

Practice Hands-on Midterm Exam

**ECE 448
Spring 2006**

All Lab Sections

5 bonus points

Instructions:

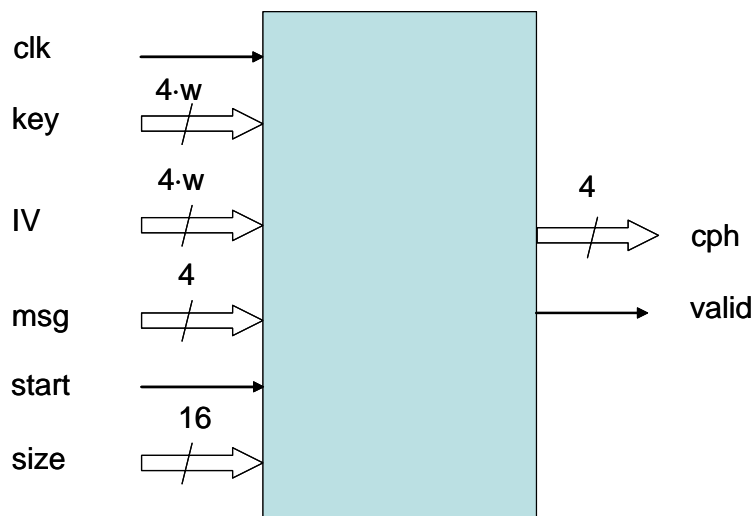
Zip all your deliverables into an archive <last_name>.zip and submit it through WebCT no later than Friday, March 10, 11:55 PM EST.

Hands-on Design Problem

Introduction

The described below circuit performs encryption of N-bit binary messages (i.e., N-bit sequences of zeros and ones obtained by encoding messages in ASCII code), using a simple algorithm based on XOR operations, additions and shifts. Messages are entered to the circuit in blocks of 4 bits, and the corresponding ciphertexts leave the circuit in blocks of 4 bits. The result of encryption depends on two input parameters, key (which needs to be kept secret) and IV (initialization vector, which can be chosen at random on the sender's side and transmitted in clear to the receiver's side).

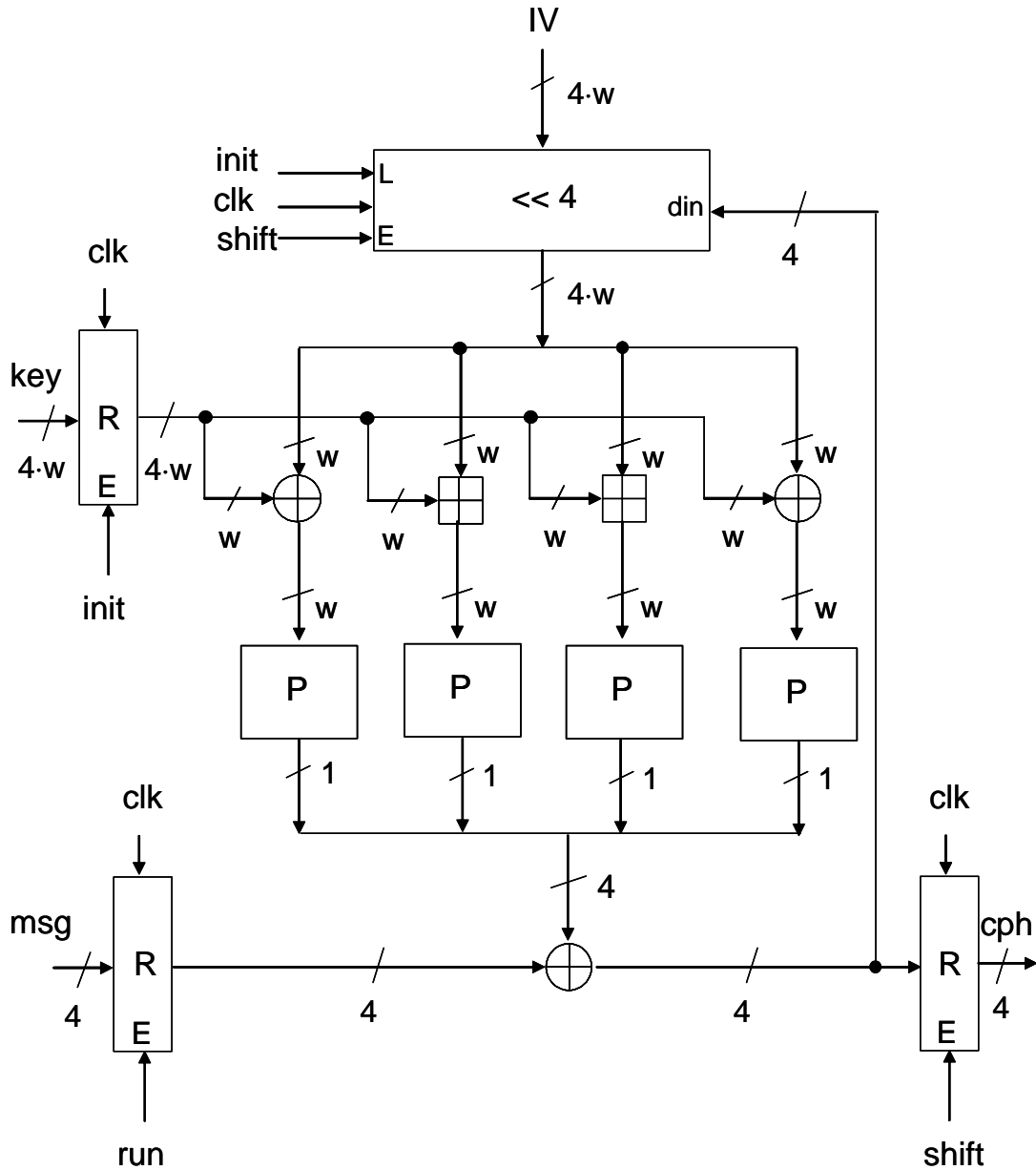
The interface of the circuit is given below:



<i>Port</i>	<i>Width (bits)</i>	<i>Mode</i>	<i>Function</i>
<i>key</i>	4-w	IN	Cipher key
<i>IV</i>	4-w	IN	Initialization vector
<i>msg</i>	4	IN	Message block before encryption
<i>cph</i>	4	OUT	Ciphertext block after encryption
<i>start</i>	1	IN	Control signal equal to one in the first clock cycle of computations for a given message (see timing waveforms below)
<i>size</i>	16	IN	Size N (in bits) of the message to be encrypted (see timing waveforms below)
<i>clk</i>	1	IN	Master clock
<i>valid</i>	1	OUT	Ciphertext output valid (equal to 1 when output cph contains a ciphertext block)

Assume the default value of parameter $w = 16$.

The execution unit of the circuit is described using the following block diagram:



Notation:

\oplus denotes bitwise XOR

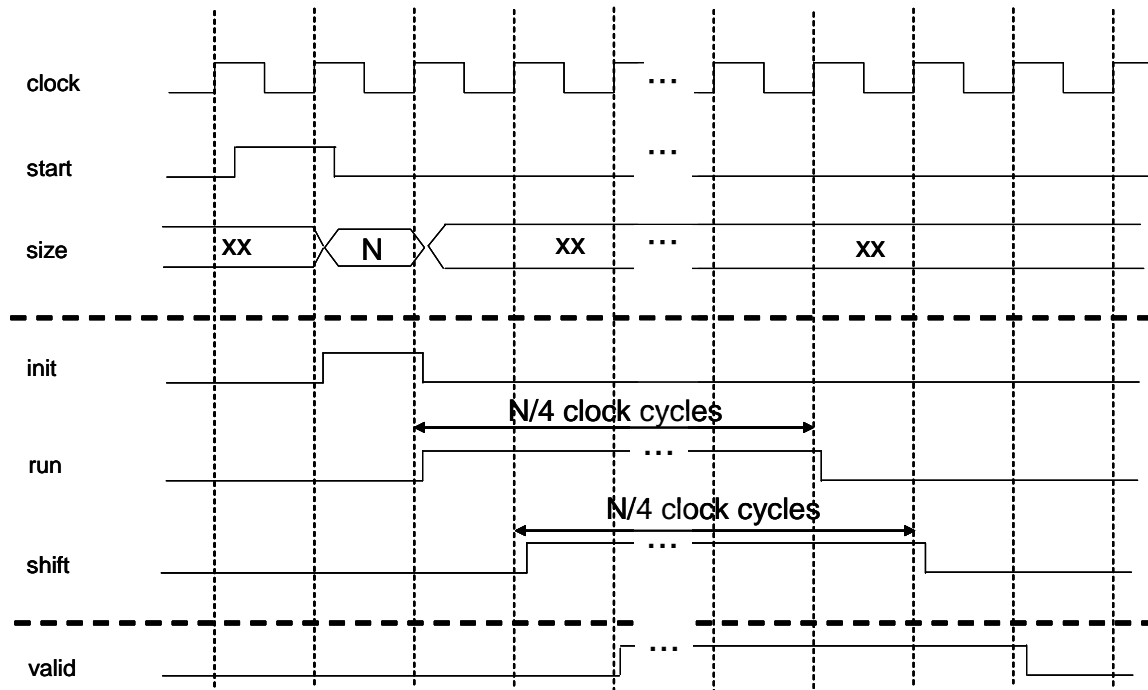
\boxplus denotes addition modulo 2^w , i.e., addition with carry out discarded

P denotes odd parity generator, i.e., the circuit that generates 1 when an odd number of its inputs is equal to 1

R represents a register with an enable input E

<< 4 represents a shift register with a parallel load signal L, enable signal E, and a digit-serial input din

Please note that the internal control input signals `init`, `run`, and `shift`, and the control output signal `valid` are all generated by the control unit of the encryption circuit using external control signals `start` and `size`, as shown in the timing waveforms below.



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 50 MHz.

Tasks

Perform the following tasks:

1. Write a VHDL code of the execution unit of the encryption circuit (shown in the block diagram above).
2. Write a testbench verifying the operation of your execution unit for the case of $w=8$.
3. Perform functional simulation of your circuit and use it to debug your VHDL code.
4. Design a control unit of your circuit. If you do not know how to do it, go to Step 6.
5. Write a testbench verifying the operation of your entire circuit (including the control unit) for the case of $w=8$, and message size $N = 64$.

6. Synthesize your circuit using Synplify Pro the case of $w=16$. Save the RTL schematic.
7. Implement your circuit using Xilinx ISE.
8. Perform timing simulations of your circuit using Active-HDL.
9. Run the static timing analysis of your circuit.
10. Based on the circuit block diagram and the implementation reports, determine the most critical path in your circuit and its length.

Deliverables

1. VHDL code of your entire circuit fulfilling the requirements specified in the *Design Requirements* section above
2. VHDL code of your testbenches
3. RTL schematic of your circuit
4. Timing waveforms from the functional simulation demonstrating the correct operation of your circuit.
5. Description of the critical path in your circuit
6. Timing waveforms from the timing simulation demonstrating the delay of the circuit most critical path
7. FPGA resource utilization
8. Minimum clock period of your circuit.