EL203

Embedded Hardware design

Project : Elevator Controller

**Group : 21**

**Group Members :**

* Bopparam Gangaraju (202001107)
* Nikhil Jethanandani (202001108)
* Darshan Kheni (202001109)
* Vihar Shah (202001110)
* Jaineet Deora (202001111)

**Problem Statement**

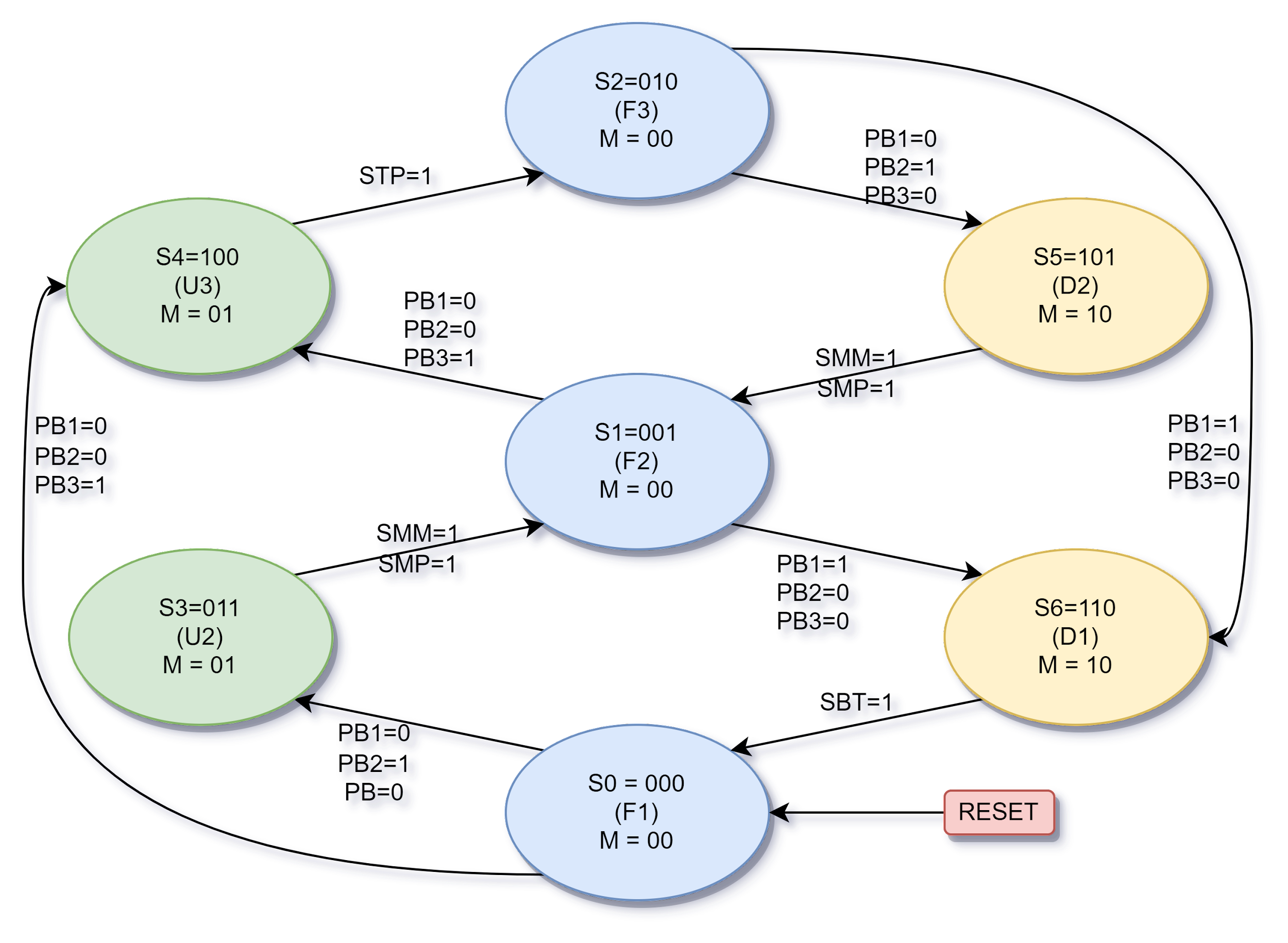
Design a simple elevator controller structure representing a lift shaft to which are fixed three call buttons and associated lift coming indicators. The motor which moves the lift up and down should be designed as a Rack-and-pinion gear system.The Elevator system is designed to operate for a three floor system and associated three call buttons at each of these floors. The alignment of the lift is directed by four sensors named as Bottom, Middle minus, Middle plus and a Top. The system should be able to handle invalid inputs. When the lift is in Floor 1 , then it shouldn’t move down further and when the lift is in Floor 3 , it shouldn’t move up.A power supply of 12V is given to the motor to operate.

**Contribution**:

| Name | Contribution | Percentage |
| --- | --- | --- |
| Bopparam Gangaraju | Code(vhdl code, TestBench),Hardware | 26% |
| Nikhil Jetanandni | Code(vhdl code, ucf),Report | 19% |
| Darshan Kheni | Code(vhdl code, ucf),Report | 19% |
| Vihar Shah | Code(vhdl code, TestBench),Hardware | 26% |
| Jaineet Deora | Print, Report | 10% |

**State Diagram**

The Elevator control system can be represented using the following Moore state machine.



**Description of State Diagram:**

Here, the state diagram consists of 7 states, each represented by F1, F2, F3, U2, U3, D2, and D1 respectively. The output of each state determines the state of the lift and that of the motor indicated by M.

* State F1 = Lift is stationary on floor 1. It is represented by logic vector 000(S0).
* State F2 = Lift is stationary on floor 2. It is represented by logic vector 001(S1).
* State F3 = Lift is stationary on floor 3. It is represented by logic vector 010(S2).
* State U2 = Lift is moving up to floor 2. It is represented by vector 011 (S3).
* State U3 = Lift is moving up to floor 3. It is represented by vector 100 (S4).
* State D2 = Lift is moving down to floor 2. It is represented by vector 101(S5).
* State D1 = Lift is moving down to floor 1. It is represented by vector 110(S6).
* State M = 00 represents motor turned off.
* State M = 01 indicates the upwards motion of the elevator.
* State M = 10 indicates the downward movement of the elevator.

The push button inputs are PB1, PB2, and PB3, each representing the call button for floor 1, floor 2, and floor 3 respectively.

**Movement of lift**

* The lift goes into one of the 4 moving states depending on the floor’s call button. If the call button is same as lift’s stationary floor, nothing happens.
* The sensor inputs are SBT, SMM, SMP, and STP, each corresponding to the Bottom sensor, Middle Minus Sensor, Middle Plus sensor, and Top sensor.
* Once in moving state, the lift remains in that state until the sensor(s) corresponding to the destination floor signals the lifts arrival.

SBT = sensor for bottom floor. The bottom floor is reached when the sensor is activated.

STP = sensor for top floor. The top floor is reached when the sensor is activated.

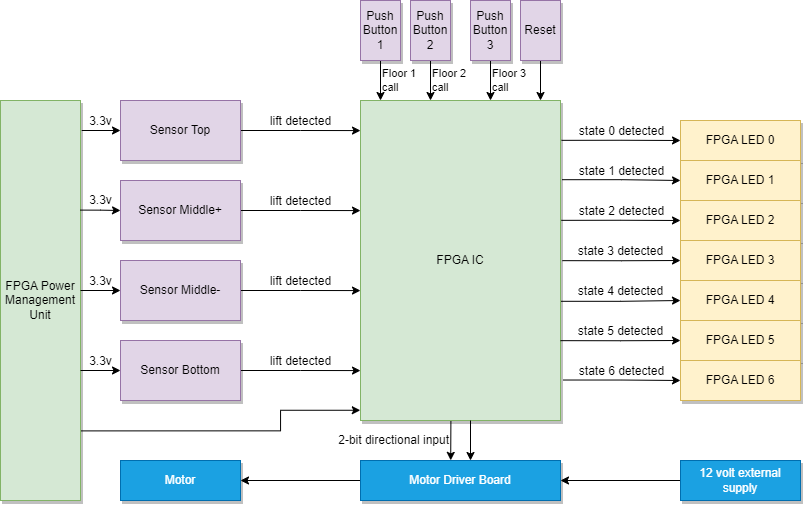
SMM = Middle minus sensor. The sensor activates to signal alignment of lift’s floor for the middle floor.

SMP = Middle plus sensor. The sensor activates to signal alignment of lift’s ceiling for the middle floor.

* The lift becomes stationary on the middle floor only when SMM and SMP both are active.
* A final input of RESET is configured as well to reset the software to state 0. The reset button is configured for software and hardware. The lift will go to floor 1 upon reset.

**State Equations:**

**Block Diagram**

****

**VHDL Snippets**

**LiftControl.vhd**

**Elevator\_control**:process(RESET, SBT,SMM,SMP,STP,PB1,PB2,PB3)

* A process “Elevator\_control” is created with its sensitivity list containing all the three call button inputs, the four sensor inputs, and reset button.
* The entire logic to be executed is configured inside this “check” process.

Handling push buttons:

 if (RESET = '0') then state<=0; end if;

    if(PB1='1' and PB2='0' and PB3='0') then

      if(state=1 or state=2) then state<=6; end if;

    elsif(PB1='0' and PB2='1' and PB3='0') then

      if(state=0) then state<=3;

      elsif (state=2) then state<=5; end if;

    elsif(PB1='0' and PB2='0' and PB3='1') then

      if(state=0 or state=1) then state<=4; end if;

    end if;

Handling Inputs from sensors

    if(SBT='1' and SMM='0' and SMP='0' and STP='0') then

      if(state=6 ) then

        state<=0;

      end if;

    elsif (SBT='0' and SMM='1' and SMP='1' and STP='0') then

      if(state=5 or state=3) then

        state<=1;

      end if;

    elsif(SBT='0' and SMM='0' and SMP='0' and STP='1') then

      if(state=4) then

        state<=2;

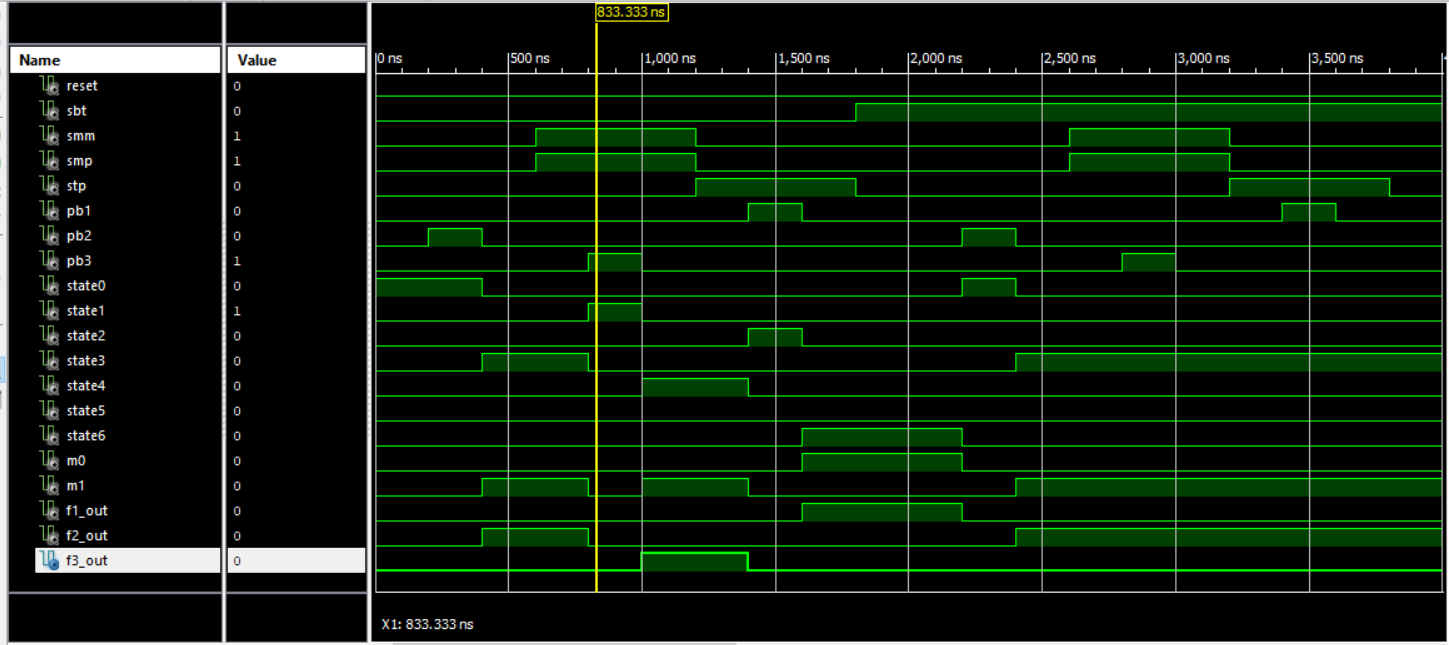
      end if;

    end if;

**Lift.ucf**

In this file, we assigned the IO pins to be used by each of the inputs. The input pins for sensor input, push buttons, and reset button were defined, where as output pins for the 7 LEDs, and 2 pins for motor output are defined. All the pins to connect the jumper wires are defined on the 4 digital J-ports. The lift r-eset button is taken as the reset button provided on the FPGA. All the input-output ports are defined to have IOSTANDARD value equal to LVCMOS33.

**Testbench Output**

****

TestBench simulation results for the state machine is as shown in above figure.

**f**