Ryerson University

Department of Electrical and Computer Engineering

COE/BME 328 – Digital Systems

Lab 5 - VHDL for Sequential Circuits: Implementing a customized State Machine

15 Marks (2 weeks)

Due Date: Week 10

1 Objectives

- To simulate and verify the operation of a sequential circuit.
- To design a finite state machine (FSM) that cycles through the individual digits of your student ID using the assigned state diagrams.
- To learn the difference between Mealy and Moore machines and express the FSMs with different state assignments.

2 Pre-Lab Preparation

1. You will be assigned one of the state machines described by the state diagrams shown in Figure 1.
2. Your implementation will either be a Mealy or Moore state machine as assigned by your lab instructor. Produce a state table and state-assigned table for your customized state machine.
3. Design the logic equations for each of the Flip-Flop inputs described in Figure 3.
4. Draw the logic diagram either as Mealy or Moore state machine for your circuit (depending on the assignment by your lab instructor.)
5. Create a file *lab5.vhd*.

3 Laboratory Work

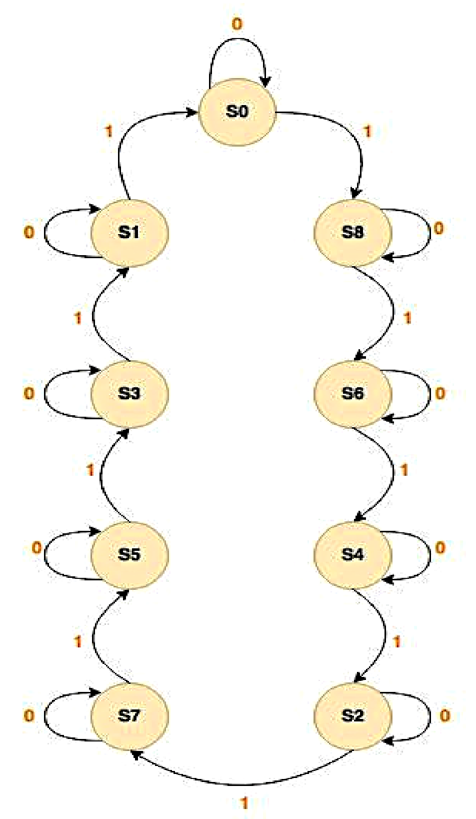
1. Create the subdirectory *lab5* in your work directory, and copy the file *lab5.vhd* to the subdirectory.
2. Consider the 9 digits of the student identifier D = {d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9} in its general representation. Then, as an example, a student with identifier: **500435429** will follow the sequence as in Table 1.
3. The corresponding Mealy/Moore machine state diagrams for representing student ID in Table 1 is depicted as in Figure 2. Your circuit design must handle non-valid states and non-valid student identifier cases by displaying an "**E**" in the seven segment display.
4. Modify and compile your design (Figures 4 and 5).
5. Demonstrate results to the instructor by displaying both the states and student identifier

NOTE: Re-use the 7-segment module from Lab3 to display states and student identifier digits.

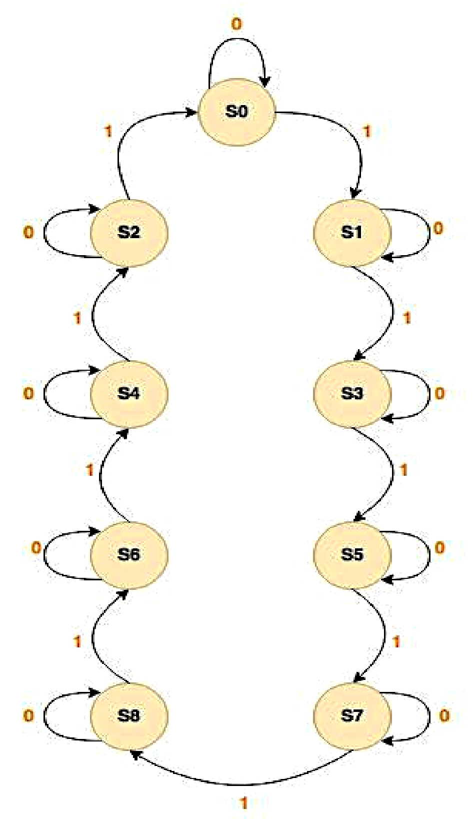
student_id_digit_1	d_1	5
student_id_digit_2	d_2	0
student_id_digit_3	d_3	0
student_id_digit_4	d_4	4
student_id_digit_5	d_5	3
student_id_digit_6	d_6	5
student_id_digit_7	d_7	4
student_id_digit_8	d_8	2
student_id_digit_9	d_9	9

Table 1 Digits Representing Student Identification number

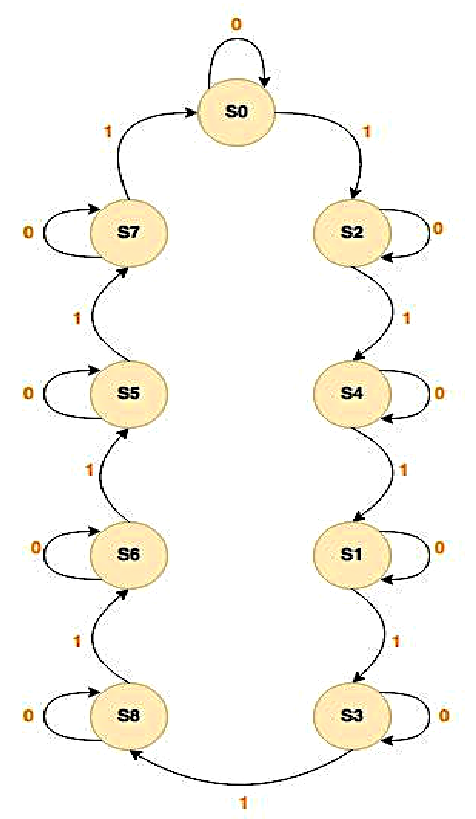
FSM #1



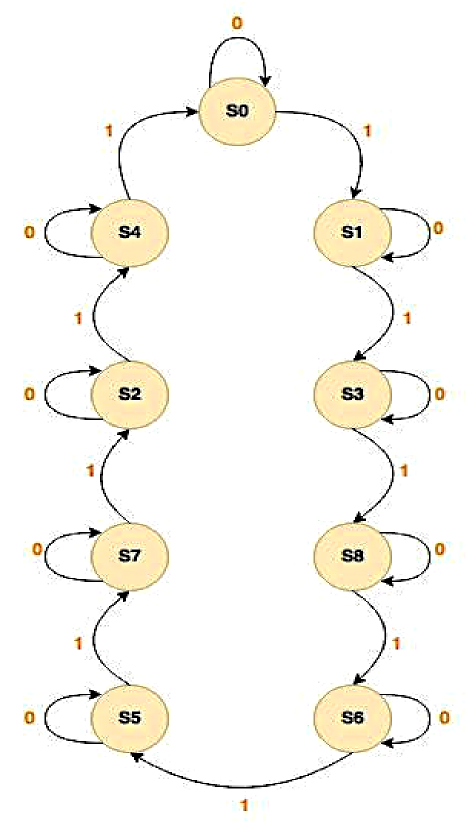
FSM #2



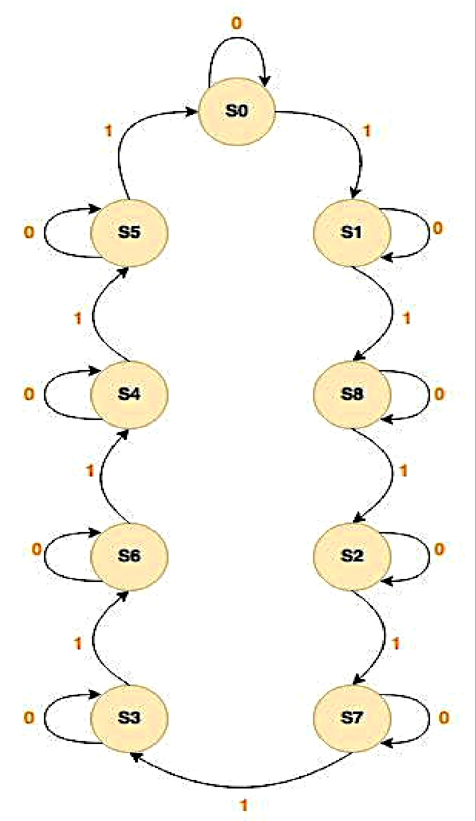
FSM #3



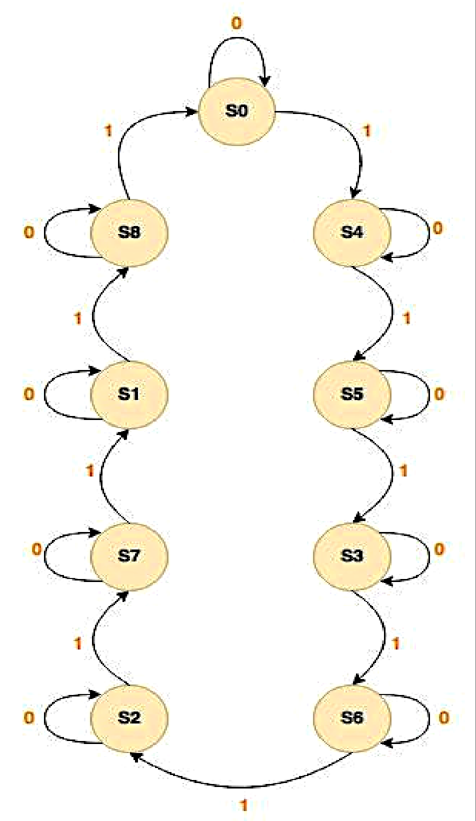
FSM #4

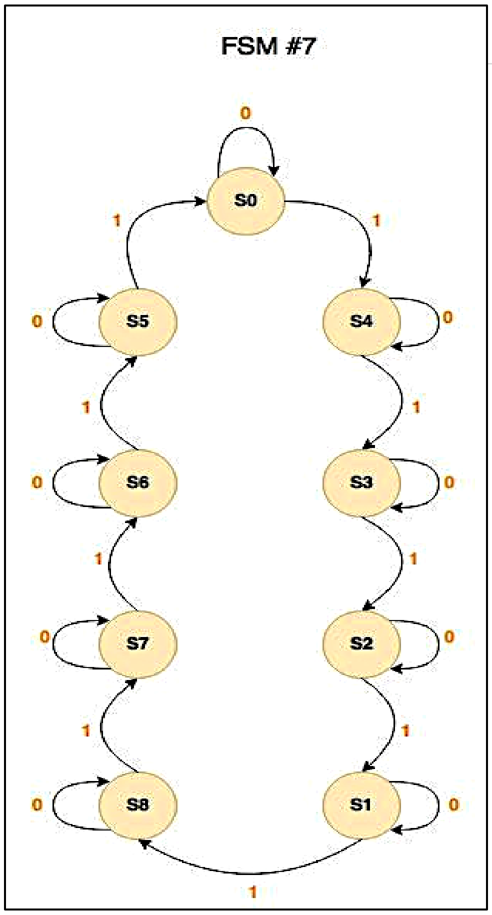


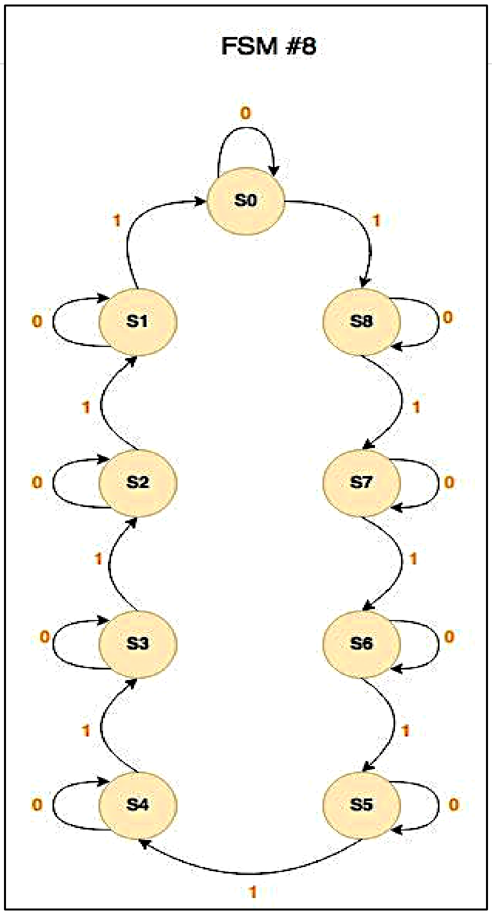
FSM #5

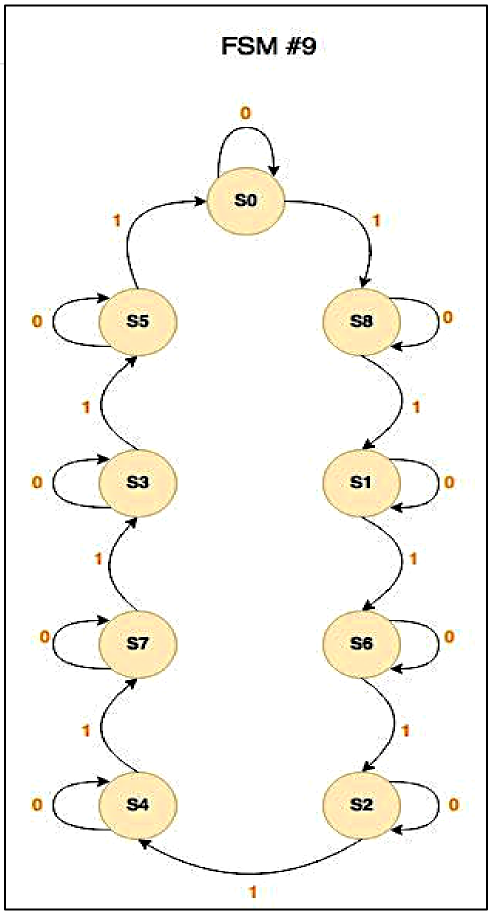


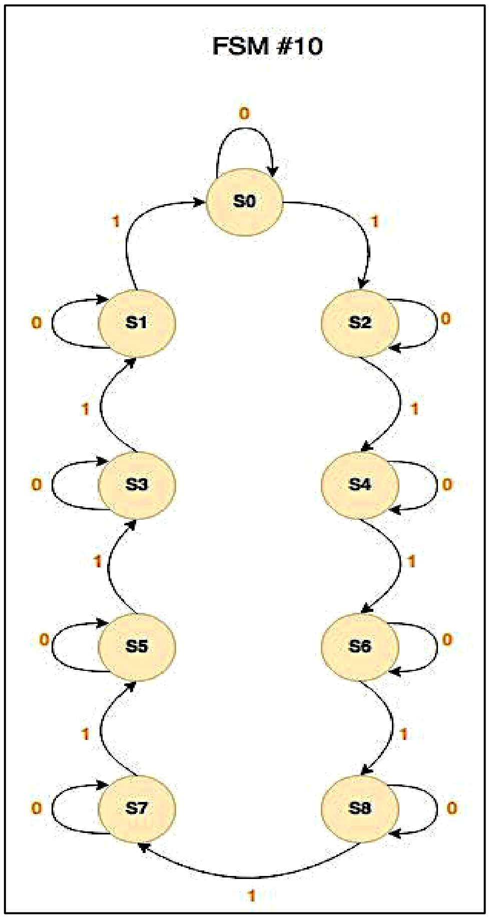
FSM #6

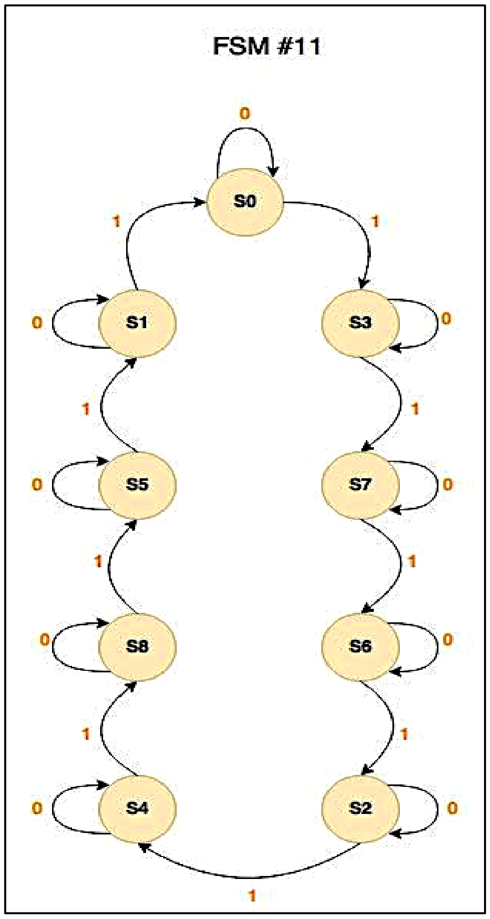












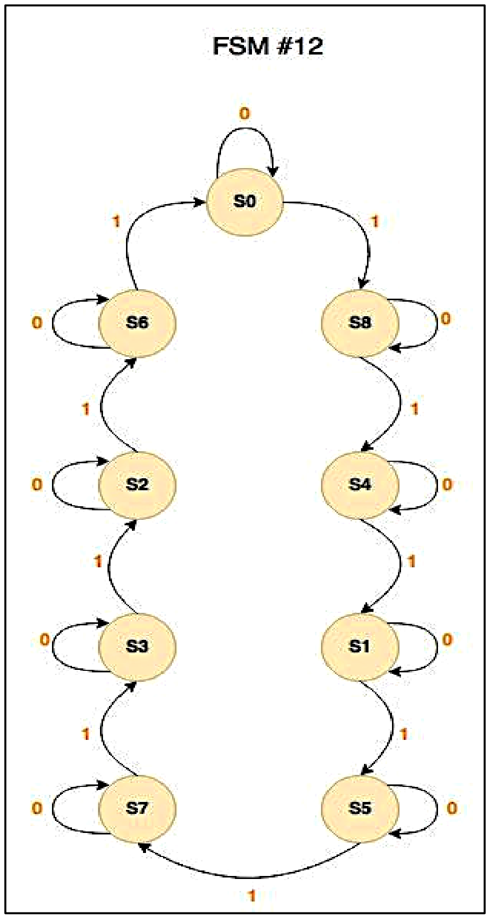
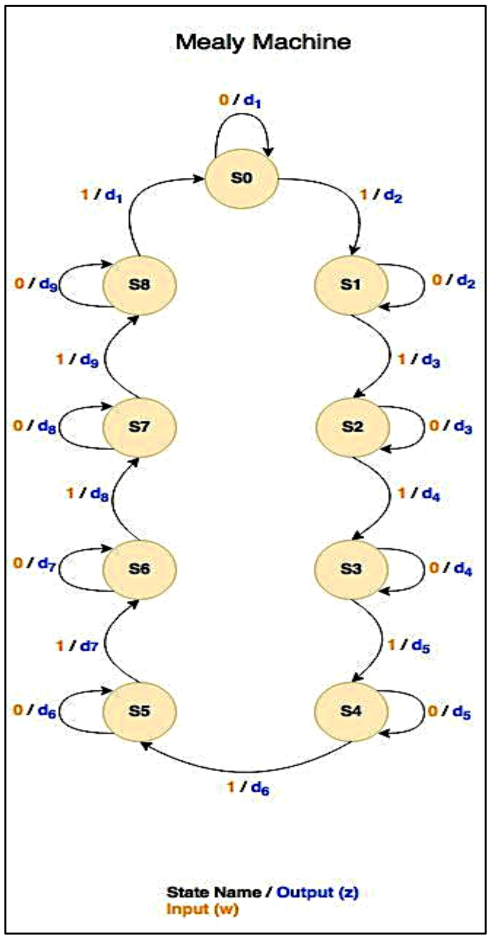


Figure 1 State Diagram Assignments



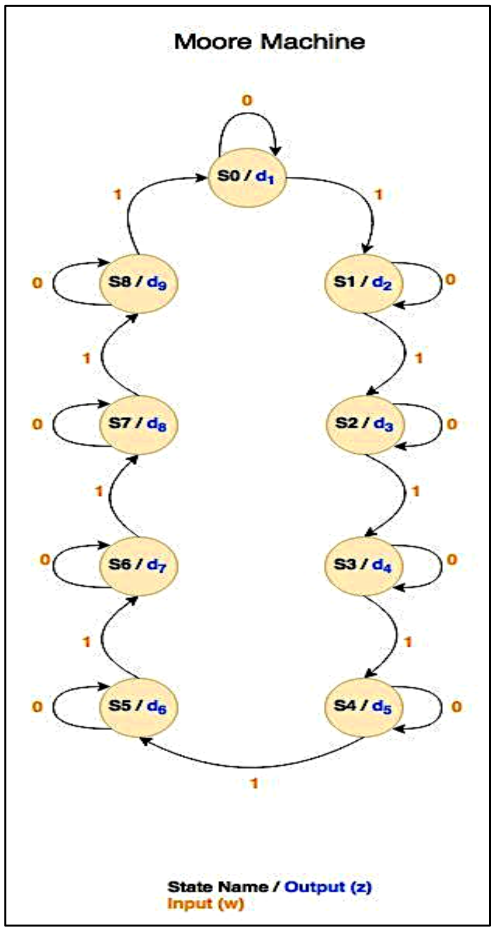


Figure 2 FSM Types for Representing Student Identification Number

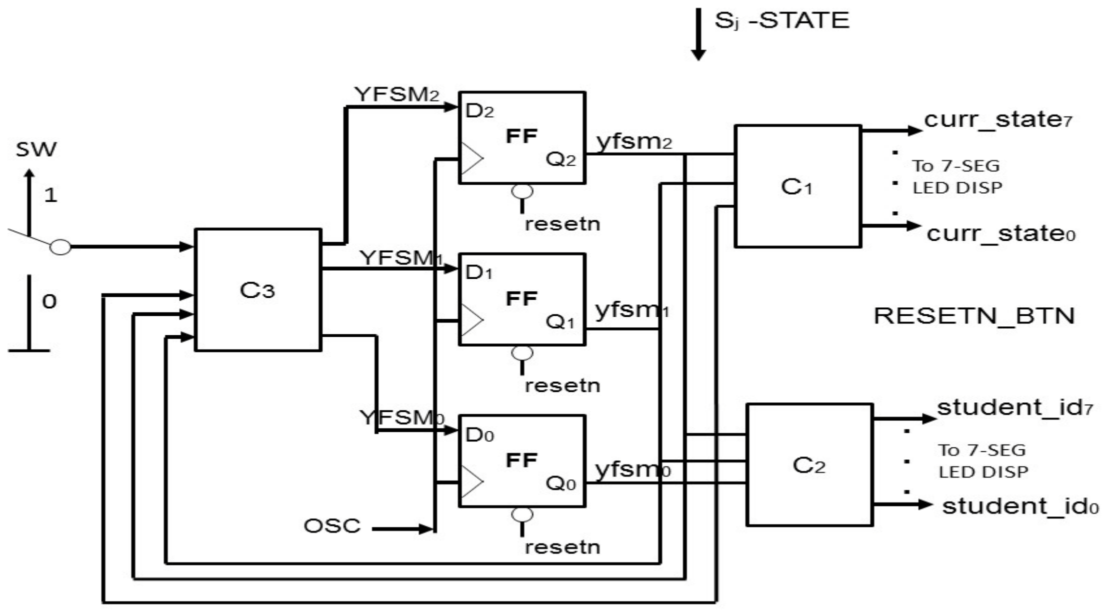


Figure 3 Finite State Machine

```
library ieee;
use ieee.std_logic_1164.all;
entity machine is
    port
    (
        clk        : in   std_logic;
        data_in    : in   std_logic;
        reset      : in   std_logic;
        student_id : out    std_logic_vector(3
downto 0);
        current_state: out  std_logic_vector(3
DOWNTO 0)
    );
end entity;
architecture fsm of machine is
    -- Build an enumerated type with 9 states for
the state machine ( 9 states for parsing 9 digits of
student id)
    type state_type is (s0, s1, s2, s3, s4, s5, s6,
s7, s8);
    -- Register to hold the current state
    signal yfsm : state_type;
begin
    process (clk, reset)
    begin
        if reset = '1' then
            yfsm <= s0;
        elsif (clk'EVENT AND clk = '1') then
            -- Determine the next state
synchronously, based on
            -- the current state and the input
            case yfsm is
                when s0=>

                when s1=>

                when s2=>
                ..........
                ..........
                ..........
                when s8=>

            end case;
        end if;
    end process;
    -- Implement the Moore or Mealy logic here
    process (yfsm, data_in) -- data_in if reqd only
    begin
        case yfsm is
            when s0=>
                ..........
            when s1=>
                ..........
            when s2=>
            ..........
            when s8=>
        end case;   end process;
```

Figure 4 VHDL Code Template

An example of the connections in the block diagram is represented in Figure 5

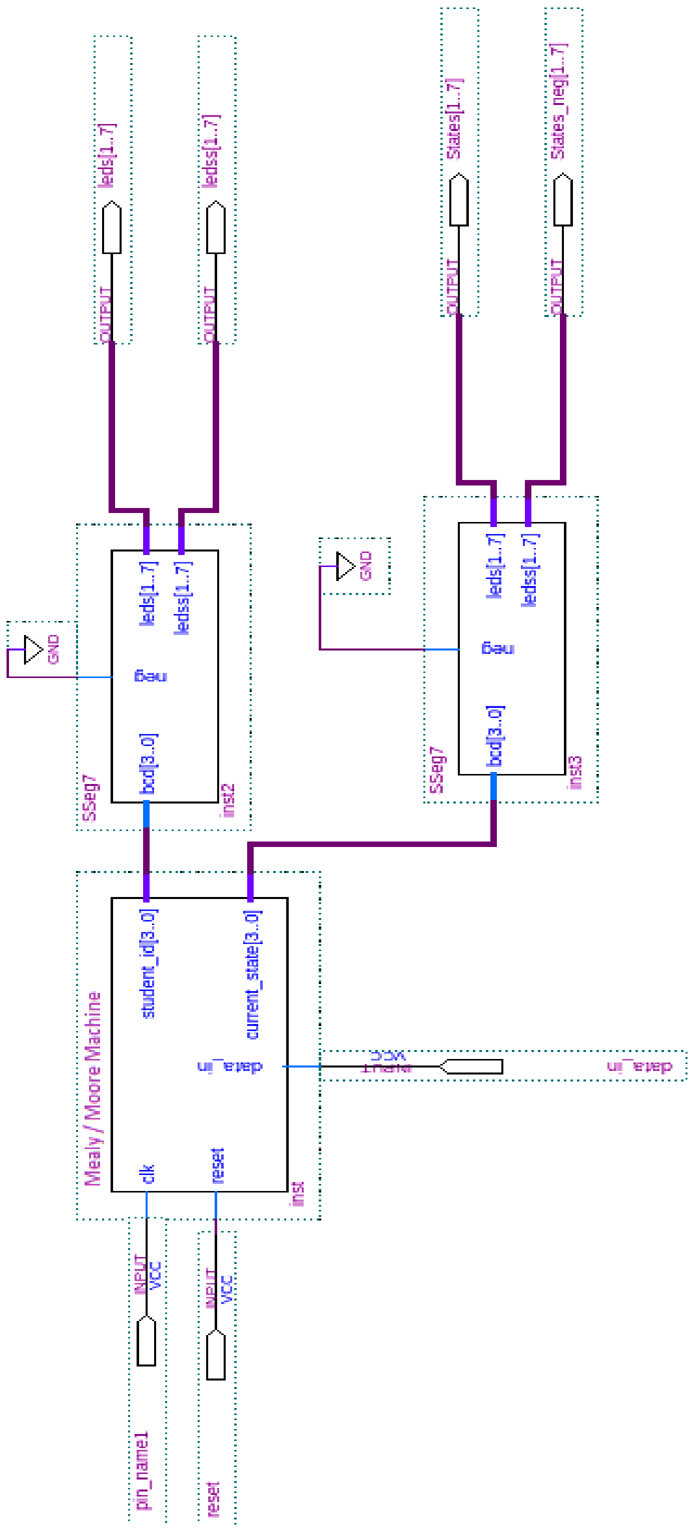


Figure 5 Block Diagram for FSM