VHDL and Verilog Crash Course for FPGA firmware design

Matt Andrew, Luca Macchiarulo
IDLab - Dept. of Physics and Astronomy - University of Hawaii at Manoa

May 5, 2014

1 Homework No. 1

The purpose of the first day's exercise is to familiarize yourselves with the ISE environment and test your understanding of basic VHDL constructs.

2 Design Specification

Design 2 modules to implement a simple asynchronous serial communication. In particular you should design and describe in synthesizable VHDL:

- **Transmitter Tx**: a clocked module whose inputs are an 8 bit value and a transmit signal. When no signal is transmitted the output is '1'. When it receives a transmit signal (active high) it will start a sequence that will output: a start bit ('0'), the 8 bits of the value Most Significant bit first, and a stop bit. Each bit will be transmitted for 8 input clock cycles.
- **Receiver Rx**: a clocked module whose input is a single bit value. If the input bit makes a transition from 1 to 0, the Rx should successively decode the 9 bits, including the start bit, and after the correct reception of the stop bit, output a 8 bit value, together with a single clock flag indicating the presence of a new decoded data. Alternatively, any error of transmission (i.e. missed stop bit), will be indicated by an error flag, also active for a single cycle.

3 Procedure

Design your modules hierarchically, using the shift register seen in class or a variant (you will need a serial-to-parallel for the Rx). Your design will use a shift register and a state machine to coordinate the activity of the shift register. That is, to start and time the transmission in the case of Tx, and the recovery of full value or error notification in the case of the Rx.

4 Testing

Write a separate testbench for each of the 2 modules, and when you are confident they work correctly, use a single testbench to connect Tx and Rx and see if the results are correct (you will not be able to test communication errors in this case).

5 Extra activity

Try to implement a 9-bit protocol that uses the extra bit for a parity check (i.e. the ninth bit should be 1 if the parity is odd, so that the total parity is even: the Rx will signal an error also in case of a incorrect parity).