

Core Overview

The Optrex 16207 LCD controller core with Avalon® Interface (LCD controller core) provides the hardware interface and software driver required for a Nios® II processor to display characters on an Optrex 16207 (or equivalent) 16×2-character LCD panel. Device drivers are provided in the HAL system library for the Nios II processor. Nios II programs access the LCD controller as a character mode device using ANSI C standard library routines, such as `printf()`. The LCD controller is SOPC Builder-ready, and integrates easily into any SOPC Builder-generated system.

The Nios II Embedded Design Suite (EDS) includes an Optrex LCD module and provide several ready-made example designs that display text on the Optrex 16207 via the LCD controller. For details about the Optrex 16207 LCD module, see the manufacturer's *Dot Matrix Character LCD Module User's Manual* available at www.optrex.com.

This chapter contains the following sections:

- “Functional Description”
- “Device and Tools Support” on page 8-2
- “Instantiating the Core in SOPC Builder” on page 8-2
- “Software Programming Model” on page 8-2

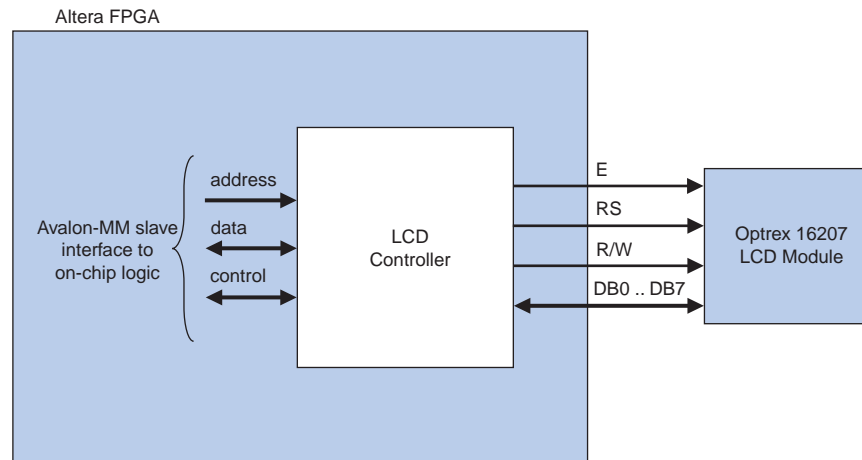
Functional Description

The LCD controller core consists of two user-visible components:

- Eleven signals that connect to pins on the Optrex 16207 LCD panel—These signals are defined in the Optrex 16207 data sheet.
 - E—Enable (output)
 - RS—Register Select (output)
 - R/W—Read or Write (output)
 - DB0 through DB7—Data Bus (bidirectional)
- An Avalon Memory-Mapped (Avalon-MM) slave interface that provides access to 4 registers.

Figure 8-1 shows a block diagram of the LCD controller core.

Figure 8-1. LCD Controller Block Diagram



Device and Tools Support

The LCD controller core supports all Altera device families. The LCD controller drivers support the Nios II processor.

Instantiating the Core in SOPC Builder

You can add the LCD controller core from the **System Contents** tab in SOPC Builder. In SOPC Builder, the LCD controller core has the name Character LCD (16x2, Optrex 16207). There are no user-configurable settings for this component.

Software Programming Model

This section describes the software programming model for the LCD controller.

HAL System Library Support

Altera provides HAL system library drivers for the Nios II processor that enable you to access the LCD controller using the ANSI C standard library functions. The Altera-provided drivers integrate into the HAL system library for Nios II systems. The LCD driver is a standard character-mode device, as described in the *Nios II Software Developer's Handbook*. Therefore, using `printf()` is the easiest way to write characters to the display.

The LCD driver requires that the HAL system library include the system clock driver.

Displaying Characters on the LCD

The driver implements VT100 terminal-like behavior on a miniature scale for the 16×2 screen. Characters written to the LCD controller are stored to an 80-column × 2-row buffer maintained by the driver. As characters are written, the cursor position is updated. Visible characters move the cursor position to the right. Any visible characters written to the right of the buffer are discarded. The line feed character (`\n`) moves the cursor down one line and to the left-most column.

The buffer is scrolled up as soon as a printable character is written onto the line below the bottom of the buffer. Rows do not scroll as soon as the cursor moves down to allow the maximum useful information in the buffer to be displayed.

If the visible characters in the buffer fit on the display, all characters are displayed. If the buffer is wider than the display, the display scrolls horizontally to display all the characters. Different lines scroll at different speeds, depending on the number of characters in each line of the buffer.

The LCD driver supports a small subset of ANSI and VT100 escape sequences that can be used to control the cursor position, and clear the display as shown in [Table 8–1](#).

Table 8–1. Escape Sequence Supported by the LCD Controller

Sequence	Meaning
BS (<code>\b</code>)	Moves the cursor to the left by one character.
CR (<code>\r</code>)	Moves the cursor to the start of the current line.
LF (<code>\n</code>)	Moves the cursor to the start of the line and move it down one line.
ESC((<code>\x1B</code>)	Starts a VT100 control sequence.
ESC [<y> ; <x> H	Moves the cursor to the y, x position specified – positions are counted from the top left which is 1;1.
ESC [K	Clears from current cursor position to end of line.
ESC [2 J	Clears the whole screen.

The LCD controller is an output-only device. Therefore, attempts to read from it returns immediately indicating that no characters have been received.

The LCD controller drivers are not included in the system library when the **Reduced device drivers** option is enabled for the system library. If you want to use the LCD controller while using small drivers for other devices, add the preprocessor option—`DALT_USE_LCD_16207` to the preprocessor options.

Software Files

The LCD controller is accompanied by the following software files. These files define the low-level interface to the hardware and provide the HAL drivers. Application developers should not modify these files.

- **altera_avalon_lcd_16207_regs.h** — This file defines the core’s register map, providing symbolic constants to access the low-level hardware.
- **altera_avalon_lcd_16207.h, altera_avalon_lcd_16207.c** — These files implement the LCD controller device drivers for the HAL system library.

Register Map

The HAL device drivers make it unnecessary for you to access the registers directly. Therefore, Altera does not publish details about the register map. For more information, the `altera_avalon_lcd_16207_regs.h` file describes the register map, and the *Dot Matrix Character LCD Module User's Manual* from Optrex describes the register usage.

Interrupt Behavior

The LCD controller does not generate interrupts. However, the LCD driver's text scrolling feature relies on the HAL system clock driver, which uses interrupts for timing purposes.

Referenced Documents

This chapter references the *Nios II Software Developer's Handbook*.

Document Revision History

Table 8-2 shows the revision history for this chapter.

Table 8-2. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
March 2009 v9.0.0	No change from previous release.	—
November 2008 v8.1.0	Changed to 8-1/2 x 11 page size. No change to content.	—
May 2008 v8.0.0	No change from previous release.	—



For previous versions of the *Quartus II Handbook*, refer to the [Quartus II Handbook Archive](#).