EL 6463: Advanced Hardware Design 2022

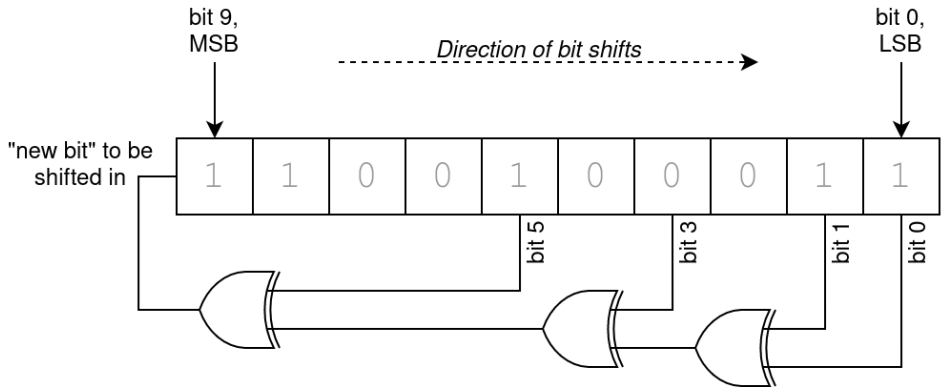
Mid-Term EXAM (20%) IN-CLASS: October 21th 2022, 2:05-4:35 PM EDT

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.
- Do not use the internet for anything other than connecting to the "hansolo" server.
- You may examine your own notes from class and the class slides.
- Clarification questions can be asked to a member of the teaching team. Do not connect to the class slack during the exam.
- 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 an LFSR in the style used for this exam is shown here:



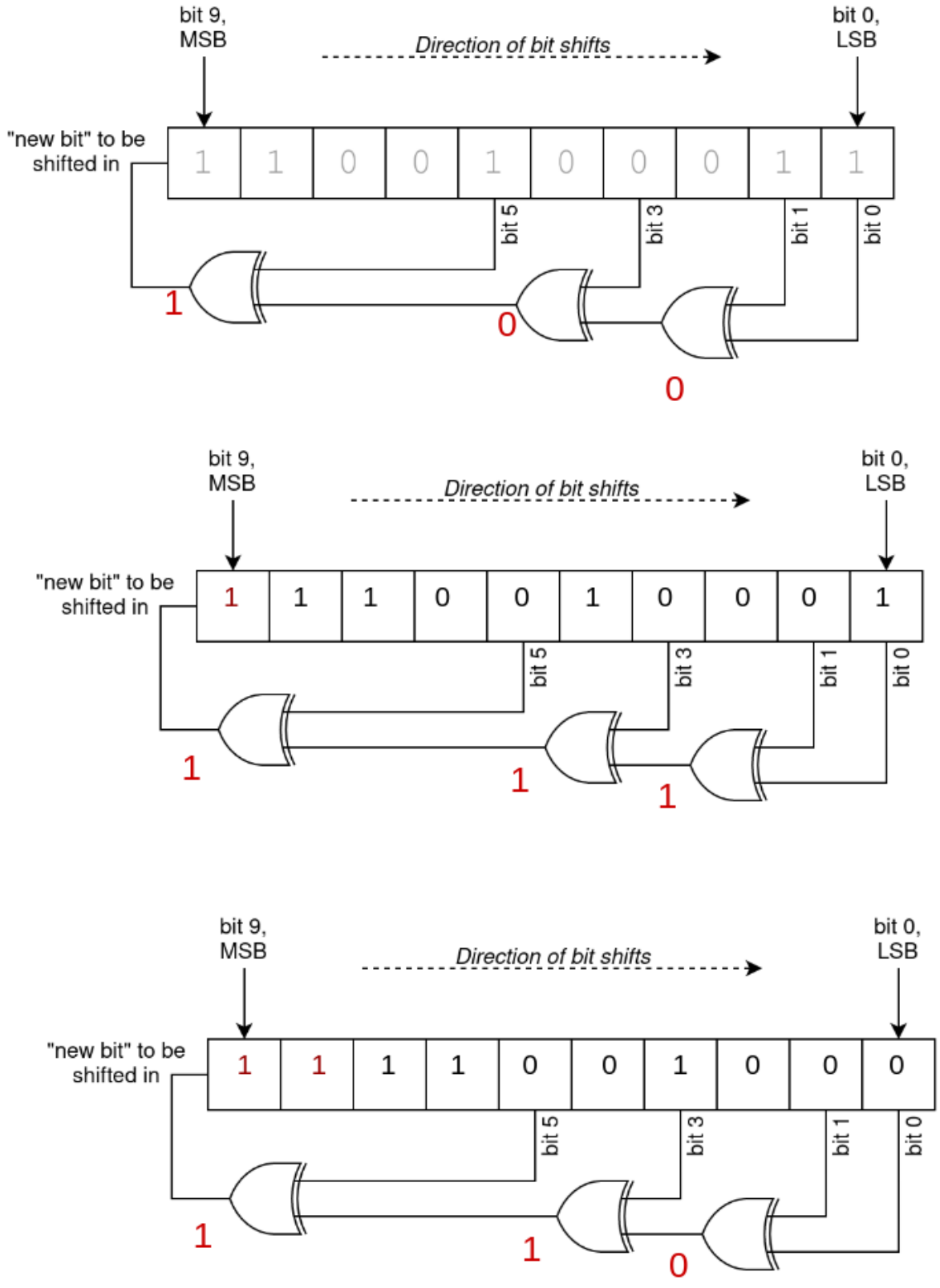
The depicted LFSR has XOR "taps" at positions [5, 3, 1, 0].

In each clock cycle, the rightmost tap 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)).

Here is the overall block diagram of the LFSR:

It has inputs "en" (enable; advances the LFSR), "rst" (reset; sets it back to initial value), "clk" (clock). en/rst are active high.
It has output "data", which is the 10-bit current value of the LFSR.

The following image shows the evolution for 2 clock cycles of the LFSR reported above:



TASK: In this mid term exam you will implement an LFSR according to the above, with the following additional specifications (essential features).

1. It will be 10 bits long (from position 9 (MSB) to 0 (LSB)), with 4 taps at positions as determined by the value of your N number according to the following:
    a. Take your N number, e.g. 35108323
    b. Take the first 4 digits, e.g. 3510
    c. Order them high->low, e.g. 5, 3, 1, 0
    d. These are your 4 taps e.g. see the first figure in the Overview.
    e. NOTE: If you have repeated digits, increment them until they are unique.
    e.g. 1122 becomes 1324
    e.g. 8899 becomes 8901
2. The initial/reset value of the LFSR will also be based on your N number
    a. Take your N number, e.g. 35108323
    b. Take the last three digits, e.g. 3, 2, 3
    c. Express these as BCD, e.g. 0011, 0010, 0011
    d. Take the right-most ten values, e.g. 11 0010 0011
    e. These are the ten bits defining the initial/reset value of your LFSR in order from MSB to LSB. e.g. see the first figure in the Overview.
3. The overall LFSR will have the I/O as defined in the Overview second figure, with:
    a. clk: input
    b. en: active-high enable signal (advances the LFSR)
    c. rst: active-high reset signal
        i. It is "your choice" if rst is a synchronous or asynchronous reset.
    d. data: 10 bit output
4. The LFSR shall advance (shift-in the XOR result) upon every clock cycle during which the active-high enable signal is present. The bits shift into position 9 (the MSB).
5. Reset signal "rst" will reset the LFSR contents to your initial value. It should take precedence over the enable signal.

Questions

[8 points] Problem A: Coding. Complete the following:

(1) **[4 points] Implement** your design in your choice of Verilog or VHDL. Comment well.

(a) Your design may be one module or multiple modules.

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

(2) **[4 points] Simulate (functional simulation only) your design** by creating a thorough testbench (in your choice of Verilog or VHDL) to demonstrate the capabilities of your design and to test **for meaningful corner cases**.

[2 points] Problem B: Reporting. Complete the following short answer questions:

(1) **Write** a short (lengths noted) README containing the following:

(a) **[500 words max, 1 point]** A description of your project, including the essential details of the code (i.e., a specification, including architectural, of what you built). Describe the testbench and justify the corner cases (i.e. convince us that it tests all functionality).

(b) **[150 words max, 0.5 points]** In a subsection, note whether or not you used asynchronous or synchronous resets, and justify your choice (why is one style better than the other).

(c) **[100 words max, 0.5 points]** Using your test-bench or any other method, determine how many cycles it will take for your LFSR to start repeating values. Report this value and explain how you determined it.

(d) [COMPULSORY!] Affirm the student code of conduct in your README.

Deliverables:
Submit a single zip file containing your VHDL or Verilog files, your complete Xilinx Project, and a README containing the short answer questions from Problem B **and your affirmation to the student code of conduct.**

COPY + COMPLETE THE FOLLOWING IN YOUR README.

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 (e.g. from the internet) 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: [type name]

Date: [type date]