Signoffs and Grade:

Name:_____

Component	Signoff	Date	Time
Working Simulation Must show entire cycle of FSM			
Working Sweeping servo			

==========

Component	Received	Possible
Prelab		10
Signoffs		90
Penalties • after the first 15 minutes of lab session 5: -10 • after the first 15 minutes of lab session 6: -25 no signoffs after the first 15 minutes of lab session 7:	-	
Total		100

Education Objective
The educational objective of this laboratory is to investigate the creation of custom IP intended for Lab5. This lab is two parts, combined with Lab 5. Lab 4 focuses on the creation of the core, and Lab 5 on the software interfacing with the component.

Technical Objective
The technical objective of this laboratory is to design a custom component that will control a servo motor, sweeping between two angles. In Lab 5, this custom component will be added to the NIOSII system, but for the purposes of Lab 4, only a VHDL component will be designed.

This custom component creates the appropriate Pulse-Width Modulation (PWM) signal to sweep the provided HiTec HS-311 servo-motor's arm.

The system operates as follows:

- The servo controller outputs a waveform to the servo motor that causes it to continuously sweep from the minimum angle to the maximum angle and back again.
  - Servo period is 20ms – managed by a period counter
  - Pulse Width is determined by current angle – managed by angle counter
  - The resulting output waveform can be sent to a GPIO pin
    - Should be the result of comparing two counters above
- The servo controller generates an output interrupt request (IRQ) that indicates when it has completed a full sweep in one direction.
  - Use a state machine to manage the sweeping
- The servo controller accepts register writes from an outside source.
  - 2 Registers
    - Minimum Angle
    - Maximum Angle
  - Write enable
  - Address
  - Write Data

Servo Motor Background
The text below is taken from ServoCity.com

Servos are controlled by sending them a pulse with a variable high time width. The control wire of the servo motor accepts this pulse. The parameters for this pulse are that it has a minimum pulse, a maximum pulse, and a constant period. A sample pulse is shown in Figure 1 below.

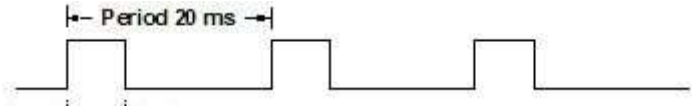

Figure 1: Sample Waveform for Servo Motor

The angle of rotation is determined by the duration of a pulse that is applied to the control wire. This is called Pulse Width Modulation (PWM). The servo expects to see a pulse every 20ms. The length of the pulse will determine how far the motor turns. For example, a 1.5ms pulse will make the

motor turn to the 90 degree position (neutral position), the minimum width pulse will make the motor turn to the 45 degree position and the maximum width pulse will turn it to the 135 degree position.

When servos are commanded to move they will move to the position and hold that position. If an external force pushes against the servo while the servo is holding a position, the servo will resist moving out of that position. The maximum amount of force the servo can exert is the torque rating of the servo. Servos will not hold their position forever though; the position pulse must be repeated to instruct the servo to stay in position.

When a pulse is sent to a servo that is less than 1.5ms the servo rotates to a position and holds its output shaft some number of degrees counterclockwise from the neutral point. When the pulse is wider than 1.5ms the opposite occurs. The minimal width and the maximum width of pulse that will command the servo to turn to a valid position are functions of each servo. Different brands, and even different servos of the same brand, will have different maximum and minimums.

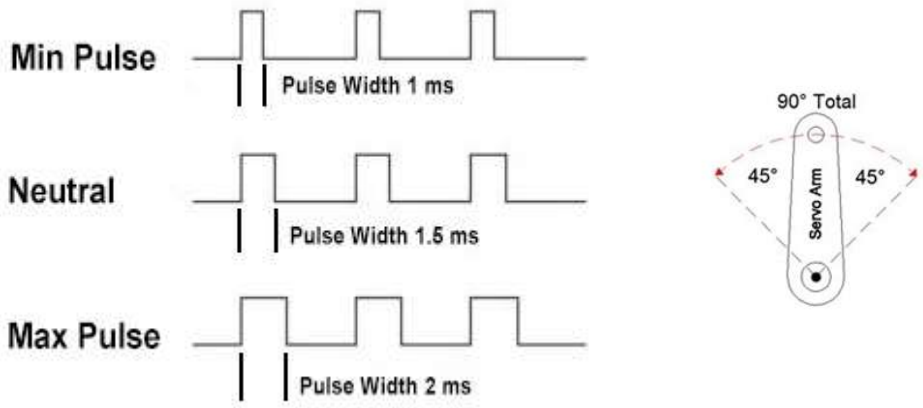

Figure 2: Servo Motor Pulse Width Timing Waveforms

Specifications for the HiTec HS-311 servo motor used in this lab can be found at the web address: https://www.servocity.com/hs-311-servo/

NOTE To create delays in VHDL, use counters. The clock on the DE1-SoC board is 50 MHz which means it has a period of 20ns. For a delay of 20ms, the count would be 20E-3 x 50E6 = 1,000,000. You will need to calculate the time for the default minimum and maximum angle values.

IP Design

The servo controller entity will have the interfaces illustrated in Figure 3. If you do not follow this component layout, you will have trouble interfacing with the processor in Lab 5.

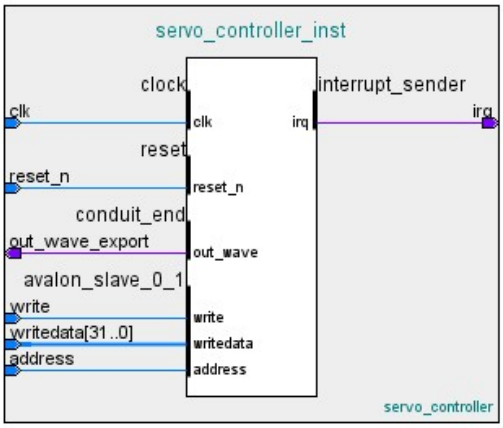

Figure3: Servo controller interface diagram

The servo controller should have 2 internal registers to store the angle information. These registers are 32 bits wide. Since there are only 2 registers, only 1 address bit is needed to choose between the two. The writedata is written to the addressed register when the write signal is active. Although it is customary to clear registers to 0 upon reset, in this case the registers should be initialized to the counts for a 45 to 135 degree sweep.

Offset	31.....0
0	Count for min angle pulsewidth
1	Count for max angle pulsewidth

Figure 4: Register map for servo controller

To create the output wave, a counter is needed to keep track of the 20ms period. While the current count is less than the angle count, the waveform is driven high. For the remainder of the period it is driven low. To enable a smooth sweep of the servo motor, the pulse width should start at the min angle width and gradually increment until it reaches the max angle width. Once it reaches the max angle width, it should gradually decrement until it reaches the min angle width.

For simulation, the increment and decrement amount should be large, so that the state machine will sweep through states quickly. For real time operation, you can experiment with that number to change the speed and “smoothness” of the sweep.

There is more than one way to write the code but it is highly recommended that you use a state machine to organize your thoughts. Below is a suggested state transition diagram, however, you may design the control logic however you would like.

Notice that writing to one of the internal registers is the transition condition out of the interrupt states and thus the action necessary for clearing the IRQ signal.

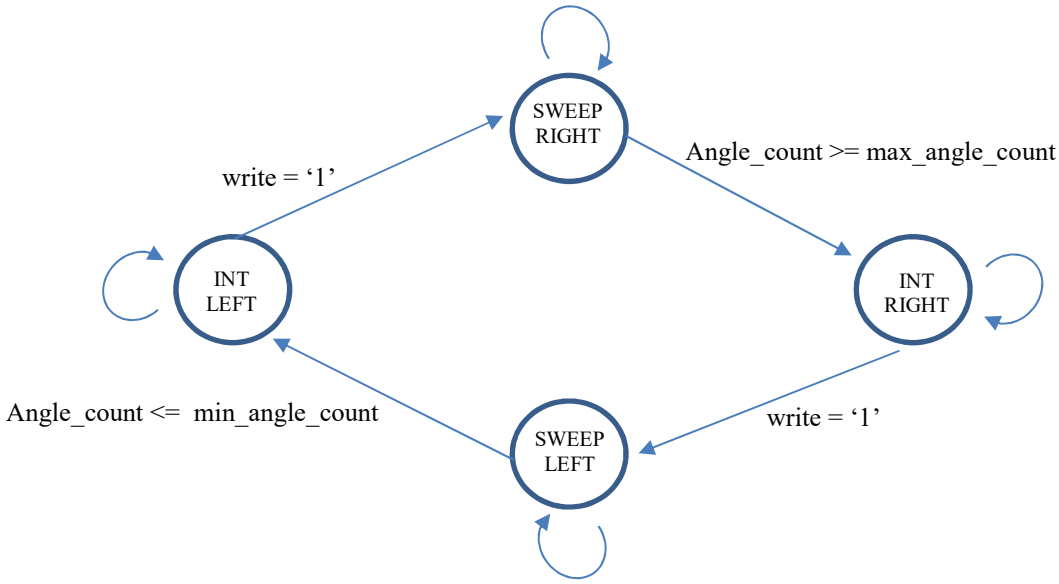

Figure 5: Possible State Transition Diagram for the servo controller

Prelab

Part 1 – Design the VHDL Servo Controller

1. Write the VHDL file to implement the servo motor controller. It does not have to work perfectly, but it should have the following components.
    a. Control logic (FSM)
    b. Register logic
    c. IRQ
    d. Period counter
    e. Angle counter
2. Open Quartus II and create a new project. Add your servo_controller.vhd file to the project.

Part 2 –

1. Write a test bench to test the servo motor controller. Remember to assign constants/generics that make sense for simulation.
    - Verify that your controller operates correctly, increasing its pulsewidth until it reaches the max and then decreasing until it reaches the min.
    - Verify that it raises the IRQ signal when the min and max angle counts are reached and that writing to the registers clears the interrupt.

Procedure

1. Demonstrate your working simulation
    a. IRQ is generated
    b. Successful writes to both registers
    c. At least 1 full pass through the state machine
    d. Obtain a signoff

2. Demonstrate that your servo controller can automatically sweep from a minimum of 45 degrees to maximum of 135 degrees when connected to a GPIO.
    a. Modify your servo controller to always acknowledge interrupts with an always HIGH write enable.
    b. Modify your servo controller to always write to register 0 the minimum value.
    c. Obtain a signoff for your working servo.

The GPIO pins for the DE1-SoC can be found online, but here is a quick reference.