ECPE 174 – Lab
Asynchronous Communication

I. Objectives

Design and implement an asynchronous communication system to transmit a value from one Cyclone board to another.

II. Problem Statement

Two Cyclone devices will communicate to transfer values between them. They utilize the following protocol:

- Each has two signals *devA* and *devB*
- When *send* initiates a transaction from device A to talk to device B, it pulls *devA* high.
- When device B is ready, it pulls *devB* high.
- Device A then pulls *devA* low, acknowledging the end of the valid data.
- Device B then concludes the transaction by pulling *devB* low.

System is bidirectional with the protocol working in reverse. (B sends data to A by pulling *devB* and so on.) For purposes of the lab, assume *devA* and *devB* are not asserted at the same time.

Implement the protocol utilizing the following inputs and outputs:

- *send* transaction will be initiated by KEY0 (and viewed on LEDR1)
- *devB* will be controlled by KEY1 (and viewed on LEDG7)
- *devA* will output on LEDR0

III. Pre-Lab

Complete the following steps before lab and turn in by 2pm:

1. Draw a timing diagram of the operation of the system.
2. Generate a state machine diagram from perspective of Device A. Make sure to indicate both transaction types (sending and receiving) in the same state machine diagram. It is your choice whether to use Mealy or Moore. Clearly indicate which you are designing and why.
3. Implement the design using behavioral VHDL. Synchronize the pushbutton inputs (ensure inputs last longer than the clock period). Decide whether you need the rising edge, falling edge, or level for your design and explain your decisions in your prelab.
4. Connect your design to the *testbench.vhd* (located in Sakai Resources), which will simulate the *devB* behavior. Review operation of *testbench* and notes at top of file on use.
5. Simulate your code using the waveform editor and functional simulation. Verify both transaction directions. Ensure simulation covers 75% or greater of the test cases and use simulation to verify VHDL matches your state machine.

Submit electronically via Sakai.

IV. In-Lab

Complete the following steps in lab:

1. Implement the VHDL design on the Cyclone II. You will need to divide the clock in order to see it function. Demonstrate for check-off.
2. Now, try to connect two boards. Recompile with *devA* and *devB* on the GPIO pins GPIO_0[0] and GPIO_0[1]. You may want to add an asynchronous reset pushbutton to insure everything can be placed in a known state.
3. Connect to your neighbors' system. DO NOT connect the cable straight across – the board has power and ground outputs on the cable which you would be connecting together (bad idea); use wires to connect cables. You should cross the wires so that your *devB* signal connects to your neighbors' *devA* signal and vice-versa. Make sure to connect the ground signals between the two boards; without a common ground, the digital abstraction falls apart and the devices do not recognize logic '0' and logic '1.' Demonstrate for check-off.
4. Reduce the clock frequency slightly for one of the devices. Does the system still function? Demonstrate for check-off.
5. Extra credit (3 lab points) – Figure out how to transmit a bit of data. Use a switch to input the data and an LED to output the data. Test on a single board first and then connect to your neighbor.

V. Post-Lab

Discuss the following your lab report:

- What happens in your design when both *devA* and *devB* are asserted at the same time? Test using the functional simulation (as well as in lab if you can). How might you modify the protocol to ensure no problems can occur with dual assertion?
- What happened when you modified the frequency of one device to differ from the other? Was this what you expected? Why or why not?
- How might you modify the system to communicate data? Outline the design changes and any assumptions you make. Modify your state diagram to include your design and discuss how you would modify the VHDL.