Laboratory III

Servomotor Controller

Introduction

This lab deals with issues of generating signals to drive the FPGA's I/O pins directly. The RC-200E brings several FPGA pins out to the 50-pin expansion header, along with some power and ground connections. In this lab you will connect a servomotor to one of the pins in the expansion header and use the pushbuttons to make the motor rotate one way, the other way, or not at all. To operate correctly, your code must generate pulses of exactly the correct frequency and pulse widths, and you will verify the correct behavior both by simulation and using an oscilloscope before actually connecting a servomotor to the RC200E.

Project Specifications

Your program is to generate a pulse every 20 msec. If one pushbutton is pressed, the pulse is to be 1.0 msec long; if the other button is pressed, the pulse is to be 2.0 msec long. If neither button (or both buttons) is pressed, the pulse is to be 1.5 msec long. The servomotors we are using, Futaba model number 3003, rotates clockwise when it receives 1.0 msec pulses, counterclockwise when it receives 2.0 msec pulses, and not at all when it receives 1.5 msec pulses. (Clockwise and counterclockwise depend on your point of view.)

Your program must use one endless loop to time the millisecond intervals and another endless loop to generate the pulses. The two loops must synchronize with each other using a Handel-C channel.

Lab Activities

1. **Work Through the First Waveform Analyzer Example**
2. **Write a Program that Outputs to a Pin**
    a. **Verify your program using the Waveform Analyzer**
    b. **Verify your program using an Oscilloscope**
3. **Write a Program that Controls a Servomotor**
    a. **Verify your program using the Waveform Analyzer**
    b. **Verify your program using an oscilloscope**
    c. **Verify your program using a servomotor**
4. **Submit a Report of Your Lab Activities**

Work Through the First Waveform Analyzer Example

Create a DK Workspace named "Laboratory III" for this lab in your "My Projects" directory. Add a new project named "View Waveforms" to this workspace, and add a Handel-C source file named *view_waveforms.hcc* to the project. You do not need to configure the project in anyway; the default settings for the Debug configuration will suffice.

Find the Waveform Analyzer Manual on your system. (…\Celoxica\DK\Documentation) and follow the directions on pages 5-8 for tracing the execution of a simple Handel-C program. Typing in the code and following the directions carefully will be instructive in itself. Write answers to the following questions within the comments of your code.

- What are DKSync.dll and DKConnect.dll?
- Measure the period of the output waveform using cursors (see pages 10-11). Relate the period to the set clock statement and the values that the variable *x* takes on in the Handel-C program.

Write a Program that Outputs to a Pin

Create a second project named "Generate Pulses" in the Laboratory III workspace, and configure it for EDIF and Simulation in the usual way. Write a macro procedure named *microsec_delay()* that delays the program for the number of microseconds passed as a parameter. Code the program so that it endlessly generates a 1 msec pulse every 20 msec. Have the output of the program connect to Pin #3 of the RC200E expansion header.

To connect a variable in your program to an output pin, you need to use a *bus_out()* interface. The syntax and some examples are given in your Handel-C Reference Manual starting on page 51. You also saw some examples of interface declarations in the Waveform Analyzer project.

You need to know what FPGA pin is connected to pine #3 of the RC200E expansion header. Look it up in the RC200 Hardware and Installation Manual available on the course web site and/or in the …\Celoxica\PDK\PSL\RC200 directory.

Verify your program using the Waveform Analyzer

The Waveform Analyzer Help web page gives you additional information on using the Waveform Analyzer for this project.

Verify your program using an oscilloscope

When you are ready to test your code using an oscilloscope, follow these instructions:

Set all the rotating knobs to their upright positions except the three labeled "VAR" which should be set to the "CAL" (calibrated) positions. Apply power to the scope, and clip the A probe to the "CAL" loop on the lower left front corner of the scope. Press the green "Auto Set" button and adjust the intensity control (under the power switch) so the display is neither too dim nor too bright. There should be two traces on the screen, one showing a square wave (the A channel) and the other showing noise (the B channel). Press the A/B button under the Auto Set button to make the B channel go away. Look at the information displayed in the dim window to the right of the screen and verify that the amplitude of the square wave is 1.25 volts and

that the period is 5 msec. You may need to adjust the X and Y positions of the trace to be able to measure these values.

With the scope set up, you can connect the probe to pin #3 of the RC200E expansion header; you probably don't need to connect the ground connection on the probe, but you might want to connect it to one of the RC200E expansion header ground pins to get a good display. Press Auto Set to get the scope to adjust to the signal you are generating. Press the left or right side of the MTB rocker switch to make the on time of your pulses take up two horizontal boxes on the display, and verify that you have 0.5 ms per box (in the dim window) and that the "VAR" knob for X is in the "CAL" position.

Write a Program that Controls a Servomotor

Create a third project named "Motor Control" in the Laboratory III workspace, and configure it for EDIF and Simulation in the usual way.

Organize your code with two *main()* methods. One main method is to do nothing but write a value to a channel every 20 msec. The second *main()* method is to execute an endless loop that reads a value from the channel, then reads the two pushbuttons (use the PAL for this) and turns on the output pin. Depending on the settings of the switches, delay for 1000, 1500, or 2000 microseconds, and then turn off the output pin. Be sure you understand the logic of this code and how it prevents the 20 msec intervals from "drifting." Include comments in your code to explain the design.

Verify your program using the Waveform Analyzer

Use the Waveform Analyzer to verify that all pulses start on 20 msec intervals and that the pulse widths are correct depending on which buttons are pressed.

Verify your program using an oscilloscope

Connect the oscilloscope to an RC200E and verify that the pulses are generated every 20 msec (with no drift) and that the width of the pulses can be varied between 1.0, 1.5, and 2.0 msec using the two pushbuttons.

Verify your program using a servomotor

Connect a servomotor to an RC200E and verify that it rotates in the expected directions when the buttons are pressed. Demonstrate your program to Dr. Vickery when you have it working.

Submit a Report of Your Lab Activities

Use a word processor to write a report of your lab activities that follows the format of the Lab Report Guidelines for this course, and email it to me by midnight of the due date.