# VGA32 – 32-bit VGA Controller


## Summary

This document provides detailed information with respect to the VGA32 peripheral device.



Core Reference
CR0169 (v2.0) March 11, 2008


The VGA32 provides a simple, 32-bit interface between a host processor and any VGA-compatible monitor. Taking a processor-generated picture (pixilated) from memory space, the Controller provides digital RGB values for each pixel, as well as horizontal and vertical synchronization signals, in order to correctly display the picture on a connected monitor.

This VGA Controller is one of three designed for use with 32-bit processors, which can be summarized as follows:

| | | |
|---|---|---|
| **VGA32** | - | standard 32-bit VGA Controller with configurable color quality (1,2,4,8bpp), support for screen resolution up to 800x600 and screen refresh rate of up to 75Hz. |
| **VGA32_16BPP** | - | as per the standard 32-bit VGA Controller, but with fixed 16bpp color quality. |
| **VGA32_TFT** | - | this 32-bit VGA Controller is specifically used to interface to a TFT panel. It supports a fixed TFT screen resolution of 240x320 and a fixed refresh rate of 50Hz. Its color quality is fixed at 16bpp. |

For more information on the VGA32_16BPP, refer to the core reference *VGA32_16BPP – 32-bit VGA Controller with 16bpp Data Support*.

For more information on the VGA32_TFT, refer to the core reference *VGA32_TFT – 32bit VGA Controller with TFT Interface*.

There is also a configurable 32-bit VGA Controller – the WB_VGA – which allows you to configure which of these three 32-bit Controllers is used after placement on the schematic sheet. For more information, refer to the core reference *WB_VGA Configurable Wishbone Display Driver*.

An 8-bit non-Wishbone VGA Controller is also available. For information, refer to the *VGA – 8-bit VGA Controller* core reference.

## Features

- Compatible with any standard VGA- or SVGA-compatible monitor
- Independent Wishbone Master and Pixel clock inputs
- Supports monitor screen resolutions up to 800x600
- Supports monitor refresh rates up to 75Hz
- Black & White, 8 bits per pixel mode (256 grey scale mode)
- Four non-indexed color modes:
  - 8 bits per pixel mode
  - 4 bits per pixel mode
  - 2 bits per pixel mode
  - 1 bit per pixel mode
- Indexed color mode
  - 2 color look-up tables
- Configurable output synchronization levels
- Composite output
- 3 interrupt outputs (HSync, VSync, Blank).

# Available Devices

The VGA32 Controller device can be found in the FPGA Peripherals integrated library (`FPGA Peripherals.IntLib`), located in the `\Library\Fpga` folder of the installation.

# Functional Description

## Symbol

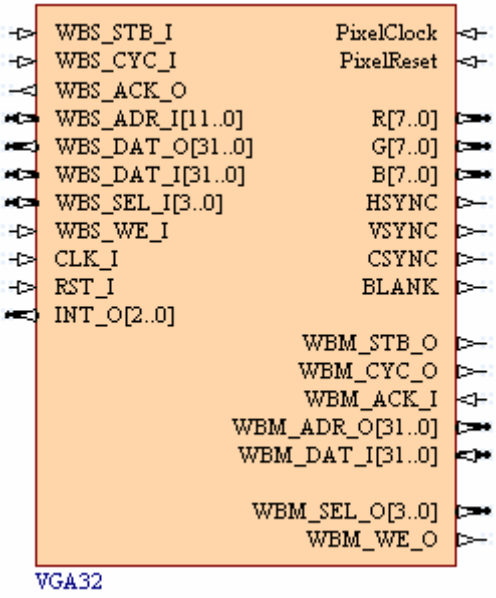


*Figure 1. VGA Controller Symbol – 32-bit Wishbone variant (VGA32)*

## Pin Description

*Table 1. VGA32 Pin description*

| Name | Type | Polarity/ Bus size | Description |
|---|---|---|---|
| **Global Control Signals** | | | |
| CLK_I | I | Rise | Global Wishbone clock input. This clock is used to drive both Master and Slave interfaces |
| RST_I | I | High | Global Wishbone reset |
| PixelClock | I | Rise | Video subsystem clock |
| PixelReset | I | High | Video subsystem reset |
| **Host Processor Interface Signals** | | | |
| WBS_STB_I | I | High | Strobe signal. When asserted, indicates the start of a valid Wishbone data transfer cycle |
| WBS_CYC_I | I | High | Cycle signal. When asserted, indicates the start of a valid Wishbone cycle |
| WBS_ACK_O | O | High | Standard Wishbone device acknowledgement signal. When this signal goes high, the Controller (Wishbone Slave) has finished execution of the requested action and the current bus cycle is terminated |
| WBS_ADR_I | I | 12 | Address bus, used to select an internal register of the device for writing to/reading from |
| WBS_DAT_O | O | 32 | Data to be sent to host processor |

| | | | |
|---|---|---|---|
| WBS_DAT_I | I | 32 | Data received from host processor |
| WBS_SEL_I | I | 4/High | Select input, used to determine where data is placed on the WBS_DAT_O line during a Read cycle and from where on the WBS_DAT_I line data is accessed during a Write cycle. Each of the data ports is 32-bits wide with 8-bit granularity, meaning data transfers can be 8-, 16- or 32-bit. The four select bits allow targeting of each of the four active bytes of a port, with bit 0 corresponding to the low byte (7..0) and bit 3 corresponding to the high byte (31..24) |
| WBS_WE_I | I | Level | Write enable signal. Used to indicate whether the current local bus cycle is a Read or Write cycle:<br>0 = Read<br>1 = Write |
| INT_O | O | 3/High | Interrupt output lines. Three interrupts are sent to the connected processor on this 3-bit bus.<br>bit 0 = VSYNC<br>bit 1 = HSYNC<br>bit 2 = BLANK |
| **Video Memory Interface Signals** | | | |
| WBM_STB_O | O | High | Strobe signal. When asserted, indicates the start of a valid Wishbone data transfer cycle |
| WBM_CYC_O | O | High | Cycle signal. When asserted, indicates the start of a valid Wishbone bus cycle. This signal remains asserted until the end of the bus cycle, where such a cycle can include multiple data transfers |
| WBM_ACK_I | I | High | Standard Wishbone device acknowledgement signal. When this signal goes High, the connected Wishbone slave device has finished execution of the requested action and the current bus cycle is terminated |
| WBM_ADR_O | O | 32 | Standard Wishbone address bus. Used to select an address in the connected Wishbone slave device for writing to/reading from |
| WBM_DAT_I | I | 32 | Data received from the connected Wishbone slave device |
| WBM_SEL_O | O | 4/High | Select output, used to determine where data is placed on the WBM_DAT_O line during a Write cycle and from where on the WBM_DAT_I line data is accessed during a Read cycle. Each of the data ports is 32-bits wide with 8-bit granularity, meaning data transfers can be 8-, 16- or 32-bit. The four select bits allow targeting of each of the four active bytes of a port, with bit 0 corresponding to the low byte (7..0) and bit 3 corresponding to the high byte (31..24) |
| WBM_WE_O | O | Level | Write enable signal. Used to indicate whether the current local bus cycle is a Read or Write cycle.<br>0 = Read<br>1 = Write<br>Note: This signal is always Low as the Controller does not write to Video memory, it only reads data from memory. |
| **VGA Monitor Control Signals** | | | |
| HSYNC | O | High/Low | Horizontal synchronization signal. This signal is used to control the horizontal deflection circuit in the VGA monitor, so that the start and end of a line of pixels is correctly displayed across the visible display area of the screen. The polarity of this signal is controlled by the hop bit in the Control register (CTRL.8) |

| | | | |
|---|---|---|---|
| VSYNC | O | High/Low | Vertical synchronization signal. This signal is used to control the vertical deflection circuit in the VGA monitor, so that the start and end of a frame (of lines) is correctly displayed between the top and bottom edges of the visible display area of the screen. The polarity of this signal is controlled by the vop bit in the Control register (CTRL.9) |
| CSYNC | O | High/Low | Composite synchronization signal. The polarity of this signal is controlled by the cop bit in the Control register (CTRL.10) |
| BLANK | O | High/Low | Blank synchronization signal. The polarity of this signal is controlled by the bop bit in the Control register (CTRL.11) |
| R | O | 8 | Provides the 8-bit digital signal for the intensity of red used in composing a pixel's displayed color |
| G | O | 8 | Provides the 8-bit digital signal for the intensity of green used in composing a pixel's displayed color |
| B | O | 8 | Provides the 8-bit digital signal for the intensity of blue used in composing a pixel's displayed color. |

# Hardware Description

## Block Diagram

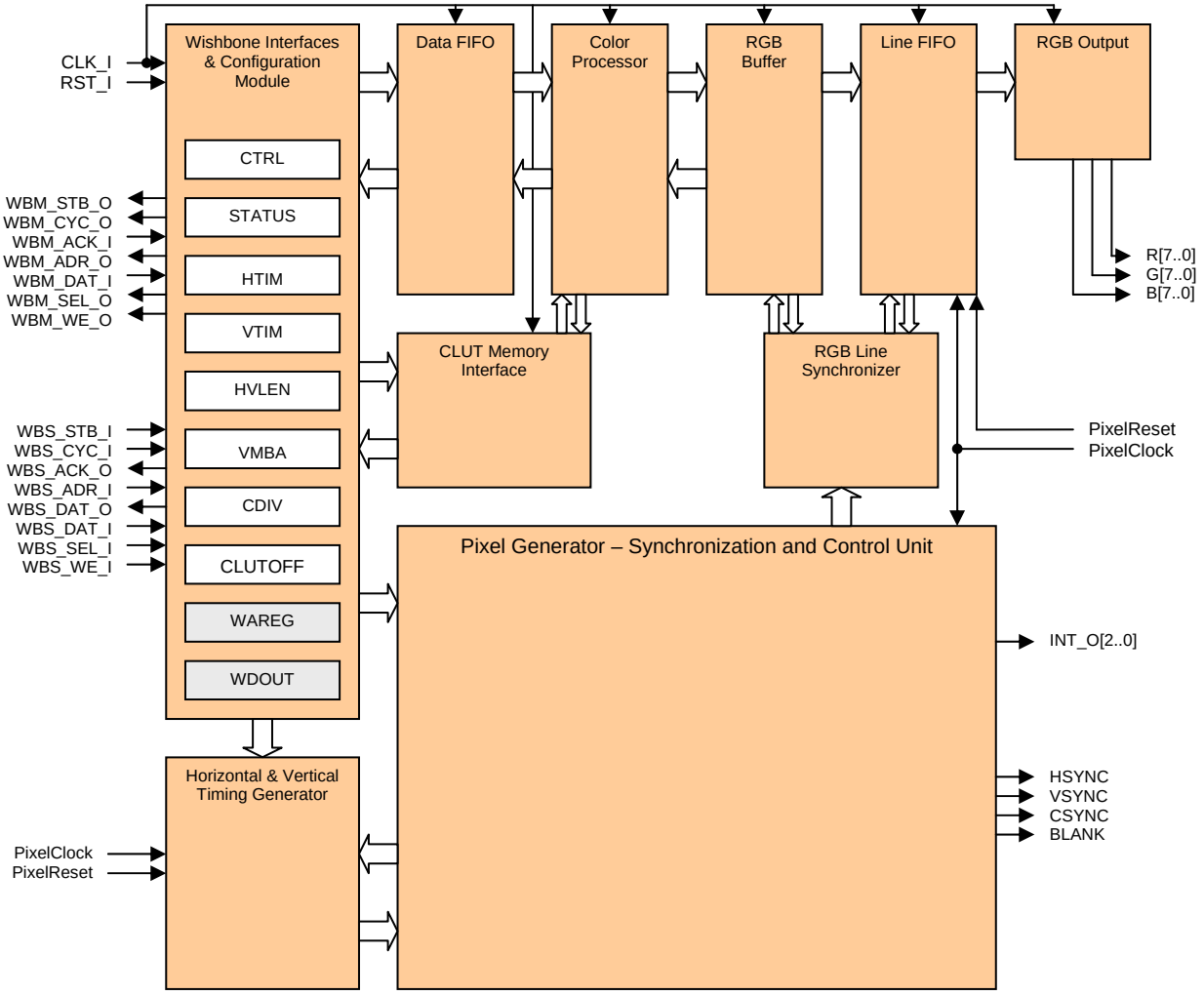


*Figure 2. VGA32 block diagram (simplified)*

## Internal Registers

The following sections detail the internal registers for the VGA32 Controller (part of the Wishbone Interfaces & Configuration Module), as well as the two color look-up tables.

## Control Register (CTRL)

**Address**: 0000000000b

**Access**: Read/Write

**Value after Reset**: 0000_0000h

This 32-bit register is used to configure and control operation of the Controller.

*Table 2. The CTRL register*

MSB LSB

| 31 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| - | ckde | bop | cop | vop | hop | cme | bm1 | bm0 | ltb | - | - | - | vse |

*Table 3. CTRL register bit functions*

| Bit | Symbol | Function |
|---|---|---|
| CTRL.31..CTRL.13 | - | Not used |
| CTRL.12 | ckde | Clock Division Enable bit |
| CTRL.11 | bop | Blanking Synchronization Output Polarity bit.<br>0 = Blanking sync pulse is active High<br>1 = Blanking sync pulse is active Low |
| CTRL.10 | cop | Composite Synchronization Output Polarity bit.<br>0 = Composite sync pulse is active High<br>1 = Composite sync pulse is active Low |
| CTRL.9 | vop | Vertical Synchronization Output Polarity bit.<br>0 = Vertical sync pulse is active High<br>1 = Vertical sync pulse is active Low |
| CTRL.8 | hop | Horizontal Synchronization Output Polarity bit.<br>0 = Horizontal sync pulse is active High<br>1 = Horizontal sync pulse is active Low |
| CTRL.7 | cme | Color Mode Enable bit.<br>0 = Controller operating in Black & White mode<br>1 = Controller operating in Indexed Color mode |
| CTRL.6 | bm1 | Bitmode control bits (bm1 bm0). Determines how many bits are used to represent a single pixel:<br>00 = 8 bits per pixel<br>01 = 4 bits per pixel<br>10 = 2 bits per pixel<br>11 = 1 bit per pixel |
| CTRL.5 | bm0 | |
| CTRL.4 | ltb | Color Look-up Table Select bit.<br>0 = color look-up table 0 is active<br>1 = color look-up table 1 is active |
| CTRL.3 | - | Not used |
| CTRL.2 | - | Not used |
| CTRL.1 | - | Not used |
| CTRL.0 | vse | Video System Enable bit.<br>0 = Controller is stopped<br>1 = Controller is started |

## Status Register (STATUS)

**Address**: 0000000001b

**Access**: Read

**Value after Reset**: 0000_0000h

This 32-bit register is used to provide status information concerning which color look-up table is currently active.

*Table 4. The STATUS register*

MSB LSB

| 31 … 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|
| - | acp | - | - | - | - |

*Table 5. STATUS register bit functions*

| Bit | Symbol | Function |
|---|---|---|
| STATUS.31..STATUS.5 | - | Not used. Returns 0 when read |
| STATUS.4 | acp | Active Color Look-up Table flag. Reflects which look-up table is currently being used:<br>0 = Color Look-up Table 0 in use<br>1 = Color Look-up Table 1 in use |
| STATUS.3 | - | Not used. Returns 0 when read |
| STATUS.2 | - | Not used. Returns 0 when read |
| STATUS.1 | - | Not used. Returns 0 when read |
| STATUS.0 | - | Not used. Returns 0 when read |

## Horizontal Timing Register (HTIM)

**Address**: 0000000010b

**Access**: Read/Write

**Value after Reset**: 0000_0000h

This 32-bit register is used to configure the horizontal timing.

*Table 6. The HTIM register*

MSB LSB

| 31 … 24 | 23 … 16 | 15 … 0 |
|---|---|---|
| hsync | hbporch | hvisible |

*Table 7. HTIM register bit functions*

| Bit | Symbol | Function |
|---|---|---|
| HTIM.31..HTIM.24 | hsync | Horizontal Synchronization Pulse length (in pixels – 1) |
| HTIM.23..HTIM.16 | hbporch | Horizontal Back Porch length (in pixels – 1) |
| HTIM.15..HTIM.0 | hvisible | Horizontal Visible Area (in pixels – 1) |

## Vertical Timing Register (VTIM)

**Address**: 0000000011b

**Access**: Read/Write

**Value after Reset**: 0000_0000h

This 32-bit register is used to configure the vertical timing.

*Table 8. The VTIM register*

MSB LSB

| 31 … 24 | 23 … 16 | 15 … 0 |
|---|---|---|
| vsync | vbporch | vvisible |

Table 9. VTIM register bit functions

| Bit | Symbol | Function |
|---|---|---|
| VTIM.31..VTIM.24 | vsync | Vertical Synchronization Pulse length (in lines – 1) |
| VTIM.23..VTIM.16 | vbporch | Vertical Back Porch length (in lines – 1) |
| VTIM.15..VTIM.0 | vvisible | Vertical Visible Area (in lines – 1) |

## Horizontal and Vertical Length Register (HVLEN)

**Address**: 0000000100b

**Access**: Read/Write

**Value after Reset**: 0000_0000h

This 32-bit register is used to store horizontal and vertical length values that together determine the actual extents of the image display area on the screen.

*Table 10. The HVLEN register*

| MSB | | | LSB |
|---|---|---|---|
| 31 | 16 | 15 | 0 |
| hlen | | vlen | |

*Table 11. HVLEN register bit functions*

| Bit | Symbol | Function |
|---|---|---|
| HVLEN.31..HVLEN.16 | hlen | This value determines the number of viewable pixels to be displayed in each line of a frame and is therefore used to control the horizontal extents of the visible display area |
| HVLEN.15..HVLEN.0 | vlen | This value determines the number of lines to be displayed in a frame and is therefore used to control the vertical extents of the visible display area |

## Video Memory Base Address Register (VMBA)

**Address**: 0000000101b

**Access**: Read/Write

**Value after Reset**: 0000_0000h

This 30-bit register is used to store the address in memory at which the video page starts. The width of the memory used to store the graphics to be displayed is actually 32 bits. When addressing locations in this memory, the 30-bit video base address value is sent on the WBM_ADR_O line as bits 31..2, with bits 1..0 always zeros.

## System Clock Division Register (CDIV)

**Address**: 0000000110b

**Access**: Read/Write

**Value after Reset**: 00h

This 8-bit register is used to store a divisor reload value, allowing you to effectively control the frequency of the clock used to drive the timing generation unit (and horizontal and vertical timing sub-units therein).

If you wish to divide CLK_I by 2, load 01h into CDIV. If you wish to divide by 3, load 02h into CDIV, and so on.

If no internal clock division is required, either load 00h into CDIV, or expressly prohibit clock division by clearing the ckde bit in the Control register (CTRL.12).

## Color Look-up Table Offset Register (CLUTOFF)

**Address**: 0000000111b

**Access**: Read/Write

**Value after Reset**: 00h

This 8-bit register is used to store an offset value used to determine the 8-bit address of a color contained within the active color look-up table. The use of this offset value depends on the bitmode setting in the Control register, as determined by the bm1 and bm0 bits (CTRL.6 and CTRL.5 respectively). The following table summarizes how the effective CLUT address is determined for each of the four possible pixel modes:

*Table 12. Use of CLUTOFF register for CLUT addressing*

| bm1 (CTRL.6) | bm0 (CTRL.5) | Mode | Use of value in CLUTOFF register |
|---|---|---|---|
| 0 | 0 | 8 bits per pixel | not used |
| 0 | 1 | 4 bits per pixel | bits 7..4 are used as the upper bits of the 8-bit CLUT address, with bits 3..0 taken from the Video Memory |
| 1 | 0 | 2 bits per pixel | bits 7..2 are used as the upper bits of the 8-bit CLUT address, with bits 1..0 taken from the Video Memory |
| 1 | 1 | 1 bit per pixel | bits 7..1 are used as the upper bits of the 8-bit CLUT address, with bit 0 taken from the Video Memory |

## Color Look-up Tables (CLUT0 and CLUT1)

Two color look-up tables are available, each of which contain 256 addresses. The two tables form a single contiguous 512 x 32-bit address space:

- CLUT0 – with address range 100h to 17Fh
- CLUT1 – with address range 180h to 1FFh

The color look-up tables can be accessed directly through the Controller's Wishbone Slave interface. Direct access is controlled using bit 11 of the WBS_ADR_I line. When this bit is High, Wishbone communications are configured to be direct with the active color look-up table.

Each 32-bit data entry in the table is comprised as follows:

- bits 31..24 – unused
- bits 23..16 – Red
- bits 15..8 – Green
- bits 7..0 – Blue

## Interfacing to a 32-bit Processor

Figure 3 shows an example of how a VGA32 device can be wired into a design that uses a 32-bit processor – in this case a TSK3000A. A configurable Wishbone Interconnect device (WB_INTERCON) is used to simplify connection and also handle addressing – taking the 24-bit address line from the processor and mapping it to the 12-bit address line used to drive the VGA32.

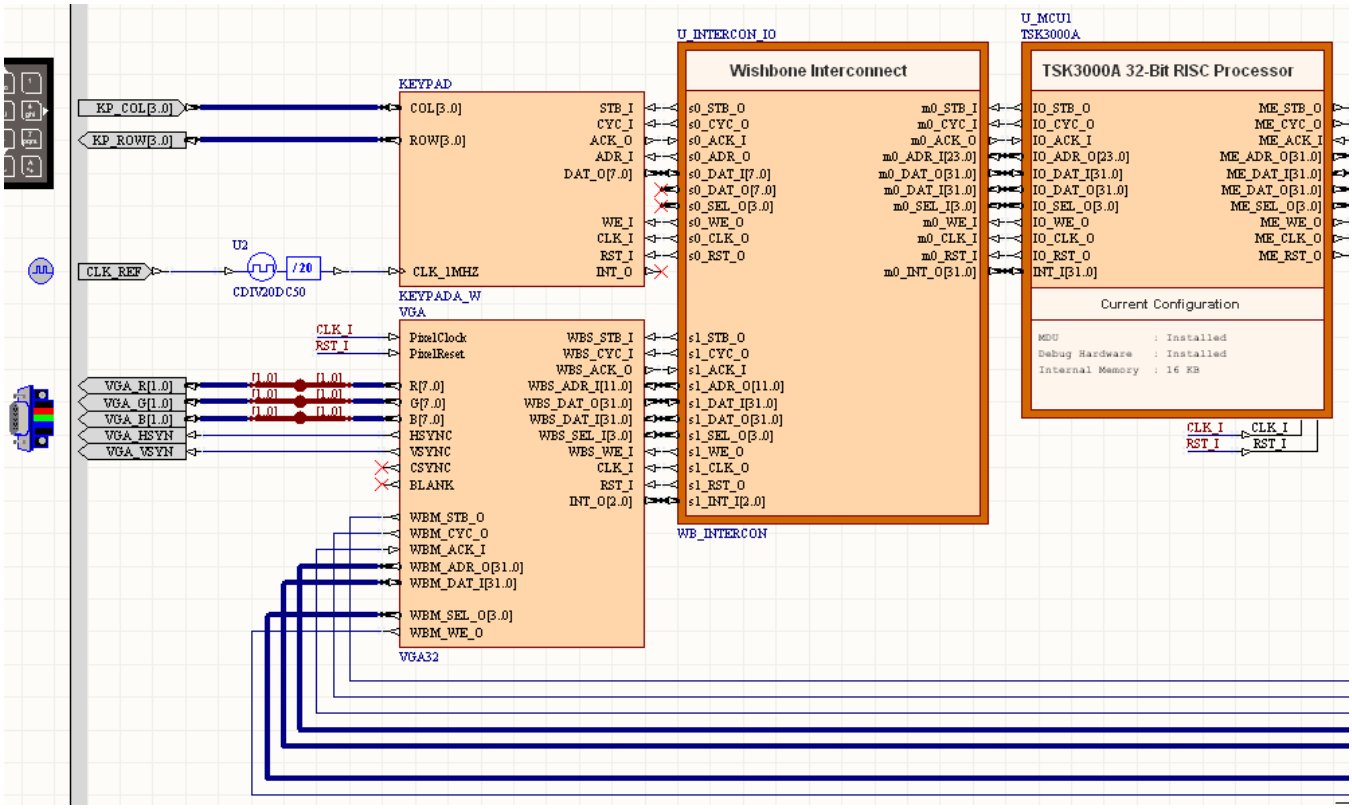


*Figure 3. Example interfacing between a 32-bit processor (TSK3000A) and a VGA32 Controller.*

For further information on the Wishbone Interconnect, refer to the *WB_INTERCON Configurable Wishbone Interconnect* core reference.

For further information on the TSK3000A processor, refer to the *TSK3000A 32-bit RISC Processor* core reference. Similar references can be found for other 32-bit processors supported by Altium Designer, by using the lower section of the **Knowledge Center** panel and navigating to the *Documentation Library » Embedded Processors and Software Development » FPGA Based and Discrete Processors* section.

### Accessing Shared Memory

If the same physical memory device is used for both processing code and video data storage, connection to the memory should be made through either a Wishbone Dual Master or Wishbone Multi-Master device. To avoid screen glitches caused by the video pipeline being starved of data, the WB_DUALMASTER or WB_MULTIMASTER must be set to give higher priority to requests from the VGA32 Controller.

Figure 4 illustrates an example design whereby a single physical memory device is shared between a TSK3000A processor and the memory-based VGA32 Controller. Connection to the memory in this case is made through a Dual Master device, with the master interface to which the VGA32 is connected given high priority.

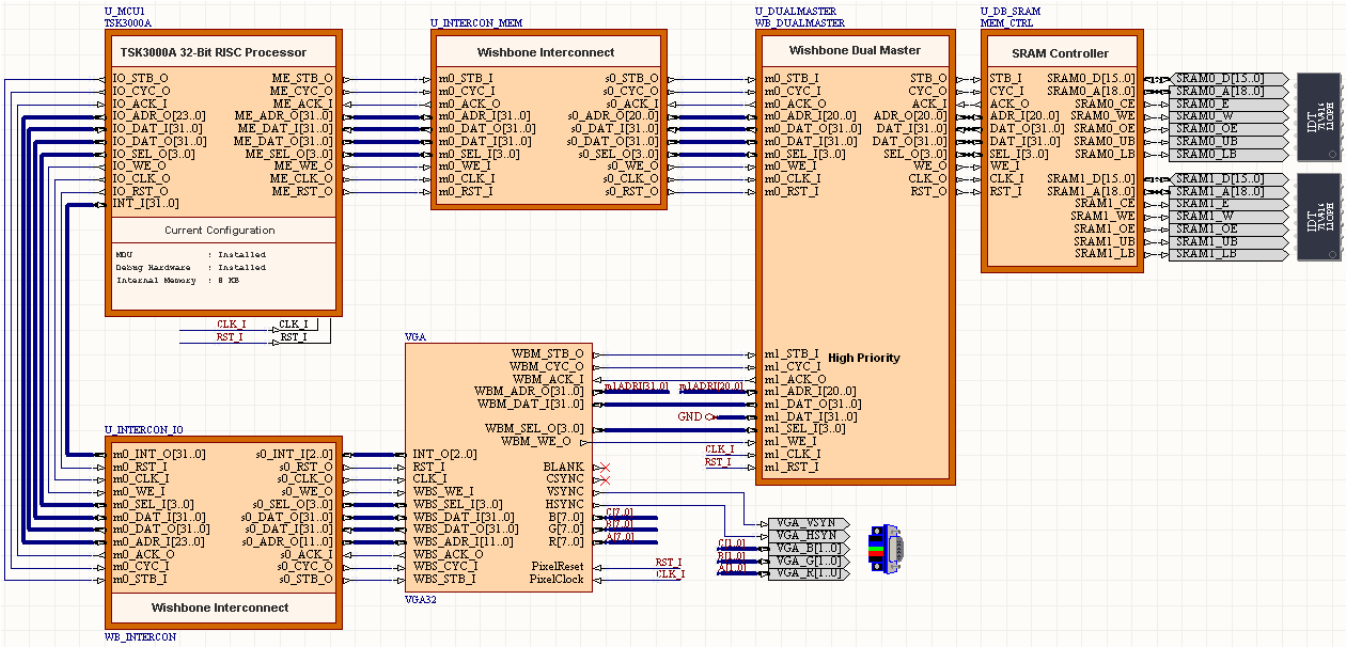


*Figure 4. Sharing physical memory between a 32-bit processor (TSK3000A) and a VGA32 Controller.*

For more information on the Dual Master device, refer to the *WB_DUALMASTER Configurable Wishbone Dual Master* core reference.

For more information on the Multi-Master device, refer to the *WB_MULTIMASTER Configurable Wishbone Multi-Master* core reference.

## Color Output using the NanoBoard-NB1

The R, G and B outputs from the Controller are 8 bits in length, supporting 8-bits per pixel (B&W or color). Together, these outputs form the 24-bit RGB value (True Color) required for driving the red, green and blue color guns of the target monitor.

On a custom/production board, you can take full advantage of this support and the subsequent full range of colors that can be achieved. When using the NanoBoard-NB1, on-board 2-bit digital to analog converter circuitry (Figure 5) provides a limitation to the number of colors that can be used/achieved.

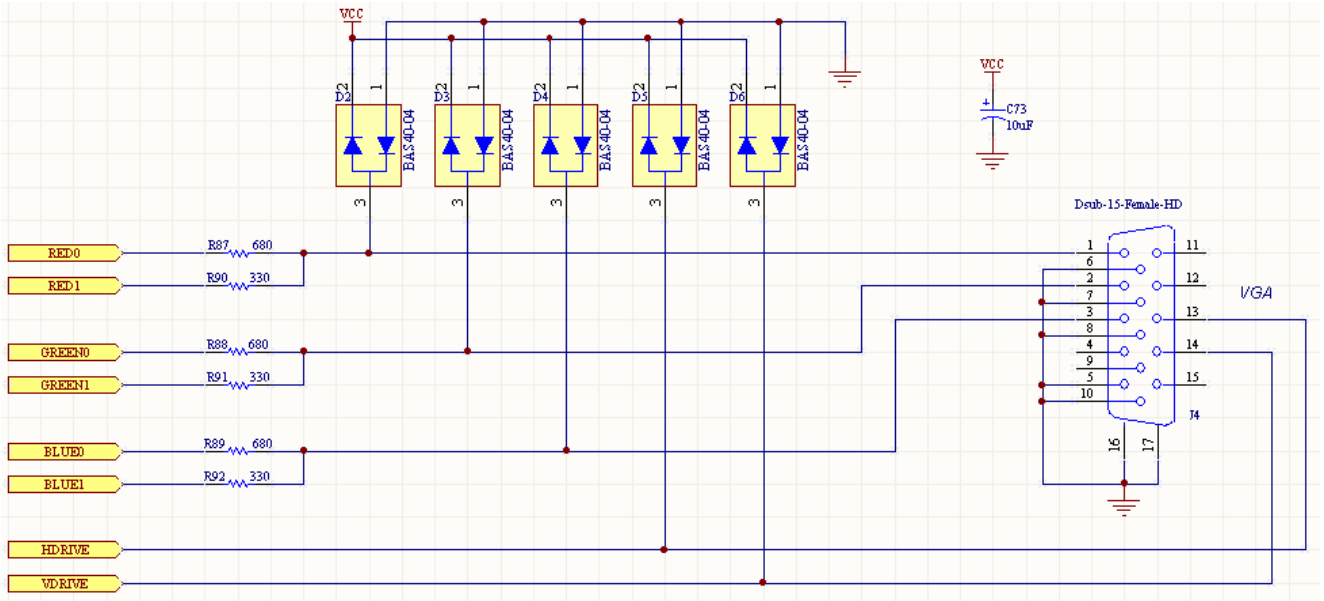


*Figure 5. Digital to Analog RGB conversion*

The 8-bit R, G and B outputs must, in the case of using the NanoBoard-NB1, be reduced to 2-bit signals through the use of relevant bus joiners, as illustrated in Figure 6. The least significant 2 bits are used in each case.

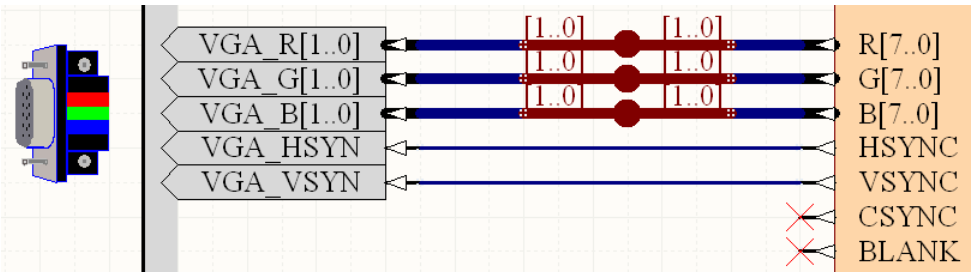


*Figure 6. Color output bit reduction*

For each color, the 2-bit digital signal can be converted into 4 distinct analog levels. These levels specify the intensity of each of the three primary colors to use when displaying the pixel on the monitor's screen. The levels range from 0V (total darkness) to 0.7V (maximum brightness).

With each analog input being one of four possible levels, the monitor can display each pixel on the screen with one of 64 different color permutations.

# Host to Controller Communications

Communications between the 32-bit host processor and the VGA32 Controller are carried out over a standard Wishbone bus interface. The following sections detail the communication cycles involved between host and Controller for writing to/reading from either the internal registers or the active color look-up table.

## Writing to an Internal Register

Data is written from the host processor to an internal register in the VGA32 Controller, in accordance with the standard Wishbone data transfer handshaking protocol.

Bit 11 of the WBS_ADR_I line is used to control direct access between either an internal register or the active color look-up table. This bit must be '0' in order to address and write to, an internal register.

The actual 24-bit address sent out from the processor on its IO_ADR_O line is constructed as follows:

VGA32 Base Address + (Internal Register Address & "00")

The base address for the VGA32 Controller is specified as part of the peripheral's definition when adding it as a slave to the Wishbone Interconnect. For example, if the base address entered for the device is 100000h (mapping it to address FF10_0000h in the processor's address space), and you want to write to the Horizontal Timing Register (HTIM) with address 0000000010b, the value entered on the processor's IO_ADR_O line would be:

100000h + 008h = 100008h

The following sections detail the write operation which, in each case, occurs on the rising edge of the CLK_I input.

### All Writable Registers (except CTRL)

When writing to an internal register, bits 11..2 of the WBS_ADR_I line are used to address the register. The write operation can be summarized as follows:

- The host presents the required 24-bit address based on the register to be written on its IO_ADR_O output and valid data on its IO_DAT_O output. It then asserts its IO_WE_O signal, to specify a write cycle
- The VGA32 receives the 12-bit address on its WBS_ADR_I input and, identifying the addressed register using bits 11..2, prepares to receive data into that register
- The host asserts its IO_STB_O and IO_CYC_O outputs, indicating that the transfer is to begin. The VGA32, which monitors its WBS_STB_I and WBS_CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by latching the data appearing at its WBS_DAT_I input into the target register and asserting its WBS_ACK_O signal – to indicate to the host that the data has been received
- The host, which monitors its IO_ACK_I input on each rising edge of the CLK_I signal, responds by negating the IO_STB_O and IO_CYC_O signals. At the same time, the VGA32 negates the WBS_ACK_O signal and the data transfer cycle is naturally terminated.

### Writing to the CTRL Register

The procedure for writing to the VGA32's Control register (CTRL) is similar to that detailed in the previous section. There are basically two main differences that apply when the VGA32 receives the 12-bit address on its WBS_ADR_I line:

- The address for the register is stored in the 10-bit non-accessible Wishbone Address Register (WAREG), which is loaded as follows:

  WAREG <= "0000" & WBS_ADR_I(7 downto 2)

  The value loaded into this register is then interrogated to determine if the internal register being addressed is indeed the CTRL register
- An internal flag signal is interrogated, to ascertain whether or not the Controller's Wishbone Master interface is currently in use – i.e. Video Memory is currently being accessed. If the Wishbone Master interface is NOT in use, then data from the host processor is latched into the Control register.

Remember that some of the internal registers are not 32-bit. The actual value stored in a register depends on the specific internal register being addressed/written. Table 13 summarizes how the 32-bit data word from the host processor is used by each of the internal registers.

*Table 13. Values loaded into internal registers during a write*

| Internal Register | Value loaded into register |
|---|---|
| CTRL | Entire 32-bit value arriving on WBS_DAT_I |
| HTIM | Entire 32-bit value arriving on WBS_DAT_I |
| VTIM | Entire 32-bit value arriving on WBS_DAT_I |
| HVLEN | Entire 32-bit value arriving on WBS_DAT_I |
| VMBA | WBS_DAT_I(31..2) |
| CDIV | WBS_DAT_I(7..0) |
| CLUTOFF | WBS_DAT_I(7..0) |

## Reading from an Internal Register

Data is read from one of the VGA32's internal registers in accordance with the standard Wishbone data transfer handshaking protocol.

Bit 11 of the WBS_ADR_I line is used to control direct access between either an internal register or the active color look-up table. This bit must be '0' in order to address and read from, an internal register.

The 10-bit Wishbone Address Register is again used to store the address of the register to be read ("0000" & WBS_ADR_I(7..2)). The actual data to be sent to the host processor is stored in an additional non-accessible internal register – the 32-bit Wishbone Data Output register (WDOUT).

The read operation occurs on the rising edge of the CLK_I input and can be summarized as follows:

- The host presents the required 24-bit address based on the register to be read on its IO_ADR_O output. It then negates its IO_WE_O signal, to specify a read cycle
- The VGA32 receives the 12-bit address on its WBS_ADR_I input and loads the WAREG register accordingly
- Having identified the addressed register, the Controller uses that register's contents to load the WDOUT register
- The host asserts its IO_STB_O and IO_CYC_O outputs, indicating that the transfer is to begin. The VGA32, which monitors its WBS_STB_I and WBS_CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by presenting the valid data stored in the WDOUT register on its WBS_DAT_O output and asserting its WBS_ACK_O signal – to indicate to the host that valid data is present
- The host, which monitors its IO_ACK_I input on each rising edge of the CLK_I signal, responds by latching the data appearing at its IO_DAT_I input and negating the IO_STB_O and IO_CYC_O signals. At the same time, the VGA32 negates the WBS_ACK_O signal and the data transfer cycle is naturally terminated.

The actual value stored in the WDOUT register and subsequently sent to the host processor depends on the specific internal register being addressed/read. Table 14 summarizes the 'make-up' of the 32-bit data word that is read back from each register.

*Table 14. Values read from internal registers during a read*

| Internal Register | Value loaded into WDOUT register (and presented to host processor) |
|---|---|
| CTRL | 32-bit value currently in the CTRL register |

| Internal Register | Value loaded into WDOUT register (and presented to host processor) |
|---|---|
| STATUS | "00000000000000000000000000" & STATUS.4 & "0000" |
| HTIM | 32-bit value currently in the HTIM register |
| VTIM | 32-bit value currently in the VTIM register |
| HVLEN | 32-bit value currently in the HVLEN register |
| VMBA | 30-bit value in the VMBA register & "00" |
| CDIV | "000000000000000000000000" & 8-bit value currently in the CDIV register |
| CLUTOFF | "000000000000000000000000" & 8-bit value currently in the CLUTOFF register |

## Writing to the Active Color Look-up Table

Data is written from the host processor to a location in the VGA32 Controller's active color look-up table, in accordance with the standard Wishbone data transfer handshaking protocol.

Bit 11 of the WBS_ADR_I line is used to control direct access between either an internal register or the active color look-up table. This bit must be '1' in order to address and write to, the active color look-up table.

When writing to the active color look-up table, bits 10..2 of the WBS_ADR_I line are used to address the required entry in the table. The two look-up tables are actually part of the same contiguous address space. This 9-bit address provides access to each of the 512 32-bit entries in this address space.

The actual 256 addresses available at any one time depend on the state of the ltb bit in the Control register (CTRL.4). When '0', the lower 256 addresses are addressable (CLUT0), when '1', the upper 256 addresses are addressable (CLUT 1).

The actual 24-bit address sent out from the processor on its IO_ADR_O line is constructed as follows:

VGA32 Base Address + ("1" & CLUT Address & "00")

For example, if the base address entered for the device is 100000h (mapping it to address FF10_0000h in the processor's address space), and you want to write to address 17Fh in CLUT0 (let's assume this is currently the active look-up table), the value entered on the processor's IO_ADR_O line would be:

100000h + DFCh = 100DFCh

The write operation can be summarized as follows:

- The host presents the required 24-bit address, based on the address in the look-up table to be written, on its IO_ADR_O output and valid data on its IO_DAT_O output. It then asserts its IO_WE_O signal, to specify a write cycle
- The VGA32 receives the 12-bit address on its WBS_ADR_I input and, identifying the addressed location in the active look-up table using bits 10..2, prepares to receive data into that location
- The host asserts its IO_STB_O and IO_CYC_O outputs, indicating that the transfer is to begin. The VGA32, which monitors its WBS_STB_I and WBS_CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by latching the lower 24 bits of data appearing at its WBS_DAT_I input into the target location and asserting its WBS_ACK_O signal – to indicate to the host that the data has been received
- The host, which monitors its IO_ACK_I input on each rising edge of the CLK_I signal, responds by negating the IO_STB_O and IO_CYC_O signals. At the same time, the VGA32 negates the WBS_ACK_O signal and the data transfer cycle is naturally terminated.

## Reading from the Active Color Look-up Table

Data is read from a location in the VGA32's active color look-up table in accordance with the standard Wishbone data transfer handshaking protocol.

Bit 11 of the WBS_ADR_I line is used to control direct access between either an internal register or the active color look-up table. This bit must be '1' in order to address and read from, the active color look-up table.

The read operation occurs on the rising edge of the CLK_I input and can be summarized as follows:

- The host presents the required 24-bit address, based on the address in the look-up table to be read, on its IO_ADR_O output. It then negates its IO_WE_O signal, to specify a read cycle
- The VGA32 receives the 12-bit address on its WBS_ADR_I input and, identifying the addressed location in the active look-up table using bits 10..2, prepares to transmit data from that location

- The host asserts its IO_STB_O and IO_CYC_O outputs, indicating that the transfer is to begin. The VGA32, which monitors its WBS_STB_I and WBS_CYC_I inputs on each rising edge of the CLK_I signal, reacts to this assertion by presenting a valid data word (“00000000” & CLUT Data at addressed location) on its WBS_DAT_O output and asserting its WBS_ACK_O signal – to indicate to the host that valid data is present
- The host, which monitors its IO_ACK_I input on each rising edge of the CLK_I signal, responds by latching the data appearing at its IO_DAT_I input and negating the IO_STB_O and IO_CYC_O signals. At the same time, the VGA32 negates the WBS_ACK_O signal and the data transfer cycle is naturally terminated.

## Timing Information

The following tables provide information for setting the Controller’s internal timing registers (HTIM and VTIM), based on a variety of different monitor resolutions and refresh rates.

*Table 15. Horizontal timing information*

| Resolution | Refresh Rate (Hz) | Pixel Clock (MHz) | Number of pixels per line | Sync pulse length (pixels) | Horizontal back porch (pixels) | Number of visible pixels |
|---|---|---|---|---|---|---|
| 640x480 | 60 | 25.175 | 800 | 96 | 51 | 640 |
| 640x480 | 72 | 31.5 | 832 | 40 | 131 | 640 |
| 720x400 | 70 | 28.322 | 900 | 108 | 57 | 720 |
| 720x350 | 70 | 28.322 | 900 | 108 | 57 | 720 |
| 800x600 | 56 | 36 | 1024 | 72 | 131 | 800 |
| 800x600 | 60 | 40 | 1056 | 128 | 91 | 800 |
| 800x600 | 72 | 50 | 1040 | 120 | 67 | 800 |
| 640x480 | 75 | 31.5 | 800 | 96 | 51 | 640 |
| 640x480 | 66 | 30.24 | 864 | 64 | 99 | 640 |

*Table 16. Vertical timing information*

| Resolution | Refresh Rate (Hz) | Number of lines per frame | Sync pulse length (lines) | Vertical back porch (lines) | Number of visible lines |
|---|---|---|---|---|---|
| 640x480 | 60 | 525 | 2 | 34 | 480 |
| 640x480 | 72 | 520 | 3 | 30 | 480 |
| 720x400 | 70 | 449 | 2 | 36 | 400 |
| 720x350 | 70 | 449 | 2 | 61 | 350 |
| 800x600 | 56 | 625 | 1 | 24 | 600 |
| 800x600 | 60 | 628 | 4 | 25 | 600 |
| 800x600 | 72 | 666 | 6 | 25 | 600 |
| 640x480 | 75 | 525 | 2 | 34 | 480 |
| 640x480 | 66 | 525 | 3 | 41 | 480 |

## Device Driver Code

Various C-header files are available with respect to the VGA32 Controller, providing definitions and built-in functions that enable you to write source code for working with the Controller quickly and efficiently – a one-stop shop of predefined code building blocks if you like. The following files are available from the `\Examples\Reference Designs\Device Driver Code` folder of the installation:

- `io_wb_vga.h` (and corresponding .C file) – providing various text-based routines
- `wb_vga.h` (and corresponding .C file) – providing various drawing and display functions
- `wb_vga_defs.h` – providing definitions for various horizontal and vertical timing modes.

## Revision History

| Date | Version No. | Revision |
|---|---|---|
| 07-Aug-2006 | 1.0 | Initial release |
| 11-Mar-2008 | 2.0 | Updated for Altium Designer Summer 08 |

Software, hardware, documentation and related materials:

Copyright © 2008 Altium Limited.

All rights reserved. You are permitted to print this document provided that (1) the use of such is for personal use only and will not be copied or posted on any network computer or broadcast in any media, and (2) no modifications of the document is made. Unauthorized duplication, in whole or part, of this document by any means, mechanical or electronic, including translation into another language, except for brief excerpts in published reviews, is prohibited without the express written permission of Altium Limited. Unauthorized duplication of this work may also be prohibited by local statute. Violators may be subject to both criminal and civil penalties, including fines and/or imprisonment. Altium, Altium Designer, Board Insight, Design Explorer, DXP, LiveDesign, NanoBoard, NanoTalk, P-CAD, SimCode, Situs, TASKING, and Topological Autorouting and their respective logos are trademarks or registered trademarks of Altium Limited or its subsidiaries. All other registered or unregistered trademarks referenced herein are the property of their respective owners and no trademark rights to the same are claimed.