Computation II: embedded system design (5EIB0)

Available from: Friday, 8 March 2024, 2:30 PM
Requested files: seq_top.v (Download)
Type of work: Individual work

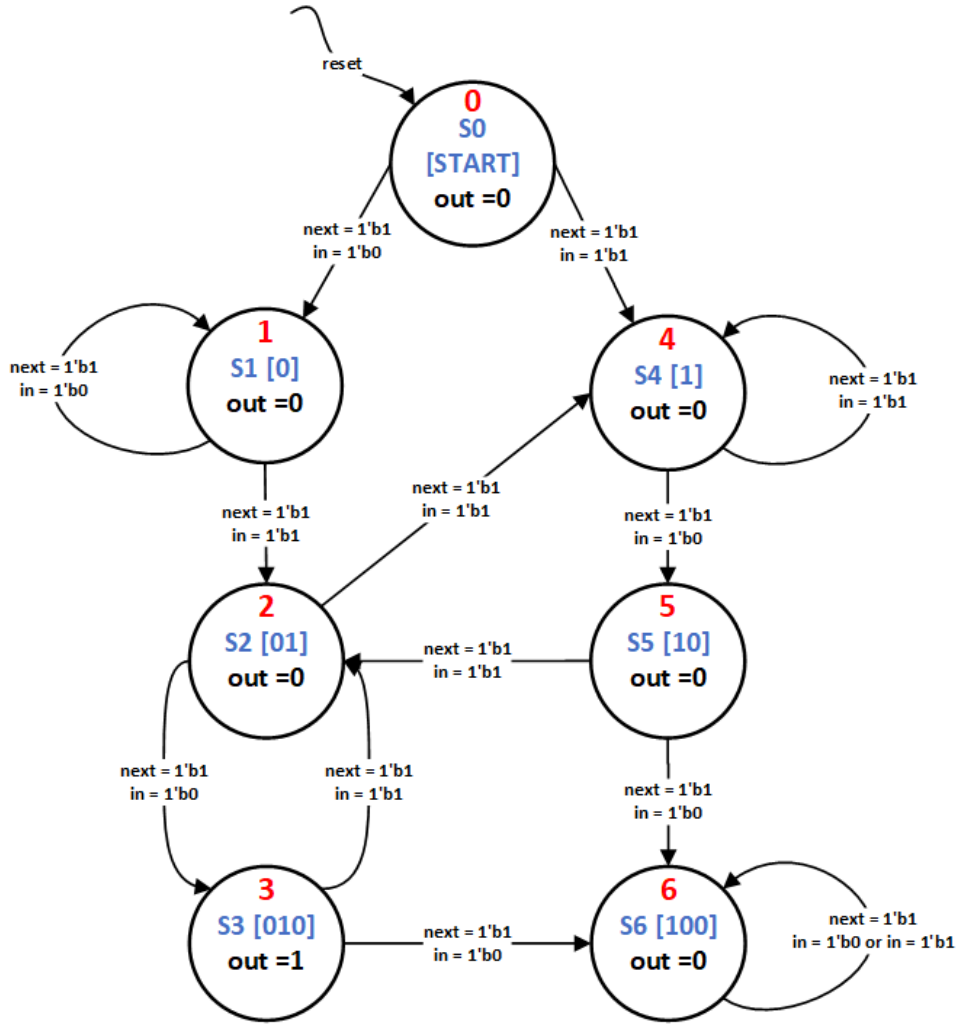
Introduction

In this exercise, you will create a verilog implementation of a Finite State Machine (FSM) which detects a finite string pattern. It has the following properties:

- one input (**in**) and one output (**out**)
- output is asserted whenever the input sequence **... 010 ...** is observed, as long as the sequence **...100...** is not seen. If **...100...** is seen, then the fsm stays in the state it is in.

Example input/output behavior:

```
in:      0 0 1 0 1 0 1 0 0 1 0 ....
out:    0 0 0 1 0 1 0 1 0 0 0 ...
```



The diagram above illustrates the transitions of the FSM that should be implemented to achieve this. It shows a Moore Machine where the output is defined by the state. The machine has six states. The state numbers are marked in red.

Assignment

Write a FSM module named **seq_top** which takes **input signals clk, reset, in, next** and produces **output signals out, state_display**. Assume all input and output signals to be 1 bit wide except state_display which is 3 bits wide and little endian. The state_display must output the number of the current state using the same numbers as the states in the FSM diagram above (shown in red). All state transition take place at posedge of clk. The reset input always makes a transition to the S0 state. The **reset** input is a synchronous signal (i.e. it goes high only at posedge of clock).

Note: A transition will happen only if **next** is high. If **next** is low, the **state does not change**. Make sure that you only make one transition when the **next** is set from 0 to 1. For doing this check whether the current value is different from its previous value. Do not use: always @ (posedge **next**).

Debug

Click on the symbol marked below to see the waveforms produced by your design. Please note that if your code has an error that prevents it being simulated it will not produce any waveforms.

VPL