Digital Design

CS223

# Emergency Elevator

# Project Report

Alkım Önen

# Description ot the Code

## 8x8 Display

I used 8x8 display module which was given ready. I examined the sample project and used its code to connect device with my project. Then I studied what represents each light on display. After I learned how to light them to red and blue, I wrote two different parts of code. First was for the number of people on floors. For this one I simple lit the red light when a passenger is added to floor and put out the lights when elevator took people. Second part was for elevator and its movements in different states. For each state I wrote different codes so the elevator would look like its moving up and down.

## 4x4 Keypad

Similar with 8x8 display, I adapted sample project to my project. I modified it so the third and fourth buttons on second to fourth lines will increase and decrease the number of passengers in related floors.

## 7-Segment Display

For this module, I took 7-segment module and modified it so the digit on the far left would simulate the movement of the elevator. As the segment can show hexadecimal numbers, I changed the representation of hexadecimal numbers “a,b,c,d,e,f”. In my implementation, I coded that each of these would only light a line of the digit so I can light them one by one. Other digits took record of the time passed from the execution to the end of the scenario.

## Reset and Scenerio Keys

I used the keys on Basys3 board to execute and reset scenario. The button on top starts the execution of the scenario. One in the middle resets the timer and one at the bottom resets the whole system.

## Elevator

I wrote all systemverilog codes in my Elevator module except ready modules (8x8 display and 4x4 keypad) and modified 7-seg module. I made the state changes and its controls here. As I will explain further later in the report, every state checks different conditions with if-else statements. I also coded the adding person function of keypad and lighting different colors function of the display.

I had 4-bit numbers to store the number of people in floors and people in elevator. When elevator stops on the floors, I decreased the number of people on a specific floor and added them to the number of people elevator is carrying. I simulated it with 8x8 display.

I had two boolean expressions for elevator movement. One is up and other one is down. I changed the values of them in states so the system can understand which direction its moving. When the elevator is on floors, I made both of them have the value 0 so system can simulate it has stopped on a floor. When the elevator started moving I coded the far left digit's value would change between 9 and 16 for 250 milliseconds so it will look like its moving.

I didn't have an extra some-bit number for timing. I simply checked digit by digit and increased them while they are smaller than 10. I checked that if a digit is nine, I made it 0 and added one to the digit on its left. For each second I increased the value of far right corner by one. Also when the user presses resetTimer button, the system assigns the value 0 to all three digits. There was a boolean expression called "timer", so the system would understand when the scenario is executed. I made its value 1 when the execution starts and 0 when the system comes to the stop state.

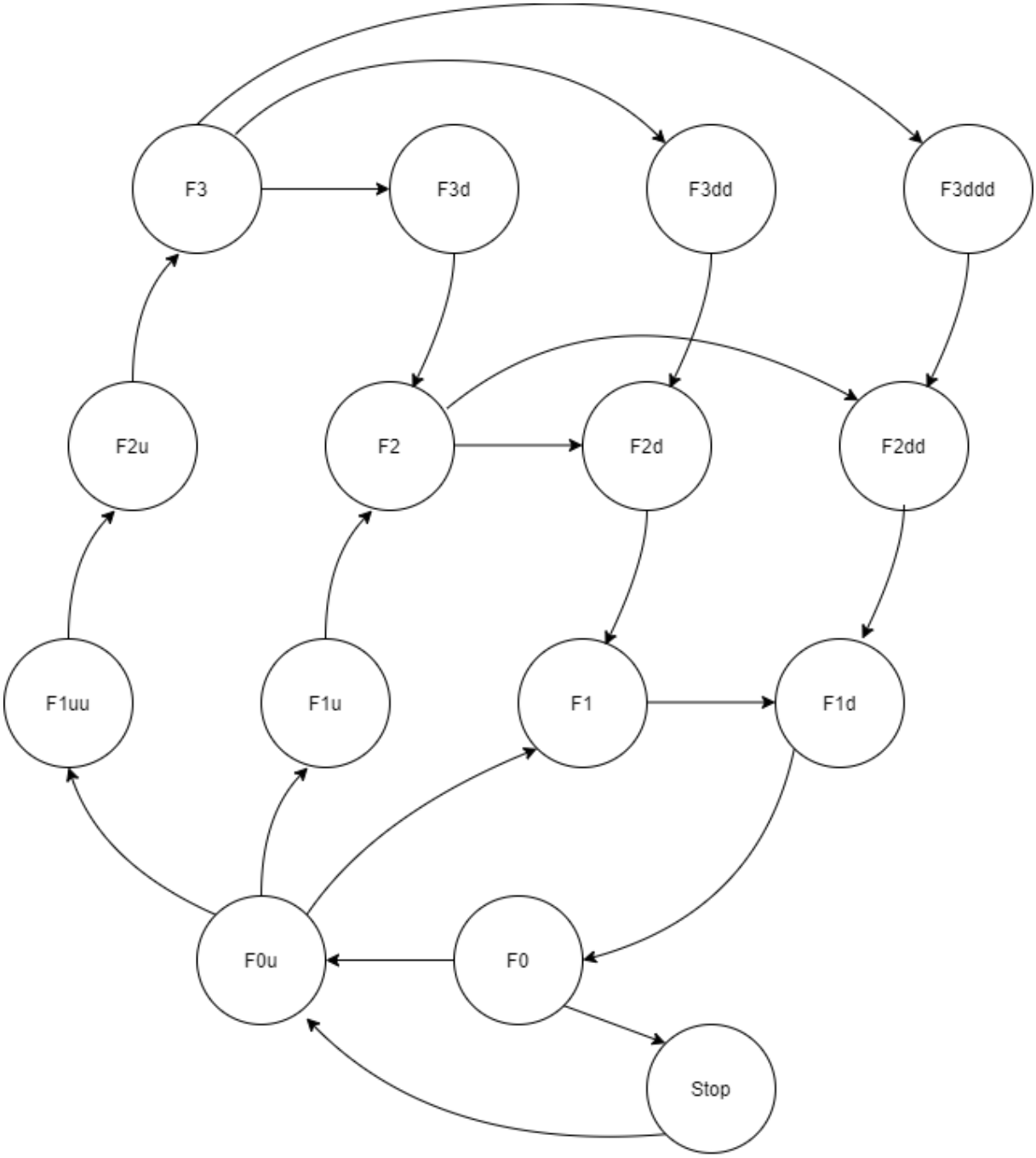
In my design I have fourteen different states. One of them is stop state. This state is the state where user can add people to floors and then start execution. When there are no people on floors, system returns to this state and stops the timer. Also when user presses resetSystem button, system returns to this state. I had four floor stop states. In these states I checked which floor should the elevator go and did the process of loading and unloading people. The system spends two seconds on these states. I also had ten floor transition states. The system spends three seconds in this states and moves the elevator to one floor up or down.

For timing, I used a similar counter code from one of the sample projects. I adapted it to my states. For transition states system counts to 300.000.000 and for floor states system counts to 200.000.000. When count reaches these values state performs its task and assign counter to 0. I also used another variable movement_counter which is for movement in first digit of 7-segment module. When this count reaches 25.000.000, corresponding value increases or decreases according to movement direction. This way, far left digit simulates movement in transition states.

## State Encoding Table

| State names | State coding |
|---|---|
| F0 | 4d'0 |
| F1 | 4d'1 |
| F2 | 4d'2 |
| F3 | 4d'3 |
| F1u | 4d'4 |
| F1uu | 4d'5 |
| F1d | 4d'6 |
| F2u | 4d'7 |
| F2d | 4d'8 |
| F2dd | 4d'9 |
| F3d | 4d'10 |
| F3dd | 4d'11 |
| F3ddd | 4d'12 |
| F0 | 4d'13 |
| Stop | 4d'14 |

# State Transition Diagram



# HLSM States

State 0: F0

Unloads the people from elevator
Goes to state 13 (F0u)

State 1: F1

If there are more than 4 people, loads 4 people into elevator and goes to state 6 (F1d)
Else loads all passengers waiting left into elevator and goes to state 6 (F1d)

State 2: F2

If there are more than 4 people, loads 4 people into elevator and goes to state 9 (F2dd)
Else if there are passengers on first floor and sum of passengers in elevator, first and second floor is less than or equal to 4, loads all people into elevator and goes to state 8 (F2d)
Else loads all people into elevator and goes to state 9 (F2dd)

State 3: F3

If there are more than 4 people, loads 4 people to elevator and goes to state 12 (F3ddd)
Else if there are passengers on second floor and the sum of passengers in second and third floor is less than or equal to 4, loads all people into elevator and goes to state 10 (F3d)
Else if there are passengers on first floor and the sum of passengers in first and third floor is less than or equal to 4, loads all people into elevator and goes to state 11 (F3dd)
Else loads all people into elevator and goes to state 12 (F3ddd)

State 4: F1u

Goes to state 2 (F2)

State 5: F1uu

Goes to state 7 (F1uu)

State 6: F1d

Goes to state 0 (F0)

State 7: F2u

Goes to state 3 (F3)

State 8: F2d

Goes to state 1 (F1)

State 9: F2dd

Goes to state 6 (F1d)

State 10: F3d

Goes to state 2 (F2)

State 11: F3dd

Goes to state 8 (F2d)

State 12: F3ddd

Goes to state 9 (F2dd)

State 13: F0u

If there are more than 3 people on third floor, goes to state 5 (F1uu)
Else if there are more than 3 people on second floor, goes to state 4 (F1u)
Else if there are mode than 3 people on first floor, goes to state 1 (F1)
Else if there are passengers on third floor, goes to state 5 (F1uu)
Else if there are passengers on second floor, goes to state 4 (F1u)
Else if there are passengers on first floor, goes to state 1 (F1)
Else goes to state 14 (Stop)

State 14: Stop

Adds passengers to the floors or removes them from the floors
If execute button is pressed, scenario with current number of people starts