BME 328 Lab 6 – VHDL for Sequential Circuits: Implementing a Customized State Machine

15 Marks (1 week)
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 2.
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 *lab6.vhd* to simulate your assigned state machine on the Cyclone- II EP2C35F672C6 FPGA. Each state of your assigned state machine corresponds to a digit of your student ID. For example, state 0 corresponds to the first digit of your student ID, state 1 corresponds to the second digit of your student ID, and so on. (Hint: Use any of the methods represented in Figures 8.29, 8.33, or 8.35 of the text book. You can use the VHDL code template in Figure 2).

3. Laboratory Work

1. Create the subdirectory *lab6* in your work directory, and copy the file *lab6.vhd* to the subdirectory.
2. Compile and simulate your design. Verify that your design is functioning properly through simulation results.
3. Take screenshots of all your Block Diagram, VHDL, and Simulation Waveform Files.

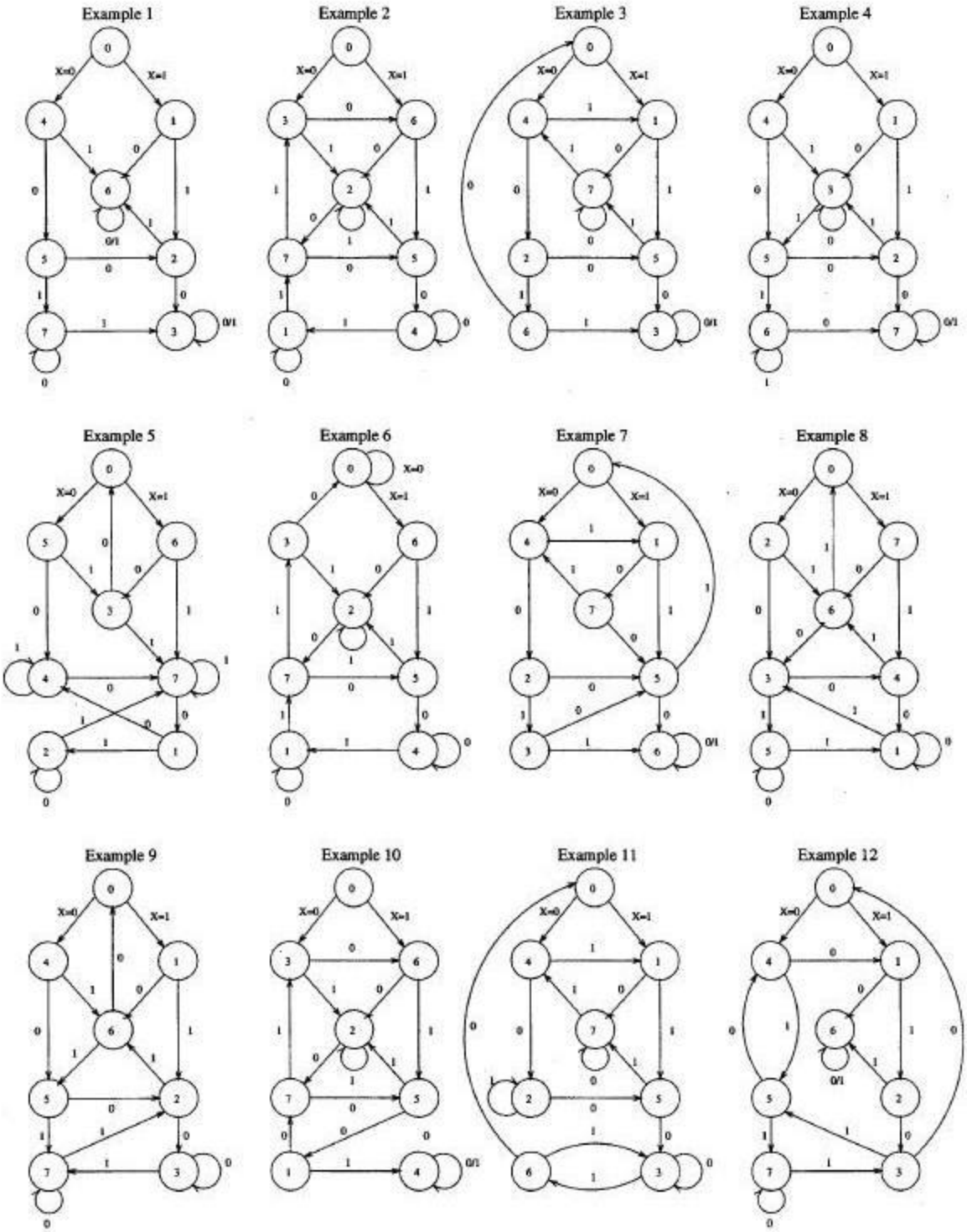

Figure 1: State Diagram Assignments

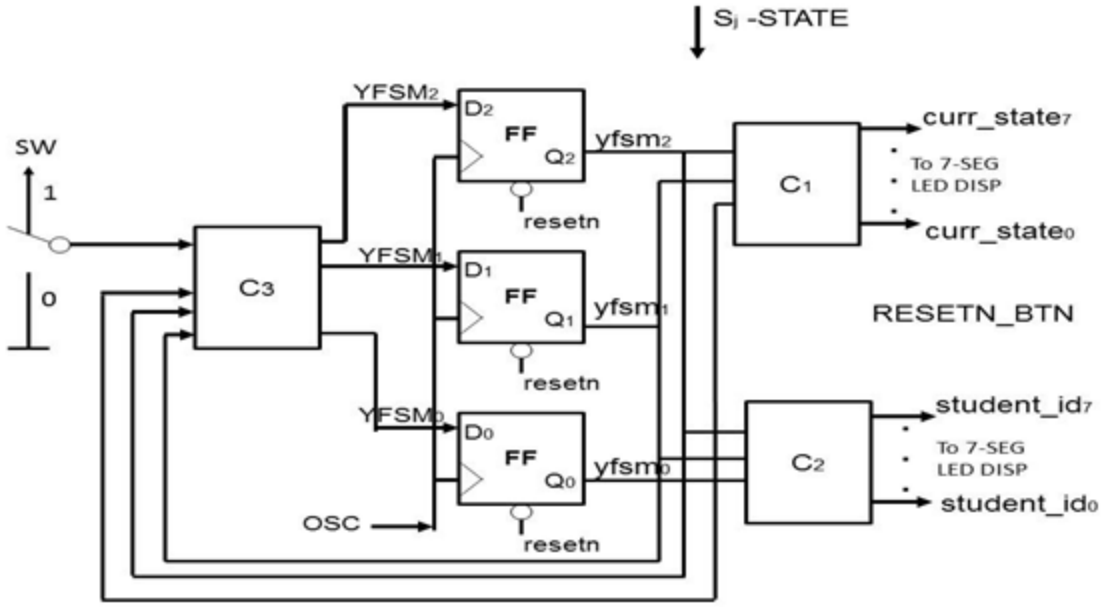

Figure 2: Finite State Machine

```
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 3: VHDL Code Template