

CPE 329: Programmable Logic and Microprocessor-Based System Design

Laboratory 1 (Revised Spring 2007 jgh)

Digital Clock Design Using Programmable Logic and VHDL

Learning Objectives

- To gain experience with the Nexys development board
- To gain further experience with a top-down approach to digital system design
- To gain further experience designing synchronous digital systems using VHDL
- To gain a further understanding of the flexibility FPGAs

WARNING: The eight switches on the Nexys board (SW7:SW0) should be placed and remain in the down position when using the LCD display; these eight signal lines also are contained in the 16 pin header that connects the LCD accessory. Because of this fact, the switches on the Nexys board will not be used when the LCD display is used.

Introduction and Overview

Lab 1 involves the design, simulation, and implementation of a digital clock on the Nexys development board. The time-keeping functions of the clock are designed using VHDL structural modeling and downloaded to the Nexys board. The time data is output and the time is displayed on the Nexys LCD module display. Two VHDL modules, arbiter and hex2bcd, are provided to the students for use in their VHDL design to interface with the LCD display.

Digital Clock Requirements

1. The clock should display the current time in one of the formats shown in Figure 1 (dependent upon the actual time). The time output is displayed in a 12-hour display mode; the proper time of day is indicated by using the *am/pm* portion of the display.



Figure 1: Output display format for digital clock.

2. Upon power-up, the clock output should be **12:00:01 am** and begin keeping time.
3. The clock's hours and minutes setting should have the ability to be changed by the user in order to set the clock's output to the appropriate time.
4. Use the minimum number of FPGA programmable logic resources (slices).

Digital Clock Specification

The time output is manually changeable using the buttons on the Nexys board. Setting the time manually is referred to as the *time-set* mode. The signals used for the time-set mode and their function are listed in Table 1. These signals are used to put the digital clock into a time-set mode and to manually configure the data associated with the minute and hours outputs. The minutes and hours display can only be manually changed when the digital clock is in the time-set mode.

Name	Function
SET	Puts digital clock in time-set mode; disables seconds output
MN_SET	Increments the minutes output
HR_SET	Increments the hours output

Table 1: Display control functions of digital clock.

- **Time-set mode:** The time-set mode is entered by asserting the SET input. When in time-set mode the seconds should not increment.
- **Setting the minutes:** to modify the minutes display, the SET input should be asserted simultaneously with the MN_SET input which causes the minutes display to increment at 1/2 second intervals. The minutes display should increment from 59 to 00 with out changing the hours display value.
- **Setting the hours:** to modify the hours display, the SET input should be asserted simultaneously with the HR_SET input which causes the hours display to increment at 1/2 second intervals. The hours display should increment until reaching 12 and then rolls over to 1 and continues incrementing. The am/pm indicator should change appropriately with the hours display value.

Digital Clock Architecture

A partial diagram of the digital clock architecture is shown in Figure 2. A brief description of the main modules in the architecture is provided below.

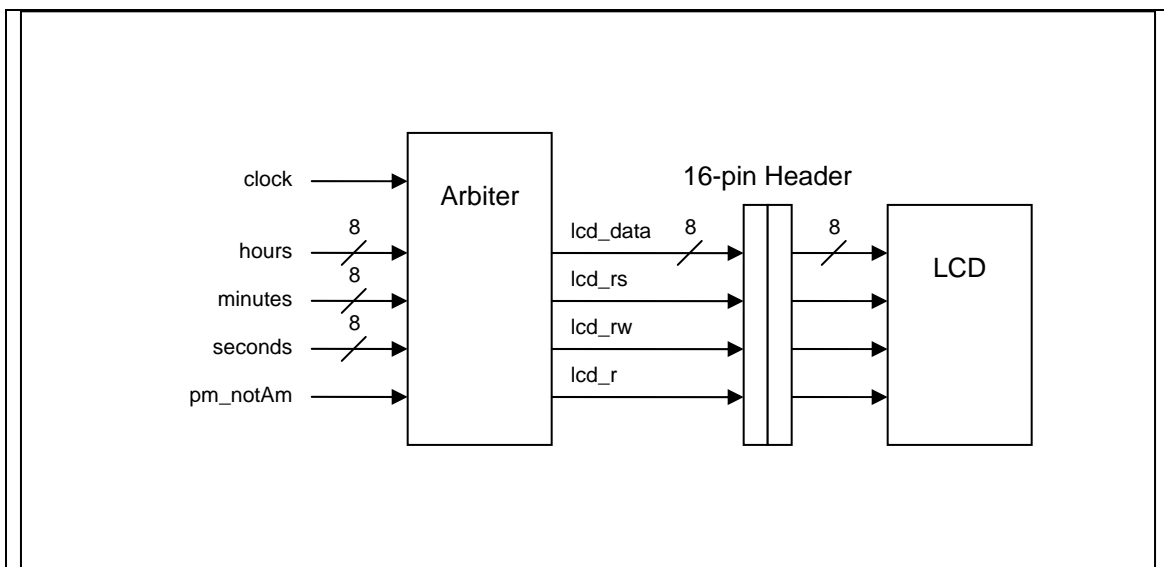


Figure 2: Digital clock system architecture.

Arbiter Module: The interface requirements for communication with the LCD display is not trivial. The LCD characters are written using appropriate address, data, and control signals. The Arbiter module function is to hide most of the low-level interface details from the designer. The Arbiter module converts the BCD information associated with the time display into ASCII characters which are then sent to the LCD controller module for display on the LCD. The VHDL code for the Arbiter module is provided on the CPE 329 website. The digital clock inputs and outputs required to interface with the Arbiter module are listed in Table 2. Note that the system clock for the design is the CLK signal used by the Arbiter module.

Signal	Mode	Description
seconds	IN	seconds count in BCD form (two 4-bit BCD numbers)
minutes	IN	minutes count in BCD form (two 4-bit BCD numbers)
hours	IN	hours count in BCD form (two 4-bit BCD numbers)
PM_notAM	IN	indicates PM or AM ('1' or '0', respectively)
CLK	IN	50MHz system clock

Table 2: Signals of particular interest on the Arbiter module.

Digital Clock Module (not shown in Figure 2): The digital clock module interfaces with the Arbiter module which handles the I/O to the LCD. The primary functions of the digital clock module are listed below.

- **1-Second clock Generation:** The system clock on the Nexys board is to run at 50MHz. This frequency must be divided down and used to drive the various modules that require a 1-second clock.
- **Time Data Generation:** The digital clock module keeps time while the Arbiter module expects three BCD inputs for the seconds, minutes, and hours data. There are several viable approaches to storing time data. Be sure to check out the HEX2BCD module (described below) before you decide on a design.
- **PM_notAM Generation:** The PM_notAM signal tells the Arbiter module whether the hours data is in the first or final twelve hours of the day. As the signal name implies, the convention PM_notAM is low for am and high for pm.
- **Button Input:** The buttons on the Nexys board set the time on the digital clock module. The input device assignments are listed in Table 3.

Input Device	Signal Name
BTN0	SET
BTN2	MN_SET
BTN3	HR_SET

Table 3: Mapping assignment between input devices and system signals.

HEX2BCD Module: This module converts a single 6-bit binary number into two representative BCD numbers. The use of this module is optional so you must examine the VHDL code for interface specifications if you decide to use it. The VHDL code for this module is found on the class website.

Digital Clock Components

The three basic components of the digital clock are the provided Arbiter and HEX2BCD VHDL code and the student designed Digital Clock VHDL code. There are no software components for this design.

Proper component development should include the following:

1. Testing each module you design by simulating the output using the ModelSim simulator. The component input vectors should provide adequate testing for each of the components.
2. Simulation results from the ModelSim simulator. These waveforms should be outputs from the ModelSim software. Screen capture (using “Alternate-Print Screen”) can be used in the ModelSim waveform window to place a copy of the simulation waveform into the editor buffer. Then you can just paste a copy into your lab report (for MS word paste with “Control-V”). The ModelSim waveforms should be annotated in order to draw attention and describe the important portions of the waveforms.

Digital Clock System Integration

Each of the modules you design should be tested before they are integrated into the final design. Each of the components in the digital clock design should be integrated using VHDL structural modeling.

Other Implementation Details: Once you’ve created and synthesized the required modules for the digital clock design, the design must be downloaded to the Nexys board. This requires an implementation constraints file that maps the required inputs and outputs to the pin assignments that exist on the Nexys board. The pin assignments for the Nexys board are listed in Table 5. These pin assignments are fixed on the board so you need to map the top level entity inputs and outputs to the correct pin assignments. Some additional information associated with mapping your design to the FPGA is listed below.

- The on board 50MHz system clock drives FPGA pin “A8.” **The clock input to the top level entity must be assigned to pin “A8”.**
- The signal names associated with the Arbiter interface and the pins for the LCD signals identified in Figure 2 are shown in Table 4 below.
- The pin assignments for the buttons on the Nexys board (BTN3:BTN0) are addressed in Exp0 of CPE 229.

Figure 2 Signal Name	Nexys Pin
lcd_data<0>	N15
lcd_data<1>	J16
lcd_data<2>	K16
lcd_data<3>	K15
lcd_data<4>	L15
lcd_data<5>	M16
lcd_data<6>	M15
lcd_data<7>	N16
lcd_rs	P15
lcd_rw	T7
lcd_r	R5

Table 4: Arbiter signal name mapping.

The Nexys board contains a Spartan 3 FPGA. The *Project Properties* listed in Table must be set when programming the Nexys board.

Property Name	Value
Device Family	Spartan3
Device	XC3S200
Speed Grade	-4
Package	FT256

Table 5: Project Properties setting for D2FT board.

Project Demonstration and Lab Report Submission

When your project is working properly and your lab report is ready to submit, zip up the project and lab report into one file and submit it electronically for grading. Be prepared to answer any questions about the experiment during the lab demonstration.

Include in your lab report the following design sections that describe the stages of the top-down design process: (For more information on lab reports reference the “A note regarding lab reports...” link on the CPE 329 website.)

1. System Requirements
2. Specification
3. Architecture
4. Components
5. System Integration

Include all VHDL code you wrote for your design with your report. Your report should be clear and concise in its description of the digital clock design. Remember, diagrams are a viable method to conveying important design information; consider using them in conjunction with written descriptions of your design.

Your lab report should also include a *Conclusions* section. This section **should not be** a summary of the procedures or a description of what was done in the experiment. This section should include an analysis of the final system and may include topics such as: were the system requirements met? Is there any error in the final system and if so estimate it? Could you have met the system requirements with a better design or an alternate approach? If so how? What tradeoffs were made in the areas of cost, performance, accuracy, power consumption or reliability (if any of these are appropriate)?

Grading

Successful completion of the system requirements and a well written lab report will earn a maximum grade of 9 out of 10 points. In order to achieve a grade above 9 on this lab, you need to demonstrate a complete understanding of the lab by adding additional features/functionality to it. Some suggested lab enhancement ideas are listed below. Higher complexity will earn a higher grade. Adding other features/functionality is permitted as long as they are approved by an instructor and documented in an “Added Features” section following the conclusion of your report. Any changes made to the original specification due to added features must be listed along with an explanation of why the enhancement was made.

Suggested added features:

- Ability to switch between standard and 24-hour time representation
- Minute and hour decrement buttons