(CSEN605) Digital System Design

Project: Smart Car

Report

1) The project idea

PROJECT: “بنزين برق Smart Car:” We created a model for a smart self-driving car that follows a straight lane black colored, We used line track sensor to detect that the car goes out of the lane, in this case 7 segment is used to display a warning message “ALERT”, We also used IR sensor to detect if there is an obstacle in front of the car, in this case the motor of the car should stop.

2) The Code

```
library ieee;
use ieee.std_logic_1164.all;

entity MyProject is port(
IR : in std_logic;
track : in std_logic;
motor : out std_logic;
seven1, seven2, seven3, seven4, seven5 : out std_logic_vector(0 to 6)
);
end MyProject;

architecture arch of MyProject is
begin
process (IR, track)
        begin
        if IR='0'
                then motor<='0';
        else
                motor <='1';
        end if;
        if track='0'
                then
                seven1 <= "0001000";
                seven2 <= "1110001";
                seven3 <= "0110000";
                seven4 <= "1111010";
                seven5 <= "1110000";
        else
                seven1 <= "1111111";
                seven2 <= "1111111";
                seven3 <= "1111111";
                seven4 <= "1111111";
                seven5 <= "1111111";
        end if;
        end process;
end arch;
```

i. Inputs

1- “IR” represents the IR sensor which is responsible for detecting obstacles.

2- “track” represents the line tracker sensor which is responsible for detecting that the car goes out the lane black colored.

ii. Outputs

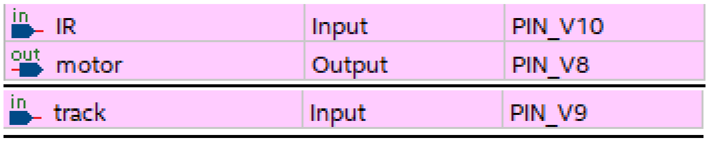
1- “motor” represents the base connecting the motor of the car to the ground to control it which is responsible for moving the car.

2- “seven1, seven2, seven3, seven4, seven5” represents the word “ALErt” that will be displayed on the 7 segment if the car goes out of the lane, where each one of them represents a letter of the word.
“seven1”→”A”, “seven2”→”L”, “seven3”→”E”, “seven4”→”r”, “seven5”→”t”.

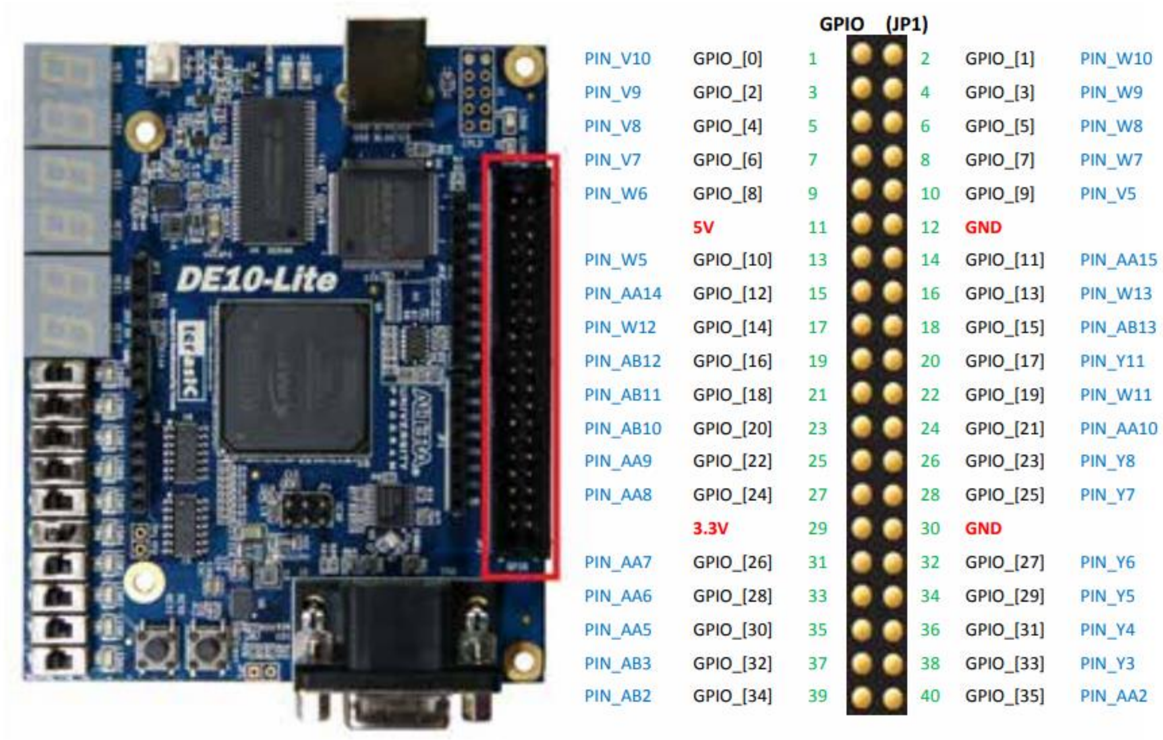
iii. Pin Assignment

*We used the data sheet to know how to assign “IR”, ”track” as input signals and to assign “motor” as output signal in “GPIO” and to assign “seven1, seven2, seven3, seven4, seven5” to 7 segments of the FPGA as follows,

IR → PIN_V10, motor → PIN_V8, track → PIN_V9.



in IR	Input	PIN_V10
out motor	Output	PIN_V8
in track	Input	PIN_V9

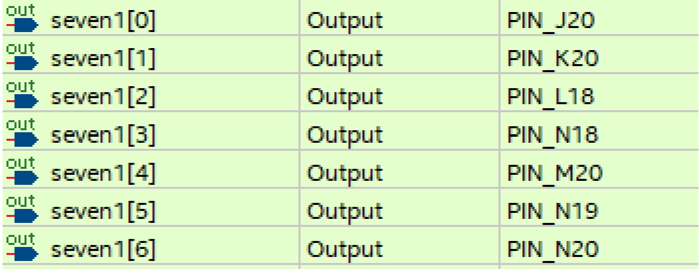


GPIO (JP1)

PIN_V10	GPIO_[0]	1	2	GPIO_[1]	PIN_W10
PIN_V9	GPIO_[2]	3	4	GPIO_[3]	PIN_W9
PIN_V8	GPIO_[4]	5	6	GPIO_[5]	PIN_W8
PIN_V7	GPIO_[6]	7	8	GPIO_[7]	PIN_W7
PIN_W6	GPIO_[8]	9	10	GPIO_[9]	PIN_V5
	5V	11	12	GND	
PIN_W5	GPIO_[10]	13	14	GPIO_[11]	PIN_AA15
PIN_AA14	GPIO_[12]	15	16	GPIO_[13]	PIN_W13
PIN_W12	GPIO_[14]	17	18	GPIO_[15]	PIN_AB13
PIN_AB12	GPIO_[16]	19	20	GPIO_[17]	PIN_Y11
PIN_AB11	GPIO_[18]	21	22	GPIO_[19]	PIN_W11
PIN_AB10	GPIO_[20]	23	24	GPIO_[21]	PIN_AA10
PIN_AA9	GPIO_[22]	25	26	GPIO_[23]	PIN_Y8
PIN_AA8	GPIO_[24]	27	28	GPIO_[25]	PIN_Y7
	3.3V	29	30	GND	
PIN_AA7	GPIO_[26]	31	32	GPIO_[27]	PIN_Y6
PIN_AA6	GPIO_[28]	33	34	GPIO_[29]	PIN_Y5
PIN_AA5	GPIO_[30]	35	36	GPIO_[31]	PIN_Y4
PIN_AB3	GPIO_[32]	37	38	GPIO_[33]	PIN_Y3
PIN_AB2	GPIO_[34]	39	40	GPIO_[35]	PIN_AA2

“seven1, seven2, seven3, seven4, seven5” required 35 pin assignments because every one of them requires 7 pin assignment to represent a single letter on the 7 segment so we have 5 letters times 7 segment resulting in 35 pin assignments as follows,

“seven1” which represents letter “A”.
the first 7 segment from the left.
seven1[0]→PIN_J20, seven1[1]→PIN_K20, seven1[2]→PIN_L18,
seven1[3]→PIN_N18, seven1[4]→PIN_M20, seven1[5]→PIN_N19, seven1[6]→PIN_N20.

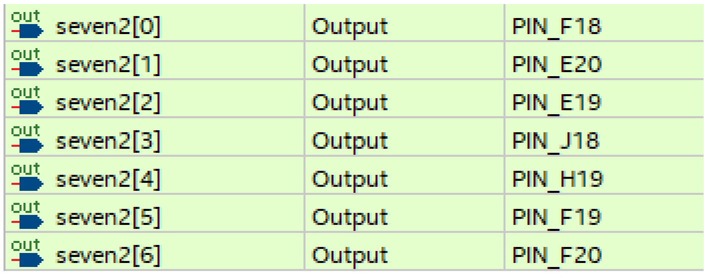


out seven1[0]	Output	PIN_J20
out seven1[1]	Output	PIN_K20
out seven1[2]	Output	PIN_L18
out seven1[3]	Output	PIN_N18
out seven1[4]	Output	PIN_M20
out seven1[5]	Output	PIN_N19
out seven1[6]	Output	PIN_N20

"seven2" which represents letter "L".

the second 7 segment from the left.

seven2[0]→PIN_F18, seven2[1]→PIN_E20, seven2[2]→PIN_E19,
seven2[3]→PIN_J18, seven2[4]→PIN_H19, seven2[5]→PIN_F19, seven2[6]→PIN_F20.

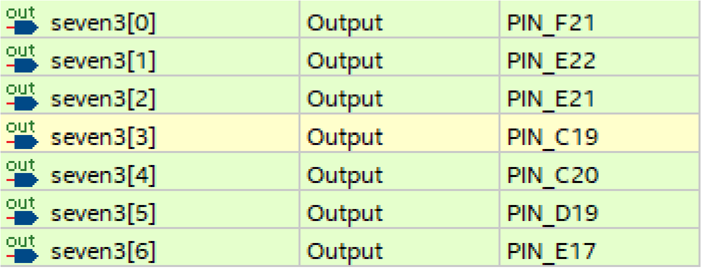


out	seven2[0]	Output	PIN_F18
out	seven2[1]	Output	PIN_E20
out	seven2[2]	Output	PIN_E19
out	seven2[3]	Output	PIN_J18
out	seven2[4]	Output	PIN_H19
out	seven2[5]	Output	PIN_F19
out	seven2[6]	Output	PIN_F20

"seven3" which represents letter "E".

the third 7 segment from the left.

seven3[0]→PIN_F21, seven3[1]→PIN_E22, seven3[2]→PIN_E21,
seven3[3]→PIN_C19, seven3[4]→PIN_C20, seven3[5]→PIN_D19, seven3[6]→PIN_E17.

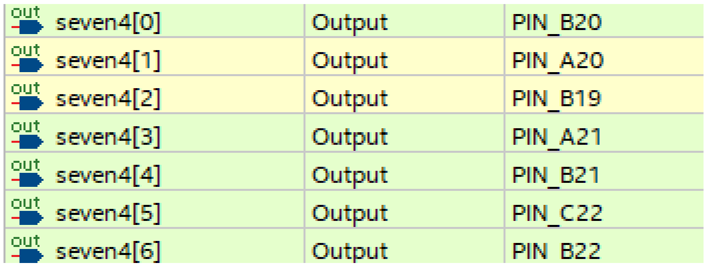


out	seven3[0]	Output	PIN_F21
out	seven3[1]	Output	PIN_E22
out	seven3[2]	Output	PIN_E21
out	seven3[3]	Output	PIN_C19
out	seven3[4]	Output	PIN_C20
out	seven3[5]	Output	PIN_D19
out	seven3[6]	Output	PIN_E17

"seven4" which represents letter "r".

the fourth 7 segment from the left.

seven4[0]→PIN_B20, seven4[1]→PIN_A20, seven4[2]→PIN_B19,
seven4[3]→PIN_A21, seven4[4]→PIN_B21, seven4[5]→PIN_C22, seven4[6]→PIN_B22.

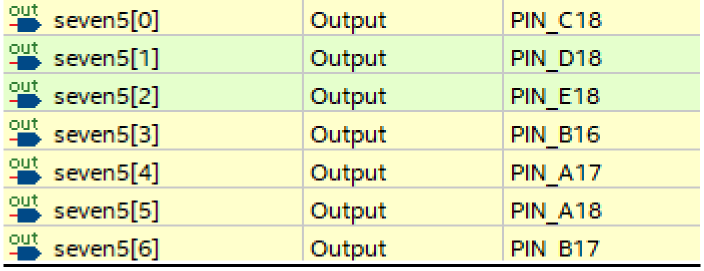


out	seven4[0]	Output	PIN_B20
out	seven4[1]	Output	PIN_A20
out	seven4[2]	Output	PIN_B19
out	seven4[3]	Output	PIN_A21
out	seven4[4]	Output	PIN_B21
out	seven4[5]	Output	PIN_C22
out	seven4[6]	Output	PIN_B22

"seven5" which represents letter "t".

the fifth 7 segment from the left.

seven5[0]→PIN_C18, seven5[1]→PIN_D18, seven5[2]→PIN_E18, seven5[3]→PIN_B16, seven5[4]→PIN_A17, seven5[5]→PIN_A18, seven5[6]→PIN_B17.



out seven5[0]	Output	PIN_C18
out seven5[1]	Output	PIN_D18
out seven5[2]	Output	PIN_E18
out seven5[3]	Output	PIN_B16
out seven5[4]	Output	PIN_A17
out seven5[5]	Output	PIN_A18
out seven5[6]	Output	PIN_B17

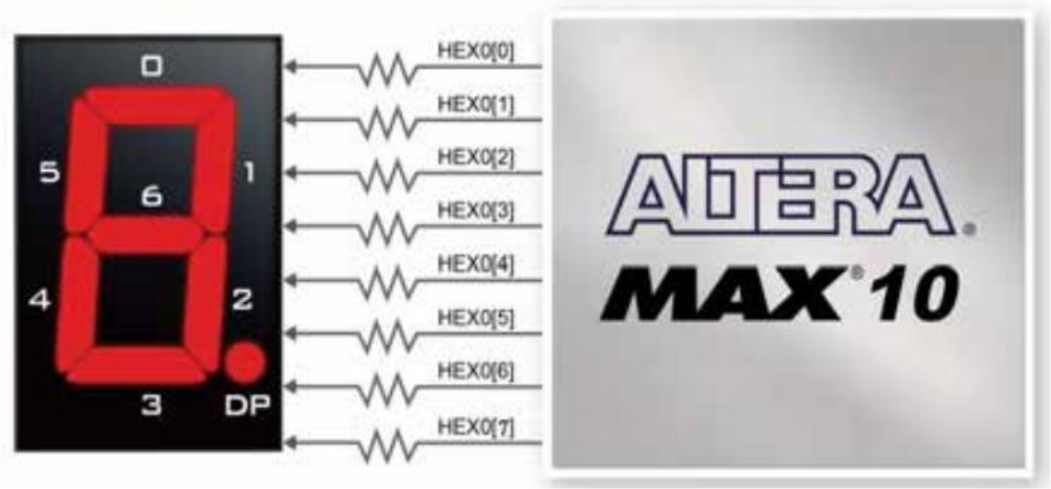
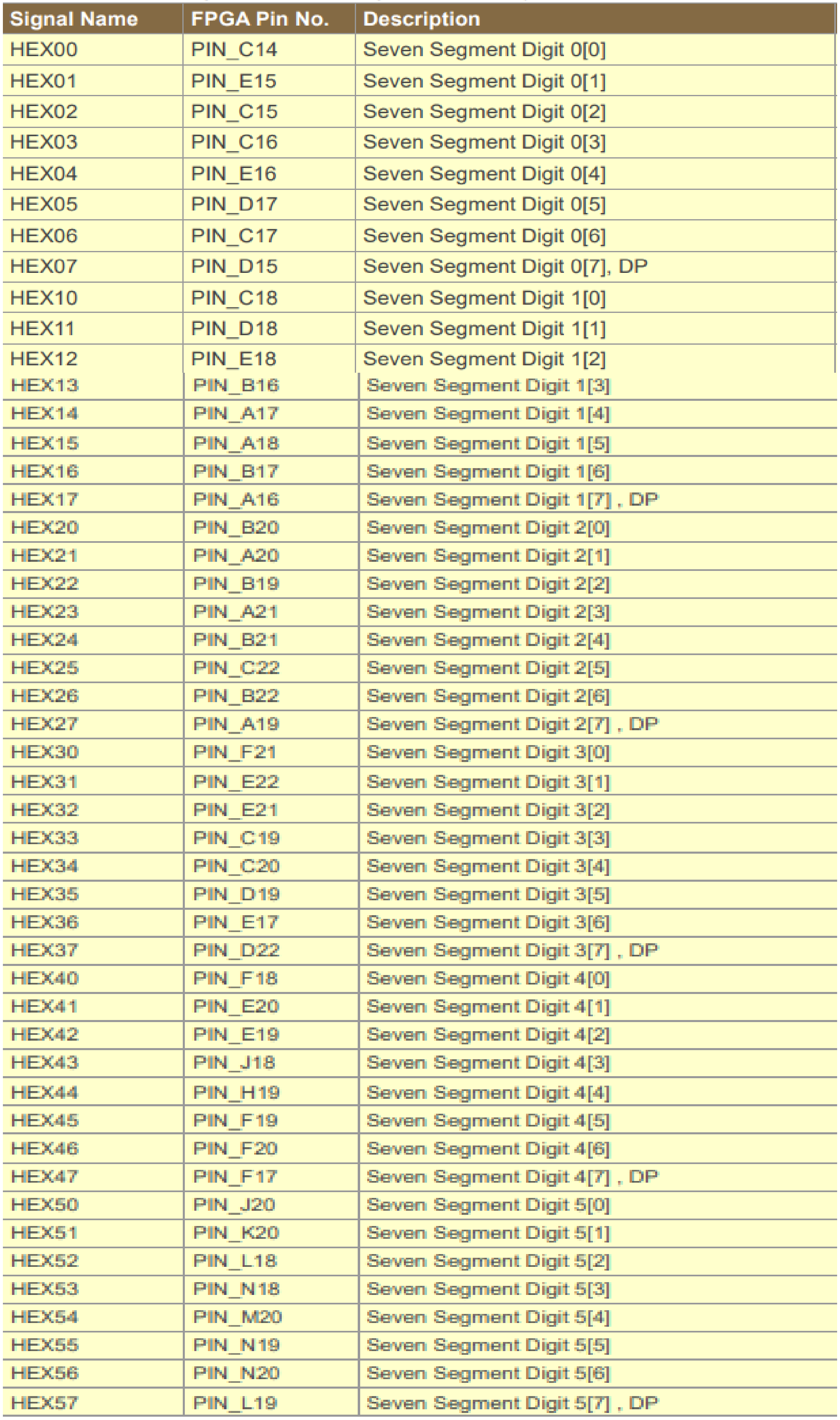


Table 3-6 Pin Assignment of 7-segment Displays



Signal Name	FPGA Pin No.	Description
HEX00	PIN_C14	Seven Segment Digit 0[0]
HEX01	PIN_E15	Seven Segment Digit 0[1]
HEX02	PIN_C15	Seven Segment Digit 0[2]
HEX03	PIN_C16	Seven Segment Digit 0[3]
HEX04	PIN_E16	Seven Segment Digit 0[4]
HEX05	PIN_D17	Seven Segment Digit 0[5]
HEX06	PIN_C17	Seven Segment Digit 0[6]
HEX07	PIN_D15	Seven Segment Digit 0[7], DP
HEX10	PIN_C18	Seven Segment Digit 1[0]
HEX11	PIN_D18	Seven Segment Digit 1[1]
HEX12	PIN_E18	Seven Segment Digit 1[2]
HEX13	PIN_B16	Seven Segment Digit 1[3]
HEX14	PIN_A17	Seven Segment Digit 1[4]
HEX15	PIN_A18	Seven Segment Digit 1[5]
HEX16	PIN_B17	Seven Segment Digit 1[6]
HEX17	PIN_A16	Seven Segment Digit 1[7] , DP
HEX20	PIN_B20	Seven Segment Digit 2[0]
HEX21	PIN_A20	Seven Segment Digit 2[1]
HEX22	PIN_B19	Seven Segment Digit 2[2]
HEX23	PIN_A21	Seven Segment Digit 2[3]
HEX24	PIN_B21	Seven Segment Digit 2[4]
HEX25	PIN_C22	Seven Segment Digit 2[5]
HEX26	PIN_B22	Seven Segment Digit 2[6]
HEX27	PIN_A19	Seven Segment Digit 2[7] , DP
HEX30	PIN_F21	Seven Segment Digit 3[0]
HEX31	PIN_E22	Seven Segment Digit 3[1]
HEX32	PIN_E21	Seven Segment Digit 3[2]
HEX33	PIN_C19	Seven Segment Digit 3[3]
HEX34	PIN_C20	Seven Segment Digit 3[4]
HEX35	PIN_D19	Seven Segment Digit 3[5]
HEX36	PIN_E17	Seven Segment Digit 3[6]
HEX37	PIN_D22	Seven Segment Digit 3[7] , DP
HEX40	PIN_F18	Seven Segment Digit 4[0]
HEX41	PIN_E20	Seven Segment Digit 4[1]
HEX42	PIN_E19	Seven Segment Digit 4[2]
HEX43	PIN_J18	Seven Segment Digit 4[3]
HEX44	PIN_H19	Seven Segment Digit 4[4]
HEX45	PIN_F19	Seven Segment Digit 4[5]
HEX46	PIN_F20	Seven Segment Digit 4[6]
HEX47	PIN_F17	Seven Segment Digit 4[7] , DP
HEX50	PIN_J20	Seven Segment Digit 5[0]
HEX51	PIN_K20	Seven Segment Digit 5[1]
HEX52	PIN_L18	Seven Segment Digit 5[2]
HEX53	PIN_N18	Seven Segment Digit 5[3]
HEX54	PIN_M20	Seven Segment Digit 5[4]
HEX55	PIN_N19	Seven Segment Digit 5[5]
HEX56	PIN_N20	Seven Segment Digit 5[6]
HEX57	PIN_L19	Seven Segment Digit 5[7] , DP

iv. Explaining the code

Lines of code

```
library ieee;
use ieee.std_logic_1164.all;
```

Meaning

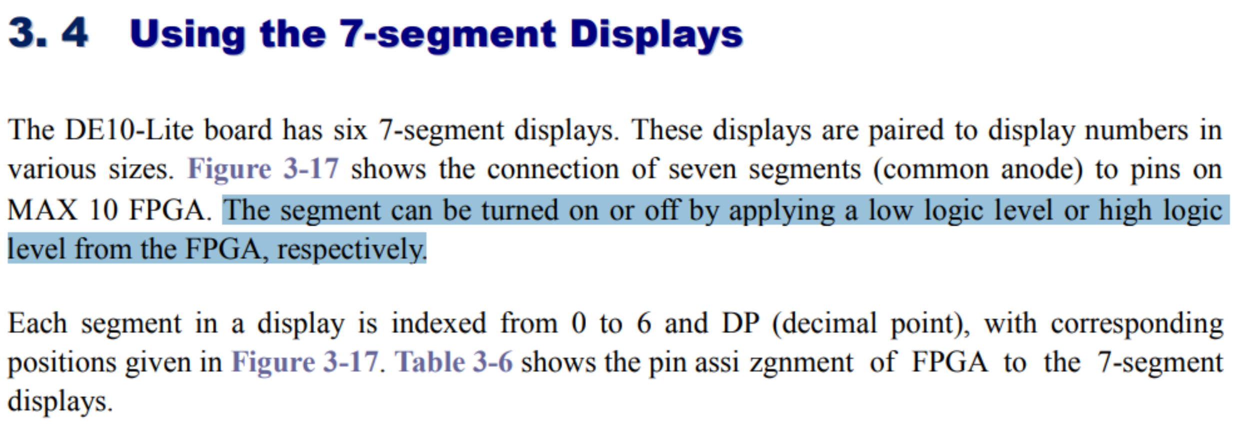
We add this package at the beginning of our code to be able to use STD_LOGIC TYPE that can represent 0,1, X unknown, Z high impedance, U uninitialized and so on.

Lines of code

```
entity MyProject is port(
IR : in std_logic;
track : in std_logic;
motor : out std_logic;
seven1, seven2, seven3, seven4, seven5 : out std_logic_vector(0 to 6)
);
end MyProject;
```

Meaning

We created the entity of our project and called “MyProject” which contains two inputs, “IR” which is assigned to FPGA to pin where the signal of our IR sensor is connected to act as the signal of IR sensor which gives ‘0’ if it detects an obstacle ‘1’ otherwise, “track” which is assigned to FPGA to pin where the signal of our line tracker sensor is connected to act as the signal of line tracker sensor which gives ‘0’ if it detects that the car goes out of the lane ‘1’ otherwise, and one output “motor” which is assigned to FPGA to pin where the base of our transistor is connected to act as the transistor which control how our motor works as if it gives ‘1’ means that the motor is working & ‘0’ means that the motor stops and another five outputs “seven1, seven2, seven3, seven4, seven5” each one of them represent a letter of the word “ALErt”on 7-segment thus each one of them must be vector of 7 indices std_logic_vector(0 to 6) to control each segment of the 7 segments as of ‘0’ means that the segment of the 7-segment is turned on and ‘1’ means that the segment of the 7-segment is turned off, then we end our entity.



3. 4 Using the 7-segment Displays

The DE10-Lite board has six 7-segment displays. These displays are paired to display numbers in various sizes. Figure 3-17 shows the connection of seven segments (common anode) to pins on MAX 10 FPGA. The segment can be turned on or off by applying a low logic level or high logic level from the FPGA, respectively.

Each segment in a display is indexed from 0 to 6 and DP (decimal point), with corresponding positions given in Figure 3-17. Table 3-6 shows the pin assi zgnment of FPGA to the 7-segment displays.

Lines of code

```
architecture arch of MyProject is
begin
```

Meaning

We add architecture called arch to control the behavior of our entity.

Lines of code

```
process (IR, track)
begin
```

Meaning

We put the inputs “IR” & “track” in the process to know immediately once “IR” detects anything if there is an obstacle or the obstacle gone and also to know immediately once “track” detects anything if the car goes out the lane or returns back to lane.

Lines of code

```
if IR='0'
    then motor<='0';
else
    motor <='1';
end if;
if track='0'
    then
    seven1 <= "0001000";
    seven2 <= "1110001";
    seven3 <= "0110000";
    seven4 <= "1111010";
    seven5 <= "1110000";
else
    seven1 <= "1111111";
    seven2 <= "1111111";
    seven3 <= "1111111";
    seven4 <= "1111111";
    seven5 <= "1111111";
end if;
end process;
end arch;
```

Meaning

We check if the “IR”=’0’ means that the IR sensor detects an obstacle so we set the “motor” to ‘0’ to make the motor stops working (Car stops), else means that “IR”=’1’ means that there is no obstacle so we set the “motor” to ‘1’ to make the motor working (Car moves), We also check if the “track”=’0’ means that the car goes out of the lane so we set the 5 outputs “seven1, seven2, seven3, seven4, seven5” to values "0001000", ”1110001”, “0110000”, “1111010”, “1110000” respectively to put the warning message “ALErt” on the 7-segment (comment → r is lower case because it looks like A if it’s upper case and t is also lower case because we can’t print it as upper case on 7-segment), else means that “track”=’1’ means that the car returns back to the lane so we set the 5 outputs “seven1, seven2, seven3, seven4, seven5” to values "1111111", ” 1111111”, “1111111”, “1111111”, “1111111” respectively to turns off the 7-segment (remove the warning message “ALErt” from the 7-segment), then we end process and finally end the architecture arch.

3) Components Used

- i. **FPGA DE10_Lite**
- ii. **Bread Board**
- iii. **DC motor 3 to 6 V**
- iv. **IR Sensor**
- v. **Line tracker Sensor**
- vi. **Transistor**
- vii. **Jumpers male to male**
- viii. **Jumpers male to Female**

4) General Idea

The laptop gives power to FPGA once we connect the FPGA to the laptop then We connect the 5V and ground from the FPGA to the positive and negative nodes on the bread board respectively. We connect the Vcc of IR sensor and line tracker sensor to the positive node of the bread board (5V) and the ground of IR sensor and line tracker sensor to the negative node (GND) of the bread board and the signal of the IR sensor and line tracker sensors to two signals in the FPGA in GPIO acting as inputs. We connect one terminal of DC motor to the positive node in the bread board (5V) and the other terminal to the Emitter of the transistor which is connected to the negative node in the bread board (GND) so the transistor acts like a switch between the motor and the ground to control the motor through the base of the transistor which is connected to signal in the FPGA GPIO acting as an output. We write our code then compile then do the pin assignments for all inputs (IR and line track) and all outputs (base of the transistor which controls the motor and the 7-segments) then recompile again then test the functionality of all components connected to FPGA which is connected to the laptop.