##### 

Program:

|  |  |
| --- | --- |
| Course Number | 1444 |
| Section Number | 1 |
| Course Title | EMBEDDED SYSTEMS ARCHITECTURE 1 |
| Semester/Year | WINTER/2019 |

|  |  |
| --- | --- |
| Instructor | **Mohsen Salahi** |

|  |  |
| --- | --- |
| **Lab No.** | **4** |

|  |  |
| --- | --- |
| Submission Date | **31/04/2019** |
| Due Date | **31/04/2019** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Student Name | **FRANCIS VIJITH** | **NIBIN SOBY** | **AKHIL GEORGE** | **VIJAYAPRAKASH** |
| Student ID | C0742980 | C0741660 | C0748466 | C0744450 |
| Signature\* | **FV** | **NS** | **AG** | **VI** |

*\*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.*

**VHDL CODE:**

library IEEE;

use IEEE.STD¬\_logic-1164.ALL;

entity THUNDER\_L4 is

PORT (clock, Rst : IN STD\_logic;

right,left, brake : IN STD\_logic;

lefta, leftb, leftc :OUT STD\_LOGIC;

righta, rightb,rightc : OUT STD\_LOGIC;

sp : OUT STD\_LOGIC)

end THUNDER-L4;

>

Architecture Behavioral of THUNDER\_L4 is

Signal Rsig, Lsig : std-logic := ‘0’;

TYPE state\_type is (s0,s1,s2,s3);

SIGNAL state : state-type;

BEGIN

Process (clock, Rst,right, left)

Begin

If (Rst= '1’) Then

state <= so;

lefta <= '0’;

leftb <= '0';

leftc <= ‘0’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

elsif rising\_edge(clock) then

case state is

when so=>

>if(brake =’0’)and(right=’0’)and(left=’1’) then

state <= s1;

lefta <= ’0’;

leftb <= ’0’;

leftc <= ’0’;

righta <= ’0’;

rightb <= ’0’;

rightc <= ’0’;

rsig <=’0’;

lsig <=’1’;

sp <=’0’;

--state 1

elsif (brake =’0’) and (Right=’1’) and (left=’0’) then

state <=s1;

lefta <= ‘0’ ;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ’0’;

rightb <= ’0’;

rightc <= ’0’;

rsing <= ‘1’;

lsing <= ‘0’;

sp <=’0’;

elsif (brake =’1’) and (right=’1’) and (left=’0’) then

state <=s1;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘1’;

rsing <=’1’;

lsing <=’0’;

sp <=’1’;

elsif (brake =’1’) and (right=’0’) and (left=’1’) then

state <=s1;

lefta <=’1’;

leftb <=’1’;

leftc <=’1’;

righta <=’0’;

rightb <=’0’;

rightc <=’0’;

rsing ,=’0’;

lsing <=’1’;

sp <=’1’;

else

state <=s0;

lefta <= ‘0’;

leftb <=’0’;

leftc <=’0’;

righta <=’0’;

rightb <= ‘0’;

rightc <=’0’;

end if;

when s1=>

if (Lsing =’1’) and (Rsing =’0’ 0 and (brake = ‘0’) then

state <= s2;

lefta <= ‘1’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

elsif (lsig = ‘0’) and (Rsig =’1’) and (brake =’0’) then

state <=s2;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘1’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

elsif (lsig =’1’) and (rsig = ‘0’) and (brake =’1’) then

state <= s2;

lefta <= ‘1’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘1’;

sp <=’1’;

elsif (Lsing = ‘0’) and (Rsing = ‘1’) and (brake=’1’) then

state <=s2;

lefta <= ‘1’;

leftb <= ‘1’;

leftc <= ‘1’;

righta <= ‘1’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’1’;

else

state <=s0;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

end if;

when s2=>

if (lsig =’1’) and (rsig =’0’) and (brake =’0’) then

state <=s3;

lefta <= ‘1’;

leftb <= ‘1’;

leftc <= ‘0’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

elsif (lsig =’0’) and (rsig =’1’) and (brake =’0’)

state <= s3;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘0’;

sp <=’0’;

elsif (lsig = ‘1’) and (rsig = ‘0’) and (brake =’1’) then

state <=s3;

lefta <= ‘1’;

leftb <= 1’;

leftc <= ‘0’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘1’;

sp <=’1’;

elsif (lsig =’0’) and (rsig =’1’) and (brake =’1’) then

state <= s3;

lefta <= ‘1’;

leftb <= ‘1’;

leftc <= ‘1’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘0’;

sp <=’1’;

else

state <= s0;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’1’;

end if;

--state 3

When s3=>

If (lsig =’1’) and (rsig = ‘0’) and (brake =’0’) then

state <=s0;

lefta <= ‘1’;

leftb <= ‘1’;

leftc <= ‘1’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

elsif (lsig =’0’) and (rsig =’1’) and (brake =’0’) then

state <=s0;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘1’;

sp <=’0’;

elsif (lsig =’1’) and (rsig= “0’) and (brake =’1’) then

state <= s0;

lefta <= ‘1’;

leftb <= ‘1’;

leftc <= ‘1’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘1’;

sp <=’1’;

elsif (lsig = ‘0’) and (rsig = ‘1’) and (brake =’1’) then

state <=’1’;

lefta <= ‘1’;

leftb <= ‘1’;

leftc <= ‘1’;

righta <= ‘1’;

rightb <= ‘1’;

rightc <= ‘1’;

sp <=’1’;

else

state <=s0;

lefta <= ‘0’;

leftb <= ‘0’;

leftc <= ‘0’;

righta <= ‘0’;

rightb <= ‘0’;

rightc <= ‘0’;

sp <=’0’;

end if;

when others =>

end case;

End if;

end process;

end Behavioral;

**Output waveform:**

