Multimedia Controller for PCI Bus

Data Sheet

Features

- Glueless interface to PCI bus. (PCI spec. 2.1 compliance).
- Glueless interface to MPEG decoders (e.g., ZR36100/ ZR36110), video digitizers (e.g., SAA7110, SAA7111), MPEG audio decoders (e.g., CS4920, SAA2502) and video encoders (e.g., SAA7188, MD0207).
- DMA transfer of video and coded bitstream.
- Smooth image down-scaler (up to 5-tap horizontal filter).
- On-chip pixel-accurate, real color-key masking.
- YUV-to-RGB converter with quantization noise reduction (error diffusion).
- Video output: 15- and 16-bit RGB pixel formats, as well as 24-bit (packed and unpacked), and YUV 4:2:2.
- I^2C bus master port.
- Plug & Play support.
- Graceful recovery from extreme bus latencies.
- Choice of emulated interlaced video display, or single field display, to eliminate motion artifacts.
- Hardware support for simple, cost effective frame grabbing.
- 160-pin PQFP package.
- Accompanying software includes MCI and VfW drivers for Windows 3.1 and Windows 95, OM1 DOS driver and a TV-on-PC application example.

Applications

- MPEG and frame-grabbing add-in boards for PCI systems.
- Multimedia and multimedia + graphics subsystems using a secondary PCI bus.
- PCI motherboards of multimedia computers.
- Multimedia solutions to 486, Pentium, PowerPC, Macintosh, and Alpha PCI systems.
- Video capture for video conferencing applications.

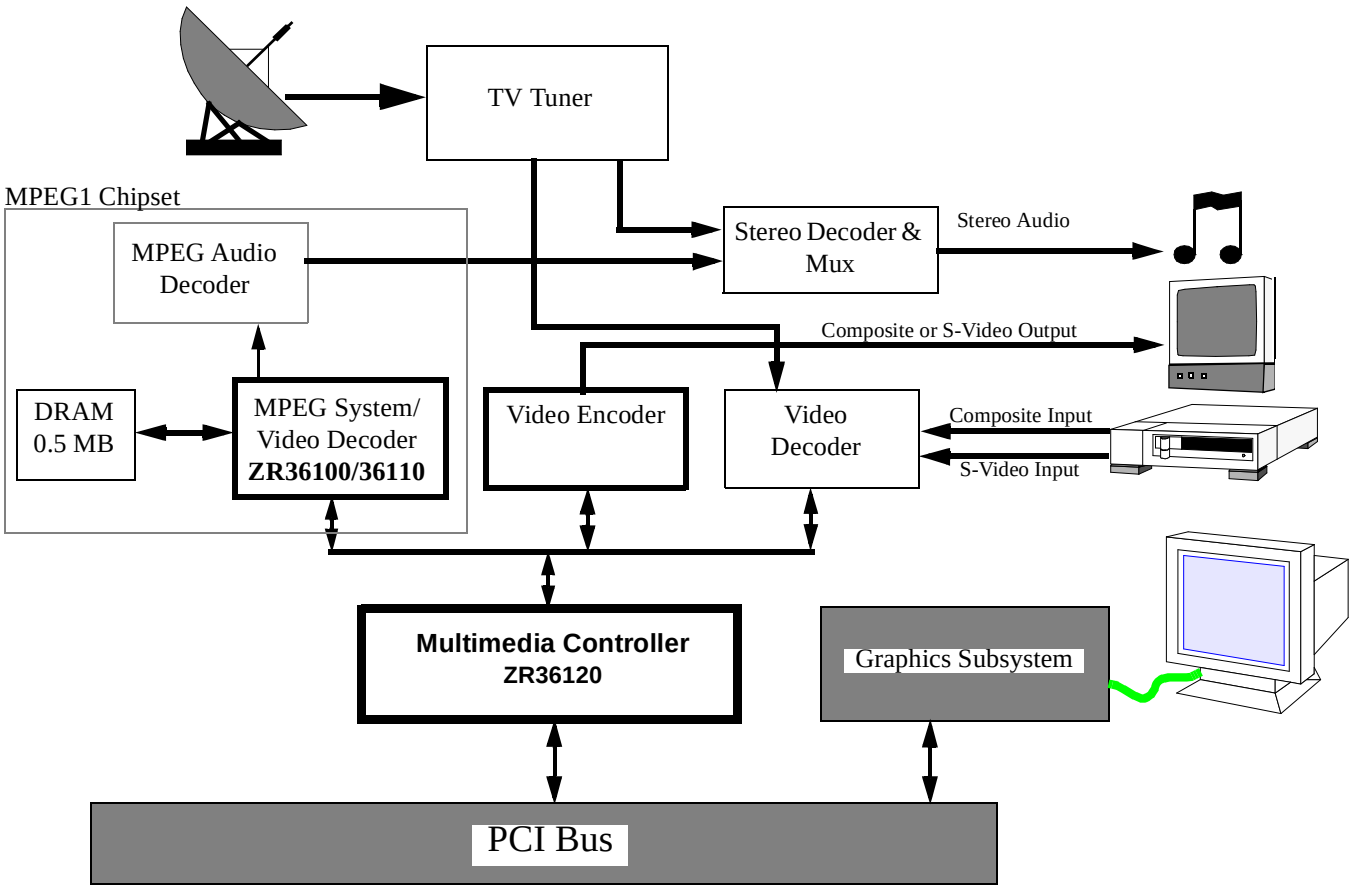


Figure 1: Block Diagram of A Typical Video Subsystem for PCI Computers

1. INTRODUCTION

ZORAN's **ZR36120** is aimed for multimedia playback applications on PCI systems. It captures digital video, such as decompressed MPEG video, or the output of a video analog-to-digital converter, and creates a scaled video window in the system display memory. It provides the host CPU with full control over up to four non-PCI multimedia devices (e.g., **ZR36100/ ZR36110**) as well as any number of I^2C slave devices, such as an audio decoder, video digitizers and encoders, TV tuners, etc. Software support includes MCI driver, Open-MPEG1 driver, and hardware abstraction API. A complete development kit, including a reference board (*mulTVideo*), is available from **ZORAN**

The **ZR36120** directly interfaces the PCI bus. As a bus master it retrieves data (e.g., MPEG bitstream) from system memory and writes digital video data onto display memory. As a bus target ("slave"), it provides the host CPU control over several non-PCI "guest" devices. For example, it can pass microcodes, parameters and on-line commands from the host CPU to an MPEG system/video decoder and an audio decoder. To fulfil this goal the **ZR36120** incorporates a microcontroller-type 8-bit "guest" bus, and an I^2C bus master port. The latter is used to control devices like the CS4920 and SAA2502 audio decoders, SAA7110 and SAA7111 video digitizers, SAA7188 video encoder, etc.

The digital video is captured from a generic digital video bus (YUV 4:2:2 format). The video data can be down-scaled, if scaling is needed. Horizontal filtering is applied before down-scaling, to reduce spatial artifacts. If necessary, the video data is converted to one of several pixel formats. When 16- or 15-bit RGB formats are selected, error diffusion can be applied, to reduce color quantization artifacts.

The host CPU dynamically programs the location and size of the desired video-window into the **ZR36120** registers. The down-scaled, packed video data is written directly to the display memory through the video DMA channel of the **ZR36120**, using PCI burst transfers . Any combination of pixels within the video window can be masked off by other resources (e.g., game graphics). The **ZR36120** video-write mechanism can skip these areas and avoid overriding the display memory in the corresponding locations.

The **ZR36120**'s video window can be configured to emulate the interlaced video input, or to ignore the bottom field and display only the top one. The latter mode is sometimes preferred with live video, to eliminate motion artifacts.

In order to overcome momentary latencies of the PCI bus, the **ZR36120** uses a 256-byte Video FIFO Buffer. Data is organized in the buffer in 32-bit words, ready for PCI bus transfers.

A second DMA channel is dedicated to transferring data from system memory to a "guest" decoder. An 8 x 32 bit Code FIFO Buffer is used to secure an average rate of bitstream, even in cases of extreme PCI bus latencies.

The **ZR36120** is packaged in a 160-pin PQFP. It requires a 5V power supply.

About this Manual

This user's manual provides hardware and software developers with the complete technical information required to design systems with the **ZR36120**. The first three chapters, which are more introductory, are also adequate for engineering managers who evaluate the **ZR36120**.

1.1 Notations and Conventions

External signals: capital, bold letters (e.g., **IDSEL**)

Active-low mark: # (e.g., **DEVSEL#**)

Buses: XXmsb_index...lsb_index (e.g., **AD31...0**)

Register fields: XXmsb_index:lsb_index (e.g., 27:16)

Configurable parameters: Capital initials, attached words (e.g., HorDcm, TopField).

Register types:

R - read only
RC - read-clear. Writing '1' clears the register bit.
RS - read-set. Writing '1' sets the register bit to '1'.
RW - read-write (contents of write can be read back)

Numbers:

Unmarked numbers are decimal
(e.g., 365, 23.19).

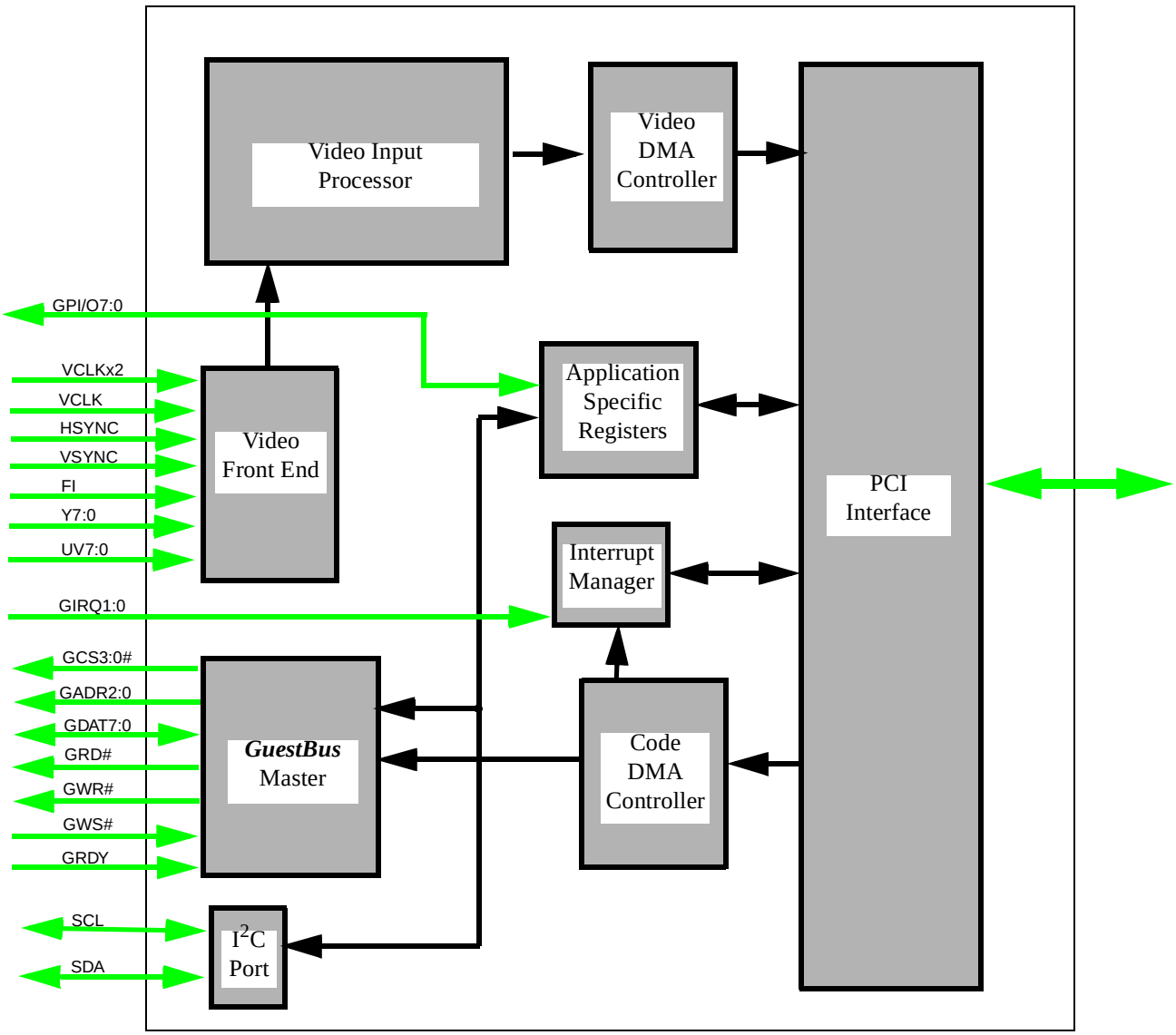
Hexadecimal numbers are marked with a '0x' prefix
(e.g., 0xB000, 0x3).

Binary numbers are marked with a 'b' suffix
(e.g., 010b, 0000110100011b).

Math expressions: following the C syntax, e.g.,
X << n - X is left-shifted n times
X++ - X=X+1

Table of Contents

Figure 2: ZR36120 Block Diagram



2. Pin Description

The following tables provide a brief description of each pin on the **ZR36120**. The following definitions are used in these tables:

I - Input signal- 3-S - 3-state signal 3-S* - sustained 3-S. Refer to PCI spec. 2.1, section 2.1
O - Output signal- - I - Standard input-only OD - open drain, shared by multiple drivers, as a wired-or.
I/O - Bidirectional signal- - O - Standard active driver

Table 1: PCI Interface Signals

Name	Dir	Type	Pin #	Description
AD31...0	I/O	3-S	24-25, 27, 29, 31-32, 34-35, 42-43, 45-46, 48-49, 51, 53, 68-69, 71-72, 74, 76, 78-79, 83, 85, 87, 89-90, 92-93, 95	Multiplexed address and data bus pins
C/BE3...0#	I/O	3-S	37, 55, 66, 82	Bus commands or byte enables
PAR	I/O	3-S	65	Even parity bit for AD31...0 and C/BE3...0#
FRAME#	I/O	3-S*	56	PCI cycle frame
TRDY#	I/O	3-S*	59	PCI target ready indicator
IRDY#	I/O	3-S*	58	PCI initiator ready
STOP#	I/O	3-S*	62	Indicates a target request to stop the current data transfer
DEVSEL#	I/O	3-S*	61	PCI device select, indicates that the target has decoded its address
IDSEL	I	I	38	PCI initialization device select. Used as a chip-select to the **ZR36120**'s configuration space.
REQ#	O	3-S	21	PCI bus request
GNT#	I	I	20	PCI bus grant
PCICLK	I	I	18	PCI clock. Minimum frequency must be 1.5 times **VCLK**
PCIRST#	I	I	16	PCI reset. When active, all **ZR36120** output pins are tri-stated. A low-to-high transition puts the **ZR36120** into its power-on reset state. Minimum active-low duration is 3 PCI clocks.
INTA#	O	OD	14	PCI interrupt request A. A low level of this signal requests an interrupt from the host.

Table 2: Digital Video Bus Signals

Name	Dir	Type	Pin #	Description
VCLKx2	I	I	141	Clock input, at double the frequency of the video data. The frequency depends on the format and standard chosen: 29.5 MHz, square pixel PAL 24.5454 MHz, square pixel NTSC 27.0 MHz, CCIR PAL and NTSC
VCLK	I	I	143	Video data clock, at half the frequency of VCLKX2. 14.75 MHz, square pixel PAL 12.2727 MHz, square pixel NTSC 13.5 MHz, CCIR PAL and NTSC This is the pixel clock.
VSYNC	I	I	146	Vertical synchronization.
HSYNC	I	I	147	Horizontal synchronization.
FI	I	I	145	Digital video bus field indicator (top/bottom).
Y7...0	I	I	11-9, 7-4, 2	8-bit luminance data.
UV7...0	I	I	158-155, 153-151, 149	8-bit multiplexed chrominance data

Table 3: *GuestBus* Signals

Name	Dir	Type	Pin #	Description
GCS3...0#	O	*O*	118-119, 123-124	Active low chip-select output to guest bus devices
GADR2...0	O	*O*	125-127	Address outputs to guest bus devices
GDAT7...0	I/O	3-S	139, 137-135, 133-132, 130-129	Guest data bus
GRD#	O	*O*	117	Active low read output to guest bus devices
GWR#	O	*O*	116	Active low write output to guest bus devices
GRDY	I		114	Active high “guest ready” input. When this signal is low the ***GuestBus*** stops the code-write transfers to the destined guest device.
GWS#	I		113	Guest Wait-State indication. Assertion of this active-low input allows the guest device to extend the ***GuestBus*** write (or read) cycle until it is capable of latching-in (or provide) the data.
GIRQ1...0	I		112-111	Positive-edge-sensitive interrupt request inputs from one or two of the GuestBus slave devices.

Table 4: I^2C Bus Signals

Name	Dir	Type	Pin #	Description
SDA	I/O	OD	109	I^2C bus data. Should be pulled up with a 1K resistor.
SCL	I/O	OD	108	I^2C bus clock (qualifies for a single master operation). Should be pulled up with a 1K resistor.

Table 5: General Purpose I/O Signals

Name	Dir	Type	Pin #	Description
GPIO7...0	I/O	3-S	107-106, 104-100, 13	General purpose input/output pins. After hardware or software reset all 8 pins are configured as inputs. Their logical levels are reflected as register bits. Any of the pins can be configured as output. In this state its logical level is driven by a register bit. These pins may be used to monitor or control various board level functions.

Table 6: Power & Test

Name	Dir	Type	Pin #	Description
ENID	I	I	98	Factory Test pin. In normal operation must be connected to ground potential (V_{SS}).
TEST	I	I	97	Factory Test pin. In normal operation must be connected to ground potential (V_{SS}).
V_{DD}	I	pow	3, 12, 17, 22, 28, 39, 52, 64, 75, 86, 96, 110, 121, 131, 140, 144, 150, 159	5 Volt power supply.
V_{SS}	I	gnd	1, 8, 15, 19, 23, 26, 30, 33, 36, 40-41, 44, 47, 50, 54, 57, 60, 63, 67, 70, 73, 77, 80-81, 84, 88, 91, 94, 99, 105, 115, 120, 122, 128, 134, 138, 142, 148, 154, 160	Power supply ground.

3. FUNCTIONAL OVERVIEW

The **ZR36120** multimedia controller performs the following functions:

- Interfacing to a YUV 4:2:2 digital video bus (e.g., Philips SAA7110 or SAA7111).
- Independent horizontal and vertical downscaling, with optional horizontal filtering, of the input image.
- Conversion of the YUV 4:2:2 digital video input into one of the following pixel formats: YUV 4:2:2, RGB 5:6:5, RGB 5:5:5 or RGB 8:8:8 (packed or unpacked).
- Two DMA channels for burst transfers of video and coded bitstream.
- Overlay support: any number of video pixels can be masked off, letting the corresponding graphics pixels appear instead of them.
- I^2C Bus mastering.
- Frame grabbing.
- Two display modes: emulation of the interlaced input video, or a single field display.

The **ZR36120** supports digital video in CCIR 601 or square pixel formats, following either NTSC or PAL video standard. Other input schemes are supported as well.

Figure 1 depicts the **ZR36120** functional block diagram.

The functional description below follows the block diagram.

3.1 Digital Video Path

Digital Video Front End

The VFE samples the incoming YUV 4:2:2 video data and sync signals with a flexible sampling scheme, that makes it compatible with a wide variety of digital video sources. The digital input video can be cropped. The input resolutions supported by the VFE range from 32x32 to 1023x1023, in continuous steps.

Video Input Processor

The chroma components of the video data are upsampled to form a YUV 4:4:4 format. All components are horizontally filtered. Five filtering schemes are implemented, with different parameters for chrominance and luminance samples. Downscaling is available (if required) horizontally and vertically. Vertical down scaling can be optimized to the output of the ZR36100 (mode: duplicated fields).

Pixel Formatting

The filtered and scaled video is packed according to the selected pixel format. YUV 4:2:2, 24-bit RGB (packed or unpacked) and 15- and 16-bit RGB are supported. An error diffusion algorithm can be applied to the RGB 5:5:5 and 5:6:5, in order to eliminate quantization artifacts on the output image.

3.2 Video DMA Controller

Pixel Bursts

The packed pixels are transferred directly to the display memory (or to the system memory), using PCI DMA bursts. Both Little and “Gib” Endian formats are supported where applicable (Refer to PCI Multimedia Design Guide, Revision 1.0).

Display Modes

The display mode can be configured to either emulated interlaced video (both input fields are displayed) or single field display. The latter is appropriate for motion artifact elimination when displaying live video.

Frame Grabbing

The **ZR36120** can grab video frames (scaled or non scaled), or fields, in any of the pixel formats listed above, directly into system memory, eliminating the need for memory on the add-in board.

Overlay Control

Graphics overlay is supported, in that display memory areas that are “owned” by graphics applications, may not be loaded with video pixels, allowing true windowing and overlay. The software driver prepares a masking map of the video rectangle, and the **ZR36120** uses this map for masking decision, when transferring the pixels to the display memory.

3.3 Host Control/Communication Services

Application-Specific Registers

The name “application-specific” distinguishes these registers from the PCI configuration space registers. These memory mapped registers provide the host CPU with full control over the operation of the **ZR36120**. The **ZR36120** claims a contiguous space of 4 KBytes in system memory.

GuestBus

Host CPU control over non-PCI devices, such as an MPEG decoder, a video encoder, etc., is done through the **ZR36120**’s ***GuestBus***. Host CPU accesses to these “guest” devices, mapped as application specific registers inside the **ZR36120**, are output as ***GuestBus*** cycles. Such accesses can either use the ***PostOffice*** handshaking protocol, or the Code DMA Controller. The first method is adequate for “random” transfer of commands, configuration data, etc., while the second method provides a faster channel, and is intended for continuous transfer of data. When used in conjunction with the Code DMA Controller, the bandwidth of the ***GuestBus*** is 5.5 MBytes/sec.

PostOffice Handshaking Protocol

The **ZR36120** ***PostOffice*** handshaking protocol, implemented over the ***GuestBus***, allows host CPU accesses to relatively slow guest devices, with no degradation of the PCI bus performance.

I²C Port

A software driven I²C port allows controlling of I²C devices.

Interrupt Manager

Interrupt requests associated with several internal and external conditions are sent to the host via the PCI bus (using INTA#). Selection of interrupt originators is programmable.

3.4 Code DMA Controller

The **ZR36120** includes a DMA channel for transferring data from main memory to a selected device on the ***GuestBus***. Typically, this would be compressed bitstream, to be decompressed by a decompression device hooked on the ***GuestBus***. Other examples are sampled audio (WAV data), MIDI bitstream, etc. Temporary latencies on the PCI bus or the ***GuestBus*** are handled without loss of data.

The ***GuestBus*** master simultaneously serves the ***PostOffice*** accesses and the code DMA transfers: DMA transfers are viewed as the "main" task of the ***GuestBus*** master, while any number of ***PostOffice*** requests may occasionally interrupt the DMA traffic.

The DMA controller supports both auto-initialized (cyclic) block transfers, or single block transfers. The size of the destination block in main memory can be selected out of several possible sizes, ranging from 8 KBytes up to 256 KBytes. The destination block may also be virtually split into several subblocks, allowing the **ZR36120** to interrupt the host CPU when a subblock has been transferred. This feature provides the software with a means of optimizing the refill accesses according to the application requirements and the disk performance.

4. Interfaces

4.1 PCI Bus Interface

In general, the **ZR36120** is compatible with the PCI 2.1 specifications.

As a bus master the **ZR36120** may initiate two types of data transfer over the PCI bus: memory-write (PCI command 0111b), from the **ZR36120**'s internal pixel buffer to the display memory (or main memory), or memory-read-line (PCI command 1110b), from system memory to the **ZR36120**'s internal code buffer.

As a bus target, the **ZR36120** responds to the following types of transfer:

- Memory Read (0110b)
- Memory Read Line (1110b)
- Memory Read Multiple (1100b)
- Memory Write (0111b)
- Configuration Read (1010b)
- Configuration Write (1011b)

All other PCI commands are ignored.

Memory Read Line and Memory Read Multiple are handled exactly as Memory Read.

Usually, as a slave, the **ZR36120** is intended to be accessed with single data phase cycles. However, multiple phase bursts are supported. When the **ZR36120** is accessed in a burst, it increments its internal address such that each data phase is routed to/from the next address location (in doublewords). Slave bursts may be especially advantageous to execute a fast batch of ***PostOffice*** writes to a guest device (although for a large batch using the Code DMA Controller is more efficient). When slave bursts are used to access guest devices, 5 to 7 PCI wait states should be inserted onto the **TRDY#** signal. This is done through a dedicated parameter in the application-specific registers.

As a slave the **ZR36120** supports byte enables, such that an access to explicit bytes within the **ZR36120** is possible.

The error reporting signals, SERR# and PERR#, are not included in the **ZR36120**: as a multimedia device it is only required to report parity errors through the PCI status register.

The **ZR36120** uses the **INTA#** PCI interrupt request line. This line can be logically associated with one or more out of three possible events.

The **ZR36120** has a special control bit for compatibility with Intel's Triton chipset (Pentium/PCI bridge). The host CPU should use this bit if it recognizes the Triton (refer to 10.7 "Video Display Configuration Register").

4.2 Digital Video Bus Interface

The **ZR36120**'s video front end (VFE) interfaces a standard YUV 4:2:2 video bus. It samples the **Y7...0**, **UV7...0**, **HSYNC**, and **VSYNC** lines with every other positive edge of **VCLKx2**. The valid positive edge (out of every two consecutive ones), which is the one used for sampling, is "marked" by **VCLK**, i.e. **VCLK** is used as a data qualifier. The qualifying polarity of **VCLK** is configured by the host. This scheme makes the **ZR36120** compatible with a wide range of digital video sources, and immune to board-level delays (caused by layout, etc.). The minimum and maximum values of **VCLKx2** are 24.0 MHz and 30.0 MHz, respectively.

The VFE internally generates a field indication signal targeted to some internal video processing units. There are two alternative ways of generating the field indication. With devices that output a field indication, the VFE may use the **FI** input as an indicator of the current field identity. The interpretation of the logical level of **FI** (top or bottom field) is configured by the host. With devices that do not provide such an indication, the VFE can detect the field identity from **HSYNC** and **VSYNC**.

The VFE can capture square pixel and CCIR-601 formats, or user defined formats, within the limitation of the parameters. The maximum theoretical total input resolution is 1023 pixels per line by 1023 lines pe r filed. Cropping of the input image is possible by proper configuration of the VFE parameters.

Table 7 lists the Video Front-End parameters. The host CPU needs to configure these parameters according to the timing parameters of the video source (e.g., SAA7110, SAA7111, **ZR36100**, **ZR36110**, etc.) and the required cropping. Note that these parameters relate to the input video, and not to the destination video window.

Table 7: Video Front-End parameters

Parameter	Meaning
VStart	Number of lines (**HSYNC**s) from the active edge (positive or negative, according to VSPol) of **VSYNC** to the first line to be sampled.
HStart	Number of pixel clocks in a line from the active edge of **HSYNC** until the first pixel to be sampled.
VEnd	Number of lines (**HSYNC**s) from the active edge (positive or negative, according to VSPol) of **VSYNC** to the last line to be sampled
HEnd	Number of pixel clocks in a line from the active edge of **HSYNC** until the last pixel to be sampled.
ExtFI	This one bit parameter indicates if the video source provides a field indication signal.
HSPol	Polarity of **HSYNC** pulse. '1' means that HStart, HEnd will be counted from the **negative** edge of **HSYNC**.
VSPol	Polarity of **VSYNC** pulse. '1' means that VStart, VEnd will be counted from the **negative** edge of **VSYNC**.
TopField	If field indication is derived from the **FI** input signal (see ExtFI), then this parameter defines which level of **FI** indicates the top field: '1' - **FI** high indicates the top field. '0' - **FI** low indicates the top field. If field indication is derived internally from **HSYNC** and **VSYNC**, then this parameter defines which level of **HSYNC** indicates the top field on the active edge of **VSYNC**: '1' - **HSYNC** high indicates the top field. '0' - **HSYNC** low indicates the top field.
VCLKPol	Polarity of **VCLK** as a data qualifier. If VCLKPol=1 the video input is sampled with those positive edges of **VCLKx2** that correspond to **VCLK**=1. If VCLKPol=0, the video front is sampled with those positive edges of **VCLKx2** that correspond to **VCLK**=0.

4.3 *GuestBus* Interface

The **ZR36120** masters a generic MCU-style bus aimed to concurrently host up to four slave devices (referred to as "guests"). Examples for such devices are the **ZR36110** MPEG1 decoder, the MD207 video encoder, etc. The bus consists of 8 data lines (GDAT7...0), 3 address lines (GADR2...0), 4 active-low chip-select lines (GCS#3...0), read and write signals (GRD#, GWR#) and a wait-state insertion line (GWS#). The bus also includes two interrupt-request and one status/acknowledge inputs (GIRQ1...0, GRDY, respectively) for a tight handshaking protocol with any of the guests. Two types of data transfers are possible on the guest bus. One is a code-write cycle, initiated by the **ZR36120** (namely the Code DMA Controller), targeted to one of the guests. The second is a ***PostOffice*** cycle, initiated by the host CPU, targeted to any one of the four guests.

A typical selection to be the target of the code-write cycles would be a decompression device.

4.3.1 Flexible *GuestBus* Timing

Different guest devices may have different bus timing requirements. In order to meet these requirements and master the ***GuestBus*** efficiently, the **ZR36120** has two timing parameters for each guest:

- T_{gdurN} is the "low" duration of a **GWR#** or **GRD#** signal when accessing guest N.
- T_{grecN} is the minimum recovery time in which **GRD#** and **GWR#** must be non active after the rising edge of the previous access (read or write) to guest N.

T_{gdur} and T_{grec} are configured by the host CPU in units of PCI clocks (3,4,12 or 15 PCI clocks are the possible values).

Additional timing parameters are specified in 12.3.3 "GuestBus Timing".

4.3.2 Code-Write Operations

Code-Write cycles are initiated by the **ZR36120** if all of the conditions below are met:

- The internal code buffer is not empty.
- A ***PostOffice*** request is not pending.
- The **GRDY** input is high ('1').

A code-write cycle consists of reading one code byte from the internal code buffer and writing it to the guest and address selected by the host for code-write cycles. The timing parameters of the code-write cycle are those programmed by the host for this specific guest device (the same parameters apply for ***PostOffice*** accesses to this guest).

4.3.2.1 Doubleword to Bytes Mapping in Code-Write Operations

The code is read in doublewords from main memory, and transferred in bytes to the guest device. The ordering of the bytes is that the least significant byte of a doubleword is the first one to be sent over the ***GuestBus***, and the most significant byte is the last one.

4.3.3 *PostOffice* Operations

When the ***PostOffice*** pending bit (POPen) in the ***PostOffice*** register is set to '1', the current code-write cycle (if such is executed) is completed, and a ***PostOffice*** cycle is executed, even when the internal code buffer is not empty. The type of the cycle (read or write) is determined by the ***PostOffice*** direction bit (PODir). The identity of the targeted guest and its specific register are also specified by the ***PostOffice*** register (POGuestID, POGuestReg). In both read and write cycles the timing parameters of the cycle are those configured by the host for the targeted guest. Upon completion of a ***PostOffice*** cycle the pending bit is set back to '0' by the **ZR36120**.

PostOffice Write

The ***GuestBus*** master transfers the 8 least significant bits of the ***PostOffice*** register (POData) out on the bus.

PostOffice Read

The ***GuestBus*** master reads from the specified target and writes the input byte into the 8 least significant bits of the ***PostOffice*** register (POData).

4.3.4 Guest Wait States

Slow guests that are equipped with a "bus hold" output can force a ***GuestBus*** cycle to extend by one or more additional PCI clocks, by asserting the **GWS#** signal right after the **ZR36120** asserts the **GRD#** or **GWR#** signal. **GWS#** is sampled with the PCI clock that follows the assertion of **GWR#** or **GRD#**. When **GWS#** is sampled high again, the cycle is completed. If T_{gdur} of the accessed guest has already expired since the assertion of **GRD#** or **GWR#**, the next PCI clock will latch the data (in case of a read) and **GRD#** (or **GWR#**) will be deasserted. The recovery portion of the cycle will then take place, and the cycle will be completed.

Insertion of wait states is possible both during code-write and ***PostOffice*** cycles.

The maximum number of PCI clock cycles allowed for **GRD#** or **GWR#**, **including wait-states**, is 32. If a guest holds the cycle until this limit expires, the **ZR36120** aborts the cycle. If the aborted cycle was a ***PostOffice*** one, the ***PostOffice*** time-out bit of the ***PostOffice*** register (POTime) is set to '1', and the ***PostOffice*** pending bit (POPen) is cleared. If the cycle was a code-write one, the code-write time-out flag (CodTime) is set to '1'.

Figure 3 shows two examples of ***GuestBus*** cycles. The upper one is a write to guest 0, register 0, followed by a read from guest 0, register 5. Note that for guest 0 T_{gdur0}=3 and T_{grec0}=4. The lower example shows a read from guest 2, register 1, with 3 wait-states inserted by the guest.

Notice that the assertions of **GADR** and **GCS#** are done together. The assertion of **GRD#** and **GWR#** is done one PCI clock after the assertion of **GADR** and **GCS#**. The deassertion of **GRD#** and **GWR#** is done one PCI clock before the deassertion of **GADR** and **GCS#**.

4.4 *PostOffice* Handshaking Protocol

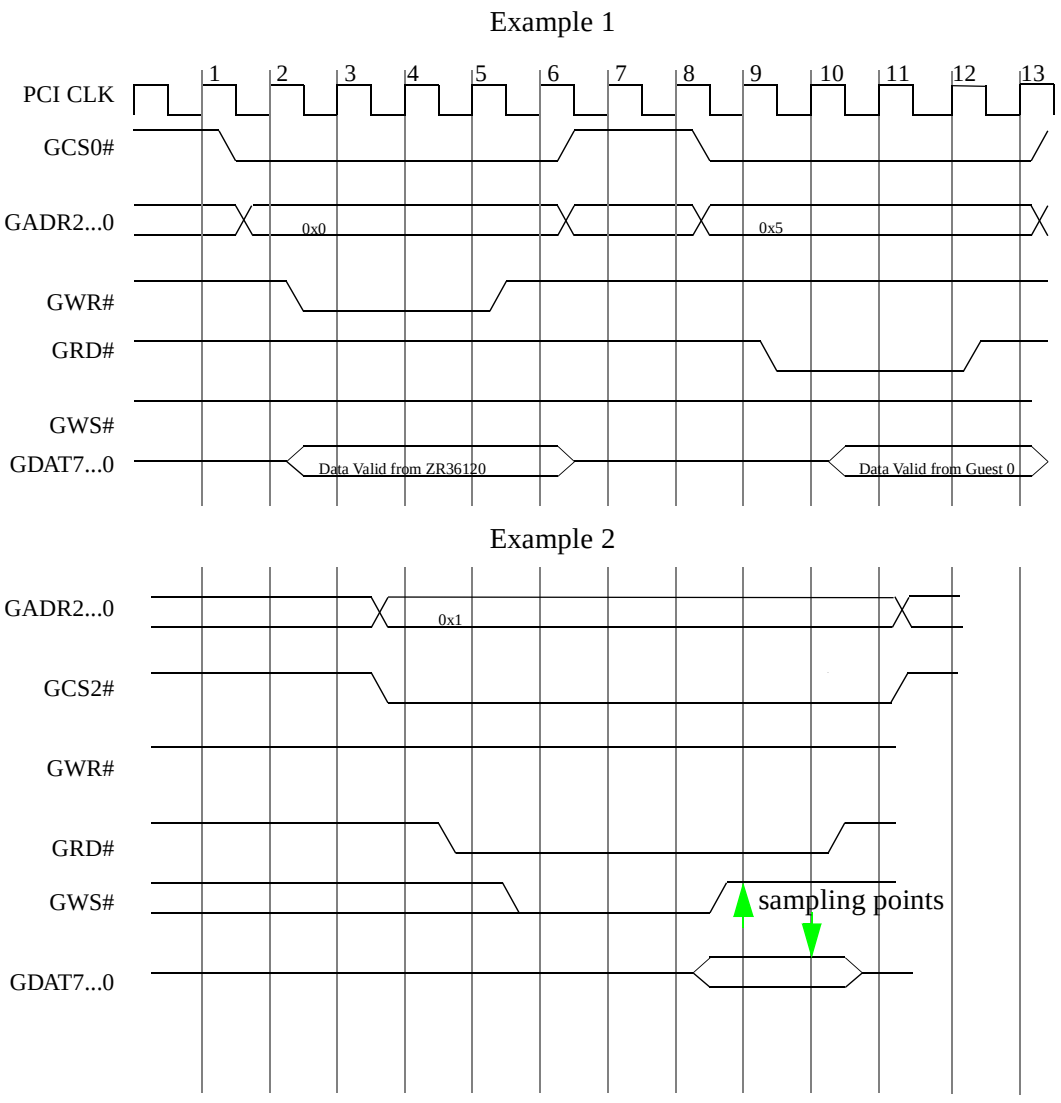
Reading data from- or writing data to any of the **ZR36120** guests using the ***PostOffice*** mechanism requires the host CPU to follow the handshaking protocol described below. The main idea is that the host CPU has to poll the ***PostOffice*** request pending bit in order to acquire the availability of the ***GuestBus*** and verify the validity of the data contained in the ***PostOffice*** data byte. In general, host CPU accesses to the ***PostOffice*** register may change the ***PostOffice*** pending bit, as explained below. Thus, the host CPU software must ensure that accesses to the ***PostOffice*** register are governed by a central routine. For example, independent accesses to the ***PostOffice*** register both from an interrupt service routine and the "main" processor task(s), or from more than one task, in a multitasking environment, might cause a deadlock, unless explicit protection measures are taken.

4.4.1 Host CPU Writes to a Guest Device

- The host reads the ***PostOffice*** register, and checks the ***PostOffice*** pending bit. If this bit is '1', the write cycle cannot be taken now, because the **ZR36120** is busy executing a previous ***PostOffice*** read or write request. Once this bit is '0', the write request can be made.
- The host writes a full doubleword to the ***PostOffice*** register, containing the data byte to be sent to the guest, the guest's identity (0,1,2, or 3), the specific guest register (0,...,7), and an indication that this is a write request (direction bit=1). As a result of writing to the data byte of the ***PostOffice*** register, the ***PostOffice*** pending bit is set to '1'.

- The **ZR36120** completes the current code-write cycle, if such is being executed, and, before executing the next code-write cycle, it executes the pending ***PostOffice*** request. At the completion of the ***GuestBus*** write cycle it clears the request pending bit.
- The host may read the ***PostOffice*** register, to verify that the pending bit is '0', meaning that the write request has been completed.
- Note that in multiple (back-to-back) ***PostOffice*** operations the host has to poll the request pending bit only once between two requests, since reading this bit zero indicates both that the previous request has been completed and that the next request can be made.

Figure 3:Two examples of *GuestBus* cycles



4.4.2 Host CPU Reads from a Guest Device

- The host reads the ***PostOffice*** register, and checks the ***PostOffice*** pending bit. If this bit is '1', the read cycle cannot be taken now, because the **ZR36120** is busy executing a previous ***PostOffice*** read or write request. Once this bit is '0', the read request can be made.
- The host writes a full doubleword to the ***PostOffice*** register, containing the guest's identity (0,1,2, or 3), the specific guest register (0,...,7), and an indication that this is a read request (direction bit=0). The data portion of the doubleword is meaningless, but should be set to a byte of zeros. As a result of writing to the data byte of the ***PostOffice*** register, the ***PostOffice*** pending bit is set to '1'.
- The **ZR36120** completes the current code-write cycle, if such is being executed, and, before executing the next code-write cycle, it executes the pending ***PostOffice*** request. It transfers the byte read from the guest to bits 7...0 of the ***PostOffice*** register. At the completion of the ***Guest-Bus*** read cycle it clears the request pending bit.
- The host may read the ***PostOffice*** register, to verify that the pending bit is '0', meaning that the read request has been completed and the data portion of the ***PostOffice*** register is the result.
  Note that in multiple (back-to-back) ***PostOffice*** operations the host has to poll the request pending bit only once between two requests, since reading this bit zero indicates both that the previous request has been completed and that the next request can be made.

4.5 I^2C Bus Interface

The I^2C port of the **ZR36120** consists of a clock signal, **SCL**, and data signal, **SDA**. Both have two possible levels: active low or passive tri-state. This configuration lets the **ZR36120** be the **only** master of an I^2C clock. Both lines must be pulled-up externally. By accessing the two **ZR36120** register bits that control **SDA** and **SCL**, the host CPU can generate valid I^2C start and stop conditions, and write (or read) address and data, bit by bit.

4.6 General Purpose I/O Pins

The **ZR36120** has 8 general purpose I/O pins, fully controlled by the host. Each one of these pins can be separately configured as input or output. When configured as an output the host can force its level through the corresponding register bit.

4.7 Interrupt Requests

The **ZR36120**'s interrupt manager connects the various conditions that may generate an interrupt request, enables/disables them according to the Interrupt Control Register, and drives the **INTA#** output. It stores the corresponding status bits in the Interrupt Status Register, and clears the status bits per host instructions.

The **ZR36120** can associate any one of the following events to an interrupt request:

- A positive edge on the GIRQ1 input pin.
- A positive edge on the GIRQ0 input pin.
- The code memory buffer pointer passing one of its report points.

Each one of these events can be separately enabled/disabled through the corresponding bit in the Interrupt Control Register. An additional bit may disable/enable **all** interrupts.

When an interrupt-associated event occurs, two things happen:

- The corresponding bit in the ISTR is set to one.
- If the interrupt is enabled, and the interrupts are enabled in general, then the **INTA#** open-drain output pin is asserted to its active-low level.

Both the status bit(s) and **INTA#** remain active, until the host CPU clears those status bits that are currently set. This is done by writing a '1' to these bits. When the host does that, the **INTA#** output signal is returned to its passive, tri-state level.

If any of the Interrupt Status Register bits is attempted to be cleared at the same time that the interrupt logic attempts to set it (because of an interrupt event), the set operation has priority over the clear operation.

5. Video Input Processor

5.1 Horizontal Filter

Prior to a significant horizontal down scaling of the input image, it is advised to apply one of the possible horizontal filters. The filter type is selected through the HFilter parameter.

HFilter = 0 --> Filter 1: No luminance filter, 3-tap pre-interpolation filter of chrominance.

HFilter = 1 --> Filter 2: 3-tap luminance filter, 3-tap pre-interpolation chrominance filter.

HFilter = 2 --> Filter 3: 4-tap luminance filter, 4-tap chrominance filter.

HFilter = 3 --> Filter 4: 5-tap luminance filter, 4-tap chrominance filter.

HFilter = 4 --> Filter 5: 4-tap luminance filter, 4-tap chrominance filter.

5.2 Horizontal/Vertical Downscaler

The horizontal and vertical down scalers are independent of each other. The horizontal scaling ratio is configured through the HorDcm parameter. HorDcm/64 indicates the number of pixels to drop out of every 64 consecutive pixels. HorDcm ranges from 0 to 63, where 0 represents the no scaling configuration (1:1 input to output ratio).

The vertical scaling ratio is configured through the VerDcm parameter. VerDcm/64 indicates the number of lines to drop out of every 64 consecutive lines. VerDcm ranges from 0 to 63, where 0 represents the no scaling configuration (1:1 input to output ratio).

The vertical downscaler can operate in two ways. If DupFld=0 it treats the top and bottom fields the same way. If DupFld=1 it uses different line dropping topologies for the top and the bottom fields, such that if the fields are equal (one field is actually duplicated, like the output of most MPEG1 decoders), then the total loss of information is minimized. For example, when the video source is the SAA7111 video decoder, it is recommended to apply DupFld=0, and when the video source is the **ZR36110**, and the CCIR size is down scaled by half or more, it is recommended to apply DupFld=1.

5.3 Color Space Converter

The color space converter converts the YUV input to RGB format. The YUV2RGB parameter determines the type of conversion:

YUV2RGB = 00b --> no conversion, output format is YUV 422.
YUV2RGB = 01b --> conversion to RGB 8:8:8 (24-bit output)
YUV2RGB = 10b --> conversion converted to RGB 5:6:5
YUV2RGB = 11b --> conversion converted to RGB 5:5:5.

When the 15- or 16-bit RGB format is selected, it is advised to apply the error diffusion option, in order to eliminate false contours from the output image. This option is determined by the ErrDif parameter (1 turns the error diffusion option on, while 0 turns it off).

6. Video Output Control

The **ZR36120** outputs the video pixels over the PCI bus, using DMA bursts. The **ZR36120**'s Video DMA Controller initiates and controls the DMA bursts. In order to enable the DMA Controller, the Master Enable bit of the PCI configuration space must be set to '1', and the VidEn bit in the Video Display Configuration Register must also be set to '1'. Once VidEn=='1', the CPU is not allowed to change registered parameters that are involved in the video processing. The register description (10. "Application-Specific Registers") specifies the conditions under which each parameter is allowed to be modified.

The **ZR36120** transfers the video to a rectangle in the display (or system) memory, defined by a base address for each field (MaskTopBase, MaskBotBase), an inter-line stride (DispStride), and the rectangle height (VidWinHt) and width (VidWinWid). Obviously, these parameters must be provided by the host prior to enabling the Video DMA Controller.

6.1 Display Modes

The **ZR36120** can either display both fields, emulating the interlaced input, or only the top field. The latter option has the advantage of reducing the motion artifacts that might be exhibited on live video. The parameter that controls the display mode is DispMod.

By a proper configuration of the display base addresses it is also possible to display two fields (from either one or two separate video sources) on two separate rectangles (video windows).

6.2 Frame Grabbing

The **ZR36120** has a special mode for capturing video frames (or fields) and storing them in system memory. This mode is invoked by setting the SnapShot parameter to '1'. When in this mode, every time the host CPU switches the FrameGrab bit from '0' to '1', the **ZR36120** downloads a frame (or a field, if DispMod==0), to memory.

Following is an **example** of a flow of actions aimed to grab one frame. The example assumes that the vertical sync is used as an interrupt source (by externally tying VSYNC to GIRQ0 or GIRQ1), and that prior to grabbing the frame, the **ZR36120** operates in the "normal" continuous scheme of live video display.

- Through a push-button click in the application GUI the user triggers a frame grabbing request.
- The host sets SnapShot=1. The **ZR36120**'s VIdeo DMA Controller waits for the next **VSYNC** and then freezes the live display. Since now SnapShot=1 and FramGrab=0, video parameters can be changed (even without VidEn=0; refer to 10. "Application-Specific Registers").
- The host sets new addresses in VidTopBase and VidBotBase. These addresses point to main memory. DispStride is also given a new value. If needed, other video parameters can be changed now, e.g., pixel format, etc.

- The host sets FrameGrab=1. The **ZR36120** waits until the next **VSYNC** and then transmits two consecutive fields to main memory.
- After the second of the two fields is completed, FrameGrab is cleared by the **ZR36120**.
- When the host senses (after constant polling or polling inside VSYNC-triggered interrupts) that FrameGrab=0 again, it sets the old addresses back to VidTopBase and VidBotBase. DispStride is given back its old value. The rest of the previous video parameters can be restored now.
- The host sets SnapShot=0, putting the **ZR36120** back to the continuous mode.
- With the next **VSYNC** the **ZR36120**'s Video DMA Controller resumes "normal" live display operation.

6.3 Output Pixel Organization

The output pixel format is determined by the following parameters: YUV2RGB, Pack24 (applicable only to RGB 888 format), and LittleEndian (applicable to all formats, excluding the 24-bit packed). Following are the bit organizations of the different pixel formats when a video doubleword is transferred over the PCI bus:

Table 8: YUV 4:2:2 Pixel Format

Endian-ness	bits 31...24	bits 23...16	bits 15...8	bits 7...0
Little Endian	$Y_1$7...0	$V_0$7...0	$Y_0$7...0	$U_0$7...0
Gib Endian	$V_0$7...0	$Y_1$7...0	$U_0$7...0	$Y_0$7...0

Table 9: RGB 5:5:5 Format

Endian-ness	bits 31...24	bits 23...16	bits 15...8	bits 7...0
Little Endian	0,$R_1$4...0, $G_1$4...3	$G_1$2...0, $B_1$4...0	0,$R_0$4...0, $G_0$4...3	$G_0$2...0, $B_0$4...0
Gib Endian	$G_1$2...0, $B_1$4...0	0,$R_1$4...0, $G_1$4...3	$G_0$2...0, $B_0$4...0	0,$R_0$4...0, $G_0$4...3

Table 10: RGB 5:6:5 Format

Endian-ness	bits 31...24	bits 23...16	bits 15...8	bits 7...0
Little Endian	$R_1$4...0, $G_1$5...3	$G_1$2...0, $B_1$4...0	$R_0$4...0, $G_0$5...3	$G_0$2...0, $B_0$4...0
Gib Endian	$G_1$2...0, $B_1$4...0	$R_1$4...0, $G_1$5...3	$G_0$2...0, $B_0$4...0	$R_0$4...0, $G_0$5...3

Table 11: 24-bit Unpacked Format

Endian-ness	bits 31...24	bits 23...16	bits 15...8	bits 7...0
Little Endian	0x0	R7...0	G7...0	B7...0
Gib Endian	B7...0	G7...0	R7...0	0x0

Table 12: 24-bit Packed Format

Bus Cycle	bits 31...24	bits 23...16	bits 15...8	bits 7...0
First	$B_1$7...0	$R_0$7...0	$G_0$7...0	$B_0$7...0
Second	$G_2$7...0	$B_2$7...0	$R_1$7...0	$G_1$7...0
Third	$R_3$7...0	$G_3$7...0	$B_3$7...0	$R_2$7...0

In the 24-bit packed format the first active pixel in a line is always packed as indicated in the first entry line of Table 12 . From then on the byte organization is as described by the table.

7. Graphics Overlay

The **ZR36120**'s Video DMA Controller is capable of masking off (i.e., not transmitting) pixels that are marked by '0' in a masking map prepared and maintained by the driver software. The masking feature, referred to as overlay, is turned on by setting the OvlEnable parameter to '1'. As long as OvlEnable='0', all pixels within the selected portion of the image are transferred to destination. The masking map is a one bit deep pattern of the video rectangle. Its location in system memory is defined by a couple of base addresses (one per each field - MaskTopBase, MaskBotBase), and an inter-line stride (MaskStride). The width of the map must be doubleword aligned. Thus, the line size is
int((VidWinWid+31) >> 5)
and the number of lines in the map is
2*VidWinHt, if DispMod==0, or VidWinHt, if DispMod==1.

In order to match the 0/1 values of the map to their corresponding pixels in the video rectangle, the map must follow the format given in Table 13 :

Table 13: Bit, Byte and Doubleword Organization of Masking Map

DW	...1	0			
Byte	...0	3	2	1	0
Bit	7....0	31...24	23...16	15... 8	7...0
Pixel index in line	39...32	31...24	23...16	15...8	7...0

8. Reset

There are two means of resetting the **ZR36120**. One is a hardware reset, which is applied through the **PCIRST#** input, and the other is a software reset, which is applied through the SoftReset register bit.

8.1 Hardware Reset

The hardware reset signal **PCIRST#** resets the internal state machines in the **ZR36120** and loads all registers with their default states. The reset state of the PCI interface pins is as defined by the PCI specifications (2.1). The reset state of the other output/bidirectional signals is as follows.

The **GPIO7...0** lines are all inputs after reset. If required for system purposes, they can be pulled high or low through 1 kΩ resistors to have fixed values on reset.

As long as **PCIRST#** is asserted, the following signals are tri-stated: **GDAT7:0**, **GADR2:0**, **GCS#3:0**, **GWR#**, **GRD#**, **SDA**, **SCL**. Once the **PCIRST#** input is deasserted, these signals go to their software reset condition (as does the entire device).

A hardware reset asserts (clears) the SoftReset bit in the system register. After the hardware reset is over, the **ZR36120** will be in software reset condition until the SoftReset bit is deasserted.

8.2 Software Reset

There are two ways in which the **ZR36120** can go into the software reset condition: one is right after hardware reset (i.e., upon the low to high transition of **PCIRST#**), the other is by clearing the SoftReset bit.

While in software reset all registers and state machines in the device are reset to their default values/states, except the Soft-Reset bit itself, and the PCI Interface (including the PCI configuration space registers).

The device continues to respond according to the PCI Specification and can be the target of a PCI transfer targeted at the ASRs (Application Specific Registers) or PCI Configuration Space. While in software reset the device will not initiate any PCI transfers, because all DMA channels are disabled.

After the SoftReset bit is deasserted, all registers obtain their default values, all DMA channels remain disabled and all ASRs are programmable according to their “normal” modification conditions.

Hence, the initialization of the device (loading all registers with the values required for the specific application) must start with setting SoftReset to ‘1’, otherwise new ASR values will not be latched in.

9. PCI Configuration Space Registers

The PCI specification 2.0 requires that a PCI device includes a configuration register space, a set of 256 8-bit configuration registers. The first 64 bytes make up the configuration header, predefined by the specification and the remaining are application specific. These registers allow device relocation, device independent system address map construction and automatic configurations. The configuration registers are accessed by the host CPU through two consecutive doubleword I/O accesses to physical addresses 0x0CF8 and 0x0CFC, respectively. The host PCI/bridge is responsible for translating these accesses to a PCI configuration cycle, including the assertion of the **IDSEL** input. The **ZR36120** then responds to these cycles. This section details the **ZR36120**'s PCI configuration registers. PCI configuration accesses to **ZR36120** configuration addresses that are not explicitly described here return zeros (in reads).

9.1 Vendor and Device ID Register

This read-only 32 bit register identifies the device and the device manufacturer.

Address: 0x00

Bit	Type	Description
31:16	R	Device ID. Hardwired to 0x6120.
15:0	R	Vendor ID. Hardwired to 0x11DE.

9.2 Command and Status Register

This read-write 32 bit register controls the generation of and reaction to PCI Bus commands. It also reflects the status of the device regarding the PCI Bus.

Address: 0x04

Bit	Type	Description
31	RC	Parity Error Detected. This bit is set when a parity error is detected, regardless of the Parity Error Response bit.
30	R	System Error Signaled. Hardwired to '0'.
29	RC	Master Abort Detected. This bit is set when a master-abort condition has been detected.
28	RC	Target Abort Detected. This bit is set when a target-abort condition has been detected.
27	RC	Target Abort Signaled. When the **ZR36120** terminates a transaction as a target (e.g., due to wrong address parity) it sets this bit.
26:25	R	**DEVSEL#** Timing. Hardwired to '00' ("fast" timing, i.e., **DEVSEL#** is asserted before rising edge of clock three within a cycle).
24	R	Data Parity Reported. Hardwired to '0'.
23	R	Fast Back-to-Back Capability. Hardwired to '0'.
22:16	R	Reserved. Returns zeros.
15:10	R	Reserved. Returns zeros.
9	R	Fast Back-to-Back Enable. Hardwired to '0'.
8	R	System Error Enable. Hardwired to '0'.
7	R	Wait Cycle (Stepping) Enable. Hardwired to '0'.
6	R	Parity Error Response. Hardwired to '0'.
5:3	R	Unused. Hardwired to '0'.
2	RW	Master Enable. When this bit is set to '1' the **ZR36120** can operate as a bus master. Default is '0'.
1	RW	Memory Access Enable. When this bit is set to one the device responds to PCI memory accesses. Default value is '0'.
0	R	I/O Access Enable. Hardwired to '0'.

9.3 Class Code and Revision ID Register

This read-only 32 bit register identifies the revision level of the device and its class code.

Address: 0x08

Bit	Type	Description
31:8	R	Class Code. Returns 0x040000 (Multimedia Video Device)
7:0	R	Revision ID. Hardwired to 0x**02** (0x01 in ES).

9.4 Latency Timer Register

This read-write 32 bit register defines the maximum latency timer limit.

Address: 0x0C

Bit	Type	Description
31:24	R	Unused. Return zeros.
23:16	R	Header Type. Returns zeros.
15:8	RW	Master Latency Timer. The number of PCI clocks that limit **ZR36120**-initiated bursts in case **GNT#** is deasserted by the bus arbiter during the **ZR36120**-initiated burst. The 3 LS bits are read-only zeros. The default value is 0x00.
7:0	R	Unused. Returns zeros.

9.5 Memory Base Address Register

This read-write 32 bit register identifies the base address of the device.

Address: 0x10

Bit	Type	Description
31:12 11:0	RW R	This value determines the base address of the **ZR36120** as a memory-mapped device. The **ZR36120** occupies a range of 4096 bytes out of the memory map: Bits 11:0 are hardwired to '0'. The default value of all other bits is '0'.

9.6 Miscellaneous Functions Register

This read-write 32 bit register defines the latency timer requirements of the device and indicates the interrupts that will be used.

Address: 0x3C

Bit	Type	Description
31:24	R	Max_Lat - Hardwired to 0x10 (i.e. 4us). This value indicates for the operating system how often the device needs access to the PCI bus.
23:16	R	Min_Gnt - Hardwired to 0x02 (i.e. 0.5us). It indicates to the operating system the minimum length of a burst.
15:8	R	Interrupt pin. Hardwired to 0x1, indicating that **INTA#** is used.
7:0	RW	Interrupt Line. These bits indicate the interrupt line that is being used (e.g. IRQ10 ==> 0xA, etc.). Default value is 0xA.

10. Application-Specific Registers

The **ZR36120** application-specific registers (ASRs) are memory-mapped. Their base address is configured by the host into the PCI configuration address 0x10. The **ZR36120** claims a contiguous range of 4K bytes of memory. PCI memory-read accesses to addresses (within the 4K bytes range) that are not explicitly described in this section return zeros.

The ASRs can be accessed in any byte combination.

The column “Mod” of the following tables defines the condition(s) under which each parameter of the ASRs is allowed to be modified by the software. The following abbreviations have been defined:

- all this parameter may be modified on the fly, i.e. anytime.
- res this parameter is set once after a reset of the **ZR36120**, no modifications during operation.
- vid this parameter may be modified if either VidEn = ‘0’ or SnapShot = ‘1’ and FrameGrab = ‘0’.
- cod this parameter may be modified if CodReadEn = ‘0’.
- snap this parameter may be modified if SnapShot = ‘1’.

Note that after a reset VidEn and CodReadEn are deasserted - all parameters are allowed to be modified.

10.1 Video Front End Horizontal Configuration Register

This 32 bit register contains the horizontal configuration parameters of the video source.

Address Offset: 0x000

Bit	Type	Mod	Description
31	R		Reserved. Returns ‘0’.
30	RW	vid	**HSPol** - **HSYNC** Polarity. HStart and HEnd will be counted from the active edge of **HSYNC**. ‘1’ - negative edge of **HSYNC**. ‘0’ - positive edge of **HSYNC** (default value).
29:20	R		Reserved. Returns zero.
19:10	RW	vid	**HStart** - Horizontal Start Offset. Number of pixel clocks in a line from the active edge of **HSYNC** until the first pixel to be sampled. Default value is 0x001.
9:0	RW	vid	**HEnd** - Horizontal End Offset. Number of pixel clocks in a line from the active edge of **HSYNC** until the last pixel to be sampled. Default value is 0x3FF.

10.2 Video Front End Vertical Configuration Register

This 32 bit register contains the vertical configuration parameters of the video source.

Address Offset: 0x004

Bit	Type	Mod	Description
31	R		Reserved. Returns ‘0’.
30	RW	vid	**VSPol** - **VSYNC** Polarity. VStart and VEnd will be counted from the active edge of **VSYNC**. ‘1’ - negative edge of **VSYNC**. ‘0’ - positive edge of **VSYNC** (default value).
29:20	R		Reserved. Returns zero.
19:10	RW	vid	**VStart** - Vertical Start Offset. Number of lines from the active edge of **VSYNC** until the first line to be sampled. Default value is 0x001.
9:0	RW	vid	**VEnd** - Vertical End Offset. Number of lines from the active edge of **VSYNC** until the last line to be sampled. Default value is 0x3FF.

10.3 Video Front End, Scaler and Pixel Format Register

This register contains the video front end configuration (byte 3), video scaler (bytes 2-1) and pixel formatter (byte 0) parameters.

Address Offset: 0x008

Bit	Type	Mod	Description
31:27	R		Reserved. Returns zero.
26	RW	vid	**ExtFI** - External Field Indication. ‘1’ - the video source provides an **FI** signal. ‘0’ - field indication is derived from **HSYNC** and **VSYNC** (default value).
25	RW	vid	**TopField** - Top Field Interpretation. If field indication is derived from the **FI** input signal (see ExtFI), then this parameter defines which level of **FI** indicates the top field: ‘1’ - **FI** high indicates the top field (default value). ‘0’ - **FI** low indicates the top field. If field indication is derived internally from **HSYNC** and **VSYNC**, then this parameter defines which level of **HSYNC** indicates the top field on the active edge of **VSYNC**: ‘1’ - **HSYNC** high indicates the top field (default value). ‘0’ - **HSYNC** low indicates the top field.
24	RW	vid	**VCLKPol** - Video Clock Polarity. When set to ‘1’, the sampling edge of **VCLKx2** is the positive edge that is qualified by **VCLK** = ‘1’. When set to ‘0’, the qualification condition is **VCLK** = ‘0’ (default value).

Address Offset: 0x008

Bit	Type	Mod	Description
23:21	RW	vid	**HFilter** - Horizontal Filter Selection. The following horizontal filters can be selected: 000b - Filter 1 = no luminance, 3-tap chrominance, 001b - Filter 2 = 3-tap luminance, 3-tap chrominance, 010b - Filter 3 = 4-tap luminance, 4-tap chrominance, 011b - Filter 4 = 5-tap luminance, 4-tap chrominance, 100b - Filter 5 = 4-tap luminance, 4-tap chrominance. 101b - 111b will result in the default filter. Default value 000b.
20	RW	vid	**DupFld** - Duplicate Field. This parameter has an effect on the vertical decimation of a frame in respect to the video source. The result is less loss of information and less distortion. '1' - indicates top and bottom fields are equal. '0' - indicates an interlaced video source, top and bottom fields are different (default value).
19:14	RW	vid	**HorDcm** - Horizontal Decimation Ratio. This parameter defines the number of pixels to be dropped out of every 64 consecutive pixels. 000000b - no horizontal decimation (default value).
13:8	RW	vid	**VerDcm** - Vertical Decimation Ratio. This parameter defines the number of lines to be dropped out of every 64 consecutive lines. 000000b - no vertical decimation (default value).
7	R		Reserved. Returns zero.
6	RW	vid	**DispMod** - Display Mode. '1' - single field. '0' - emulated interlaced video (default value).
5	R		Reserved. Returns zero.
4:3	RW	vid	**YUV2RGB** - YUV to RGB Conversion. This parameter defines the pixel output format: 00b - YUV 4:2:2, 01b - RGB 8:8:8, 10b - RGB 5:6:5 (default value), 11b - RGB 5:5:5.
2	RW	vid	**ErrDif** - Error Diffusion. This parameter has an effect only in RGB 5:6:5 and 5:5:5 output formats. '1' - error diffusion is active '0' - simple truncation of RGB 8:8:8 is executed (default value).

Address Offset: 0x008

Bit	Type	Mod	Description
1	RW	vid	**Pack24** - 24 Bit Packed Format. This bit is applicable only if YUV2RGB = 01b. '1' - RGB 8:8:8 is packed such that it takes 3 PCI cycles to transfer 4 pixels. '0' - one pixel per PCI cycle is transferred (default value).
0	RW	vid	**LittleEndian** - Little Endian Format flag. '1' - pixel layout on PCI is little endian (default value). '0' - pixel layout on PCI is gib endian.

10.4 Video Display "Top" Register

This register contains the DWORD base address of the top field.

Address Offset: 0x00C

Bit	Type	Mod	Description
31:2 1:0	RW R	vid	**VidTopBase** - Video Top Field Base Address. This is the destination starting address for the top field. Default value is 0xFFFFFFFC. Bits 1...0 are hardwired to 00b.

10.5 Video Display "Bottom" Register

This register contains the DWORD base address of the bottom field.

Address Offset: 0x010

Bit	Type	Mod	Description
31:2 1:0	RW R	vid	**VidBotBase** - Video Bottom Field Base Address. This is the destination starting address for the bottom field. Default value is 0xFFFFFFFC. Bits 1...0 are hardwired to 00b.

10.6 Video Stride, Status and FrameGrab Register

This register contains parameters for display addressing (bytes 2-3), status of internal pixel buffer (byte 1) and frame grab control (byte 0).

Address Offset: 0x014

Bit	Type	Mod	Description
31:18 17:16	RW R	vid	**DispStride** - Display Stride. This register defines the address increment in bytes to be added to the address of the last pixel of a display line, to generate the address of the next consecutive display line. Default value is 0xFFFC. Bits 1...0 are hardwired to 00b.
15:9	R		Reserved. Returns zero.

Address Offset: 0x014

Bit	Type	Mod	Description
8	RC	all	**VidOvf** - internal pixel buffer overflow flag. **This bit is asserted by the** internal pixel buffer **Server when an overflow of the** internal pixel buffer **occurs. This bit is cleared when the host tries to write '1' to it.** **In case of concurrent accesses to this bit, it remains '1'.** **'1' - an** internal pixel buffer **overflow occurred.** **'0' - no overflow (default value).**
7:2	R		Reserved. Returns zero.
1	RW	all	**SnapShot** - Frame Grabbing Mode. If this bit is asserted the **ZR36120** goes into frame grab mode. When deasserted continuous display of video is resumed. '1' - frame grab mode. '0' - continuous display mode (default value).
0	RS	snap	**FrameGrab** - Frame Grabbing Command/ Status. When this bit is asserted by the host and SnapShot is asserted, the **ZR36120** will transfer the next two fields (indicated by the **VSYNC** signal) to memory. At the end of the second field this bit will be cleared internally, indicating that the frame grabbing has been completed and video transfer has been stopped. In case of concurrent accesses to this bit, the result is '0'. '1' - start frame grabbing. '0' - frame grabbing completed (default value).

10.7 Video Display Configuration Register

This register contains the configuration parameters for the video display.

Address Offset: 0x018

Bit	Type	Mod	Description
31	RW	all	**VidEn** - Video Display Enable. If this bit is cleared by the host, video write DMA transfers are disabled. When enabled, the video DMA controller operates normally. '1' - normal video transfer mode. '0' - video write transfers disabled (default value).
30:25	RW	vid	**MinPix** - Pixel FIFO Threshold. This parameter indicates the number of DWORD pairs that define the threshold. When the number of DWORD pairs inside the internal pixel buffer has reached this value a video-write burst is requested. Default value is 0x07 (7 pairs=14 DWORDs). Range: 0x01 - 0x1E.

Address Offset: 0x018

Bit	Type	Mod	Description
24	RW	res	**TriCom** - **Triton Compatibility**. This bit must be cleared ('0') if the host/PCI bridge is Intel's Triton (82437FX). Otherwise it must be set to '1'.
23:22	R		Reserved. Returns zero.
21:12	RW	vid	**VidWinHt** - Video Window Height. This register defines the number of lines that should be displayed by the **ZR36120**. If DispMod = 0, VidWinHt is half the number, if DispMod = 1, it is the entire number of display lines. Default value is 0x0F0.
11:10	R		Reserved. Returns zero.
9:0	RW	vid	**VidWinWid** - Video Window Width. This register defines the width of the video window in number of pixels. Default value is 0x3FF.

10.8 Masking Map "Top" Register

This register contains the DWORD base address of the top masking map.

Address Offset: 0x01C

Bit	Type	Mod	Description
31:2 1:0	RW R	vid	**MaskTopBase** - Masking Map Top Base Address. This is the source starting address of the top field for the masking map read transfers. Default value is 0xFFFFFFFC. Bits 1...0 are hardwired to 00b.

10.9 Masking Map "Bottom" Register

This register contains the DWORD base address of the bottom masking map.

Address Offset: 0x020

Bit	Type	Mod	Description
31:2 1:0	RW R	vid	**MaskBotBase** - Masking Map Bottom Base Address. This is the source starting address of the bottom field for the masking map read transfers. Default value is 0xFFFFFFFC. Bits 1...0 are hardwired to 00b.

10.10 Overlay Control Register

This register contains parameters controlling overlay (byte 1) and masking map addressing (byte 0).

Address Offset: 0x024

Bit	Type	Mod	Description
31:16	R		Reserved. Returns zero.
15	RW	vid	**OvlEnable** - Overlay Enable flag. When enabled the masking information in the video mask is evaluated to allow overlay of other windows or graphics. When disabled the video window is always on top. '1' - overlay enabled, '0' - overlay disabled (default value).
14:8	R		Reserved. Returns zero.
7:0	RW	vid	**MaskStride**. This register defines the address increment in DWORDs that is needed to get from the end of a mask line to the beginning of the next. Default value is 0xFF.

10.11 System, PCI and General Purpose Pins Control Register

This register contains the software reset bit (byte 3), a PCI core control (byte 2) and the General Purpose Pins direction parameter (byte 0).

Address Offset: 0x028

Bit	Type	Mod	Description
31:25	R		Reserved. Returns zero.
24	RW	all	**SoftReset** - Software Reset. This bit is asserted by the host to reset the **ZR36120**. If this bit is set to '0' all registers in the device will be reset to their default value, except: - the **SoftReset** bit, - the PCI interface. The device continues to respond according to the PCI Specification and can be the target of a PCI transfer targeted at the ASRs or configuration. space. The transfer (single- or burst-write) asserting this bit will not be terminated abnormally. The device will not initiate any PCI transfers during reset, because all DMA channels are disabled. This register continues to be programmable. During reset, actually, only this bit of the ASRs can be modified by the host (e.g. turning reset off). A power on (hardware) reset also asserts this bit. After the hardware reset is over, a large portion of the **ZR36120** (see above) will remain in reset mode until the **SoftReset** bit is deasserted. After this bit is deasserted, all registers display their default value, all DMA channels remain disabled and all ASRs are programmable respectively of their modification condition. '1' - no reset, '0' - reset (default value after power on).
23:19	R		Reserved. Returns zero.
18:16	RW	all	**WaitState** - PCI Wait State Control. This parameter defines the number of wait states inserted by the PCI slave logic. During each PCI transfer cycle to the **ZR36120**, the device will deassert **TRDY#** according to this value. Default value 000b.
15:8	R		Reserved. Returns zero.
7:0	RW	res	GenPurDir - General Purpose Pins Direction. These eight bits define the direction of the **GPIO7...0** pins, respectively. A '1' defines the corresponding pin as an input, a '0' as an output. Default value is 0xFF (all inputs).

10.12 General Purpose Pins and GuestBus Control Register

This register contains the values for General Purpose outputs (byte 3) and the GuestBus timing parameters (bytes 1-0).

Address Offset: 0x02C

Bit	Type	Mod	Description
31:24	RW	all	**GenPurIO** - General Purpose Input/Output. The function of this register depends on the setting of GenPurDir. For each pin configured as input (default): - reading this bit will return the current value of the input pin. - writing to an input has no meaning, no change of that bit. For each pin configured as output: - reading this bit will return the value on the corresponding output pin. (If there is no external shortcut, this is the last value written by the host.) - writing will change the output to the value specified. Default value is 0xF0.
23:16	R		Reserved. Returns zero.
15:14	RW	res	duration time for guest 3: 00b - T_{dur3} = 3 PCI clocks (default value), 01b - T_{dur3} = 4 PCI clocks, 10b - T_{dur3} = 12 PCI clocks, 11b - T_{dur3} = 15 PCI clocks.
13:12	RW	res	recovery time for guest 3: 00b - T_{rec3} = 3 PCI clocks (default value), 01b - T_{rec3} = 4 PCI clocks, 10b - T_{rec3} = 12 PCI clocks, 11b - T_{rec3} = 15 PCI clocks.
11:8	RW	res	duration and recovery times for guest 2 (same structure as defined for guest 3 above).
7:4	RW	res	duration and recovery times for guest 1 (same structure as defined for guest 3 above).
3:0	RW	res	duration and recovery times for guest 0 (same structure as defined for guest 3 above).

10.13 Code Source Address Register

This register contains the DWORD base address for code DMA transfers.

Address Offset: 0x030

Bit	Type	Mod	Description
31:2 1:0	RW R	cod	**CodMemBase** - Code Memory Base Address. This is the source starting address for the code-read DMA transfers. Default value is 0xFFFFFFFC. Bits 1.0 are hardwired to 00b.

10.14 Code Transfer Control Register

This register contains status and control bits and configuration parameters for code DMA transfers.

Address Offset: 0x034

Bit	Type	Mod	Description
31	R		Reserved. Returns zero.
30	RC	all	**CodTime** - Code-Write Time-out flag. This bit is set to '1' by the ***GuestBus*** master if a code-write cycle on the ***GuestBus*** lasts more than 32 PCI clocks. This might happen when the accessed guest holds **GWS#** low too long. It is cleared ('0') by the host writing a '1'. Priority is given to the ***GuestBus*** master in case of concurrent accesses to this bit. '1' - a code-write cycle has timed out. '0' - no time-out occurred (default value).
29	R	all	**CEmpty** - Code FIFO Empty. This bit reflects the status of the internal code buffer. When the buffer is empty this bit shows one, otherwise zero. '1' - buffer is empty (default value), '0' - buffer is not empty.
28	RW	cod	**CFlush** - Code FIFO Flush. This bit is used by the host to reset the internal code buffer. When it is asserted remaining code in the buffer is lost. After it is deasserted the buffer is empty and ready to receive data. '1' - flush internal code buffer (default value), '0' - normal operation of buffer.
27:22	R		Reserved. Returns zero.
21:20	RW	cod	**CodGuestID** - Code Guest Identification. These two bits select the guest to be accessed for code DMA transfers. Default after reset is 00b.
19	R		Unused. Returns '0'.
18:16	RW	cod	**CodGuestReg** - Code Guest Register. Register indication of accessed guest for code DMA transfers. Within each guest up to eight registers can be addressed. Default after reset is 000b.
15	R		Unused. Returns '0'.
14:12	RW	cod	**CodMemSize** - Code Memory Buffer Size. This value determines the size of the contiguous memory buffer allocated by the CPU for storage of coded data: 000b - 8 kbyte, 001b - 16 kbyte, 010b - 32 kbyte, 011b - 64 kbyte (default value), 100b - 128 kbyte, 101b - 256 kbyte. 110b - 111b will result in the default value.
11	R		Reserved. Returns '0'.

Address Offset: 0x034

Bit	Type	Mod	Description
10:8	RW	cod	**CodMemStep** - Code Memory Report Step. This value determines the amount of coded data, in quanta of 8 kbytes, after which the **ZR36120** notifies its position (within the buffer) to the host CPU, by requesting an interrupt. For proper operation, the buffer size must be greater or equal than the report step size: 000b - IRQ every 8 kbytes, 001b - IRQ every 16 kbytes (default value), 010b - IRQ every 32 kbytes, 011b - IRQ every 64 kbytes, 100b - IRQ every 128 kbytes, 101b - IRQ every 256 kbytes. 110b - 111b will result in the default value.
7	RW	all	**CodReadEn** - Code-Read Enable. If this bit is cleared by the host, the code-read transfers are stopped. The current code-read pointer retains its value. When this bit is set to '1', the **ZR36120** resumes code-read transfers. Default value is '0'.
6:4	R		Reserved. Returns zero.
3:1	RW	cod	**CodTrshld** - internal code buffer Threshold. If the fullness of buffer drops below this threshold value (in DWORDs) and CodRead-En is set to '1', a code-read burst is requested. Default value is 0x6.
0	RW	cod	**CodAutoEn** - Code-Read Auto Re-initialize Enable. If this bit is cleared, every time the code memory pointer reaches the end of the allocated space (i.e. CodMemBase plus CodMemSize) the code-read transfer is stopped. If the bit is set the code memory pointer is reset at the end of the allocated space and code-read transfers run in a cyclic manner. Default value is '0'.

10.15 Code Memory Pointer Register

This register contains the pointer to the code memory space of the host.

Address Offset: 0x038

Bit	Type	Mod	Description
31:16	R		Unused. Returns zero.
15:0	RW	cod	**CodMemPoint** - Code Memory Pointer. This register reflects the current position of the code memory pointer within the range of the allocated system memory. The value represents the number of DWORDs from the base address. Writing to this register by the host can be used for reset or moving the pointer inside the code memory space, only if CodReadEn is deasserted. A value pointing outside the memory size should not be used. Default value is zero.

10.16 Interrupt Status Register

This register contains the status of the interrupt sources.

Address Offset: 0x03C

Bit	Type	Mod	Description
31	R		Reserved. Returns zero.
30	RC	all	**GIRQ1** - **GIRQ1** Input Pin. A '1' indicates that a guest requested an interrupt on the **GIRQ1** input pin. This bit is cleared when the host tries to write '1' to it. In case of concurrent accesses to this bit, it remains '1'. Default value is '0' (no IRQ).
29	RC	all	**GIRQ0** - **GIRQ0** Input Pin. A '1' indicates that a guest requested an interrupt on the **GIRQ0** input pin. This bit is cleared when the host tries to write '1' to it. In case of concurrent accesses to this bit, it remains '1'. Default value is '0' (no IRQ).
28	RC	all	**CodRepIRQ** - Code Report Step Interrupt Request. A '1' indicates that the code memory buffer pointer has passed a report step. This bit is cleared when the host tries to write '1' to it. In case of concurrent accesses to this bit, it remains '1'. Default value is '0' (no IRQ).
27:0	R		Reserved. Returns zero.

10.17 Interrupt Control Register

This register contains the control byte for the interrupt handling.

Address Offset: 0x040

Bit	Type	Mod	Description
31	R		Reserved for future interrupt source. Returns '0'.
30	RW	all	**GIRQ1En** - **GIRQ1** Enable. When enabled and IntPinEn is set to '1', each positive edge of the **GIRQ1** input will generate an interrupt request on the PCI Bus **INTA#** output pin. When cleared, GIRQ1 continues to reflect the corresponding interrupt pin. '1' - GIRQ1 enabled, '0' - GIRQ1 disabled (default value).
29	RW	all	**GIRQ0En** - **GIRQ0** Enable. When enabled and IntPinEn is set to '1', each positive edge of the **GIRQ0** input will generate an interrupt request on the PCI Bus INTA# output pin. When cleared, GIRQ0 continues to reflect the corresponding interrupt pin. '1' - GIRQ0 enabled, '0' - GIRQ0 disabled (default value).
28	RW	all	**CodRepIrqEn** - Code Report Step Interrupt Enable. When enabled and IntPinEn is set to '1', an interrupt request will be generated on the PCI Bus **INTA#** output pin each time the code memory buffer pointer passes a report step. When cleared, **CodRepIrq** continues to reflect the internal report step interrupt request. '1' - interrupt request enabled, '0' - disabled (default value).
27:25	R		Reserved for future interrupt sources. Returns zero.
24	RW	all	**IntPinEn** - INTA# Pin Enable. When cleared, **non** of the events that may cause an interrupt request on the PCI Bus **INTA#** pin is enabled. Nevertheless the interrupt status register continues to reflect all interrupt input pins and internal interrupt requests. '1' - every interrupt request is passed onto the PCI Bus, '0' - **INTA#** disabled (default value).
23:0	R		Reserved. Returns zero.

10.18 I^2C-Bus Register

This register contains the control bits of the I^2C-Bus pins.

Address Offset: 0x044

Bit	Type	Mod	Description
31:2	R		Unused. Returns zero.
1	RW	all	**SDA** - I^2C-SDA Line. When the host writes '0' to this bit, the **SDA** output signal goes low. When the host writes '1', **SDA** goes into tri-state. When the host reads this bit it reflects the current level on the **SDA** pin. Default value is '1'.
0	RW	all	**SCL** - I^2C-SCL Line. When the host writes '0' to this bit, the **SCL** output signal goes low. When the host writes '1', **SCL** goes into tri-state. When the host reads this bit it reflects the current level on the **SCL** pin. Default value is '1'.

10.19 *PostOffice* Register

This register contains the status (byte 3), control (byte 2) and data (byte 0) parameters for ***PostOffice*** transfers.

Address Offset: 0x200 - 0x2FF

Bit	Type	Mod	Description
31:26	R		Reserved. Returns zero.
25	R	all	**POPen** - ***PostOffice*** Request Pending flag. This bit is set internally to '1' when the host CPU writes to the ***PostOffice*** data byte. It is cleared when a ***PostOffice*** cycle is completed or a ***PostOffice*** time-out occurred. In case of concurrent accesses to this bit, the result is '0'. '1' - ***PostOffice*** request is pending. '0' - ***PostOffice*** request is not pending (default value).
24	RC	all	**POTime** - ***PostOffice*** Time-out flag. This bit is set to '1' by the ***GuestBus*** master if a ***PostOffice*** cycle on the ***GuestBus*** lasts more than 32 PCI clocks. This might happen when the accessed guest holds **GWS#** low for too long. It is cleared ('0') by the host writing a '1'. In case of concurrent accesses to this bit, it remains '1'. '1' - ***PostOffice*** cycle has timed out. '0' - no time-out occurred (default value).
23	RW	all	**PODir** - ***PostOffice*** Direction flag. This bit defines the direction of the ***PostOffice*** operation: '0' - Read (host reads guest). '1' - Write (host writes to guest). Default after reset is '1'.
22	R		Reserved. Returns 0.
21:20	RW	all	**POGuestID** - ***PostOffice*** Guest Indication. These two bits select the guest to be accessed, i.e., they determine which of the four GCSn# will be asserted in the requested ***PostOffice*** cycle. Default after reset is 00b.
19	R		Reserved. Returns 0.
18:16	RW	all	**POGuestReg** - ***PostOffice*** Guest Register. Register indication of accessed guest. Within each guest up to eight registers can be addressed. The **POGuestReg** bits determine the register address that will be presented on the GADR2:0 pins in the requested ***PostOffice*** cycle. Default after reset is 000b.
15:8	R		Reserved. Returns zero.
7:0	RW	all	**POData** - ***PostOffice*** Data. An eight bit register containing the data being transferred during ***PostOffice*** reads and writes. Default after reset is 00000000b.

11. ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... -65 C to +150 C
Supply Voltage to Ground (V_{DD}) .....-0.5V to +6.0V
DC Output Voltage ..... -0.5V to V_{DD}+0.5V
DC Input Voltage..... -0.5V to V_{DD}+0.5V
DC Input Current, any single input.....-100 mA to +100 mA
DC Output Current, any single output..... -100 mA to +100 mA
Supply Current (I_{DD}) ..... 0 mA to 320 mA
Total Power Dissipation (P_{TOT}..... 1.6W (max)

NOTE: Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above those limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

12.1 OPERATING RANGE

Temprature.....0^oC to $+70^oC$

Supply Voltage.....4.75V to 5.25V

12.2 DC CHARACTERISTICS

Symbol	Parameter	Min	Typ	Max	Units	Test Conditions
V_{IL}	Input Low Voltage			0.8	V	
V_{IH}	Input High Voltage	2.0			V	
V_{IIL}	Input Low Voltage (SDA,SCL)			0.3 VCC	V	
V_{IIH}	Input High Voltage (SDA,SCL)	0.7 VCC			V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL} =6 mA, all outputs except VCLK and VCLKX2
						I_{OL} = 12 mA, VCLK and VCLKX2
V_{OH}	Output High Voltage	3.8			V	I_{OH} = -6 mA, all outputs except VCLK and VCLKX2
						I_{OH} = -12 mA, VCLK and VCLKX2
I_{LI}	Input Leakage Current			±10	µA	V_{IN} = V_{CC} or GND
I_{LO}	Output Leakage Current			±10	µA	V_{OUT} = V_{CC} or GND, output disabled
C_{IN}	Input Capacitance			8	pF	
C_{IO}	Output and I/O capacitance			8	pF	
PCI Bus						
V_{PIL}	Input Low Voltage			0.8	V	
V_{PIH}	Input High Voltage	2.0			V	
V_{OL}	Output Low Voltage			0.55	V	I_{OL} = 6 mA
V_{OH}	Output High Voltage	2.4			V	I_{OH} = -2 mA
I_{LI}	Input Leakage Current			±70	µA	V_{INH} = 2.7 or 0.5 V
C_{PCICLK}	PCICLK Input Capacitance	5		12	pF	

12.3 AC CHARACTERISTICS

12.3.1 PCI Bus Timing

Symbol	Parameter	Min	Max	Unit	Notes
TPCI	PCICLK Period	30		ns	
TPH	PCICLK High	12		ns	
TPL	PCICLK Low	12		ns	
TA3	PCICLK Slew Rate	1	4	V/ns	
TA4	PCICLK to signal valid delay - bussed signals	2	11	ns	
TA5	PCICLK to signal valid delay - point-to-point	2	12	ns	
TA6	Float to active delay	2	11	ns	
TA7	Active to float delay		28	ns	
TA8	Input setup time to PCICLK - bussed signals	7		ns	
TA9	Input setup time to PCICLK - point-to-point	10		ns	
TA10	Input hold time from PCICLK	0		ns	
TA11	Unloaded output rise/fall time	1	5	V/ns	between 0.4V and 2.4V

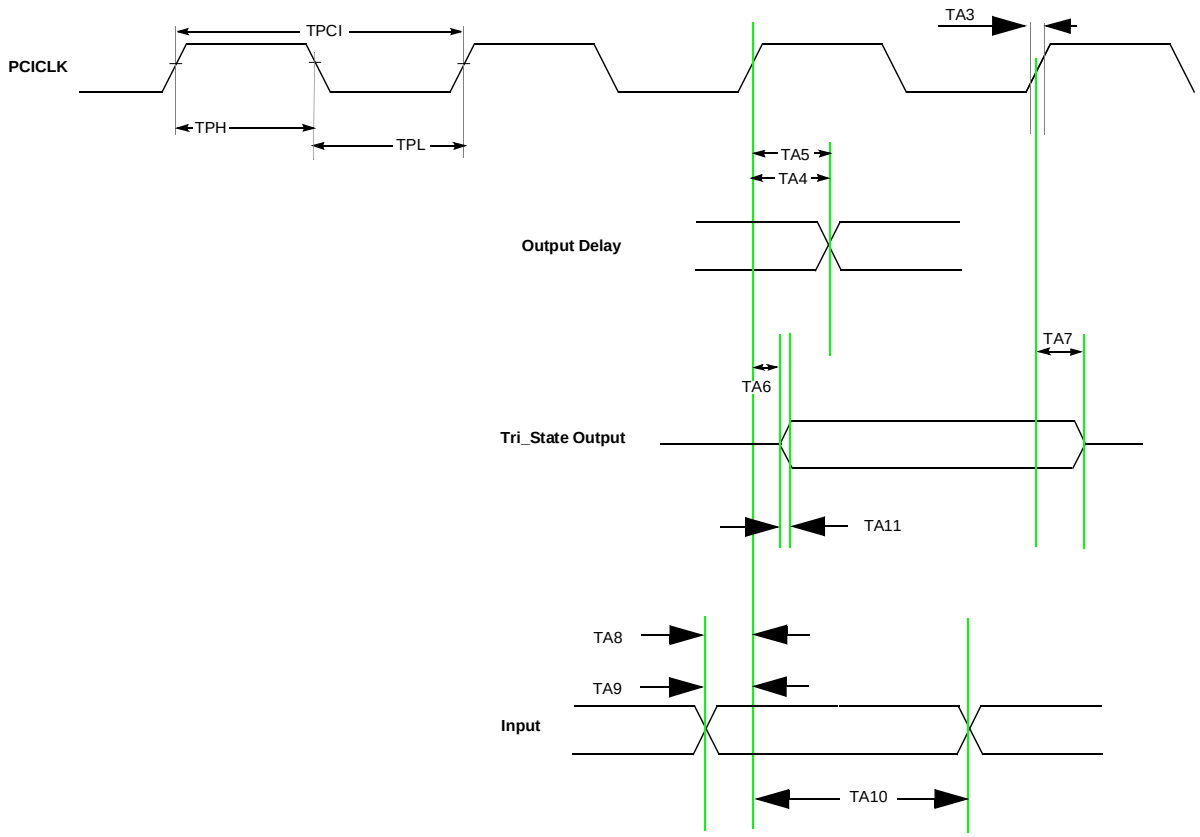


Figure 4: PCI Bus Interface Timing

12.3.2 Digital Video Bus Timing

Symbol	Parameter	Min	Max	Unit	Notes
TA13	VCLKx2 Period	31	45	ns	40% to 60% duty cycle
TA14	VCLKx2 Rise/Fall Transition		5	ns	
TA15	Input Setup	15		ns	
TA16	Input Hold	0		ns	
TA17	VLCK Hold	-2		ns	

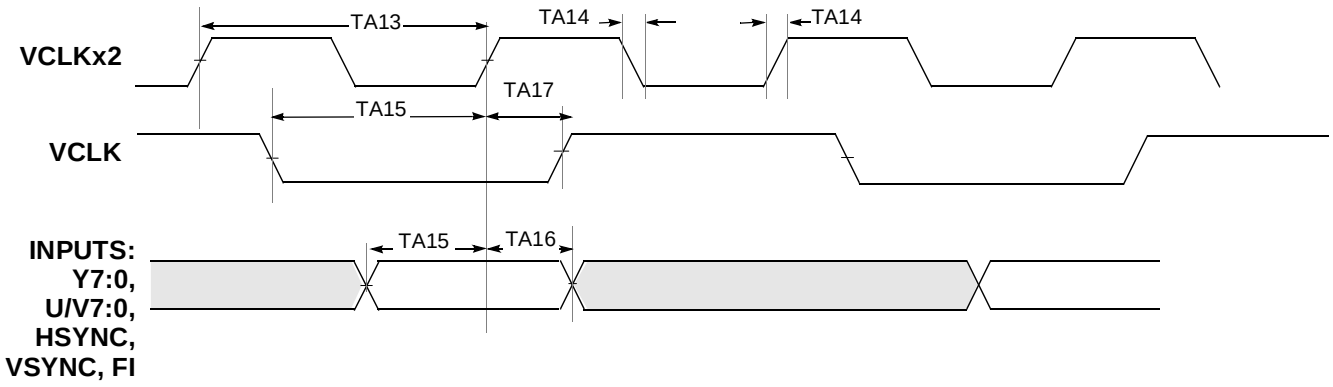


Figure 5: Digital Video Bus Timing

12.3.3 *GuestBus* Timing

Symbol	Parameter	Min	Max	Unit	Test Conditions
TA18	GCSn# / GADR2...0 Setup	16		ns	
TA19	GCSn# / GADR2...0 Hold	16		ns	
TA20	GWR# / GRD# High	90 [1]		ns	
TA21	GWR# / GRD# Low	90 [2]	32 * TPCI [3]	ns	
TA22	Propagation Delay	1	15	ns	CL = 50 pF
TA23	Data Hold (write)	TPCI - 14	TPCI + 14	ns	
TA24	Data Setup (read)	11		ns	
TA25	Data Hold (read)	0		ns	
TA26	GWS# Setup [4]	(T_{DUR} - 1) * TPCI - TA22 - TA24		ns	
TA27	GWS# Low	TPCI [5]		ns	
TA28	GWS# High Delay	TPCI + TA22 + TA24		ns	
TA29	GIRQ1:0 High	2 * TPCI		ns	

1. according to parameter T_{REC}
2. according to parameter T_{DUR}
3. maximum duration resulting from GWS#
4. GWS# is checked internally one PCI clock before the expiration of the T_{DUR} parameter of the accessed guest.
5. For each period of TPCI the duration of a transfer will be extended for one PCI clock period.

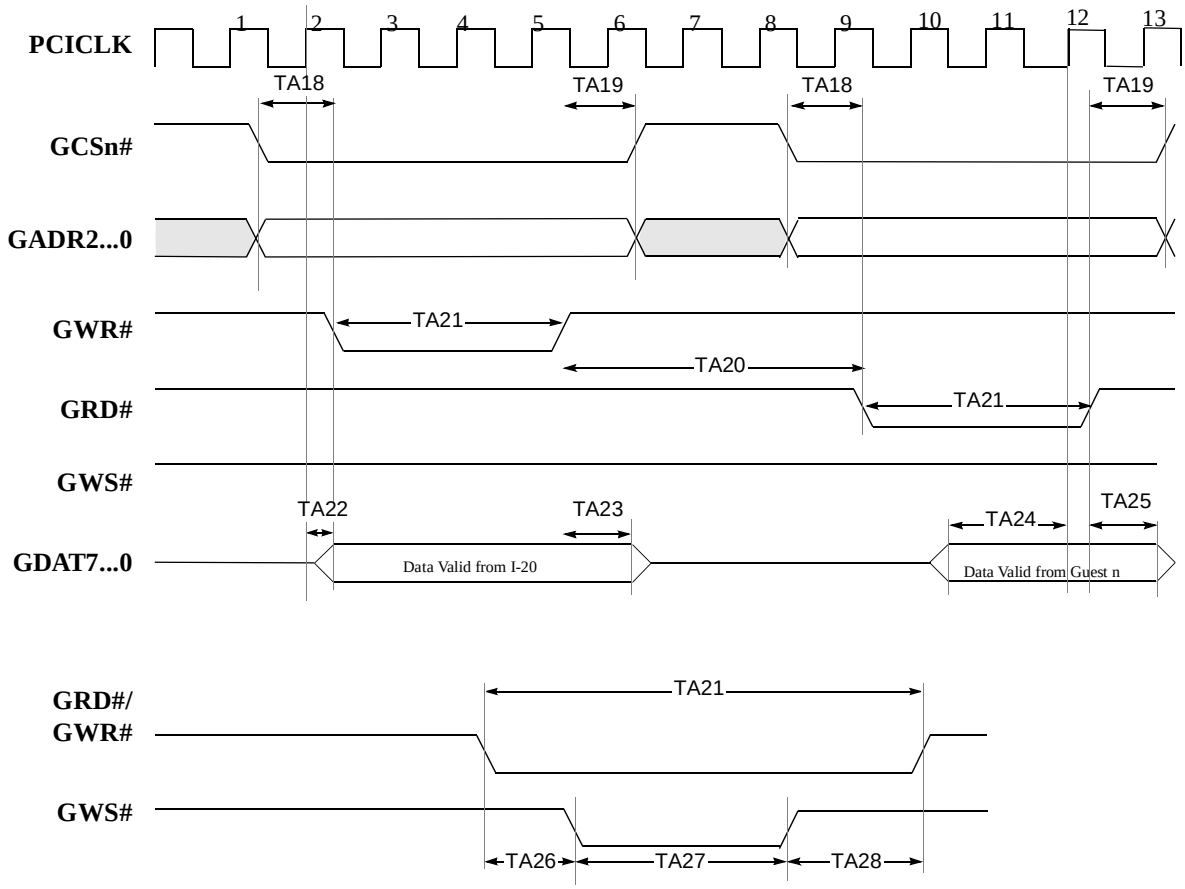


Figure 6: GuestBus Timing

13. Mechanical Data

13.1 Pinout

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
1	VSS	21	REQ#	41	VSS	61	DEVSEL#	81	VSS	101	GPIO2	121	VDD	141	VCLKx2
2	Y0	22	VDD	42	AD23	62	STOP#	82	C/BE0#	102	GPIO3	122	VSS	142	VSS
3	VDD	23	VSS	43	AD22	63	VSS	83	AD7	103	GPIO4	123	GCS1#	143	VCLK
4	Y1	24	AD31	44	VSS	64	VDD	84	VSS	104	GPIO5	124	GCS0#	144	VDD
5	Y2	25	AD30	45	AD21	65	PAR	85	AD6	105	VSS	125	GADR2	145	FI
6	Y3	26	VSS	46	AD20	66	C/BE1#	86	VDD	106	GPIO6	126	GADR1	146	VSYNC
7	Y4	27	AD29	47	VSS	67	VSS	87	AD5	107	GPIO7	127	GADR0	147	HSYNC
8	VSS	28	VDD	48	AD19	68	AD15	88	VSS	108	SCL	128	VSS	148	VSS
9	Y5	29	AD28	49	AD18	69	AD14	89	AD4	109	SDA	129	GDAT0	149	UV0
10	Y6	30	VSS	50	VSS	70	VSS	90	AD3	110	VDD	130	GDAT1	150	VDD
11	Y7	31	AD27	51	AD17	71	AD13	91	VSS	111	GIRQ0	131	VDD	151	UV1
12	VDD	32	AD26	52	VDD	72	AD12	92	AD2	112	GIRQ1	132	GDAT2	152	UV2
13	GPIO0	33	VSS	53	AD16	73	VSS	93	AD1	113	GWS#	133	GDAT3	153	UV3
14	INTA#	34	AD25	54	VSS	74	AD11	94	VSS	114	GRDY	134	VSS	154	VSS
15	VSS	35	AD24	55	C/BE2#	75	VDD	95	AD0	115	VSS	135	GDAT4	155	UV4
16	PCIRST#	36	VSS	56	FRAME#	76	AD10	96	VDD	116	GWR#	136	GDAT5	156	UV5
17	VDD	37	C/BE3#	57	VSS	77	VSS	97	TEST	117	GRD#	137	GDAT6	157	UV6
18	PCICLK	38	IDSEL	58	IRDY#	78	AD9	98	ENID	118	GCS3#	138	VSS	158	UV7
19	VSS	39	VDD	59	TRDY#	79	AD8	99	VSS	119	GCS2#	139	GDAT7	159	VDD
20	GNT#	40	VSS	60	VSS	80	VSS	100	GPIO1	120	VSS	140	VDD	160	VSS

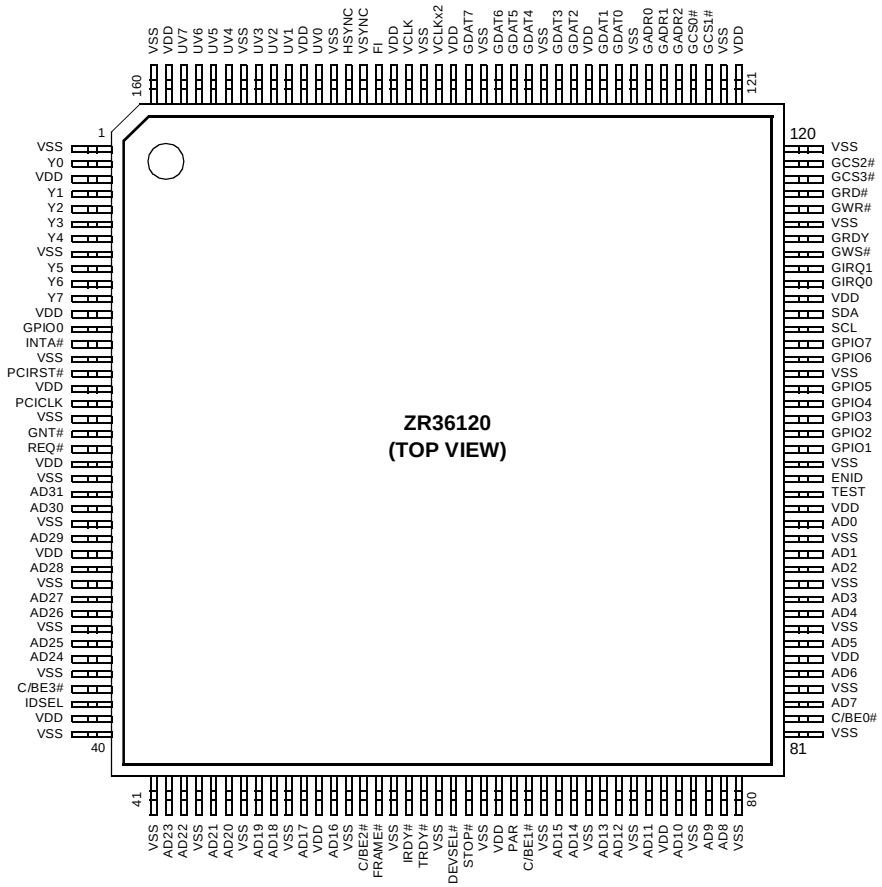
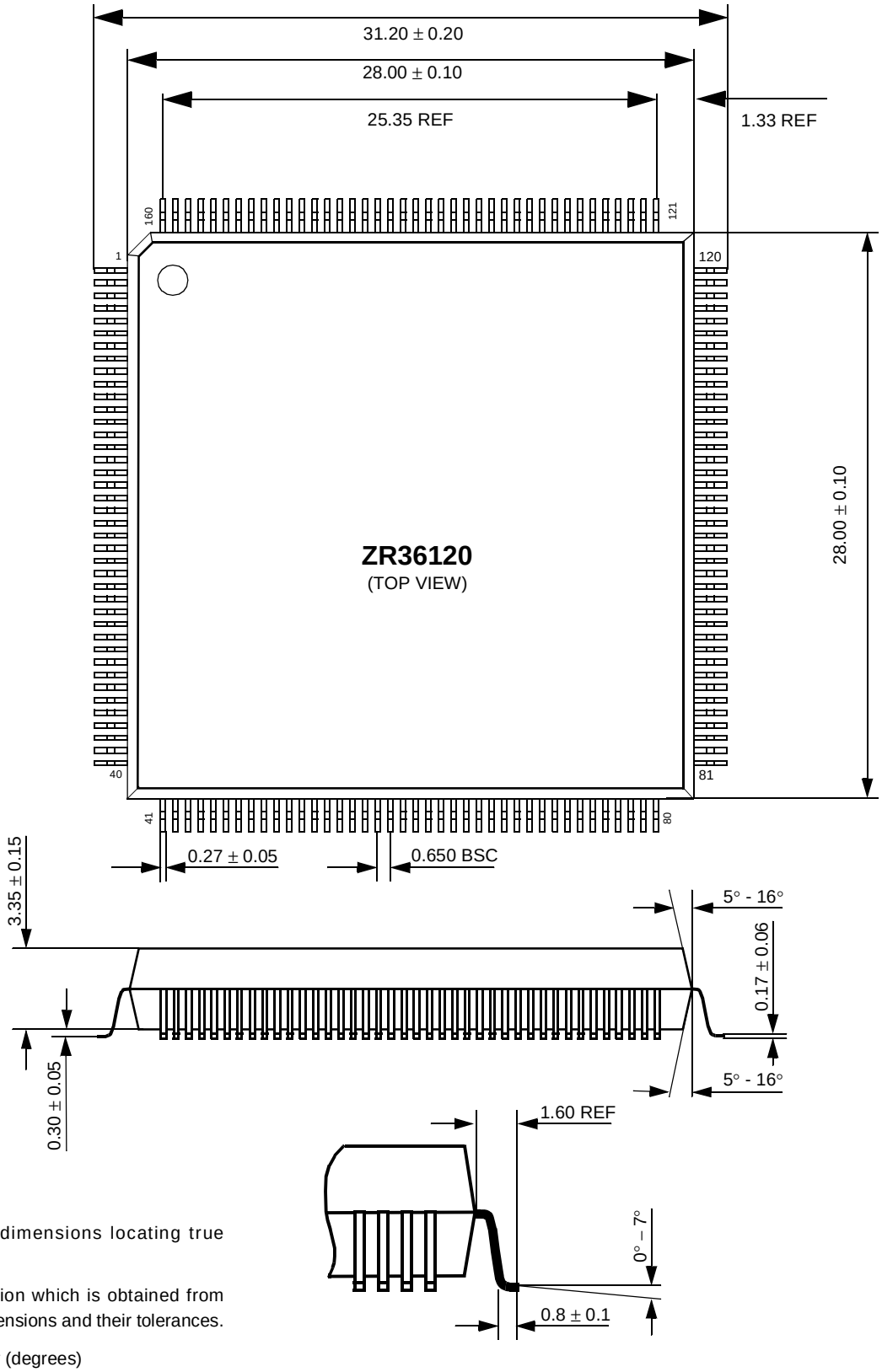


Figure 7: 160-Pin Quad Flat Pack Pinout

13.2 Dimensions



Legend

BSC BASIC, dimensions locating true position

REF A dimension which is obtained from other dimensions and their tolerances.

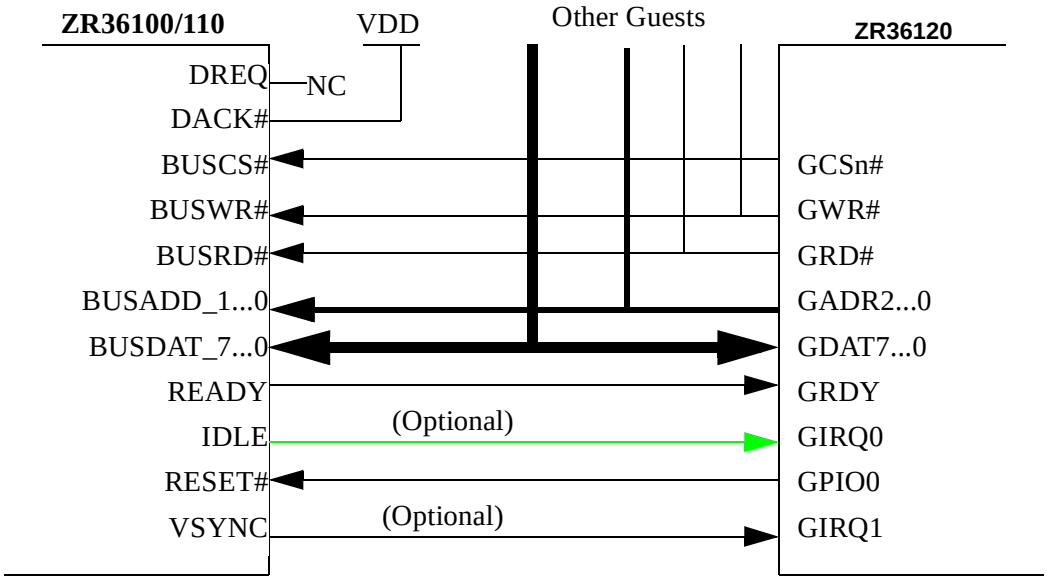
Angularity (degrees)

Linear dimensions in millimeters

14. Appendix A: Interfacing the ZR36120 to the ZR36100/ZR36110 MPEG1 Decoder

Figure 8 describes a recommended interconnection between the **ZR36120** and the **ZR36100**/**ZR36110** host bus.

Figure 8: ZR36120 - ZR36100/ZR36110 Host (Guest) Bus Interconnection



- **ZR36100/ZR36110 Reset**

  Any of the software controlled GPI/O pins (configured as output) of the **ZR36120** may be used as a **RESET** input of the decoder. The software then directly manipulates the **RESET** signal through the corresponding register bit. Since the default configuration of the GPIO pins after reset is input, a pull down resistor should be applied to the ZR36100/ZR36110 **RESET** input.

- **Mapping the ZR36100/ZR36110 on the ZR36120's *GuestBus***

  The driver software must map the ZR36100/ZR36110 on the ***GuestBus***: the ZR36100/ZR36110's guest ID number (0,1,2, or 3) must be configured as a code-write target. The proper timing parameters (t_{dur} = 3, to ensure 82ns, t_{rec} = 4, to ensure 100ns) of the ZR36100/ZR36110 must be loaded to the ***GuestBus*** control register (address 0x030). The ZR36100/ZR36110 occupies only four 8-bit registers out of the eight registers dedicated to each guest.

- **ZR36100/ZR36110 Initialization**

  The initialization consists of loading the **ZR36100/ZR36110** microcodes and parameters. This is done using the ***PostOffice*** mechanism. The host interface of the **ZR36100/ZR36110** must be set to 8 bit, Intel format, I/O only. The BSLN parameter should be set to 2 or 4, for efficient operation.

- **On-Line Commands and Status**

  On-line command writes and status reads are also done using the ***PostOffice*** mechanism.

- **Bitstream Transfer**

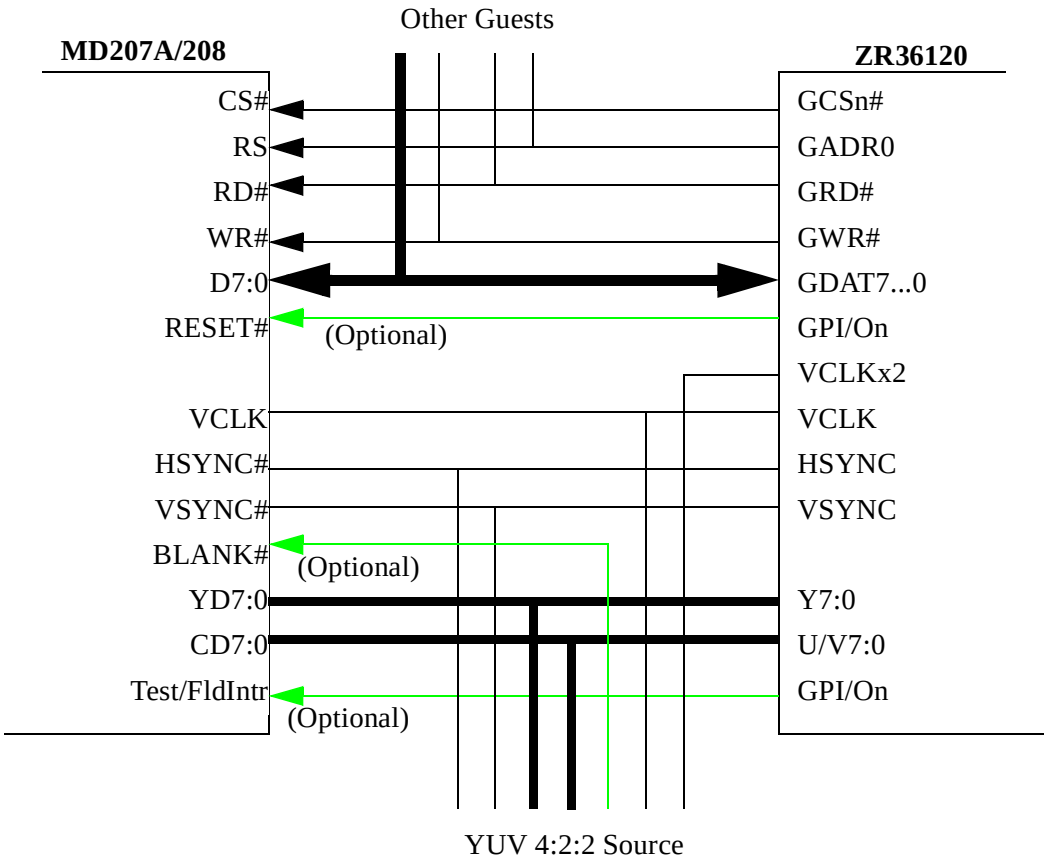
  Some preparations must be done prior to triggering the ZR36100/ZR36110 with a ***go*** command.
  The host must allocate a **contiguous** code buffer in the system memory. Bitstream retrieved from the MPEG source is stored in this buffer. The **ZR36120** reads data from this buffer in a DMA fashion and transfers it, through the internal code buffer, to the MPEG decoder. There are several possible sizes of the memory buffer. The host must inform the **ZR36120** of the buffer address, size and "report step." After the code buffer in memory is allocated, reported to the **ZR36120**, **and filled up for the first time**, a ZR36100/ZR36110 ***go***

on-line command can be issued. Immediately after this, the DMA code-read cycles must be enabled by setting the DMA Code-Read Enable bit to '1'. The **ZR36120** then starts fetching data from the main memory buffer using cyclic addressing. Whenever it passes a "report step" it initializes an interrupt request. Within the interrupt service routine the host CPU should check the current position of the **ZR36120** Code Memory Buffer Pointer, and decide if it should refresh an old portion of the buffer with new data from the MPEG source. Once the coded data arrives at the internal code buffer, the ***GuestBus*** master logic starts writing it over the ***GuestBus*** to the **ZR36100**.

15. Appendix B: Interfacing the ZR36120 to the MD207A/208 Video Encoder

This appendix suggests the basic interconnection between the **ZR36120** and the **MD207A/208** video encoder. Naturally, when these two devices are connected together there must be a third device, mastering the YUV bus. Figure 8 describes a basic interconnection between the **ZR36120** and the **MD207A/208**, with an arbitrary YUV source. This minimum example does not use the graphics overlay capability of the **MD207A/208**

Figure 9: ZR36120 - MD207A/208 Basic Interconnection



- **MD207A/208 Reset**

  Software controlled usage of the **RESET#** input of the **MD207A/208** is optional. Generally, it is more efficient to connect it to the power-up reset of the circuit, and control the device through the software reset register bit of the **MD207A/208**. If a hardware reset is needed in the design, any of the software controlled GPI/O pins (configured as output) of the **ZR36120** may be used as a **RESET#** signal. The software then directly manipulates the **RESET#** signal through the corresponding register bit of the **ZR36120**. Since the default configuration of the GPI/O pins after reset is input, a pull down resistor should be applied to the **MD207A/208 RESET#** input.

- **Mapping the MD207A/208 on the ZR36120's *GuestBus***

  The driver software must map the **MD207A/208** on the ***GuestBus***. The proper timing parameters (t_{dur} = 12, t_{rec} = 15) of the **MD207A/208** must be loaded to the ***GuestBus*** control register.

  Reading/writing one byte from/to the **MD207A/208** requires two ***GuestBus*** cycles: in the first cycle the address (index) of the internal **MD207A/208** register is written, in the second one the data byte is read/written. The **RS** (register select) input of the **MD207A/208** is used to distinguish between the two types of cycles. Connecting this pin to the **ZR36120**'s **GADR0** virtually creates two **MD207A/208** registers at the level of the ***GuestBus***: when **GADR0** is low (even registers), the **MD207A/208** expects address to be transferred

on its **D7:0** bus, when **GADR0** is high (odd registers) data is output or input on these lines. Another way is to connect **GADR2** to **RS**.

- **Sync Polarity**

  Since, unlike the **ZR36120**, the sync polarity of the **MD207A/208** is not programmable, then, depending on the YUV 4:2:2 source, it might be necessary to invert the **HSYNC** and **VSYNC** of the **MD207A/208**.

- **Vertical Interpolation with the MD208**

  Pin 8 is the only one that is different between the **MD207A** and the **MD208**. While in the **MD207A** it is a test pin, normally connected to ground, the **MD208** uses this input to switch its internal vertical interpolation mechanism on and off. When this mechanism is on, one field out of every pair is vertically interpolated and the interpolated lines are the ones sent out. Since this operation is not always desired (e.g., in high resolution still pictures of VideoCD 2.0) it must be controlled by the software. The natural way to obtain this control is using one of the GPI/Os. It is better to pull this pin down in order for the same layout to support both the **MD207A** and the **MD208**.

16. Appendix C: Fitting the Input Size to the Required Display Window

The **ZR36120** can crop the input video and scale it **down** to match any display size required by different applications, as long as the required size is not larger than the original input. This appendix provides some programming guidelines for proper setting of the **ZR36120** parameters involved in this process.

For better understanding, a typical example is detailed along with the general explanations.

It is assumed that the driver software "knows" the following basic parameters about the incoming video:

Wt - **total** width of the input field. For example, in CCIR NTSC Wt=858.

Wa - **active** width of the input field. For example, in CCIR NTSC Wa=720.

Ht - total height of the input frame. For example, in CCIR NTSC Ht=525.

Ha - active height of the input frame. For example, in CCIR NTSC Ht=480.

It is assumed that Ha is an even number.

It is also assumed that from knowing the video input format, the driver knows how to set the following **ZR36120** parameters, such that the entire active portion of the video input would have been sampled:
HSPol - the polarity of HSYNC, as defined in the **ZR36120** data sheet.
VSPol - the polarity of VSYNC, as defined in the **ZR36120** data sheet.
HStart' - the number of pixels, from the active edge of HSYNC, after which the **ZR36120** starts to sample the input.
HEnd' - the number of pixels, from the active edge of HSYNC, after which the **ZR36120** stops sampling the input.

The following equation connects HStart', HEnd', and Wa:

Wa = HEnd' - HStart' + 1

In the example, Hend' = 841, HStart' = 122.

VStart' - the number of lines, from the active edge of VSYNC, after which the **ZR36120** starts to sample the input.

VEnd' - the number of lines, from the active edge of VSYNC, after which the **ZR36120** stops sampling the input.

The following equation connects VStart', VEnd', and Ha:

Ha/2 = VEnd' - VStart' + 1

In the example, Vend' = 249, VStart' = 10.

The driver receives from the application software the parameters that define the size of the rectangle, on the monitor, that should be filled with video pixels. This is referred to as the destination rectangle:

VidWinWid - width of the video rectangle.

VidWinHt - If DispMode=1 (single field display) this is the height of the video rectangle. If DispMode=0 (emulation of interlaced video) this is half of the height of the video rectangle, or, in other words, height of one destination field.
Note that if DispMode=0 the vertical size of the video rectangle must be an even number. This limitation might conflict with application that will require a video window of a specific, odd vertical size. Such conflicts will be resolved by the driver software, which will actually build, in these cases, a rectangle shorter by one line than the required, and will fill-in the missing line (a-priori) with some background color.

Obviously, in many cases the size of the video rectangle to be displayed is different (smaller) than the size of the active video input sampled by the **ZR36120**.
In the example: VidWinWid=597, VidWinHt=199. (Assuming DispMod=0, this means that the target "video window" in the example is 597x398).
In such cases, out of the parameters above, the driver software must determine the following **ZR36120** parameters, such that the quality of the displayed video is optimal in the sense that the decimation factors will be as small as possible and the portions of the input that are "cut out" (cropped) will also be as small as possible.

hcrop1 - the number of pixels, that will be dropped from the beginning of the active line. hcrop1 is not really a **ZR36120** parameter, but it is useful to obtain HStart:

HStart = HStart' + hcrop1.

hcrop2 - the number of pixels, that will be dropped from the end of the active line. hcrop2 is not really a **ZR36120** parameter, but it is useful to obtain HEnd:

HEnd = HEnd' - hcrop2.

vcrop1 - the number of lines, that will be dropped from the beginning of the active field. vcrop1 is not really a **ZR36120** parameter, but it is useful to obtain VStart:

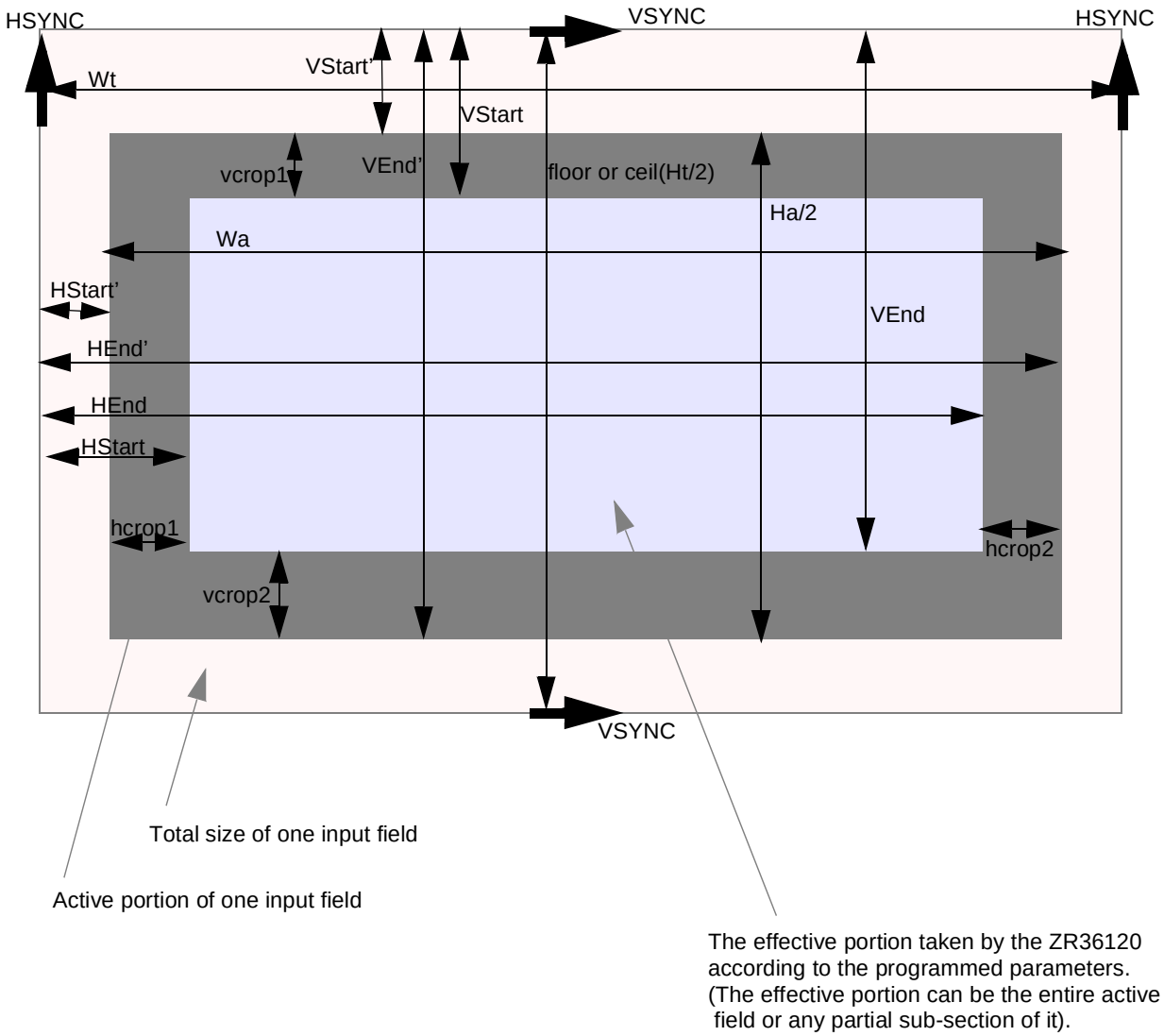
VStart = VStart' + vcrop1.

vcrop2 - the number of lines, that will be dropped from the end of the active field. vcrop2 is not really a **ZR36120** parameter, but it is useful to obtain VEnd:

VEnd = VEnd' - vcrop2.

HorDcm - ratio of horizontal decimation. A number of HorDcm pixels will be dropped out of every consecutive 64 pixels in an input line. HorDcm/64 is the horizontal decimation factor.
VerDcm - ratio of vertical decimation. A number of VerDcm pixels will be dropped out of every consecutive 64 lines in an input field. VerDcm/64 is the vertical decimation factor.
HFilter - the horizontal filter through which the input is passed. The filter is selected according to the horizontal decimation factor that is first determined.
The following figure illustrates the above parameters:

Figure 10: Input Image Parameters



Calculating the horizontal parameters:

X and We denote two temporary variables:

X = ceil (VidWinWid*64/Wa)

We = floor (VidWinWid*64/X)

HorDcm = 64 - X

hcrop1 = 2*floor((Wa-We)/4)

hcrop2 = Wa - We - hcrop1

HStart and Hend are then calculated from hcrop1 and hcrop2.

In the example:

X = ceil(597*64/720) = 54

We = floor(597*64/54) = 707

HorDcm= 64 - 54 = 10

hcrop1 = 2*floor((720-707)/4)=6

hcrop2 = 720-707 - 6 = 7

HStart = HStart' + hcrop1 = 122 + 6 = 128

HEnd = HEnd' - hcrop2 = 841 - 7 = 834

(834 - 128 + 1 = 707)

The **ZR36120** will actually sample-in 707 pixels from every line.
The first 6 and the last 7 active pixels will be cut out.

Calculating the vertical parameters

(Regardless of DispMod!)

Y and He denote two temporary variables:

Y = ceil (VidWinHt*64*2/Ha)

He = floor (VidWinHt*64/Y)

VerDcm = 64 - Y

vcrop1 = floor((Ha/2 - He)/2)

vcrop2 = Ha/2 - He - vcrop1

VStart and VEnd are then calculated from vcrop1 and vcrop2.

In the example:

Y = ceil(199*64*2/480) = 54

He = floor(199*64/54) = 235

VerDcm= 64 - 54 = 10

vcrop1 = floor(((480/2) - 235) / 2) = 2

vcrop2 = (480/2) - 235 - 2 = 3

VStart = VStart' + vcrop1 = 10 + 2 = 12

VEnd = VEnd' - vcrop2 = 249 - 3 = 246

(246 - 12 + 1 = 235)

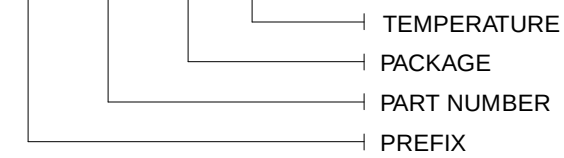
The **ZR36120** will actually sample-in 235 lines from every field.

The first 2 and the last 3 active lines of every field will be cut out.

Notes:

ORDERING INFORMATION

ZR 36120 PQ C



PACKAGE

PQ - Plastic Quad Flat Pack (EIAJ)

Temperature Range

C - 0°C to +70°C (V_{CC} = 4.75V to 5.25V)

SALES OFFICES

■ ***U.S. Headquarters***
Zoran Corporation
2041 Mission College Blvd.
Santa Clara, CA 95054 USA
Telephone: 408-986-1314
FAX: 408-986-1240

■ ***Israel Operations***
Zoran Microelectronics, Ltd.
Advanced Technology Center
P.O. Box 2495
Haifa, 31024 Israel
Telephone: 972-4-854-5777
FAX: 972-4-855-1550

The material in this data sheet is for information only. Zoran Corporation assumes no responsibility for errors or omissions and reserves the right to change, without notice, product specifications, operating characteristics, packaging, etc.

Zoran Corporation assumes no liability for damage resulting from the use of information contained in this document.
Copyright © Zoran Corporation