

**Department of Physics**

Course Number	COE 328
Course Title	Digital Circuits
Semester/Year	Fall 2021
Instructor	Dr. Reza Sedaghat
TA Name	Sajjad Rostami Sani

Lab/Tutorial Report No.	5
-------------------------	---

Report Title	Lab 5
--------------	-------

Section No.	03
Submission Date	20-11-2021
Due Date	29-11-2021

Student Name	Student ID	Signature*
Ahmad El-Gohary	501011852	<i>Ahmad El-Gohary</i>

*\*By signing above, you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at:*

*<http://www.ryerson.ca/content/dam/senate/policies/pol60.pdf>*

## **Objective**

The objective of this lab was to simulate the operation of a sequential circuit and to design a finite state machine that cycles through the digits of my student number using the assigned state diagram.

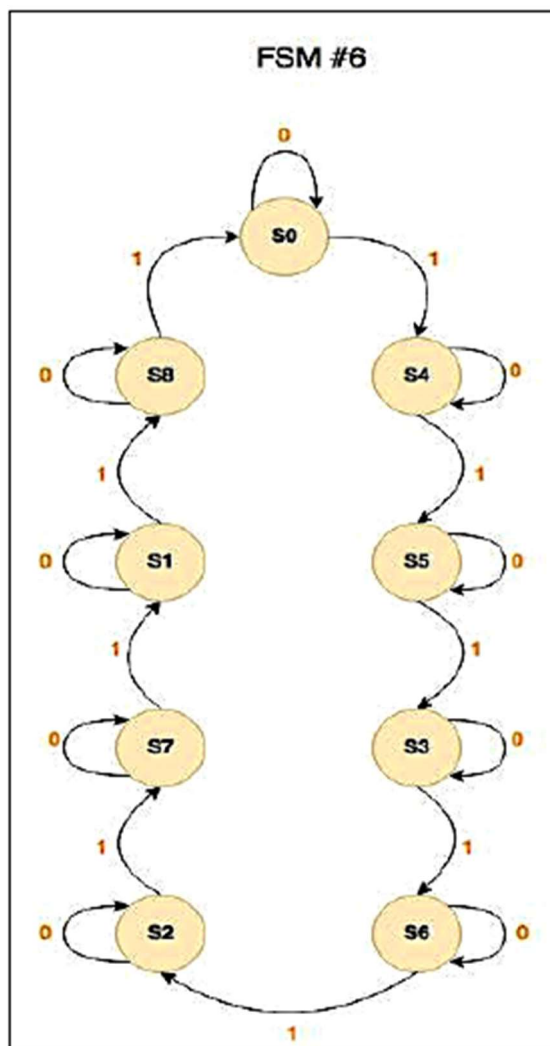
*[This part of page is left blank intentionally]*

## Procedure

- To get the assigned state machine take the last 4 digits of my student number and divide it by the number of states to get the remainder.
- The remainder is the state number

1852/13 has a remainder of 6

Assigned Finite State Machine:



The machine shows that when data in is equal to 0 the state is assigned to itself but when data in is equal to 1 it is assigned to the state to its right for example:

when data in for s0 is 1 s0 is assigned to s4.

Value of s0 to s8 is assigned the value its respective student number digits according to the FSM.

student_id_digit_1	5	0101	s0
student_id_digit_2	0	0000	s4
student_id_digit_3	1	0001	s5
student_id_digit_4	0	0000	s3
student_id_digit_5	1	0001	s6
student_id_digit_6	1	0001	s2
student_id_digit_7	8	1000	s7
student_id_digit_8	5	0101	s1
student_id_digit_9	2	0010	s8

Implementing the previous information into VHDL code provided in the lab

manual page 6:

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  entity lab5 is
4  | port
5  | (
6  |   clk    : in std_logic;
7  |   data_in : in std_logic;
8  |   reset   : in std_logic;
9  |   student_id : out std_logic_vector(3 downto 0);
10 |   current_state: out std_logic_vector(3 DOWNT0 0) );
11 | end entity;
12 | architecture fsm of lab5 is
13 | type state_type is (s0, s1, s2, s3, s4, s5, s6,
14 |   s7, s8);
15 | signal yfsm : state_type;
16 |
17 | begin
18 | process (clk, reset)
19 | | begin
20 | | if reset = '1' then
21 | | | yfsm <= s0;
22 | | elsif (clk 'EVENT AND clk = '1') then
23 | | | case yfsm is
24 | | | |
25 | | | |   when s0 =>
26 | | | |
27 | | | | if data_in = '1' then
28 | | | | | yfsm <= s4;
29 | | | | else yfsm <= s0;
30 | | | | end if;
31 | | | |
32 | | | |   when s1 =>
33 | | | |
34 | | | | if data_in = '1' then
35 | | | | | yfsm <= s8;
```

```

36  else yfsm <= s1;
37  end if;
38  |
39  when s2 =>
40  |
41  if data_in = '1' then
42  |yfsm <= s7;
43  else yfsm <= s2;
44  end if;
45  |
46  when s3 =>
47  |
48  if data_in = '1' then
49  |yfsm <= s6;
50  else yfsm <= s3;
51  end if;
52  |
53  when s4 =>
54  |
55  if data_in = '1' then
56  |yfsm <= s5;
57  else yfsm <= s4;
58  end if;
59  |
60  when s5 =>
61  |
62  if data_in = '1' then
63  |yfsm <= s3;
64  else yfsm <= s5;
65  end if;
66  |
67  when s6 =>
68  |
69  if data_in = '1' then
70  |yfsm <= s2;

```

```

71  else yfsm <= s6;
72  end if;
73
74  when s7 =>
75
76  if data_in = '1' then
77  yfsm <= s1;
78  else yfsm <= s7;
79  end if;
80
81  when s8 =>
82
83  if data_in = '1' then
84  yfsm <= s0;
85  else yfsm <= s8;
86  end if;
87
88  end case;
89  end if;
90  end process;
91  process(yfsm, data_in)
92  begin
93  case yfsm is
94  when s0=>
95      student_id <="0101";
96      current_state <= "0000";
97  when s4 =>
98      student_id <="0000";
99      current_state <= "0100";
100  when s5 =>
101      student_id <="0001";
102      current_state <= "0101";
103  when s3 =>
104      student_id <="0000";
105      current_state <= "0011";

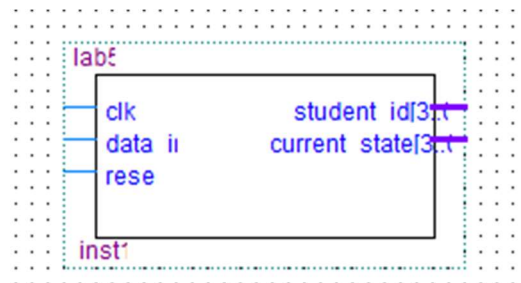
```

```

105 current_state <= "0011";
106 when s6 =>
107     student_id <="0001";
108     current_state <= "0110";
109     when s2 =>
110         student_id <="0001";
111         current_state <= "0010";
112         when s7 =>
113             student_id <="1000";
114             current_state <= "0111";
115             when s1 =>
116                 student_id <="0101";
117                 current_state <= "0001";
118             when s8 =>
119                 student_id <="0010";
120                 current_state <= "1000";
121     end case;
122 end process;
123 end architecture;

```

Converting the previous VHDL code into a block diagram:

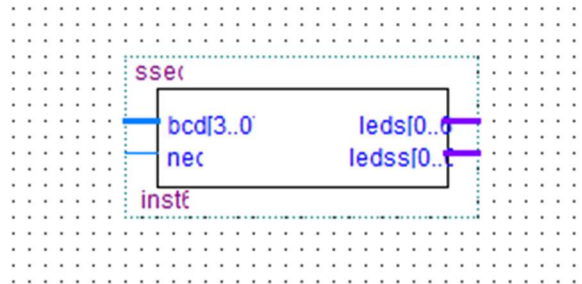




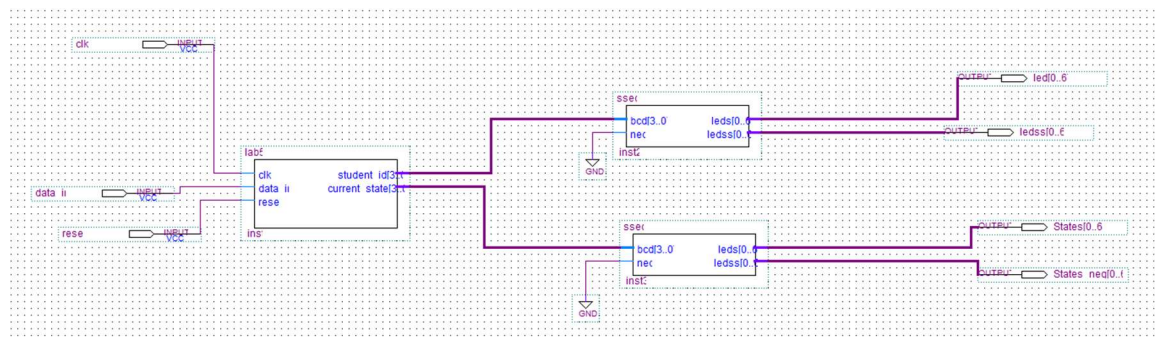
7 segment display code from lab 3:

```
1  LIBRARY ieee ;
2  USE ieee.std_logic_1164.all ;
3
4  ENTITY seg7 IS
5  PORT ( bcd : IN STD_LOGIC_VECTOR(3 DOWNT0 0) ;
6        neg : IN STD_LOGIC ;
7        leds : OUT STD_LOGIC_VECTOR(0 TO 6);
8        ledss: OUT STD_LOGIC_VECTOR(0 TO 6) ) ;
9  END seg7 ;
10
11 ARCHITECTURE Behavior OF seg7 IS
12 BEGIN
13     PROCESS ( bcd )
14     BEGIN
15         CASE bcd IS -- abcdefg
16             WHEN "0000" => leds <= "1111110";
17             WHEN "0001" => leds <= "0110000";
18             WHEN "0010" => leds <= "1101101";
19             WHEN "0011" => leds <= "1111001";
20             WHEN "0100" => leds <= "0110011";
21             WHEN "0101" => leds <= "1011011";
22             WHEN "0110" => leds <= "1011111";
23             WHEN "0111" => leds <= "1110000";
24             WHEN "1000" => leds <= "1111111";
25             WHEN "1001" => leds <= "1110011";
26
27             WHEN "1010"    => leds <= "1110111"; --a
28             WHEN "1011"    => leds <= "0011111"; --b
29             WHEN "1100"    => leds <= "1001110"; --c
30             WHEN "1101"    => leds <= "0111101"; --d
31             WHEN "1110"    => leds <= "1101111"; --e
32             WHEN "1111"    => leds <= "1000111"; --f
33
34         END CASE ;
35     END PROCESS ;
36
37     PROCESS (neg)
38     BEGIN
39         IF (neg = '1') THEN
40             ledss <= "0000001";
41
42         ELSE
43             ledss <= "0000000";
44
45         END IF;
46     END PROCESS;
47 END Behavior ;
```

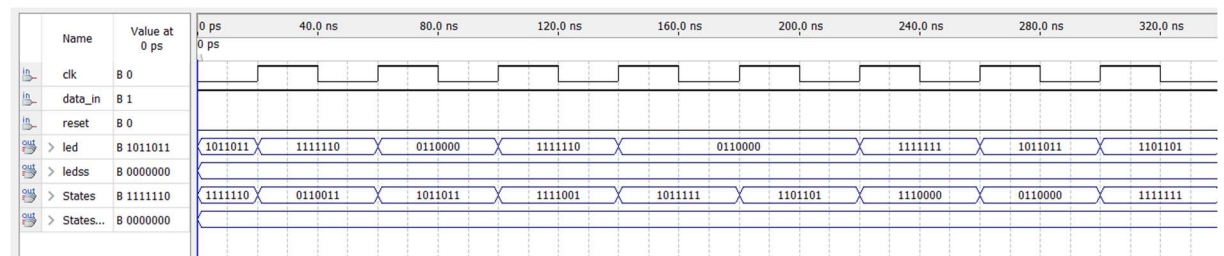
Converting the previous VHDL code into a block diagram:



Connecting the 7-segment display with the FSM in a circuit:



Waveform for the sequential circuit:



## References

Brown, S. D., & Vranesic, Z. G. (2009). *Fundamentals of Digital Logic with VHDL Design*. New York, United States: McGraw-Hill Education.