	In				
MD7	5	In				
MD6	6	In				
MD5	7	In				
MD4	8	In				
MD3	9	In				
MD2	10	In				
MD1	11	In				
MD0	12	In				
MCLK	16	In				
INTR	19	Out	1	0	0WS	INTR

[a] 'n/c' indicates a pin that is a no-connect.

Some pins are not accessible in the pin scan, as listed in Table B7-3. These pins are analog, not digital.

Table B7-3. Non-Scanned Pins

Pin Name	Number
XTAL	14
OSC	17
RSET	94
BLUE	100
GREEN	101
RED	102

Appendix B8

DDC2B/I²C Support

DDC2B/I2C SUPPORT

1. INTRODUCTION

The CL-GD5446 supports a two-pin open-collector interface. This is typically used for DDC2B or I2C support.

The interface is mechanized with two open-collector pins that can be driven to ground with register bits and whose levels can be sensed with register bits. Table B8-1 shows the pins and the associated register bits. SR8[6] must be programmed to '1' for the remaining bits in register SR8 to be configured for DDC support.

Table B8-1. DDC2B Support

CL-GD5446 Pin Name	Alternate Pin Name	Read Port	Write Port	DB15 Pin	Feature Connector
DDCDAT (pin 106)	EEDI	SR8[7]	SR8[1]	MID1 (pin 12)	Z13
DDCCLK (pin 107)	EECS	SR8[2]	SR8[0]	MID3 (pin 15)	Z7

Appendix B9

GENLOCK Support

GENLOCK SUPPORT

1. INTRODUCTION

When mixing video from multiple sources onto a single screen, it is necessary to synchronize the sources. This synchronization can be done by using a frame buffer, or it can be done by forcing the multiple video sources into synchronization with each other. In the latter case, one of the sources provides the timing and the others slave themselves to this timing. This is referred to as *GENLOCK*.

2. GENLOCK ON THE CL-GD5446

The CL-GD5446 can GENLOCK to an external VSYNC, an external HSYNC, or both. When using GENLOCK, the CL-GD5446 must be supplied with an external VCLK (it does not recover the pixel clock from HSYNC). The clock is supplied on the DCLK pin, with EDCLK# being held low. 3C2[3:2] must be programmed to '1X' so that the external clock drives the CRTC as well as the DAC.

The external master supplies HSYNC and VSYNC to the display, as well as to the CL-GD5446. The diagram in Figure B9-1 shows the connections that must be made when video is being overlaid via the P[7:0] bus.

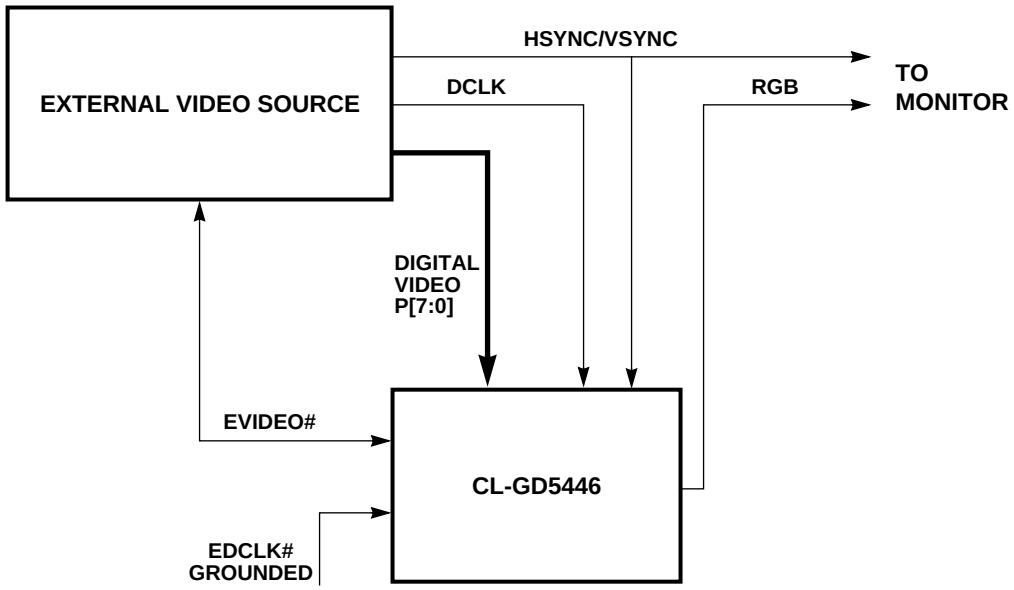


Figure B9-1. Video Overlay Connections

The external source provides all raster timing in the form of DCLK and both SYNCs. EDCLK# on the CL-GD5446 is tied low so that DCLK is an input. The CL-GD5446 is programmed for both VSYNC and HSYNC GENLOCK so both SYNC pins are inputs, preventing a bus crash. The CL-GD5446 is also programmed for overlay mode (see Extension register CR1A) so that its P[7:0] pins are inputs. Depending on the overlay mode, EVIDEO# is an input or an output, as indicated in Table B9-1. Also, depending on the overlay mode, the digital video from the external source may be either 8- or 16-bpp.

Table B9-1. Overlay Mode I/O Indicator

CR1A[3:2]	Overlay Mode	EVIDEO#	Note
01	EVIDEO#	In	External source provides overlay timing.
10	EVIDEO# AND'ed with Color Key	In	External source timing AND'ed with Color key.
11	Color Key	Out	VGA data specifies pixels to be overlaid.

3. VSYNC GENLOCK PROGRAMMING

If CR1C[7] is programmed to '1', VSYNC GENLOCK is enabled. The VSYNC pin becomes an input. The falling edge of the VSYNC input is synchronized to VCLK and, after two VCLK edges, the vertical counter resets on the next HSYNC. The next HSYNC after that signals the beginning of the first scanline of the next field.

Program VTOTAL (registers CR7–CR6) so that the external VSYNC occurs before the programmed value is reached. That is, program the CRTC timing for a somewhat greater than the actual vertical period, then truncate as necessary.

4. HSYNC GENLOCK PROGRAMMING

If CR1C[6] is programmed to '1', HSYNC GENLOCK is enabled. The HSYNC pin becomes an input. The falling edge of the HSYNC input is synchronized to VCLK, and after two VCLK edges, the character clock generator (which is a VCLK counter) is cleared.

The next character clock (now synchronized to HSYNC at a VCLK resolution) forces the horizontal timing generator to the state equivalent to horizontal total. The horizontal counter is cleared three character clocks later, and display data begins after the display pipeline delays normally present for the current display mode (typically three character clocks plus five VCLKs).

Program HTOTAL (register CR0) so that the external HSYNC occurs before the programmed value is reached. That is, program the CRTC timing for somewhat greater than the actual horizontal period, then truncate as necessary.

Appendix B10

Manufacturing Test

MANUFACTURING TEST

1. INTRODUCTION

The Manufacturing Test program (MFGTST.EXE) tests a graphics system based on the CL-GD5446. This can be the final test before the product is shipped. These tests include:

- Write/read/compare tests of all register groups.
- Write/read/compare tests of all frame buffer RAM.
- Display patterns for visual verification of all primary display modes.
- 2D drawing engine verification.

Also included is a complete set of miscellaneous display tests designed to verify proper operation of the CL-GD5446. Extensive use of the enhanced signature generator automates system verification.

2. OPERATING INSTRUCTIONS

The MFGTST.EXE program is easy to install and operate. The [Tab] key allows the user to jump between the three different windows on the MFGTST display: Test Groups, Test Cases, and Test Log windows. The arrow keys highlight the users selections and controls the cursor within the MFGTST windows.

To obtain additional help concerning a function in MFGTST, the test technician can press the F1 key. Help in the MFGTST dialog boxes is available by tabbing to the Help option and pressing [Enter].

To exit to the DOS prompt, the test technician can simultaneously press the [Alt] and [X] keys or tab to the diamond in the upper-left corner of the MFGTST display and press [Enter].

When a triangle is displayed in the lower-right corner of the Test Cases window, there are more listed test cases for a specific group. To go to the last Test Cases, the test technician can use the arrow keys. To view the remainder of the tests listed, continue using the down arrow key.

3. INSTALLING AND STARTING MFGTST

To install the MFGTST.EXE program, first copy two files onto the disk drive. An optional set of .MFG files can also be copied to the drive to provide additional test cases. Follow the procedure listed below to install and start the MFGTST program.

1) Create a new directory called MFGTST and copy the files MFGTST.EXE and DOS4GW.EXE (a DOS extender file allowing the program to run as a 32-bit program) into the mfgtst directory. (Reference the DOS documentation on creating directories and copying files. DOS4GW.EXE is not required in a future release).

2) To create a custom initialization file (MFGTST.INI) when first starting MFGTST, refer to Section 3.1, “Command Line Options”. This provides the correct command line extensions and cancels the next two steps.

3) Type `mfgtst` at the MFGTST directory to start the MFGTST.EXE program.

4) Press [ENTER].

NOTE: To redirect MFGTST to a serial terminal setup for ANSI/VT100 emulation, type `mfgtst /com:1:19200:n:8:1`. This sets MFGTST to load and direct output to COM1 at 19200 baud, no parity, 8 data bits, and 1 stop bit.

5) A dialog box inquires whether to "Save the Program Configuration (MFGTST.INI)?" Select OK to create the initialization file MFGTST.INI. Select Cancel if an initialization file is unnecessary. To customize your .INI file, press [Alt] and [X] simultaneously and go back to the DOS prompt. Then reference Section 3.1, "Command Line Options" to obtain the correct command line extensions for customization.

For effective and efficient use of the MFGTST.EXE program, read Section 4 on page B10-4 before running any tests.

3.1 Command Line Options

MFGTST creates an initialization file (MFGTST.INI). Add command line options to customize or change the initialization file. To have the MFGTST program custom-build a .INI file:

1) Type `mfgtst command` at the mfgtst directory when starting the MFGTST program to create or change the MFGTST.INI (where `command` is a command line option selected from Table B10-1). For example, to run in Quiet mode, type `mfgtst/q` at the MFGTST directory.

Table B10-1. MFGTST Command Line Options

Command	Description of Option
/v	Verbose Mode
/a	OEM Auto
/c	Continue
/b	Batch mode
/f	Log file
/m	OEM Menu mode
/q	Quiet mode
/r	Set random seed
/e	Engineering mode
/vga	Output to VGA device
/vgaalt	Output to VGA device using alternate colors
/mda	Output to MDA (monochrome) adapter
/com	Output to COM port. Input several parameters.
/tl:<n>	Set number of output terminal lines to <n>. (Default = 25 lines.)
/ll:<n>	Set number of log window (menu mode) lines to <n>. (Default = 6 lines.)
/fu:<fl>	Set User Configuration file to <fl> and read in at start-up.

2) A MFGTST prompts a dialog box to save the Program Configuration. Press OK to have MFGTST automatically create the new MFGTST.INI file.

The MFGTST.INI dialog box is always available to create a new initialization file. This is invoked by pressing [Alt] and [F] simultaneously and then pressing [S], which saves the current configuration. The command line options for loading MFGTST are saved in the .INI file. All command line options override corresponding settings in the initialization file (MFGTST.INI).

4. USING MFGTST

To run a single test in a Test Group once, tab the cursor to the Test Cases window and use the arrow keys to scroll up and down the list. When the name of the desired test is highlighted, press [Enter]. A 'p:' in front of the Test Group name signifies that the test has passed. A 'F!' in front of the Test Group name signifies that the test has failed. A 'c!' indicates the test did not run.

To select or deselect several Test Groups and Test Cases, press the space bar. A symbol appears to the left of the selected groups and cases. Tab between the Test Groups and Test Cases and use the arrow keys to scroll up and down the list. Simultaneously pressing [Alt] and [M], marks all tests in a group to run (symbols appear next to all marked tests). To unmark all tests in a group, press [Alt] and [U] simultaneously.

To run several Test Cases for a Test Group, mark the appropriate tests, and tab the cursor to the Test Groups window. Press [Enter] on the name of the Test Group. All selected tests in that group then run.

To run all selected Test Cases in all marked Test Groups, press [Alt] and [R] to run all marked tests.

4.1 Special Keystrokes

To easily navigate around the three windows in the MFGTST display, refer to Table B10-2. Using the keystrokes marked with an asterisk (*) in the Description column prompts additional text windows, described in detail below.

Table B10-2. MFGTST Keystrokes

Keystroke	Description
F1	Help *
F2	Run test controls dialog box *
Alt+F	File operations *
Alt+R	Run all marked tests
Alt+X	Exit program
Alt+V	Display modes dialog
Alt+M	Mark all test cases in a group
Alt+U	Unmark all test cases in a group

4.1.1 Using the [F1] Keystroke

When you press [F1], the Help Topic dialog box appears displaying the following options:

- (A)bout
- (K)eystrokes
- (T)est:…
- (R)esults:…

To get to the respective option's text box, tab to the option and press [Enter] or press the highlighted letter of the option. Selecting the '(A)bout' option displays the Cirrus Logic copyright and version information for the MFGTST program. Selecting the '(K)eystrokes' option displays the keystroke information found in Table B10-2. The '(T)est:…' option is available, depending on the currently highlighted test. When '(T)est:…' is available, it provides additional information about that test. The '(R)esults:…' option is available, depending on the currently highlighted test. When '(R)esults:…' is available, it provides additional information about the results of that test.

4.1.2 Using the [F2] Keystroke

Press [F2] to bring up a dialog box of test control options that controls looping through various tests. Note that not all test controls are fully implemented. To use the test controls, click on or tab to one of the following options and press [Enter].

- [Options for Run All Marked Tests]
- [General Options for Run Each Test]
- [Specific Options for Run Each Test]
- [Run All Marked Tests]

The '[Options for Run All Marked Tests]' controls duration and the amount of loop times in the tests. '[General Options for Run Each Test]' controls the duration for repeating the test and for what display modes and random speed. '[Specific Options for Run Each Test]' is not yet implemented. '[Run All Marked Tests]' runs all selected tests in all selected groups.

4.1.3 Using the [Alt]+[F] Keystrokes

After simultaneously pressing the [Alt]+[F] keys, the File Operations menu appears and displays the following options:

- (S)ave Configuration
- (U)ser File...
- (L)og File...
- E(x)it Program

To use the above options, tab to the option and press [Enter] or press the highlighted letter of the option. Selecting '(S)ave Configuration' saves a new MFGTST.INI file with the current options. If no MFGTST.INI file is found, this option automatically comes up at launch. Selecting '(U)ser File' prompts a dialog box, allowing loading and saving the User Options File by file name. User Options Files currently acknowledge the tests/groups that are selected and provide additional information about running tests/options. Selecting '(L)og File' prompts a dialog

box to create log files. Press the ‘Log File Help’ button for additional information. Selecting ‘E(x)it Program’ closes the MFGTST program.

5. UPDATES

The manufacturing test program is regularly updated. As time permits, more tests are added to the manufacturing test software. These tests support new functions and features of current and future products. Contact Cirrus Logic for up-to-date manufacturing test software and documentation.

Appendix B11

General-Purpose I/O

GENERAL-PURPOSE I/O

1. INTRODUCTION

The CL-GD5446 provides support for a single additional peripheral device, such as an MPEG decoder or adapter card. The support includes address decoding, an 8- or 16-bit data port, and cycle timing generation. This allows an additional device without violating the PCI 'one-load' specification.

2. GPIO CONFIGURATION

On Revision A, three configuration bits specify whether GPIO exists and the address space used. In addition, register PCI14 specifies where the GPIO is in the address space (Revision A).

Table B11-1. GPIO Configuration (Revision A)[a]

CF3 MD51	CF8 MD56	CF4 MD52	GPIO	VGA Register Relocation	PCI14
1	d/c	d/c	Disabled	Disabled	Returns all '0's
0	0	0	128-bytes I/O	Disabled	31:7 specify base, bit 0 = 1
0	0	1	32 bytes memory	32 bytes memory	31:12 specify base, bit 0 = 0 4096 bytes claimed
0	1	0	32 bytes I/O	Disabled	31:5 specify base, bit 0 = 1
0	1	1	Disabled	32-bytes I/O	31:12 specify base, bit 0 = 1

[a] '0' indicates the presence of a pull-down resistor.

In Revision A, GPIO can be configured for 128 or 32 bytes of I/O space or 32 bytes of memory space. When GPIO is configured for memory space, the VGA registers are relocated into memory space as well and GPIO begins 32 bytes above the offset in register PCI14. Whenever register PCI14 is configured for I/O, address bits 31:16 are not decoded.

In Revision B, two configuration bits specify how GPIO is configured and PCI18 claims the address space.

Table B11-2. GPIO Configuration (Revision B)

CF8 MD56	CF4 MD52	GPIO	PCI18
0	0	Reserved	Do not configure
0	1	32 bytes memory	31:12 specify base, bit 0 = 0 4096 bytes claimed
1	0	32 bytes I/O	31:5 specify base, bit 0 = 1
1	1	Disabled	31:12 specify base, bit 0 = 1

3. PINS REDEFINED

Table B11-3 indicates pins used for the GPIO bus.

Table B11-3. GPIO Pins

Pin Number	GPIO	Pin Name	Note
35	GPD15	BIOSA10	Pin also defined as PIXD15
36	GPD14	BIOSA11	Pin also defined as PIXD14
37	GPD13	BIOSA12	Pin also defined as PIXD13
38	GPD12	Reserved	Pin also defined as PIXD12
39	GPD11	Reserved	Pin also defined as PIXD11
40	GPD10	BIOSA13	Pin also defined as PIXD10
44	GPD9	Reserved	Pin also defined as PIXD9
45	GPD8	Reserved	Pin also defined as PIXD8
31	GPD7	BIOSA7	
30	GPD6	BIOSA6	
29	GPD5	BIOSA5	
28	GPD4	BIOSA4	
27	GPD3	BIOSA3	
26	GPD2	BIOSA2	
25	GPD1	BIOSA1	
24	GPD0	BIOSA0	
14	GPA6	XTAL	128 bytes only
42	GPA5/GPA1	BIOSA15	GPA5 for 128 bytes, GPA1 for 32 bytes
41	GPA4/GPA0	BIOSA14	GPA4 for 128 bytes, GPA0 for 32 bytes
109	GPA3	OVRW#	
105	GPA2	TWR#	
108	GPCS#	Reserved	Active-low Chip Select
32	GPIORD#	BIOSA8	Active-low IO Read Command
34	GPIOWR#	BIOSA9	Active-low IO Write Command
66	GPRDY/GPDT	Reserved	RDY or DTACK#

GPD[15:0] is the bidirectional data bus. GR19[0] controls the bus width as indicated in Table B11-4.

Table B11-4. GPIO Bus Width Notes

GR19[0]	Bus Width	Word Access (PCI Bus)	Byte Access (PCI Bus)
0	8 bits	Converted to two byte accesses	Single byte access
1	16 bits	Single word access	Not permitted

GPA[5:2] or GPA[3:0] are the GPIO address bus, according to whether it is configured for 128 or 32 bytes. The detailed decoding of each address bit is shown in Table B11-5.

Table B11-5. GPIO Address Generation

Configuration	GPA6	GPA5	GPA4	GPA3	GPA2	GPA1	GPA0
32-bytes	–	–	–	PCI AD[3]	PCI AD[2]	PCI BE[3:0]	
128-bytes	PCI AD[6]	PCI AD[5]	PCI AD[4]	PCI AD[3]	PCI AD[2]	–	–

4. CYCLE TIMING CONTROL

Bits in register GR19 control the timing of the I/O cycles on the GPIO port. There are two basic cycle types, one based on the ISA-bus IORD#/IOWR# cycle, the other based on the ISA bus DMA cycle. Table B11-6 summarizes these two cycles.

Table B11-6. Cycle Termination

GR19[1]	Cycle Type	Data Valid for write	Data Sampled on Read	TRDY# Returned
0	IORD/IOWR	Prior to IOWR#	Trailing edge of IORD#	Next CLK after RDY is sampled true
1	DTACK	Prior to IOWR#	Leading edge of DTACK#	Next CLK after DTACK#

Table B11-7 summarizes the meaning of the timing control bits in register GR19.

Table B11-7. Timing Control Bits in GR19

GR19[x]	Use	'0'	'1'	Symbol in Figures
5	Minimum I/O strobe timing	2 CLK	4 CLK	t_3
4	CS# to Command (IORD#/IOWR#)	1 CLK	2 CLK	t_2
3	Command to RDY sampled	1 CLK	2 CLK	t_4
2	Address valid to CS# delay	1 CLK	2 CLK	t_1

Table B11-7. Timing Control Bits in GR19 *(cont.)*

GR19[x]	Use	'0'	'1'	Symbol in Figures
1	Cycle termination	RDY	DTACK#	
0	Data port width	8 bits	16 bits	

Figure B11-1 shows a cycle when the CL-GD5446 is configured to terminate a cycle with GPRDY. If GPRDY is sampled true at the end of t_4, the strobe width is controlled by t_3. If GPRDY is sampled false at the end of t_4, it controls the cycle width. For a write cycle, the data is valid before GPIOWR# goes active. for a read cycle, the data is sampled in the CL-GD5446 with the trailing edge of IORD#. TRDY# is forced active on the next CLK after GPRDY is sampled true.

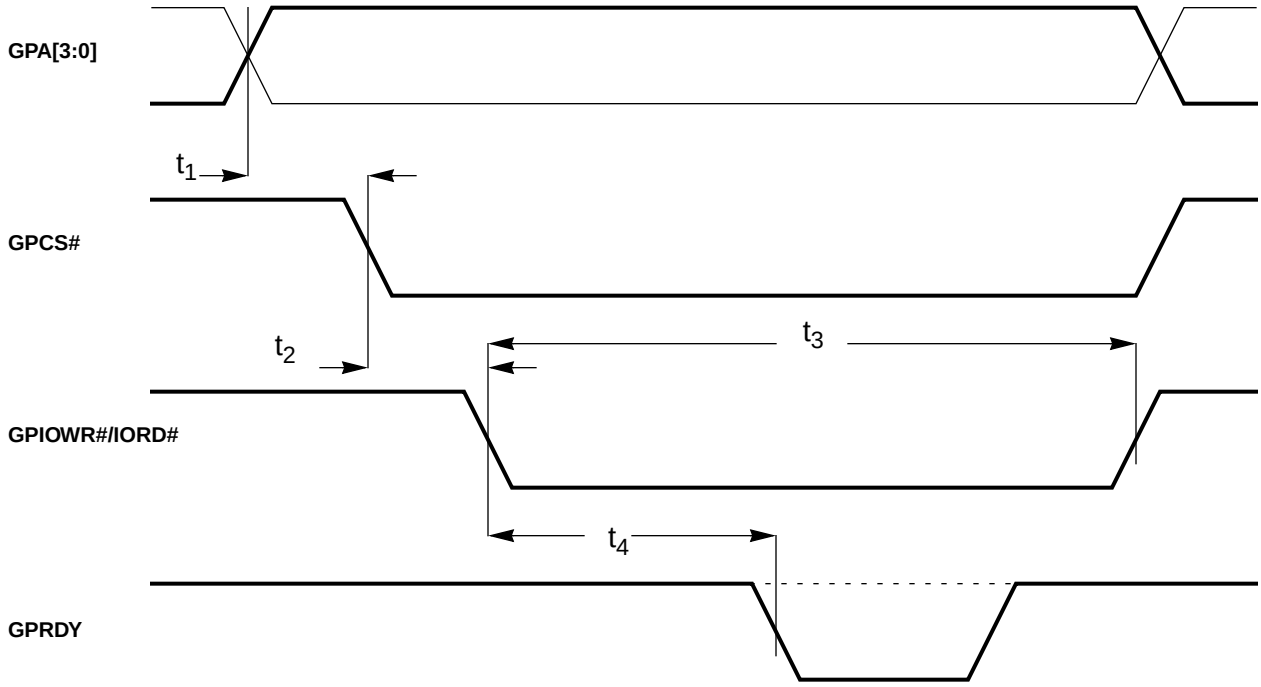


Figure B11-1. GPIO Timing: RDY Termination

Figure B11-2 shows a cycle when the CL-GD5446 is configured to terminate a cycle with GPDT. For a write cycle, the data is valid before GPIOWR# goes active. For a read cycle, the data is sampled in the CL-GD5446 with the leading edge of GPDT. TRDY# is forced active on the next CLK after GPDT.

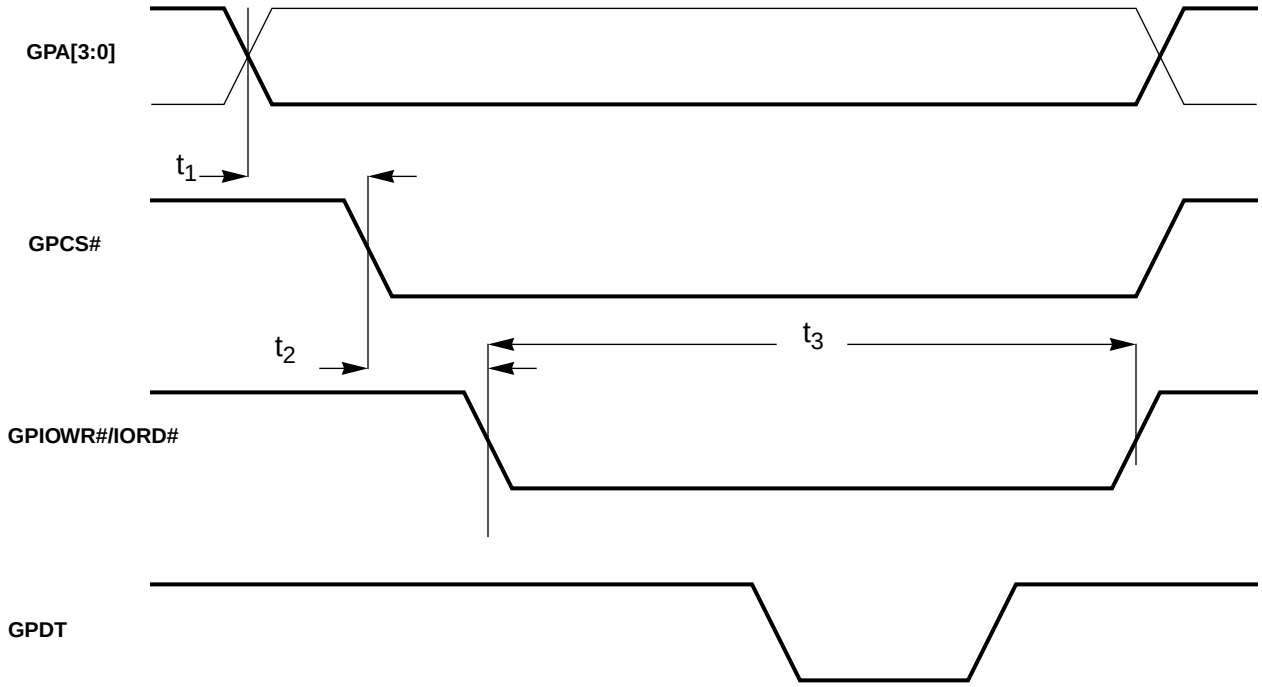


Figure B11-2. GPIO Timing: GPDT Termination

5. GPIO PROGRAMMING EXAMPLE

The following assembly language program example shows how GPIO is initialized for operation. This assumes that GPIO is configured in the I/O space (as opposed to memory space).

```
.MODEL small
.STACK 100h
.DATA
;IO_OFFSET will contain the GPIO base address
             IO_OFFSET DW 0000
.486p
.CODE

MAIN         proc far
             mov         dx,@DATA
             mov         ds,dx
             call        FIND46                              ;;find 5446
device
             jc          ERROR       ;no 5446
             call        READCFG14   ;read GPIO offset
             jc          ERROR       ;can't read PCI config 14
             cmp         cx,00000000h;see if GPIO is activated (CF8)
             je          ERROR       ;no GPIO
             and         cx,0FFFCh   ;drop low order two bits and
                                     ;keep 16-bit I/O address
             mov         [IO_OFFSET], cx
             call        SET_ISA     ;set GPIO to 8 bit ISA mode
             mov         dx,[IO_OFFSET];get GPIO base address
             mov         al,55h      ;typical data pattern
             out         dx,al       ;and write to GPIO port
ERROR:
             call        TODOS       ;return to DOS
endp         MAIN

;*************************************************************
;Find 5446 in PCI system. Carry will be set if unsuccessful.
;Bus number in BH and device and function number in BL are passed
;on to READCFG14
FIND46       proc        near
             mov         ax,0B102h   ;call to find PCI device
             mov         cx,00B8h    ;46 device ID
             mov         dx,1013h    ;Cirrus Logic Vendor ID
```

```
                mov         si,0000h    ;device index is 0000h
                int         1Ah
                ret                     ;return w/ carry set or not
endp            FIND46

;*************************************************************
;Read base registers for I/O accesses. Carry will be set if call
;is unsuccessful. ECX will contain the value from the PCI Config
;register 14. This is the base address for controlling TV out. Bus
;number, device and function number are passed on from the
;FIND46 call.
READCFG14       proc        near
                mov         ax,0B10Ah   ;call read config DWORD
                mov         di,0014h    ;configuration reg 14h
                int         1Ah
                ret
endp            READCFG14

;*************************************************************
;program GPIO for ISA-style controls and termination with
;minimum cycle length
SET_ISA         proc        near
                mov         dx,03CEh    ;point to GR registers
                mov         ah,00h      ;all timings are minimum
                mov         al,19h      ;GR19
                out         dx,ax       ;write it
                ret
endp            SET_ISA

;*************************************************************
;exit to dos
TODOS           proc        near
                mov         ah,4Ch
                int         21h         ;exit to dos
                ret                     ;avoid assembler warning
endp            TODOS

END
```


Appendix C1

Software Support

SOFTWARE SUPPORT

1. INTRODUCTION

The CL-GD5446 VGA controller is available with the CL-GD5446 VGA BIOS and an extensive set of utilities and software drivers. The following sections briefly describe some of the utilities and software drivers available.

To get an up-to-date list of BIOS, utilities, and software drivers supported, refer to the latest release of CL-GD5446 VGA BIOS and utilities or the CL-GD5446 Display Drivers Release Kits. These programs are also available on the Cirrus Logic BBS, through ftp, and on the Cirrus Logic web site (see Appendix D1, “Cirrus Logic Bulletin Board Service, FTP, and WWW”).

2. CL-GD5446 VGA SOFTWARE UTILITIES

This section describes the software utilities provided with the CL-GD5446 VGA controllers, and explains the function and usage of each.

2.1 CLMODE — A Display Mode Configuration Utility

The CL-GD5446 VGA controllers have many more display modes than the original IBM VGA. To take advantage of high-resolution modes, a compatible monitor must be installed and the proper monitor parameters selected.

The CLMODE utility provides a number of graphics configuration options, all of which are selectable from a menu or by direct keyboard input at the DOS command line. The menu options include:

- Monitor-type selection based on monitor vertical and horizontal sync frequencies.
- Selection and setting of VGA BIOS Standard and Extended display modes.
- Extended display mode preview to verify high-resolution modes supported by the display subsystem.
- Selection and setting of display refresh rates for each individual display mode resolution (that is, 640×480, 800×600, 1024×768, and 1280×1024) to match display refresh rates supported by a monitor.

2.2 VGA.EXE — RAMBIOS Utility

This VGA.EXE utility is an executable version of the VGA BIOS EPROM code that can be loaded into DOS memory. VGA.EXE utility allows the VGA BIOS to be executed out of 16- or 32-bit system memory, instead of an 8- or 16-bit bus ROM. This permits operations that use the VGA BIOS to run much faster.

The most noticeable performance boost is in text modes where VGA BIOS character-write and text-scrolling functions are used. Common benchmark programs that can demonstrate this increased performance are: PC Bench™ from Ziff-Davis® Publishing Company, and QA Plus™ from Diagsoft®.

However, most GUI applications do not heavily use the BIOS, so actual performance increases vary between applications, and in some cases, may not be significant.

Many of the ’386 and later PCs automatically cache (copy into system memory) the VGA BIOS on boot up. This is sometimes called ‘Shadow RAM’. Here, the BIOS is already executing out of 16- or 32-bit memory, and the VGA.EXE utility is unnecessary.

The VGA.EXE utility can be executed from the command line or named in the AUTOEXEC.BAT. The utility then automatically installs in system memory at the correct address, shortly after power-on or a warm-boot.

The RAMBIOS utility only works with MS-DOS or PC-DOS, but does not work with OS/2®, Unix®, or Xenix®.

2.3 OEMSI (OEM System Integration) Utility

The OEMSI utility allows the Cirrus Logic VGA BIOS to be customized to system requirements. Since OEMSI operates upon the binary (executable) image of the BIOS, source code is not necessary for customization. Several different derivative BIOSes can be easily generated from the same core, reducing maintenance problems and simplifying software-generation control. If OEMSI utility is used, do not ship the RAMBIOS since using it overrides the changes.

A wide range of parameters, default values, and tables can be easily modified or replaced using the OEMSI program. The following is a list of components of the Cirrus Logic VGA BIOS that can be modified with the OEMSI program.

Sign-on Message	In addition to the Cirrus Logic copyright notices displayed on system boot-up, custom copyright messages can be inserted. The positioning of the cursor, after the copyright messages have been displayed, can also be changed.
Monitor-type Configuration	The mechanism for how monitor type is determined is selected by reading software-configuration switches, or by a software INT 15H call.
Hardware Configuration Registers	The CL-GD5446 VGA BIOS hardware configuration table includes register values programmed at POST. This allows customizing of register values that program display dot clocks, memory clocks, and other programmable settings.
Display Mode Parameter Tables	These tables contain the complete set of register values for each Standard and Extended display mode. Values for both the Standard VGA registers and Cirrus Logic Extension registers are contained in these tables, and can be customized to configure display refresh rates for individual display mode resolutions.
Font Tables	All fonts used by the Cirrus Logic VGA BIOS can be modified or replaced by using the OEMSI utility. A flexible scheme is implemented, whereby font tables can be exported from the binary image of the BIOS or imported to it.

2.4 WINMODE Utility

The WINMODE utility is a Windows application that conveniently sets the resolution and number of colors of the display.

3. CL-GD5446 VGA SOFTWARE DRIVERS

Several text and graphics device drivers are available to enhance the operation of VGA graphics applications. These drivers are discussed in the following sections.

3.1 Driver Applicability

The CL-GD5446 VGA controller needs no software drivers to run applications in Standard VGA modes. The drivers listed in Table C1-1 are provided as a service to the user for improved resolution and performance of many software packages.

Cirrus Logic recognizes that quality device drivers are an important feature of any display subsystem and as such, our list of available device drivers is continuously expanding. For the latest list of available device drivers, please refer to the CL-GD5446 Software Drivers and Utilities Kit.

Table C1-1. Software Driver Support

Software Drivers	Resolution Supported[a]	No. of Colors
Microsoft® / Intel® DCI (display control interface), DirectDraw™, VPM™ Provider	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
Microsoft® Windows® v3.x Microsoft® Windows® 95	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
Microsoft® Windows NT™ v3.5, v3.51, v4.0	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	16 and 256
	640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
OS/2® v2.11, v3.0	640 × 480, 800 × 600, 1024 × 768,1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768, 1280 × 1024	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million
AutoCAD® v12.0, v13.0 Autoshade® v2.0 with Renderman 3D Studio™ v1.0, v2.0, v3.0, v4.0	640 × 480, 800 × 600, 1024 × 768, 1280 × 1024	16
	640 × 480, 800 × 600, 1024 × 768, 1280 × 1024	256
	640 × 480, 800 × 600, 1024 × 768	32,768
	640 × 480, 800 × 600, 1024 × 768	65,536
	640 × 480, 800 × 600, 1024 × 768	16.8 million

[a] All monitor types do not support all resolutions; 640 × 480 drivers will run on PS/2®-type monitors. Extended resolutions are dependent upon monitor type and VGA system implementation.

Appendix C2

VGA BIOS

VGA BIOS

1. BIOS OVERVIEW

The CL-GD5446 VGA BIOS is a high-performance firmware product optimized to take full advantage of the CL-GD5446 VGA controller. The CL-GD5446 BIOS is based on proven BIOS technology, and is fully compatible with the IBM VGA BIOS Interrupt 10h interface. The BIOS is designed to provide a well-defined interface between MS-DOS, application software, and special OEM utility programs. In addition, it provides an extended set of functions to support the CL-GD5446 VGA controller. This document discusses the standard VGA BIOS functions. The extended BIOS functions are discussed in Appendix C3, “BIOS Extensions”.

1.1 Main BIOS Features

The CL-GD5446 VGA BIOS supports the following key features:

- Fully IBM VGA-compatible BIOS
- Support for high-resolution, extended 256-color, Direct-Color™ and 16.8-million color, True-color Display modes
- High-performance operation
- Modular, proven design
- Adapter or system board implementation (C000/E000 segments)
- Can be integrated with system BIOS
- Supports switchless configuration
- Can be customized without source code
- VESA-compatible modes and interface

1.2 Extended Display Mode Support

The CL-GD5446 VGA BIOS provides full support for all extended high-resolution display modes through INT 10h function calls. In addition, the CL-GD5446 VGA BIOS supports a variety of extended functions, such as VGA display configuration and extended VGA inquiry. For a detailed description of these functions, refer to Appendix C3.

1.3 Direct-Color Operation

The CL-GD5446 BIOS supports Direct-color and True-color display modes. These modes allow CL-GD5446 to display 32K, 64K, or 16.8 million colors at resolutions of up to 1024 $\times$ 768.

1.4 High Performance

The BIOS is optimized to provide maximum performance in adapter or motherboard implementations. The CL-GD5446 local bus, display memory interface, memory clock, and dot clock configurations are configurable using the VGA BIOS.

In addition, time-critical routines (such as TTY output and scroll) are designed to provide maximum throughput in both text and graphics modes.

1.5 System Integration

The CL-GD5446 VGA BIOS can be easily integrated for an adapter or motherboard design. The BIOS, 32 Kbytes in size, is provided for both the C000 and E000 address segments. To save space on the system board, the CL-GD5446 VGA BIOS can be incorporated into the system BIOS ROM at either C000 or E000 addresses.

The BIOS does not require DIP switches or external hardware for configuration. A well-defined interface to the CL-GD5446 BIOS configuration is available for system BIOS or OEM set-up routines.

1.6 Customization

The default CL-GD5446 BIOS is designed to be implemented without modification in almost all environments. However, the CL-GD5446 BIOS can also be easily customized for a specific system environment. Modifications can also be accomplished with the Cirrus Logic OEMSI utility program; such modifications do not require the CL-GD5446 VGA BIOS source code. Hundreds of BIOS parameters and features can be modified, including:

- Sign-on message
- Display type configuration
- Display mode parameter tables
- Font tables

1.7 Compatibility

The CL-GD5446 BIOS is 100% compatible with the IBM VGA BIOS and supports BIOS-level compatibility for adapter card or integrated VGA on the system board. In addition, the CL-GD5446 BIOS fully complies with the display modes and specifications issued by VESA.

2. CL-GD5446 VGA BIOS INITIALIZATION

The CL-GD5446 VGA BIOS is shipped in two formats: a segment C000-based VGA BIOS configured for VGA adapter cards plugged into PC systems, and segment E000-based VGA BIOS configured for integrated VGA display subsystems on the motherboard. The following procedure is performed by the VGA BIOS at power-up initialization.

For Segment C000 Adapter-Based VGA BIOS

1) Checks if VGA BIOS vector INT 10h is already initialized.

2) If so, calls the INT 10h function to disable the existing VGA card by placing it in Sleep mode.

3) Disables the VGA adapter by writing a 10h value to I/O Port 46E8h.

4) Programs I/O Port 102h with data 01h to enable display subsystem.

5) Writes a value of 08h to 46E8h to enable I/O and memory addressing.

6) Writes I/O Port 4AE8h to disable 8514/A.

7) Disables VGA display (by programming Sequencer Clocking Mode register, SR1[5] = 1).

8) Initializes video vectors INT 10h and INT 42h.

9) Enables extensions by writing register SR06 (I/O Port 3C4, Index = 06) with data 012h.

10) Initializes CL-GD5446 Extension registers.

11) Checks for co-resident MDA display adapter. Initializes co-resident bits and sets up MDA adapter if MDA is present.

12) Checks for CGA. If CGA is present, initializes co-resident bits, sets VGA to monochrome, and enables CGA.

13) Runs display memory tests.

14) Initializes Text mode 3.

15) Displays sign-on message.

16) Prints error messages if any POST error flags are set.

For Segment E000 Motherboard-Based VGA BIOS

1) Enables VGA Setup mode by writing VGA Display Sleep Enable register (VSE I/O Port 46E8h with value of 10h).

2) Programs POS 2 register (I/O Port 102h with data 01h) to enable display subsystem.

3) Enables motherboard VGA by programming VSE register 46E8 with data value 1.

4) Writes I/O Port 4AE8h to disable 8514/A.

5) Disables VGA display (by programming Sequencer Clocking Mode register, SR1[5] = 1).

6) Initializes Video Vectors INT 10h and INT 42h.

7) Enables extensions by writing SR06 (I/O Port 3C4, Index = 06h) with data 012h.

8) Initializes CL-GD5446 Extension registers.

9) Checks for co-resident MDA display adapter. If MDA is present, initializes co-resident bits and sets up MDA adapter.

10) Checks for CGA. If CGA is present, initializes co-resident bits, and sets VGA to monochrome and enables CGA.

11) Runs display memory tests.

12) Initializes Text mode 3.

13) Displays sign-on message.

14) Prints error messages if any POST error flags are set.

15) Checks to see if a VGA adapter is also present in the system. If so, disables motherboard VGA controller.

3. VIDEO BIOS INTERRUPT VECTORS

The interrupt vectors that must be initialized by DOS (including the planar and video BIOSes) are listed in Table C2-6 on page C2-56. Of these, the vectors (at locations 0:0040, 0000:007C, 0000:0108, 0000:010C) corresponding to vectors 10, 1F, 42, 43 are managed by the Display BIOS.

10h — Display Services (Vector Location = 0000:0040h)

The CL-GD5446 BIOS functions are accessed using INT 10h. Application programs place a function code in Ah and if required, in other registers calling parameters, then executes an INT 10h instruction. When the BIOS gains control, the appropriate code is executed to perform the function; parameter values may be left in processor registers to be returned to the calling program upon exit from the interrupt routine.

The functions supported by the CL-GD5446 BIOS allow the calling program to

- Set the current mode
- Manipulate the cursor
- Place characters and individual pixels on the display
- Scroll the screen
- Load character fonts and color palette values
- Read the light-pen position.

These functions are described in the following sections.

1Dh — 6845 Initialization (Vector Location = 0000:0074h)

This vector points to the parameter tables that set up the 6845.

1Fh — CGA Character Set (Vector Location = 0000:007Ch)

This pointer is for the table of the upper 128 characters in CGA modes 4, 5, and 6. The INT 43h vector is for the lower 128 characters of these modes.

42h — Old Video Services Pointer (Vector Location = 0000:0108h)

This location was the INT 10h vector for planar BIOS video services. When the EGA/VGA is installed, BIOS routines reload this address with a pointer to the planar INT 10h video service routine entry point.

43h — Graphics Character Table (Vector Location = 0000:010Ch)

BIOS routines use this vector to point to a table of bitmaps used when graphics characters are displayed. This table is for the lower 128 characters in Display modes 4, 5, and 6. This table is also for 256 characters in all additional graphics modes (both IBM standard and Cirrus Logic extensions).

INT 10h calls constitute the bulk of the services provided by the display BIOS and are described in detail in Section 4. They are listed along with the function and subfunction that define the particular service required. Note that some INT 10h services were introduced with the VGA and are not available on the earlier EGA. The services have been divided up into functional groupings.

4. INTERRUPT 10h: INDEX

Table C2-1. BIOS Routines Index

Function	Subfunction	Name	Page
00h		Set Display Mode	C2-8
01h		Set Cursor Type	C2-9
02h		Set Cursor Position	C2-10
03h		Get Cursor Position	C2-10
04h		Get Light Pen Position	C2-11
05h		Select Active Display Page	C2-11
06h		Window Scroll Up	C2-11
07h		Window Scroll Down	C2-12
08h		Read Character/Attribute at Cursor Position	C2-12
09h		Write Character/Attribute at Cursor Position	C2-13
0Ah		Write Character at Cursor Position	C2-14
0Bh	00h	Set Background/Border Color	C2-14
0Bh	01h	Select the Palette Set	C2-15
0Ch		Write Dot (Pixel)	C2-15
0Dh		Read Dot (Pixel)	C2-15
0Eh		Write Character to Active RAM in Teletype mode	C2-16
0Fh		Get Display State	C2-16
10h	00h	Set Individual Palette Register	C2-16
10h	01h	Set Overscan (Border) Register	C2-17
10h	02h	Set All Palette Registers and Overscan Register	C2-17
10h	03h	Toggle Intensity/Blinking	C2-17
10h	07h	Read Individual Palette Register	C2-18
10h	08h	Read Overscan (Border) Register	C2-18
10h	09h	Read All Palette Registers and Overscan Register	C2-18
10h	10h	Set Individual Color Register	C2-18
10h	12h	Set Block of Color Registers	C2-19
10h	13h	Select Color Page	C2-19
10h	15h	Read Individual Color Register	C2-20

Table C2-1. BIOS Routines Index *(cont.)*

Function	Subfunction	Name	Page
10h	17h	Read Block of Color Registers	C2-20
10h	1Ah	Read Current State of Color Page	C2-21
10h	1Bh	Sum Color Values to Gray Shades	C2-21
11h	00h	Load User Text Font	C2-21
11h	01h	Load 8 × 14 ROM Text Font	C2-22
11h	02h	Load 8 × 8 ROM Text Font	C2-23
11h	03h	Select Block Specifier	C2-24
11h	04h	Load 8 × 16 ROM Text Font	C2-25
11h	10h	Load User Text Font and Reprogram Controller	C2-25
11h	11h	Load 8 × 14 ROM Text Font and Reprogram Controller	C2-27
11h	12h	Load 8 × 8 ROM Text Font and Reprogram Controller	C2-28
11h	14h	Load 8 × 16 ROM Text Font and Reprogram Controller	C2-29
11h	20h	Set Pointer of User's Graphics Font Table to Interrupt 1Fh	C2-30
11h	21h	Set Pointer of User's Graphics Font Table to Interrupt 43h	C2-31
11h	22h	Set Pointer of 8 × 14 ROM Graphics Font Table to Interrupt 43h	C2-31
11h	23h	Set Pointer of 8 × 8 ROM Graphics Font Table to Interrupt 43h	C2-32
11h	24h	Set Pointer of 8 × 16 ROM Graphics Font Table to Interrupt 43h	C2-32
11h	30h	Get Font Information	C2-33
12h	10h	Get Configuration Information	C2-33
12h	20h	Select Alternate PrintScreen Routine	C2-34
12h	30h	Select Scanlines (Alphanumeric Mode)	C2-34
12h	31h	Enable/Disable Default Palette Loading	C2-35
12h	32h	Enable/Disable display	C2-35
12h	33h	Enable/Disable Grayscale Summing	C2-35
12h	34h	Enable/Disable Cursor Emulation	C2-36
12h	35h	Switch Active Display	C2-36
12h	36h	Enable/Disable Screen Refresh	C2-37
13h		Write String in Teletype	C2-37
1Ah	00h	Get Display Combination Code	C2-38
1Ah	01h	Set Display Combination Code	C2-38

Table C2-1. BIOS Routines Index *(cont.)*

Function	Subfunction	Name	Page
1Bh		Get Functionality/State Information	C2-39
1Ch	00h	Get Buffer Size for Display State	C2-42
1Ch	01h	Save Display State	C2-42
1Ch	02h	Restore Display State	C2-43

5. DESCRIPTION OF FUNCTIONS

Each function and subfunction in the standard VGA BIOS is described in the following sections. These sections are ordered by function, then subfunction number.

5.1 Function 00h: Set Display Mode

[Entry]

AH = 00h
AL = Display Mode (see below)

[Return]

NONE

[Note]

1) Display mode table for standard VGA:

Mode	Resolution	Type	Colors	Pages
00h/01h	40 × 25 (360 × 400)	Text	16	8
02h/03h	80 × 25 (640 × 400)	Text	16	8
04h/05h	320 × 200 (40 × 25)	Graphics	4	1
06h	640 × 200) (80 × 25)	Graphics	2	1
07h	80 × 25 (720 × 400)	Text	Monochrome	8
08h - 0Ch	Reserved			
0Dh	320 × 200 (40 × 25)	Graphics	16	8
0Eh	640 × 200 (80 × 25)	Graphics	16	4
0Fh	640 × 350 (80 × 25)	Graphics	Monochrome	2
10h	640 × 350 (80 × 25)	Graphics	16	2
11h	640 × 480 (80 × 25)	Graphics	2	1

12h	640 × 480 (80 × 25)	Graphics	16	1
13h	320 × 200 (40 × 25)	Graphics	256	1

2) If bit 7 of AL is set, the display buffer is not cleared. Otherwise, the display buffer clears during mode setting (EGA, VGA only) (Clear Screen).

3) No hardware cursor in graphics modes.

4) Default mode during POST: mode 3h = Color monitor, mode 07h = Monochrome monitor.

5) There is no difference between modes 00h and 01h, 02h and 03h, or 05h and 06h on EGA/VGA. They are only different on CGA, which supports composite-display displays.

6) The default settings of each display mode can be overridden by several subfunctions in Function 12h or by supply user's display service table, whose address is stored in BIOS data area 0040:A8h.

7) See Appendix C3, "BIOS Extensions", for extended modes.

5.2 Function 01h: Set Cursor Type

[Entry]

AH = 01h
CH = Start scanline of cursor (0 base)
CL = End scanline of cursor (0 base)

[Return]

NONE

[Notes]

1) This function is available in text modes only. The values of cursor type are stored at [40:60].

2) The definition of value in register CH:

Bit	Definition
7:6	Reserved = 0
5	1 = No cursor display 0 = Normal blinking cursor
4:0	Start scanline (0 base)

3) The definition of value in register CL:

Bit	Definition
7	Reserved = 0
6:5	Number of character skew
4:0	End scanline (0 base)

4) Default setting:

Font Size	Start	End
8 × 8	6	7
8 × 14	11	12
8 × 16	13	14

5) Turn off cursor emulation to allow cursor displaying as the values set in the function call. The Cursor Emulation Flag is located in bit 0 of [40:87]. It can be turned on/off by Subfunction 34h of Function 12h call.

6) If Cursor Emulation is turned off, the values passed in CH and CL are programmed literally into the hardware.

5.3 Function 02h: Set Cursor Position

[Entry]

AH = 02h
BH = Display page (0 base)
DH = Row number of cursor location start (0 base)
DL = Column number of cursor location end (0 base)

[Return]

NONE

[Notes]

1) This function is available for both text and graphics modes.

2) If register DL is specified over the width of screen displayable area, the cursor wraps to the next row. If register DH is specified over the height of screen displayable area, the cursor disappears.

3) Default setting for each mode: Cursor location at 0000h.

4) BIOS maintains one cursor location for each page and supports up to eight pages. These values are recorded at [40:50], and occupy eight words (one word for each location).

5.4 Function 03h: Get Cursor Position

[Entry]

AH = 03h
BH = Display page (0 base)

[Return]

CH = Start scanline of cursor (0 base)
CL = End scanline of cursor (0 base)
DH = Row number of cursor start location (0 base)
DL = Column number of cursor end location (0 base)

[Note]

1) The cursor type is same for all pages. The cursor location of each page is separately maintained.

5.5 Function 04h: Get Light Pen Position (no longer supported)

[Entry]

AH = 04h

[Return]

AH = 00h	Light Pen inactive
or	
AH = 01h	Light Pen active and returns following values
BX = Pixel column	(X coordinate in graphics modes (0 base)
CX = Pixel row	(Y coordinate in graphics modes above Mode 06h (0 base)
CH = Pixel row	(Y coordinate in Graphics Modes 04h – 06h (0 base)
DH = Character row	(Y coordinate in text modes (0 base)
DL = Character column	(X coordinate in text modes (0 base)

[Notes]

1) The color of background and foreground affects the sensitivity of light pen.

2) High-resolution device affects the accuracy of light pen.

5.6 Function 05h: Select Active Display Page

[Entry]

AH = 05h
AL = Display page (0 base)

[Return]

None

[Notes]

1) The contents of each page is not altered by changing to other pages.

2) Please refer to the display mode table of Function 00h.

5.7 Function 06h: Window Scroll Up

[Entry]

AH = 06h
AL = Number of rows to be scrolled up (0 = scroll up and clear entire window)
BH = Attribute to be used in inserting blank lines
CH = Y coordinate of top left corner of window (0 base)
CL = X coordinate of top left corner of window (0 base)
DH = Y coordinate of bottom right corner of window (0 base)
DL = X coordinate of bottom right corner of window (0 base)

[Return]

None

[Notes]

1) This function clears the entire window when it encounters the number of rows of window equal to the value in register AL or AL = 0.

2) The image outside the window is unchanged. The cursor is not updated.

3) A new blank line with attribute value specified in BH is inserted from the bottom of the window whenever an old line at the top of the window is scrolled out of the window.

4) This function is available for both Text and Graphics modes.

5.8 Function 07h: Window Scroll Down

[Entry]

AH = 07h
AL = Number of rows to be scrolled down (0 = scroll down and clear entire window)
BH = Attribute to be used in inserting blank lines
CH = Y coordinate of top left corner of window (0 base)
CL = X coordinate of top left corner of window (0 base)
DH = Y coordinate of bottom right corner of window (0 base)
DL = X coordinate of bottom right corner of window (0 base)

[Return]

None

[Notes]

1) This function clears the entire window when it encounters the number of rows of window equal to the value in register AL or AL = 0.

2) The image outside the window is unchanged. The cursor is not updated.

3) A new blank line with attribute value specified in BH is inserted from the top of the window whenever an old line at the bottom of the window is scrolled out of the window.

4) This function is available for both Text and Graphics modes.

5.9 Function 08h: Read Character/Attribute at Cursor Position

[Entry]

AH = 08h
BH = Display page (0 base)

[Return]

AH = Attribute (Valid on text modes)
AL = ASCII character code

[Notes]

1) This function can read data from other valid inactive pages in multiple page modes at any time.

2) The cursor is not updated after reading a character from the screen, and must be explicitly moved.

3) No control characters (such as, LF, CR, BACKSPACE, and BELL) are recognized.

4) In Graphics modes 04h–06h of the CGA adapter, the first half of the character font (Code 00h–7Fh) is only maintained in system ROM. To support the second half of the character font (Code

80h–FFh), the Interrupt Vector 1Fh at 0000:007Ch, must be initialized to point to the second half of the character font.

5) Graphics modes return the Character Code only. Three characters, 00h/20h/FFh, cannot be distinguished, and the function always reads them back as Character Code 00h.

6) The character codes are read back as Character Code 00h when they are written with a color the same as the background color in graphics modes.

5.10 Function 09h: Write Character/Attribute at Cursor Position

[Entry]

AH = 09h
AL = ASCII character code
BH = Display page (0 base)

or

BL = Attribute (text modes)
Display color (graphics modes)
CX = Repeat character count

[Return]

None

[Notes]

1) This function can write data to other valid inactive pages in multiple page modes at any time.

2) The cursor is not updated after writing a character to the screen, and must be explicitly moved.

3) No control characters (such as, LF, CR, BACKSPACE, and BELL) are recognized.

4) Graphics modes 04h–06h of CGA adapter, the first half of the character font (Code 00h–7Fh) is only maintained in system ROM. To support the second half of the character font (Code 80h–FFh), the Interrupt Vector 1Fh at 0000:007Ch, must be initialized to point to the second half of the character font.

5) In Graphics modes, the color (attribute) is treated as pixel color to generate an ASCII character pattern. The color value is masked according to the number of colors in the display modes.

6) The character codes are displayed as blank when they are written with the color the same as the background color in graphics modes.

7) The characters written to the screen, specified in CX, should not extend to the next row in graphics modes or invalid results are generated.

8) If bit 7 of register BL is set, the function takes the color value XOR'ed with the value in display memory (valid in all graphics modes except mode 13h). This feature can be used in fast character/dot erasing.

5.11 Function 0Ah: Write Character at Cursor Position

[Entry]

AH = 0Ah
AL = ASCII character code
BH = Display page (0 base)

or

BL = Foreground color (graphics modes only)
CX = Repeat character count

[Return]

None

[Notes]

1) This function can write data to other valid inactive pages in multiple page modes at any time.

2) The cursor is not updated after writing a character to screen and must be explicitly moved.

3) No control characters (such as, LF, CR, BACKSPACE, and BELL) are recognized.

4) Graphics modes 04h–06h of CGA adapter, the first half of the character font (Code 00h–7Fh) is only maintained in system ROM. To support the second half of the character font (Code 80h–FFh), the Interrupt Vector 1Fh at 0000:007Ch, must be initialized to point to the second font.

5) In Graphics modes, the color (attribute) is treated as pixel color to generate AN ASCII character pattern. The color value is masked according to the number of colors in the display modes.

6) The character codes are displayed as blank when they are written with the same color as the background color in Graphics modes.

7) The characters written to the screen, as specified in CX, should not extend to next row in Graphics modes or invalid results are generated.

8) If bit 7 of register BL is set, the function takes the color value XOR'ed with the value in display memory (valid in all Graphics modes except mode 13h). This feature can be used in fast character/dot erasing.

5.12 Function 0Bh, Subfunction 00h: Set Background/Border Color

[Entry]

AH = 0Bh
BH = 00h
BL = Color Value (0 - 31: Low-Intensity Colors = 0–15, High-Intensity Colors = 16–31)
– Border Color for text modes (modes 00h–03h)
– Color for 640 × 200 Graphics mode (mode 06h)

[Return]

None

[Note]

1) There are several functions in Function 10h that allow extensive display-colors control for both Text and Graphics modes.

5.13 Function 0Bh, Subfunction 01h: Select Palette Set

[Entry]

AH = 0Bh
BH = 01h (Valid on Modes 04h and 05h 320 × 200 only)
BL = 0 – palette set: Background, Green, Red, Brown
1 – palette set: Background, Cyan, Magenta, White

[Return]

None

[Note]

1) For the CGA adapter, the palette set is defined as follows:

Mode	BL	Palette Set
04h	00h	Background, Green, Red, Yellow
	01h	Background, Cyan, Violet, White
05h	00/01h	Background, Cyan, Red, White

5.14 Function 0Ch: Write Dot (Pixel)

[Entry]

AH = 0Ch
AL = Color value for pixel (Bit 7 is XOR flag)
BH = Display page (0 base)
CX = X coordinate, column number (0 base)
DX = Y coordinate, row number (0 base)

[Return]

None

[Notes]

1) For the coordinate range, please refer to the resolution field of display mode table in Function 00h.

2) If bit 7 of register AL is set, the requesting color value is XOR'ed with memory color value.

5.15 Function 0Dh: Read Dot (Pixel)

[Entry]

AH = 0Dh
BH = Display page (0 base)
CX = X coordinate, column number (0 base)
DX = Y coordinate, row number (0 base)

[Return]

AL = Dot (Pixel) color

[Note]

1) For the coordinate range, please refer to the resolution field of display mode table in Function 00h.

5.16 Function 0Eh: Write Character to Active RAM in Teletype Mode

[Entry]

AH = 0Eh
AL = ASCII character
BL = Foreground color in graphics modes

[Return]

None

[Notes]

1) Control characters (such as LF, CR, Backspace, and BELL) are recognized. (ASCII codes: LF = 0Ah, CR = 0Dh, Backspace = 08h, BELL = 07h).

2) Line wrapping and screen scrolling are supported.

3) Cursor moves to next position after writing a character to the screen.

4) PC BIOS version 10/19/81 or earlier; register BH must be set to '0'.

5) If bit 7 of register BL is set in Graphics modes, the color value in the register is XOR'ed with the contents of display memory.

6) In Text modes, the attribute of a character written to a new line is taken from the attribute of the last character in previous line. To control the attribute for a character, use Function 09h with blank character/attribute first before the function issued.

5.17 Function 0Fh: Get Display State

[Entry]

AH = 0Fh

[Return]

AH = Number of displayable columns (1 base)
AL = Current display mode
BH = Current active page (0 base)

5.18 Function 10h, Subfunction 00h: Set Individual Palette Register (Internal Palette Register)

[Entry]

AH = 10h
AL = 00h (Subfunction)
BH = Color value
BL = Palette register (0–0Fh)

[Return]

None

[Notes]

1) Color value in the Internal Palette register serves as a pointer that points to one of external registers

(RAMDAC).

2) The color is not changed by this function on mode 13h.

5.19 Function 10h, Subfunction 01h: Set Overscan (Border) Register

[Entry]

AH = 10h
AL = 01h (Subfunction)
BH = Color value (00h–FFh)

[Return]

None

[Note]

1) The border color is driven by one of 256 external registers.

5.20 Function 10h, Subfunction 02h: Set All Palette Registers and OverScan Register

[Entry]

AH = 10h
AL = 02h (Subfunction)
ES: DX = Point to a 17-byte buffer

[Return]

None

[Notes]

1) The first 16 bytes in the buffer store the values for 16 internal Palette registers. The last byte is the value for the Overscan register.

2) The display color is not affected, except Overscan register in mode 13h.

5.21 Function 10h, Subfunction 03h: Toggle Intensify/Blinking Bit

[Entry]

AH = 10h
AL = 03h (Subfunction)
BL = 00h — Intensify
01h — Blinking

[Return]

None

[Notes]

1) Bit 7 of the Attribute Byte is interpreted according to the state set by this function. This function can provide 16 background colors (in intensify state) of 16-color Text modes.

2) This function also supports Monochrome modes (07h, 0Fh).

5.22 Function 10h, Subfunction 07h: Read Individual Palette Register (Internal Palette Register)

[Entry]

AH = 10h
AL = 07h (Subfunction)
BL = Palette register (0–0Fh)

[Return]

BH = Color value

[Note]

1) The color value in the internal Palette register serves as a pointer to one of the external registers (RAMDAC).

5.23 Function 10h, Subfunction 08h: Read OverScan (Border) Register

[Entry]

AH = 10h
AL = 08h (Subfunction)

[Return]

BH = Color value

[Note]

1) The border color is from 00h–FFh.

5.24 Function 10h, Subfunction 09h: Read All Palette Registers and OverScan Register

[Entry]

AH = 10h
AL = 09h (Subfunction)
ES: DX = Point to a 17-byte buffer
(The first 16 bytes returns values from the 16 Palette registers, respectively and the last byte is for the Overscan register).

[Return]

ES: DX = Point to the same buffer provided from the entry of function call.

5.25 Function 10h, Subfunction 10h: Set Individual Color Register (RAMDAC/External Palette Registers)

[Entry]

AH = 10h
AL = 10h (Subfunction)
BX = Color Register (00h–FFh)
DH = Red color

CH = Green color
CL = Blue color

[Return]
None

[Notes]

1) Currently, each color is a 6-bit value. All three colors, RGB, as a group form an 18-bit datum stored in the Color register.

2) The maximum displayable colors are 256 out of 256K colors (2 exponential 18).

3) In standard VGA, mode 13h uses all 256 color registers to display colors.

4) The BIOS loads default values into the Color registers whenever Function 00h (Set Display mode) is called. This is only true when the disable flag of the default palette loading is not set (refer to Subfunction 31h of Function 12h).

5) With the gray-summing flag set or a grayscale monitor attached, the BIOS weighs the three color values and the sum-to-grayshade value, then uses the value for all three colors.

5.26 Function 10h, Subfunction 12h: Set Block of Color Registers

[Entry]
AH = 10h
AL = 12h (Subfunction)
BX = Start Color register (00h–FFh)
CX = Number of color registers to set
ES: DX = Point to table of color values (each table entry is in RGB format)

[Return]
None

[Notes]

1) Currently, each color is a 6-bit value. All three colors, RGB, as a group form an 18-bit datum stored in the Color register.

2) The maximum displayable colors are 256 out of 256K colors (2 exponential 18).

3) In standard VGA, mode 13h uses all 256 color registers to display colors.

5.27 Function 10h, Subfunction 13h: Select Color Page (Not valid in Mode 13h)

[Entry]
AH = 10h
AL = 13h (Subfunction)
BL = 00h (select paging mode)
01h (select color page)
When BL = 00h
BH = 00h (select 4 pages of 64-color register page)
01h (select 16 pages of 16-color register page)
When BL = 01h
BH = Color page number (0 base)

[Return]

None

[Notes]

1) All display modes, except 256-color modes, are supported by this function.

2) This function treats the 256-color registers as sets of 16- or 64-color registers. It can quickly display different colors by switching among color sets (pages).

3) The default setting is Page 0 of 64-color Page mode after Display mode is set. Normally, Function 00h (Set Display mode) loads the default colors of the first 64-color registers (Page 00h) for all standard VGA modes, except mode 13h (248 registers loading).

5.28 Function 10h, Subfunction 15h: Read Individual Color Register (RAMDAC/External Palette Registers)

[Entry]

AH = 10h
AL = 15h (Subfunction)
BX = Color register (00h–FFh)

[Return]

DH = Red color
CH = Green color
CL = Blue color

[Notes]

1) Currently, each color is a 6-bit value. All three colors, RGB, as a group form an 18-bit datum stored in the Color register.

2) The maximum displayable colors are 256 out of 256K colors (2 exponential 18).

3) In standard VGA, mode 13h uses all 256-color registers to display colors.

4) With the gray-summing flag set, the only value returned from all three color elements of the Color register is the grayshade value.

5.29 Function 10h, Subfunction 17h: Read Block of Color Registers

[Entry]

AH = 10h
AL = 17h (Subfunction)
BX = Start Color register (00h–FFh)
CX = Number of Color registers to read
ES: DX = Point to user-provided buffer for returned color values

[Return]

ES: DX = Point to same buffer from function call entry (buffer is treated as a color table and each entry of the table consists of three bytes in RGB format).

[Notes]

1) Currently, each color is a 6-bit value. All three colors, RGB, as a group form an 18-bit datum stored in the Color register.

2) The maximum displayable colors are 256 out of 256K colors (2 exponential 18).

3) In standard VGA, mode 13h uses all 256-color registers to display colors.

4) BIOS loads the default values into the Color registers whenever Function 00h (Set Display mode) is called. This is only true when the disable flag of the default palette loading is not set (please refer to Subfunction 31h of Function 12h).

5) With the gray-summing flag set, the only value returned for all three colors is the grayshade value.

5.30 Function 10h, Subfunction 1Ah: Read Current State of Color Page (Not valid in Mode 13h)

[Entry]

AH = 10h
AL = 1Ah (Subfunction)

[Return]

BH = Current page (Value depends on Paging mode, 00h is default)
BL = Current Paging mode
(00h = 4 pages of 64-Color registers (default), 01h = 16 pages of 16-color registers)

[Notes]

1) All display modes, except 256-color modes, are supported by the function.

2) This function treats 256-color registers as sets of 16- or 64-color registers. It can be used to quickly display different colors by switching among color sets (pages).

3) Default setting is page 00h of 6-color page mode after Set Display mode.

5.31 Function 10h, Subfunction 1Bh: Sum Color Values to Grayshades

[Entry]

AH = 10h
AL = 1Bh (Subfunction)
BX = Start Color register (00h–FFh)
CX = Number of Color registers to sum

[Return]

None

[Note]

1) This function sums the Color registers desired into grayshade values regardless the gray-summing flag.

5.32 Function 11h, Subfunction 00h: Load User Text Font

[Entry]

AH = 11h
AL = 00h (Subfunction)

BH = Number of bytes per character
BL = Block to load (00h–07h)
CX = Number of characters to store
DX = ASCII character ID of the first character in the font table (ES: BP)
ES: BP = Point to user-provided font table

[Return]
None

[Notes]

1) This function is only available for Text modes. The value in register BH represents the height of each character. The value can be the specified maximum of 32-bytes-per-character in the standard VGA specification.

2) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is available for the Character Generator only. Consequently, the block value specified in register BL is in the range of 00h (default) to 07h.

3) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) The bit 3 of Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors)
if Subfunction 00h of Function 10h is called with BX = 0712h.
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors)

4) Default setting by the BIOS loads a font into Block 0, which is for both the primary and secondary font (256 displayable characters).

5) Font loading by Subfunction 00h requires caution. Since the controller is not reprogrammed, abnormal character display may occur. For example:

a) If a font table is loaded to override the font in a block, a double image may result (especially when the loaded font size is smaller than the one previously displayed). In this instance, the application should use subfunction 10h.

5.33 Function 11h, Subfunction 01h: Load 8 × 14 ROM Font

[Entry]
AH = 11h
AL = 01h (Subfunction)
BL = Block to load (00h–07h)

[Return]
None

NOTE: This is actually an 8 × 16 font.

[Notes]

1) This function is only available for Text modes.

2) The character height is 14 bytes, but the height of the display cell is same as the default setting.

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is only available for the Character Generator. Consequently, the block value specified in register BL is in the range of 00h (Default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors)
if Subfunction 00h of Function 10h is called with BX = 0712h
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors)

5) The default setting by the BIOS loads a font into Block 0 used for both the primary and secondary fonts (256 displayable character).

6) Font loading by Subfunction 00h requires caution. Since the controller is not reprogrammed, abnormal character display may occur. For example:

a) If a font table is loaded to override the font in a block, a double image may result (especially when the loaded font size is smaller than the one previously displayed). In this instance, the application should use subfunction 11h.

5.34 Function 11h, Subfunction 02h: Load 8 × 8 ROM Font

[Entry]

AH = 11h
AL = 02h (Subfunction)
BL = Block to load (00h–07h)

[Return]

None

[Notes]

1) This function is only available for Text modes.

2) The character height is eight bytes, but the height of the display cell is the same as the default setting.

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is available only for the character generator. Consequently, the block value specified in register BL is in the range of 00h (Default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two different font blocks out of eight font blocks; one for primary font, the other for secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors)
if Subfunction 00h of Function 10h is called with BX = 0712h
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors)

5) The default setting by the BIOS loads a font into Block 0 used for both the primary and secondary fonts (256 displayable characters).

6) The font loading by Subfunction 00h requires caution. Since the controller is not reprogrammed, abnormal character display may occur. For example:

a) If a font table is loaded to override the font in a block, a double image may result (especially when the loaded font size is smaller than the one previously displayed). In this instance, the application should use subfunction 12h.

5.35 Function 11h, Subfunction 03h: Select Block Specifier

[Entry]
AH = 11h
AL = 03h (Subfunction)
BL = Selection of character generator blocks

[Return]
None

[Notes]

1) The definition of the value in register BL as follows:

Bits	Font Blocks
4, 1, 0	Primary Font Block (00h–07h)
5, 3, 2	Secondary Font Block (00h–07h)

2) For EGA-compatible operation, bits 1:0 are used for the primary font; bits 3:2 for the secondary font.

3) To retain eight consistent colors during 512-character display, Subfunction 00h of Function 10h must be called first with the following setting:

AX = 1000h, BX = 0712h

5.36 Function 11h, Subfunction 04h: Load 8 × 16 ROM Font

[Entry]

AH = 11h
AL = 04h (Subfunction)
BL = Block to load (00h–07h)

[Return]

None

[Notes]

1) This function is only available for Text modes.

2) The character height is 16 bytes, but the height of the display cell is the same as the default setting.

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts and is available for the Character Generator only. Consequently, the block value specified in register BL is in the range of 00h (default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors).
if Subfunction 00h of Function 10h is called with BX = 0712h
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors).

5) The default setting by the BIOS loads a font into Block 0 used for both the primary and secondary fonts (256 displayable characters).

6) The font loading by Subfunction 00h requires caution. Since the controller is not reprogrammed, abnormal character display may occur. For example:

a) If a font table is loaded to override the font in a block, a double image may result (especially when the loaded font size is smaller than the one previously displayed). In this instance, the application should use subfunction 14h.

5.37 Function 11h, Subfunction 10h: Load User Text Font and Reprogram Controller

[Entry]

AH = 11h
AL = 10h (Subfunction)
BH = Number of bytes per character
BL = Block to load (00h–07h)
CX = Number of characters to store
DX = ASCII character ID of the first character in the font table (ES: BP)

ES: BP = Point to user-provided font table

[Return]

None

[Notes]

1) This function is only available for Text modes.

2) The value in register BH represents the height of each character. The value can be the specified maximum of 32 bytes-per-character in standard VGA specification.

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is available for the Character Generator only. Consequently, the block value specified in register BL is in the range of 00h (default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors).

if Subfunction 00h of Function 10h is called with BX = 0712h

Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)

else,

Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors).

5) The default setting by the BIOS loads a font into Block 0 used for both the primary and secondary fonts (256 displayable characters).

6) Subfunction 10h is almost identical to Subfunction 00h, except for the following differences:

a) Page 00h must be active.

b) Character height (bytes-per-character) is recalculated.

c) Number of rows (0 base) are recalculated as:
(scanlines-per-screen ÷ Character Height) – 1.

d) The length of the display buffer is recalculated as (Total number of rows × Total number of columns) × 2 (1 base).

e) The CRTC Registers are reprogrammed as follows:

Index	Register NameChange
09h	Maximum scanlinesCharacter Height – 1
0Ah	Cursor Start Character Height – 2
0Bh	Cursor EndCharacter Height – 1
12h	Vertical Display Enable End

For 350 or 400 scanline modes

(Rows per screen × Character Height) – 1

For 200 scanline modes

(Rows per screen × Character Height) × 2) – 1

14h	Underline Location	Character Height – 1 (Mode 07h only)

f) It must be called immediately after Function 00h call (Set Display Mode) or the result is unpredictable.

5.38 Function 11h, Subfunction 11h: Load 8 × 14 ROM Font and Reprogram Controller

[Entry]

AH = 11h
AL = 11h (Subfunction)
BL = Block to load (00h–07h)

[Return]

None

NOTE: This is actually an 8 × 16 font.

[Notes]

1) This function is only available for Text modes.

2) The character and display cells are both 14-bytes high (scanlines).

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is available for the Character Generator only. Consequently, the block value specified in register BL is in the range of 00h (default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors).
if Subfunction 00h of Function 10h is called with BX = 0712h
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors).

5) The default setting by the BIOS loads a font into Block 0 used for both the primary and secondary fonts (256 displayable characters).

6) Subfunction 11h is almost identical to Subfunction 01h, except for the following differences:

a) Page 00h must be active.

b) Character Height = 14.

c) Number of rows (0 base) are recalculated as:
(scanlines-per-screen ÷ Character Height) – 1

d) The length of the display buffer is recalculated as:

(Total number of rows × Total number of columns) × 2 (1 base)

e) The CRTC registers are reprogrammed as follows:

Index	Register Name	Change
09h	Maximum Scanlines	13 (0Dh)
0Ah	Cursor Start	12 (0Ch)
0Bh	Cursor End	13 (0Dh)
12h	Vertical Display Enable End	(Rows per screen × 14) – 1
14h	Underline Location (mode 07h only)	13 (0Dh)

f) It must be called immediately after Function 00h call (Set Display Mode) or the result is unpredictable.

5.39 Function 11h, Subfunction 12h: Load 8 × 8 ROM Font and Reprogram Controller

[Entry]

AH = 11h
AL = 12h (Subfunction)
BL = Block to load (00h–07h)

[Return]

None

[Notes]

1) This function is only available for Text modes.

2) The height of character and display cell are all 8 bytes (scanlines).

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is available for the Character Generator only. Consequently, the block value specified in register BL is in the range of 00h (default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 — Primary font selected and normal display (eight foreground colors).

if Subfunction 00h of Function 10h is called with BX = 0712h
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors).

5) The default setting by the BIOS loads a font into Block 0 used for both the primary and secondary fonts (256 displayable characters).

6) Subfunction 12h is almost identical to Subfunction 02h, except for the following differences:

a) Page 00h must be active.

b) Character Height = 8.

c) Number of rows (0 base) are recalculated as
(Scanlines per screen/Character Height) – 1.

d) The length of display buffer is recalculated as:
(Total number of rows × Total number of columns) × 2 (1 base).

e) The CRTC registers are reprogrammed as follows:

Index	Register Name	Change
09h	Maximum Scanlines	7 (07h)
0Ah	Cursor Start	6 (06h)
0Bh	Cursor End	7 (07h)
12h	Vertical Display Enable End	(Rows per screen × 8) – 1
14h	Underline Location (mode 07h only)	7 (07h)

f) It must be called immediately after Function 00h call (Set Display Mode). Otherwise, the result is unpredictable.

5.40 Function 11h, Subfunction 14h: Load 8 × 16 ROM Font and Reprogram Controller

[Entry]
AH = 11h
AL = 14h (Subfunction)
BL = Block to load (00h–07h)

[Return]
None

[Notes]

1) This function is only available for Text modes.

2) The height of character and display cells are all 16 bytes (scanlines).

3) In VGA, the character font is loaded into RAM Map 2 (0 base), which can contain up to eight fonts at any time and is available for the Character Generator only. Consequently, the block value specified in register BL is in the range of 00h (default) to 07h.

4) Two out of eight character fonts can be used at any time. This provides 512 simultaneously displayable characters, instead of 256 characters. The way to display two different fonts at one time can be accomplished by the following:

a) Load fonts into desired blocks.

b) Subfunction 03h (Select Block Specifier) of Function 11h is called to select two out of eight different font blocks; one for the primary font, the other for the secondary font.

c) Bit 3 of the Attribute Byte serves as the Font Block Selector and foreground intensity for the character.

Bit 3 = 0 – Primary font selected and normal display (eight foreground colors).
If Subfunction 00h of Function 10h is called with BX = 0712h
Bit 3 = 1 — Secondary font selected and normal display (eight foreground colors)
else,
Bit 3 = 1 — Secondary font selected and intensity display (16 foreground colors).

5) The default setting by the BIOS loads a font into Block 0 used for both primary font and secondary font (256 displayable characters).

6) Subfunction 14h is almost identical to Subfunction 04h, except for the following differences:

a) Page 00h must be active.

b) Character Height = 16.

c) Number of Rows (0 base) are recalculated as:
(Scanlines per screen ÷ Character Height) – 1.

d) The length of display buffer is recalculated as (Total number of rows × Total number of columns) × 2 (1 base).

e) The CRTC registers are reprogrammed as follows:

Index	Register Name	Change
09h	Maximum Scanlines	15 (0Fh)
0Ah	Cursor Start	14 (0Eh)
0Bh	Cursor End	15 (0Fh)
12h	Vertical Display Enable End	(Rows per screen × 16) – 1
14h	Underline Location	15 (0Fh) (mode 07h only)

f) It must be called immediately after Function 00h call (Set Display Mode) or the result is unpredictable.

5.41 Function 11h, Subfunction 20h: Set Pointer of User's Graphics Font Table to Interrupt 1Fh

[Entry]

AH = 11h
AL = 20h (Subfunction)
ES: BP = Point to user's graphics font table

[Return]

None

[Notes]

1) The value in this Interrupt Vector serves as a pointer that points to the Graphics font for character codes (80h–FFh) in modes 04h, 05h, and 06h.

2) In the CGA adapter, the planar BIOS only provides 128 character codes (00h–7Fh). The user can supply the other half of the character codes (80h–FFh), or use GRAFTABL in DOS to load these character codes.

3) This function must be called immediately after Set Display Mode.

5.42 Function 11h, Subfunction 21h: Set Pointer of User's Graphics Font Table to Interrupt 43h

[Entry]

AH = 11h
AL = 21h (Subfunction)
BL = Character rows specifier
- 00h = Value in register DL (the number of displayable rows specified by user)
- 01h = 14 (0Eh) character rows
- 02h = 25 (19h) character rows
- 03h = 43 (2Bh) character rows

CX = Bytes per character
DL = Number of character rows (if register BL = 00h)
ES: BP = Point to user's graphics font table

[Return]

None

[Note]

1) The value in this interrupt vector serves as a pointer that points to the Graphics font for character codes (00h–7Fh) in modes 04h, 05h, and 06h. In all other Graphics modes, the vector also manages the Graphics font for character codes (00h–FFh).

2) Only call this function immediately after Set Display Mode.

3) The portion of character rows above displayable rows is not displayed unless it is scrolled up.

4) The overlapping screen may occur on display modes that use all display memory addresses, such as mode 13h.

5.43 Function 11h, Subfunction 22h: Set Pointer of ROM 8 × 14 Graphics Font Table to Interrupt 43h

[Entry]

AH = 11h
AL = 22h (Subfunction)
BL = The specifier of character rows on screen
- 00h = Value in register DL (the number of displayable rows specified by user)
- 01h = 14 (0Eh) character rows
- 02h = 25 (19h) character rows
- 03h = 43 (2Bh) character rows

DL = Number of character rows to display (if register BL = 00h)

[Return]

None

NOTE: This is actually an 8 × 16 font.

[Notes]

1) The value in this interrupt vector serves as a pointer that points to the Graphics font for Character Codes (00h–7Fh) in Modes 04h, 05h, and 06h. In all other Graphics modes, the vector also manages the Graphics font for character codes (00h–FFh).

2) Only call this function immediately after setting Display Mode.

3) The portion of character rows above displayable rows are be displayed unless it is scrolled up.

4) The overlapping screen can occur on display modes that use all Display Memory addresses, such as mode 13h.

5.44 Function 11h, Subfunction 23h: Set Pointer of ROM 8 × 8 Graphics Font Table to Interrupt 43h

[Entry]

AH = 11h
AL = 23h (Subfunction)
BL = Specifier of character rows on screen
- 00h = Value in Register DL (the number of displayable rows specified by user)
- 01h = 14 (0Eh) character rows
- 02h = 25 (19h) character rows
- 03h = 43 (2Bh) character rows

DL = Number of character rows to display (if Register BL = 00h)

[Return]

None

[Notes]

1) The value in this interrupt vector serves as a pointer that points to the Graphics font for character codes (00h–7Fh) in modes 04h, 05h, and 06h. In all other Graphics modes, the vector also manages the Graphics font for character codes (00h–FFh).

2) Only call this function immediately after setting Display Mode.

3) The portion of character rows above displayable rows are not displayed unless it is scrolled up.

4) The overlapping screen can occur on display modes that use all Display Memory addresses, such as mode 13h.

5.45 Function 11h, Subfunction 24h: Set Pointer of ROM 8 × 16 Graphics Font Table to Interrupt 43h

[Entry]

AH = 11h
AL = 24h (Subfunction)
BL = Specifier of character rows on screen
- 00h = Value in Register DL (the number of displayable rows specified by user)
- 01h = 14 (0Eh) character rows
- 02h = 25 (19h) character rows
- 03h = 43 (2Bh) character rows

DL = Number of character rows to display (if Register BL = 00h)

[Return]

None

[Notes]

1) The value in this interrupt vector serves as a pointer that points to the Graphics font for character codes (00h–7Fh) in modes 04h, 05h, and 06h. In all other Graphics modes, the vector also manages the Graphics font for character codes (00h–FFh).

2) Only call this function immediately after setting Display mode.

3) The portion of character rows above displayable rows are not displayed unless it is scrolled up.

4) The overlapping screen can occur on display modes that use all Display Memory addresses, such as mode 13h.

5.46 Function 11h, Subfunction 30h: Get Pointer Information of Fonts

[Entry]

AH = 11h
AL = 30h (Subfunction)
Bh = Pointer information of fonts
00h – Current Font Pointer stored in Interrupt Vector 1Fh
01h – Current Font Pointer stored in Interrupt Vector 43h
02h – Font Pointer of ROM 8 × 14 Font Table
03h – Font Pointer of ROM 8 × 8 Font Table (Character Codes 00h–7Fh)
04h – Font Pointer of ROM 8 × 8 Font Table (Character Codes 80h–FFh)
05h – Font Pointer of ROM 8 × 16 Alternate Font Table
06h – Font Pointer of ROM 8 × 16 Font Table
07h – Font Pointer of ROM 9 × 16 Alternate Font Table

[Return]

CX = Current character height (Bytes per Character)
DL = Number of rows of current display mode (0 base)
ES: BP = Pointer information of desired font

5.47 Function 12h, Subfunction 10h: Get Current Display Configuration

[Entry]

AH = 12h
BL = 10h (Subfunction)

[Return]

BH = 00h – Color Mode (3Dx)
01h – Monochrome Mode (3Bx)
BL = Display Memory Size
00h = 64 Kbytes
01h = 128 Kbytes
02h = 192 Kbytes
03h = 256 Kbytes

CH = Feature Bits	Feature Control Output Bit Setting	Input Status Register 0 – (3C2h)
0	0	Bit 5
1	0	Bit 6
2	1	Bit 5
3	1	Bit 6
4–7	Reserved	

CL = Switch Settings	Description
Bit 0	Configuration Switch-1
Bit 1	Configuration Switch-2
Bit 2	Configuration Switch-3
Bit 3	Configuration Switch-4
Bits 4–7	Reserved

5.48 Function 12h, Subfunction 20h: Alternate PrintScreen Handler

[Entry]

AH = 12h
BL = 20h (Subfunction)

[Return]

None

[Note]

1) This function call replaces the original PrintScreen Interrupt Handler (Interrupt 05h) to support the modes whose displayable rows on screen are over 25 rows.

5.49 Function 12h, Subfunction 30h: Select Scanlines for Text Modes

[Entry]

AH = 12h
BL = 30h (Subfunction)
AL = Specifier of scanlines
  00h = 200 scanlines
  01h = 350 scanlines
  02h = 400 scanlines

[Return]

AL = 12h (function supported)

[Notes]

1) The selected scanlines take effect on next mode setting.

2) Mode 07h only supports 350/400 scanlines. Modes 00h – 03h supports 200, 350, 400 scanlines,

respectively.

3) 200 scanlines modes are double-scanned to 400 scanlines.

5.50 Function 12h, Subfunction 31h: Enable/Disable Default Palette Loading

[Entry]

AH = 12h
BL = 31h (Subfunction)
AL = 00h – Enable default palette loading
01h – Disable default palette loading

[Return]

AL = 12h (function supported)

[Notes]

1) This function takes effect on next mode setting.

2) All Internal/External Palette registers are affected.

5.51 Function 12h, Subfunction 32h: Enable/Disable Display

[Entry]

AH = 12h
BL = 32h (Subfunction)
AL = 00h – Enable Display
01h – Disable Display

[Return]

AL = 12h (function supported)

[Note]

1) Display subsystem does not respond to any I/O or Display Memory addressing.

5.52 Function 12h, Subfunction 33h: Enable/Disable Summing-to-Grayshades

[Entry]

AH = 12h
BL = 33h (Subfunction)
AL = 00h – Enable summing-to-grayshades
01h – Disable summing-to-grayshades

[Return]

AL = 12h (function supported)

[Note]

1) This function takes effect on a subsequent mode setting or internal/external palettes setting.

5.53 Function 12h, Subfunction 34h: Enable/Disable Cursor Emulation

[Entry]

AH = 12h
BL = 34h (Subfunction)
AL = 00h – Enable cursor emulation
01h – Disable cursor emulation

[Return]

AL = 12h (function supported)

[Notes]

1) This function takes effect on a subsequent mode setting or Function 01h call (set cursor type).

2) Bit 0 of Address [40:87] Emulation Flag is affected.

5.54 Function 12h, Subfunction 35h: Switch Display

[Entry]

AH = 12h
BL = 35h (Subfunction)
AL = 00h – Initial display adapter turned off
(ES:DX must point to a 128-byte buffer for switching state save area)
01h – System board display turned on
02h – Active display turned off (ES:DX must point to a buffer for switching state save area)
03h – Inactive display turned on (ES:DX must point to a buffer which saves switching state previously)
ES:DX = Buffer for switching state (valid when AL = 00h, 02h, or 03h)

[Return]

AL = 12h (function supported)

[Notes]

1) There are several requirements that must be met before using this function:

a) Two display subsystems coexisting: system board display and display adapter.

b) The usage of display resources conflicts between two display systems.

c) Display adapter is primary display; system board is secondary display.

d) This function must be supported by system board display and display adapter.

2) If the first time switching from display adapter to system board display:

Call the function with register AL = 00h
Call the function with register AL = 01h
else,
Call the function with register AL = 02h
Call the function with register AL = 03h

5.55 Function 12h, Subfunction 36h: Enable/Disable Screen Display

[Entry]

AH = 12h
BL = 36h (Subfunction)
AL = 00h – Enable screen display
01h – Disable screen display

[Return]

AL = 12h (function supported)

[Note]

1) This function can be used for fast display memory updating without losing synchronization.

5.56 Function 13h: Write Teletype String

[Entry]

AH = 13h
AL = Write function specifier
00h – Write character string without updating cursor (BL = Attribute)
01h – Write character string with updating cursor (BL = Attribute)
02h – Write character/attribute string without updating cursor
03h – Write character/attribute string with updating cursor
BH = Display page (0 base)
BL = Attribute (valid when AL = 00h or 01h)
CX = String length
DH = Start Y coordinate of string displayed on screen
DL = Start X coordinate of string displayed on screen
ES: BP = Start address of string (in Segment: Offset Format)

[Return]

None

[Notes]

1) Control Characters ((such as, LF, CR, Backspace, and BELL) are recognized. (ASCII codes: LF = 0Ah, CR = 0Dh, Backspace = 08h, BELL = 07h).

2) String can be written to any pages regardless of active state.

3) Line wrapping and screen scrolling are supported. Screen scrolling is only supported on active page.

4) if bit 7 of the BL register is set in graphics modes, the color value in the register is XOR'ed with the contents of display memory.

5.57 Function 1Ah, Subfunction 00h: Get Display Combination Code (DCC)

[Entry]

AH = 1Ah
AL = 00h (Subfunction)

[Return]

If function supported:
AL = 1Ah
BH = Alternate display code
BL = Active display code

[Notes]

1) The index of current DCC entry in DCC table is stored in address [40:8A].

2) Display Combination code definition:

Code	Definition
00h	No Display
01h	Monochrome Display Adapter (MDA)
02h	Color Display Adapter (CGA)
03h	Reserved
04h	EGA with Color Monitor (EGA)
05h	EGA with Monochrome Monitor (MEGA)
06h	Professional Graphics Adapter with Color Display (PGA)
07h	Video Graphics Array with Analog Monochrome Monitor (MVGA)
08h	Video Graphics Array with Analog Color Monitor (VGA)

5.58 Function 1Ah, Subfunction 01h: Set Display Combination Code (DCC)

[Entry]

AH = 1Ah
AL = 01h (Subfunction)
BH = Alternate display code
BL = Active display code

[Return]

If function supported:
AL = 1Ah

[Note]

1) Display Combination code definition:

Code	Definition
00h	No Display
01h	Monochrome Display Adapter (MDA)
02h	Color Display Adapter (CGA)
03h	Reserved
04h	EGA with Color Monitor (EGA)
05h	EGA with Monochrome Monitor (MEGA)

06h	Professional Graphics Adapter with Color Display (PGA)
07h	Video Graphics Array with Analog Monochrome Monitor (MVGA)
08h	Video Graphics Array with Analog Color Monitor (VGA)

2) User is responsible for providing correct DCC. There is no physical checking of the device.

5.59 Function 1Bh: Collection of Display Information

[Entry]

AH = 1Bh
BX = 00h
ES: DI = Pointer points to 128-byte buffer

[Return]

If function supported
AL = 1Bh

[Notes]

1) Display information in 128-byte buffer:

Offset	Size	Definition
00h	2 words	Pointer points to collection of static functionality information
04h	Byte	Current Display Mode
05h	Word	Number of columns (1 base)
07h	Word	Refresh Buffer Length (unit: byte)
09h	Word	The starting address of Refresh Buffer (Offset value relates to start of display memory; default = 0000h)
0Bh	8 words	Cursor Position for each page (maximum eight pages supported)
1Bh	Word	Current Cursor Type (High byte = start scanline; Low byte = end scanline)
1Dh	Byte	Active Display Page
1Eh	Word	Base Port Address of CRT Controller (CRTC) (Monochrome = 3Bxh, Color = 3Dxh)
20h	Byte	Current setting of 3B8h or 3D8h (Mode Control Register)
21h	Byte	Current setting of 3B9h or 3D9h
22h	Byte	Number of rows (1 base)
23h	Word	Character height (1 base; unit: scanline)
25h	Byte	Active Display Code
26h	Byte	Alternate Display Code
27h	Word	Number of displayable colors (1 base; monochrome = 0000h)
29h	Byte	Number of Pages (1 base)
2Ah	Byte	Specifier of vertical resolution 00h = 200 scanlines 01h = 350 scanlines 02h = 400 scanlines 03h = 480 scanlines 04h–FFh = Reserved

2Bh	Byte	Primary Font Block (00h–07h)
2Ch	Byte	Secondary Font Block (00h–07h)
2Dh	Byte	Flags of Display State:

Bit	**Definition**
7:6	Reserved
5	0 = Background intensity 1 = Blinking (Default)
4	0 = Cursor Emulation Disable 1 = Cursor Emulation Enable
3	0 = Default Palette Loading Enable 1 = Default Palette Loading Disable
2	0 = Color Monitor Attached 1 = Monochrome Monitor Attached
1	0 = Summing-to-grayshades Disable 1 = Summing-to-grayshades Enable
0	1 = All Modes are active on all displays

2E–30h	Reserved	
31h	Byte	Specifier of total display RAM 00h = 64 Kbytes 01h = 128 Kbytes 02h = 192 Kbytes 03h = 256 Kbytes 04h–FFh = Reserved
32h	Byte	Save pointer state information:

Bit	Definition
7:6	Reserved
5	1 = Extension of Display Combination Code Active
4	1 = Palette Override Active
3	1 = Graphics Font Override Active
2	1 = Alpha Font Override Active
1	1 = Dynamic Save Area Active
0	1 = 512-character Set Active

33–3Fh	Reserved

2) Collection of static functionality information:

Offset	**Size**	**Definition**
00h	Byte	Available display modes if bit set:

Bit	**Display Mode**
0	00h
1	01h
2	02h
3	03h
4	04h
5	05h
6	06h
7	07h

01h	Byte	Available display modes if bit set:
		Bit **Display Mode**
		0 08h
		1 09h
		2 0Ah
		3 0Bh
		4 0Ch
		5 0Dh
		6 0Eh
		7 0Fh
00h	Byte	Available display modes if bit set:
		Bit **Display Mode**
		0 10h
		1 11h
		2 12h
		3 13h
		4–7 Reserved
03–06h	Reserved	
07h	Byte	Number of scanlines available in text modes: (Subfunction 30h, Function 12h)
		Bit **Scanlines (if Bit = 1)**
		0 200
		1 350
		2 400
		3–7 Reserved
08h	Byte	Number of active character blocks available in text modes
09h	Byte	Maximum number of character blocks available in text modes
0Ah	Byte	Supported functions (No. 1):
		Bit **Function (if Bit = 1)**
		0 All Modes on All Displays
		1 Summing to grayshades
		2 Character fonts Loading
		3 Default Palette Loading
		4 Cursor Emulation
		5 EGA Palettes (Internal Palettes)
		6 Color Palettes (External Palettes/RAMDAC)
		7 Color Paging
0Bh	Byte	Supported functions (No. 2):
		Bit **Function (if Bit = 1)**
		0 Reserved
		1 Save/Restore Display State
		2 Background Intensity/Blinking Control
		3 Set Display Combination Code
		4–7 Reserved
0C–0Dh	Reserved	
0Eh	Save Pointer Functions:	
		Bit **Function (if Bit = 1)**

0	512-character Set
1	Dynamic Save Area
2	Alpha Font Override
3	Graphics Font Override
4	Palette Override
5	Extension of Display Combination Code
6–7	Reserved

0Fh Reserved

5.60 Function 1Ch, Subfunction: 00h: Get Buffer Size for Display State

[Entry]

AH = 1Ch

AL = 00h (Subfunction)

CX = Requested Display State:

Bit	Display State
0	Hardware State
1	BIOS Data Area
2	Color Registers (External Palettes/RAMDAC)
3–15	Reserved

[Return]

If function supported

AL = 1Ch

BX = Blocks/Buffer (Unit: 64 byte/block)

[Note]

1) This function reports the sufficient size of buffer to save display state. Call this subfunction first to guarantee Subfunction 01h and 02h are performed successfully.

5.61 Function 1Ch, Subfunction 01h: Saving Display State

[Entry]

AH = 1Ch

AL = 01h (Subfunction)

CX = Requested Display States:

Bit	Display States
0	Hardware State
1	BIOS Data Area
2	Color Registers (External Palettes/RAMDAC)
3–15	Reserved

ES: BX = Pointer points to buffer (Segment: Offset format)

[Return]

If function supported:

AL = 1Ch

ES: BX = State information saved in user-supplied buffer

5.62 Function 1Ch, Subfunction 02h: Restore Display State

[Entry]

AH = 1Ch

AL = 02h (Subfunction)

CX = Requested Display States:

Bit	Display States
0	Hardware State
1	BIOS Data Area
2	Color Registers (External Palettes/RAMDAC)
3–15	Reserved

ES: BX = Pointer points to previous saved buffer (Segment: Offset format)

[Return]

If function supported:

AL = 1Ch

6. VGA SLEEP MODE AND DISPLAY SWITCHING

The IBM VGA standard supports a Sleep mode feature to enable/disable CPU addressing of the VGA subsystem display memory and I/O ports. For integrated VGA subsystems on the motherboard, the display subsystem is enabled or disabled by programming a Display Subsystem Enable register at I/O Port 3C3h. On VGA adapter cards, a control register at I/O Port 46E8h is used. These two separate schemes of enabling/disabling addressing allows two VGAs (driving separate display monitors) to coexist in a system and have the capability to switch active display from one display to another. The IBM standard VGA BIOS supports a set of function calls to select Sleep mode and display switching features.

The CL-GD5446 VGA controller, depending on the application, can be programmed to respond at either 3C3h or 46E8h I/O port for enabling/disabling CPU addressing. This allows for full IBM VGA compatibility, whether the design is an integrated motherboard VGA or an adapter card solution.

7. ADDRESS MAPS

The following tables provide background information regarding the usage of system memory, port address space, and interrupt vectors by DOS and its I/O routines (Planar and Peripheral BIOSes). The areas of interest to display subsystem users and designers are highlighted in bold text.

Table C2-2. MS DOS Memory Map After Loading

FE0000–FFFFFF	128 Kbytes to ‘shadow’ system ROM BIOS
100000–FDFFFF	15 Mbytes of extended memory in protected mode only
FFFF:000F (1 Mbyte)	Planar BIOS
F000:0000	Expansion BIOS (motherboard video BIOS)
E000:0000	Voice Communication BIOS/LIM EMS page map area
D000:8000	Network BIOS/LIM EMS page map area
D000:0000	LIM EMS page map area
C000:C000	Hard disk BIOS
C000:8000	EGA/VGA adapter BIOS

Table C2-2. MS DOS Memory Map After Loading *(cont.)*

C000:0000	EGA display RAM
B000:C000	CGA display RAM (or HGC mode graphics RAM)
B000:8000	HGC display RAM
B000:4000	MDA/HGC display RAM
B000:0000	**EGA/VGA display RAM**
A000:0000	Top of system RAM COMMAND.COM (transient portion), free RAM, COMMAND.COM (resident portion), installable device drivers, file control blocks, disk buffers, DOS tables, DOS kernel (MSDOS.SYS), resident DOS device drivers (IO.SYS)
0000:0600	ROM BIOS data area
0000:0400	Interrupt vectors
0000:0000	

Table C2-3. BIOS Data Area Assignments

0040:0000 WORD	COM1 Port base address
0040:0002 WORD	COM2 Port base address
0040:0004 WORD	COM3 Port base address
0040:0006 WORD	COM4 Port base address
0040:0008 WORD	Printer 1 port base address
0040:000A WORD	Printer 2 port base address
0040:000C WORD	Printer 3 port base address
0040:000E WORD	Printer 4 port base address

0040:0010 WORD EQUIPMENT_FLAG

Bit	Description
D[15:14]	Number of printer adapters
D[13:12]	Reserved
D[11:9]	Number of RS232-C
D8	Reserved
D[7:6]	Number of diskette drives
D[5:4]	Identify the current primary display device:
D[3:2]	Reserved
D1	Math coprocessor
D0	IPL diskette

D5	D4	Adapter
0	0	EGA (or none)
0	1	CGA 40 × 25
1	0	CGA 80 × 25
1	1	MDA

Table C2-3. BIOS Data Area Assignments *(cont.)*

0040:0012 BYTE		Reserved
0040:0013 WORD	USABLE_RAM	Usable memory size in Kbytes
0040:0015 BYTE		Reserved
0040:0016 BYTE		Reserved
0040:0017 WORD	KBD_CNTRL	Stores status of special keys
0040:0019 BYTE	ALT_KBD	Alternate keypad entry
0040:001A WORD	KBD_BUF_HD	Points to head of keyboard buffer
0040:001C WORD	KBD_BUF_TL	Points to tail of keyboard buffer
0040:001E 16 WDS	KBD_BUFFER	Circular keyboard buffer
0040:003E BYTE		Diskette drive recalibrate status
0040:003F BYTE		Diskette drive motor status
0040:0040 BYTE		Diskette drive motor off counter
0040:0041 BYTE		Last diskette driver operation status
0040:0042 7 BYTES		Diskette driver controller status
0040:0049 BYTE	**VIDEO_MODE**	**Current BIOS Video mode**
0040:004A WORD	**COLUMNS**	**Number of text columns**
0040:004C WORD	**PAGE_LENGTH**	**Length of each page in bytes**
0040:004E WORD	**START_ADDR**	**Start Address register value for page**
0040:0050 8 WDS	**CURSOR_POS**	**Cursor positions for all eight pages The high byte of each word contains the character row; the low byte contains the column.**
0040:0060 WORD	**CURSOR_TYPE**	**Start and ending lines for text cursor The high byte contains the start line.**
0040:0062 BYTE	**ACTIVE_PAGE**	**Currently displayed page number**
0040:0063 WORD	**ADDR_CRTC**	**I/O Port address of 6845/CRTC address register (3B4h monochrome; 3D4h color)**
0040:0065 BYTE	**CRT_MODE_SET**	**Current value for Mode Control register (3B8h MDA; 3D8h CGA) The EGA and VGA values emulate the MDA/CGA values.**
0040:0066 BYTE	**CRT_PALETTE**	**Current value for the CGA color select register (3D9) Emulated by EGA/VGA.**

Table C2-3. BIOS Data Area Assignments *(cont.)*

0040:0067 DWORD	Pointer to MCA PS/2 reset code
0040:006B BYTE	Reserved
0040:006C DWORD	Timer counter
0040:0070 BYTE	Timer overflow
0040:0071 BYTE	Break key state
0040:0072 WORD	RESET flag
0040:0074 BYTE	Last hard disk drive operation status
0040:0075 BYTE	Number of hard disk drives attached
0040:0076 BYTE	PC XT hard disk drive control
0040:0077 BYTE	PC XT hard disk drive controller port
0040:0078 BYTE	Printer 1 timeout value
0040:0079 BYTE	Printer 2 timeout value
0040:007A BYTE	Printer 3 timeout value
0040:007B BYTE	Printer 4 timeout value
0040:007C BYTE	COM1 timeout value
0040:007D BYTE	COM2 timeout value
0040:007E BYTE	COM3 timeout value
0040:007F BYTE	COM4 timeout value
0040:0080 WORD	Keyboard Buffer Start Offset pointer
0040:0082 WORD	Keyboard Buffer End Offset pointer

Table C2-3. BIOS Data Area Assignments *(cont.)*

0040:0084	**BYTE**	**ROWS**	**Number of text rows minus 1**
0040:0085	**WORD**	**CHAR_HEIGHT**	**Bytes-per-character**
0040:0087	**BYTE**	**INFO_1**	

Bit	Description
D7	Equals bit D7 from AL register on most recent mode select ('1' indicates display memory was not cleared by mode select)
D[6:5]	Display memory size (00 = 64K, 01 = 128K, 10 = 192K, 11 = 256K).
D4	Reserved.
D3	'0' indicates EGA is the primary display
D2	'1' forces the BIOS to wait for Vertical Retrace before memory write.
D1	'1' indicates that EGA is in Monochrome mode.
D0	'0' indicates that CGA cursor emulation enabled. The cursor shape is modified if enhanced text is used.

Table C2-3. BIOS Data Area Assignments *(cont.)*

0040:0088 BYTE INFO_3

Bit	Description
D[7:4]	Feature Control bits (from Feature Control register)
D[3:0]	EGA Configuration Switch settings

0040:0089 BYTE FLAGS Miscellaneous flags

Bit	Description
D7	Alphanumeric scanlines (with bit 4): 7 / 4 / Mode 0 / 0 / 350-line mode 0 / 1 / 400-line mode 1 / 0 / 200-line mode 1 / 1 / Reserved
D6	'1' indicates display switching enabled
	'0' indicates display switching disabled
D5	Reserved
D4	(see bit 7)
D3	'1' indicates default palette loading disabled
	'0' indicates default palette loading enabled
D2	'1' indicates using monochrome monitor
	'0' indicates using color monitor
D1	'1' indicates grayscale summing enabled
	'0' indicates grayscale summing disabled
D0	'1' indicates VGA active
	'0' indicates VGA not active

Table C2-3. BIOS Data Area Assignments *(cont.)*

0040:008A BYTE	Reserved
0040:008B BYTE	Media control
0040:008C BYTE	Hard disk drive controller status
0040:008D BYTE	Hard disk drive error status
0040:008E BYTE	Hard disk drive interrupt control
0040:008F BYTE	Reserved
0040:0090 BYTE	Drive 0 Media state
0040:0091 BYTE	Drive 1 Media state
0040:0092 BYTE	Reserved
0040:0093 BYTE	Reserved
0040:0094 BYTE	Drive 0 Current cylinder
0040:0095 BYTE	Drive 1 Current cylinder
0040:0096 BYTE	Keyboard Mode State and Type flags
0040:0097 BYTE	Keyboard LED flags
0040:0098 WORD	Address offset to User Wait Complete flag
0040:009A WORD	Segment address to User Wait Complete flag
0040:009C WORD	User wait count – low word (µsecs.)
0040:009E WORD	User wait count – high word (µsecs.)
0040:00A0 BYTE	Wait active flag
0040:00A1 BYTE	Reserved
0040:00A2 BYTE	Reserved
0040:00A3 BYTE	Reserved
0040:00A4 BYTE	Reserved
0040:00A5 BYTE	Reserved
0040:00A6 BYTE	Reserved
0040:00A7 BYTE	Reserved
0040:00A8 DWORD	**SAVE_PTR Pointer to BIOS Save Pointer Table**

NOTE: The next 84 bytes from 0040:00A1 to 0040:00FF are reserved.

Table C2-4. I/O Port Assignment for PC XT and AT Computers

Port Usage for PC XT	I/O Address	Port Usage for AT
DMA controller	000–0lF	DMA controller
Interrupt controller	020–03F	Interrupt controller
Timer	040–04F	Coprocessor access, Timer
	050–05F	Timer
PPI (system configuration)	060–063	
	060–06F	Keyboard
Reserved	070–07F	Real-time clock
DMA Page register	080–09F	DMA Page register
NMI Mask Register	0A0–0AF	
	0A0–0BF	Interrupt Controller
Reserved	0B0–0FF	
	0C0–0DF	DMA controller
	0F0–0FF	Math coprocessor
Unusable	100–13F	Reserved
Unusable	140–14F	Token ring adapter
Unusable	150–15F	Advanced color graphics display
Unusable	160–16F	Advanced mono. graphics display
Unusable	170–177	Fixed-disk adapter
Unusable	1C0–1CF	Token ring adapter
Unusable	1E8–1EF	Streaming tape drive adapter
Unusable	lF0–lF7	Fixed-disk adapter
Unusable	1F8–1FF	Reserved
Game I/O	200–20F	Game I/O
Expansion Unit	210–217	
Multifunction Card	218–21F	Multifunction card
Reserved	220–24F	
	278–27F	Parallel Port 2

Table C2-4. I/O Port Assignment for PC XT and AT Computers *(cont.)*

Port Usage for PC XT	I/O Address	Port Usage for AT
Clock Calenda	2C0–2CF	Clock calendar
	2D0–2DF	3278/79 Emulation adapter, clock/calendar
Serial Port 4	2E0–2E7	
Serial Port 3 or 4	2E8–2EF	
Reserved	2F0–2F7	Interrupt sharing
Serial Port 2	2F8–2FF	Serial Port 2
Prototype Card	300–31F	Prototype card
Fixed Disk	320–32F	
	360–36F	PC network
Parallel Port 1	378–37F	Parallel Port 1
SDLC	380–38F	SDLC, bisync 2
Bisync	3A0–3AF	Bisync 1
MDA and printer adapter	**3B0–3BF**	**MDA, EGA/VGA and printer adapter**
EGA/VGA Adapter	3C0–3CF	EGA/VGA
CGA	3D0–3DF	CGA, EGA/VGA
Reserved	3E0–3E7	
Serial Port 3	3E8–3EF	
Diskette Controller	3F0–3F7	Diskette controller
Serial Port 1	3F8–3FF	Serial Port 1
	400–43F	Reserved
	440–44F	Coprocessor access
	450–50F	Reserved
	510–52F	Multi-protocol adapter
	550–557	Coprocessor to main CPU communication
	6F0–6F7	Interrupt sharing

Table C2-4. I/O Port Assignment for PC XT and AT Computers *(cont.)*

Port Usage for PC XT	I/O Address	Port Usage for AT
	910–92F	Multi-protocol adapter
	D10–D2F	Extended monochrome graphics display
	E90–E9F	PSLA
	1230–124F	1st address range: multi-port async
	2230–224F	2nd address range: multi-port async
	3230–324F	3rd address range: multi-port async
	4230–424F	4th address range: multi-port async
	46E8	**VGA add-in Adapter Sleep Enable**

NOTE: Use of port for this function is common, but not standard.

Table C2-5. Interrupt Vector Assignments

Vector Table Entry	INT Number	Name	
0000:0000	0	Divide by Zero	
0000:0004	1	Single Step	
0000:0008	2	Non-maskable	
0000:000C	3	Break-Point	
0000:0010	4	Overflow	
0000:0014	5	Print Screen	
0000:0018	6	Reserved	
0000:001D	7	Reserved	
0000:0020	8	Time	H/W IRQ0
0000:0024	9	Keyboard	H/W IRQ1
0000:0028	A	Network	H/W IRQ2
0000:002C	B	Communications Port 2	H/W IRQ3
0000:0030	C	Communications Port 1	H/W IRQ4

Table C2-5. Interrupt Vector Assignments *(cont.)*

Vector Table Entry	INT Number	Name	
0000:0034	D	Hard Disk	H/W IRQ5
0000:0038	E	Diskette	H/W IRQ6
0000:003C	F	Printer	H/W IRQ7
0000:0040	**10**	**EGA/VGA BIOS Video Services**	
0000:0044	11	Equipment Check	
0000:0048	12	Determine Memory Size	
0000:004C	13	Diskette/Disk	
0000:0050	14	Communications	
0000:0054	15	Cassette	
0000:0058	16	Keyboard	
0000:005C	17	Printer	
0000:0060	18	Resident BASIC	
0000:0064	19	Bootstrap	
0000:0068	1A	Time of Day	
0000:006C	1B	Keyboard Break	
0000:0070	1C	Timer Tick	
0000:0074	**1D**	**Video Initialization**	
0000:0078	1E	Diskette Parameters	
0000:007C	1F	Optional Pointer to Upper 128 CGA 8×8 characters	

Table C2-6. Interrupt Vectors Reserved for Use by DOS

Vector Table Entry	INT Number	Name
0000:0080	20	Program Terminate
0000:0084	21	Function Request
0000:0088	22	Terminate Process Exit Address
0000:008C	23	Control-C Handler Address
0000:0090	24	Critical Error Handler Address
0000:0094	25	Absolute Disk Read
0000:0098	26	Absolute Disk Write
0000:009C	27	TSR (Terminate But Stay Resident)
0000:00AA–00B8	28–2E	Reserved
0000:00BC	2F	Print Spool Control
0000:00C0–00FC	30–3F	Reserved
0000:0108	**42**	**Old BIOS Video Services**
0000:010C	**43**	**Pointer to CGA 8 × 8 Character Set**

NOTES:

1) The INT 15 interrupt service handler has an additional responsibility in systems with an E000 segment video BIOS; besides cassette service, it manages video subsystem services.

2) The complete list of interrupt numbers goes to FFh; each vector is a dword so the pointer for INT xh is stored at absolute location 4xh.

Appendix C3

BIOS Extensions

BIOS EXTENSIONS

1. INTRODUCTION

This appendix discusses the extensions to the VGA BIOS. The extensions unique to Cirrus Logic are covered in Section 2. The VESA charter is reprinted as Section 3, and the VESA extensions are covered in Section 4.

The display mode tables are in Chapter 3, “Data Book”.

2. CIRRUS LOGIC EXTENSIONS

The Cirrus Logic BIOS supports all standard VGA BIOS Interrupt 10h video service functions. In addition, the BIOS provides extensive support for various features of the Cirrus Logic VGA controller. These functions are available as extended functions under Interrupt 10h.

The standard VGA BIOS Interrupt 10h video service functions are described in Appendix C2, “VGA BIOS”.

All extended function calls preserve the CPU registers, except those used to pass information from the BIOS.

2.1 Function Summary

Table C3-1 provides an overview of the extended functions provided by the Cirrus Logic BIOS.

Table C3-1. Function Summary

AH Register	BL Register	Function
12h	80h	Inquire VGA Type
12h	81h	Inquire BIOS Version Number
12h	82h	Inquire Design Revision Code
12h	85h	Return Installed Memory
12h	9Ah	Inquire User Options
12h	A0h	Query Display Mode Availability
12h	A1h	Read Monitor Type and ID
12h	A4h	Set Monitor Type
12h	A5h	Generic Fixup Support
12h	A7h	Return Capabilities
12h	AEh	Get High Refresh
12h	AFh	Set High Refresh

2.2 Inquire VGA Type

This function provides a mechanism for software to determine the type of Cirrus Logic VGA controller, silicon revision number, and corresponding hardware capabilities. BIOS versions that do not support this family of function will preserve the input value in AL register. The VGA types of particular importance to the CL-GD5446 are in **bold** type.

Input: AH= 12h
BL= 80h

Output: AX= Controller type
0 = No extended alternate select support
1 = Reserved
2 = CL-GD510/520
3 = CL-GD610/620
4 = CL-GD5320
5 = CL-GD6410
6 = CL-GD5410
7 = CL-GD6420
8 = CL-GD6412
10h = CL-GD5401
11h = CL-GD5402
12h = CL-GD5420
13h = CL-GD5422
14h = CL-GD5424
15h = CL-GD5426
16h = CL-GD5420r1
17h = CL-GD5402r1
18h = CL-GD5428
19h = CL-GD5429
20h = CL-GD6205
21h = CL-GD6215
22h = CL-GD6225
23h = CL-GD6235
24h = CL-GD6245
31h = CL-GD5434
32h = CL-GD5430
33h = CL-GD5434 Rev. E and F
35h = CL-GD5440
36h = CL-GD5436
39h = CL-GD5446
40h = CL-GD6440
41h = CL-GD7542
50h = CL-GD5452

BL = Silicon revision number
0–7Fh= Silicon revision
80h = Silicon revision number not available

2.3 Inquire BIOS Version Number

This function provides a mechanism for software to determine the BIOS version number.

Input: AH = 12h
BL = 81h

Output: AH = Major BIOS version number
AL = Minor BIOS version number

Example: If BIOS version is 1.02, then AH is 01 and AL is 02.

2.4 Inquire Cirrus Logic Design Revision Code

This function provides a mechanism for software to determine the revision of Cirrus Logic silicon.

Input: AH = 12h
BL = 82h

Output: AL = Chip revision

2.5 Return Installed Memory

The function returns the amount of frame buffer memory present in 64K units.

Input: AH = 12h
BL = 85h

Output: AL = Amount of frame buffer memory present in 64K units.

2.6 Inquire User Options

This function returns the current status of user options. The values of the vertical frequencies and maximum vertical resolution correspond to the values defines as input for functions A4h, Set Monitor Type (Vertical).

Input: AH = 12h
BL = 9Ah

Output: AX = Contains the following options
Bits 13:0 = Reserved
Bit 14 = Vertical montype 640 × 480 frequency (VGA refresh)
Bit 15 = Reserved
BX = Reserved
CX = Contains the following options
Bit 0 = Reserved
Bits 3:1 = 1280 × 1024 vertical frequency
Bits 7:4 = Maximum Vertical Resolution
Bits 11:8 = 800 × 600 vertical frequency
Bits 15:12 = 1024 × 768 vertical frequency
DX = Bits 15:14 = 1152 vertical frequency
Bits 13:12 = 1600 vertical frequency

2.7 Query Display Mode Availability

Input:
AH= 12h
AL= Display mode number (0–7fh)
BL= A0h

Output:
AH= Bit 0
0= Display mode not supported
1= Display mode supported

DS:SI: Pointer to standard display parameters or FFFF:FFFF if standard parameters undefined for this mode
ES:DI: Pointer to supplemental display parameters or FFFF:FFFF if supplemental parameters undefined for this mode

BX= Offset to BIOS subroutine that 'fix up' the parameters pointed to by DS:SI. This routine requires ES:DI pointers to the proper supplemental display parameters.

2.8 Read Monitor ID/Type

This function uses the analog sense circuitry to detect the type of monitor attached. The digital Monitor ID pins are not used to read the monitor type. The Monitor ID returned in BH is determined by the monitor type sensed (color, monochrome, or none) and may not correspond to the actual digital Monitor ID of the current monitor. This function is used as a diagnostic function to test the monitor sense occurring during POST. The capabilities (refresh rates supported) of the monitor are determined by the parameters passed to function A4h, Set Monitor Type (Vertical).

Input:
AH = 12h
BL = A1h Read monitor ID and type from 15-pin connector

Output:
BH = Monitor ID
0Dh = IBM 8503 or equivalent
0Eh = IBM 8512/8513 or equivalent
0Fh = No monitor
00 thru 0C = reserved

BL = Monitor gender
00 = Color display
01 = Grayscale display
02 = No display

2.9 Set Monitor Type

This function sets the monitor type in terms of vertical timings. The monitor type information is used by the BIOS to determine which frequency to use when selecting an extended mode. It is also used (in conjunction with the amount of display memory available) to determine what extended modes are available. The monitor type can be read back using function 9A.

To maintain compatibility with previous Cirrus Logic BIOS releases, obsolete frequencies were not removed from this function. The appearance of any frequency in the description of this BIOS call is no guarantee that any given BIOS actually supports that frequency. In general, the trend is toward supporting higher frequencies and deleting support of lower frequencies.

NOTE: Calls to the obsolete functions 0A2h (Set Monitor Type – Horizontal) and 0A3h (Set Refresh Type) is converted into this call.

Input:

AH = 012h
BL = 0A4h
AL[3:0] = Maximum Vertical Resolution
- 000h = 480 scanlines
- 001h = 600 scanlines
- 002h = 768 scanlines
- 003h = 1024 scanlines
- 004h = 1200 scanlines
- 005h–00Fh = Reserved

AL[4] = 640 × 480 Frequency
- 0h = 60 Hz
- 1h = High Refresh: Refer to Subfunction AEh to select 72 or 75 Hz

AL[7:5] = Reserved
BH[3:0] = 800 × 600 Frequency
- 000h = 56 Hz
- 001h = 60 Hz
- 002h = 72 Hz
- 003h = 75 Hz
- 004h = 85 Hz
- 005h–00Fh = Reserved

BH[7:4] = 1024 × 768 Frequency
- 000h = 43i Hz
- 001h = 60 Hz
- 002h = 70 Hz
- 003h = 72 Hz
- 004h = 75 Hz
- 005h = 85 Hz
- 005h–00Fh = Reserved

CH[1:0] = 1152 x 864 Frequency
- 000h = 70 Hz
- 001h = 75 Hz

CH[3:2] = 1600 x 1200 Frequency
- 000h = 48i Hz

CH[7:4] = 1280 × 1024 Frequency
- 000h = 43i Hz
- 001h = 60 Hz
- 002h = 71.2 Hz (CL-GD5434 Revision E and F)
- 003h = 75 Hz (CL-GD5434 Revision E and F)
- 004h–00Fh = Reserved

CL = Reserved
DX = Reserved

2.10 Process Generic Fixup Table

Generally, the Generic Fixup table is processed at the end of each call to INT 10h function 0h, SetMode. This call is used to immediately process the Generic Fixup table without needing to perform a call to Setmode.

Input: AH = 012h
BL = 0A5h
AL = 1
Output: Nothiing

2.11 Return Chip Capabilities

This call distinguishes between the various CL-GD543X/'4X devices connected. The capabilities are returned in AL.

Input:		AH = 012h	
		BL = 0A7h	
		CX = 0000h	
Output:		AX = 1200h	Function not supported
		AX = 00XXh	Function supported
	AL	Bit 0 = 1	Always
		Bit 1 = 0	SR08 DDC Implementation
		Bit 1 = 1	Pixel Bus DDC Implementation
		Bit 2 = 1	DDC1 is disabled (or BIOS does not support it)
		Bit 3 = 1	DDC2 is disabled (or BIOS does not support it)
		BX = XXXXh	
		CX = XXXXh	
		DL = Refresh offset in Table	

2.12 Get High Refresh

This function returns the vertical refresh rate of the 640 × 480 high refresh modes.

Input: AH = 012h
BL = 0AEh
Output: AL = Bit 0 indicates 640 × 480 high-refresh rate
0 = 72 Hz
1 = 75 Hz

2.13 Set High Refresh

This function sets the vertical refresh rate of the 640 × 480 high-refresh modes. The application must call this function after calling Set Monitor Type (Subfunction A4h), specifying 640 × 480 high-refresh mode.

Input: AH = 012h
BL = 0AFh
AL = Bit 0 indicates 640 × 480 high refresh rate
0 = 72 Hz
1 = 75 Hz

3. VESA® SUPER VGA STANDARD

Video Electronics Standards Association
2150 N. First Street, Suite 440
San Jose, CA 95131-2020
TEL: (408) 435-0333 FAX: (408) 435-8225

Purpose

To standardize a common software interface to Super VGA display adapters to provide simplified software application access to advanced VGA products.

Summary

The standard provides a set of functions that an application program can use to:

- Obtain information about the capabilities and characteristics of specific Super VGA implementations, and
- Control the operation of such hardware in terms of display mode initialization and display memory access. The functions are provided as an extension to the VGA BIOS video services, accessed through INT 10h.

3.1 Goals and Objectives

The purpose of the VBE (VESA VGA BIOS extension) is to provide a common software interface for developers to design applications successfully on widely disparate architectures. Because it is the common software interface to Super VGA graphics products, the primary objective is to enable application and system software to adapt to and exploit the wide range of features available in these VGA extensions.

The VBE attempts to address the following two main issues to:

- Return information about the display environment to the application, and
- Assist the application in initializing and programming the hardware.

3.1.1 Video Environment Information

The VBE provides several functions to return information about the video environment. These functions return system-level information, as well as display-mode-specific details. Function 00h returns general system-level information, including an OEM identification string. The function also returns a pointer to the supported display modes. Function 01h, used by the application, can obtain information about each supported display mode. Function 03h returns the current display mode.

3.1.2 Programming Support

The VBE provides several functions to interface to the different Super VGA hardware implementations. The most important of these is function 02h, Set Super VGA Display Mode. This function isolates the application from the tedious and complicated task of setting up a display mode. Function 05h provides an interface to the underlying memory-mapping hardware.

Function 04h enables an application to save and restore a Super VGA state without determining specifics of the particular implementation.

3.1.3 Compatibility

The primary design objective of the VBE is to preserve maximum compatibility to the standard VGA environment. In no way should the BIOS extensions compromise compatibility or performance. Another related concern is minimizing the changes required to an existing VGA BIOS. RAM, as well as ROM-based implementations of the BIOS extension should be possible.

3.2 Standard VGA BIOS

A primary design goal with the VBE is to minimize the effects on the standard VGA BIOS. Standard VGA BIOS functions should need to be modified as little as possible. This is important since ROM, as well as RAM-based versions of the extension can be implemented.

Two standard VGA BIOS functions are affected by the VESA extension: function 00h (Set Display Mode) and function 0Fh (Read Current Display State). VESA-aware applications do not set the display mode using VGA BIOS function 00h, nor do such applications use VGA BIOS function 0Fh. VESA BIOS functions 02h (Set Super VGA Mode) and 03h (Get Super VGA Mode) are used instead.

VESA-unaware applications (such as, old Pop-up programs and other TSRs, or the CLS command of MS-DOS), might use VGA BIOS function 0Fh to get the present display mode. Later it can call VGA BIOS function 00h to restore/reinitialize the old display mode.

To make such applications run properly, VESA recommends that whatever value is returned by VGA BIOS function 0Fh (it is the responsibility of the OEM to define this number), it can be used to reinitialize the display mode through VGA BIOS function 00h. Thus, the BIOS should record the last Super VGA mode in effect.

It is recommended, but not mandatory, to support output functions (such as TTY-output, scroll, set pixel, and so on) in Super VGA modes. If the BIOS extension does not support such output functions, clear bit D2 (Output functions supported) of the ModeAttributes field (returned by VESA BIOS function 01h).

3.3 Super VGA Mode Numbers

Standard VGA mode numbers are 7-bits wide and ranges from 00h–13h. OEMs have defined extended display modes in the range 14h–7Fh. Values in the range 80h–FFh cannot be used, since VGA BIOS function 00h (Set Display Mode) interprets bit 7 as a flag to clear/not clear display memory.

Due to the limitations of 7-bit mode numbers, VESA Display mode numbers are 15-bits-wide. To initialize a Super VGA mode, its number is passed in the BX register to VESA BIOS function 02h (Set Super VGA Mode).

The format of VESA mode numbers is as follows:

D0-D8 =	Mode number If D8 = 0, this is not a VESA-defined mode If D8 = 1, this is a VESA-defined mode
D9-D14 =	Reserved by VESA for future expansion (= 0)
D15 =	Reserved (= 0)

4. EXTENDED VESA® BIOS FUNCTIONS

Several new BIOS calls are defined to support Super VGA modes. For maximum compatibility with the standard VGA BIOS, these calls are grouped under one function number. This number is passed in the AH register to the INT 10h handler.

The designated Super VGA extended function number is 4Fh. This function number is presently unused in most, if not all, VGA BIOS implementations. A standard VGA BIOS performs no action when function call 4F is made. Super VGA Standard v.S911022 defines subfunctions 00H–08H. Subfunction numbers 09H–0FFH are reserved for future use.

4.1 Status Information

Every function returns status information in the AX register. The format of the status word is:

AL =	4Fh:	Function is supported
AL ! =	4Fh:	Function is not supported
AH =	00h:	Function call successful
AH =	01h:	Function call failed

Software should consider a non-zero value in the AH register as a general failure condition. New error codes might be defined in later versions of the VBE.

4.2 Function 00h — Return Super VGA Information

The purpose of this function is to provide information to the calling program about the general capabilities of the Super VGA environment. The function fills an information block structure at the address specified by the caller. The information block size is 256 bytes.

Input: AH = 4Fh Super VGA support
AL= 00h Return Super VGA information
ES:DI= Pointer to buffer

Output: AX = Status
All other registers are preserved.

The information block has the following structure:

```
VgaInfoBlock struc
VESASignature db          'VESA'      ; 4 signature bytes
VESAVersion   dw          ?           ; VESA version number
OEMStringPtr  dd          ?           ; Pointer to OEM string
Capabilities  db          4 dup (?)   ; capabilities of the display
environment
DisplayModePtrdd          ?           ; pointer to supported Super VGA
Modes
TotalMemory   dw          ?           ; Number of 64kb memory blocks on
board
Reserved      db          236 dup (?) ; Remainder of VgaInfoBlock
VgaInfoBlock ends
```

VESASignature

The VESASignature field contains the characters ‘VESA’ if this is a valid block.

VESAVersion

The VESAVersion is a binary field specifying what level of the VESA standard the Super VGA BIOS conforms to. The higher byte specifies the major version number. The lower byte specifies the minor version number. The current VESA version number is 1.2. Applications written to use the features of a specific version of the VBE are guaranteed to work in later versions. The VBE is fully upwards compatible.

OENStringPtr

The OEMStringPtr is a far pointer to a null-terminated OEM-defined string. The string can used to identify the chip, display subsystem board, memory configuration, and so on, to hardware-specific display drivers. There are no restrictions on the format of the string.

Capabilities

The Capabilities field describes what general features are supported in the display environment. The bits are defined as:

D0 = DAC is switchable
0 = DAC is fixed width, with 6-bits per primary color
1 = DAC width is switchable
D1–31 = Reserved

DisplayModePtr

The DisplayModePtr points to a list of supported Super VGA (VESA-defined as well as OEM-specific) mode numbers. Each mode number occupies one word (16 bits). The list of mode numbers is terminated by –1 (0FFFFh). Please refer to Section 3.3 for a description of VESA mode numbers. Depending on the specific implementation, the pointer could point into either ROM or RAM. Either the list is a static string stored in ROM, or the list is generated at run-time in the information block (see above) in RAM. It is the applications responsibility to verify the current availability of any mode returned by this function through the Return Super VGA mode information (function 1) call. Some of the returned modes may not be available due to the display subsystem boards current memory and monitor configuration.

TotalMemory

The TotalMemory field indicates the amount of memory installed on the VGA board. Its value represents the number of 64-Kbyte blocks of memory currently installed.

4.3 Function 01h — Return Super VGA Mode Information

This function returns information about a specific Super VGA Display mode that was returned by function 0. The function fills a mode information block structure at the address specified by the caller. The mode information block size is a maximum 256 bytes.

Some information provided by this function is implicitly defined by the VESA mode number. However, some Super VGA implementations might support other display modes than those defined by VESA. To provide access to these modes, this function also returns various other information about the mode.

Input:	AH =	4Fh	Super VGA support
	AL=	01h	Return Super VGA Mode information
	CX =		Super VGA Display Mode number
	ES:DI =		Pointer to 256 byte buffer
Output:	AX =		Status
	All other registers are preserved		

ModeInfoBlock

The mode information block has the following structure:

```
ModeInfoBlock   struc
; mandatory information
ModeAttributesdw         ?           ; mode attributes
WinAAttributesdb         ?           ; window A attributes
WinBAttributesdb         ?           ; window B attributes
WinGranularitydw         ?           ; window granularity
WinSize       dw         ?           ; window size
WinASegment   dw         ?           ; window A start segment
WinBSegment   dw         ?           ; window B start segment
WinFuncPtr    dd         ?           ; pointer to window function
BytesPerScanLinedw       ?           ; bytes per scanline extended
information
XResolution   dw         ?           ; horizontal resolution
YResolution   dw         ?           ; vertical resolution
XCharSize     db         ?           ; character cell width
YCharSize     db         ?           ; character cell height
NumberOfPlanesdb         ?           ; number of memory planes
BitsPerPixel  db         ?           ; bits per pixel
NumberOfBanks db         ?           ; number of banks
MemoryModel   db         ?           ; memory model type
BankSize      db         ?           ; bank size in kb
NumberOfImagePagesdb     ?           ; Number of Images
Reserved      db         1           ; reserved for page function
RedMaskSize   db         ?           ;size of direct color red mask in
bits
RedFieldPositiondb       ?           ;bit position of lsb of red mask
GreenMaskSize db         ?           ;size of direct color green mask in
bits
GreenFieldPositiondb     ?           ;bit position of lsb of green mask
BlueMaskSize  db         ?           ;size of direct color blue mask in
bits
```



```
BlueFieldPositiondb      ?           ;bit position of lsb of blue mask
RsvdMaskSize  db         ?           ;size of direct color reserved mask in
bits
RsvdFieldPositiondb      ?           ;bit position of lsb of reserved mask
DirectColorModeInfo db   ?           ;Direct Color Mode attributes
Reserved     db          216 dup (?) ; remainder of ModeInfoBlock
ModeInfoBlock   ends
```

ModeAttributes

The ModeAttributes field describes certain important characteristics of the display mode. Bit D0 specifies whether this mode can be initialized in the present display configuration. This bit can block access to a display mode if it requires a certain monitor type, and the monitor is presently not connected. Bit D1 specifies whether extended mode information is available. This information is required in VBE v1.2 and later. Bit D2 indicates whether the BIOS supports output functions (such as, TTY output, scroll, pixel output, and so on) in this mode (it is recommended, but not mandatory, that the BIOS supports all output functions).

The field is defined as:

D0= Mode supported in hardware
- 0 = Mode not supported in hardware
- 1 = Mode supported in hardware

D1 = Extended information available
- 0 = Extended mode information not available
- 1 = Extended mode information available

D2 = Output functions supported by BIOS
- 0 = Output functions not supported by BIOS
- 1 = Output functions supported by BIOS

D3 = Monochrome/Color mode (see note below)
- 0 = Monochrome mode
- 1 = Color mode

D4 = Mode type
- 0 = Text mode
- 1 = Graphics mode

D5–D15 = Reserved

NOTE: Monochrome modes have their CRTC address at 3B4h. Color modes have their CRTC address at 3D4h. Monochrome modes have attributes where only bit 3 (video) and bit 4 (intensity) of the attribute controller output are significant. Therefore, monochrome text modes have attributes of: off, video, high intensity, blink, and so on. Monochrome graphics modes are two-plane graphics modes and have attributes of off, video, high intensity, and blink. Extended two-color modes that have their CRTC address at 3D4h, are color modes with 1-bit-per-pixel and one plane. The standard VGA modes, 06h and 11h are classified as Color modes, while the standard VGA modes 07h and 0fh are classified as Monochrome modes.

BytesPerScanline

The BytesPerScanline field specifies how many bytes each logical scanline consists of. The logical scanline could be equal to or larger than the displayed scanline.

WinAttributes (A and B)

The WinAAttributes and WinBAttributes describe the characteristics of the CPU windowing scheme such as whether the windows exist and are read/writable, as follows:

D0 = Window supported
- 0 = Window is not supported
- 1 = Window is supported

D1 = Window readable
- 0 = Window is not readable
- 1 = Window is readable

D2 = Window writable
- 0 = Window is not writable
- 1 = Window is writable

D3–D7 = Reserved

If neither window is supported (bit D0 = 0), then an application can assume that window paging is not supported, and that the display memory buffer resides at the CPU address appropriate for the MemoryModel of the mode.

WinGranularity

WinGranularity specifies the smallest boundary, in Kbytes, on which the window can be placed in the display memory. If WinGranularity = 0, CPU display memory windowing is not supported.

WinSize

The WinSize specifies the size of the window in Kbytes.

WinSegment (A and B)

The WinASegment and WinBSegment addresses specify the segment addresses where the windows are located in the CPU address space.

WinFuncPtr

The WinFuncPtr specifies the address of the CPU display memory windowing function. The windowing function can be invoked either through VESA BIOS function 05h, or by calling the function directly. A direct call provides faster access to the Hardware Paging registers (rather than using the INT 10h call) and is intended to be used by high-performance applications. If WinFuncPtr is NULL (0000:0000) then CPU display memory windowing is not supported.

Resolution (X and Y)

The XResolution and YResolution specify the width and height of the display mode. In graphics modes, this resolution is in pixel units. In Text modes this resolution is in character units. Note that text mode resolutions, in pixel units, can be obtained by multiplying XResolution and YResolution by the cell width and height, if the extended information is present.

CharSize (X and Y)

The XCharSize and YCharSize specify the size of the character cell in pixels.

NumberOfPlanes

The NumberOfPlanes field specifies the number of memory planes available to software in that mode. For standard 16-color VGA graphics, this is set to four. For standard Packed-pixel modes, the field is set to '1'.

BitsPerPixel

The BitsPerPixel field specifies the total number of bits that define the color of one pixel. For example, a standard VGA Four-plane, 16-color Graphics mode has a four in this field, and a packed-pixel 256-color Graphics mode specify an eight in this field. The number of bits-per-pixel-per-plane can normally be derived by dividing the BitsPerPixel field by the NumberOf-Planes field.

MemoryModel

The MemoryModel field specifies the general type of memory organization used in this mode. The following models are defined:

00h = Text Mode
01h = CGA graphics
02h = Hercules graphics
03h = Four-plane planar
04h = Packed pixel
05h = Non-chain4, 256 color
06h = Direct Color
07h = YUV
08h–0fh = Reserved, to be defined by VESA
10h–ffh = To be defined by OEM

In v1.1 and earlier versions of the VESA Super VGA BIOS extension, Direct Color 1:5:5:5, 8:8:8, and 8:8:8:8 are defined as a Packed-Pixel model with 16-, 24-, and 32-bpp, respectively. In v1.2 and later, it is recommended that Direct-color modes use the Direct-color Memory-Model and use the MaskSize and FieldPosition fields of the ModeInfoBlock to describe the pixel format. BitsPerPixel is always defined to be the total size of the pixel, in bits.

NumberOfBanks

The NumberOfBanks is the number of banks into which the scanlines are grouped. The remainder from dividing the scanline number by the number of banks, is the bank that contains the scanline and the quotient is the scanline number within the bank. For example, CGA graphics modes have two banks and Hercules Graphics mode has four banks. For modes that don't have scanline banks (such as VGA Modes 0Dh–13h), set this field to '1'.

BankSize

The BankSize field specifies the size of a bank (group of scanlines) in 1-Kbyte units. For CGA and Hercules graphics modes this is an eight, as each bank is 8192 bytes in length. For modes that do not have scanline banks (such as, VGA modes 0Dh–13h), set this field to '0'.

NumberOfImagePages

The NumberOfImagePages field specifies the number of additional complete display images that fit into the VGA memory at one time in this mode. If this field is a non-zero, the application can load more than one image into the VGA memory and flip the display between the images.

Reserved

The Reserved field is defined to support a future VBE feature and in this version is always set to '1'.

MaskSize (Red, Green, Blue and Rsvd)

The RedMaskSize, GreenMaskSize, BlueMaskSize, and RsvdMaskSize fields define the size, in bits, of the red, green, and blue components of a Direct-color pixel. A bit mask can be constructed from the MaskSize fields using simple shift arithmetic. Example MaskSize values for Direct-color 5:6:5 mode are 5, 6, 5, and 0 for the red, green, blue, and reserved fields, respectively. In modes using a MemoryModel that does not have pixels with component fields, set the MaskSize fields to '0'.

FieldPosition (Red, Green, Blue and Rsvd)

The RedFieldPosition, GreenFieldPosition, BlueFieldPosition, and RsvdFieldPosition fields define the bit position within the Direct-color pixel or YUV pixel of the least-significant bit of the respective color component. A color value can be aligned with its pixel field by left-shifting the value by the FieldPosition. Example FieldPosition values for Direct-color 5:6:5 mode are 11, 5, 0, and 0 for the red, green, blue, and reserved fields, respectively. In modes using a MemoryModel that does not have pixels with component fields, set the FieldPosition fields to '0'.

DirectColorModeInfo

The DirectColorModeInfo field describes important characteristics of Direct-color modes. Bit D0 specifies whether the color ramp of the DAC is fixed or programmable. If the color ramp is fixed, then it cannot be changed. If the color ramp is programmable, it is assumed that the red, green, and blue LUTs can be loaded using a standard VGA DAC Color registers BIOS call (AX = 1012h). Bit D1 specifies whether the Rsvd field of the Direct-color pixel can be used by the application or is reserved, thus unusable.

D0 = Color ramp is fixed/programmable
- 0 = Color ramp is fixed
- 1 = Color ramp is programmable

D1 = Rsrvd field is usable/reserved
- 0 = Rsvd field is reserved
- 1 = Rsvd field is usable by the application

MapFuncAddr

The MapFuncAddr specifies the address of the mapping function. The mapping function can be invoked either through VESA BIOS function 06h, or by calling the function directly. A direct call provides a faster memory mapping than using INT 10h, and is intended to be used by high-performance applications.

NOTE: v1.1 and later VBEs zero-out all unused fields in the ModeInfoBlock, always returning exactly 256 bytes. This facilitates upward compatibility with future versions of the standard, as any newly-added fields are designed so that zero values indicate nominal defaults or non-implementation of optional features. (For example, a field containing a bit-mask of extended capabilities reflects the absence of all such capabilities.) Applications that need to be backwards-compatible to v1.0 VBEs should pre-initialize the 256-byte buffer before calling Return Super VGA mode information.

4.4 Function 02h — Set Super VGA Display Mode

This function initializes a display mode. The BX register contains the display mode number. The format of VESA mode numbers is described in Section 2 on page C3-2. If the mode cannot be set, the BIOS should leave the display environment unchanged and return a failure error code.

Input:
AH = 4Fh Super VGA support
AL = 02h Set Super VGA Display mode
BX = Display mode
D0–D14 = Display mode number
D15 = Clear memory flag
0 = Clear display memory
1 = Don't clear display memory

Output:
AX = Status
All other registers are preserved.

4.5 Function 03h — Return Current Display Mode

This function returns the current display mode in the BX register. The format of the VESA Display mode numbers is described in Section 2 on page C3-2.

Input:
AH = 4Fh Super VGA support
AL = 03h Return current display mode

Output:
AX = Status
BX = Current display mode number
All other registers are preserved.

NOTE: In a standard VGA BIOS, function 0Fh (Read Current Display State) returns the current display mode in the AL register. In D7 of the AL register, it also returns the status of the Memory Clear bit (D7 of 40:87). This bit is set if the mode was set without clearing memory. In this Super VGA function, the Memory Clear bit is not returned in the BX register, since the purpose of the function is to return the display mode only. If an application must obtain the Memory Clear bit, it should call VGA BIOS function Fh.

4.6 Function 04h — Save/Restore Super VGA Display State

These functions provide a mechanism to save and restore the Super VGA display state. The functions are a superset of the three subfunctions under standard the VGA BIOS function 1Ch (Save/Restore Display State). The complete Super VGA display state (except display memory) should be saveable/restorable by setting the requested states mask (in the CX register) to 000Fh.

Input:
AH = 4Fh Super VGA support
AL = 04h Save/Restore Super VGA display state
DL = 00h Return save/restore state buffer size
CX = Requested states
D0 = Save/restore display hardware state
D1 = Save/restore display BIOS data state
D2 = Save/restore display DAC state
D3 = Save/restore Super VGA state

Output:
AX = Status
BX = Number of 64-byte blocks to hold the state buffer
All other registers are preserved.

Input:
AX = 4Fh Super VGA support
AL = 04h Save/Restore Super VGA display state
DL = 01h Save Super VGA display state
CX = Requested states (see above)
ES:BX = Pointer to buffer

Output:
AX = Status
All other registers are preserved.

Input:
AH = 4Fh Super VGA support
AL = 04h Save/Restore Super VGA display state
DL = 02h Restore Super VGA display state
CX = Requested states (see above)
ES:BX = Pointer to buffer

Output:
AX = Status
All other registers are preserved.

4.7 Function 05h — CPU Display Memory Window Control

This function sets or gets the position of the specified window in the display memory. The function allows direct access to the Hardware Paging registers. To use this function properly, the software should use VESA BIOS function 01h (Return Super VGA Mode Information) to determine the size, location, and granularity of the windows.

Input:
AH = 4Fh Super VGA support
AL = 05h Super VGA display memory window control
BH = 00h Select super VGA display memory window
BL = Window number
0 = Window A
1 = Window B
DX = Window position in display memory
(in window granularity units)

Output:
AX = Status
See note below.

Input:
AH = 4Fh Super VGA support
AL = 05h Super VGA display memory window control
BH = 01h Return super VGA display memory window
BL = Window number
0 = Window A

		1 =	Window B
Output:	AX =		Status
	DX =		Window position in display memory (in window granularity units)
	See note below.		

NOTE: This function is also directly accessible through a far call from the application. The address of the BIOS function can be obtained by using VESA BIOS function 01h, Return Super VGA Mode Information. A field in the ModeInfoBlock contains the address of this function. Note that this function may be different among display modes in a particular BIOS implementation so obtain the function pointer after each set mode.

In the far call version, no status information is returned to the application. Also, in the far call version, the AX and DX registers are destroyed. Therefore if AX and/or DX register must be preserved; the application must preserve these registers prior to making the far call.

The application must load the input arguments in the BH, BL, and DX registers (for set window), but does not need to load either the AH or AL register to use the far call version of this function.

4.8 Function 06h — Set/Get Logical Scanline Length

This function sets or gets the length of a logical scanline. This function allows an application to set up a logical display memory buffer that is wider than the displayed area. Function 07h then allows the application to set the starting position that is to be displayed.

Input:	AH =	4fh	Super VGA support
	AL =	06h	Logical scanline length
	BL =	00h	Select scanline length
	CX =		Desired width in pixels
Output:	AX =		Status
	BX =		Bytes per scanline
	CX =		Actual pixels per scanline
	DX =		Maximum number of scanlines
Input:	AH =	4fh	Super VGA support
	AL =	06h	Logical Scanline length
	BL =	01h	Return scanline length
Output:	AX =		Status
	BX =		Bytes per scanline
	CX =		Actual pixels per scanline
	DX =		Maximum number of scanlines

NOTE: The desired width in pixels may not be achievable because of VGA hardware considerations. The next larger value is selected that can accommodate the desired number of pixels, and the actual number of pixels is returned in the CX register. The BX register returns a value that, when added to a pointer into display memory, points to the next scanline. For example, in mode 13h this is 320, but in mode 12h this is 80. The DX register returns the number of logical scanlines based upon the new scanline length and the total memory installed and usable in this display mode. This function is also valid in Text modes. In Text modes, the application determines the current character cell width through VESA function 1 (or VGA BIOS function 1BH), multiply the cell width by the desired number of characters per line, and pass that value in register CX.

4.9 Function 07h — Set/Get Display Start

This function selects the pixel to be displayed in the upper-left corner of the display from the logical page. This function can to pan and scroll around logical screens that are larger than the displayed screen. This function can also rapidly switch between two different displayed screens for double-buffered animation effects.

Input:	AH =	4fh	Super VGA support
	AL =	07h	Display start control
	BH =	00h	Reserved and must be ‘0’
	BL =	00h	
	CX =		First displayed pixel in scanline
	DX =		First displayed scanline
Output:	AX =	Status	
Input:	AH =	4fh	Super VGA support
	AL =	07h	Display start control
	BL =	01h	Return display start
Output:	AX =		Status
	BH =		00h – reserved and is ‘0’
	CX =		First displayed pixel in scanline
	DX =		First displayed scanline

NOTE: This function is also valid in Text modes. In Text modes, the application should find out the current character cell width through VESA function 1 (or VGA BIOS function 1BH), multiply the cell width by the desired starting character column, and pass that value in register CX. The application should also multiply the current character cell height by the desired starting character row, and pass that value in register DX.

4.10 Function 08h — Set/Get DAC Palette Control

This function queries and selects the operating mode of the DAC palette. Some DACs are configurable to provide 6, 8, or more bits of color definition per red, green, and blue primary color. The DAC palette width is assumed to be reset to standard VGA 6 bits-per-primary during a standard or VESA Set Super VGA mode (AX = 4F02h) call.

Input:	AH =	4fh	Super VGA support
	AL=	08h	Set/get DAC palette control
	BL =	00h	Set DAC palette width
	BH =		Desired number of bits of color per primary (Standard VGA = 6)

Output:	AX =		Status
	BH =		Current number of bits of color per primary (Standard VGA = 6)
Input:	AH =	4fh	Super VGA support
	AL =	08h	Set/get DAC palette control
	BL =	01h	Get DAC palette width
Output:	AX =		Status
	BH =		Current number of bits of color per primary (Standard VGA = 6)

An application can determine if DAC switching is available by querying bit D0 of the Capabilities field in the VgaInfoBlock structure returned by VESA Return Super VGA Information (AX=4F00h). The application can then attempt to set the DAC palette width to the requested value. If the Super VGA is not capable of selecting the requested palette width, then the next lower value that the Super VGA is capable of selecting. The resulting palette width is returned.

4.10.1 Function 10h — Display Power Management Extensions

The following three functions are defined by VESA as a proposal for VBE/PM, which is a software interface to DPMS: Report Display Power Management Capabilities, Set Power State, and Get Power State. These functions are included to specify the Cirrus Logic implementation of display power management. For questions regarding VESA, VBE/PM, or DPMS, please refer to the Video Electronics Standards Association, the address is in Section 3 on page C3-8.

Report VBE/PM Capabilities

Input:	AH =	4fh Super VGA support
	AL =	10f VBE/PM services
	BL =	00h Report VBE/PM capabilities
	ES:DI=	Null pointer, must be 0000:0000 in version 1.0
Output:	AX =	Status
	BH =	Power saving state signals support by the controller: 1 = supported, 0 = not supported bits 7:4 reserved bit 3 REDUCED ON (not supported by DPMS 1.0) bit 2 OFF bit 1 SUSPEND bit 0 STAND BY
	BL =	VBE/PM version number bits 7:4 Major version number bits 3:0 Minor version number

Set Display Power State

Input:	AH =	4fh Super VGA support
	AL =	10h VBE/PM services
	BL =	01h Set display power state
	BH =	Requested power state 00h ON 01h STAND BY 02h SUSPEND

04h OFF
08h REDUCED ON (not supported by DPMS 1.0)

Output: AX = Status
BH= Unchanged

Get Display Power State

Input: AH = 4fh Super VGA support
AL = 10h VBE/PM services
BL = 02h Get display power state

Output: AX = Status
BH = Display power state
00h ON
01h STAND BY
02h SUSPEND
04h OFF
08h REDUCED ON (not supported by DPMS 1.0)
bits 7:4 are reserved and
should be ignored to ensure upward compatibility.

4.11 Function 15h — Display Identification Extensions

The VESA VBE subfunction 15h implements the VBE/DDC services. The VBE/DDC services are defined below and are not included in the VBE standard documentation. For questions regarding VESA, or VBE/DDC, please refer to the Video Electronics Standards Association the address is in Section 3 on page C3-8.

Report VBE/DDC Capabilities

Input: AH = 4fh VESA extension
AL = 15h VBE/DDC services
BL = 00h Report DDC capabilities
CX = 00h Controller unit number (00 = primary controller)
ES:DI = Null pointer, must be 0000:0000 in version 1.0

Output: AX = Status
BH = Approx time in seconds, rounded up, to transfer one EDID block (128 bytes)
BL = DDC level supported by both the display and controller
bit 0 = 0 DDC1 not supported
bit 0 = 1 DDC1 supported
bit 1 = 0 DDC2 not supported
bit 1 = 1 DDC2 supported
bit 2 = 0 Screen not blanked during data transfer
bit 2 = 1 Screen blanked during data transfer (This refers to the behavior of the controller and software
CX = Unchanged
ES:DI = Unchanged
All other register contents may be destroyed

Read EDID

Input:	AH =	4fh	VESA extension
	AL =	15h	VBE/DDC services
	BL =	01h	Read EDID
	CX =	00h	Controller unit number (00 = primary controller)
	DX =	00h	EDID block number. Zero is the only valid value in version 1.0
	ES:DI =	Pointer to area in which the EDID block (128 bytes) shall be returned	
Output:	AX =	Status	
	BH =	Unchanged	
	CX =	Unchanged	
	ES:DI =	Pointer to area in which the EDID block is returned	
	All other register contents may be destroyed		

NOTE: Subfunction 02 (Read VDIF Block) is not supported.

Appendix D1

D

Cirrus Logic Bulletin Board Service, FTP, and WWW

CIRRUS LOGIC BBS, FTP, AND WWW

1. INTRODUCTION

Cirrus Logic maintains a BBS (bulletin board system) 24 hours a day for customers to obtain up-to-date files and information. For the CL-GD5446, the BBS allows access to utilities, schematics, and software upgrades.

Cirrus Logic strictly controls access to the BBS. All downloadable files are monitored by Cirrus Logic. Customers cannot upload files to any publicly downloadable area.

To download files, follow the procedure below. If you require access to more restricted files or to exchange files with Cirrus Logic personnel on a regular basis, contact your Cirrus Logic representative to obtain expanded access privileges.

1. Set your communication parameters as follows:

- **a)** 8 data bits
- **b)** No parity
- **c)** 1 stop bit
- **d)** Baud rate up to 14,400 bps

2. Dial the Cirrus Logic BBS at (510) 440-9080.

3. When connected, do one of the following:

- **a)** Enter your name and password. First-time users can establish an account by entering a name and password and completing a questionnaire.
- **b)** Or, follow the instructions to log on as a 'guest'.

4. Select [J], Join Product Area.

5. Select 7, 5430/'34/'36/'40/'46.

6. Select [F], File Menu.

7. You may now choose among the options.

Many BBS files are compressed in a 'zipped' file format (using PKZIP.EXE v2.04G). These files have the .ZIP suffix. It is necessary to be uncompress these files after downloading using PKUNZIP.EXE. If you do not have this 'unzip' utility, you may download it in self-extracting format from the Cirrus Logic BBS. From any area, download the file called PKZ204G.EXE.

The software running on this bulletin board is Wildcat!™ IM, written by Mustang Software®, Inc. Cirrus Logic operates eight remote nodes and one local node to the BBS.

2. FIRST-TIME LOG ON

Upon connection with the bulletin board, a name and password are requested for a guest account. If the user is merely downloading drivers, demonstration programs, or utilities, it is best to use the guest account as it allows access to most files.

If schematic diagrams or BIOS files are needed or to upload files, it is required that you log on with your name (or company name). The system prompts a password entry and requires that a questionnaire be completed. When complete, the name and password selected is added to the user database and access is the same as a guest log on. Note your log in name and password for future reference.

3. UPGRADED ACCESS

If requiring access to BIOS image or schematic diagram files or if exchanging files with Cirrus Logic personnel on a regular basis, an account upgrade is necessary. Account upgrades are handled through your contact at Cirrus Logic (usually through a sales office). Three to five working days are required for the request to be processed after being received at Cirrus Logic in Fremont, California.

The General Public Messages area contains non-specific programs, such as PKWare and ID_CHIP. The upload areas also reside in the General Public Messages area. The upload areas are readable only by Cirrus Logic employees.

The other product areas are connected to five file areas each. These file areas are listed in Table D1-1.

Table D1-1. File Areas within Product Areas

File Area	Note	Download Access
Drivers	Driver disks	All
Demonstrations	BitBLT demo, slide viewers, and so on	All
Utilities	for example, CLMODE	All
BIOS	BIOS images, MFGTST	Integrator, OEM
Schematics	OrCAD® schematic diagrams	OEM

3.1 Using the FTP Server

In addition to the BBS, Cirrus Logic maintains an anonymous FTP site on the internet. The address is `ftp.cirrus.com`. Any password may be used to log on as anonymous or FTP.

NOTE: The FTP server is limited to released software drivers. BIOS, schematics, and beta software is available only on the BBS and is restricted to licensed OEMs.

When you log into the FTP site, you will be in the FTP directory. Change directories to `/pub/support`. This the main directory of the FTP site.

In the `support` directory, a document named `ftp_contents.doc` shows the FTP file names and contents. Like the BBS, the FTP site is arranged by device. The `support` directory provides access to the desktop, laptop, modem, PC Card, and SIO areas. Within these directories are the chipset subdirectories. Within these subdirectories are software files.

3.2 Web Access

To use a web browser to access the above files, go to the Cirrus Logic web site at:

http://www.cirrus.com/prodtech/

Or, to directly access the ftp site, type the location:

ftp://ftp.cirrus.com/pub/support/desktop/5436-46.html

Appendix E1

E

Glossary and Bibliography

GLOSSARY

AccuPak™: A video compression method proprietary to Cirrus Logic, Inc. See Chapter 9, "Programming Notes".

Add-in Card, Adapter Card: A circuit board that plugs into a computer motherboard and connects to an external device, such as a display monitor or storage subsystem.

AN: An acronym for alpha numeric. Only patterns that are defined in the font tables can be displayed. Information can usually be displayed more quickly than with APA images because fewer bits need to be manipulated by the software. These modes are also referred to as Text modes.

APA: An acronym for all points addressable. Each pixel on the screen is individually programmable. Any pattern (subject to the resolution of the system) can be displayed. This typically requires that one to two orders of magnitude more information be manipulated than is the case with an AN system. These modes are also referred to as Graphics modes. Nearly all extended modes are APA.

Analog Interface: An interface between a display controller and a display in which pixel colors are determined by the voltage levels on three output lines (RGB). Theoretically, an unlimited number of colors can be supported by this method (24 bits per pixel allows 16,777,216). The voltage level on any line varies between zero volts (for black) to about 700 millivolts (for maximum brightness). The lines are typically terminated in 75 Ω at the monitor end and 150 or 75 Ω at the graphics controller end. In the IBM world, the analog interface is usually mechanized with a 15-pin, three-row connector (DB15).

Analog Monitor: A display monitor that uses an analog interface. In IBM terms, commonly known as a PS/2 monitor, or a VGA monitor when used in conjunction with a VGA controller.

Analog: A signal that can assume intermediate levels between on and off. Contrast with digital.

ASCII: American Standard Code for Information Interchange. This is a 7-bit code that encodes alphanumeric information. In the IBM-compatible world, this is expanded to 8 bits to encode a total of 256 (AN) alphanumeric and control characters.

AUTOEXEC.BAT: A file to direct a series of activities occurring during system boot-up.

Auto-Monitor Detect: A feature of Cirrus Logic VGA controllers and BIOS that sense the type of monitor connected. This uses a scheme that involves the use of comparators to sense the terminations present on the RGB lines. This is being replaced with DDC2B. See also **DDC**.

BIOS-Level Compatibility: With regard to a VGA subsystem, this means that the BIOS supplied is in compliance with the IBM VGA Standard. This is the minimum level of compatibility necessary to accommodate the majority of standard applications.

BIOS: An acronym for basic input output system. In IBM-compatible personal computers, this is a set of ROM-based firmware routines that control the resources of the system and make them available to application programs in an orderly manner. These routines provide basic input/output services for the operating system and for applications programs that use interrupts to call them. Also called ROM BIOS. The Cirrus Logic BIOS is written in 80386/486 Assembly Language.

Bit: A binary digit. A single piece of information: on or off, 0 or 1, high or low, closed or open, up or down, in or out, alive or dead, black or white.

BitBLT (BLT), Bit Boundary Block Transfer: A graphics operation that moves a rectangle of data from one area of display memory to another, or moves data from system memory to display memory. Graphics controllers frequently include varying degrees of hardware to help speed BitBLT operations.

Bitmap: A rectangular array of locations, each associated with a location (pixel) on a monitor. The contents of each location determines the color of the pixel.

Block Diagram: A diagram where blocks represent components or subsystems of a system. Usually the blocks are connected with lines indicating data or control flow.

Byte: A group of 8 bits addressed as a unit. Can take any of 256 (2^8) values.

CAS, Column Address Strobe: One of the DRAM control signals.

CCIR: International Radio Consultive Committee.

CGA, Color Graphics Adapter: This was the first color adapter available for the IBM personal computer. It has low resolution, both spatial and color. While CGA is generally considered obsolete, the VGA standard includes the display modes originally designed for CGA.

Character Cell Matrix: In Text mode, the area of display used to display one character. On the VGA, character cells are either 8-, 9-, 12-, or 16-pixels-wide and usually are either 8-, 14-, or 16-pixels-high.

Character Clock: This clock is generated by dividing the VCLK by either eight or nine. The monitor timing signals (HSYNC, VSYNC) are derived by dividing the character clock.

Chroma Key: The CL-GD5446 can compare the Y, U, and V values of video pixels to upper and lower limits to determine which should be replaced with graphics data. For example, television news programs use chroma key to superimpose the weather reporter on a background (‘blue screen’ effect).

CLUT, Color Lookup Table: Translates color information from the display memory into color information for the CRT display. It can be found in a Video DAC. It is the palette portion of the palette DAC.

Color Key: The CL-GD5446 can overlay the computer-generated graphics on a pixel-by-pixel basis, with external video. One method of determining whether to overlay a pixel involves comparing it with a specific color or range of colors.

Color Modes: Use 2, 4, 8, or more bits-per-pixel. Table E1-1 summarizes the number of colors and standards for the colors that were first available.

Table E1-1. Color Modes

Bits per Pixel	Number of Colors	Standards
2	4	CGA
4	16	CGA
8	256	VGA
15	32,368	TARGA™
16	65,536	VGA/XGA™
24	16,777,216	Cirrus Logic True Color
32	16,777,216	True Color with Alpha channel

Color Planes: In planar modes, the display memory is separated into four independent planes of memory, with each plane dedicated to controlling one color component (Red, Green, Blue, and Intensify). Each pixel of the display occupies one bit position in each plane. Planar modes are generally 16 colors. In Character and Packed-pixel modes, data is organized differently.

Comparator: A hardware element that performs an arithmetic or logical comparison between two fields. The two fields are typically the same width. Arithmetic comparisons include equal, greater than, and less than. Logical comparisons are generally identity.

CONFIG.SYS: A file that provides the system with information regarding application requirements. This information may include peripherals that are connected and require special drivers (such as a mouse). Other information that might be specified is the number of files that can be open simultaneously, or the number of disk drives that can be accessed.

CMOS: Complementary metal oxide semiconductor. A digital logic family characterized by high density, low-to-medium power, and medium-to-high speeds. All modern VGA controllers are fabricated using CMOS.

CPU, Central Processing Unit: The master computer unit in a system. In the VGA world, this is typically a 80386, 80486, or Pentium® microprocessor.

CRT, Cathode Ray Tube: An electron beam is generated, accelerated, and made to strike a phosphor coating on the inside of an evacuated glass enclosure. The phosphor glows as a result of the energy imparted by the beam. By precisely controlling the position and intensity of the electron beam, meaningful patterns are made to appear in the phosphor and are visible through the glass.

DAC: digital-to-analog converter. The DACs in a VGA system convert the 6 or 8 digital bits per color (Red, Green, and Blue) to analog levels suitable for the Analog interface.

DCLK: The package pin on which the pixel clock (or a multiple or sub-multiple) is present. See also **VCLK**.

DDC™, Display Data Channel: A definition of a communication channel between a computer display and the host system. This is a VESA proposal. Cirrus Logic components and software support either DDC1 or DDC2B.

Digital Interface: A type of interface used between the display controller and display where display color is controlled by digital color control lines switching on and off. The number of colors that can be supported depends on the number of signal lines in the interface, and is generally either 8, 16, or 64. Most digital interfaces are TTL- (transistor-transistor logic) compatible. CGA, MDA, and EGA use Digital interfaces. In the IBM world, the Digital interface is usually mechanized with a 9-pin connector.

Digital Monitor (TTL): A monitor that receives input in the form of a digital code. Typical digital monitors can display 8, 16, or 64 colors. Digital monitors are becoming obsolete.

Digital: A method of representing data where the individual components are either fully on or fully off.

Digitize: To convert an analog image or signal to a corresponding series of numbers.

Display Memory: The area in the computer memory where the information used to update the screen is kept. In the IBM-compatible world, the range of addresses for this data is A000:0 through BFFF:F.

Display Modes: In the IBM-compatible world, a number of standard display modes have been defined. In addition to the standard modes enumerated in Table E1-2, there are many Extended Display modes, as shown in Table E1-3 on page E1-6.

Table E1-2. Standard Display Modes

Mode(s)	Colors	Alphanumeric Resolution	Pixel Resolution	AN/ APA
0,1	16	40 × 25	360 × 400	AN
2,3	16	80 × 25	720 × 400	AN
4,5	4	40 × 25	320 × 200	APA
6	2	80 × 25	640 × 200	APA
7	Monochrome	80 × 25	720 × 400	AN
D	16	40 × 25	320 × 200	APA
E	16	80 × 25	640 × 200	APA
F	Monochrome	80 × 25	640 × 250	APA
10	16	80 × 25	640 × 350	APA
11	2	80 × 30	640 × 480	APA
12	16	80 × 30	640 × 480	APA
13	256	40 × 25	320 × 200	APA

Table E1-3. Cirrus Logic Extended Display Mode Numbers

Resolution	Pixel Size				
	4-bpp	8-bpp	16-bpp	24-bpp	32-bpp
640 × 480	12	5F	64	71	76
800 × 600	58/6A	5C	65	78	72
1024 × 768	5D	60	74	79	73
1152 × 864		7C	7D	7E	
1280 × 1024	6C	6D	75	77	
1600 × 1200		7B			

Dithering: To intersperse a pattern of one color (for example, blue) with a pattern of another color (for example, red) to give the subjective effect of a color somewhere between the two colors (blue and red combined make magenta). This technique is effective over large surfaces, but fails if the area is too small. This technique creates the appearance of more colors at the expense of resolution.

DIP, Dual Inline Package: A method of packaging semiconductor chips that was essentially ubiquitous until the 1980s. It is being replaced with plastic quad flatpack and pin grid arrays for devices with high pin counts, and small outline packages for devices with low or medium pin counts.

DPMS, Display Power Management Signaling: A VESA proposal to standardize on a common definition and methodology in which the display controller sends a signal to the display that enables it to enter various power management states. The VGA controller can instruct the monitor go enter one of a number of reduced power states. DPMS is supported by current Cirrus Logic desktop products.

DRAM, Dynamic Random Access Memory: A memory technology characterized by extreme high density, low power, and low cost. It must be more or less continuously refreshed to avoid loss of data. DRAM is the least expensive memory available.

Driver: A software module that interfaces a particular display device to an application program to allow operation at higher resolutions than standard VGA.

Dual-Page Mapping: Refers to using both Offset registers, 0 and 1, as the window into display memory. This mode is chosen when GRB[0] is programmed to '1'. In this mode, register SA15 chooses between Offset registers 0 and 1. Linear addressing has largely replaced frame buffer mapping.

EDO, Extended Data Out DRAM: A DRAM technology characterized by very short Fast-Page mode cycle times. The CL-GD5446 supports EDO devices.

EEPROM, Electrically Erasable Programmable Read-only Memory: A memory storage device that can be written repeatedly with no special erasure fixture. EEPROMs do not lose their contents when they are powered down. The Cirrus Logic BIOS can use EEPROMs to record information regarding the connected monitor. This is being replaced with DDC2B.

EGA: Enhanced Graphics Adapter. This was the second color adapter available for IBM-compatible computers. While EGA is generally considered obsolete, the VGA standard includes the modes originally designed for EGA.

Emulation: Simulation of unavailable hardware by available hardware and software. Emulations improve the usefulness of a product by making it compatible with other products. EGA is capable of emulating MDA and sometimes CGA and Hercules. VGA is capable of emulating EGA, CGA, and MDA.

EPROM, Electrically Programmable Read-only Memory: A memory storage device that can be written once (per erasure cycle) and read many times. In the VGA world, it stores the BIOS.

Fast-Page Mode: A read or write mode of DRAMs characterized by a decrease in cycle time of about 2–3 times and a corresponding increase in performance. The data accessed in Fast-Page mode cycles must be adjacent in memory. See **EDO**.

Feature Connector: An expansion connector on the VGA that can accept or drive video signals to or from the VGA. This is used in applications involving video overlay. This is also called the VESA Pass-through connector.

FIFO, First In First Out: A memory that temporarily holds data so that the sending device can send data faster than the receiving device can accept it. The sending and receiving devices typically operate asynchronously.

Fixed-Frequency Monitor: A monitor that can accept a fixed-horizontal frequency, usually 31.5 kHz. Such monitors can accommodate different vertical resolutions by operating at different vertical frequencies, usually either 60 or 70 Hz.

Frame Buffer: Another term for display memory.

Frequency Synthesizer: An electronic circuit that generates a number of frequencies from a fixed-reference frequency. Some frequency synthesizers generate only a relatively small number of frequencies; others generate hundreds of different frequencies.

Glue Logic: Additional logic devices required to interconnect the major components of a system.

Graphics: This term is used when referring to pixels, frame buffer contents, and so on, that are locally computer-generated and displayed in a background window. This data is in contrast with **Video** data.

Graphics Controller: On EGA and VGA, a section of circuitry that provides hardware assist for graphics drawing algorithms by performing logical functions on data written to display memory.

Graphics Mode: (Also APA) A display mode where all pixels on the display screen are independently controlled to draw graphics objects (as opposed to Text mode, where only a predefined set of characters can be displayed).

Hardware: A computing system is normally said to have two major components: hardware and software. Hardware is the portion that executes the step-by-step procedure necessary to perform a particular task as instructed by the software.

HERC, (HGC) Hercules Graphics Adapter: The third display format standardized for the PC family of computers, following the MDA and CGA. It provides standard 80-character-by-25-row alphanumeric display, and 720 horizontal by 348 vertical pixels in Monochrome Graphics mode. It was designed to replace MDA, and provided monochrome APA.

Hex Code, Hexadecimal: A numbering system using base 16. The allowable digits are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. A base 16 numbering system is useful because conversion to and from base 2 is trivial. Numbers written in base 16 are typically denoted by a prepended '0x' or an appended 'h'.

Interlaced: A display system where the even scanlines are refreshed in one vertical cycle (field) and the odd scanlines are refreshed in another vertical cycle. The advantage is that the bandwidth is roughly half that required for a non-interlaced system of the same resolution. This results in less costly hardware. It can also make it possible to display a resolution that would otherwise be impossible to display on certain hardware. The disadvantage of an interlaced system is flickering, especially when displaying objects that are only a single scanline high.

Interpolation: This method involves creating 'in-between' pixels and scanlines. This is done by replicating existing pixels and scanlines or by interpolating (averaging) between existing pixels and scanlines. Using interpolation avoids the blockiness associated with replication. Y-interpolation (vertical interpolation) increases the required display memory bandwidth because two scanlines must be fetched for each scanline displayed. See also **Zoom**.

ISA, Industry Standard Architecture: When referring to IBM-compatible computers, this was the definition of the standard bus until the introduction of VESA and PCI in the early 1990s.

Linear Addressing: This is a modern method of addressing the display memory. The display memory (in the IBM PC world) was originally was located in a 128-Kbyte area from A000:0 through BFFF:F, too small for the display systems of today with multi-megabyte memories. Linear addressing allows the display memory to be addressed in upper memory, where a large contiguous area is set aside for it.

Mapping: Refers to the definition of memory for storing data used by a particular display mode. The range of addresses reserved for graphics information in IBM-compatible systems is from A000:0 to BFFF:F.

MCGA, Multicolor Graphics Array: A graphics adapter designed for the PS/2 series of personal computers, with similar function to the CGA and downwardly compatible to the CGA at the BIOS, control register, and display memory levels. The MCGA drives either an analog monochrome or analog RGB monitor.

MDA, Monochrome Display Adapter: The original display adapter marketed by IBM for personal computers. MDA has no bit-mapped graphics capability.

Monitor: Another term for a CRT display.

Motherboard: The large printed circuit board in a personal computer into which the adapter boards plug. It contains the CPU and core memory. It may also contain the display controller or a number of other peripheral controllers.

MPEG, Moving Picture Experts Group: An international standards organization that develops compression algorithms for motion video. This term has come to also mean the compression algorithms themselves, and even video clips that have been compressed using the algorithms. MPEG algorithms take advantage of the frame-to-frame redundancy of a motion video sequence, as well as the redundancy within each frame.

Multiple-FIFO Architecture: A display controller architecture characterized by having multiple (two or more) FIFOs or write buffers. There is typically one FIFO or write buffer at the CPU interface, and one or more FIFOs in the display pipeline. The CL-GD5446 has three FIFOs in the display pipeline that allow simultaneous video display, video capture, and occlusion or Y-interpolated zooming.

Multifrequency Monitor: A monitor that accommodates a variety of horizontal and vertical synchronization frequencies. This monitor type accepts inputs from many different display adapters, and is typically capable of either analog or digital input.

Nibble: A group of four contiguous bits. It can take any of 16 (2^4) values.

Non-interlaced: A display system where every pixel is refreshed during every vertical scan. A non-interlaced system is normally more expensive than an interlaced system of the same resolution, and is usually said to have a more pleasing appearance.

NTSC: A color encoding scheme for television. NTSC is used in North America and Japan, as well as some other regions. NTSC is often said to be a timing standard; it is an extension of RS-170.

Occlusion: The superimposition of individual pixels of graphics onto video or video onto graphics. The CL-GD5446 supports occlusion based on color key or chroma key.

Offset: The distance in display memory between the beginnings of adjacent character rows or scanlines. See also **Pitch**.

Overlay: The superimposition of video (typically live) onto computer-generated graphics.

Overscan: That portion on all four sides of the display between active pixels and blanking.

Packed Pixel: Color information for a pixel packed into one word of memory data. For a system with few colors, this packed pixel may require only a part of one word of memory; for very elaborate systems, a packed pixel might be several words long. See **Planar**.

PAL: A color encoding scheme for television. PAL is used in Europe (except France), as well as some other regions.

Palette: The range of colors available on the screen, not necessarily simultaneously. For VGA, this is either 16 or 256 simultaneous colors out of 262,144. For CL-GD5446, the palette is extended to 32,768, 65,536, or 16,777,216 simultaneous colors on the screen.

Palette DAC: The triple 8-bit DAC with its associated LUT.

PCI, Peripheral Component Interconnect: "The PCI local bus is a high-performance, 32- or 64-bit bus with multiplexed address and data lines. It is intended for use as an interconnect mechanism between highly integrated peripheral controller components, peripheral add-in boards, and processor/memory systems."[1] In 1995, PCI replaced VESA VL-Bus as the high-performance display controller/host interface.

Pitch: The distance in display memory between the beginnings of adjacent character rows or scanlines. Pixels that are vertically adjacent on the screen are separated in the frame buffer by a distance called the pitch. See also **Offset**.

---

[1] PCI Local Bus Specification, Revision 2.1.

Pixel: An acronym for picture element, and is also called a pel. A pixel is the smallest addressable graphic on a display screen. In RGB systems, the color of a pixel is defined by some red intensity, some green intensity, and some blue intensity.

Planar: In display terms, the pixel color information is stored in four bits across four memory planes. This allows a maximum of 16 colors (2^4). See **Packed Pixel**.

RAM BIOS: The BIOS can be copied from relatively slow ROM into relatively fast RAM. When this is done, it executes faster, enhancing performance of the subsystem being controlled.

RAM, Random Access Memory: This term has come to mean any semiconductor memory whose write access time is approximately the same as its read access time. Typically this includes SRAMs (static RAMs) and DRAMs (dynamic RAMs). This definition specifically eliminates memories that cannot be altered and memories that require a special fixture for erasing (such as EPROMs).

RAS, Row Address Strobe: A DRAM control signal.

Refresh (Display or Screen Refresh): An image drawn on a CRT display remains visible only for a few milliseconds (the persistence of the screen phosphor), unless it is redrawn continuously. This process is called display refresh or screen refresh. Different displays use different refresh rates, but display refresh is normally required between 60 and 80 times a second to avoid any visible screen flickering. 75 times a second is a common refresh rate. In general, a higher refresh rate results in more stable appearing display.

Register-Level Compatibility: If a peripheral is compatible at the register level with another peripheral, it means that every bit in every register of the two devices has precisely the same meaning. This implies that application programs can circumvent the BIOS and directly program registers in a peripheral device without functionality problems. The CL-GD5446 is register-level-compatible with the IBM VGA standard.

Registers: In an VGA controller, these are the storage elements that contain data relating to the mode or configuration of the device, as opposed to the display memory, which contains the image. Traditionally, the registers are divided into six groups: General, Sequencer, CRT Controller, Graphics Controllers, Attribute, and Extensions. The VGA registers are accessed by a number of addressing schemes, usually involving an index or address register and a data register.

Resolution, Color: The number of simultaneous colors is determined by the number of bits associated with each pixel in the display memory. The more colors, the more bits. If n bits-per-pixel are used, 2^n color combinations can be generated. EGA uses from 1–4 bits-per-pixel, permitting up to 16 (2^4) colors to be displayed on the screen simultaneously. The VGA has an added mode that supports 8 bits-per-pixel, or 256 (2^8) simultaneous colors. The CL-GD5446 has additional modes to support up to 24 bits-per-pixel or 16,777,216 (2^{24}) simultaneous colors. In addition, some modes use a fourth byte, the alpha byte.

Resolution, Spatial: The number of pixels in an area or on the screen. Resolution is typically specified as pixels per scanline and scanlines per frame. High-resolution images require more processing and greater storage requirements per image. In addition, monitor costs increase with resolution, particularly above one million pixels. Also, different applications require different resolutions.

RGB: An interface used with color displays that uses three color signals — Red, Green, and Blue, as opposed to an interface used with a monochrome display requiring only a single signal. Both digital and analog RGB interfaces exist.

ROM, Read-only Memory: A type of memory characterized by not being alterable (see EPROM). ROMs are typically used to contain low-level programs that do not change, such as BIOS.

Simultaneous Colors: The number of colors in a display system that can be displayed on the screen at one time. This number is limited by the circuitry of the display adapter, and is often much smaller than the number of colors the display device can actually support. The number of simultaneous colors a display adapter supports is normally determined by the number of color planes, or bits per pixel, that it uses. For example, a device with 4 bits-per-pixel supports 16 simultaneous colors.

Single-Page Mapping: Refers to always using Offset register 0 (GR9) as the window into display memory. The mode is selected when GRB[0] is programmed to '0'.

Software: A computing system is normally spoken of as having two major components: hardware and software. Software is that portion that instructs the hardware in the step-by-step procedure necessary to perform a particular task.

Sprite: See **Bitmap**.

SVGA, Super VGA: Graphics adapters that extend the capabilities of the features provided by the original IBM VGA. The first Super VGA provided a $640 \times 480 \times 256$-Color mode.

SMT, Surface Mount Technology: A method of mounting devices (such as integrated circuits, resistors, capacitors, and so on) on a printed circuit board, characterized by not requiring mounting holes. Rather, the devices are soldered to pads on the printed circuit board. Surface-mount devices are typically smaller than the equivalent through-hole devices.

Swizzle: To swap bits within a byte. For example, bit 7 with bit 0 and bit 0 with bit 7; bit 6 with bit 1 and bit 1 with bit 6; bit 5 with bit 2 and bit 2 with bit 5; bit 4 with bit 3 and bit 3 with bit 4 and so on.

TTL, Transistor-Transistor Logic: A collection of logic families developed beginning in the 1960s. TTL is gradually being replaced with CMOS, for all but the fastest or most cost-sensitive applications.

True-color: 24- or 32-bits-per-pixel color providing photo-realistic image quality.

VCLK: The internal signal operating at the pixel rate.

Vertical Retrace: The time interval immediately following the completion of a complete frame (or field for an interlaced display). The electron beam returns to the top of the display screen in preparation for the next frame or field during this period.

VESA VL-Bus: This bus interface is very closely modeled on the 80486 processor bus. Therefore, its life-span closely approximates that of the 80486 processor.

VESA, Video Electronics Standards Association: A consortium of CRT monitor vendors, graphics chip vendors, and graphics software vendors that set hardware and software standards for PC-compatible graphics monitors and software interfaces. Cirrus Logic is an active participant on many of the VESA committees.

VGA, Video Graphics Array: The VGA standard was introduced by IBM in 1987. In the IBM definition, the maximum spatial resolution is 640 × 480 (modes 11 and 12), and the maximum color resolution is 256 colors (Mode 13). This is enhanced or extended by third-party chip vendors to up to 1280 × 1024 and up to 16,777,216 colors.

Video: This term is used when referring to pixels, frame buffer contents, and so on, that are displayed in a video window. This applies especially to data that is captured through the V-Port. This is in contrast with **Graphics** data.

VLSI, Very Large Scale Integration: The technology of manufacturing integrated circuits (chips) with thousands of transistors on a single device. The personal computer was made possible because of VLSI technology.

Wait State: When a system processor is reading or writing a memory or peripheral device that cannot respond fast enough, one or more time intervals (typically tens of nanoseconds per interval) are inserted, during that time the processor does nothing but wait for the slower device. While this has an detrimental effect on system throughput, it is unavoidable. The number of wait states can be reduced using techniques such as CPU-bus caches or write FIFOs.

Word: The amount of memory that a given computer can access in a single cycle. In the IBM-compatible world, this is either 16 or 32 bits.

Write Buffer: A term used to denote the buffer that is logically positioned between the CPU interface and the display memory.

YCrCb: A color space defined in the CCIR recommendation CCIR601.

YUV: A color space generally associated with color television.

Zoom: To make an image (typically video) occupy an area on the screen that encompasses more pixels than it contains. The pixels and scanlines that occupy the in-between locations can be generated by replicating existing data or by interpolating (averaging) between existing data. See also **Interpolation**.

BIBLIOGRAPHY

Adams, Lee. – *Visualization Graphics in C*
Windcrest/McGraw-Hill, Inc., 1991.

Ferraro, Richard F. – *Programmer's Guide to the EGA and VGA Cards*
Addison-Wesley, third edition, 1995.

Jack, Keith – *Video Demystified*
HighText Publications, Inc., 1993.

Kliewer, Bradley Dyck. – *EGA/VGA - A Programmers Reference Guide*
McGraw-Hill, 1988.

Phoenix Technologies, Ltd. – *System BIOS for IBM PC/XT/AT Computers and Compatibles*
Addison-Wesley, 1989.

Sutty, George and Steve Blair. – *Advanced Programmers Guide to Super VGAs*
Brady Books, Simon & Schuster, Inc., 1990.

Sutty, George and Steve Blair. – *Programmer's Guide to the EGA/VGA*
Brady Books, Simon & Schuster, Inc., 1988.

Wilton, Richard. – *PC & PS/2 Video Systems*
Microsoft Press, 1987.

Register Index

REGISTER INDEX

Index

INDEX

C

D

E

F

G

H

N

O

R

S

T

U

V

W

Y

Z

Direct Sales Offices

Domestic

N. CALIFORNIA
Fremont
TEL: 510/623-8300
FAX: 510/252-6020

S. CALIFORNIA
Irvine
TEL: 714/453-5961
FAX: 714/453-5962

Westlake Village
TEL: 805/371-5860
FAX: 805/371-5861

SOUTH CENTRAL AREA
Austin, TX
TEL: 512/255-0080
FAX: 512/255-0733

Dallas, TX
TEL: 214/252-6698
FAX: 214/252-5681

Houston, TX
TEL: 281/257-2525
FAX: 281/257-2555

NORTHEASTERN AREA
Andover, MA
TEL: 508/474-9300
FAX: 508/474-9149

SOUTHEASTERN AREA
Duluth, GA
TEL: 770/935-6110
FAX: 770/935-6112

Raleigh, NC
TEL: 919/859-5210
FAX: 919/859-5334

Boca Raton, FL
TEL: 407/241-2364
FAX: 407/241-7990

International

FRANCE
Paris
TEL: 33/1-48-12-2812
FAX: 33/1-48-12-2810

GERMANY
Munich
TEL: 49/81-52-40084
FAX: 49/81-52-40077

HONG KONG
Tsimshatsui
TEL: 852/2376-0801
FAX: 852/2375-1202

ITALY
Milan
TEL: 39/2-3360-5458
FAX: 39/2-3360-5426

JAPAN
Tokyo
TEL: 81/3-3340-9111
FAX: 81/3-3340-9120

KOREA
Seoul
TEL: 82/2-565-8561
FAX: 82/2-565-8565

SINGAPORE
TEL: 65/743-4111
FAX: 65/742-4111

TAIWAN
Taipei
TEL: 886/2-718-4533
FAX: 886/2-718-4526

UNITED KINGDOM
London, England
TEL: 44/1727-872424
FAX: 44/1727-875919

The Company

Headquartered in Fremont, California, Cirrus Logic is a leading manufacturer of advanced integrated circuits for desktop and portable computing, telecommunications, and consumer electronics. The Company applies its system-level expertise in analog and digital design to innovate highly integrated, software-rich solutions.

Cirrus Logic has developed a broad portfolio of products and technologies for applications spanning multimedia, graphics, communications, system logic, mass storage, and data acquisition.

The Cirrus Logic formula combines innovative architectures in silicon with system design expertise. We deliver complete solutions — chips, software, evaluation boards, and manufacturing kits — on-time, to help you win in the marketplace.

Cirrus Logic's manufacturing strategy ensures maximum product quality, availability, and value for our customers.

Talk to our systems and applications specialists; see how you can benefit from a new kind of semiconductor company.

Cirrus Logic Inc.
3100 West Warren Ave., Fremont, CA 94538
TEL: 510/623-8300 FAX: 510/252-6020

Publications Ordering: 800/359-6414 (USA) or 510/249-4200
World Wide Web: http://www.cirrus.com

ASCII Character Set: 00–7f

Char	0x	1x	2x	3x	4x	5x	6x	7x
0	NUL	DLE	**space**	**0**	**@**	**P**	**´**	**p**
1	SOH	DC1	**!**	**1**	**A**	**Q**	**a**	**q**
2	STX	DC2	**“**	**2**	**B**	**R**	**b**	**r**
3	ETX	DC3	**#**	**3**	**C**	**S**	**c**	**s**
4	EOT	DC4	**$**	**4**	**D**	**T**	**d**	**t**
5	ENQ	NAK	**%**	**5**	**E**	**U**	**e**	**u**
6	ACK	SYN	**&**	**6**	**F**	**V**	**f**	**v**
7	BEL	ETB	**`**	**7**	**G**	**W**	**g**	**w**
8	BS	CAN	**(**	**8**	**H**	**X**	**h**	**x**
9	HT	EM	**)**	**9**	**I**	**Y**	**i**	**y**
a	LF	SUB	*****	**:**	**J**	**Z**	**j**	**z**
b	VT	ESC	**+**	**;**	**K**	**[**	**k**	**{**
c	FF	FS	**,**	**<**	**L**	****	**l**	**|**
d	CR	GS	**-**	**=**	**M**	**]**	**m**	**}**
e	SO	RS	**.**	**>**	**N**	**^**	**n**	**~**
f	SI	US	**/**	**?**	**O**	**_**	**o**	**DEL**