ECE 351- Spring 2020
Homework #4

Submit your deliverables to your Homework #4 dropbox by 10:00 PM on Wednesday 03-June-2020. THERE Will BE NO LATE ASSIGNMENTS ACCEPTED AFTER NOON ON THURSDAY, 04-JUNE

Source code should be structured and commented with meaningful variables. Include a header at the top of each file listing the author and a description. You may use the following template:

```
////////////////////////////////////////////////////////////
// <filename>.sv - <one line description>
//
// Author:  <your name> (<your email address>)
// Date:    <date you created the code>
//
// Description:
// ------------
// <text description of what function the module performs>
////////////////////////////////////////////////////////////
```

Files to submit:

- report.pdf or report.docx
- uart_tx.sv (your implementation of the UART Transmitter)
- source code for any starter code files that you changed
- You can submit all the design files but ensure that you use the given files and follow the naming conventions.

Note: the name of the SystemVerilog source code files should match the name of the module. The file should have a .sv extension for a SystemVerilog source code file or package.

Create a single .zip or .rar file containing your source code files and your report. Name the file <yourname>_hw4.zip (ex: rkravitz_hw4.zip).

Assignment (100 pts)

In this project, you are to complete the UART Transmitter RTL module and simulate the UART design discussed in class using the test bench provided. You are provided the starter code (/source directory) for the UART Design with internal modules like FIFO, Baud rate generator, the Receiver and incomplete code for the Transmitter.

Using the Receiver HDL code (`uart_rx.sv`) and the ASMD chart (`/docs` directory) as a reference, you need to produce an ASMD chart or a State Transition diagram (either of them is fine) for the UART Transmitter which will assist you in developing the Transmitter RTL module.

Deliverables

- Source code for your SystemVerilog module(s) and any others SystemVerilog files that you changed
- Project report.

Grading

You will be graded on the following for this project:

- Correct implementation of the UART Transmitter. [30 pts]
- Top level simulation results. [40 pts]
- Project report. [25 pts]
- The quality of your design expressed in your SystemVerilog source code. Your code should be cleanly structured and commented. [5 pts]

Tasks list

STEP 1: Download the project release package

Download and unzip `ece351sp20_hw4_release.zip` from D2L. This .zip file contains the design files and the testbench. A waveform configuration file (`wave.do`) for ModelSim or QuestaSim is also included. This file should work for your design as long as the signal names match.

STEP 2: Study/Understand the schematic and the HDL provided

Understand the working of UART design modules using the schematic and the HDL provided.

STEP 3: Complete the UART Transmitter RTL design

Using the ASMD chart or State Transition diagram you made for the UART Transmitter, complete the UART Transmitter RTL design module code building up on the starter code.

STEP 4: Study/Understand the testbench(es)

You are provided with two testbench modules. One is the UART unit level test bench (`uart_tb.sv`) which tests the UART design at (`uart.sv`) unit level. Another test bench is the UART Top level test bench (`uart_tb_top.sv`) which tests the UART design at (`uart_top.sv`) Top level.

At the top level, the UART DUT is instantiated into a UART Top module (`uart_top.sv`) with an I/O register model; this normally how the UART design is integrated into a system with a processor and other I/O devices.

Register Model Overview:

There are four mode registers for UART at the top level.

a. RD_DATA – to read data and status of UART.
b. WR_DVSR – to set the baud rate.
c. WR_UART – to perform a write/TX to UART.
d. RD_UART – to perform a read/RX from UART.

In the top level testing, we are configuring these registers to stimulate the UART design to test whether it is able to perform data transmission and reception correctly. Analyze the top level testbench to identify these register operations.

In the unit level testing for the UART, we can directly generate stimulus on the UART design signals to test whether the design is able to perform data transmission and reception correctly.

Both testbenches runs through all 128 character codes and performs a loopback test between the transmitter and the receiver. They also include checks to see if you are receiving correct data.

STEP 5: Test and debug your design at the unit level

After adding the transmitter code, test your UART design at the unit level using the unit level test bench (`uart_tb.sv`).

STEP 6: Test your design in an I/O register-based model

Once your design passes the unit level testing you can move on top level testing using the top level test bench (`uart_tb_top.sv`).

The `/sim` directory in the project release provides two waveform configuration files (wave.do) for both unit level and top level that should prove useful in debugging your design and submitting your results.

STEP 7: Document your work.

Your report must have following things:

- High level overview/introduction of the assignment.
- Top level black box diagram of UART which depict the top level inputs/outputs with a tabular short description of the design signals.
- UART block diagram.
- Waveform screenshot from the top level testing (unit level wave is not required, but you can add it)
- Transcript from the top level testing.
- Challenges faced and learnings from the project.
- Conclusion/Summary.