***King Fahd University of Petroleum and Minerals***

***College of Computer Science and Engineering***

***Computer Engineering Department***

**COE 202: Digital Logic Design (3-0-3)**

**Term 162 (Winter 2017)**

**Final Exam**

**Wednesday, May 24th, 2017**

**Time: 120 minutes, Total Pages: 10**

**Name:\_\_KEY\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_\_**

**Notes:**

Do not open the exam book until instructed

**Calculators are not allowed** (*basic, advanced, cell phones, etc*.)

Answer all questions

All steps must be shown

Any assumptions made must be clearly stated

|  |  |  |
| --- | --- | --- |
| **Question** | **Maximum Points** | **Your Points** |
| **1** | **6** |  |
| **2** | **8** |  |
| **3** | **10** |  |
| **4** | **7** |  |
| **5** | **9** |  |
| **6** | **8** |  |
| **7** | **13** |  |

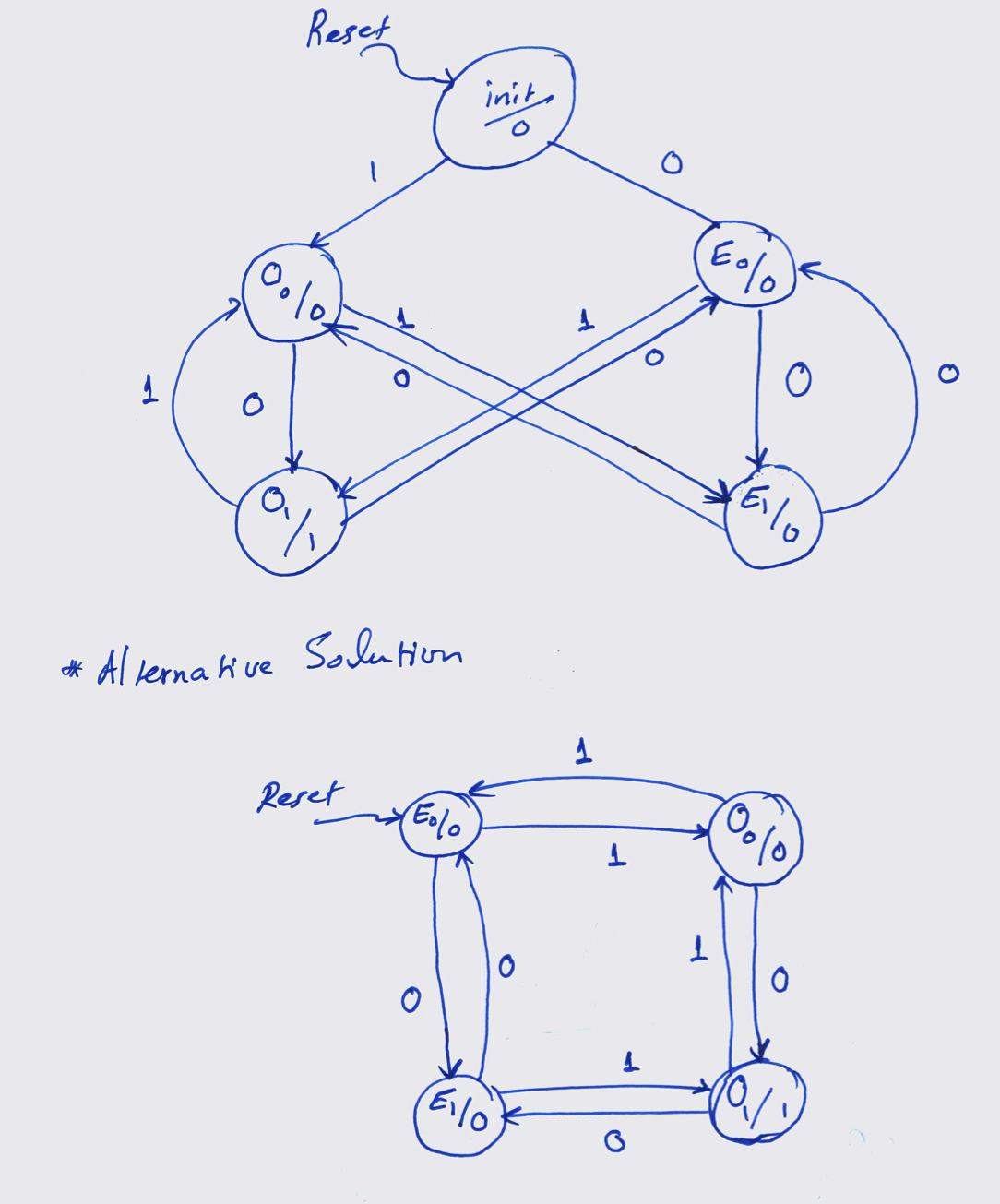
|  |  |  |
| --- | --- | --- |
| **Total** | **61** |  |

**Question 1: (6 points)**

A **Moore** *Odd parity detector* circuit has a single input ***x***and a single output signal ***parity****.* The input consists of 2-bit data chunks that are ***serially*** *received* at the input ***x****.* The ***parity*** output is 1 whenever the received 2-bit stream has an odd number of 1’s, and 0 otherwise. Draw the state diagram of this circuit. **The circuit has an asynchronous reset input to reset the machine to a reset state with an output of 0.** You are *only* required to draw the state diagram **Nothing MORE**)

Example: t = 0 time

|  |  |  |
| --- | --- | --- |
| Input | ***x*** | 1 1\_1 0\_0 0\_0 1\_0 0\_1 0\_ |
| Output | ***parity*** | 0 0\_**0** 0\_**1** 0\_**0** 0\_**1** 0**\_0** 0**\_1** |

****

**Question 2:**   **(8 points)**

1. The shown state diagram is for a **Moore** FSM of a sequence detector with a single input **X** and a single output **Y**. The circuit can detect overlapping (**overlapping**/non-overlapping) versions of the sequence **0110** (write the sequence). **(3 points)**

A/0

B/0

C/0

D/0

E/1

1

0

0

1

0

1

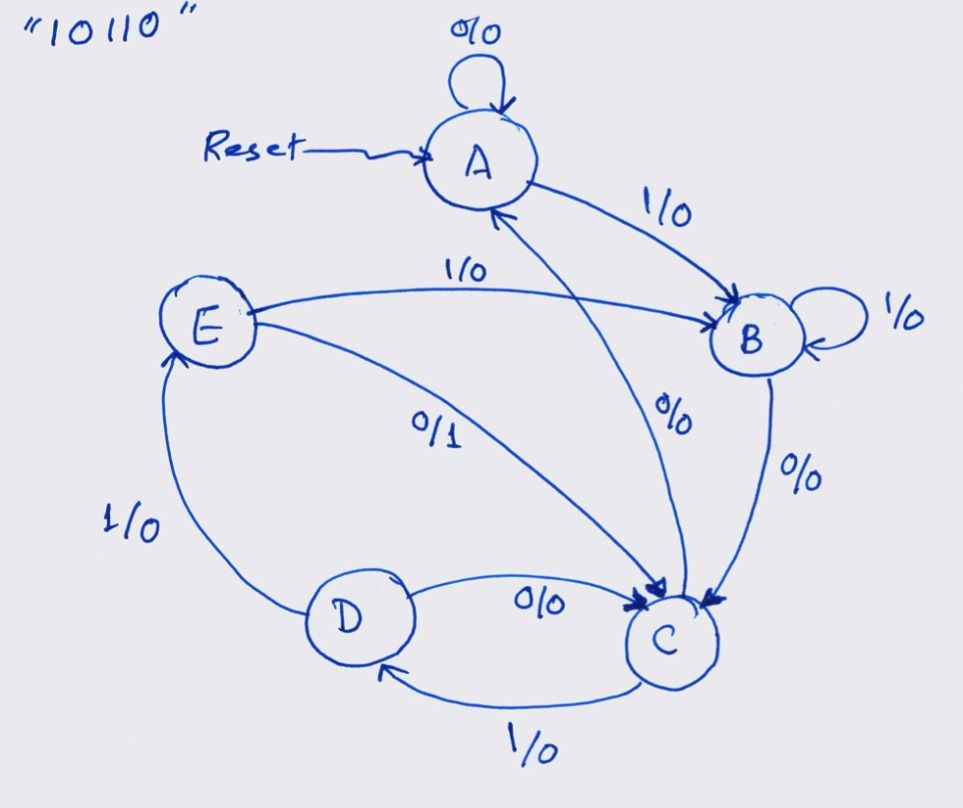
0

1

1

0

2. It is required to design a sequence detector that detects the sequence {10110} (i.e., 1 followed by 0 followed by 1 followed by 1 followed by 0) in a serial input **Z** and produces 1 at the output **W** when the sequence is detected. Assuming overlapping sequences derive the state diagram of the circuit assuming a **MEALY** model. Also, assume the existence of an asynchronous *reset* input to reset the circuit to a reset state. You are *only* required to draw the state diagram **Nothing MORE**.

**** **(5 points)**

**Question 3:** **(10 points)**

The state transition table below is for a sequential circuit with one input **X** and one output **Y**. The circuit has two state variables **A** and **B,** and an asynchronous input Reset that resets the circuit to state 00:

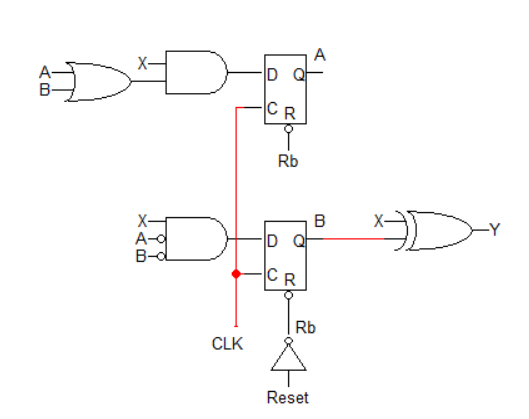
Reset State

|  |  |  |
| --- | --- | --- |
| Present State  A B | Next State  X=0 X=1  A+ B+ A+ B+ | Output  X=0 X=1  Y Y |
| 0 0 | 0 0 0 1 | 0 1 |
| 0 1 | 0 0 1 0 | 1 0 |
| 1 0 | 0 0 1 0 | 0 1 |
| 1 1 | 0 1 1 0 | 1 0 |

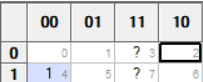
1. Does this circuit has any unused states? Briefly explain your answer (**2 points**)

**Yes, state 11, no normal input sequence can get the circuit to that state**

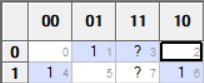
1. Design the above circuit using minimum number of logic gates and D-FFs (with asynchronous reset inputs) and draw the logic diagram of the designed circuit. The circuit should have asynchronous reset that reset it to state 00 (**8 points**)



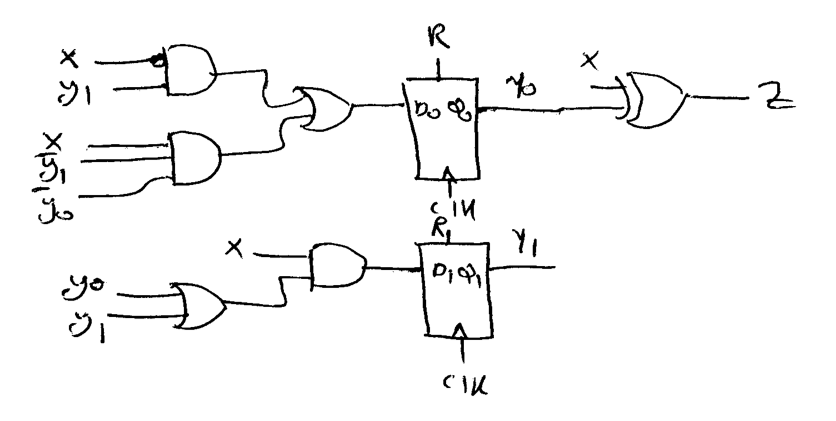
DA = X (A+B)



DB = XA’B’



Y = X ⊕ B



**B**

**A**

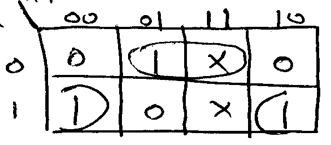
**A**

**B**

**B’**

**A**

**A’**



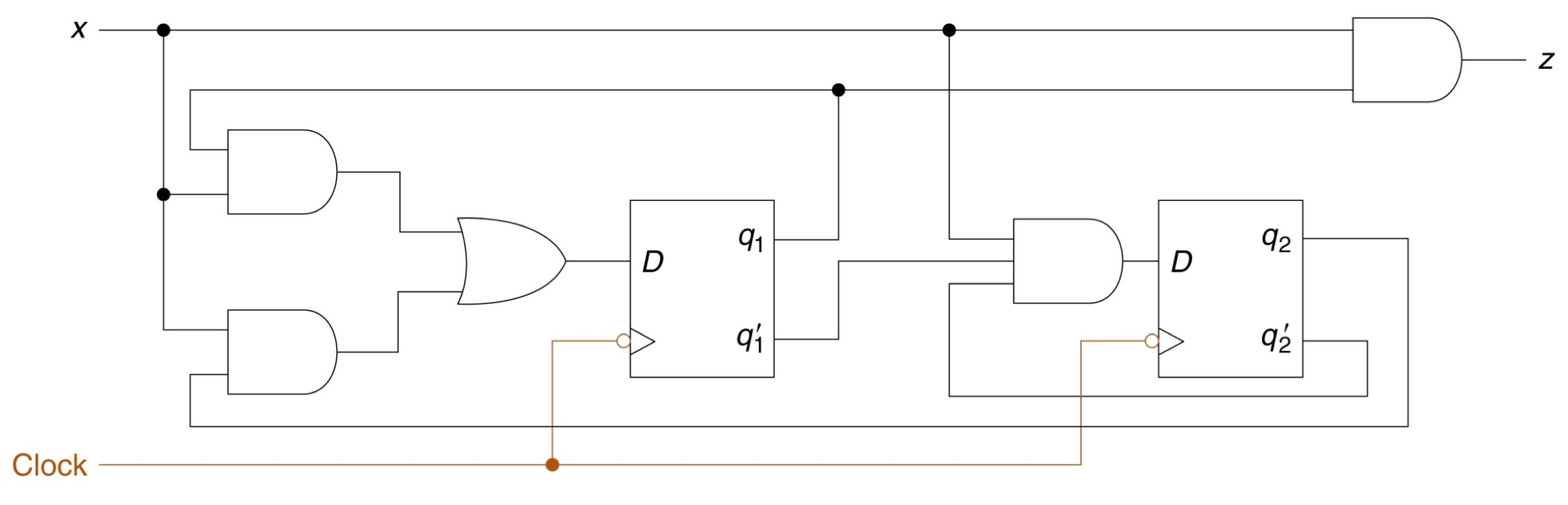
**Y = X XOR B**

**X**

**AB**

**Question 4: (7 points)**

The sequential circuit shown below has a single input ***x***together with a **RESET** input to initialize the circuit. The used D-FFs have direct/asynchronous Clear inputs (shown in the figure as CLR).



CLR

RESET

CLR

*A*

*A’*

*B*

*B’*

*DA*

*DB*

1. Obtain the Boolean expressions for the DA , DB (flip flop inputs) and the output ***z***. **(3 points)**

DA = (A + B) x

DB = A’B’x

z = Ax

1. Derive the state transition table of the circuit (fill the table below). **(4 points)**

|  |  |  |  |
| --- | --- | --- | --- |
| Present State | Input | Next State | Output |
| **A B x** | | **A+ B+** | **z** |
| 0 0 0  0 0 1  0 1 0  0 1 1  1 0 0  1 0 1  1 1 0  1 1 1 | | 0 0 0 1  0 0 1 0  0 0 1 0  0 0  1 0 | 0  0  0  0  0  1  0  1 |

**Question 5:**   **(9 points)**

The state diagram below is for a sequential circuit that one input **X** (in addition to an **asynchronous** **Reset** input), one output **Y,** and state variables **A** and **B**.

0/0

0/1

1/0

0/0

Reset

1/1

1/0

1/1

0/1

1. Obtain the state transition table of this circuit. (**2 points**)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *A* | *B* | *X* | *A+* | *B+* | *Y* |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 |

1. Specify whether this circuit is a Mealy or Moore Machine? Explain (**1 point**)

**Moore, the output in each and every state is constant (i.e. does not depend on the input)**

1. If the circuit is in state **00**, what is minimum number of clock cycles required to reach state **11**?

**3 clock cycles (00**🡪**01**🡪**10**🡪**11)** (**1 point**)

What is the required input sequence? (**1 point**)

**1-1-1**

1. Complete the following timing diagram of the circuit for the inputs shown assuming that the FFs are **positive edge triggered**: (**4 points**)

**Reset**

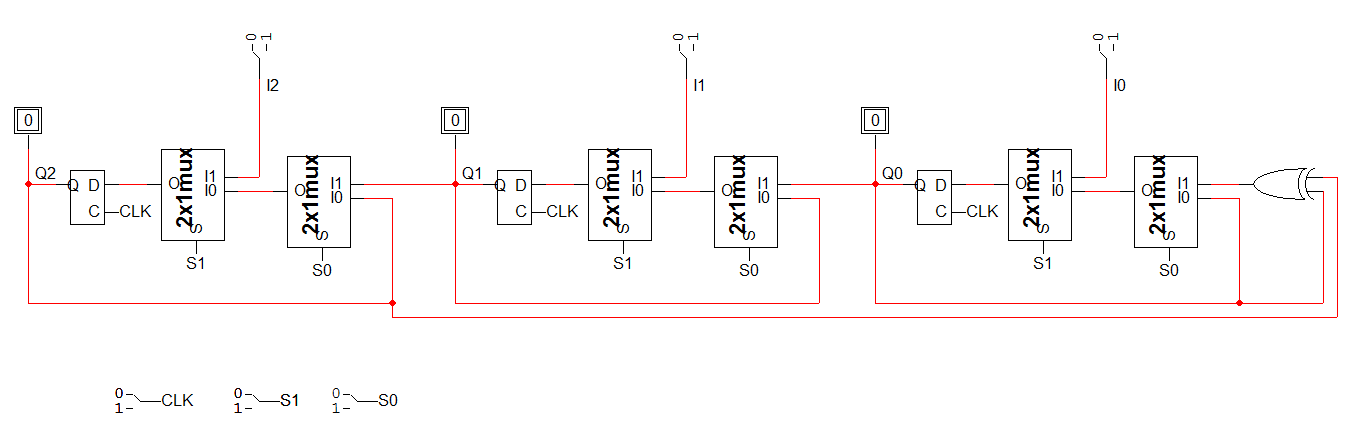
**Y**

**AB** **00 01 00 01**

**Question 6: (8 Points)**

## Using minimum number of D-FFs and other needed standard components and logic gates, show the design of a 3-bit register Q that has two control inputs: S1 and S0. The register has a 3-bit external input I2I1I0. The table below shows the functionality of the register. **(4 points)**

|  |  |  |
| --- | --- | --- |
| S1 | S0 | Action |
| 0 | 0 | No change in Q |
| 1 | X | Load parallel input (i.e. Q2Q1Q0🡨 I2I1I0) |
| 0 | 1 | Q2🡨Q1,Q1🡨Q0,Q0🡨Q2⊕Q0 |



## Complete the following table by showing the content of register Q after each clock cycle:

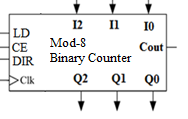
## **(4 points)**

Inputs in a cycle affect the register in the next cycle

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clock # | S1 | S0 | I2 I1 I0 | Q2 Q1 Q0 |
| 1 | 1 | 0 | 1 1 1 | 0 0 0 |
| 2 | 0 | 1 | 1 0 1 | 1 1 1 |
| 3 | 0 | 1 | 1 1 1 | 1 1 0 |
| 4 | 0 | 1 | 0 1 1 | **1 0 1** |
| 5 | 0 | 0 | 1 0 1 | **0 1 0** |
| 6 | 1 | 0 | 1 1 0 | **0 1 0** |
| 7 | 0 | 0 | 1 1 1 | **1 1 0** |

**Question 7**  **(13 Points)**

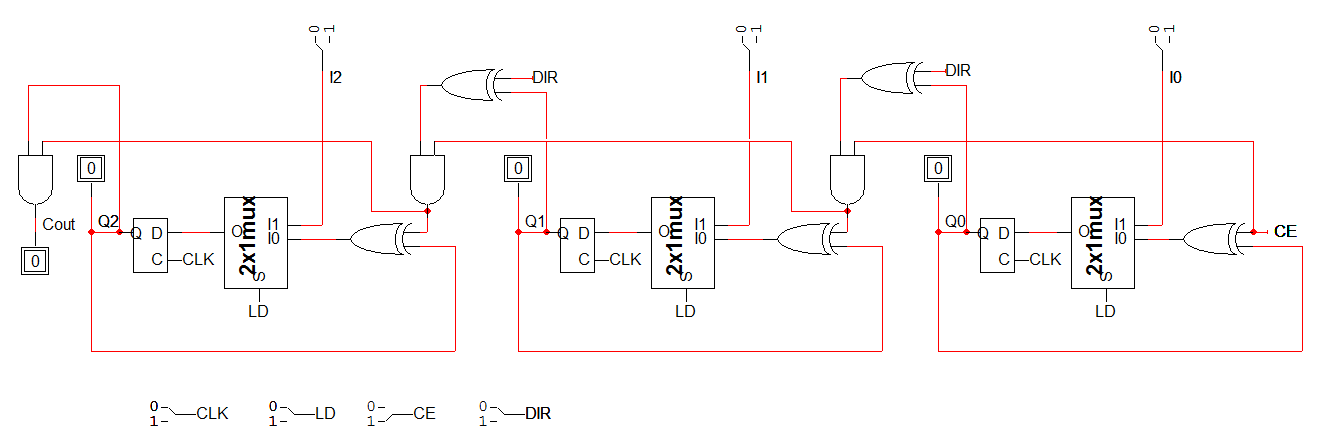
1. It is required to design a **mod 8** **up/down counter** that has the following control inputs:

* **LD** (parallel load), together with its associated inputs I2, I1, I0.
* **CE** (Count Enable)
* **DIR (**when 0 counting up and when 1 counting down**)**

The counter produces an output signal (**Cout**) which equals **1** when its output equals 7 when DIR=0 and CE=1 OR when its output equals 0 when DIR=1 and CE=1. Design the counter using D-FFs and **minimum** number of **logic gates and minimum-size Multiplexers**. **Note: Do not use an adder in your solution.**

**(6 Points)**

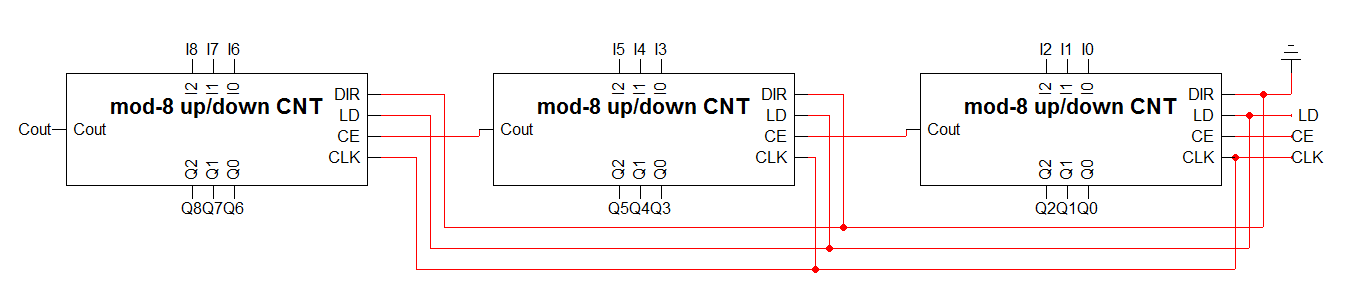
|  |  |  |  |
| --- | --- | --- | --- |
| **LD** | **CE** |  | **Counter Next Content after the clock pulse** (Q2Q1Q0)+ |
| 1 | X | X | I2 I1 I0 (load) |
| 0 | 1 | 0 | (Q2Q1Q0)+1 (Increment up by 1) |
| 0 | 1 | 1 | (Q2Q1Q0)-1 (Decrement up by 1) |
| 0 | 0 | X | Q2Q1Q0  (no change) |



1. Given that the clock frequency of the **mod-8 up/down counter** is 32 MHZ, what is the clock frequency of the Q2 output of the counter when the counter is set as an up counter? **(1 Point)**

The clock frequency of Q2 will be 32 MHZ / 8 = 4 MHZ.

1. Using any number of the above **mod 8** **up/down counter** and other needed logic gates design a **mod 512 up counter**, which has the control inputs: LD, and CE. Clearly label all inputs and outputs. Note that only an **up counter** is required. **(3 Points)**



1. Modify your design in (III) to obtain a clock frequency divider that divides the clock frequency by **258**. Show the signal to be used as a clock that has the divided frequency with the name CLKnew.

**(3 Points)**

