Special Assignment

Digital Logic Design (2EC303)

Design of Elevator Controller Using Verilog HDL

Project Report

Submitted By:

21BEC030 Dhruv Shah

21BEC035 Dhruv Gobbi

Guided By:

Prof. Manish Patel

INDEX

ABSTRACT

The aim of the project is to design and implement an Elevator/Lift Controller using Verilog hardware descriptive language (HDL). The Elevator Controller is a device used to control a lift motion and to indicate the direction of motion, and the present floor level, etc. The device controls the lift motion by means of accepting the floor level as input and generate control signals (for control the lift motion) as output The elevator control system is based on the finite state machine technology. The elevator process may be specified using several states according to FSM technology. There is a transition from one state to another with FSM technology, just as there will be a change from one floor to another in the elevator. Every conceivable path is assigned a path, and the computer code for the elevator controller is written using the FSM idea. The entire software is set up such that there are desirable switches on each floor, as well as within the elevator, to control user commands. We simply need the up switch and nothing else while the elevator is on the ground level to move upward. For the upper floor, we use the same approach. To travel lower, there is just one down switch. All other levels, however, feature two switches, one for travelling up and one for moving down, between the bottom and top floors. Inside the elevator there must be at least 'n' switches for the implementation of an 'n' floor elevator controller. The elevator will move in the direction indicated by the user's desired input. The design features a straightforward approach that strives for a fast reaction time without requiring any additional logic hardware.

INTRODUCTION

The high growth of the semiconductor industry over past two decades has put Very Large-Scale Integration in demand all over the world. The basics of digital logic theory and techniques are easily understood by the design based on VLSI technology. The principles of rapid, high-speed complicated digital circuits are as follows. Technology is rapidly advancing day by day. As a result, the designs must be simplified in order to reap the benefits. An Elevator Controller is modelled to do this. A VERILOG RTL code is constructed in the suggested architecture to regulate the lift moment based on the request it will get. For that a finite sate machine is developed to know from which state to state the controller is changing based on the requests from the end user. Elevator or car are other names for a lift. The architecture is based on a synchronous input that should operate at a constant frequency. Finally, the RTL is verified and implemented. The real-time three-lift controller will be simulated in this study utilizing Verilog HDL code and a Finite-State Machine (FSM) model to achieve the logic in the most efficient manner.

Defining Elevator

An Elevator is a type of vertical transport equipment that efficiently moves people or goods between floors (levels, decks) of a building, vessel or other structure. Elevators are generally powered by electric motors that either drive traction cables or counterweight systems like a hoist, or pump hydraulic fluid to raise a cylindrical piston like a jack.

In agriculture and manufacturing, an elevator is any type of conveyor device used to lift materials in a continuous stream into bis or silos. There are several varieties, including the chain and bucket elevator, grain auger screw conveyer based on Archimedes' screw, and hay elevators with chain and paddles/forks.

PRINCIPLE

Elevator controller is an elementary system consisting of elevator serving 20 floors. The elevator car has a pair of control buttons (up/down) for moving the elevator up and down. There are additional call buttons on the floors for elevator service. The following principles have been applied during the design of elevator controller: The floors are defined as first floor and second etc.

1) A floor call is serviced using the elevator.
2) Upon arrival at a floor, the doors open immediately.
3) Doors remain open before closure.
4) Elevator Up/Down buttons are connected to elevator units.

The elevator control system is based on the finite state machine technology idea. The elevator process may be specified using several states according to FSM technology. In the FSM technology there is change from one state to another state likewise in the elevator there will be a change from one floor to another. Every conceivable path is assigned a path, and the computer code for the elevator controller is written using the FSM idea. The complete programme is built up such that desirable switches are available on each floor, as well as within the elevator, to regulate user commands. If we push any of the floor switches while the elevator is still on the ground level, the elevator will proceed higher until it reaches the specified floor and the door will open. The doors remain open for a short length of time before closing and the elevator moving to the next floor. The elevator will move in the direction indicated by the user's desired input. The design features a straightforward approach that strives for a fast reaction time without requiring any additional logic hardware.

FINITE STATE MACHINE

A finite-state machine (FSM) is a mathematical abstraction sometimes used to design digital logic or computer programs. It is a model of behavior made up of a finite number of states, transitions between them, and actions. The finite-state machine is similar to a flow graph in which one can inspect the way logic runs when certain conditions are met. It has finite internal memory, an input feature (reading symbols in a sequence, one at a time without going backward), and an output feature, which may be in the form of a user interface, once the model is implemented. The number and names of the states are normally determined by the memory's different potential states; for example, a three-bit memory has eight possible states. The state that reads the first symbol of the input sequence is called the start state and the state which signifies the successful operation of the machine is called the accept state. Every state may lead to a different one depending on the next input symbol and output can be provided during the transition.

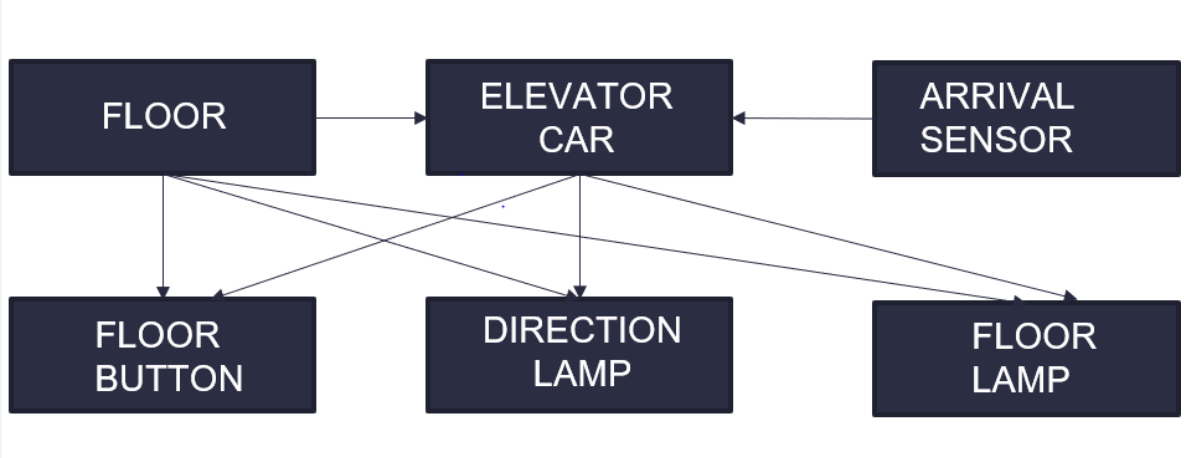
THE ELEVATOR ALGORITM

The elevator algorithm, a simple algorithm by which a single elevator an decide where to stop, is summarized as follows:

- Keep going in the same path as long as there are demands in that direction.
- If there are no further requests in that direction, then stop and become idle, or change direction if there are requests in the opposite direction.

The elevator algorithm has found an application in computer operating systems as an algorithm for scheduling hard disk requests. To determine which request to serve next, modern elevators employ more complicated algorithms.

BLOCK DIAGRAM

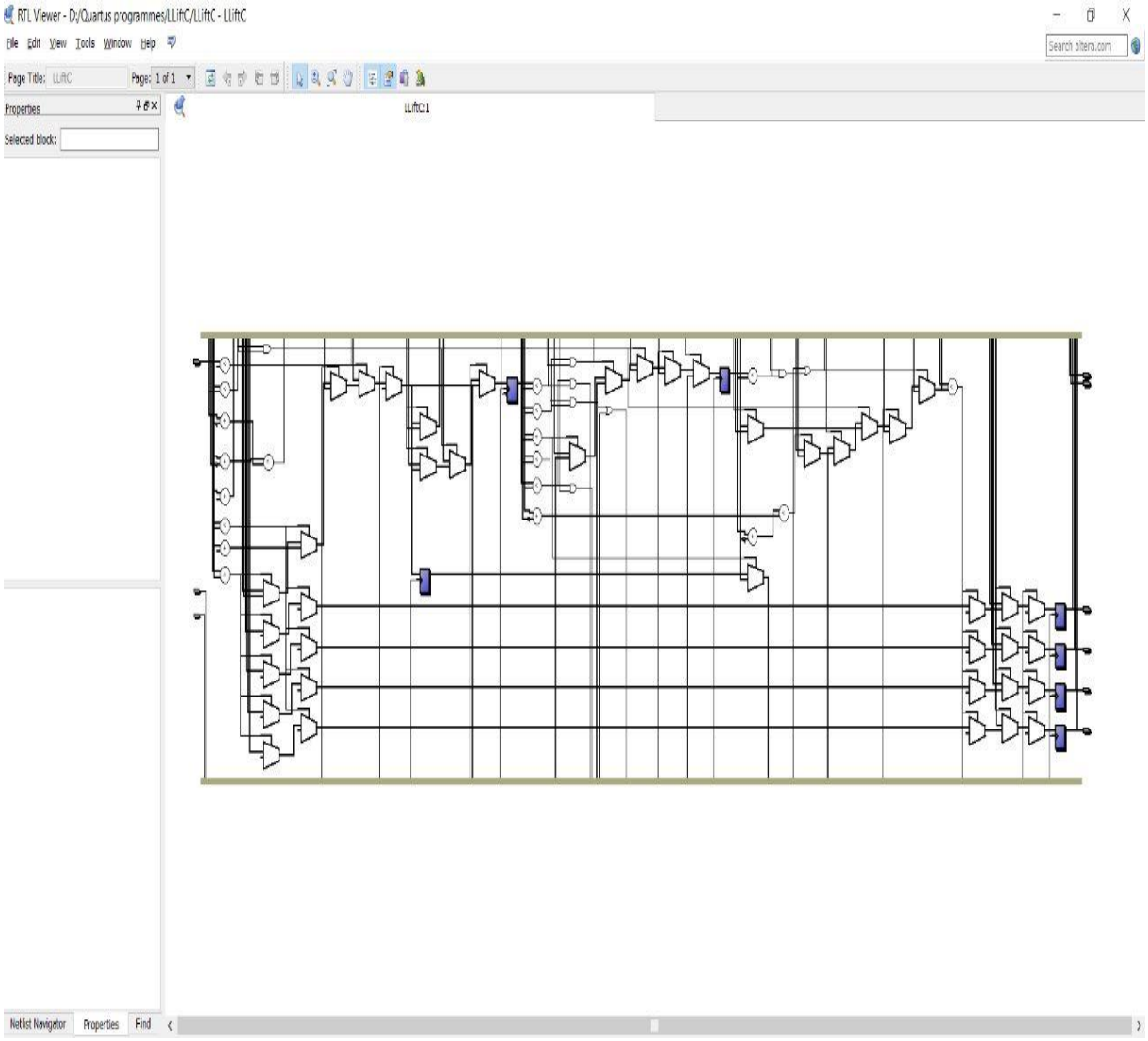


VERILOG CODE

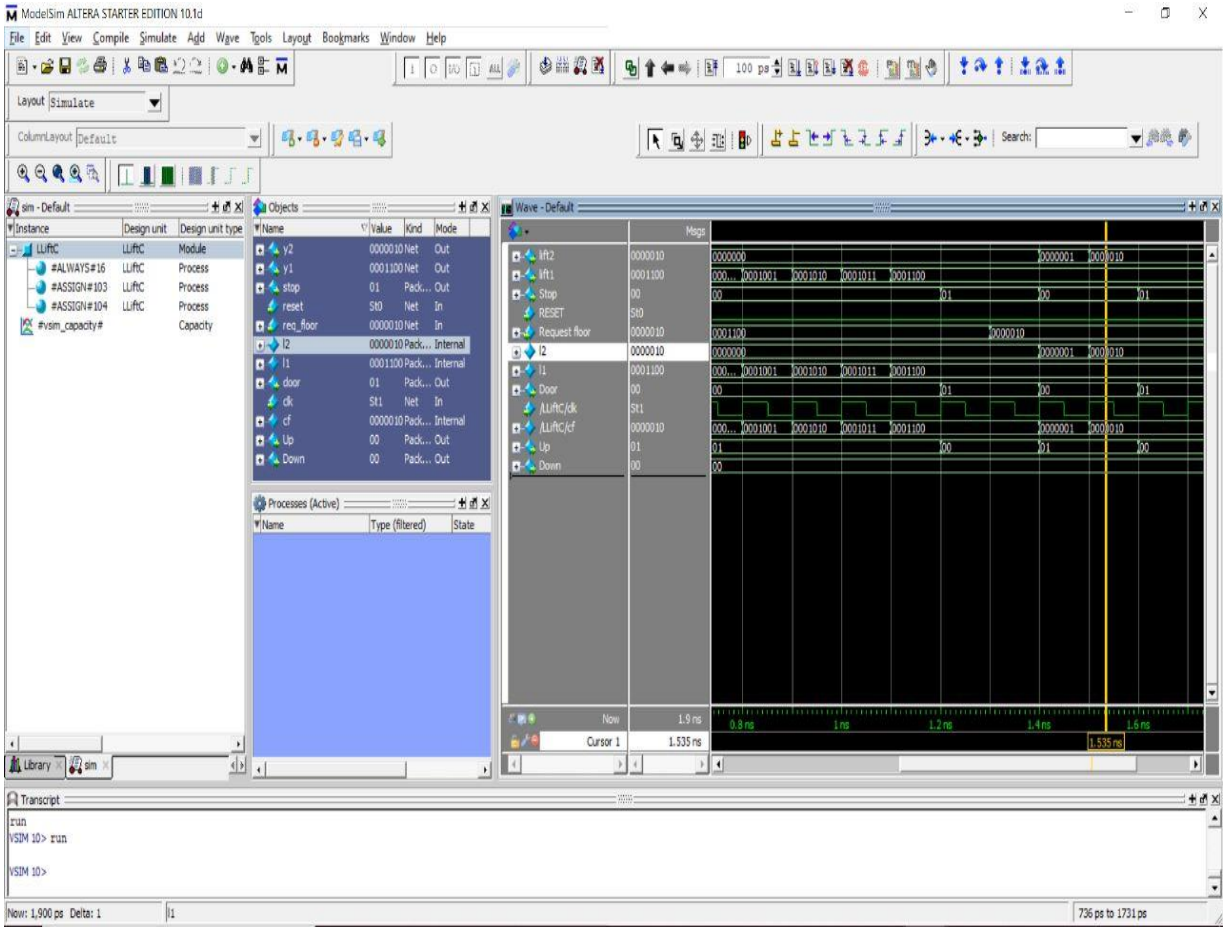
```
module LLiftC(clk,reset,req_floor,stop,door,Up,Down,y1,y2);

input clk,reset;

input [6:0] req_floor;
output reg[1:0] door;
output reg[1:0] Up;
output reg[1:0] Down;
output reg[1:0] stop;

output [6:0] y1;
output [6:0] y2;
reg [6:0] cf;
reg [6:0] l1 = 6'd0;
reg [6:0] l2 = 6'd0;
always @ (posedge clk)
begin
  if(((l2 > l1) && (l1 > req_floor)) || ((req_floor > l1) && (l1 > l2)))
          cf = l1;
        else if(((l1 > l2) && (l2 > req_floor)) || ((req_floor > l2) && (l2 > l1)))
          cf = l2;
        else if((l1 > req_floor) && (req_floor > l2))
        begin
          if((l1 - req_floor) > (req_floor - l2))
                cf = l2;
              else
                cf = l1;
        end
        else if((l2 > req_floor) && (req_floor > l1))
        begin
          if((l2 - req_floor) > (req_floor - l1))
                cf = l1;
              else
                cf = l2;
        end
        else if(l1 == l2)
          cf = l1;


        if(reset)
  begin
    cf=6'd0;
    stop=6'd1;
    door = 1'd1;
    Up=1'd0;
    Down=1'd0;
  end
  else
  begin
    if(req_floor < 6'd61)
    begin
```

```
        if(req_floor < cf )
        begin
          cf=cf-1;
          door=1'd0;
          stop=6'd0;
          Up=1'd0;
          Down=1'd1;
        end

        else if (req_floor > cf)
        begin
          cf = cf+1;
          door=1'd0;
          stop=6'd0;
          Up=1'd1;
          Down=1'd0;
        end

        else if(req_floor == cf )
        begin
          cf = req_floor;
          door=1'd1;
          stop=6'd1;
          Up=1'd0;
          Down=1'd0;
        end
      end
    end
          if(((l2 > l1) && (l1 > req_floor)) || ((req_floor > l1) && (l1 > l2)))
            l1 = cf;
          else if(((l1 > l2) && (l2 > req_floor)) || ((req_floor > l2) && (l2 > l1)))
            l2 = cf;
          else if((l1 > req_floor) && (req_floor > l2))
          begin
            if((l1 - req_floor) > (req_floor - l2))
                    l2 = cf;
                  else
                    l1 = cf;
          end
          else if((l2 > req_floor) && (req_floor> l1))
          begin
            if((l2 - req_floor) > (req_floor - l1))
                    l1 = cf;
                  else
                    l1 = cf;
          end
          else if(l1 == l2)
              l1 = cf;

end

assign y1 = l1;
assign y2 = l2;


endmodule
```

RTL SIMULATION



SIMULATION


ModelSim ALTERA STARTER EDITION 10.1d
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
sim - Default
Objects
Wave - Default
Processes (Active)
Transcript
run
VSIM 10> run
VSIM 10>

CONCLUSIONS

The report demonstrates how to construct an Elevator Controller in Verilog HDL and verify it with ModelSim Altera.. Due to its numerous benefits over traditional schematic-based design, Verilog HDL was prioritised in the design of this project. By optimising to fulfil the intended functionality early on, functional verification may be done. In the proposed design of a Elevator Controller Verilog, The RTL code is designed to regulate the lift moment in response to the request. The RTL has been confirmed and put into practice. In this work, To achieve the logic in an optimum method, the real-time lift controller will be developed with Verilog HDL code utilizing a Finite-State Machine (FSM) model. According to the FSM technology the elevator process can be defined with the help of different states. There is a transition from one floor to the next in the FSM technology. Every conceivable path is assigned a path, and the computer code for the elevator controller is written using the FSM concept. The while program is designed in such a way that there are desired switches on each floor, as well as within the elevator, to regulate user instructions.

Finally, the working of the elevator controller is been verified, tested and the results have been plotted.

REFERENCES

Design of Elevator Controller using Verilog HDL

https://www.slideshare.net/visheshsingh19/design-of-elevator-controller-using-verilog-hdl

https://iopscience.iop.org/article/10.1088/1757-899X/225/1/012137/pdf

https://www.youtube.com/watch?v=34q6AIcZvhk&ab_channel=Mr.SunilKumarG.R