Ain Shams University

Faculty of Engineering

CHEP- CESS

Mohammed Ehab Elsaeed

16P8160

m.elsaeed1998@gmail.com

Electronic Design Automation

CSE 215

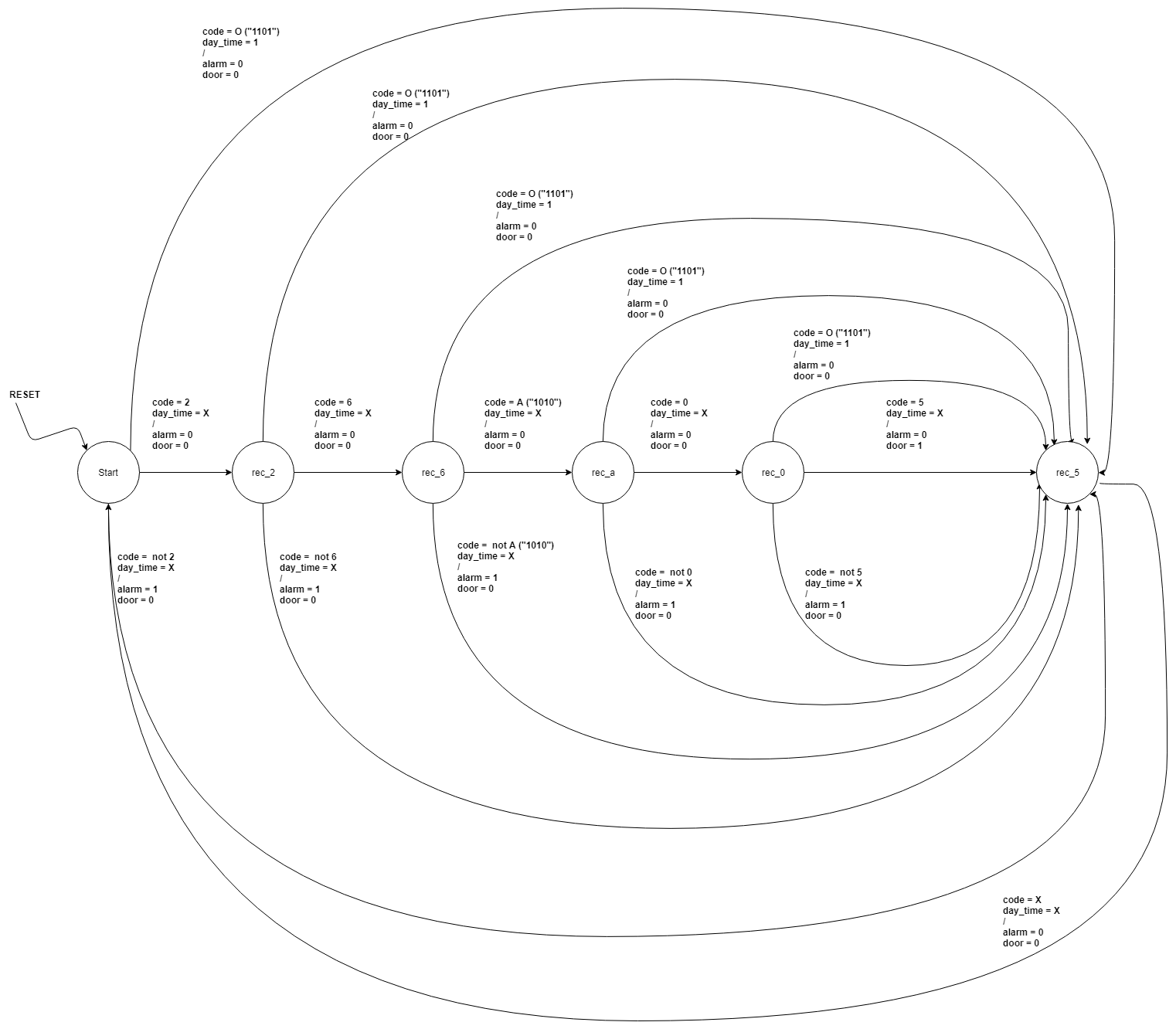
Digital Access Control Finite State Machine Project 1

Introduction

The purpose of this document is to illustrate the work done to create a digital access control finite state machine in accordance to the specifications provided.

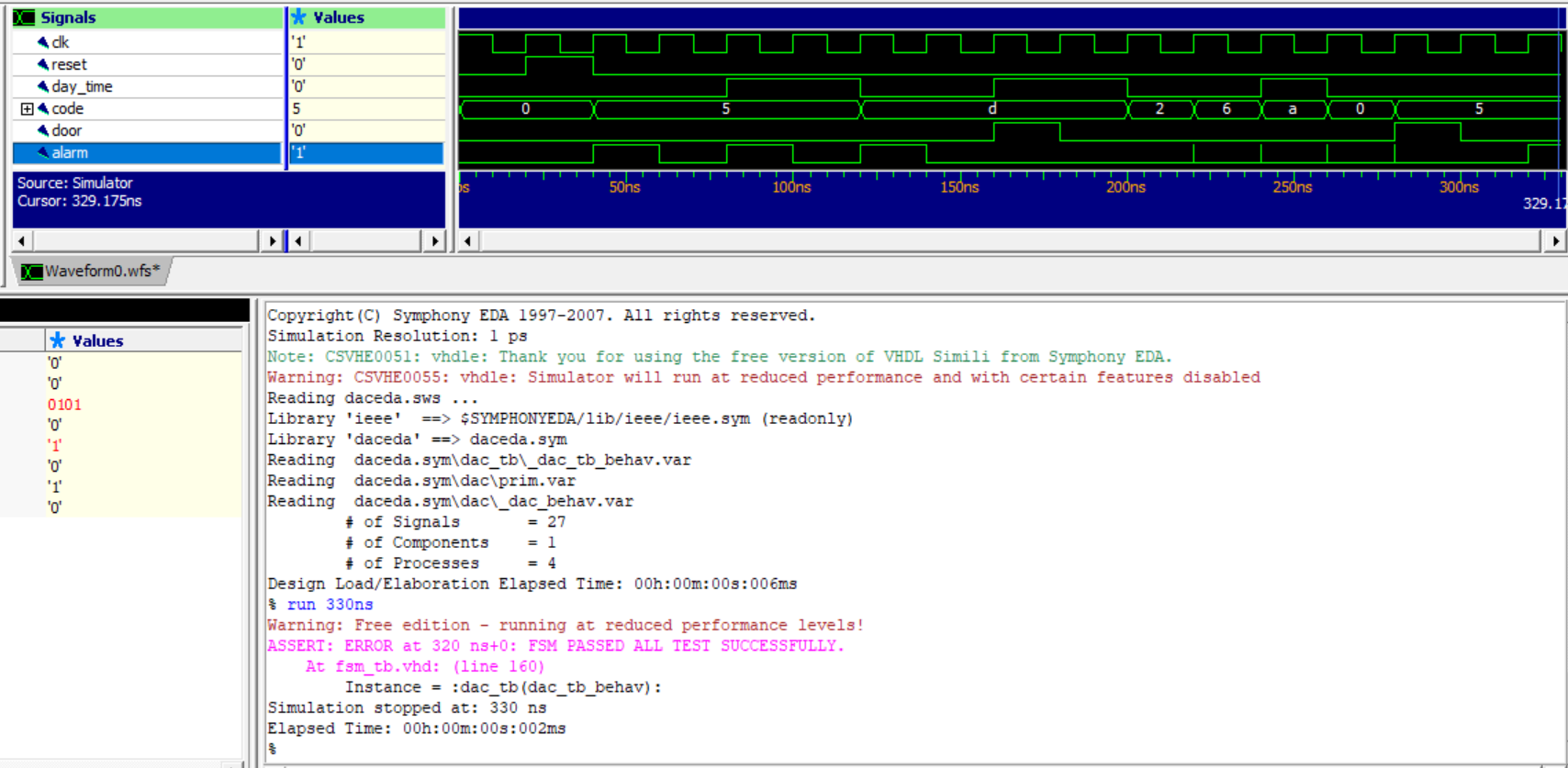
The finite state machine was implemented as a **Mealy** machine in an attempt to reduce the number of states due to the fact that outputs depend on transitions instead of states. As a sacrifice, the finite state machine is supposedly less stable than a Moore machine.

Table of Contents

1. Finite State Machine Diagram
2. Testbench Strategy
3. Simulation of The Finite State Machine
4. Finite State Machine VHDL Code
5. Finite State Machine VHDL Testbench Code
6. Finite State Machine State Diagram
7. Testbench Strategy

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| reset | day\_time | code |  | door | alarm | Justification |
| 1 | X | X |  | 0 | 0 | Setting reset = 1 should reset the circuit to its initial state |
| 0 | 0 | 5 |  | 0 | 1 | Entering a wrong code,  should trigger the alarm |
| 0 | **X** | **X** |  | **0** | **0** | **Waiting for a clock cycle, should reset both door and alarm, to zero** |
| 0 | 1 | 5 |  | 0 | 1 | Entering a wrong code, even during daytime  should trigger the alarm |
| 0 | **X** | **X** |  | **0** | **0** | **Waiting for a clock cycle, should reset both door and alarm, to zero** |
| 0 | 0 | "O" |  | 0 | 1 | Entering "O" during night time  should trigger the alarm |
| 0 | **X** | **X** |  | **0** | **0** | **Waiting for a clock cycle, should reset both door and alarm, to zero** |
| 0 | 1 | "O" |  | 1 | 0 | Entering "O" during daytime should open the door |
| 0 | **X** | **X** |  | **0** | **0** | **Waiting for a clock cycle, should reset both door and alarm, to zero** |
| 0 | 0 | 2 |  | 0 | 0 | Entering 2 neither opens the door, nor triggers the alarm |
| 0 | 0 | 6 |  | 0 | 0 | Entering 2, 6 neither opens the door, nor triggers the alarm |
| 0 | 1 | "A" |  | 0 | 0 | Entering 2, 6, A neither opens the door, nor triggers the alarm, switching daytime to 1 and not pressing "O", will not open the door nor trigger the alarm |
| 0 | 0 | 0 |  | 0 | 0 | Entering 2, 6, A, 0 neither opens the door, nor triggers the alarm |
| 0 | 0 | 5 |  | 1 | 0 | Entering 2, 6, A, 0, 5 opens the door, but doesn't trigger the alarm |
| 0 | **X** | **X** |  | **0** | **0** | **Waiting for a clock cycle, should reset both door and alarm, to zero** |

1. Simulation



Note: “d” is the hexadecimal code for “1101” (Which is the **code** of input **O**)

Glitches in the alarm signal are due to the fact of using a **Mealy machine.**

The system **passes** all the assertions.

1. Finite State Machine VHDL Code
2. -- 0 to 9 binary encoded
3. -- A => 1010
4. -- B => 1011
5. -- O => 1101
6. entity dac is
7. port (
8. reset : in bit;
9. day\_time : in bit;
10. code : in bit\_vector(3 downto 0);
11. door : out bit;
12. alarm : out bit;
13. clk : in bit;
14. vdd : in bit;
15. vss : in bit
16. );
17. end dac;
18. architecture dac\_behav of dac is
19. type state is (start, rec\_2, rec\_6, rec\_a, rec\_0, rec\_5);
20. signal current\_state : state;
21. signal next\_state : state;
22. constant a : bit\_vector(3 downto 0) := "1010";
23. constant b : bit\_vector(3 downto 0) := "1011";
24. constant o : bit\_vector(3 downto 0) := "1101";
25. --pragma current\_state current\_state
26. --pragma next\_state next\_state
27. --pragma clock clk
28. begin
29. process (clk)
30. begin
31. if clk = '1' and not clk'stable then
32. current\_state <= next\_state;
33. end if;
34. end process;
35. process (current\_state, reset, day\_time, code)
36. begin
37. if reset = '1' then
38. next\_state <= start;
39. else
40. case current\_state is
41. when start =>
42. if day\_time = '1' and code = o then
43. door <= '1';
44. alarm <= '0';
45. next\_state <= rec\_5;
46. elsif code = x"2" then
47. door <= '0';
48. alarm <= '0';
49. next\_state <= rec\_2;
50. else
51. door <= '0';
52. alarm <= '1';
53. next\_state <= rec\_5;
54. end if;
55. when rec\_2 =>
56. if day\_time = '1' and code = o then
57. door <= '1';
58. alarm <= '0';
59. next\_state <= rec\_5;
60. elsif code = x"6" then
61. door <= '0';
62. alarm <= '0';
63. next\_state <= rec\_6;
64. else
65. door <= '0';
66. alarm <= '1';
67. next\_state <= rec\_5;
68. end if;
69. when rec\_6 =>
70. if day\_time = '1' and code = o then
71. door <= '1';
72. alarm <= '0';
73. next\_state <= rec\_5;
74. elsif code = a then
75. door <= '0';
76. alarm <= '0';
77. next\_state <= rec\_a;
78. else
79. door <= '0';
80. alarm <= '1';
81. next\_state <= rec\_5;
82. end if;
83. when rec\_a =>
84. if day\_time = '1' and code = o then
85. door <= '1';
86. alarm <= '0';
87. next\_state <= rec\_5;
88. elsif code = x"0" then
89. door <= '0';
90. alarm <= '0';
91. next\_state <= rec\_0;
92. else
93. door <= '0';
94. alarm <= '1';
95. next\_state <= rec\_5;
96. end if;
97. when rec\_0 =>
98. if day\_time = '1' and code = o then
99. door <= '1';
100. alarm <= '0';
101. next\_state <= rec\_5;
102. elsif code = x"5" then
103. door <= '1';
104. alarm <= '0';
105. next\_state <= rec\_5;
106. else
107. door <= '0';
108. alarm <= '1';
109. next\_state <= rec\_5;
110. end if;
111. when rec\_5 =>
112. door <= '0';
113. alarm <= '0';
114. next\_state <= start;
115. when others =>
116. assert false
117. report "Invalid state"
118. severity failure;
119. end case;
120. end if;
121. end process;
122. end dac\_behav;
123. Testbench VHDL Code
124. entity dac\_tb is
125. end dac\_tb;
126. architecture dac\_tb\_behav of dac\_tb is
127. component dac is
128. port (
129. reset : in bit;
130. day\_time : in bit;
131. code : in bit\_vector(3 downto 0);
132. door : out bit;
133. alarm : out bit;
134. clk : in bit;
135. vdd : in bit;
136. vss : in bit
137. );
138. end component dac;
139. signal reset : bit;
140. signal day\_time : bit;
141. signal code : bit\_vector(3 downto 0);
142. signal door : bit;
143. signal alarm : bit;
144. signal clk : bit;
145. signal vdd : bit := '1';
146. signal vss : bit := '0';
147. for all : dac use entity work.dac(dac\_behav);
148. constant clk\_period : time := 20 ns;
149. constant a : bit\_vector(3 downto 0) := "1010";
150. constant b : bit\_vector(3 downto 0) := "1011";
151. constant o : bit\_vector(3 downto 0) := "1101";
152. begin
153. dut : dac port map(reset, day\_time, code, door, alarm, clk, vdd, vss);
154. process begin
155. wait for clk\_period;
156. -- 1
157. reset <= '1';
158. -- day\_time<='0';
159. -- code <= x"";
160. wait for clk\_period;
161. assert door = '0' and alarm = '0'
162. report "Setting reset = 1 should reset the circuit to its initial state"
163. severity failure;
164. -- 2
165. reset <= '0';
166. day\_time <= '0';
167. code <= x"5";
168. wait for clk\_period;
169. assert door = '0' and alarm = '1'
170. report "Entering a wrong code, should trigger the alarm"
171. severity failure;
172. -- 3
173. wait for clk\_period;
174. assert door = '0' and alarm = '0'
175. report "Waiting for a clock cycle, should reset both door and alarm, to zero"
176. severity failure;
177. -- 4
178. reset <= '0';
179. day\_time <= '1';
180. code <= x"5";
181. wait for clk\_period;
182. assert door = '0' and alarm = '1'
183. report "Entering a wrong code, even during daytime should trigger the alarm"
184. severity failure;
185. -- 5
186. wait for clk\_period;
187. assert door = '0' and alarm = '0'
188. report "Waiting for a clock cycle, should reset both door and alarm, to zero"
189. severity failure;
190. -- 6
191. reset <= '0';
192. day\_time <= '0';
193. code <= o;
194. wait for clk\_period;
195. assert door = '0' and alarm = '1'
196. report "Entering 'O' during night time should trigger the alarm"
197. severity failure;
198. -- 7
199. wait for clk\_period;
200. assert door = '0' and alarm = '0'
201. report "Waiting for a clock cycle, should reset both door and alarm, to zero"
202. severity failure;
203. -- 8
204. reset <= '0';
205. day\_time <= '1';
206. code <= o;
207. wait for clk\_period;
208. assert door = '1' and alarm = '0'
209. report "Entering 'O' during daytime should open the door"
210. severity failure;
211. -- 9
212. wait for clk\_period;
213. assert door = '0' and alarm = '0'
214. report "Waiting for a clock cycle, should reset both door and alarm, to zero"
215. severity failure;
216. -- 10
217. reset <= '0';
218. day\_time <= '0';
219. code <= x"2";
220. wait for clk\_period;
221. assert door = '0' and alarm = '0'
222. report "Entering 2 neither opens the door, nor triggers the alarm"
223. severity failure;
224. -- 11
225. reset <= '0';
226. day\_time <= '0';
227. code <= x"6";
228. wait for clk\_period;
229. assert door = '0' and alarm = '0'
230. report "Entering 2, 6 neither opens the door, nor triggers the alarm"
231. severity failure;
232. -- 12
233. reset <= '0';
234. day\_time <= '1';
235. code <= a;
236. wait for clk\_period;
237. assert door = '0' and alarm = '0'
238. report "Entering 2, 6, A neither opens the door, nor triggers the alarm, switching daytime to 1 and not pressing 'O' , will not open the door nor trigger the alarm"
239. severity failure;
240. -- 13
241. reset <= '0';
242. day\_time <= '0';
243. code <= x"0";
244. wait for clk\_period;
245. assert door = '0' and alarm = '0'
246. report "Entering 2, 6, A, 0 neither opens the door, nor triggers the alarm"
247. severity failure;
248. -- 14
249. reset <= '0';
250. day\_time <= '0';
251. code <= x"5";
252. wait for clk\_period;
253. assert door = '1' and alarm = '0'
254. report "Entering 2, 6, A, 0, 5 opens the door, but doesn't trigger the alarm"
255. severity failure;
256. -- 15
257. wait for clk\_period;
258. assert door = '0' and alarm = '0'
259. report "Waiting for a clock cycle, should reset both door and alarm, to zero"
260. severity failure;
261. assert false
262. report "FSM PASSED ALL TEST SUCCESSFULLY."
263. severity error;
264. wait;
265. end process;
266. process begin
267. clk <= '1', '0' after (clk\_period/2);
268. wait for clk\_period;
269. end process;
270. end architecture dac\_tb\_behav;