Name: Rami Omer
Student#: 215855620

Lab 2 Report

In this report I try to explain the design process and the actual coding that went into implementing the deliverable material of Lab 2. The first section of this report discusses the design process using a text-based quasi-FSM technique. The second section discusses the implementation of that design in the form of a pseudo-Verilog code.

Section 1: The Design Process

Before I get into the FSM diagram, it is significantly important to highlight the following notes in order to make it easier for the reader to understand what is happening in the FSM diagram:

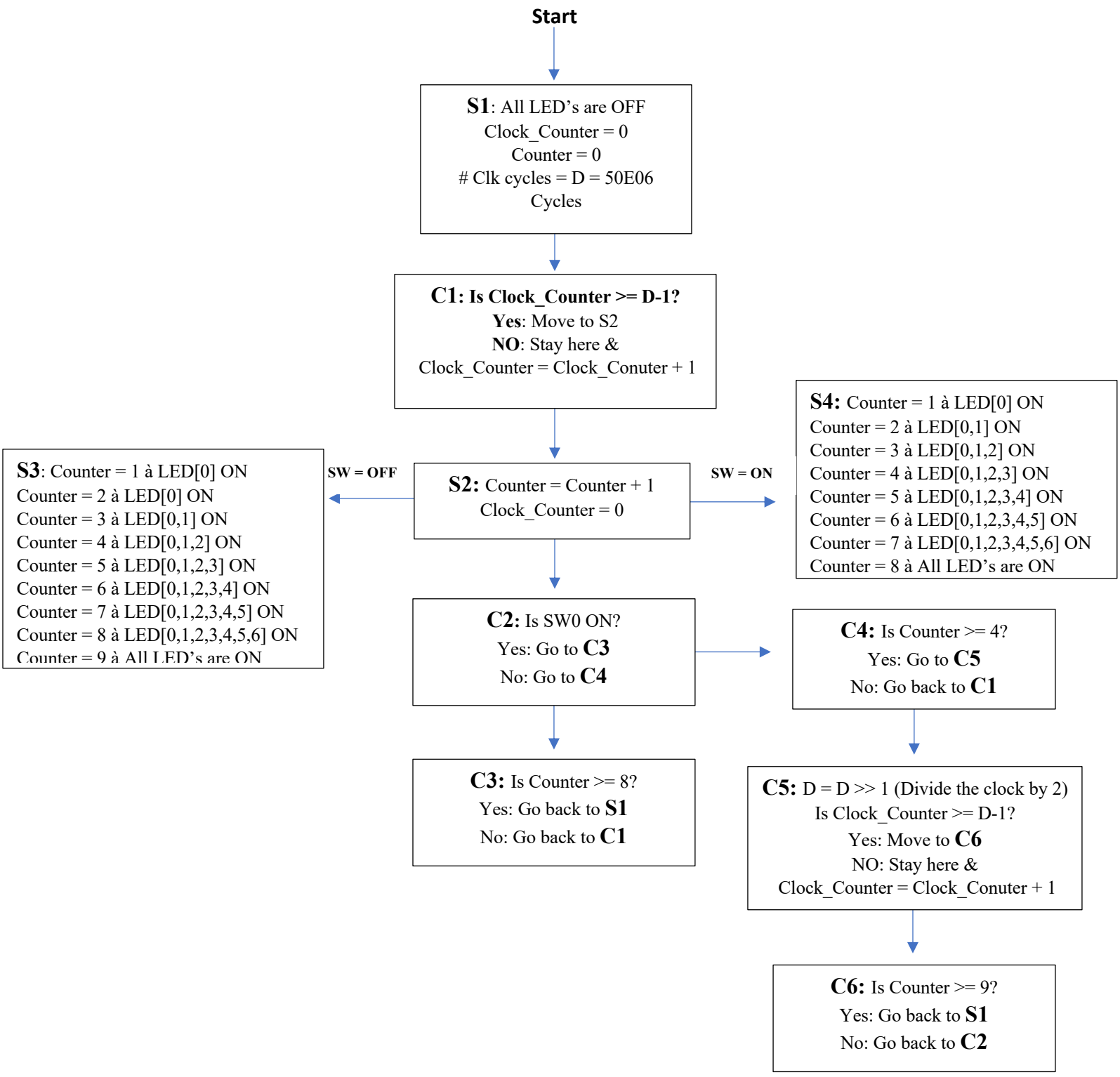
The first note: The frequency of the clock used is 50MHz. To calculate one second in real life unit of time we need to count 50E06 DE-10 clock cycles

The second note: The leftmost LED on the board is given the label LED[0] and the rightmost LED on the board is given the label LED[7] in this report.

The third note: What happens when SW0 changes position midway through the operation?
The system is designed in a way that allows the pattern to be sensitive to the position of SW0. In other words, the pattern that LED's follow is asynchronous with the SW0. An example is provided below:

If the staring position of SW0 was 1 and we change SW0 from 1à 0 as the third LED is about to turn ON, the third LED and all the way to the eighth LED will follow the pattern associated with SW0 being in the OFF position whereas the first 3 LED's will follow the pattern wired for when SW0 is in the ON position.

Name: Rami Omer
Student#: 215855620



Name: Rami Omer
Student#: 215855620

Section 2: The Code

I used 3 modules to build the circuit needed to achieve the goal of Lab 2. The first module deals with the clock and the time counter. The second module takes the output timer of the first module and translates it using combinatorial logic into ON/OFF state for each of the 8 LED's. The third module is the top-level hierarchy and it is the module that ties the first two modules together.

The First Module: Clock Diver (Counter)

Inputs: DE-10 internal clock & SW0 state
output: Counter stored in a 4-bit register

The frequency of the DE-10 clock stored in parameter D = 50E06 cycles
Create a 32-bit register to store clock ticks

At every rising edge of the clock, the circuit is designed to do the following:

1- If SW0 = 1, do the following:

- Count 50E06 cock cycles
- When the clock ticks count is 50E06, do the following:
  - Increment the output Counter by 1 à Every 50 clock cycles the module counts one second
  - Set the clock tick register to zero
  - When the Counter is equal to 8 or more, set the Counter to 0

2- If SW0 =, do the following:

- Count 50E06 cock cycles
- When the clock ticks count is 50E06, do the following:
  - Increment the output Counter by 1 à Every 50 clock cycles the module counts one second
  - Set the clock tick register to zero
  - When the Counter is equal to 4 or more, do the following:
    - ➔ D = D >> 1 (Shift the clock ticks to the right one bit). In other words, for the 4th LED make the cycle count drop from 50E06 to 25E06 which means that we are now incrementing counter every ½ a second. But since we are shifting D to the left and updating the value of D to the shifted value, the 5th LED make the cycle count drop from 25E06 to 125E05 ticks, so counter now is incremented every ¼ second and so on
  - When the Counter is equal to 9 or more, set the Counter to 0

The Second Module: A Decoder

Inputs: a 4-bit counter & SW0 state
outputs: the 8 LEDs: LED[0]-LED[7]

The following is the truth table for each LED needed to create the combinatorial logic to control each LED. Note that each LED is a function of SW0 state and the 4 bits of the counter.
L[i] = f(SW0, C[3:0])

Name: Rami Omer
Student#: 215855620

Example of how this works in the first 2 LED's is shown below:

LED[7] is ON when counter is equal to 1-8 and SW0 =1 or when SW=0 and counter is equal 1-9

LED[6] is ON when counter is equal 2-8 and SW0 =1 or when SW=0 and counter is equal to 3-9 (2 seconds wait time when SW=0)

The Second Module: Connecting the first two modules

Inputs: DE-10 clock, SW0
output: the 8 LEDs

Calling the first two modules in the top-level hierarchy module:

Counter(clock, SW0, Counter)
Decoder(SW0, Counter, LED[7:0])