CS152A Lab 1 Workshop 2

In this session, you will focus more on extending the existing **testbench** that's provided to you.

Nicer UART Output

The existing testbench program contains a uart model that announces each byte that's received by the model. While generic, the output of this model is somewhat hard to read. Modify model_uart.v such that the per-byte output is surpressed. Instead, model_uart.v shall output one line at a time after a carriage-return character ('\r') is received. For example, print out "0003\n" instead of "0\n" "0\n" "0\n" and "3\n" for the first line of UART output.

Note: you are only allowed to use $display in this task (e.g. no $write is allowed). We suggest that read the UART protocol to further understand the test bench before start to work on the task.

An Easier Way to Load Sequencer Program

The existing sequencer testbench sends a static sequence of instructions to the UUT (Unit Under Test) after the reset. In the last session you have observed this in the waveform viewer how the instructions are sent. Answer the following questions in your lab report.

1. Identify the part of the tb.v where the instructions are sent to the UUT.
2. Which user tasks are called in this process?

Now we would like to change the static set of instructions. Instead, we will be loading instructions from a text file. The format of the file is the following:

1. The name of the file is "seq.code"
2. The file is up to 1024 lines long.
3. The first line of the file contains a binary number that indicates how many instructions are included in this file.
4. Each of the remaining (n-1) lines contains a single instruction in binary.

For example, here's the file-equivalent of the simple sequencer instructions currently in use:

Line 1: 1001
Line 2: 00000100
Line 3: 00000000
Line 4: 00010011
Line 5: 10000110

Line 6: 01100011
Line 7: 11000000
Line 8: 11010000
Line 9: 11100000
Line 10: 11110000

Modify the testbench such that it loads seq.code into an array, and executes every instruction in the file.

Hint: for file I/O, you may use the built-in Verilog system task $readmemb (google Verilog quick reference), or the c-like $fopen and $fscanf tasks.

Fibonacci Numbers

Now that you have an easier way to program the sequencer from simulation, design a sequence of instructions such that the first 10 numbers of the Fibonacci series is printed from the UART.