CSE:215 EDA

Digital Access Control Project 1

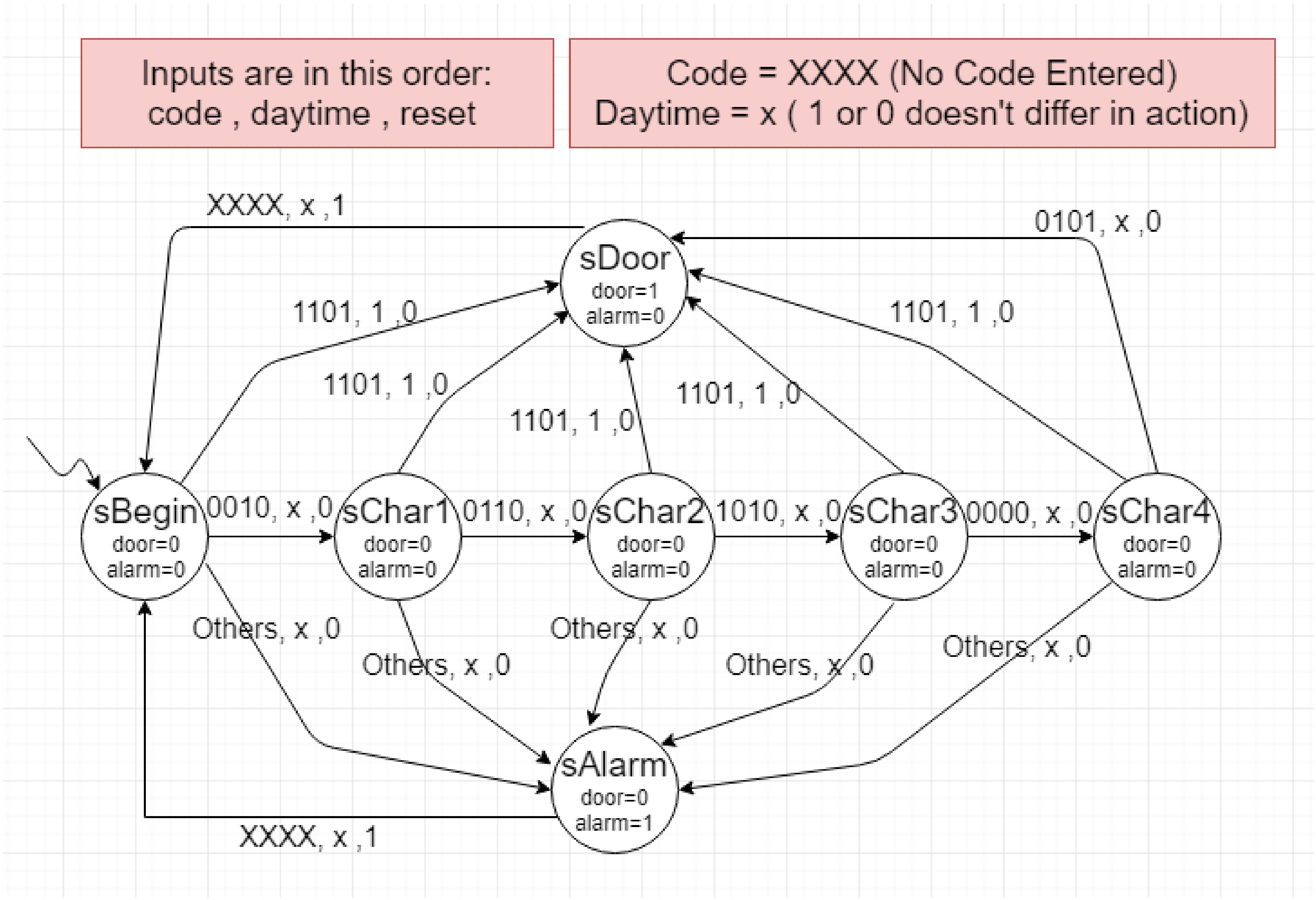
Name: Ahmed Bahaa Ibrahim
ID: 16P6057
Email:ahmedbahaa.6251@gmail.com
Department: CESS

Introduction:

This is a report for Part 1 of the Project of Electronic Design Automation which is about a finite state machine system which is Digital Access Control.

This Report consist of :

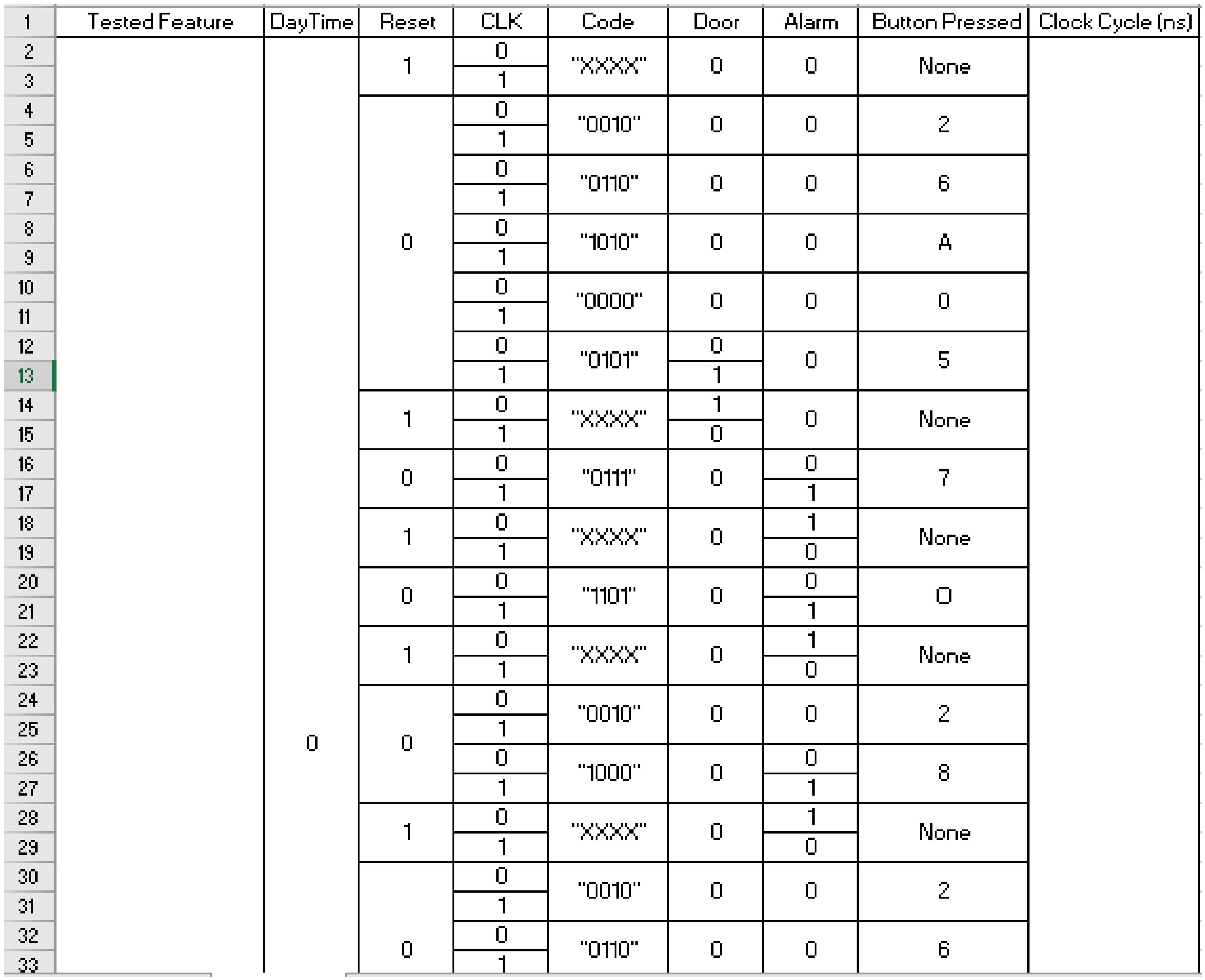
1. FSM State Diagram



2. Architecture Used and Why

I used Moore Architecture , this is because the outputs in my design depend on the states not the inputs , and to keep every output change With the Clock rise. I preferred Moore than Mealy.

3.Testbench Strategy Table

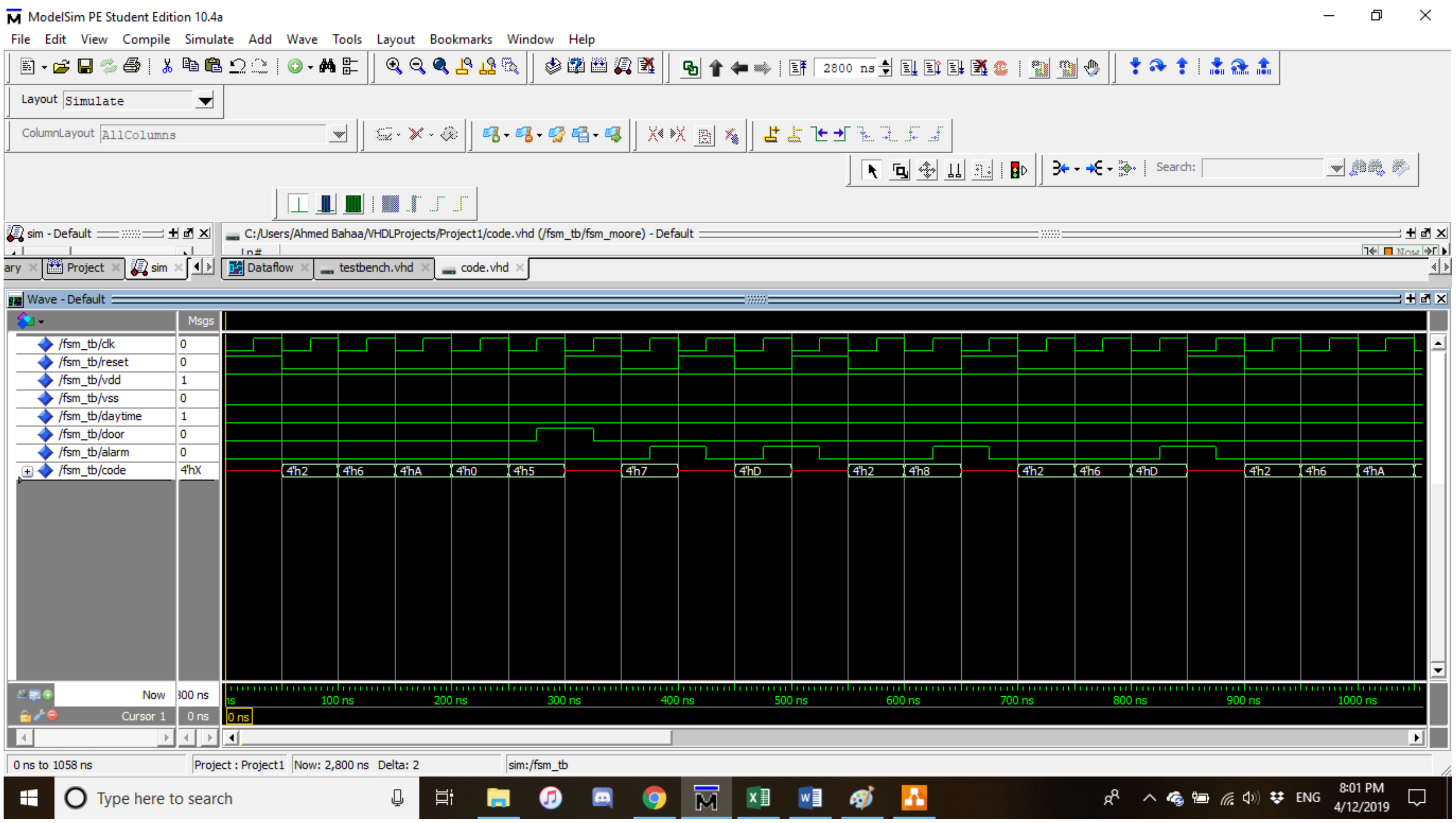


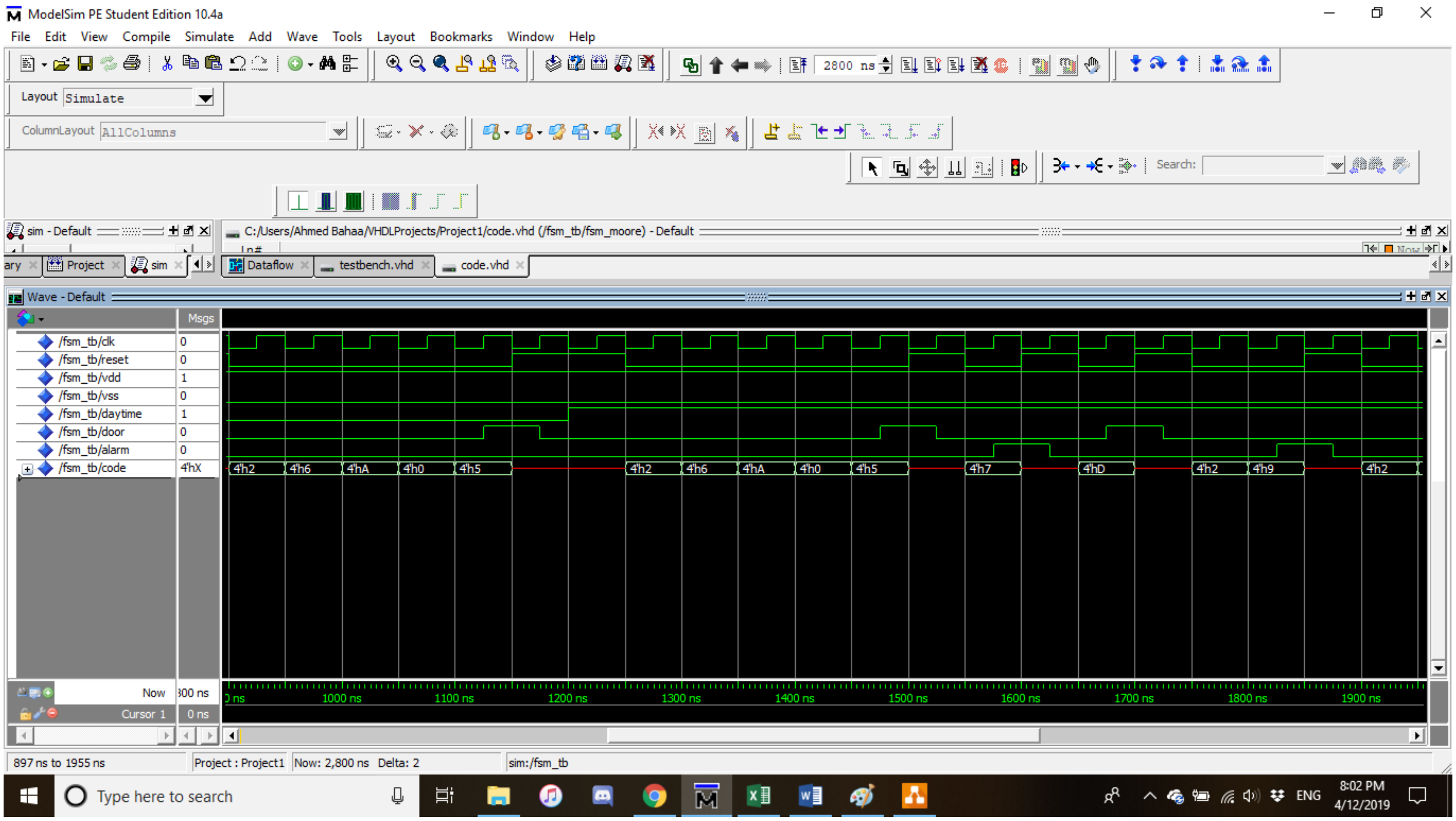
1	Tested Feature	DayTime	Reset	CLK	Code	Door	Alarm	Button Pressed	Clock Cycle (ns)
2		0	1	0	"XXXX"	0	0	None	
3				1					
4			0	0	"0010"	0	0	2	
5				1					
6				0	"0110"	0	0	6	
7				1					
8				0	"1010"	0	0	A	
9				1					
10				0	"0000"	0	0	0	
11				1					
12				0	"0101"	0	0	5	
13				1		1			
14			1	0	"XXXX"	1	0	None	
15				1		0			
16			0	0	"0111"	0	0	7	
17				1			1		
18			1	0	"XXXX"	0	1	None	
19				1			0		
20			0	0	"1101"	0	0	D	
21				1			1		
22			1	0	"XXXX"	0	1	None	
23				1			0		
24			0	0	"0010"	0	0	2	
25				1					
26				0	"1000"	0	0	8	
27				1			1		
28			1	0	"XXXX"	0	1	None	
29				1			0		
30			0	0	"0010"	0	0	2	
31				1					
32				0	"0110"	0	0	6	
33				1					

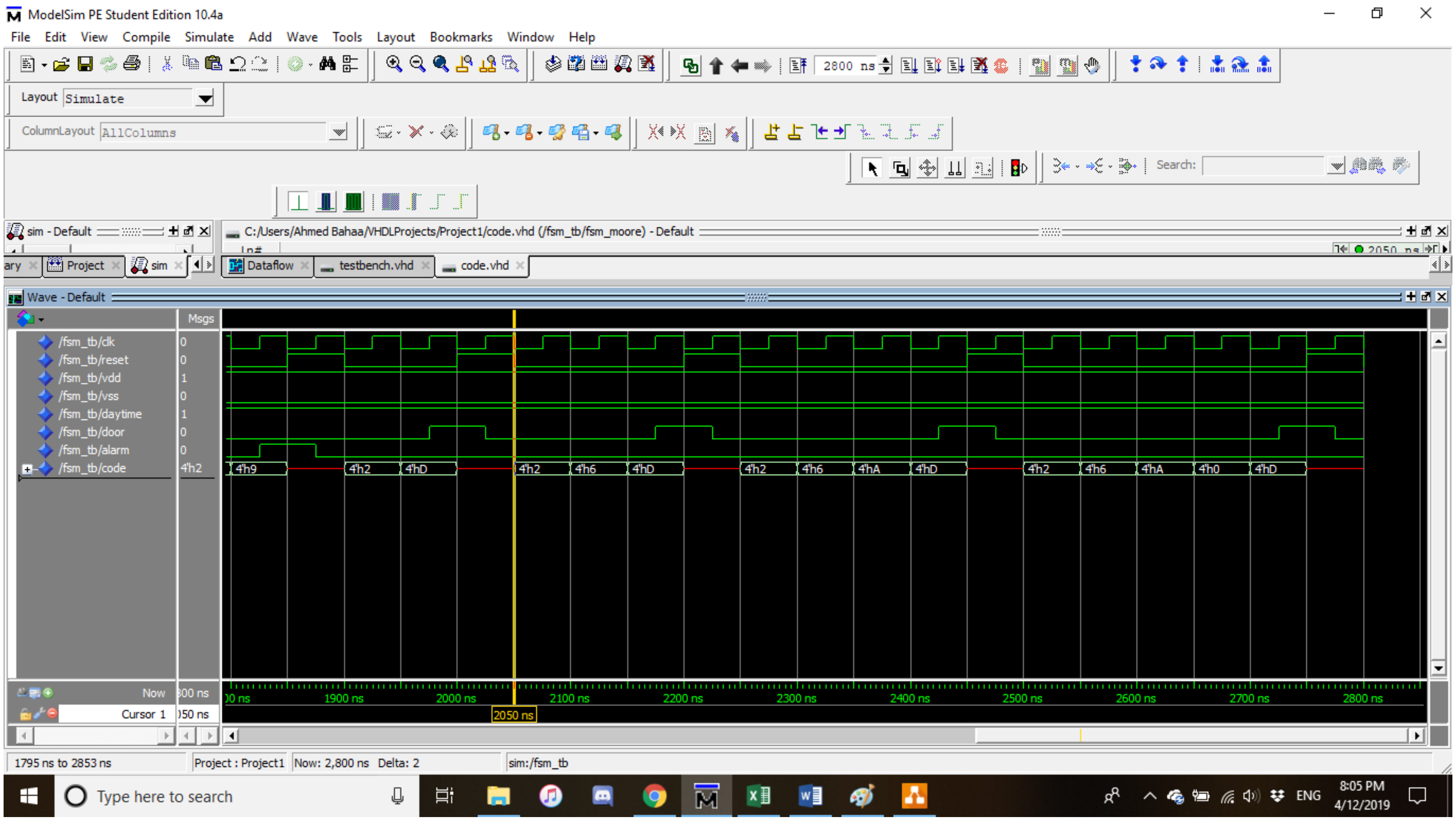
33				1					
34				0	"1101"	0	0	D	
35				1			1		
36			1	0	"XXXX"	0	1	None	
37				1			0		
38			0	0	"0010"	0	0	2	
39				1					
40				0	"0110"	0	0	6	
41				1					
42				0	"1010"	0	0	A	
43				1					
44				0	"0000"	0	0	0	
45				1					
46				0	"0101"	0	0	5	
47				1		1			
48			1	0	"XXXX"	1	0	None	
49				1		0			
50	Digital Access Control		1	0	"XXXX"	0	0	None	50
51				1					
52			0	0	"0010"	0	0	2	
53				1					
54				0	"0110"	0	0	6	
55				1					
56				0	"1010"	0	0	A	
57				1					
58				0	"0000"	0	0	0	
59				1					
60				0	"0101"	0	0	5	
61				1		1			
62			1	0	"XXXX"	1	0	None	
63				1		0			
64			0	0	"0111"	0	0	7	
65				1			1		

65				1			1		
66			1	0	"XXXX"	0	1	None	
67				1			0		
68			0	0	"1101"	0	0	O	
69				1		1			
70			1	0	"XXXX"	1	0	None	
71				1		0			
72			0	0	"0010"	0	0	2	
73				1					
74		1		0	"1101"	0	0	O	
75				1		1			
76			1	0	"XXXX"	1	0	None	
77				1		0			
78			0	0	"0010"	0	0	2	
79				1					
80				0	"0110"	0	0	6	
81				1					
82				0	"1101"	0	0	O	
83				1		1			
84			1	0	"XXXX"	1	0	None	
85				1		0			
86			0	0	"0010"	0	0	2	
87				1					
88				0	"0110"	0	0	6	
89				1					
90				0	"1010"	0	0	A	
91				1					
92				0	"1101"	0	0	O	
93				1		1			
94			1	0	"XXXX"	1	0	None	
95				1		0			
96			0	0	"1101"	0	0	O	
97				1		1			
98			1	0	"XXXX"	1	0	None	
99				1		0	0		

4.Simulation Results




ModelSim PE Student Edition 10.4a
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
2800 ns
Layout Simulate
ColumnLayout AllColumns
Search:
sim - Default
C:/Users/Ahmed Bahaa/VHDLProjects/Project1/code.vhd (/fsm_tb/fsm_moore) - Default
Project sim Dataflow testbench.vhd code.vhd
Wave - Default
Msgs
/fsm_tb/clk 0
/fsm_tb/reset 0
/fsm_tb/vdd 1
/fsm_tb/vss 0
/fsm_tb/daytime 1
/fsm_tb/door 0
/fsm_tb/alarm 0
/fsm_tb/code 4'hX
4'h2 4'h6 4'hA 4'h0 4'h5 4'h2 4'h6 4'hA 4'h0 4'h5 4'h7 4'hD 4'h2 4'h9 4'h2
Now 800 ns
Cursor 1 0 ns
1000 ns 1100 ns 1200 ns 1300 ns 1400 ns 1500 ns 1600 ns 1700 ns 1800 ns 1900 ns
897 ns to 1955 ns
Project : Project1 Now: 2,800 ns Delta: 2
sim:/fsm_tb
Type here to search
ENG 8:02 PM 4/12/2019


ModelSim PE Student Edition 10.4a
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
2800 ns
Layout Simulate
ColumnLayout AllColumns
Search:
sim - Default
C:/Users/Ahmed Bahaa/VHDLProjects/Project1/code.vhd (/fsm_tb/fsm_moore) - Default
Project sim Dataflow testbench.vhd code.vhd
Wave - Default
Msgs
/fsm_tb/clk 0
/fsm_tb/reset 0
/fsm_tb/vdd 1
/fsm_tb/vss 0
/fsm_tb/daytime 1
/fsm_tb/door 0
/fsm_tb/alarm 0
/fsm_tb/code 4'h2
4'h9 4'h2 4'hD 4'h2 4'h6 4'hD 4'h2 4'h6 4'hA 4'hD 4'h2 4'h6 4'hA 4'h0 4'hD
Now 800 ns
Cursor 1 050 ns
1900 ns 2000 ns 2050 ns 2100 ns 2200 ns 2300 ns 2400 ns 2500 ns 2600 ns 2700 ns 2800 ns
1795 ns to 2853 ns
Project : Project1 Now: 2,800 ns Delta: 2
sim:/fsm_tb
Type here to search
ENG 8:05 PM 4/12/2019

5.Appendix

code.vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_arith.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;

entity fsm is
    port(
    clk,vdd,vss,daytime:in bit;
    reset:in bit;
    code:in std_ulogic_vector(3 downto 0);
    door,alarm:out bit
    );
    end entity fsm;
architecture moore of fsm is
    type state_type is(sBegin,sChar1,sChar2,sChar3,sChar4,sDoor,sAlarm);
    signal current_state: state_type;
    signal next_state: state_type;
-- Synthesis directives :
-- pragma current_state current_state
-- pragma next_state next_state
-- pragma clock clk
    begin
        process(clk)
        begin
            if clk='1' and clk'event then
```

```
                current_state <=next_state;
            end if;
            end process;



        process(current_state,code,reset)
        begin
           -- if reset ='1' then
          --      next_state <= sBegin;
           -- else
            --Night Conditions
if daytime ='0' then
            case current_state is
                when sBegin =>
                    door <= '0';
                    alarm <= '0';
                     if reset ='1' then
                             next_state <= sBegin;
           else
                if code= "0010" then
                    next_state <= sChar1;
                else
                     next_state <= sAlarm;
                   end if;
end if;
                   when sChar1 =>
                        door<= '0';
                       alarm <= '0';
```

```
if code= "0110" then
    next_state <= sChar2;
else
    next_state <= sAlarm;
   end if;

   when sChar2 =>
   door<= '0';
   alarm <= '0';
   if code= "1010" then
       next_state <= sChar3;
   else
       next_state <= sAlarm;
      end if;

      when sChar3 =>
      door<= '0';
      alarm <= '0';
      if code= "0000" then
          next_state <= sChar4;
      else
          next_state <= sAlarm;
         end if;

         when sChar4 =>
         door<= '0';
         alarm <= '0';
         if code= "0101" then
```

```
                                   next_state <= sDoor;
                                else
                                   next_state <= sAlarm;
                                  end if;
                                 when sDoor =>
                                 door<= '1';
                                 alarm <= '0';
                                 next_state <= sBegin;
                                 when sAlarm =>
                                 door<= '0';
                                 alarm <= '1';
                                 next_state <= sBegin;
                end case;
          else
                 case current_state is
                      when sBegin =>
                         door<= '0';
                         alarm <= '0';
                      if code= "0010" then
                         next_state <= sChar1;
                          elsif code= "1101" then
                               next_state <= sDoor;
                      else
                          next_state <= sAlarm;
                        end if;
                        when sChar1 =>
                            door<= '0';
                           alarm <= '0';
```

```
  if code= "0110" then
      next_state <= sChar2;
elsif code= "1101" then
                 next_state <= sDoor;
  else
      next_state <= sAlarm;
     end if;
     when sChar2 =>
     door<= '0';
     alarm <= '0';
     if code= "1010" then
         next_state <= sChar3;
       elsif code= "1101" then
                      next_state <= sDoor;
     else
         next_state <= sAlarm;
        end if;
        when sChar3 =>
        door<= '0';
        alarm <= '0';
        if code= "0000" then
            next_state <= sChar4;
       elsif code= "1101" then
                      next_state <= sDoor;
        else
            next_state <= sAlarm;
           end if;
           when sChar4 =>
```

```
                        door<= '0';
                        alarm <= '0';
                        if code= "0101" then
                           next_state <= sDoor;
                        elsif code= "1101" then
                                    next_state <= sDoor;
                        else
                           next_state <= sAlarm;
                          end if;
                         when sDoor =>
                         door<= '1';
                         alarm <= '0';
                               if reset ='1' then
                         next_state <= sBegin;
                         end if;
                         when sAlarm =>
                         door<= '0';
                         alarm <= '1';
                               if reset ='1' then
                         next_state <= sBegin;
                                   end if;
          end case;
    end if;
          end process;
       end architecture moore;
```

testbench.vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_arith.ALL;
use IEEE.STD_LOGIC_unsigned.ALL;

entity fsm_tb is
    end entity fsm_tb;

    architecture tb_arch of fsm_tb is
        component fsm is
            port(
                clk,vdd,vss,daytime:in bit;
                     reset:in bit;
                code:in std_ulogic_vector(3 downto 0);
                door,alarm:out bit
                );
            end component fsm;
            signal clk:bit :='0';
            signal reset:bit := '1';
            signal vdd:bit := '1';
            signal vss:bit := '0';
            signal daytime:bit := '0';
            signal door:bit := '0';
            signal alarm: bit := '0';
            signal code:std_ulogic_vector(3 downto 0) := "XXXX";
            constant clk_period: time:=50 ns;
            for fsm_moore: fsm use entity work.fsm(moore);
```

```
          begin
             process is
             begin
                clk <= '0';
                wait for clk_period/2;
                clk <= '1';
                wait for clk_period/2;
                end process;

             process is
             begin
daytime<='0';
  --Night
        code<="XXXX";
  wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
  reset<='0';
                code<="0010";
                wait for 50 ns;
                assert door='0' and alarm='0'
                report "there is error"
                severity warning;
                code<="0110";
                wait for 50 ns;
                assert door='0' and alarm='0'
```

```
                    report "there is error"
                    severity warning;
                    code<="1010";
                    wait for 50 ns;
                    assert door='0' and alarm='0'
                    report "there is error"
                    severity warning;
                    code<="0000";
                    wait for 50 ns;
                    assert door='0' and alarm='0'
                    report "there is error"
                    severity warning;
                    code<="0101";
                    wait for 50 ns;
                    assert door='1' and alarm='0'
                    report "Door Must Open"
                    severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
   assert door='0' and alarm='0'
   report "Reset error"
   severity warning;
reset<='0';
                    code<="0111";
                    wait for 50 ns;
                    assert door='0' and alarm='1'
                    report "Alarm Must Ring"
```

```
             severity warning;

reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
             code<="1101";
             wait for 50 ns;
             assert door='0' and alarm='1'
             report "Alarm Must Ring"
             severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
             code<="0010";
             wait for 50 ns;
             assert door='0' and alarm='0'
             report "there is error"
             severity warning;
             code<="1000";
             wait for 50 ns;
```

```
                    assert door='0' and alarm='1'
                    report "Alarm Must Ring"
                    severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
   assert door='0' and alarm='0'
   report "Reset error"
   severity warning;
reset<='0';
                    code<="0010";
                    wait for 50 ns;
                    assert door='0' and alarm='0'
                    report "there is error"
                    severity warning;
                    code<="0110";
                    wait for 50 ns;
                    assert door='0' and alarm='0'
                    report "there is error"
                    severity warning;
                    code<="1101";
                    wait for 50 ns;
                    assert door='0' and alarm='1'
                    report "Alarm Must Ring"
                    severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
```

```
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
             code<="0010";
             wait for 50 ns;
             assert door='0' and alarm='0'
             report "there is error"
             severity warning;
             code<="0110";
             wait for 50 ns;
             assert door='0' and alarm='0'
             report "there is error"
             severity warning;
             code<="1010";
             wait for 50 ns;
             assert door='0' and alarm='0'
             report "there is error"
             severity warning;
             code<="0000";
             wait for 50 ns;
             assert door='0' and alarm='0'
             report "there is error"
             severity warning;
             code<="0101";
             wait for 50 ns;
             assert door='1' and alarm='0'
             report "Door Must Open"
```

```
                    severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
   assert door='0' and alarm='0'
   report "Reset error"
   severity warning;
reset<='0';

   daytime<='1';
    reset<='1';
code<="XXXX";
--Morning
wait for 50 ns;
assert door='0' and alarm='0'
report "Reset error"
severity warning;
reset<='0';
                 code<="0010";
                 wait for 50 ns;
                 assert door='0' and alarm='0'
                 report "there is error"
                 severity warning;
                 code<="0110";
                 wait for 50 ns;
                 assert door='0' and alarm='0'
                 report "there is error"
                 severity warning;
```

```
            code<="1010";
            wait for 50 ns;
            assert door='0' and alarm='0'
            report "there is error"
            severity warning;
            code<="0000";
            wait for 50 ns;
            assert door='0' and alarm='0'
            report "there is error"
            severity warning;
            code<="0101";
            wait for 50 ns;
            assert door='1' and alarm='0'
            report "Door Must Open"
            severity warning;

reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';

            code<="0111";
            wait for 50 ns;
            assert door='0' and alarm='1'
            report "Alarm Must Ring"
```

```
                severity warning;

reset<='1';
code<="XXXX";
wait for 50 ns;
   assert door='0' and alarm='0'
   report "Reset error"
   severity warning;
reset<='0';
                code<="1101";
                wait for 50 ns;
                assert door='1' and alarm='0'
                report "Door Must Open"
                severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
   assert door='0' and alarm='0'
   report "Reset error"
   severity warning;
reset<='0';
                code<="0010";
                wait for 50 ns;
                assert door='0' and alarm='0'
                report "there is error"
                severity warning;
                code<="1001";
                wait for 50 ns;
```

```
           assert door='0' and alarm='1'
           report "Alarm Must Ring"
           severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
code<="0010";
wait for 50 ns;
assert door='0' and alarm='0'
report "there is error"
severity warning;
code<="1101";
wait for 50 ns;
assert door='1' and alarm='0'
report "Door Must Open"
severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
           code<="0010";
```

```
                wait for 50 ns;
                assert door='0' and alarm='0'
                report "there is error"
                severity warning;
                code<="0110";
                wait for 50 ns;
                assert door='0' and alarm='0'
                report "there is error"
                severity warning;
                code<="1101";
wait for 50 ns;
assert door='1' and alarm='0'
report "Door Must Open"
severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
                code<="0010";
                wait for 50 ns;
                assert door='0' and alarm='0'
                report "there is error"
                severity warning;
                code<="0110";
                wait for 50 ns;
```

```
                 assert door='0' and alarm='0'
                 report "there is error"
                 severity warning;
                 code<="1010";
                 wait for 50 ns;
                 assert door='0' and alarm='0'
                 report "there is error"
                 severity warning;
                 code<="1101";
                 wait for 50 ns;
                 assert door='1' and alarm='0'
                 report "Door Must Open"
                 severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
   assert door='0' and alarm='0'
   report "Reset error"
   severity warning;
reset<='0';
                code<="0010";
                 wait for 50 ns;
                 assert door='0' and alarm='0'
                 report "there is error"
                 severity warning;
                 code<="0110";
                 wait for 50 ns;
                 assert door='0' and alarm='0'
```

```
                    report "there is error"
                    severity warning;
                    code<="1010";
                    wait for 50 ns;
                    assert door='0' and alarm='0'
                    report "there is error"
                    severity warning;
                        code<="0000";
                    wait for 50 ns;
                    assert door='0' and alarm='0'
                    report "there is error"
                    severity warning;
                    code<="1101";
                    wait for 50 ns;
                    assert door='1' and alarm='0'
                    report "Door Must Open"
                    severity warning;
reset<='1';
code<="XXXX";
wait for 50 ns;
  assert door='0' and alarm='0'
  report "Reset error"
  severity warning;
reset<='0';
                      wait;
                      end process;
                      fsm_moore: fsm port map(clk,vdd,vss,daytime,reset,code,door,alarm);
      end architecture tb_arch;
```