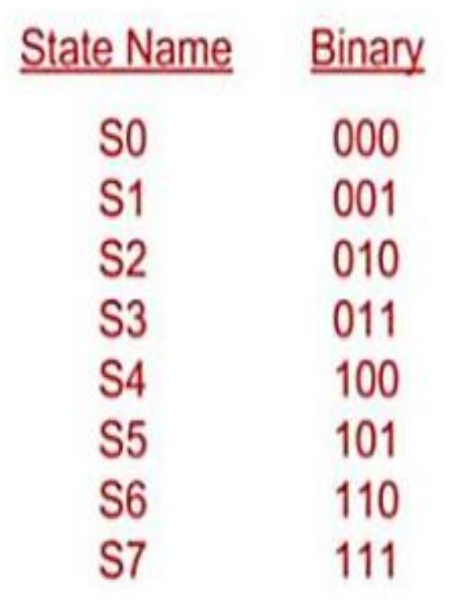
Professor Bhavnagarwala Anik Barua

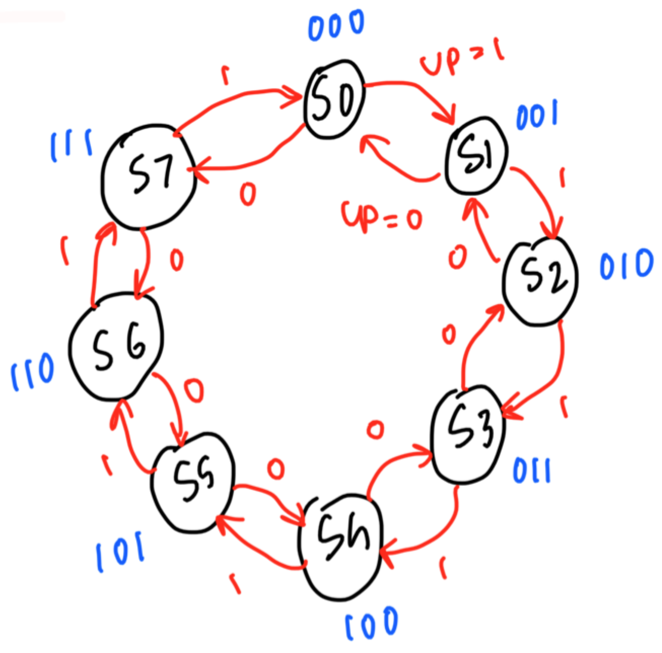
CS 2204 - Section C 11-12-2021

Lab 8: Up/down Counter using Verilog

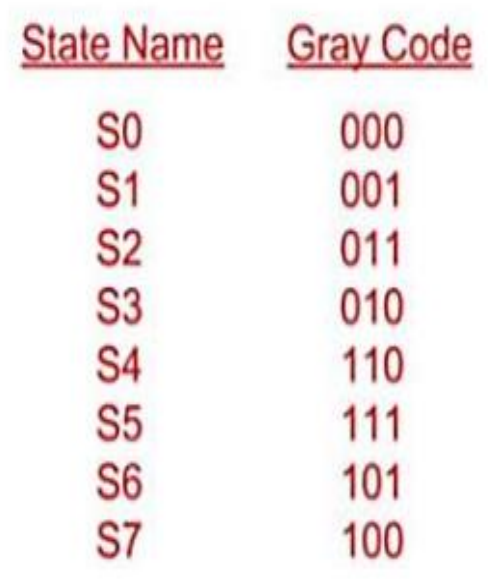
Today's lab is about developing Verilog modules for three up/down counters - binary (3 bit), Gray Code (3 bit) and One-Hot (8 bit). Counter is a special type of finite state machine that traverses through states within a state diagram circling around all states. This circling around allows a special type of output topology called state-encoded-outputs and states can be associated with the counter output value because each state in the counter represents a unique counter. In addition, the current state code can be used as the output of the entire system. In this lab we will focus on binary, gray code and one-hot counter. In a binary counter, the input determines if the counter increments or decrements by changing its states. For example: if we consider a 3 bit binary counter, we have 2^3 = 8 possible states in total. If input is 1 it increments to next state, if 0 it decrements to previous state. If we see it in state diagram, it looks like -



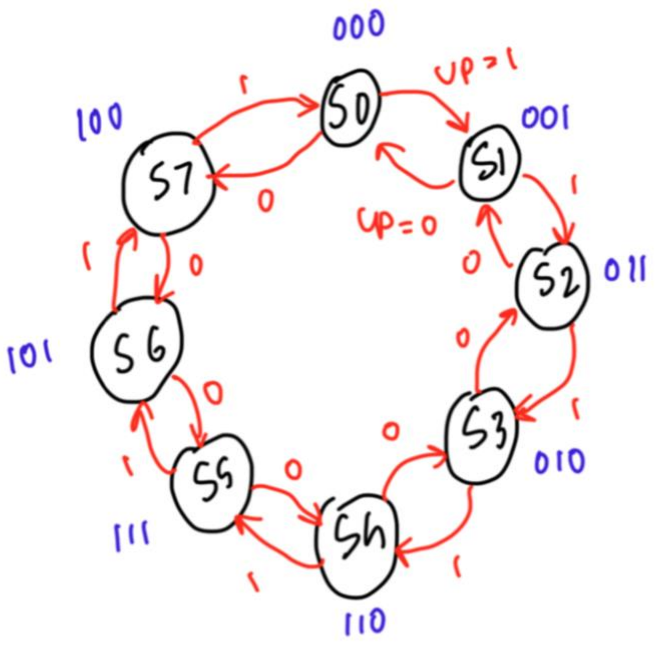
State Name	Binary
S0	000
S1	001
S2	010
S3	011
S4	100
S5	101
S6	110
S7	111



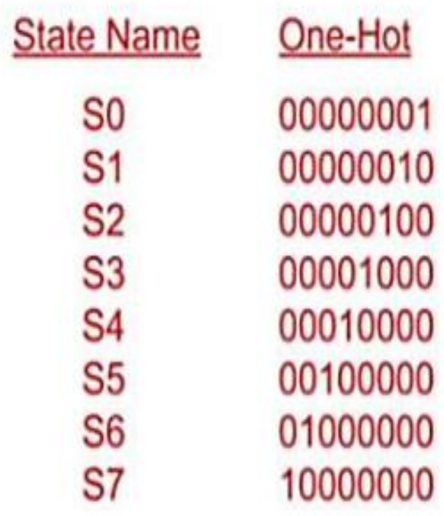
As we can see we have 8 states (S1 -> S7) in total and when up = 1, we go to next higher states (010 to 011) but when up = 0, we decrement by 1 i.e. we go back to last state (010 to 001). Each state is linked to another and loops back to start which is 000. Here, Next state logic depends on both current state and input (up value). Next is Gray code counter. Gray code is one which the value of a code differs by only one bit from any of its neighbors, (i.e., 00, 01, 11, 10...) like we have seen in k-map. For 3 bit Gray Code the sequence will be - 000 -> 001 -> 011 -> 010 -> 110 -> 111 -> 101 -> 100 -> 000 (loops back). It is useful for reducing the number of bit transitions on the state codes when the machine has a transition sequence that is linear and it can reduce the amount of power consumption and noise generated by the circuit.



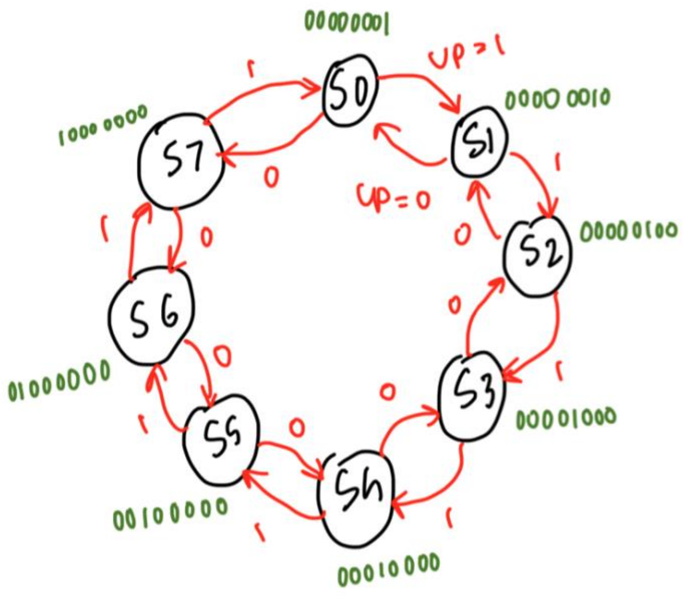
State Name	Gray Code
S0	000
S1	001
S2	011
S3	010
S4	110
S5	111
S6	101
S7	100



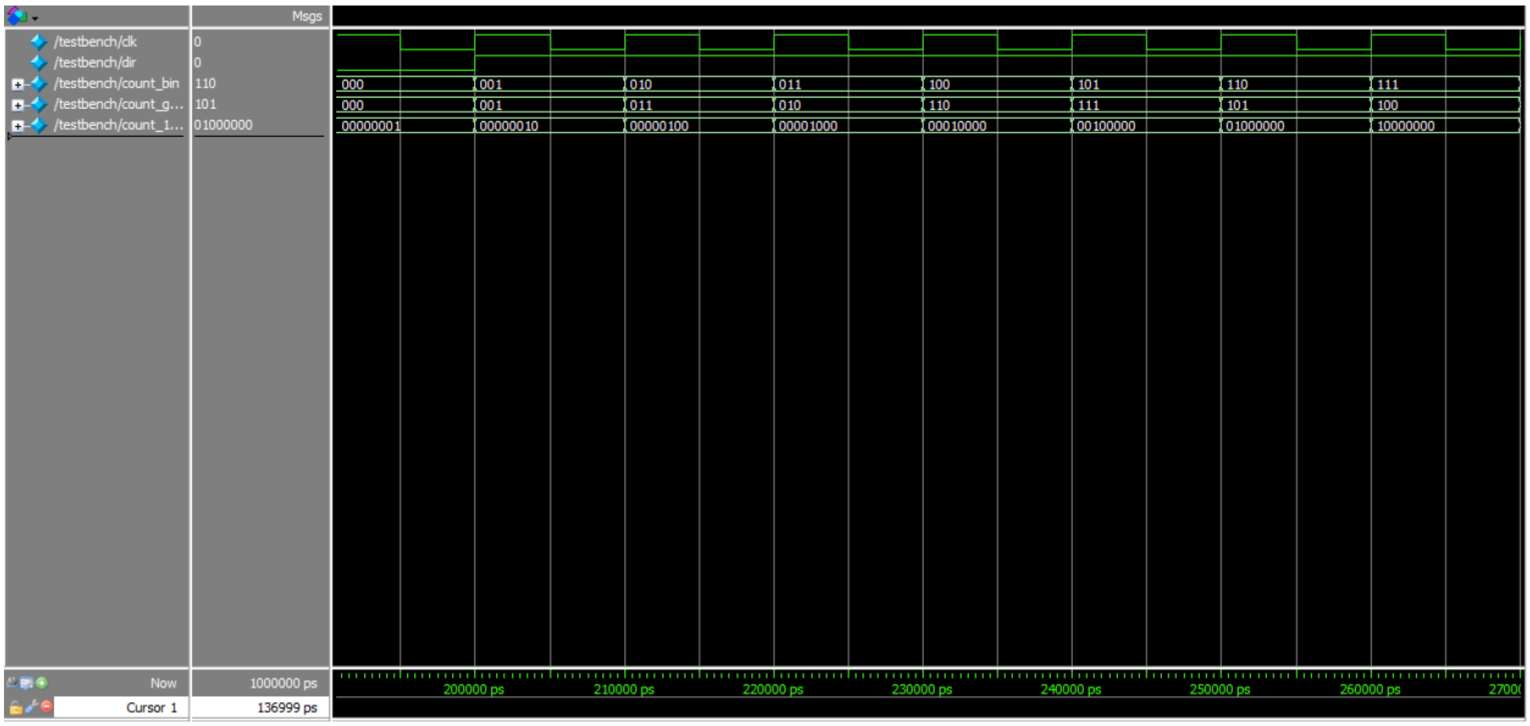
The last one is One-Hot counter that will output an incrementing one-hot pattern on every rising edge of CLK: 001→010→100→001. For this lab we will implement 8 bit one-hot so the sequence will be 00000001 -> 00000010 -> 00000100 -> 00001000 …



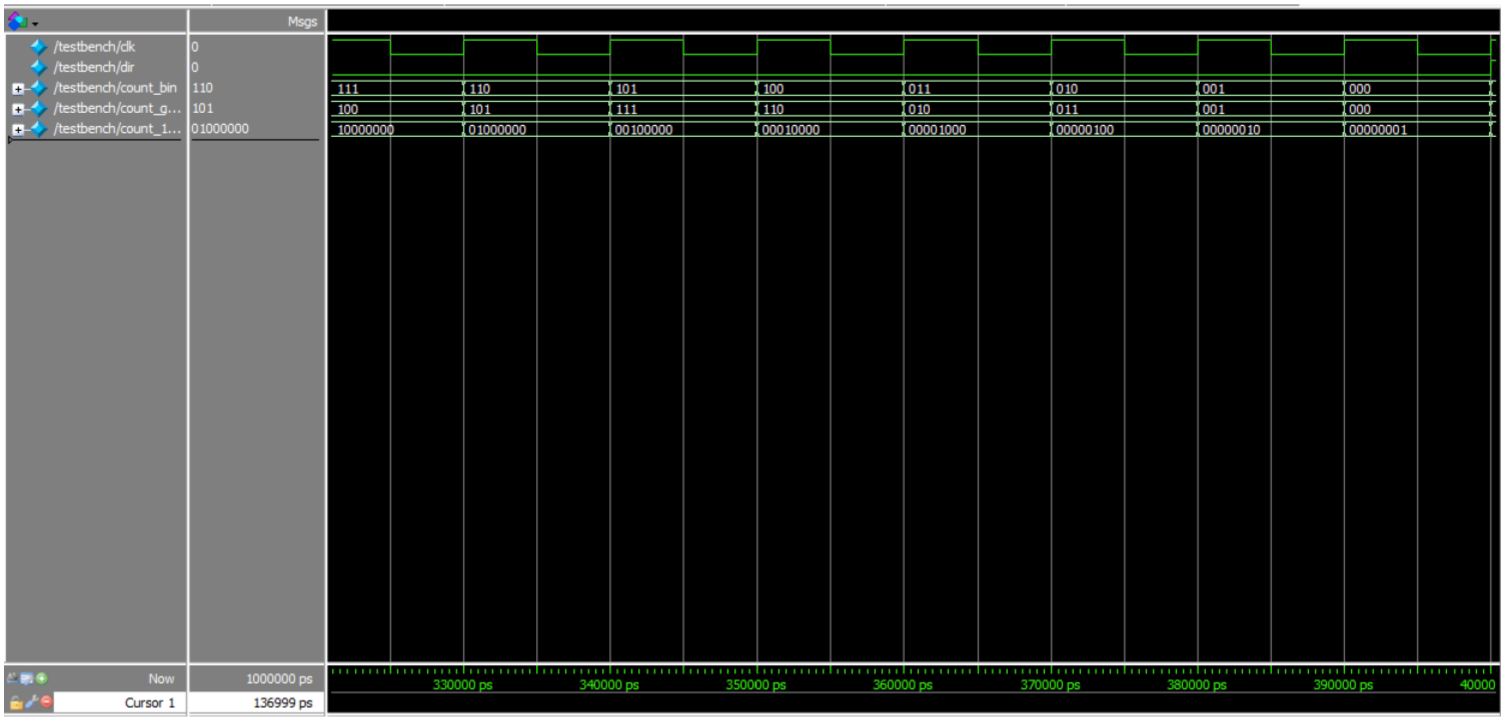
State Name	One-Hot
S0	00000001
S1	00000010
S2	00000100
S3	00001000
S4	00010000
S5	00100000
S6	01000000
S7	10000000



We need to implement both up and down counter for each - binary, gray code and one-hot, so next step is writing the Verilog module for each. First we will start with input and outputs for each counter. For input it will take clk (clock signal) and dir (direction) that is a 1’b number (1 or 0). The dir value (1 and 0) determines the up/down transition from each state, for example: if dir is 1 we go to next state, if 0 we go to previous state. Now for the output, we will have reg 3 bit number assigned as count for binary and gray, and 8 bit number for one hot. Then we declare parameters for our each 8 states starting with S0 = ‘000’ to S7 = ‘111’ and 3 bit state initialized to 000. Next we will use an always block to make sure at positive edge of clock, we begin our transition of states. Inside the always block, we will use a case method for each state S0 to S7, where using if-else block we will set the next state and previous state for each depending on direction value. So for example: if dir == 1’b we go to next state, else (dir = 1’b0) we go to previous state. After setting up the direction for each state, we have to setup our counter value. For that we can have a second always block for state, and using if else we can assign 3’b output value for each count output (binary and gray code) and 7’b output value for One-Hot. Now we have to create the testbench to verify their functionality using the waveforms. For testbench, we first instantiate reg clk = 0, reg dir = 0, wire 3 bit count output for binary and gray-code and 7 bit for one-hot. Then we instantiate each counter by assigning clk, dir, count as the same sequence of each module. After that using always blocks, we negate the clk value every #5 seconds and dir value every #100 seconds. If we simulate the testbench, we get the waveforms for each counter-



We can see the waveforms represents the correct functionality for binary up counter, gray code up counter and one hot up counter. At every positive edge of the clock, with dir value 1, we can see the counter increments by 1. As we know when dir = 1, we go to next state and we can verify it at every positive edge of the clock that goes up by a state. For example: in binary, when dir = 1 (up counter), at each positive edge, it increments by 1, 000 -> 001 -> 010 -> 011 … The same functionality works for gray code up counter and one hot up counter. Next, we will evaluate the down counter (decrement by 1 at every positive edge of clock when dir = 0) -



We can see, this waveforms represents the correct functionality for binary down counter, gray code down counter and one hot down counter. At every positive edge of the clock, with dir value 0, the counter decrements by 1. As we know when dir = 0, we go to previous state and we can verify it at every positive edge of the clock that goes down by a state. For example: in binary, when dir = 0 (down counter), at each positive edge, it decrements by 1, 111 -> 110 -> 101 -> 100…

Now for second part, we will use the code for each up/down counters - binary (3 bit), Gray Code (3 bit) and One-Hot (8 bit) in Vivado to implement it on FPGA Board where we will drive the outputs of the counters to the 7-segment display with the up/down input taken from a push button switch in the board.

Here are the YouTube video links for Binary, Gray Code and One-Hot up/down counter -

Binary Counter - https://youtu.be/whqOsvT8U3s

Gray Code Counter - https://youtu.be/nzaMO22q4XE

One-Hot Counter - https://youtu.be/w_WxGeSKi4c