Automotive Graphics System Reference Design

Version 1.0, December 2004 Application Note 371

Introduction

The Altera® Automotive Graphics System Reference Design demonstrates Altera Cyclone™ FPGAs in a graphics system targeted at the automotive sector. The reference design runs on a Nios development board, Cyclone edition with a Lancelot VGA video controller and has the following features:

- SDRAM program store and frame buffer
- Video input with clipping, color space conversion, and horizontal and vertical scaling
- LCD controller supporting 5-layer display, picture-in-picture
- Fits on a Cyclone 1C12 device

For information on the Nios development board, Cyclone edition, see the *Nios II Development Kit, Cyclone Edition Getting Started User Guide* and the *Nios Development Board Reference Manual, Cyclone Edition*.

For information on the Lancelot VGA video controller, see **www.altera.com/products/devkits/partners/kit-lancelot.html**.

Reference Design Hardware

Figure 1 shows the reference design block diagram.

Figure 1. Block Diagram

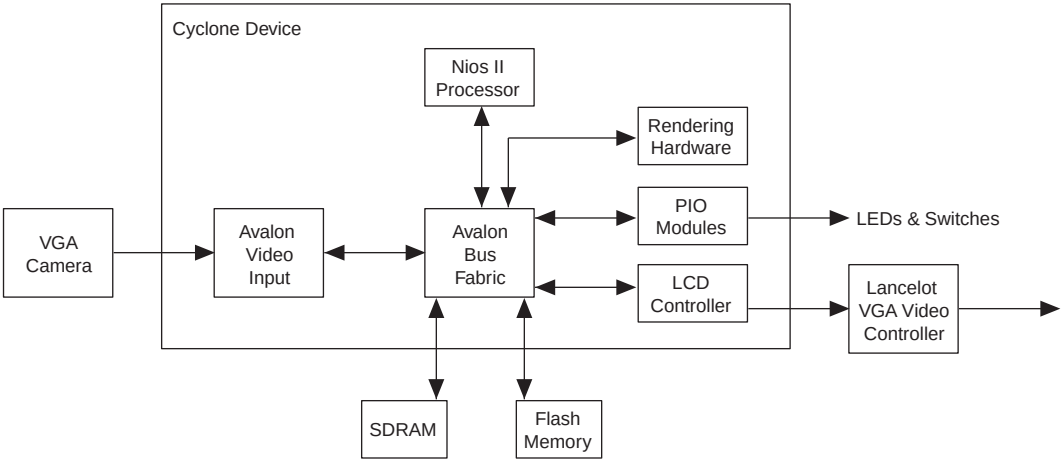


Table 1 shows the reference design PLL usage.

Table 1. PLL Usage

PLL	Pin	Frequency (MHz)	Purpose
PLL1	Input	50	On-board crystal.
	e0	25	Lancelot VGA board and feedback to PLL2 input.
	c0	26	Camera module and video input.
	c1	25	Pixel clock to LCD controller.
PLL2	Input	25	Feedback from PLL1.
	e0	75	SDRAM clock.
	c0	75	Nios II, Avalon™ interface, and video clock.
	c1	–	Spare.

One additional global clock input is used as the input clock from the camera module.

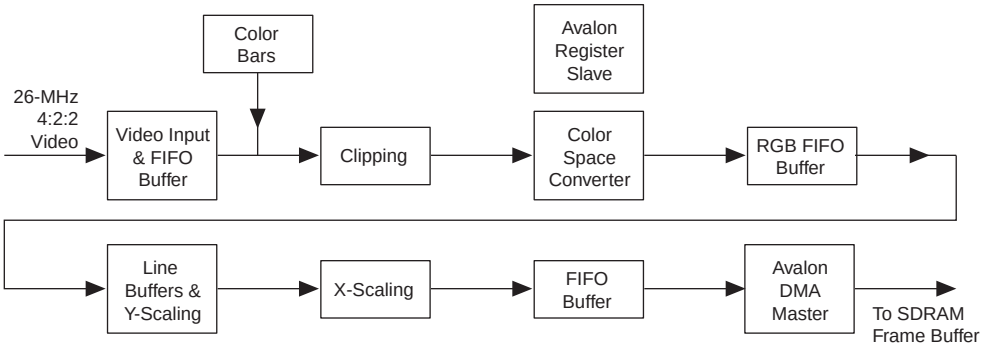
Video Input Module

The Avalon video input module provides a flexible video capture solution that may be implemented in Altera Cyclone II, Cyclone, Stratix® II, Stratix GX, or Stratix devices. The Avalon video input module has the following features:

- Component video interface to VGA camera module
- Color-bar test pattern generator
- Clipping of input image
- Horizontal (Y) scaling of input image
- Vertical (X) scaling of input image
- Avalon direct memory access (DMA) master to write image(s) to frame buffer memory
- Avalon register slave for control and status

Figure 2 shows the video input module.

Figure 2. Video Input Module



For more details on the Avalon video input module see *AN373: Avalon Video Input Module*.

LCD Controller

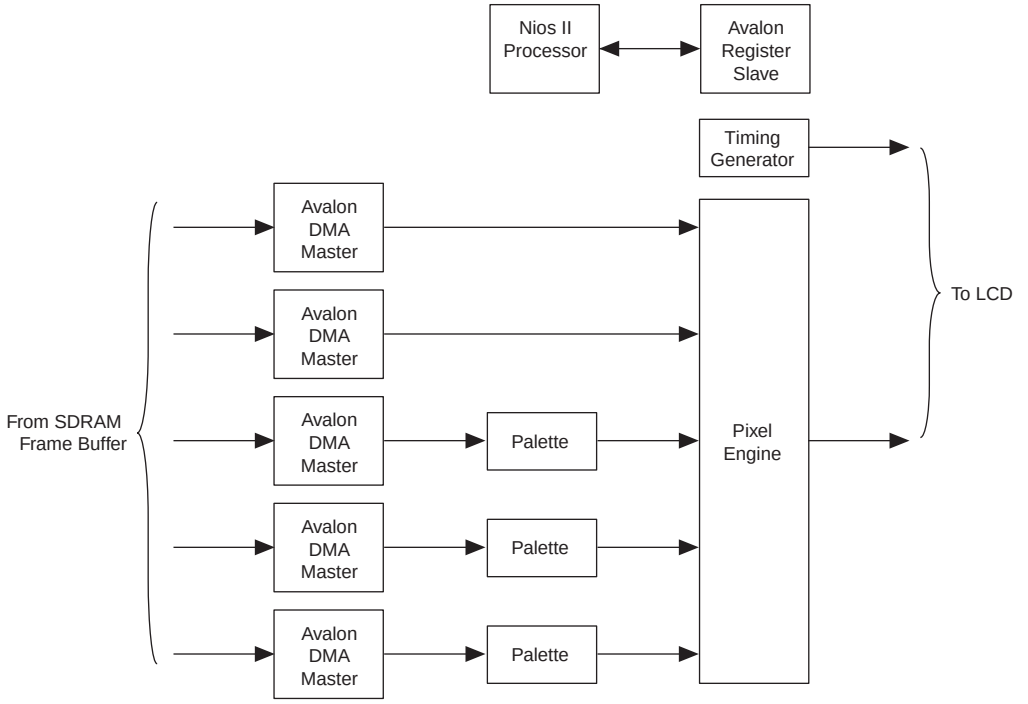
The Avalon LCD controller module provides a flexible video display solution for driving LCDs with the following features:

- Avalon DMA master for frame buffer access
- Avalon register slave interface for control and status
- Backdrop layer—16 bit RGB 565
- Video layer—16 bit RGB 565, layer alpha

- Three drawing layers—8 bit palette, RGB666 + 6 bit pixel alpha, layer alpha
- Picture-in picture

Figure 3 shows the LCD controller.

Figure 3. LCD Controller



For more details on the Avalon LCD controller, see *AN372: Avalon LCD Controller*. For more information on the Avalon Bus, see the *Avalon Bus Specification Reference Manual*. For more information on SOPC Builder, see the *SOPC Builder User Guide* and *AN 333: Developing Peripherals for SOPC Builder*.

Nios II Processor

The Nios II processor is generated using Altera's SOPC Builder tool. A variety of configurations are possible including data and instruction caches of varying size.

Avalon Bus Fabric

The Avalon bus fabric is generated using Altera's SOPC Builder tool.

Flash Memory

Flash memory stores the FPGA image, reference design software, and images to be displayed by the LCD controller.

Images for display are stored in the zip file system created in the Nios II integrated design environment (IDE).

SDRAM

At power up, the reference design software is copied from flash memory to SDRAM, where it is executed.

The SDRAM also holds the frame buffers for the video input and LCD controller modules.

PIO Modules

Three PIO modules interface to LEDs, 7-segment LEDs, and switches on the Nios development board, Cyclone edition.

Hardware-Accelerated 2D Rendering

The optional rendering hardware supports acceleration of 2D rendering by providing hardware rendering for filled rectangles. If the hardware is included, the software detects its presence through the system definition generated by SOPC Builder. If the hardware is present, there is a `#define` for the base address, which can be tested for. The software uses hardware rendering for the following purposes:

- Horizontal line drawing
- Vertical line drawing
- Polygon filling

Software Concepts

This section discusses the following topics:

Frame Buffers In SDRAM

The LCD controller requires that the start address of each frame buffer is word aligned.

Frame buffers should be located in non-cacheable memory regions.

16-Bit Frame Buffers

Layers 1 and 2 in the LCD controller use 16 bits per pixel to represent 65,536 colors using RGB565 encoding.

The frame buffer memory for these layers must be allocated with two bytes per pixel, e.g., a 640 × 480 display requires a frame buffer of 614,400 bytes.

8-Bit Frame Buffers

Layers 3 to 5 in the LCD controller use 8 bits per pixel as an index into a 256 entry color palette. Each palette entry is 24 bits comprising 18-bit color in RGB666 format plus a 6-bit pixel alpha value (pixel transparency). The reference design does not allow writing of the palette memory. Instead, the palette must be predetermined and the palette memory data included in the Quartus II compilation of the design. Each of the 8-bit layers has its own palette, but the design currently initializes each palette with same contents.

Frame buffer memory for these layers must be allocated with one byte per pixel, e.g., a 640 × 480 display requires a frame buffer of 307,200 bytes.

Interrupts

One interrupt is available from the Avalon video input module at the end of each captured video frame.

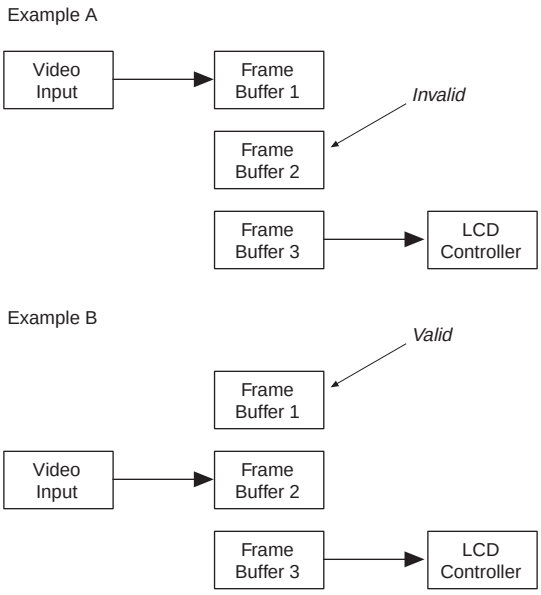
Another interrupt is available from the LCD controller at the end of each displayed frame.

The two interrupts allow updating of video input and LCD controller registers during frame blanking periods, to ensure flicker-free display of video and drawn objects.

Video Capture & Display

To ensure a stable, flicker-free display, when the input video and LCD frame rates differ, three frame buffers must be employed in a circular fashion to synchronize the different frame rates. The video input must complete writing to a frame buffer before it can be used for display by the LCD controller. At any time, the video input is writing to one frame buffer and the LCD controller is reading from a second frame buffer. The third frame buffer is either invalid or holds a just-written video frame ready to be displayed. Figure 4 shows these two situations.

Figure 4. Frame Buffers for Video Capture & Display



When the video input interrupts at the end of a frame, software must determine if the next frame buffer is free for use or is still being used by the LCD controller for display. If the next frame buffer is in use, the video input must re-use the buffer it just wrote for the next video frame (i.e., a video frame is lost). This situation occurs when the video frame rate is greater than the LCD frame rate.

When the LCD controller interrupts at the end of a frame, software must determine if the next frame buffer is available for display or is still being written by the video input. If the next frame buffer is still being written,

the LCD controller must re-use the buffer it just displayed for the next frame (i.e., a video frame is displayed twice). This situation occurs when the video frame rate is less than the LCD frame rate.

Updating the Control Registers

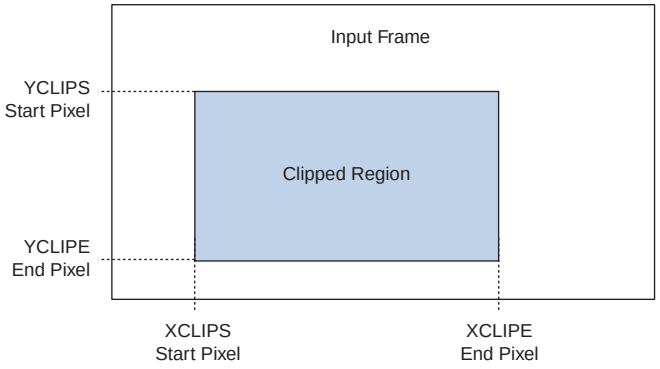
Control registers for the video in module may only be changed during the video input frame blanking period and with the video input disabled.

Control registers for the LCD controller may only be changed during the LCD controller frame blanking period and with the associated LCD controller layer disabled.

Video Clipping & Resizing

The clipping function allows an arbitrary rectangular area of the input video frames to be selected. This clipped area (which may be the whole of the input frame) can then be independently resized in the horizontal and vertical directions. Figure 5 shows the video input clipping.

Figure 5. Video Input Clipping



Ensure that the LCD controller is set up to display the same size image as that generated from the video input by clipping and resizing.

Image Display

Images, other than video input, may be displayed by either loading them into a frame buffer from a zip file system in the flash memory or by having the processor render the image into a frame buffer.

Using the Flash Zip File System

In the reference design, two images are stored in the flash file system. One is a 16-bit per pixel image used as a background for display on layer 0. The other is an 8-bit per pixel text image with a transparent background. These images are copied from the zip file system into appropriate frame buffers to enable them to be displayed.

Using the Nios II Processor for Image Rendering

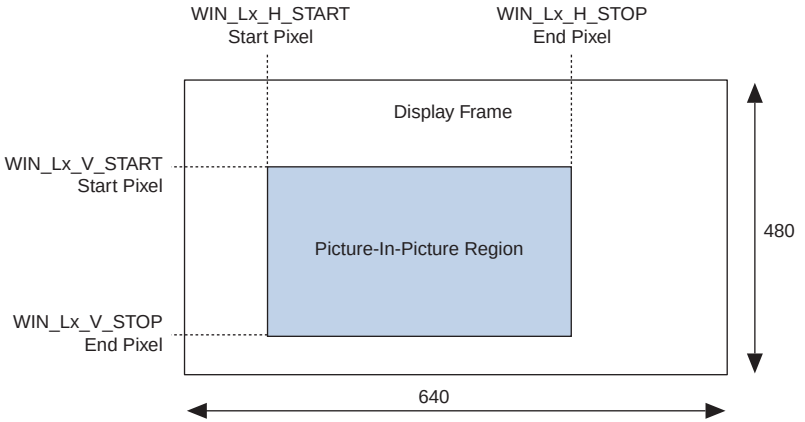
The software includes functions to support 2D rendering by the Nios II processor. The hardware rendering module (if present) accelerates some of the following functions:

- Line draw
- Polygon draw
- Polygon fill
- Circle draw
- Circle fill
- Text

Picture in Picture

The picture-in-picture feature, available on display layers 1 to 4, allows an image in a small frame buffer to be displayed anywhere on the screen. The advantage over using a full screen sized frame buffer with transparent borders is the use of less frame buffer memory bandwidth for the associated layer. Figure 6 shows the picture-in-picture control registers.

Figure 6. Picture-in-Picture Control Registers



Cursor Display

A hardware cursor can be implemented by dedicating one display layer to the cursor and using the picture-in-picture feature to display a small image (e.g., 32 × 32 pixels) as a cursor. The cursor position should be updated during the LCD controller frame blanking period by updating the picture-in-picture control registers (see "Updating the Control Registers" on page 8).

Reference Design Software

The reference design software is compiled and downloaded to the Nios development board, Cyclone edition, using the Nios II IDE. The compiled software may also be programmed into the on-board flash memory with the FPGA image and display images using the flash programmer within the IDE.

The video input is based on the output from a VGA camera module (part number DA3530-30XF1) from Dialog Semiconductor.

For more details on the Nios II IDE see the *Nios II Software Developers Handbook*.

You can run a number of different software modules; you select them by closing one of the push-button switches while resetting the Nios development board, Cyclone edition.

Reference Design Demonstration

The main reference design demonstration runs if no switches are closed during reset. The demonstration cycles through the following actions:

- Fade in background image
- Fade in/out introductory text image
- Fade in/out full frame video
- Fade in/out half frame video
- Dynamic video resizing
- Move picture-in-picture video around the screen
- Fade in/out end text image
- Fade out background image

Performance Tests

The 2D rendering performance tests run if switch SW0 is closed during reset. The rendering performance tests cycle through the following actions and report results in terms of the number of objects drawn:

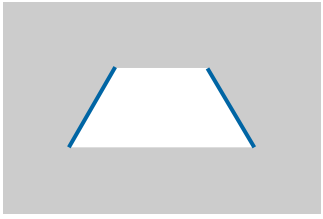
- Short random lines constrained to be within one quarter of the screen
- Long random lines over the whole screen
- Horizontal line 80 pixels
- Vertical line 60 pixels
- Diagonal line 100 pixels
- Horizontal line 160 pixels
- Vertical line 120 pixels
- Diagonal line 200 pixels
- Horizontal line 320 pixels
- Vertical line 240 pixels
- Diagonal line 400 pixels
- Horizontal line 640 pixels
- Vertical line 480 pixels
- Diagonal line 800 pixels
- 4 vertex polygon within an 80 × 60 box
- 4 vertex polygon within a 160 × 120 box
- 4 vertex polygon within a 240 × 180 box
- 4 vertex polygon within a 320 × 240 box
- 4 vertex polygon within a 400 × 300 box
- 4 vertex polygon within a 480 × 360 box
- 4 vertex polygon within a 560 × 420 box
- 4 vertex polygon within a 640 × 480 box

Reverse Parking Aid

A simple reverse parking aid demonstration is run if SW1 is closed during reset.

This demonstration shows a live video image from the camera module. The video is overlaid with a trapezoid clear region in the center of the screen surrounded by a partially opaque border. This overlay uses one 8-bit layer filled with a 50% alpha-value black. The trapezoidal region is then rendered with a 0 alpha value (transparent). Two lines are drawn as an aid to represent the direction of travel of a reversing vehicle. Figure 7 shows the reverse-parking aid overlay.

Figure 7. Reverse-Parking Aid Overlay



Getting Started

The getting started involves the following steps:

1. "Hardware & Software Requirements"
2. "Install the Design"

Hardware & Software Requirements

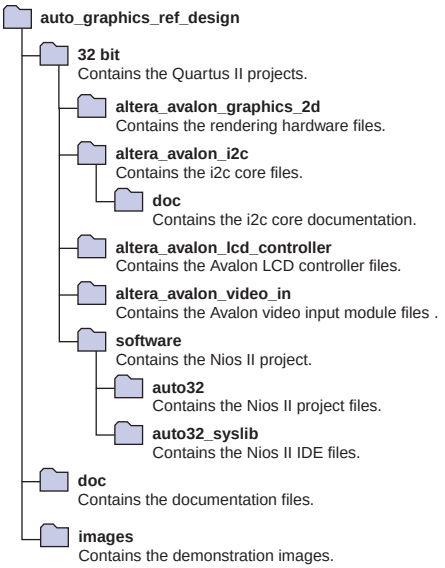
The reference design requires the following hardware and software:

- Nios II Development Kit, Cyclone Edition
- Lancelot VGA video controller
- Quartus® II software version 4.2, or higher
- Verilog HDL simulator (e.g., Synopsys VCS version 6.0.1 or ModelSim version 5.7e)

Install the Design

To install the reference design, run the **.exe** and follow the installation instructions. Figure 8 shows the directory structure.

Figure 8. Directory Structure

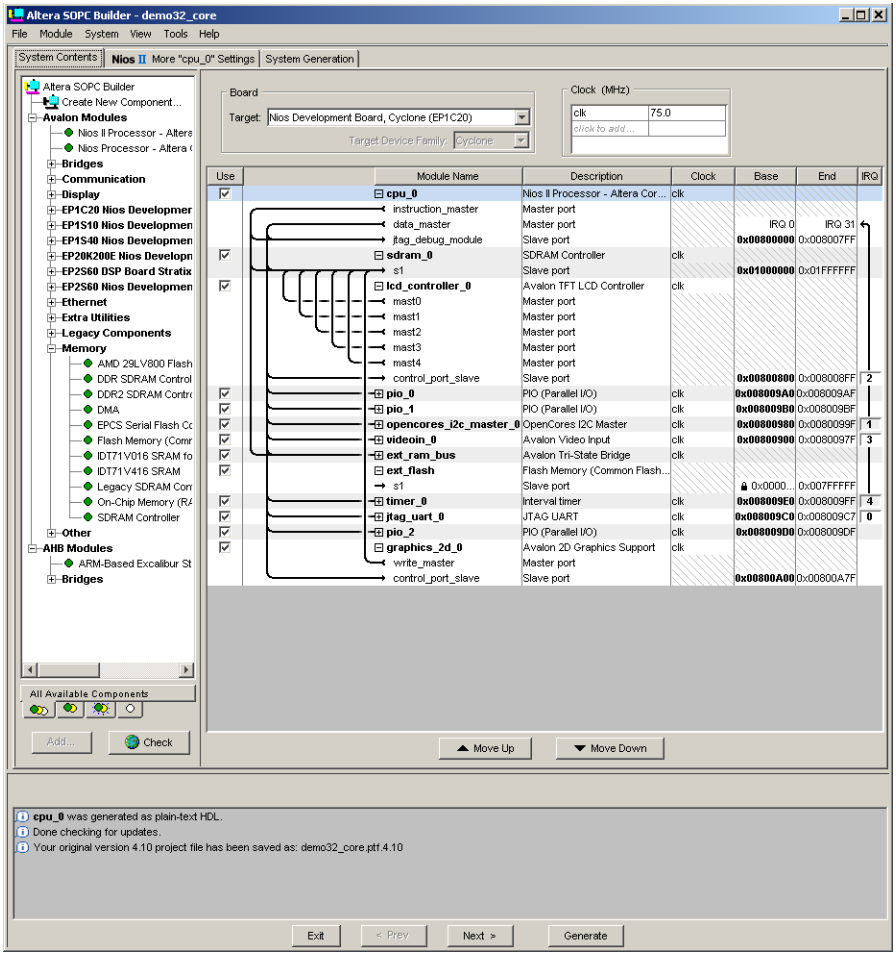


Run SOPC Builder

To open and generate the SOPC Builder project, follow these steps:

1. Start the Quartus II software.
2. Choose **Open Project** (File menu), choose **\auto_graphics_ref_design\32bit\demo32.qpf**, and click **Open**.
3. Choose **SOPC Builder** (Tools menu), see Figure 9.

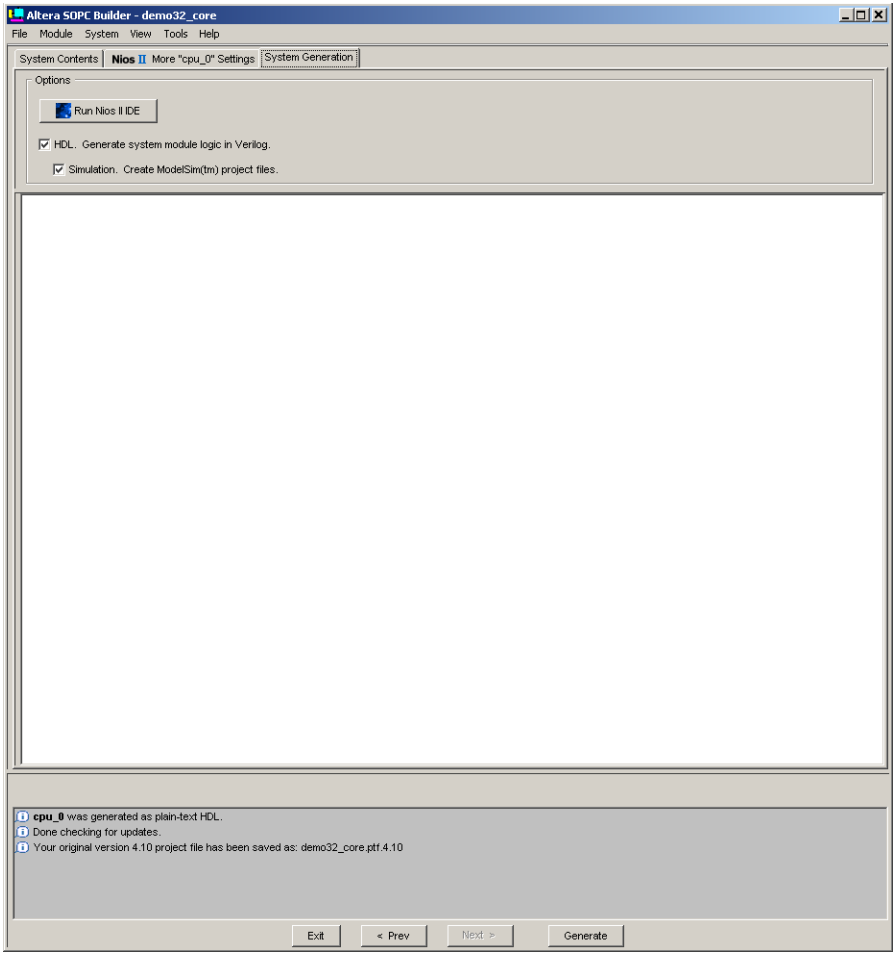
Figure 9. SOPC Builder



4. Turn off the **graphics_2d_0** check box, if you do not want hardware rendering.

5. Click the **System Generation** tab and click **Generate** (see Figure 10).

Figure 10. Generate



6. When the system generation has completed, click **Exit**.

Compile in the Quartus II Software

To compile the demonstration in the Quartus II software choose **Start Compilation** (Tools menu).

Run the Demonstration

1. Connect the Lancelot VGA controller and your PC to the Nios development board, Cyclone edition.

2. Power up the Nios development board, Cyclone edition.

For more information on connecting and using the Nios development board, Cyclone edition, see the *Nios II Development Kit, Cyclone Edition Getting Started User Guide*.

3. Use the Nios II IDE to download the Nios II project to the Nios development board, Cyclone edition.

    a. Open

For information on the software modules that you can run, see "Reference Design Software" on page 10.

Performance

The reference design f_{MAX} is limited by the Avalon bus matrix because of the large number of masters connected to the SDRAM controller. In an EP1C20F400I7 device the f_{MAX} is 75 MHz. The f_{MAX} is higher for designs using fewer layers in the LCD controller. A large improvement in f_{MAX} could be gained by designing the LCD controller with a single DMA master servicing all layers, rather than the current scheme of one DMA master per layer.

Utilization

The complete reference design based upon a Nios II(s) processor with 512 byte instruction cache uses approximately 10,700 logic cells and 33 M4K memory blocks when implemented in the Cyclone family.

The complete reference design based upon a Nios II(f) with 4 Kbyte instruction cache and 4 Kbyte data cache uses approximately 11,300 logic cells and 54 M4K memory blocks when implemented in the Cyclone family.

A Stratix or Cyclone II implementation could use fewer logic cells by employing hardware multipliers.

101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
www.altera.com
Applications Hotline:
(800) 800-EPLD
Literature Services:
lit_req@altera.com