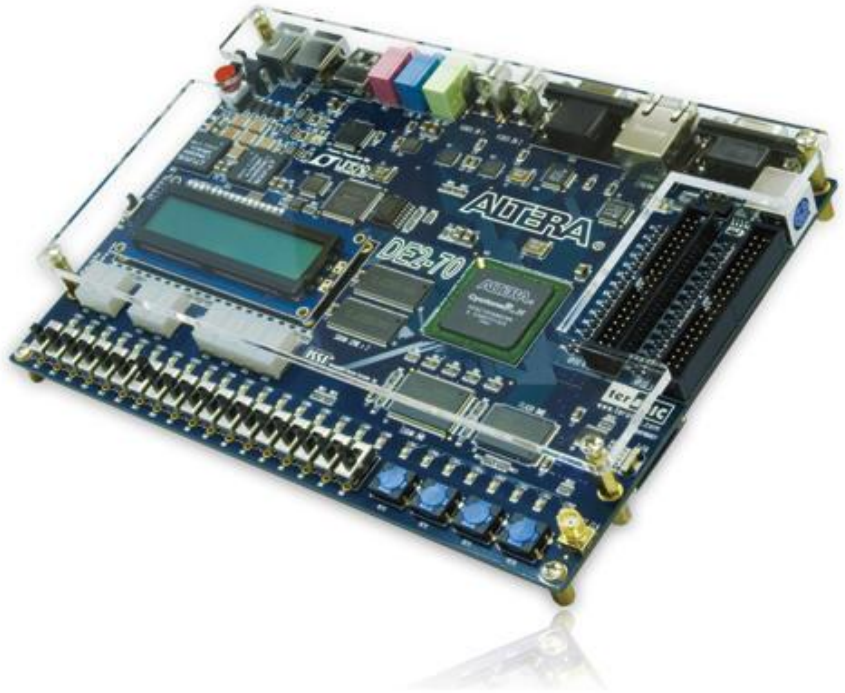
University of Jordan
Faculty of Engineering and Technology
Department of Computer Engineering
Digital Logic Laboratory 0907234

Labsheet8: Shift Register & Counter Design



Name:

Student ID:

Section:

Description

In this experiment, you will implement the circuit shown in Figure 1. The circuit consists of two main components: a 3-bit synchronous counter and a 3-bit shift register. Note that the clock input of the counter is obtained from the FPGA oscillator (after being slowed down by dividing the frequency). The clock of the 3-bit register, on the other hand, is obtained from the ***zero_count*** output of the 3-bit counter, which is 0 for all count values and 1 for count 0 only.

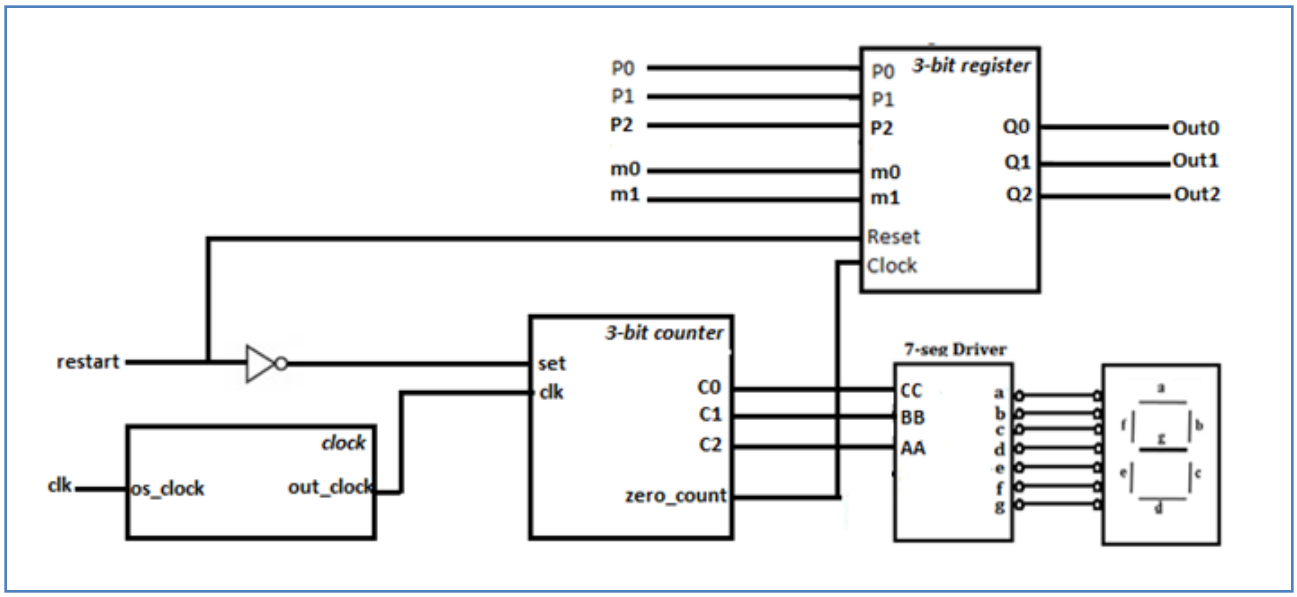


Figure 1.Counter-controlled Register Block Diagram

Part 1: 3-bit Counter Design:

In this part, you are required to design a (count-down) 3-bit counter that counts 7, 6, 5, 4, 3, 2, 1, 0, 7, 6, 5, 4,...., using D flip flops.

1. Fill the following state table according to the required counter design.

Present State			Next State					
Q2	Q1	Q0	Q2	Q1	Q0	D2	D1	D0
0	0	0						
0	0	1						
0	1	0						
0	1	1						
1	0	0						
1	0	1						
1	1	0						
1	1	1						

2. Write the input equations of the three flip flops.

D0=

D1=

D2 =

3. Draw the sequential circuit that implements the 3-bit counter. Then, add to your circuit the gate(s) required in order to generate an output signal called ***zero_count*** which is one only when the count is 0.

4. In the file ***'threebitcounter.bdf'***, build the schematic diagram of the sequential circuit you designed by following these steps:
   a. Add the gate(s) and D flip-flop symbols to your bdf file and make the required connections according to your design.

   Note: In order to add the D flip-flop symbol to your bdf file, select the symbol of AND gate on the tool bar (from where you get the ICs usually) then select primitives→storage→dff.

   b. Add the following input pins to your circuit:
   1. ***clk***: which is connected to the clock inputs of all flip flops.
   2. ***set***: which is connected to the set inputs (***PRN***) of all flip flops.

   c. Add the following output pins to your circuit:
   1. ***C2, C1, C0***: which represents the current count and corresponds to the outputs of the flip flops (Q2, Q1, Q0).
   2. ***Zero_count***: which indicates that zero count is reached.

   Note that the reset (***CLRN***) inputs of the flip flops should be deactivated by connecting them to ***VCC***.

5. Convert the 3-bit counter schematic diagram you built to a block as in Figure 2. This block will be used in the final circuit implementation in part 3.
   File → Create/Update →Create Symbol Files for Current Files

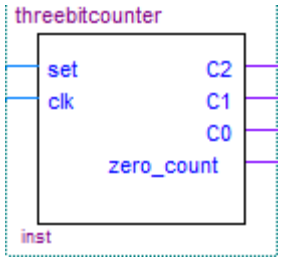


Figure 2. Block symbol for the three-bit counter.

Part 2: 3-bit Shift-Register:

1. In the file ***reg3b.v***, write a Verilog module that implements the 3-bit register shown in Figure 3 structurally.

Note: You will need to use the 4-to-1 MUX module implemented in the file ***mux1.v*** and the D-flip flop with reset module, which you implemented in experiment 7, in the file ***dff1.v***.

Note that the register has 4 modes of operation according to the following table:

m1	m0	Operation
0	0	Hold
0	1	Parallel Load
1	0	Rotate Right
1	1	Rotate Left

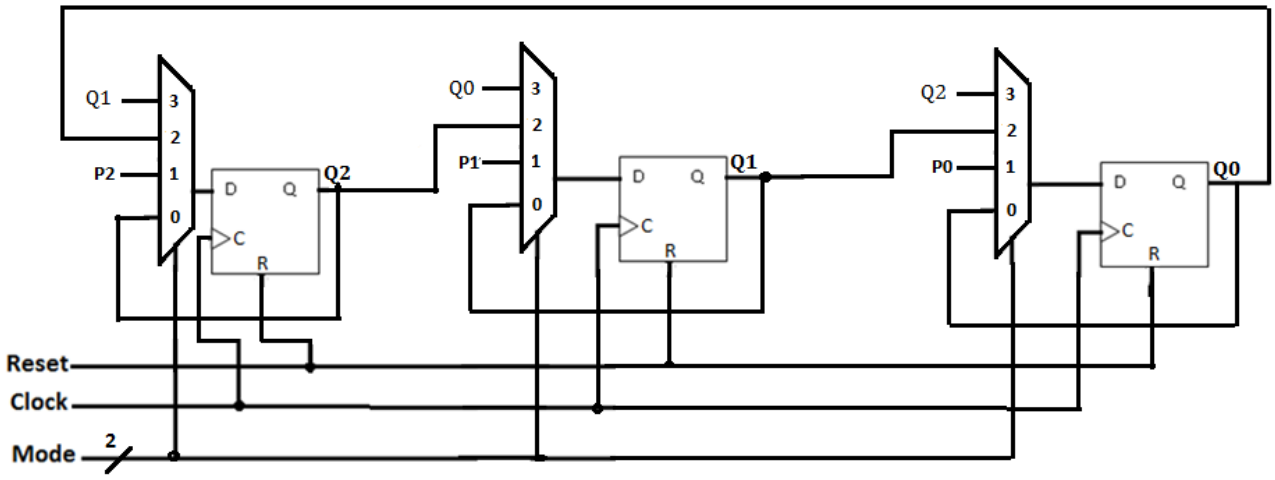


Figure3. 3-bit Register Diagram

6. Convert the 3-bit registerVerilog module you implemented to a block as in Figure 4. This block will be used in the final circuit implementation in part 3.
   File → Create/Update →Create Symbol Files For Current Files

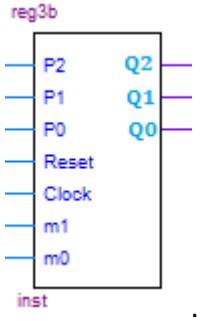


Figure 4. Block symbol for the three-bit register.

Part 3: Final Circuit Implementation:

In the file ***circuit1.bdf***, build the schematic diagram shown in Figure 1. The following steps detail how to add needed blocks to your diagram.

1. Add one segdecoder symbol, clock symbol, 3-bit counter (threebitcounter) symbol and 3-bit register (reg3b) symbol to your bdf file.

Note: In order to add these symbols to your design, select the symbol of AND gate on the tool bar (from where you get the ICs usually) then expand the project menu. If you cannot find any of the two symbols under the project menu, you can add them by typing the symbol name in the text box below.

2. Clock input of the counter:

Note that the block '***clock.bsf***' has one input '***os_clock***' and one output '***out_clock***'. This module implements a frequency division circuit which will be used to divide the high frequency clock of the FPGA oscillator (28 MHz) to obtain a slower clock (10 Hz) so that changes in the counter value can be detected on the 7-segment display and provide sufficient time for register setup. Hence, in your schematic diagram you are required to connect the "***os_clock***" to an input pin called "***clk***" (which will be assigned to the oscillator clock when you do pins assignment) and connect "***out_clock***" to the counter clock input.

3. Clock input of the register:

In your schematic diagram connect the clock input of the register to the output ***zero_count*** of the counter. This means that the state of the register will be updated each time the counter reaches count 0 according to the ***mode*** setting.

4. Connect the outputs (***C2, C1, C0***) of the counter to the inputs of the segdecoder. Note that ***A*** is the MSB.

5. Connect seven output ports to the outputs of the segdecoder (***a, b, c, d, e, f, g***) with the same names. These will be assigned to a seven segment display when you do pins assignment.

6. Connect an input pin called ***restart***, which resets the counter to count 7 and the register to 0. Note that this input is connected after being inverted to the ***set*** input of the counter (since it's active low), and connected as is to the ***Reset*** input of the register.

7. Perform the following pins assignment for your input and outputs.

Input		
clk	key0	PIN_E16
restart	iSW[1]	PIN_AB26
m0	iSW[2]	PIN_AB25
m1	iSW[3]	PIN_AC27
P0	iSW[4]	PIN_AC26
P1	iSW[5]	PIN_AC24
P2	iSW[6]	PIN_AC23
Outputs of the segdecoder		
a	HEX0	PIN_AE8
b	HEX0	PIN_AF9
c	HEX0	PIN_AH9
d	HEX0	PIN_AD10
e	HEX0	PIN_AF10
f	HEX0	PIN_AD11
g	HEX0	PIN_AD12
Outputs of the register		
Out0	oLEDR[1]	PIN_AK5
Out1	oLEDR[2]	PIN_AJ5
Out2	oLEDR[3]	PIN_AJ4

8. Download your circuit on the FPGA and test it.