Nios II 3C25 Microprocessor with LCD Controller Data Sheet

DS-01003-1.1

Introduction

This data sheet describes a single instance of a Nios® II-based processor system with a built-in LCD controller targeted for an Altera® Cyclone® III 3C25F324 FPGA on the Altera Nios II Embedded Evaluation Kit, Cyclone III Edition. The Nios II 3C25 Microprocessor with LCD Controller is a complete system-on-a-programmable-chip (SOPC) solution that incorporates a rich set of system peripherals and standard interfaces for a wide range of embedded applications involving video processing and LCD touch panel control.

The key benefit of implementing a processor system in an FPGA is that you can customize your system using intellectual property (IP) cores, custom logic, and hardware acceleration to optimize to your target application. Nearly every feature in the system is available for you to configure, customize, duplicate, or remove easily. You can further enhance your system by adding additional IP to the Cyclone III 3C25 device, or you can remove IP or select options that reduce logic utilization allowing you to port to a smaller device to reduce cost.

Although this data sheet describes a system targeted for the Cyclone III 3C25 FPGA device, the data sheet also shows you how to configure the processor system for another Altera FPGA device and hardware platform of your choice.

Features

The following list summarizes the main features of the Nios II 3C25 microprocessor with LCD controller.

Target Hardware Board

- Altera Nios II Embedded Evaluation Kit, Cyclone III Edition

Device

- System name: cycloneIII_3c25_niosII_video
- Family: Cyclone III
- Device: 3C25F324
- Total logic elements (LE) used: 22,875 / 24,624 (93%)
- Total pins used: 167 / 216 (77%)
- Total memory used: 163,270 / 608,256 (27%)

Processor

- Nios II/f processor core
- Nominal metrics: 113 DMIPS at 100 MHz, 1,400–1,800 LEs, MMU/MPU option disabled
- 4-KByte instruction cache, 2-KByte data cache

- JTAG debug module for downloading software, 300–400 LEs

Memory Interfaces

- Common flash interface (CFI) flash memory
  - 16 MBytes
- High performance DDR SDRAM memory
  - Nominal frequency: 133 MHz, 16 bits, 32 MBytes
- SD/MMC card serial peripheral interface (SPI)
  - 20-MHz SPI interface clock frequency
  - Supports up to 1-Gbit SD card memory
- Synchronous SRAM memory
  - 1 MByte

Communication Interfaces

- Ethernet MAC 10/100/1000 Base T
  - Integrated in receive and transmit FIFO 512 × 32 bits each
  - Media independent interface (MII)/gigabit media independent interface (GMII) support
  - 32-bit transmit and receive scatter gather direct memory access (SG-DMA) channels
- JTAG UART with integrated read and write FIFO
- UART for RS-232 serial communication
  - 115,200 baud rate, no parity, 8 data bits, 1 stop bit
- 2-wire interface
  - Implemented using general purpose PIOs
  - Dedicated to LCD controller interface

Video Subsystem

- Integrated LCD controller IP
  - Configured to 800 × 480 resolution
  - Interface for LCD control using 2-wire interface
    - Implemented using general purpose PIOs
- Integrated touch panel controller IP
  - Interfaces to LCD using 3-wire SPI
  - Master mode 8 bit data register, 32 KHz

- Video pipeline
  - Streaming video data path
  - Video frame buffer
  - RGB Sync generation IP
  - 128-byte dual clock FIFO

System Peripherals

- Timers/counters
  - System clock timer
    - 32-bit counter size, 10-ms time-out period
  - High resolution timer
    - 32-bit counter size, 10-µs time-out period
  - Performance counter
    - 1 simultaneous measured section
- 4 button PIOs (input only)
- 2 LED PIOs (output only)
- System ID
- Cyclone III remote update controller configuration peripheral

Description

The Nios II 3C25 microprocessor with LCD controller incorporates a Nios II processor, LCD controller, memories, a video pipeline, and more in a single Cyclone III 3C25 FPGA. You can configure nearly every aspect of the processor system to suit your application requirements. You can configure the Nios II processor as one of following cores:

- A size-optimized economy (/e) core
- A performance-optimum fast (/f) core
- An optimum size-to-performance standard (/s) core

In addition, the fast core comes with options to include a memory management unit (MMU) as well as various precise exceptions and memory protection features. The Nios II processor supports custom instructions allowing you to implement software functions in hardware to increase system performance. You can configure the JTAG debug module to support hardware breakpoints, data triggers, and instruction and data on-chip and off-chip trace.

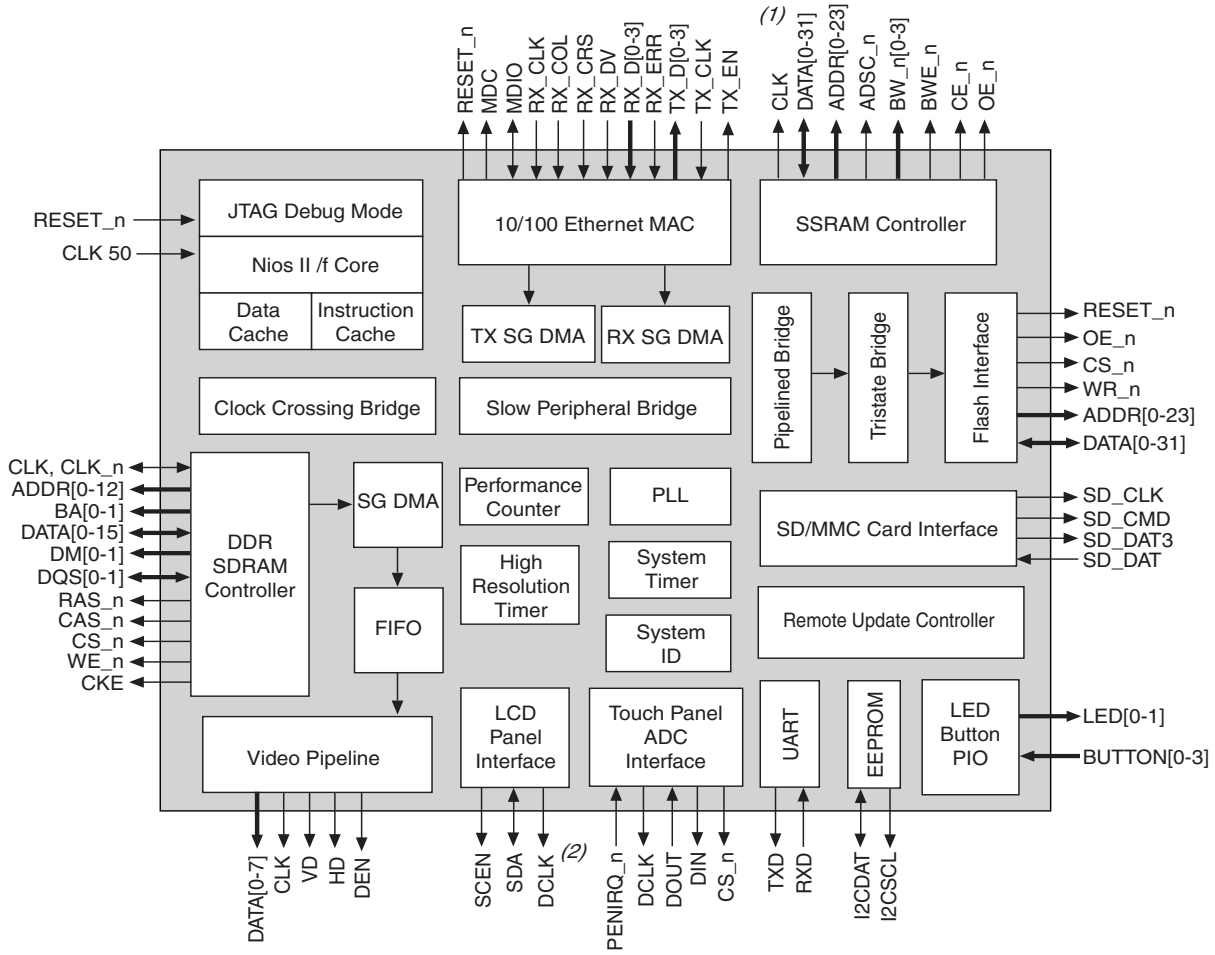
The microprocessor supports an LCD color touch panel by integrating the LCD controller hardware, which has been implemented as part of a video pipeline. The video pipeline is fully logic based, composed of IP cores you can modify to suit any resolution or aspect ratio.

A DDR SDRAM memory interface holds the video buffer. A 1-MByte SSRAM memory is available for processor program code. For code larger than 1 MByte, the DDR SDRAM memory can also hold processor program code. User-selectable SRAM memory (on-chip) holds the DMA descriptors. A CFI flash controller supports flash memory to store application code and FPGA configuration data.

The microprocessor integrates an SD/MMC controller that is SD card compliant. The system also includes a 10/100/1000 Ethernet MAC with SG-DMA channels for network access.

Figure 1 shows the functional blocks of the Nios II 3C25 microprocessor with LCD controller system.

Figure 1. Block Diagram



Notes to Figure 1

(1) The SSRAM controller shares the address and data buses with the flash interface.

(2) The LCD panel interface shares the clock signal with the touch panel ADC interface.

Table 1 lists the pin definitions and signal descriptions. In the **Location** column, letter values refer to the pin row within the FPGA and numeric values refer to the pin column. So for example, A1 denotes the pin located in row A and column 1.

For the Cyclone III 3C25 device pin list, refer to *Pin Information for the Cyclone III EP3C25 Device*.

Table 1. Signal Descriptions (Part 1 of 5)

Pin Name	Location	Signal Description	Direction	I/O Standard
System Clock and Reset Inputs				
top_clkin_50	V9	50 MHz crystal oscillator input	Input	2.5 V
top_reset_n	N2	FPGA reset input (active low)	Input	2.5 V
DDR SDRAM Memory (A2S56D40CTP-G5PP) Interface				
top_mem_addr[0]	U1	Address	Output	SSTL-2 Class I
top_mem_addr[1]	U5	Address	Output	SSTL-2 Class I
top_mem_addr[2]	U7	Address	Output	SSTL-2 Class I
top_mem_addr[3]	U8	Address	Output	SSTL-2 Class I
top_mem_addr[4]	P8	Address	Output	SSTL-2 Class I
top_mem_addr[5]	P7	Address	Output	SSTL-2 Class I
top_mem_addr[6]	P6	Address	Output	SSTL-2 Class I
top_mem_addr[7]	T14	Address	Output	SSTL-2 Class I
top_mem_addr[8]	T13	Address	Output	SSTL-2 Class I
top_mem_addr[9]	V13	Address	Output	SSTL-2 Class I
top_mem_addr[10]	U17	Address	Output	SSTL-2 Class I
top_mem_addr[11]	V17	Address	Output	SSTL-2 Class I
top_mem_addr[12]	U16	Address	Output	SSTL-2 Class I
top_mem_ba[0]	V11	Bank address	Output	SSTL-2 Class I
top_mem_ba[1]	V12	Bank address	Output	SSTL-2 Class I
top_mem_cas_n	T4	CAS command (active low)	Output	SSTL-2 Class I
top_mem_cke	R13	Clock enable	Output	SSTL-2 Class I
top_mem_clk	U2	Positive differential clock input	Bidirectional	SSTL-2 Class I
top_mem_clk_n	V2	Negative differential clock input	Bidirectional	SSTL-2 Class I
top_mem_cs_n	V1	Chip select input (active low)	Output	SSTL-2 Class I
top_mem_dm[0]	V3	Data mask	Output	SSTL-2 Class I
top_mem_dm[1]	V8	Data mask	Output	SSTL-2 Class I
top_mem_dq[0]	U4	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[1]	V4	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[2]	R8	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[3]	V5	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[4]	P9	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[5]	U6	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[6]	V6	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[7]	V7	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[8]	U13	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[9]	U12	Data I/O	Bidirectional	SSTL-2 Class I

Table 1. Signal Descriptions (Part 2 of 5)

Pin Name	Location	Signal Description	Direction	I/O Standard
top_mem_dq[10]	U11	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[11]	V15	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[12]	U14	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[13]	R11	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[14]	P10	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dq[15]	V14	Data I/O	Bidirectional	SSTL-2 Class I
top_mem_dqs[0]	U3	Data strobe	Bidirectional	SSTL-2 Class I
top_mem_dqs[1]	T8	Data strobe	Bidirectional	SSTL-2 Class I
top_mem_ras_n	V16	RAS command (active low)	Output	SSTL-2 Class I
top_mem_we_n	U15	Write enable command (active low)	Output	SSTL-2 Class I
Parallel Flash (PC28F256P30B85) and SSRAM (IS61LPS25636A-200TQL1) Memory Interface				
top_flash_cs_n	E2	Flash chip enable (active low)	Output	2.5 V
top_flash_oe_n	D17	Flash output enable (active low)	Output	2.5 V
top_flash_reset_n	C3	Flash reset (active low)	Output	2.5 V
top_flash_ssram_a[1]	E12	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[2]	A16	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[3]	B16	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[4]	A15	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[5]	B15	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[6]	A14	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[7]	B14	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[8]	A13	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[9]	B13	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[10]	A12	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[11]	B12	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[12]	A11	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[13]	B11	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[14]	C10	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[15]	D10	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[16]	E10	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[17]	C9	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[18]	D9	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[19]	A7	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[20]	A6	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[21]	B18	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[22]	C17	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_a[23]	C18	Flash and SSRAM shared address	Output	2.5 V
top_flash_ssram_d[0]	H3	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[1]	D1	Flash and SSRAM shared data	Bidirectional	2.5 V

Table 1. Signal Descriptions (Part 3 of 5)

Pin Name	Location	Signal Description	Direction	I/O Standard
top_flash_ssram_d[2]	A8	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[3]	B8	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[4]	B7	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[5]	C5	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[6]	E8	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[7]	A4	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[8]	B4	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[9]	E7	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[10]	A3	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[11]	B3	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[12]	D5	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[13]	B5	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[14]	A5	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[15]	B6	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[16]	C16	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[17]	D12	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[18]	E11	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[19]	D2	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[20]	E13	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[21]	E14	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[22]	A17	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[23]	D16	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[24]	C12	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[25]	A18	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[26]	F8	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[27]	D7	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[28]	F6	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[29]	E6	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[30]	G6	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_ssram_d[31]	C7	Flash and SSRAM shared data	Bidirectional	2.5 V
top_flash_wr_n	D18	Flash write enable (active low)	Output	2.5 V
top_ssram_adsc_n	F7	SSRAM address status controller (active low)	Output	2.5 V
top_ssram_bw_n[0]	F10	SSRAM byte write controls (active low)	Output	2.5 V
top_ssram_bw_n[1]	F11	SSRAM byte write controls (active low)	Output	2.5 V
top_ssram_bw_n[2]	F12	SSRAM byte write controls (active low)	Output	2.5 V
top_ssram_bw_n[3]	F13	SSRAM byte write controls (active low)	Output	2.5 V
top_ssram_bwe_n	G13	SSRAM byte write enable (active low)	Output	2.5 V
top_ssram_ce_n	F9	SSRAM chip select (active low)	Output	2.5 V
top_ssram_clk	A2	SSRAM clock	Output	2.5 V

Table 1. Signal Descriptions (Part 4 of 5)

Pin Name	Location	Signal Description	Direction	I/O Standard
top_ssram_oe_n	E9	SSRAM output enable (active low)	Output	2.5 V
Touch Panel Analog to Digital Converter (AD7843) Interface				
top_HC_ADC_CS_N	R5	Chip select for ADC (active low)	Output	2.5 V
top_HC_ADC_DCLK	V18	Data clock for ADC	Output	2.5 V
top_HC_ADC_DIN	U18	Data input for ADC	Output	2.5 V
top_HC_ADC_DOUT	L18	Data output for ADC	Input	2.5 V
top_HC_ADC_PENIRQ_N	N17	Pen interrupt ADC (active low)	Input	2.5 V
LCD Panel Interface				
top_HC_SCEN	M6	Serial clock enable	Output	2.5 V
top_HC_SDA	T2	Serial data enable	Bidirectional	2.5 V
Video Pipeline Interface				
top_clk_to_offchip_video	D14	Clock for LCD touch panel	Output	2.5 V
top_HC_DEN	R17	Data enable for ADC	Output	2.5 V
top_HC_HD	M14	LCD horizontal sync input	Output	2.5 V
top_HC_VD	L13	LCD vertical sync input	Output	2.5 V
top_HC_LCD_DATA[0]	R4	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[1]	T17	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[2]	T18	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[3]	L16	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[4]	M17	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[5]	N6	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[6]	M13	Multiplexed RGB data for LCD touch panel	Output	2.5 V
top_HC_LCD_DATA[7]	N13	Multiplexed RGB data for LCD touch panel	Output	2.5 V
10/100 Ethernet Interface (National PHY DP83848C)				
top_HC_ETH_RESET_N	H18	Ethernet PHY reset (active low)	Output	2.5 V
top_HC_MDC	P18	Management data clock	Output	2.5 V
top_HC_MDIO	N7	Management data I/O	Bidirectional	2.5 V
top_HC_RX_CLK	F17	MII receive clock	Input	2.5 V
top_HC_RX_COL	G17	MII collision detect	Input	2.5 V
top_HC_RX_CRS	L3	MII carrier sense/receive	Input	2.5 V
top_HC_RX_D[0]	P2	Receive data	Input	2.5 V
top_HC_RX_D[1]	P1	Receive data	Input	2.5 V

Table 1. Signal Descriptions (Part 5 of 5)

Pin Name	Location	Signal Description	Direction	I/O Standard
top_HC_RX_D[2]	T3	Receive data	Input	2.5 V
top_HC_RX_D[3]	R3	Receive data	Input	2.5 V
top_HC_RX_DV	G18	Receive data valid	Input	2.5 V
top_HC_RX_ERR	L4	Receive error	Input	2.5 V
top_HC_TX_CLK	N18	MII transmit clock	Input	2.5 V
top_HC_TX_D[0]	M18	Transmit data	Output	2.5 V
top_HC_TX_D[1]	L14	Transmit data	Output	2.5 V
top_HC_TX_D[2]	L15	Transmit data	Output	2.5 V
top_HC_TX_D[3]	P17	Transmit data	Output	2.5 V
top_HC_TX_EN	L17	Transmit enable	Output	2.5 V
SD/MMC Card Interface				
top_HC_SD_CLK	M2	SD clock	Output	2.5 V
top_HC_SD_CMD	L6	SD command	Output	2.5 V
top_HC_SD_DAT	M3	SD data	Input	2.5 V
top_HC_SD_DAT3	N8	SD data3	Output	2.5 V
EEPROM (24LC00) Interface				
top_HC_ID_I2CDAT	D3	Serial data	Bidirectional	2.5 V
top_HC_ID_I2CSCL	H6	Serial clock	Output	2.5 V
RS-232 UART Transceiver ADM3202 Interface				
top_HC_UART_RXD	E18	UART receive data	Input	2.5 V
top_HC_UART_TXD	H17	UART transmit data	Output	2.5 V
LED Interface				
top_led[0]	P13	LED output	Output	2.5 V
top_led[1]	T1	LED output	Output	2.5 V
Push Button Interface				
top_button[0]	F1	Parallel I/O for push button	Input	2.5 V
top_button[1]	F2	Parallel I/O for push button	Input	2.5 V
top_button[2]	A10	Parallel I/O for push button	Input	2.5 V
top_button[3]	B10	Parallel I/O for push button	Input	2.5 V

Table 2 lists the device pin-outs.

Table 2. Device Pin-Outs (Part 1 of 6)

Pin Name	Location	Direction	I/O Standard	I/O Bank
top_ssram_clk	A2	Output	2.5 V	8
top_flash_ssram_d[10]	A3	Bidirectional	2.5 V	8
top_flash_ssram_d[7]	A4	Bidirectional	2.5 V	8
top_flash_ssram_d[14]	A5	Bidirectional	2.5 V	8
top_flash_ssram_a[20]	A6	Output	2.5 V	8

Table 2. Device Pin-Outs (Part 2 of 6)

Pin Name	Location	Direction	I/O Standard	I/O Bank
top_flash_ssram_a[19]	A7	Output	2.5 V	8
top_flash_ssram_d[2]	A8	Bidirectional	2.5 V	8
top_button[2]	A10	Input	2.5 V	7
top_flash_ssram_a[12]	A11	Output	2.5 V	7
top_flash_ssram_a[10]	A12	Output	2.5 V	7
top_flash_ssram_a[8]	A13	Output	2.5 V	7
top_flash_ssram_a[6]	A14	Output	2.5 V	7
top_flash_ssram_a[4]	A15	Output	2.5 V	7
top_flash_ssram_a[2]	A16	Output	2.5 V	7
top_flash_ssram_d[22]	A17	Bidirectional	2.5 V	7
top_flash_ssram_d[25]	A18	Bidirectional	2.5 V	7
top_flash_ssram_d[11]	B3	Bidirectional	2.5 V	8
top_flash_ssram_d[8]	B4	Bidirectional	2.5 V	8
top_flash_ssram_d[13]	B5	Bidirectional	2.5 V	8
top_flash_ssram_d[15]	B6	Bidirectional	2.5 V	8
top_flash_ssram_d[4]	B7	Bidirectional	2.5 V	8
top_flash_ssram_d[3]	B8	Bidirectional	2.5 V	8
top_button[3]	B10	Input	2.5 V	7
top_flash_ssram_a[13]	B11	Output	2.5 V	7
top_flash_ssram_a[11]	B12	Output	2.5 V	7
top_flash_ssram_a[9]	B13	Output	2.5 V	7
top_flash_ssram_a[7]	B14	Output	2.5 V	7
top_flash_ssram_a[5]	B15	Output	2.5 V	7
top_flash_ssram_a[3]	B16	Output	2.5 V	7
top_flash_ssram_a[21]	B18	Output	2.5 V	6
top_flash_reset_n	C3	Output	2.5 V	1
top_flash_ssram_d[5]	C5	Bidirectional	2.5 V	8
top_flash_ssram_d[31]	C7	Bidirectional	2.5 V	8
top_flash_ssram_a[17]	C9	Output	2.5 V	8
top_flash_ssram_a[14]	C10	Output	2.5 V	7
top_flash_ssram_d[24]	C12	Bidirectional	2.5 V	7
top_flash_ssram_d[16]	C16	Bidirectional	2.5 V	7
top_flash_ssram_a[22]	C17	Output	2.5 V	6
top_flash_ssram_a[23]	C18	Output	2.5 V	6
top_flash_ssram_d[1]	D1	Bidirectional	2.5 V	1
top_flash_ssram_d[19]	D2	Bidirectional	2.5 V	1
top_HC_ID_I2CDAT	D3	Bidirectional	2.5 V	1
top_flash_ssram_d[12]	D5	Bidirectional	2.5 V	8
top_flash_ssram_d[27]	D7	Bidirectional	2.5 V	8

Table 2. Device Pin-Outs (Part 3 of 6)

Pin Name	Location	Direction	I/O Standard	I/O Bank
top_flash_ssram_a[18]	D9	Output	2.5 V	8
top_flash_ssram_a[15]	D10	Output	2.5 V	7
top_flash_ssram_d[17]	D12	Bidirectional	2.5 V	7
top_clk_to_offchip_video	D14	Output	2.5 V	7
top_flash_ssram_d[23]	D16	Bidirectional	2.5 V	7
top_flash_oe_n	D17	Output	2.5 V	6
top_flash_wr_n	D18	Output	2.5 V	6
top_flash_cs_n	E2	Output	2.5 V	1
top_flash_ssram_d[29]	E6	Bidirectional	2.5 V	8
top_flash_ssram_d[9]	E7	Bidirectional	2.5 V	8
top_flash_ssram_d[6]	E8	Bidirectional	2.5 V	8
top_ssram_oe_n	E9	Output	2.5 V	8
top_flash_ssram_a[16]	E10	Output	2.5 V	8
top_flash_ssram_d[18]	E11	Bidirectional	2.5 V	7
top_flash_ssram_a[1]	E12	Output	2.5 V	7
top_flash_ssram_d[20]	E13	Bidirectional	2.5 V	7
top_flash_ssram_d[21]	E14	Bidirectional	2.5 V	7
top_HC_UART_RXD	E18	Input	2.5 V	6
top_button[0]	F1	Input	2.5 V	1
top_button[1]	F2	Input	2.5 V	1
top_flash_ssram_d[28]	F6	Bidirectional	2.5 V	8
top_ssram_adsc_n	F7	Output	2.5 V	8
top_flash_ssram_d[26]	F8	Bidirectional	2.5 V	8
top_ssram_ce_n	F9	Output	2.5 V	8
top_ssram_bw_n[0]	F10	Output	2.5 V	7
top_ssram_bw_n[1]	F11	Output	2.5 V	7
top_ssram_bw_n[2]	F12	Output	2.5 V	7
top_ssram_bw_n[3]	F13	Output	2.5 V	7
top_HC_RX_CLK	F17	Input	2.5 V	6
top_flash_ssram_d[30]	G6	Bidirectional	2.5 V	8
top_ssram_bwe_n	G13	Output	2.5 V	7
top_HC_RX_COL	G17	Input	2.5 V	6
top_HC_RX_DV	G18	Input	2.5 V	6
top_flash_ssram_d[0]	H3	Bidirectional	2.5 V	1
top_HC_ID_I2CSCL	H6	Output	2.5 V	1
top_HC_UART_TXD	H17	Output	2.5 V	6
top_HC_ETH_RESET_N	H18	Output	2.5 V	6
top_HC_RX_CRS	L3	Input	2.5 V	2
top_HC_RX_ERR	L4	Input	2.5 V	2

Table 2. Device Pin-Outs (Part 4 of 6)

Pin Name	Location	Direction	I/O Standard	I/O Bank
top_HC_SD_CMD	L6	Output	2.5 V	2
top_HC_VD	L13	Output	2.5 V	5
top_HC_TX_D[1]	L14	Output	2.5 V	5
top_HC_TX_D[2]	L15	Output	2.5 V	5
top_HC_LCD_DATA[3]	L16	Output	2.5 V	5
top_HC_TX_EN	L17	Output	2.5 V	5
top_HC_ADC_DOUT	L18	Input	2.5 V	5
top_HC_SD_CLK	M2	Output	2.5 V	2
top_HC_SD_DAT	M3	Input	2.5 V	2
top_HC_SCEN	M6	Output	2.5 V	3
top_HC_LCD_DATA[6]	M13	Output	2.5 V	4
top_HC_HD	M14	Output	2.5 V	5
top_HC_LCD_DATA[4]	M17	Output	2.5 V	5
top_HC_TX_D[0]	M18	Output	2.5 V	5
top_reset_n	N2	Input	2.5 V	2
top_HC_LCD_DATA[5]	N6	Output	2.5 V	3
top_HC_MDIO	N7	Bidirectional	2.5 V	3
top_HC_SD_DAT3	N8	Output	2.5 V	3
top_HC_LCD_DATA[7]	N13	Output	2.5 V	4
top_HC_ADC_PENIRQ_N	N17	Input	2.5 V	5
top_HC_TX_CLK	N18	Input	2.5 V	5
top_HC_RX_D[1]	P1	Input	2.5 V	2
top_HC_RX_D[0]	P2	Input	2.5 V	2
top_mem_addr[6]	P6	Output	SSTL-2 Class I	3
top_mem_addr[5]	P7	Output	SSTL-2 Class I	3
top_mem_addr[4]	P8	Output	SSTL-2 Class I	3
top_mem_dq[4]	P9	Bidirectional	SSTL-2 Class I	3
top_mem_dq[14]	P10	Bidirectional	SSTL-2 Class I	4
top_led[0]	P13	Output	2.5 V	4
top_HC_TX_D[3]	P17	Output	2.5 V	5
top_HC_MDC	P18	Output	2.5 V	5
top_HC_RX_D[3]	R3	Input	2.5 V	2
top_HC_LCD_DATA[0]	R4	Output	2.5 V	2
top_HC_ADC_CS_N	R5	Output	2.5 V	2
top_mem_dq[2]	R8	Bidirectional	SSTL-2 Class I	3
top_mem_dq[13]	R11	Bidirectional	SSTL-2 Class I	4
top_mem_cke	R13	Output	SSTL-2 Class I	4
top_HC_DEN	R17	Output	2.5 V	5
top_led[1]	T1	Output	2.5 V	2

Table 2. Device Pin-Outs (Part 5 of 6)

Pin Name	Location	Direction	I/O Standard	I/O Bank
top_HC_SDA	T2	Bidirectional	2.5 V	2
top_HC_RX_D[2]	T3	Input	2.5 V	2
top_mem_cas_n	T4	Output	SSTL-2 Class I	3
top_mem_dqs[1]	T8	Bidirectional	SSTL-2 Class I	3
top_mem_addr[8]	T13	Output	SSTL-2 Class I	4
top_mem_addr[7]	T14	Output	SSTL-2 Class I	4
top_HC_LCD_DATA[1]	T17	Output	2.5 V	5
top_HC_LCD_DATA[2]	T18	Output	2.5 V	5
top_mem_addr[0]	U1	Output	SSTL-2 Class I	3
top_mem_clk	U2	Bidirectional	SSTL-2 Class I	3
top_mem_dqs[0]	U3	Bidirectional	SSTL-2 Class I	3
top_mem_dq[0]	U4	Bidirectional	SSTL-2 Class I	3
top_mem_addr[1]	U5	Output	SSTL-2 Class I	3
top_mem_dq[5]	U6	Bidirectional	SSTL-2 Class I	3
top_mem_addr[2]	U7	Output	SSTL-2 Class I	3
top_mem_addr[3]	U8	Output	SSTL-2 Class I	3
top_mem_dq[10]	U11	Bidirectional	SSTL-2 Class I	4
top_mem_dq[9]	U12	Bidirectional	SSTL-2 Class I	4
top_mem_dq[8]	U13	Bidirectional	SSTL-2 Class I	4
top_mem_dq[12]	U14	Bidirectional	SSTL-2 Class I	4
top_mem_we_n	U15	Output	SSTL-2 Class I	4
top_mem_addr[12]	U16	Output	SSTL-2 Class I	4
top_mem_addr[10]	U17	Output	SSTL-2 Class I	4
top_HC_ADC_DIN	U18	Output	2.5 V	4
top_mem_cs_n	V1	Output	SSTL-2 Class I	3
top_mem_clk_n	V2	Bidirectional	SSTL-2 Class I	3
top_mem_dm[0]	V3	Output	SSTL-2 Class I	3
top_mem_dq[1]	V4	Bidirectional	SSTL-2 Class I	3
top_mem_dq[3]	V5	Bidirectional	SSTL-2 Class I	3
top_mem_dq[6]	V6	Bidirectional	SSTL-2 Class I	3
top_mem_dq[7]	V7	Bidirectional	SSTL-2 Class I	3
top_mem_dm[1]	V8	Output	SSTL-2 Class I	3
top_clkin_50	V9	Input	2.5 V	3
top_mem_ba[0]	V11	Output	SSTL-2 Class I	4
top_mem_ba[1]	V12	Output	SSTL-2 Class I	4
top_mem_addr[9]	V13	Output	SSTL-2 Class I	4
top_mem_dq[15]	V14	Bidirectional	SSTL-2 Class I	4
top_mem_dq[11]	V15	Bidirectional	SSTL-2 Class I	4
top_mem_ras_n	V16	Output	SSTL-2 Class I	4

Table 2. Device Pin-Outs (Part 6 of 6)

Pin Name	Location	Direction	I/O Standard	I/O Bank
top_mem_addr[11]	V17	Output	SSTL-2 Class I	4
top_HC_ADC_DCLK	V18	Output	2.5 V	4

Device Specifications

The microprocessor consists of a soft core processor and other components within an FPGA device.

Refer to the *Cyclone III Device Handbook* for FPGA device specific information, such as device core architecture, I/O considerations, power considerations, external memory (including interfaces, configuration, hot socketing, and remote system update), and packaging information.

For specific information about electrical DC and switching characteristics, refer to the *DC and Switching Characteristics* chapter in volume 2 of the *Cyclone III Device Handbook*.

Address Map

Table 3 defines the address mapping for the reference design from the viewpoint of the instruction and data masters of the Nios II processor. Each peripheral might have several ports. For instance, `pll.s1` designates a slave port for the phase locked loop (PLL) component.

Table 3. Address Map (Part 1 of 2)

Port Name	cpu.instruction_master	cpu.data_master
pll.s1	—	0x08002700 – 0x080027ff
cpu.jtag_debug_module	0x06000800 – 0x06000fff	0x06000800 – 0x06000fff
descriptor_memory.s1	—	0x08000000 – 0x08000fff
flash_ssram_pipeline_bridge.s1	0x04000000 – 0x05ffffff	0x04000000 – 0x05ffffff
pipeline_bridge_before_tristate_bridge.s1	0x04000000 – 0x05ffffff	0x04000000 – 0x05ffffff
descriptor_offset_bridge.s1	—	—
flash_ssram_tristate_bridge.avalon_slave	—	—
ssram.s1	0x05000000 – 0x050fffff	0x05000000 – 0x050fffff
ext_flash.s1	0x04000000 – 0x04ffffff	0x04000000 – 0x04ffffff
ddr_sdram.s1	0x00000000 – 0x01ffffff	0x00000000 – 0x01ffffff
cpu_ddr_clock_bridge.s1	0x00000000 – 0x03ffffff	0x00000000 – 0x03ffffff
slow_peripheral_bridge.s1	—	0x08000000 – 0x08003fff
tse_ddr_clock_bridge.s1	—	—
tse_ssram_clock_bridge.s1	—	—
sgdma_tx.csr	—	0x08001c00 – 0x08001fff
sgdma_rx.csr	—	0x08001800 – 0x08001bff
lcd_sgdma.csr	0x02000000 – 0x020003ff	0x02000000 – 0x020003ff

Table 3. Address Map (Part 2 of 2)

Port Name	cpu.instruction_master	cpu.data_master
tse_mac.control_port	—	0x08002000 – 0x080023ff
sys_clk_timer.s1	—	0x08002800 – 0x080028ff
high_res_timer.s1	—	0x08002500 – 0x080025ff
sysid.control_slave	—	0x08002f40 – 0x08002f7f
performance_counter.control_slave	—	0x08002600 – 0x080026ff
jtag_uart.avalon_jtag_slave	—	0x08002f00 – 0x08002f3f
uart1.s1	—	0x08002a00 – 0x08002aff
button_pio.s1	—	0x08002b00 – 0x08002b7f
led_pio.s1	—	0x08002d00 – 0x08002d7f
pio_id_eeprom_scl.s1	—	0x08002e00 – 0x08002e7f
lcd_i2c_scl.s1	—	0x08002c00 – 0x08002c7f
lcd_i2c_en.s1	—	0x08002b80 – 0x08002bff
pio_id_eeprom_dat.s1	—	0x08002d80 – 0x08002dff
lcd_i2c_sdat.s1	—	0x08002c80 – 0x08002cff
touch_panel_pen_irq_n.s1	—	0x08002e80 – 0x08002eff
touch_panel_spi.spi_control_port	—	0x08002900 – 0x080029ff
el_camino_sd_card_controller.avalon_slave	—	0x08002400 – 0x080024ff
remote_update.s1	—	0x08001000 – 0x080017ff

System Architecture

The microprocessor design is based on the Nios II/f core and provides a typical mix of peripherals, memories, and a video pipeline. The design provides an interface to each hardware component on the Altera Nios II Embedded Evaluation Kit, Cyclone III Edition, such as DDR SDRAM, LEDs, RS-232 connector, Ethernet MAC/10/100 PHY, and 800 × 480 pixel LCD. The video pipeline provides high bandwidth memory access that allows for flicker-free display on the color LCD.

Design Considerations

Use of Clock Domains

To increase the overall f_{MAX} performance, exercise prudent use of independent clock domains where slow peripheral components run in a slower clock region than the Nios II processor and other memory components. Your attentiveness can increase the overall f_{MAX} of the design without adding additional memory latency.

The Nios II processor runs at a frequency of 100 MHz and is connected to high performance DDR SDRAM memory, on-chip descriptor memory, and CFI flash memory. Clock crossing bridges are required between the Nios II processor and the DDR SDRAM memory and slow peripherals components because these components run in different clock regions. A pipeline bridge between the Nios II processor and the flash tri-state bridge to external flash component ensures system f_{MAX} is not affected and that every master sees every slave at the same address.

The DDR SDRAM memory runs at 133 MHz. The ddr_sdram memory controller runs at half rate at local interface with a 64-bit data width, connected to a 32-bit width Nios II data bus and a 64-bit width SG-DMA through clock crossing bridges.

Summary of Clock Domains

The system assumes an oscillator clock of 50 MHz and uses a PLL to generate the rest of the clocks in the system.

- pll is driven by 50-MHz oscillator clock using four generated clocks:
  - pll_c0, 100 MHz: used as system clock for Nios II processor, video clock for LCD touch panel circuitry, memory clock for CFI flash, and clock for SSRAM controller
  - pll_c1, 100 MHz: used as memory clock for the external SSRAM memory
  - pll_c2, 60 MHz: used as slow peripherals clock
  - pll_c3, 40 MHz: used as remote update controller clock
- ddr_sdram is driven by 50-MHz oscillator clock, and runs at 133 MHz
  - runs at half-rate 66.5 MHz at local interface
- 100-MHz clock region:
  - Nios II processor
  - CFI flash
  - SSRAM
  - Video pipeline components
- 60-MHz clock region:
  - System peripherals
  - SD MMC components
  - JTAG UART
  - UART
  - PIOs
- 66.5-MHz clock region (must run at the same speed as ddr_sdram):
  - lcd_sgdma SG-DMA, 64-bit data width
  - lcd_ta_sgdma_to_fifo timing adapter
- 40-MHz clock region:
  - Remote update controller

Use of Bridges

Bridges control the topology of an SOPC Builder system. Without bridges, SOPC Builder generates a system interconnect fabric with maximum parallelism, where all masters drive slaves concurrently, as long as each master accesses different slaves. For systems that do not require such a high degree of concurrency, you can use a bridge to control topology and provide optimal performance.

For more information, refer to the *Avalon Memory-Mapped Bridges* chapter in volume 4 of the *Quartus II Handbook*.

Bridges are used for the following reasons:

- Clock crossing between two components clocked at different frequencies:
  - `cpu_ddr_clock_bridge`: between Nios II processor and `ddr_sdram`
  - `tse_ddr_clock_bridge`: between the `sgdma_rx`, `sgdma_tx` and `ddr_sdram`
  - `tse_ssram_clock_bridge`: between the `sgdma_rx`, `sgdma_tx` and `ssram`
  - `slow_peripheral_bridge`: between Nios II processor and slow peripheral components
- Increase the performance and f_{MAX} of the path between two components:
  - `pipeline_bridge_before_tristate_bridge`: between Nios II processor and `flash_tristate_bridge`
  - `flash_ssram_pipeline_bridge`: between Nios II processor and `pipeline_bridge_before_tristate_bridge`

The triple speed ethernet components run in the 60-MHz clock region:

- `tse_mac`
- `sgdma_tx`
- `sgdma_rx`
- `descriptor_memory`

The following components are partitioned by the `slow_peripheral_bridge`. This allows non-f_{MAX} critical components to be clocked at a slower clock rate to give better fitting for components that are f_{MAX}-critical. Unless otherwise noted, all the components run in the 60-MHz clock region.

- `sys_clk_timer`
- `high_res_timer`
- `performance_counter`
- `jtag_uart`
- `uart`
- `sysid`
- `pll` (50-MHz external clock)
- `button_pio`
- `led_pio`
- `pio_id_eeprom_dat`
- `pio_id_eeprom_scl`
- `touch_panel_spi`
- `touch_panel_pen_irq_n`

- lcd_i2c_scl
- lcd_i2c_en
- lcd_i2c_sdat
- el_camino_sd_card_controller
- remote_update_controller (40 MHz)

Arbitration Priority Considerations

Arbitration priority allows the Avalon® Memory-Mapped (Avalon-MM) interface masters to obtain the required bandwidth for transfers in the SOPC Builder system. For the Nios II processor, the optimum value of arbitration priority is eight in this system, and uses 32-byte instruction and data cache line sizes. The scatter-gather direct memory access (SG-DMA) requires an arbitration priority value of eight for descriptor read and write operations. To ensure video pipelining components operate smoothly, highest arbitration priority is given to lcd_sgdma to access the ddr_sdram memory component.

Interrupt Priority Considerations

The Nios II system processes interrupts from components with the lowest IRQ value first. In this system, priority is given to the video pipeline component, then the triple speed ethernet component, timers, JTAG UART, and lastly the slow peripheral components.

Processor

Processor Core

Processor name: cpu

Version: 8.1

Processor type: Fast (Nios /f)

- 32-bit RISC
- Instruction cache
- Data cache
- Branch prediction
- Hardware multiply
- Hardware divide
- Barrel shifter
- Dynamic branch prediction

Nominal metrics:

- Nominal performance at 100 MHz: Up to 113 DMIPS
- Nominal logic usage: 1400-1800 LEs
- Nominal memory usage: Three M9K + Cache

Reset vector:

- Memory: ext_flash
- Offset: 0x0
- Physical address: 0x04000000

Exception vector:

- Memory: ssram
- Offset: 0x20
- Physical address: 0x05000020

Memory Management Unit

MMU option: Disabled

MPU option: Disabled

Cache

Instruction master:

- Cache size: 4 KBytes

Data master:

- Cache size: 2 KBytes
- Data cache line size: 32 Bytes

JTAG Debug Module (Level 1)

- JTAG target connection
- Download software
- Software breakpoints
- Nominal logic usage: 300–400 LEs
- Nominal memory usage: Two M9Ks

Custom Instructions

Custom Instructions Enabled: None

Processor Instruction Set Reference

For details about the Nios II processor core, the processor architecture, exception and interrupt control, JTAG debug module, programming model, and instruction set architecture, refer to the *Nios II Processor Reference Handbook*.

System Clock Settings

The system expects one external clock sources named `clk` (50 MHz). A PLL creates the various clocks required for the clock domains and external memories from the 50-MHz clock source. Table 4 shows the clock sources for the system.

Table 4. Clock Sources

Name	Source	MHz
clk	external	50.0
pll_c0	pll.c0	100.0
pll_c1	pll.c1	100.0
pll_c2	pll.c2	60.0
pll_c3	pll.c3	40.0
ddr_sdram_sysclk	ddr_sdram.sysclk	66.5
ddr_sdram_auxfull	ddr_sdram.auxfull	133
ddr_sdram_auxhalf	ddr_sdram.auxhalf	66.5

For more information about the PLL megafunction, refer to the *altpll Megafunction User Guide*.

System Boot and Configuration

Cyclone III Configuration Interface and Remote System Update

The Nios II 3C25 microprocessor with LCD controller is configured for active parallel configuration on power up. Upon power up, the dedicated configuration circuitry in Cyclone III loads the configuration data stored in the parallel flash device at default offset 0x20000.

For more information about Cyclone III active parallel configuration, refer to the *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*.

The microprocessor system comes with the Cyclone III remote system update feature. The system can reconfigure devices and update the configuration data from a remote location via communication protocols. The remote update controller within the system provides an interface for users to communicate with the dedicated remote system update circuitry of the Cyclone III.

For more information on the Cyclone III remote system update feature, refer to the *Remote System Upgrade With Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook* and the *Cyclone III Remote Update Controller Core* chapter in volume 5 of the *Quartus II Handbook*.

System ID

The Altera System ID core is a simple read-only device that provides SOPC Builder systems with a unique identifier. Nios II processor systems use the system ID core to verify that an executable program was compiled targeting the actual hardware image configured in the FPGA.

For more information about the system ID core, refer to the *System ID Core* chapter in volume 5 of the *Quartus II Handbook*.

Timers

High Resolution Timer

Use a high resolution Altera interval timer core as an interval timekeeper, watch-dog timer, or counter. Table 5 shows the high resolution timer parameters and corresponding parameter values.

Table 5. High Resolution Timer Parameters

Parameter	Value
Name	high_res_timer
Time-out period	10 ms
Timer counter size	32 bits
Hardware presets	Full featured

For more information about the high resolution interval timer core, refer to the *Timer Core* chapter in volume 5 of the *Quartus II Handbook*.

Performance Counter

The Altera performance counter core measures software performance by tracking the time to execute one or more functions. A performance counter is useful for software engineers during software development. Table 6 shows the performance counter parameters and corresponding parameter values.

Table 6. Performance Counter Parameters

Parameter	Value
Name	performance_counter
Simultaneously-measured sections	1
Counter width	32 bits

For more information about this core, refer to the *Performance Counter Core* chapter in volume 5 of the *Quartus II Handbook*.

PIOs

The Altera programmed input/output (PIO) cores transfer data between the processor and certain input/output (I/O) devices. The microprocessor includes LED and Button PIOs. Table 7 shows the LED PIO parameters and corresponding parameter values.

Table 7. LED PIO Parameters

Parameter	Value
Name	led_pio
Width (1-32 bits)	2
Direction	Output ports only
Output port reset value	0x0

Table 8 shows the Button PIO parameters and corresponding parameter values.

Table 8. Button PIO Parameters

Parameter	Value
Name	`button_pio`
Width (1-32 bits)	4
Direction	Input ports only
Input options	■ Edge capture register: synchronous capture, rising edge ■ Interrupt generated: edge based

For information about these cores, refer to the *PIO Core* chapter in volume 5 of the *Quartus II Handbook*.

EEPROM ID Interface

The EEPROM ID interface is a 2-bit interface to an I2C serial EEPROM constructed using the PIOs, `pio_id_eeprom_dat` and `pio_id_eeprom_scl`. Table 9 shows the EEPROM I2C data PIO parameters and corresponding parameter values.

Table 9. EEPROM I2C Data PIO Parameters

Parameter	Value
Name	pio_id_eeprom_dat
Width (1-32 bits)	1
Direction	Bidirectional (tri-state) ports
Output port reset value	0x0

Table 10 shows the EEPROM I2C clock PIO parameters and corresponding parameter values.

Table 10. EEPROM I2C Clock PIO Parameters

Parameter	Value
Name	pio_id_eeprom_scl
Width (1-32 bits)	1
Direction	Output ports only
Output port reset value	0x0

For information about these cores, refer to the *PIO Core* chapter in volume 5 of the *Quartus II Handbook*.

Memory Interface

CFI Flash Interface

The Altera CFI-compliant flash memory controller core controls an external flash device (Intel). This flash device stores both application program code and FPGA configuration data. With 16 MByte capacity, it is possible to store multiple configuration images in flash memory and configure the FPGA with one of the images. Table 11 shows the CFI flash interface parameters and corresponding parameter values.

Table 11. CFI Flash Interface Parameters

Parameter	Value
Name	`ext_flash`
Address width (bits)	23
Data width	16
Flash capacity	16 MByte
Timing settings	■ Setup: 25 ns ■ Wait: 100 ns ■ Hold: 20 ns

For more information about this core, refer to the *Common Flash Interface Controller Core* chapter in volume 5 of the *Quartus II Handbook*.

DDR SDRAM Memory Controller

The Altera DDR SDRAM High Performance MegaCore function is used to interface to a PowerChip Semiconductor A2S56D40CTP-G5PP DDR SDRAM device, creating a video frame buffer, and Ethernet data transmit and receive buffer. The video data (RGB) is stored in the video frame buffer in unpacked 64-bit format. An SG-DMA is used to transfer the 64-bit-wide video stream from the memory into the video pipeline. The MegaCore function is configured to a 16-bit width clocked at 133 MHz. The local interface to the SG-DMA is configured to a 64-bit width clocked at 66.5 MHz and hence the memory controller is configured for half-rate operation. Table 12 shows the MegaCore function parameters and corresponding parameter values.

Table 12. DDR SDRAM Memory Controller Parameters

Parameter	Value
Name	`ddr_sdram`
PLL reference clock frequency	50 MHz
Memory clock frequency	133 MHz
Local interface clock frequency	66.5 MHz
Local interface width	64 bits

For more information about this MegaCore function, refer to the *DDR and DDR2 SDRAM High-Performance Controller User Guide*.

SD/MMC SPI

The SD/MMC SPI core connects to standard multimedia card (MMC) and secure digital (SD) flash based memory devices. The MMC and SD card are universal low cost data storage memories. The SD/MMC SPI core is available from El Camino GmbH in encrypted format, for evaluation purposes. The SD/MMC SPI core also comes with low-level driver routines to access the MMC and SD devices.

Features:

- 2,400 KBytes per second read and 2,400 KBytes per second write performance
- Supports MMC and SD in SPI mode
- Variable data rate up to 25 Mbps (SD only) and 20 Mbps (SD/MMC)
- Hardware assisted CRC calculation
- Low level drivers included

Table 13 shows the SD/MMC SPI parameters and corresponding parameter values.

Table 13. SD/MMC SPI Parameters

Parameter	Value
Name	`el_camino_sd_card_controller`
Avalon bus clock frequency	60 MHz
SPI clock frequency	20 MHz

For detailed information about the El Camino SD Card MMC SPI core, refer to *SD/MMC SPI Core with Avalon Interface* available on the El Camino GmbH website (www.elcamino.de).

Purchase this core directly from El Camino GmbH. If you open the microprocessor project in SOPC Builder before purchasing the core, an error message might appear.

If your design does not require an SD card controller, you can turn off the core in the **Use** column in SOPC Builder and regenerate the system.

Communications Interface

Ethernet Interface

Because Ethernet has become a common communication interface for embedded systems, this data sheet provides a pre-generated Ethernet solution featuring the Altera Triple Speed Ethernet MegaCore function and NicheStack TCP/IP Network Stack, Nios II Edition.

Triple Speed Ethernet MegaCore Function

Table 14 shows the Triple Speed Ethernet MegaCore function parameters and corresponding parameter values.

Table 14. Triple Speed Ethernet MegaCore Function Parameters

Parameter	Value
Name	tse_mac
Core Configuration	
Core variation	10/100/1000 Mbit Ethernet MAC
Interface	MII/GMII
Use internal FIFO	Enabled
MAC Options	
Enable MAC 10/100 half duplex support	
Enable MII/GMII/RGMII loopback logic	
Enable supplemental MAC unicast addresses	
Implement statistics counters	
Implement multicast hash table	
Enable magic packet detection	
MDIO Module	
Include MDIO module	
Host clock divisor	40
FIFO Options	
Memory block	Auto
Memory width	32 bits
Transmit and receive FIFO depth	512 × 8 bits

For more details about the Triple Speed Ethernet MegaCore function, refer to the following documents:

- *AN483: Triple Speed Ethernet Data Path Reference Design*
- *Triple Speed Ethernet MegaCore Function User Guide*

Transmit SG-DMA

As part of the Ethernet solution, the Altera transmit SG-DMA controller core facilitates high-speed Ethernet data transfer between the DDR SDRAM High Performance MegaCore function and the transmit FIFO in the Triple Speed Ethernet MegaCore function. The processor software creates a set of descriptors to specify the data to transmit and then the SG-DMA performs the series of DMA transfers. Table 15 shows the transmit SG-DMA parameters and corresponding parameter values.

Table 15. Transmit SG-DMA Parameters

Parameter	Value
Name	sgdma_tx
Transfer mode	Memory to stream
Data width	32
Source error width	1
FIFO depth	2

For more information about this core, refer to the *Scatter-Gather DMA Controller Core* chapter in volume 5 of the *Quartus II Handbook*.

Receive SG-DMA

The Altera receive SG-DMA controller core facilitates high speed Ethernet data transfer between the Triple Speed Ethernet MegaCore function receive FIFO and DDR SDRAM High Performance MegaCore function. The processor software creates a set of descriptors to specify where to receive the data from and then the SG-DMA performs the series of DMA reads specified by the descriptors. Table 16 shows the receive SG-DMA parameters and corresponding parameter values.

Table 16. Receive SG-DMA Parameters

Parameter	Value
Name	`sgdma_rx`
Transfer mode	Stream to memory
Data width	32
Sink error width	6
FIFO depth	2

For more information about this core, refer to the *Scatter-Gather DMA Controller Core* chapter in volume 5 of the *Quartus II Handbook*.

NicheStack TCP/IP Network Stack, Nios II Edition

The NicheStack TCP/IP Network Stack, Nios II Edition is a small code footprint implementation of the transmission control protocol / internet protocol (TCP/IP) suite. Altera provides the NicheStack as a software core that you can add to your system library or board support package.

For more details about the network stack, refer to the *Ethernet and the NicheStack TCP/IP Stack - Nios II Edition* chapter in the *Nios II Software Developer's Handbook*.

RS-232 UART Serial Port

The Altera UART core is used for serial communication via the RS-232 protocol. Table 17 shows the RS-232 UART serial port parameters and corresponding parameter values.

Table 17. RS-232 UART Serial Port Parameters

Parameter	Value
Name	`uart1`
Baud rate	115,200 bps
Parity	None
Data bits	8
Stop bits	1

For more information about this core, refer to the *UART Core* chapter in volume 5 of the *Quartus II Handbook*.

JTAG UART

The Altera JTAG UART core provides serial character stream communication between a PC host and the SOPC Builder system. The JTAG UART core uses the JTAG circuitry built into Altera FPGAs and provides host access via the JTAG pins on the FPGA. The host PC can connect to the FPGA via any Altera JTAG download cable, such as the USB-Blaster cable. You can use the JTAG link to download and run the Nios II software executable file, debug the software program using the Nios II integrated development environment (IDE) debugger, or debug the hardware system using the SignalTap® II Logic Analyzer. Table 18 shows the JTAG UART parameters and corresponding parameter values.

Table 18. JTAG UART Parameters

Parameter	Value
Name	`jtag_uart`
Write FIFO (data from Avalon to JTAG)	
Buffer depth	8 bytes
IRQ threshold	4
Read FIFO (data from JTAG to Avalon)	
Buffer depth	8 bytes
IRQ threshold	4

For more information about this core, refer to the *JTAG UART Core* chapter in volume 5 of the *Quartus II Handbook*.

Video Subsystem

The video subsystem consists of the following parts:

- LCD controller interface—interfaces with the LCD controller IC to configure the LCD panel for brightness, resolution, gamma curves, etc.
- LCD touch panel interface—interfaces with the touch panel controller IC
- Video pipeline—feeds a multiplexed RGB video stream to the LCD controller

LCD Controller Interface

The LCD controller interface is a 3-bit interface constructed with three Altera PIO cores. Table 19 shows the LCD I2C clock PIO parameters and corresponding parameter values.

Table 19. LCD I2C Clock PIO Parameters

Parameter	Value
Name	`lcd_i2c_scl`
Width (1-32 bits)	1
Direction	Output ports only
Output port reset value	0x0

Table 20 shows the LCD I2C data PIO parameters and corresponding parameter values.

Table 20. LCD I2C Data PIO Parameters

Parameter	Value
Name	`lcd_i2c_sdat`
Width (1-32 bits)	1
Direction	Bidirectional (tri-state) ports
Output port reset value	0x0

Table 21 shows the LCD I2C enable PIO parameters and corresponding parameter values.

Table 21. LCD I2C Enable PIO Parameters

Parameter	Value
Name	`lcd_i2c_en`
Width (1-32 bits)	1
Direction	Output ports only
Output port reset value	0x0

For information about these cores, refer to the *PIO Core* chapter in volume 5 of the *Quartus II Handbook*.

LCD Controller Software API

The LCD controller software application program interface (API) provides a high-level initialization function and a set of low-level functions for communicating with the LCD module registers. During normal operation, you only need to communicate with the LCD module during system configuration. This communication occurs through the software API.

For more information about the software API, refer to *AN 527: Implementing an LCD Controller*.

Touch Panel ADC Interface

The touch panel ADC interface is a 4-bit interface consisting of a 3-wire SPI to communicate with the touch panel controller and a PIO core to capture the interrupt generated when the pen moves on the touch panel. Table 22 shows the touch panel ADC SPI parameters and corresponding parameter values.

Table 22. Touch Panel ADC SPI Parameters (Part 1 of 2)

Parameter	Value
Name	`touch_panel_spi`
Master/Slave	Master
Number of select (`SS_n`) signals	1
SPI clock rate (`SCLK`)	32 KHz
Data register width	8 bits

Table 22. Touch Panel ADC SPI Parameters (Part 2 of 2)

Parameter	Value
Shift direction	MSB first
Timing clock polarity	0
Timing clock phase	0

For information about the SPI core, refer to the *SPI Core* chapter in volume 5 of the *Quartus II Handbook*.

Table 23 shows the LCD touch panel pen interrupt PIO parameters and corresponding parameter values.

Table 23. Touch Panel ADC Pen Interrupt PIO Parameters

Parameter	Value
Name	`touch_panel_pen_irq_n`
Width (1–32 bits)	1
Direction	Input ports only
Input options	■ Edge capture register: synchronous capture, falling edge ■ Interrupt generated: edge based

For information about the PIO core, refer to the *PIO Core* chapter in volume 5 of the *Quartus II Handbook*.

Touch Panel ADC Software API

The touch panel ADC software API provides your application with an abstract pen interface consisting of x and y coordinates, and pen state (up or down).

API operations are available to perform the following distinct actions on the touch panel:

- Initialize
- Calibrate
- Operate
- Stop

For more information about the API software, refer to *AN 527: Implementing an LCD Controller*.

Video Pipeline

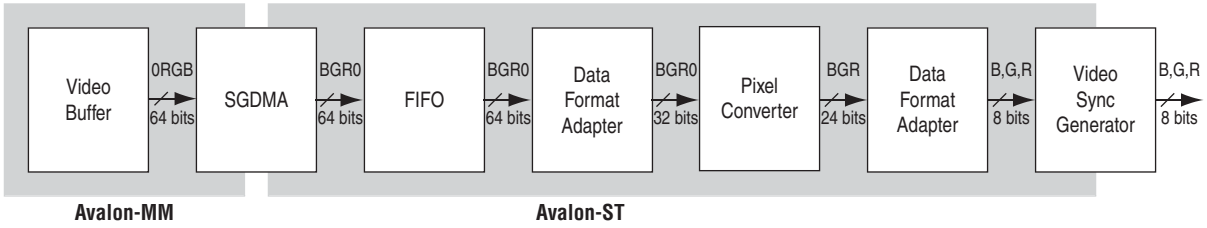
The video pipeline components produce the appropriate pixel data and sync signals to LCD touch panel. The video subsystem consists of the following ordered operational components:

1. Video frame buffer—resides in DDR memory.
2. Memory-to-stream DMA controller—reads memory 64 bits at a time and produces a stream of 64-bit data values.
3. Data format adapter—breaks the 64-bit stream into sequential 32-bit pixel values.

4. FIFO—rate matches video stream.
5. Pixel format converter—converts the 32-bit BGR0 stream to a 24-bit BGR stream.
6. Data format adapter—produces a stream of 8-bit values.
7. Video sync-generator—takes the RBG values and produces HSYNC and VSYNC timing signals for the LCD display.

Figure 2 shows the video pipeline.

Figure 2. Video Pipeline



Video Frame Buffer

The image to display resides in a video buffer as 32-bit unpacked pixel data (0RGB). The video buffer is part of the DDR SDRAM memory.

LCD SG-DMA Controller

For minimal interruption during the display of the video stream, the processor sets up an SG-DMA that accesses the pixel data stored in the video buffer on the DDR SDRAM and sends the pixel data through the video pipeline. The processor loads up a series of transfers in a descriptor table and the SG-DMA services the descriptor table and thus the stream video. Table 24 shows the LCD SG-DMA controller parameters and corresponding parameter values.

Table 24. LCD SG-DMA Controller Parameters

Parameter	Value
Name	`lcd_sgdma`
Transfer options	Memory to stream
Data width	64 bits
Source error width	0
Data transfer FIFO depth	2

For more information about this core, refer to the *Scatter-Gather DMA Controller Core* chapter in volume 5 of the *Quartus II Handbook*.

LCD Pixel FIFO

The LCD pixel FIFO is a dual clock on-chip FIFO used to buffer video packets in the event that the rate of pixel display is slower that the rate of retrieving video packets from the video buffer. Table 25 shows the LCD pixel FIFO parameters and corresponding parameter values.

Table 25. LCD Pixel FIFO Parameters

Parameter	Value
Name	lcd_pixel_fifo
Depth	128
Clock setting	Dual clock mode
Input	Avalon Streaming (Avalon-ST)
Output	Avalon-ST

For more information about this core, refer to the *On-Chip FIFO Memory Core* chapter in volume 5 of the *Quartus II Handbook*.

Pixel Format Converter

The video pipeline outputs a multiplexed stream of R, G, and B channels of 8 bits each. The Altera pixel format converter core simply takes the 32 bits and discards 8 bits to produce the 24 bits that represent the RGB stream. Table 26 shows the pixel format converter parameters and corresponding parameter values.

Table 26. Pixel Format Converter Parameters

Parameter	Value
Name	lcd_pixel_converter
Source symbols per beat	3

For more information about this core, refer to the *Video Sync Generator and Pixel Converter Cores* chapter in volume 5 of the *Quartus II Handbook*.

Data Format Adapter

The lcd_64_to_32_bits_dfa data format adapter converts 64-bit video frame data to 32-bit data. Table 27 shows the data format adapter (64 to 32) parameters and corresponding parameter values.

Table 27. Data Format Adapter (64 to 32) Parameters

Parameter	Value
Name	lcd_64_to_32_bits_dfa
Input interface	
Data symbols per beat	8
Output interface	
Data symbols per beat	4
Common to Input and Output	
Channel signal width	0
Max channel	0
Include packet support	Enabled
Error signal width	0
Data bits per symbol	8

The lcd_32_to_8_bits_dfa data format adapter converts the 24-bit stream representing RGB to three streams of 8 bit data, with one stream each respectfully representing the R, G, and B pixels. Table 28 shows the data format adapter (32 to 8) parameters and corresponding parameter values.

Table 28. Data Format Adapter (32 to 8) Parameters

Parameter	Value
Name	lcd_32_to_8_bits_dfa
Input interface	
Data symbols per beat	3
Output interface	
Data symbols per beat	1
Common to Input and Output	
Channel signal width	0
Max channel	0
Include packet support	Enabled
Error signal width	0
Data bits per symbol	8

For more information about this core, refer to the *Avalon Streaming Interconnect Components* chapter in volume 4 of the *Quartus II Handbook*.

Video Sync Generator

The Altera video sync generator core generates horizontal and vertical synchronization signals to the pixel data and outputs the data to an off-chip display controller. Horizontal synchronization signals synchronize the RGB pixels with the clock to produce a single line of the image. Vertical synchronization signals align a given number of lines to produce individual frames in an image. Table 29 shows the video sync generator parameters and corresponding parameter values. The parameters are user-configurable and define the timing characteristics of the horizontal and vertical synchronization signals.

Table 29. Video Sync Generator Parameters (Part 1 of 2)

Parameter	Value
Name	lcd_sync_generator
Data stream bit width	8
Beats per pixel	3
Number of columns	800
Number of rows	480
Horizontal blank pixels	216
Horizontal front porch pixels	40
Horizontal sync pulse pixels	1
Horizontal sync pulse polarity	0
Vertical blank lines	35

Table 29. Video Sync Generator Parameters (Part 2 of 2)

Parameter	Value
Vertical front porch lines	10
Vertical sync pulse lines	1
Vertical sync pulse polarity	0
Total horizontal scan pixels	1,056
Total vertical scan lines	525

Figure 3 shows the horizontal synchronization timing when the parameters are 8 bit data width and 3 beats (R, G, and B) per pixel.

Figure 3. Horizontal Synchronization Timing

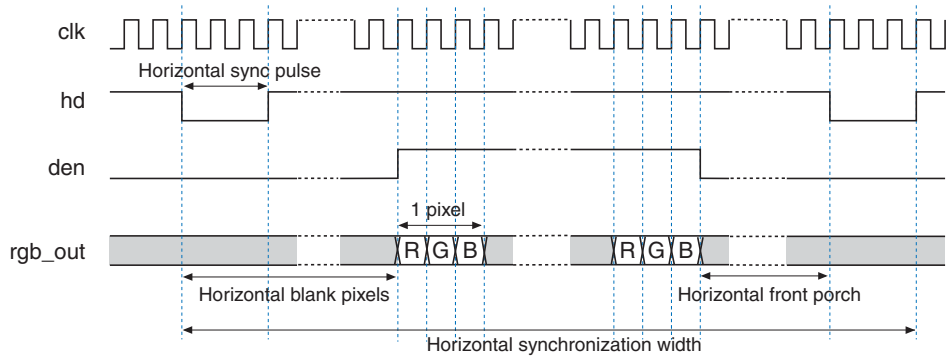
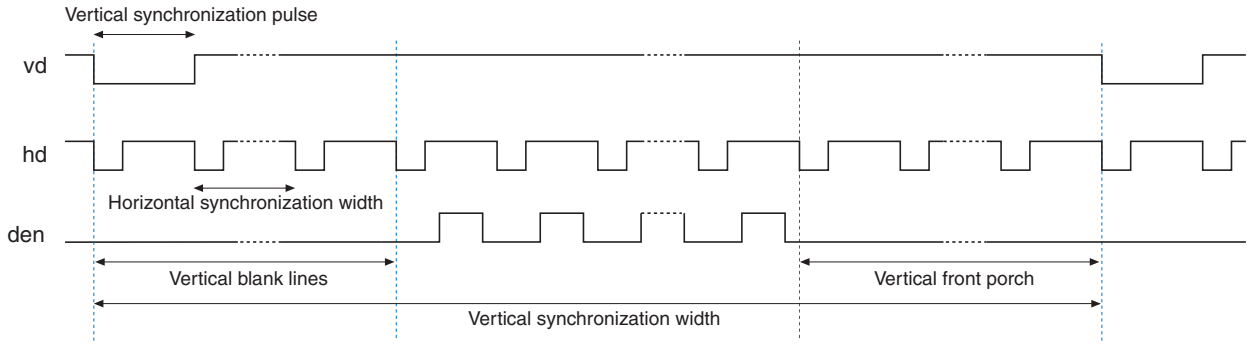


Figure 4 shows the relationship between the horizontal and vertical synchronization signals.

Figure 4. Relationship between Horizontal and Vertical Synchronization Signals



For more information about this core, refer to the *Video Sync Generator and Pixel Converter Cores* chapter in volume 5 of the *Quartus II Handbook*.

Video Pipeline Software API

The video pipeline software API provides a self-contained method to control the video pipeline and manage the graphical frame buffers.

API operations are available to perform the following three distinct actions on the video pipeline:

- Initialize
- Stop
- Manage and manipulate frame buffers

For more information about this software, refer to *AN 527: Implementing an LCD Controller*.

Software Interface

The **system.h** file provides a software description of the microprocessor hardware. The **system.h** file describes each peripheral in the system and provides the following details:

- The hardware configuration of the peripheral
- The base address and offset
- The IRQ priority (if any)
- Accessibility of slave by different masters
- A symbolic name for the peripheral

You can include the **system.h** file in your software application source code to interface with the hardware system.

For more information about **system.h** and the hardware abstraction layer (HAL), refer to the *Hardware Abstraction Layer* section of the *Nios II Software Developer's Handbook*.

Hardware Interface

SDC Timing Constraints

The following Synopsys Design Constraint (**.sdc**) files reside in the project directory:

- **cycloneIII_3c25_niosII_video.sdc**
- **ddr_sdram_phy_ddr_timing.sdc**

To add an **.sdc** file to the system, add the following constraint to your Quartus II Settings File (**.qsf**):

```
set_global_assignment -name SDC_FILE <SDC filename>.sdc
```

For further details on your Quartus II settings, refer to the **.qsf** file in the project directory.

Customization

Changing Device and Pin Assignments

This LCD system design is targeted to a Cyclone III 3C25 FPGA. You can target your design to other FPGAs by specifying options on the **Device Settings** page in the Quartus II software.

For more information about specifying a device family, refer to Quartus II Help.

The pin assignments for the microprocessor have to be re-assigned after changing the device. Use the Quartus II software's Assignment Editor to assign the pins according to the printed circuit board (PCB) schematic design for the FPGA and other devices on your board.

For more information about using the Assignment Editor, refer to the *Assignment Editor* chapter in volume 2 of the *Quartus II Handbook*.

Configuring the Processor

The Nios II processor is configurable, soft-core processor. Use the Nios II processor MegaWizard interface in SOPC Builder to reconfigure the Nios II processor settings for the existing LCD system design.

For more information about the Nios II processor MegaWizard interface and settings, refer to the *Instantiating the Nios II Processor in SOPC Builder* chapter in the *Nios II Processor Reference Handbook.*

Adding IP cores

You can add new IP cores into the existing LCD system design in the SOPC Builder.

For more information about adding IP cores in SOPC Builder, refer to *Volume 4: SOPC Builder* of the *Quartus II Handbook*.

Adding Custom Components

You can create your own custom components and add them to the LCD system design. The SOPC Builder's Component Editor helps integrate your custom components into a system.

For more information about the Component Editor, refer to the *Component Editor* chapter in volume 4 of the *Quartus II Handbook.*

You can develop custom device drivers for the custom components added in the system.

For more information about developing device driver, refer to the *Developing Device Drivers for the Hardware Abstraction Layer* chapter in *Nios II Software Developer's Handbook*.

Scaling Performance

There are several ways to improve system performance. Documents in the following list describe improvement methods:

- To add custom instruction to the Nios II processor, refer to the *Nios II Custom Instruction User Guide*.
- To create a multiprocessor system, refer to *Creating Multiprocessor Nios II Systems Tutorial*.

- To use C-to-hardware (C2H) acceleration for time critical code, refer to the following documents:
  - *Accelerating Nios II Systems with the C2H Compiler Tutorial*
  - *Nios II C2H Compiler User Guide*
- To use tightly coupled memory with the processor, refer to *Using Tightly Coupled Memory with the Nios II Processor Tutorial*.
- To incorporate system level design optimization techniques, refer to *System-Level Design* section of the *Embedded Design Handbook*.
- To accelerate Nios II networking applications, refer to *AN 440: Accelerating Nios II Networking Applications*.

Referenced Documents

This data sheet references the following documents:

- *Accelerating Nios II Systems with the C2H Compiler Tutorial*
- *altpll Megafunction User Guide*
- *AN 440: Accelerating Nios II Networking Applications*
- *AN 483: Triple Speed Ethernet Data Path Reference Design*
- *AN 527: Implementing an LCD Controller*
- *Assignment Editor* chapter in volume 2 of the *Quartus II Handbook*
- *Avalon Streaming Interconnect Components* chapter in volume 4 of the *Quartus II Handbook*
- *Common Flash Interface Controller Core* chapter in volume 5 of the *Quartus II Handbook*
- *Component Editor* chapter in volume 4 of the *Quartus II Handbook*
- *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*
- *Creating Multiprocessor Nios II Systems Tutorial*
- *Cyclone III Device Handbook*
- *Cyclone III Remote Update Controller Core* chapter in volume 5 of the *Quartus II Handbook*
- *DC and Switching Characteristics* chapter in volume 2 of the *Cyclone III Device Handbook*
- *DDR and DDR2 SDRAM High-Performance Controller User Guide*
- *Developing Device Drivers for the Hardware Abstraction Layer* chapter in *Nios II Software Developer's Handbook*
- *Ethernet and the NicheStack TCP/IP Stack - Nios II Edition* chapter in the *Nios II Software Developer's Handbook*
- *Hardware Abstraction Layer* section of the *Nios II Software Developer's Handbook*

- *Instantiating the Nios II Processor in SOPC Builder* chapter in the *Nios II Processor Reference Handbook*
- *JTAG UART Core* chapter in volume 5 of the *Quartus II Handbook*
- *Nios II C2H Compiler User Guide*
- *Nios II Custom Instruction User Guide*
- *On-Chip FIFO Memory Core* chapter in volume 5 of the *Quartus II Handbook*
- *Performance Counter Core* chapter in volume 5 of the *Quartus II Handbook*
- *Pin Information for the Cyclone III EP3C25 Device*
- *PIO Core* chapter in volume 5 of the *Quartus II Handbook*
- *Remote System Upgrade With Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*
- *Scatter-Gather DMA Controller Core* chapter in volume 5 of the *Quartus II Handbook*
- *SPI Core* chapter in volume 5 of the *Quartus II Handbook*
- *System ID Core* chapter in volume 5 of the *Quartus II Handbook*
- *System-Level Design* section of the *Embedded Design Handbook*
- *Timer Core* chapter in volume 5 of the *Quartus II Handbook*
- *Triple Speed Ethernet MegaCore Function User Guide*
- *UART Core* chapter in volume 5 of the *Quartus II Handbook*
- *Using Tightly Coupled Memory with the Nios II Processor Tutorial*
- *Video Sync Generator and Pixel Converter Cores* chapter in volume 5 of the *Quartus II Handbook*
- *Volume 4: SOPC Builder* of the *Quartus II Handbook*

Revision History

Table 30 shows the revision history for this data sheet.

Table 30. Revision History

Date and Revision	Changes Made	Summary of Changes
March 2009 v1.1	Made corrections to block diagram and pin tables.	—
January 2009 v1.0	Initial Release.	—

101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support
www.altera.com/support