EL 6463: Advanced Hardware Design 2021

Mid-Term EXAM (20%) DUE: October 29th 2021, 2 PM EDT

Instructions

- Read and complete **all** questions carefully.
- This exam is to be completed **individually.** Collaboration/plagiarism will not be tolerated.
- Please sign the code of conduct (located at the bottom of this exam) and include it in your submission.
- Clarification questions can be posted on Slack in #midterm, but do not put any specifics about your own answers in the channel.
- During the exam period, you will not be allowed to receive assistance from the teaching staff regarding the use of the tools or implementation/simulation.

Overview: In this examination you will design, implement, and simulate digital hardware with sequential and combinational logic to implement a *pseudo-random number generator* algorithm. This algorithm will be implemented via hardware known as a *linear feedback shift register (LFSR).* These have applications in generating simple randomness in gaming, cryptography, audio and signal processing.

An example of a 16-bit LFSR is shown below:

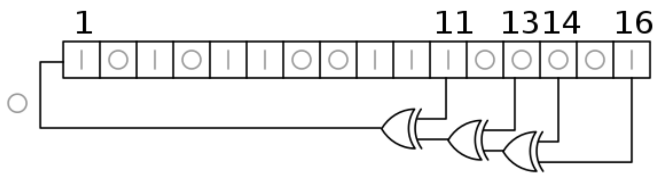

Image courtesy: CC0 https://en.wikipedia.org/wiki/File:LFSR-F16.svg

The depicted LFSR has XOR "taps" at positions [16, 14, 13, 11] (or zero-indexed: 15, 13, 12, 10).

In each clock cycle, the rightmost tap (which is known as the output bit) is XOR'd sequentially with the next right-most taps and the result is "shifted in" to the left-most bit (the most significant bit (MSB)).

In this mid term exam you will implement an LFSR with the following specifications (essential features).

1. It will be 32 bits long (from position 0 (MSB) to 31 (LSB)), with taps at positions as determined:
    a. Each of the unique adjacent 2-digit numbers in your N number modulo 32.
    b. The 32 bit (position 31, the LSB).
    c. For example, if your N number is 01123497, you will have taps at positions
        i. [01, 11, 12, 23, (34 % 32 =) 2, (49 % 32 =) 17, ~~(97 % 32 =) 1~~] *("1" value is not unique)*
        ii. = [1, 2, 11, 12, 17, 23, 31] *(31 comes from point 1b)* - remember these are zero-indexed
2. Top I/O:
    a. clk: input
    b. rstn: active-low reset signal
    c. data_out: 8 bit output
    d. get_random: input 1 bit control signal
3. Unless otherwise specified, the LFSR will update (shift-in the XOR result) upon every clock cycle. The bits shift into position 0 (the MSB).

4. If “output” is requested by setting “get_random” to 1, the least significant 8 bits will be output to “data_out” in the next clock cycle, the next 8 in the next clock cycle, the next 8 in the next cycle, and the most significant 8 bits will be output in the final (fourth) cycle. No LFSR shifts will occur during this output sequence. At all other times the “data_out” will be all zeros.
   NOTE:
   1. If the get_random signal is held high for longer than 4 cycles then the LFSR would presumably immediately begin exporting the contents again, and so it wouldn't perform any shifts, however there might be one clock cycle where the internals could shift. This part of the spec is upto to you.
   2. Similarly if get_random is high for less than 4 cycles, it should continue with the output until it is completed.
5. Reset signal “rstn” will reset the LFSR contents to 0x02468ACD
   (Reset signal “rstn” should always take precedence over get_random)

Questions

Problem A: Complete the following: [4 points total]

(1) **Draw** a high-level FSM + Datapath of your LFSR. **[1 point]**

(2) **Implement and simulate (functional only, no timing)** your design in your choice of Verilog or VHDL. Comment well. **[3 points]**

  (a) Your design should comprise a single "top" file and **instantiate multiple** components/modules

  (b) **Note**: As these have not been taught, **do not** use sub-programs, functions, or procedures.

  (c) **Write** a thorough testbench to demonstrate the capabilities of your design and to test **for meaningful corner cases**. Justify the corner cases.

Problem B: Complete the following: [4 points total]

(1) **Extend** your design to support the loading of custom **seeds** (a.k.a. start contents). **[3 points]**

  (d) Add a new 1-bit input “load_seed” and an 8-bit input “data_in”.

  (e) If “load_seed” is set to 1, the lower 8 bits will be loaded into the LFSR from “data_in” in the first clock cycle, the next 8 in the next, the next 8 in the next, and the upper 8 bits will be loaded in the final (fourth) cycle. No LFSR shifts will occur during loading. After loading, shifts will resume as normal.

  (f) **Extend your testbench** to test this new functionality.

(2) **Write** a short (2 pages max.) report describing your design process. Justify your code and relate it to your desired FSM + datapath design. Justify your testbench (where did test values come from?), and describe how it covers all possible execution scenarios. **[1 point]**

Problem C: Short Answer (~100 words each) Questions: [2 points total]

(1) Does your design use asynchronous or synchronous resets? Why did you choose one over the other? What are the benefits of using each reset style? **[0.4 points]**

(2) Why is it necessary to use registers in a digital design? How many total register bits did you use in your design? **[0.4 points]**

(3) Research and explain in your own words one real-world application of an LFSR. Provide sources. **[0.6 points]**

(4) Assuming the seed is the specified reset value (0x02468ACD), how many cycles will it take for your LFSR to start repeating values? You may determine this mathematically, via simulation using the testbench, or via any other method. **Explain the method.** Is this period “good” for a pseudo-random number generator? How could you make the period longer? **[0.6 points]**

Deliverables:
Submit a zip file with your VHDL or Verilog files and Xilinx Project and a PDF report containing your short report and answers to the short answer questions.

After the submissions, you **must** be able to answer questions about your design in a live interview with a member of the teaching team. Failure to answer questions about your own design can be treated as proof of copying / plagiarism. Details TBA.

Student Exam Code of Conduct

I certify and affirm that,

1. I am aware of the NYU Code of Conduct.
2. Work presented is 100% my own. I have not collaborated with anyone on the test.
3. I did not misrepresent someone else's work as my own.
4. I did not discuss, nor in any way divulge the content of this test or my answers.
5. I understand that my failure to abide by this code of conduct will result in consequences outlined in the NYU Code of Conduct.

Name: ___________________________________

Signature: ________________________________

Date: ____________________________________