Midterm Exam

ECE 448
Spring 2010

Wednesday Section

(15 points)

Instructions

Please read this entire document carefully before beginning!

Zip all your deliverables into an archive <last_name>.zip and submit it through Blackboard no later than Wednesday, March 17, 10:15 PM EST.

Introduction

Your task is to describe in VHDL, debug, and implement a simple stream cipher based on (but not exactly equivalent to) GRAIN-128.

The operation of this cipher is described below using the following pseudocode, interface, and the block diagram.

Pseudocode:

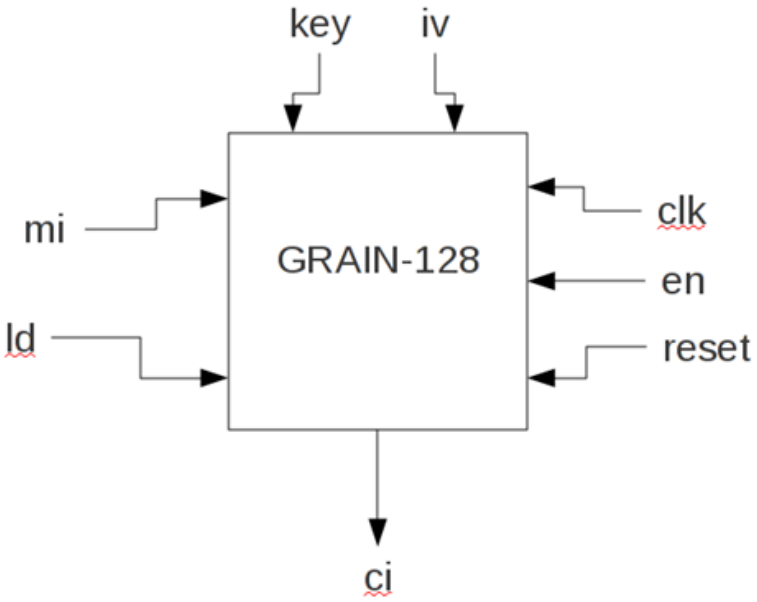
```
when ld = 1 and rising_edge(clk)
{
        NFSR = key;
        LFSR = iv;
}

for i=1 to 20
{
 hi = NFSR12LFSR8 ⊕ LFSR13LFSR20 ⊕ NFSR95LFSR42 ⊕
      ⊕ LFSR60LFSR79 ⊕ NFSR12NFSR95LFSR95
 ki = hi ⊕ NFSR2 ⊕ NFSR15 ⊕ NFSR36 ⊕ NFSR45 ⊕ NFSR64 ⊕
      ⊕ NFSR73 ⊕ NFSR89 ⊕ LFSR93
 ci = mi ⊕ ki
 lfsr_in = LFSR0 ⊕ LFSR7 ⊕ LFSR38 ⊕ LFSR70 ⊕ LFSR81 ⊕ LFSR96
 nfsr_in = LFSR0 ⊕ NFSR0 ⊕ NFSR26 ⊕ NFSR56 ⊕ NFSR91 ⊕
           ⊕ NFSR96 ⊕ NFSR3NFSR67 ⊕ NFSR11NFSR13 ⊕
           ⊕ NFSR17NFSR18 ⊕ NFSR27NFSR59 ⊕ NFSR40NFSR48 ⊕
           ⊕ NFSR61NFSR65 ⊕ NFSR68NFSR84
 NFSR = (NFSR << 1) ||  nfsr_in
 LFSR = (LFSR << 1) ||  lfsr_in
}
```

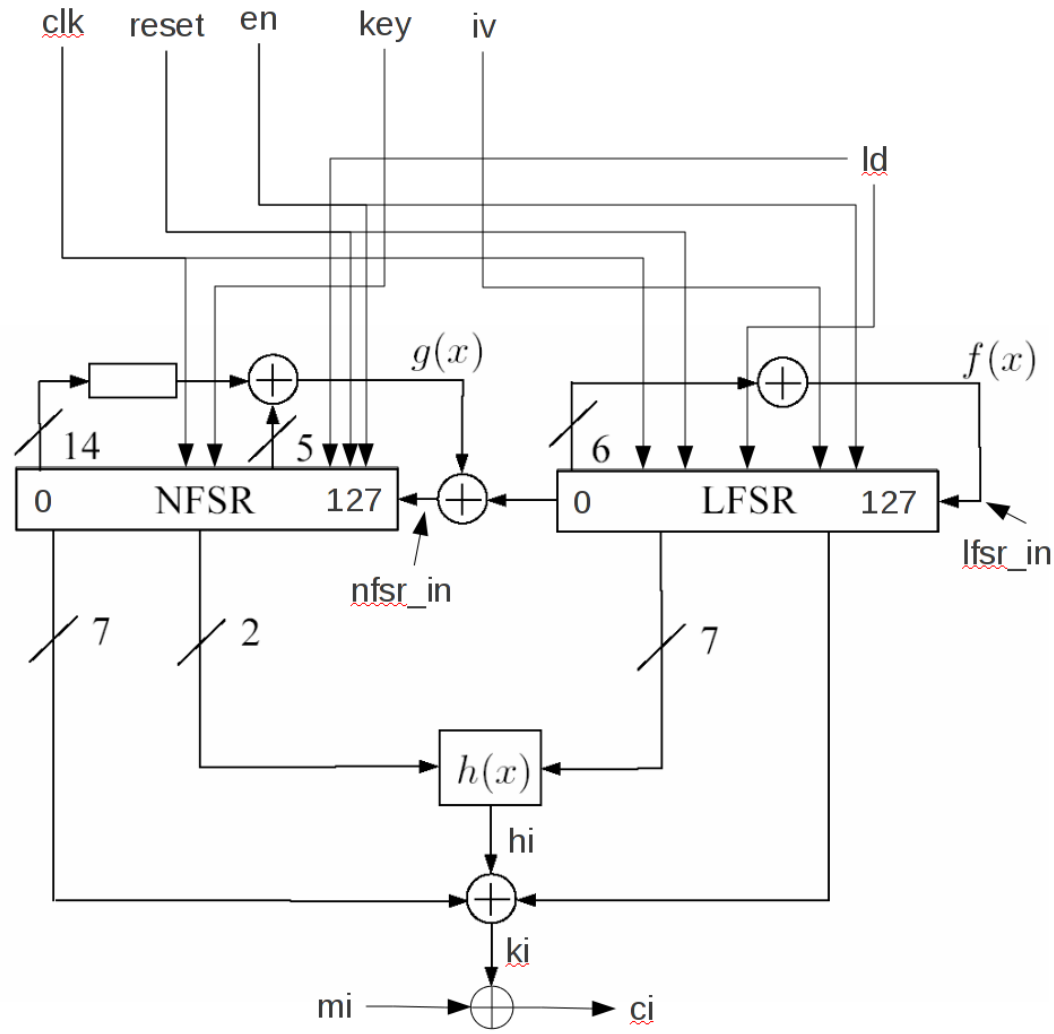
Notation:
A || B : A concatenated with B
<< 1 : logical shift by one positions to the left
⊕: exclusive-or (xor)
$NFSR_i$: i-th bit of NFSR
$LFSR_i$: i-th bit of LFSR
NFSR = $NFSR_0$ || $NFSR_1$ || $NFSR_2$ || … || $NFSR_{127}$
LFSR = $LFSR_0$ || $LFSR_1$ || $LFSR_2$ || … || $LFSR_{127}$
$NFSR_i$ $NFSR_j$: $NFSR_i$ AND $NFSR_j$

Interface:

Port	Mode	Width	Function
clk	Input	1	System clock.
reset	Input	1	System reset.
key	Input	128	The key of the cipher (initial value of NFSR)
iv	Input	128	The initialization vector of the cipher (initial value of LFSR)
mi	Input	1	Message input.
ld	Input	1	Control signal indicating loading KEY and IV.
en	Input	1	Control signal indicating encryption phase. Enable signal for LFSR and NFSR.
ci	Output	1	Ciphertext (encrypted message) output.

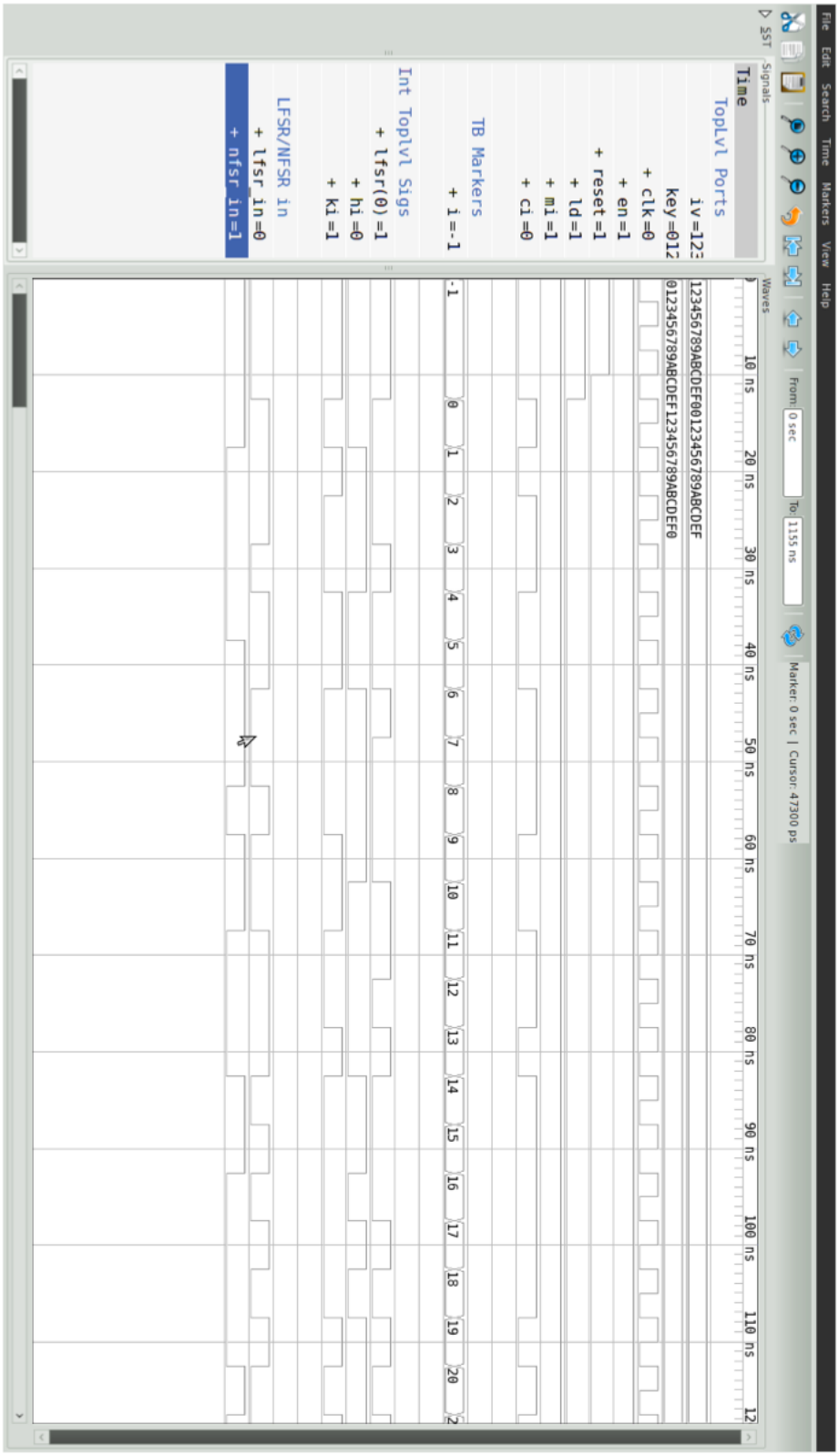


Block diagram:



The timing diagrams below show the operation of the circuit for the following values of inputs:

```
key = 0x0123456789abcdef123456789abcdef0
iv =  0x123456789abcdef00123456789abcdef
mi =  all 1's
```



The output for 20 bits of ciphertext output should be
$ci_{0\text{-}19}$ = 0xB39BE

Design Requirements

The combinational portion of the circuit should be described using the dataflow VHDL code, and the sequential portion of the circuit should be described using the synthesizable behavioral code. Your code should infer a circuit that requires a minimum amount of FPGA resources. The target clock frequency should be 100 MHz.

Tasks
Perform the following tasks:
1. Write a synthesizable VHDL code representing the described above circuit.
2. Write a testbench verifying the operation of your circuit for inputs shown on the waveforms above.
3. Perform functional simulation of your circuit and use it to debug your VHDL code. Save the obtained timing waveforms.
4. Synthesize your circuit. Save the obtained RTL schematic.
5. Perform post-synthesis simulations of your circuit. Save the obtained timing waveforms.
6. Implement your circuit using
   FPGA family: Spartan3E,
   Device: XC3S1600E
   Speed Grade: -4.
7. Run the static timing analysis of your circuit.
8. Based on the circuit block diagram and the implementation reports, determine the most critical path in your circuit and the circuit maximum clock frequency.
9. Based on the implementation reports and the report from the static timing analysis, determine the number of CLB slices, Logic Cells, LUTs, D flip-flops, and pins used by the circuit.

Deliverables
1. VHDL code of your entire circuit fulfilling the requirements specified in the Design Requirements section above.
2. VHDL code of your testbench.
3. RTL schematic of your circuit.
4. Timing waveforms from the functional and post-synthesis simulations demonstrating the correct operation of your circuit.
5. Description of the critical path of your circuit.
6. FPGA resource utilization (as defined in Task 9 above)
7. Minimum clock period and maximum clock frequency of your circuit.