CSE 140L Lab 4

(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 explain, at a high level, how encryption using a LFSR works. (4 pts)

Word limit: 200 words

A linear feedback shift register is a register of bits that shift all of the bits one position to the left, and replaces the vacated bit by the XOR of the bit shifted off and the bit at the TAP position in the register. Because of the tap position, we are able to generate a random sequence of bits that can be used to encrypt each ASCII character. Without knowing the initial sequence of bits in the LSFR and the tap, the encryption can not be decrypted by hand. The receiver is also able to figure out the starting point because of the preamble, which is a known string that we include at the beginning of our encrypted sequence.

Words: 118

2. Please explain, in detail, the behavior of a LFSR. (4 pts)

Word limit: 200 words. Think about the input, output, timing, and behavior of the module.

In our case, we implemented a 6 tap LFSR. An example of a visualization of a 6 bit LSFR is one with the 0 bit on the left, and the 5 bit on the right. The bits in the LFSR state that effect the input are called taps. As with any LFSR, the input is the bits inside of the LSFR where two of the bits will be XOR-ed with one another and fed back into the first bit. The output is the bits in the LFSR after each operation. For timing, an LFSR is initialized with values one register at a time since as the data from the xor comes into the first register, the rest of the registers are still X, as the data has not propagated to the 2nd, let alone 3rd, 4th etc. registers. Once all the registers are filled one by one every clock cycle, then the data essentially “shifts” one register every clock cycle.

3. How did you manipulate your address pointers (both read and write) to implement your design? (4 pts)

Word limit: 200 words

In the default case that covers clock cycles 6 to 72, I incremented the only write address when writing the preamble xor bits, since the values to xor were the current LFSR state and a hexadecimal underscore. After writing the preamble bits, I incremented the read address along with the write address, taking care to “read the current value as I write the previous”. Both read and write address increments are based on offsets of the clock cycle counter since the clock cycle counter is a stable and known increment, so offset increments should also be known and stable for easier debugging via the waveform.

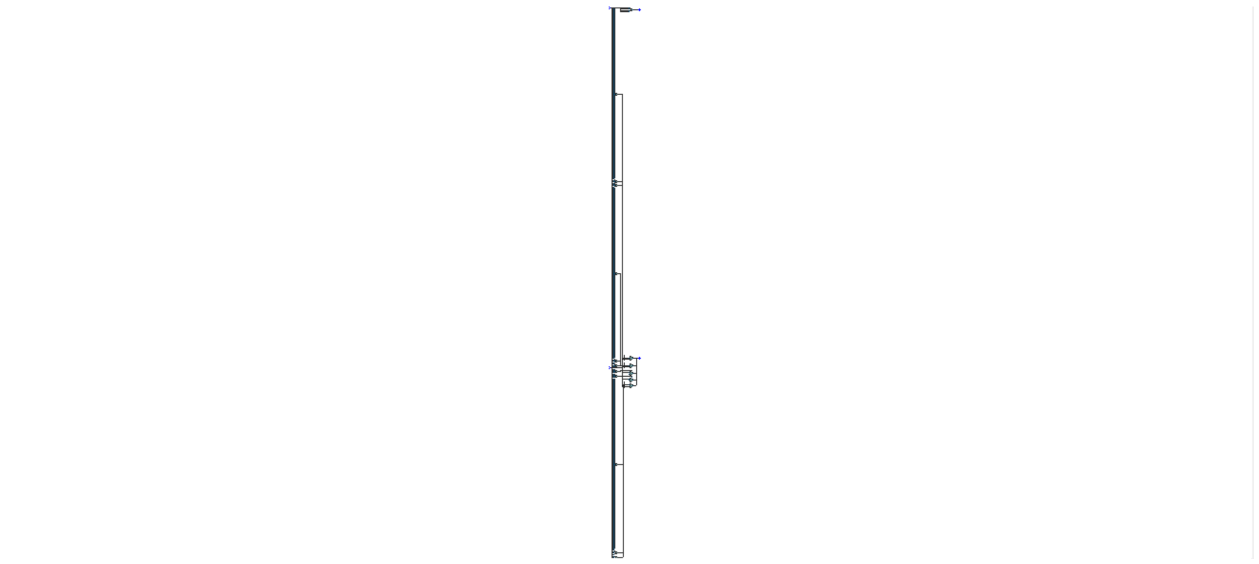
4. What is the purpose of making our messages have a preamble in our design? (3 pts)

Word limit: 200 words
If we provide a known sequence of data (in our case the hexadecimal underscore) that has been encrypted by our 6-tap LFSR, then we can figure out the specific tap pattern used when we decrypt in no more in 6 transitions (7 clock cycles).

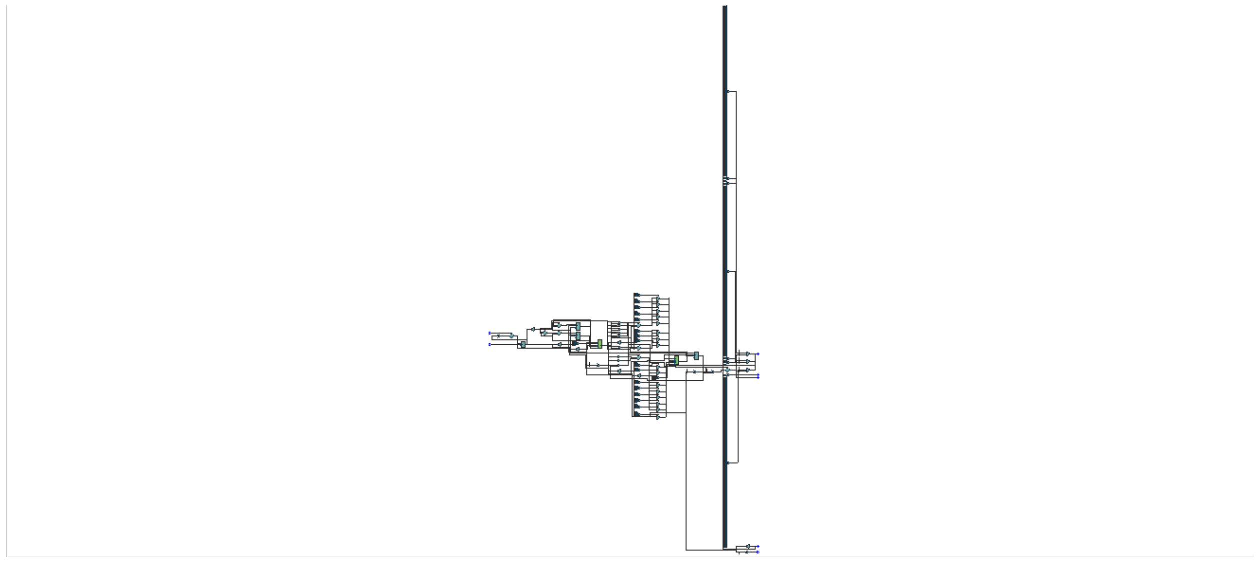
Screenshots

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

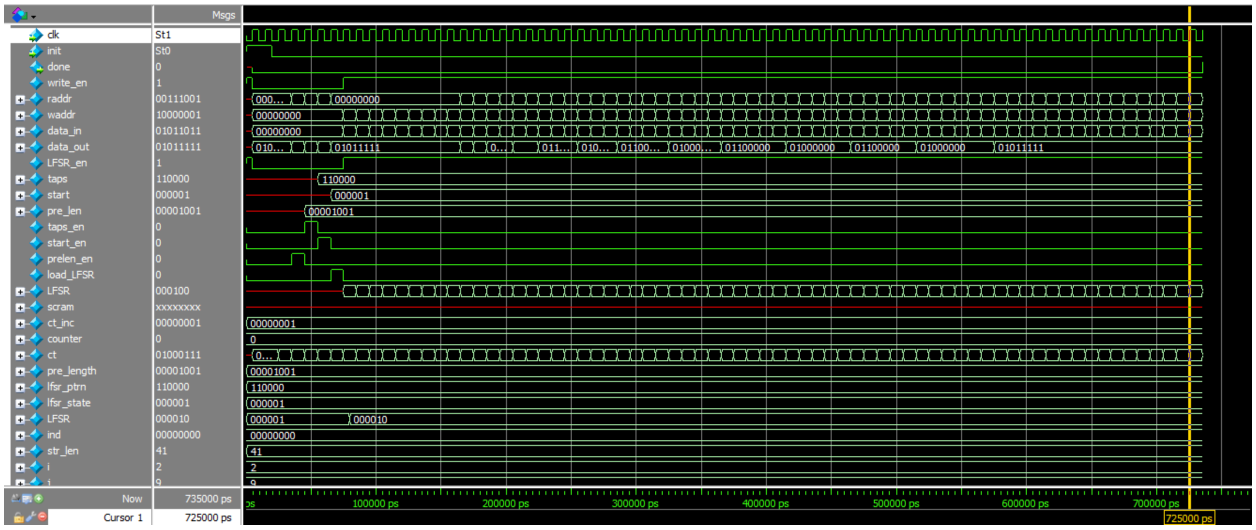
Page1



Page2



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



	Msgs
clk	St1
init	St0
done	0
write_en	1
raddr	00111001
waddr	10000001
data_in	01011011
data_out	01011111
LFSR_en	1
taps	110000
start	000001
pre_len	00001001
taps_en	0
start_en	0
prelen_en	0
load_LFSR	0
LFSR	000100
scram	xxxxxxxx
ct_inc	00000001
counter	0
ct	01000111
pre_length	00001001
lfsr_ptrn	110000
lfsr_state	000001
LFSR	000010
ind	00000000
str_len	41
i	2
i	9
Now	735000 ps
Cursor 1	725000 ps

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.