CSE 140L Lab 5

(Amy Nguyen A16125627); (Mark Lorenzo, A15955698); (Jared Villanueva, A15821317);

Academic Integrity

Your work will not be graded unless the signatures of all members of the group are present beneath the honor code.

To uphold academic integrity, students shall:

- Complete and submit academic work that is their own and that is an honest and fair representation of their knowledge and abilities at the time of submission.
- Know and follow the standards of CSE 140L and UCSD.

Please sign (type) your name(s) below the following statement:

I pledge to be fair to my classmates and instructors by completing all of my academic work with integrity. This means that I will respect the standards set by the instructor and institution, be responsible for the consequences of my choices, honestly represent my knowledge and abilities, and be a community member that others can trust to do the right thing even when no one is watching. I will always put learning before grades, and integrity before performance. I pledge to excel with integrity.

(Amy Nguyen)
(Mark Lorenzo)
(Jared Villanueva)

Free Response

Please answer the following questions.

1. Please describe, at a high level, how each stage of decryption using LFSR works. (4 pts)

Word limit: 200 words.
The stages we are looking for you to describe are the preamble stage, training stage, and decryption stage.

In the preamble stage, we XOR the data read from memory with the current LFSR state (index "foundit" = 0) to undo the original XOR and get back the original hexadecimal underscore.
In the training stage, we loop through the different LFSR states and find one that matches, which sets "foundit" to a decimal value that encodes the tap position that was active.
In the decryption stage, using "foundit" as an index, we XOR the current LFSR state with the data read from memory to undo the original encrypting XOR and get back the original message.

2. Why do we use 6 LFSRs in our top level module? (4 pts)

Word limit: 200 words.

There are 6 possible maximal-length feedback tap patterns that we can choose from. The LSFR can reach all 2^6-1 (63) nonzero states since they can be made from any of the six different sets of taps. There are 63 nonzero states since we're restricting ourselves to ASCII values from 0x40 to 0x7F so that our message can be printable.

Words: 59

3. Please explain how the testbench tests your design. (4 pts)

Word limit: 200 words.
You can briefly describe the major steps taken by the testbench.

The testbench first sets a preamble length of 7, and then proceeds to select tap pattern 2 and set the LFSR starting state. It then displays the original message without, then with preamble padding and some stats about it. It then encrypts the original message and displays it. Following this, the testbench displays the original message again but this time in hexadecimal. The write enable is set to on and the encrypted message is written to memory according to our top level. The testbench then displays each memory element we wrote alongside the original message in hex. We are then able to visually compare to see if what we wrote matches the original message in hex. The testbench also compares the memory elements we wrote to the

original message in hex and increments a fault counter that displays how many discrepancies there are between our result and the intended result.

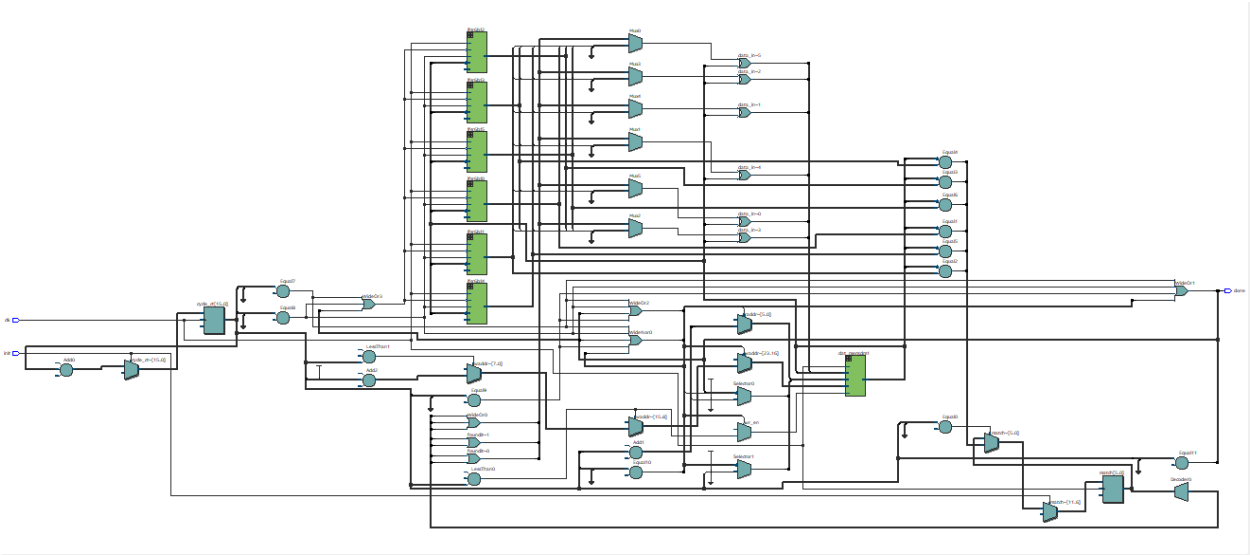
4. In our implementation, the preamble is an underscore character. Is it possible to decrypt messages with an unknown preamble character? How would you edit your decryption design to accommodate for different character preambles? (3 pts)

Word limit: 200 words.
Please answer both questions. For the second question, just explain at a high level what you would do. No need to give us extra code for this question. Unknown means that the receiver does not know what the preamble character is.

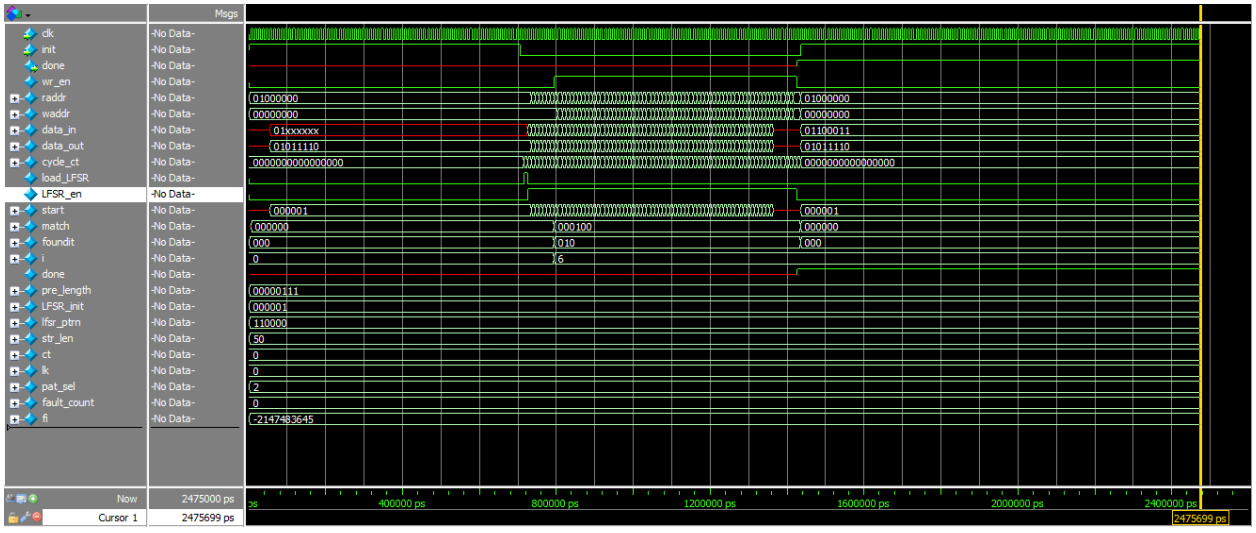
It is not possible to decrypt messages with an unknown preamble character, it's only because it is known that messages are decryptable. If a different known preamble character was used, like instead of an underscore it was a hyphen, then I would change what I XOR in the decryption stage for prelen cycles to XOR with a hexadecimal hyphen instead of an underscore. Of course, the XOR to read from the start of the encrypted message and decrypt the message would remain unchanged.

Screenshots

Screenshot of the RTL viewer top level schematic/block diagram in Quartus (5 pts)



Screenshot of your waveform viewer, including variables from both the testbench and the design under test. (5 pts)



Please edit your testbench to pipe the transcript to an output file in your submission, and name the output file “output.txt” (5 pts)

We will be looking for a text file with that name specifically, so be sure to rename it. Nothing is required in the writeup for this question.