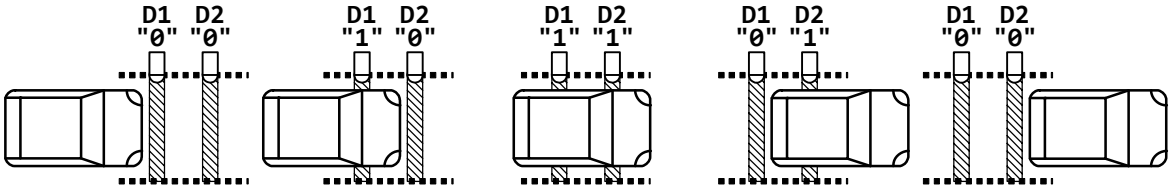
EE342 - Digital System Design

Laboratory Experiment - 6
Finite State Machine

Preliminary Work

1. Read the lecture notes on **Finite State Machines**.

2. Write a finite state machine (FSM) module that counts cars going through a gate. A car entering through the gate activates two detectors successively as shown below.



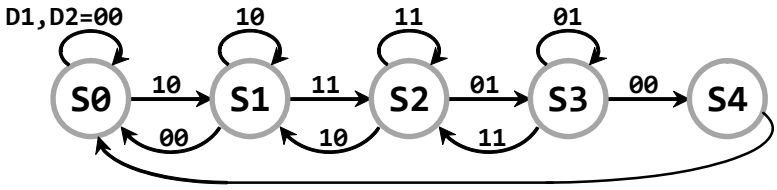
D1 and **D2** detector signals are the inputs of the FSM. If a car goes through the gate then **D1** and **D2** generate the following sequence:

"00" → "10" → "11" → "01" → "00"

If a car stops in the middle of the gate and goes back, then the sequence will be different:

"00" → "10" → "11" → "10" → "00"

The FSM states should track the sequence of **D1**, **D2** inputs to be sure about the passage of the car as shown in the state diagram given below.



S4 state is included just to enable the counter. FSM stays at **S4** for only one clock cycle before it goes back to the idle state **S0**, and the counter is incremented when the FSM state is **S4**.

3. Create a new project for this experiment. Set the **State Machine Processing** option to "**User-Encoded**" to obtain the state codes determined in the program.

- Select the "**Settings…**" item under the "**Assignments**" menu.
- Select "**Analysis & Synthesis Settings**" on the left.
- Click on the "**More Settings…**" button.
- Select "**State Machine Processing**" option in the drop-down list.
- Select "**User-Encoded**" setting.

Compile the FSM module and debug the code, if necessary.

Procedure

1. Test the FSM module you wrote in the preliminary work. Create a new simulation waveform file as follows.

- Include all inputs, counter output, and the FSM state registers.
- Set "**End Time**" of the waveform file to **10 µs**.
- Set "**Grid Size**" of the waveform file to **200 ns**.
- Generate a **10 MHz** clock as the **`Clock`** input.
- Apply different waveforms to **D1** and **D2** inputs to check all possibilities:

"00"→"10"→"11"→"01"→"00"	=> increment the counter
"00"→"10"→"00"	=> car goes back, don't count
"00"→"10"→"11"→"10"→"00"	=> car goes back, don't count
"00"→"10"→"11"→"01"→"11"→"10"→"00"	=> car goes back, don't count

Note that, there must be at least one clock cycle between any pair of **D1** and **D2** transitions for FSM to work properly.

2. Change the **State Machine Processing** option to "**One-Hot**" following the steps described in the preliminary work. Compile the project and run simulation again. You should include the new state variables in the waveform file to see the one-hot encoded state values.

Restore the **State Machine Processing** option back to "**User-Encoded**".

3. Modify the FSM module to count the cars that go through the gate in both directions. Increment the counter if a car enters through the gate and decrement it if a car exits. The sequence of **D1** and **D2** transitions will be reversed when a car exits through the gate:

	D1 and D2 transitions
A car enters through the gate:	**"00"→"10"→"11"→"01"→"00"**
A car exits through the gate:	**"00"→"01"→"11"→"10"→"00"**

4. Compile the new FSM and test it by using simulation waveforms similar to those described in **step-1**. Prepare detector waveforms for a car that exit through the gate in addition to the waveforms for entry through the gate.