A Report on implementation of Hospital rover FSM

Sidharth V Menon
Neha Ann Biju
Saintgits College of Engineering
16 June 2023

CONTENT

Abstract:

This report explores the design and implementation of a hospital rover FSM(Finite State Machine). This project is aimed at tracking the location of a rover between rooms in a hospital floor, it performs 8 state transitions across 8 rooms. The design and implementation is done using Verilog HDL(Hardware Description Language). Verilog HDL is used to design the FSM itself and to design a testbench for the design to verify the working. The results obtained from the test bench are analysed and conclusions are drawn. Possible improvements and enhancements are discussed.

Introduction:

There are many rooms in a hospital that require lowering the presence of personnel due to high hygiene requirements. This rover will help to transport medical supplies into such rooms and help decrease the human presence in such rooms even further. The FSM is designed to track the movement of the rover through each of the rooms(8 in our case).

Methodology:

The main steps for designing an FSM are:

- The design specifications and requirements are noted.
- State diagram for the FSM is constructed. A state diagram shows all the transitions and states that a FSM covers.
- Based on the state diagram Verilog code is developed to undergo the transitions observed in the state diagram.
- Then a testbench is written to validate the functioning of the FSM so as to verify that the code follows the state diagram accurately.

Design specifications:

There are assumed to be that the rover will traverse 8 rooms it will decide which location it travels to based on a “**move switch”** which will be either “1” or “0” and it’s current location.

The Head nurse room is considered as a focal point and has many locations that return to it as it is the refilling point and the point from which all items required are dispensed. Hence, the Head nurse room is chosen as the starting location and the reset location.

The FSM will be a Mealy FSM as it considers both the current location and the input from a switch as it’s conditional for choosing a state. A Mealy FSM will be more consistent and provide more reliable operation which in this case is of very high importance.

Assumptions:

The rover moves freely without blockade and all other hardware used are working correctly.

State definitions:

The rover will have transition states based on a 0 or 1 input on the move switch.

The movement of the FSM based on the switch will follow the truth table given below:

CURRENT STATE	MOVE SWITCH 1	MOVE SWITCH 0
Head nurse room	Move to Isolation room	Stay state
Isolation room	Move to ICU	Move to Head nurse room
ICU	Move to Contact Precaution Room	Move to Negative pressure room
Contact precaution room	Move to burns unit	Move to Airborne infection room
Burns unit	Move to ICU	Move to CCU
CCU	Move to Airborne infection room	Move to Head nurse room
Airborne infection room	Move to Negative pressure room	Move to Head nurse room
Negative Pressure room	Move to ICU	Move to Head nurse room

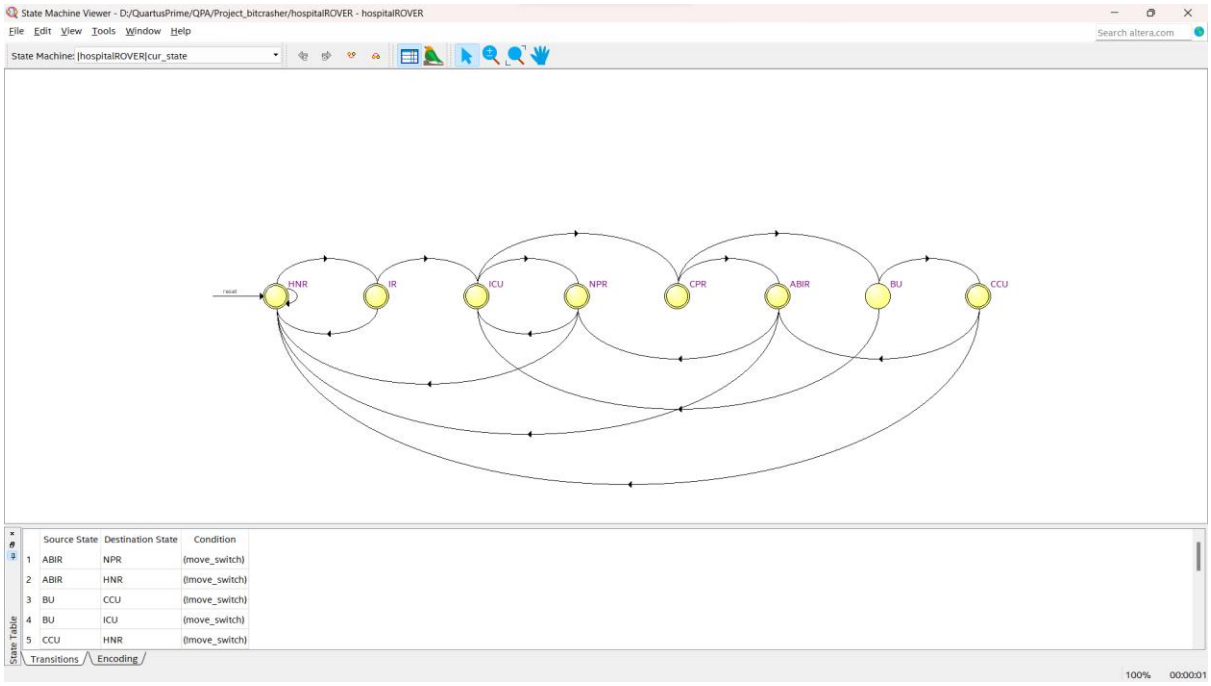

Figure 1:State Diagram of Rover

Verilog Code:

BitCrasher/HospitalRover-FSM (github.com)

Algorithm:

1. The module has various outputs and inputs such as (clk,reset,move_switch) and (current_loc). Clock, reset are basic inputs for any hardware device. The **move_switch** input is used to control the next transition of the rover from its current state.
2. Constants are assigned with the corresponding binary code for each room as listed below:

ROOM NAME	BINARY CODE
Head nurse room	000
Isolation room	001
Contact precaution room	010
Airborne infection room	011
Negative pressure room	100
ICU	101
CCU	110
Burns unit	111

3. Internal registers are declared for the variables “**next_state**” and “**cur_state**” since they repeatedly change value each iteration they store the location each iteration.
4. The output register is declared with the name “**current_loc**”.
5. The always@(move_switch or cur_state) block takes the move switch input and the current state and makes the required transition into the next state of the rover.
6. The always @(posedge clk or posedge reset) block resets the location when the reset switch is triggered otherwise it updates the current state to the next state.
7. The always@(cur_state) block updates the current location with the current state.

Functional verification:

The Verilog code for the hospital rover FSM is verified by using a test bench. The testbench is used to input various patterns for the move switch that cause the rover to move in varied patterns and if all the states are covered successfully the rover is functionally verified. The functional simulation of the FSM is done using Modelsim Altera and the functionality is verified by cross checking with the state diagram of the rover and if they match the FSM is considered functionally verified.

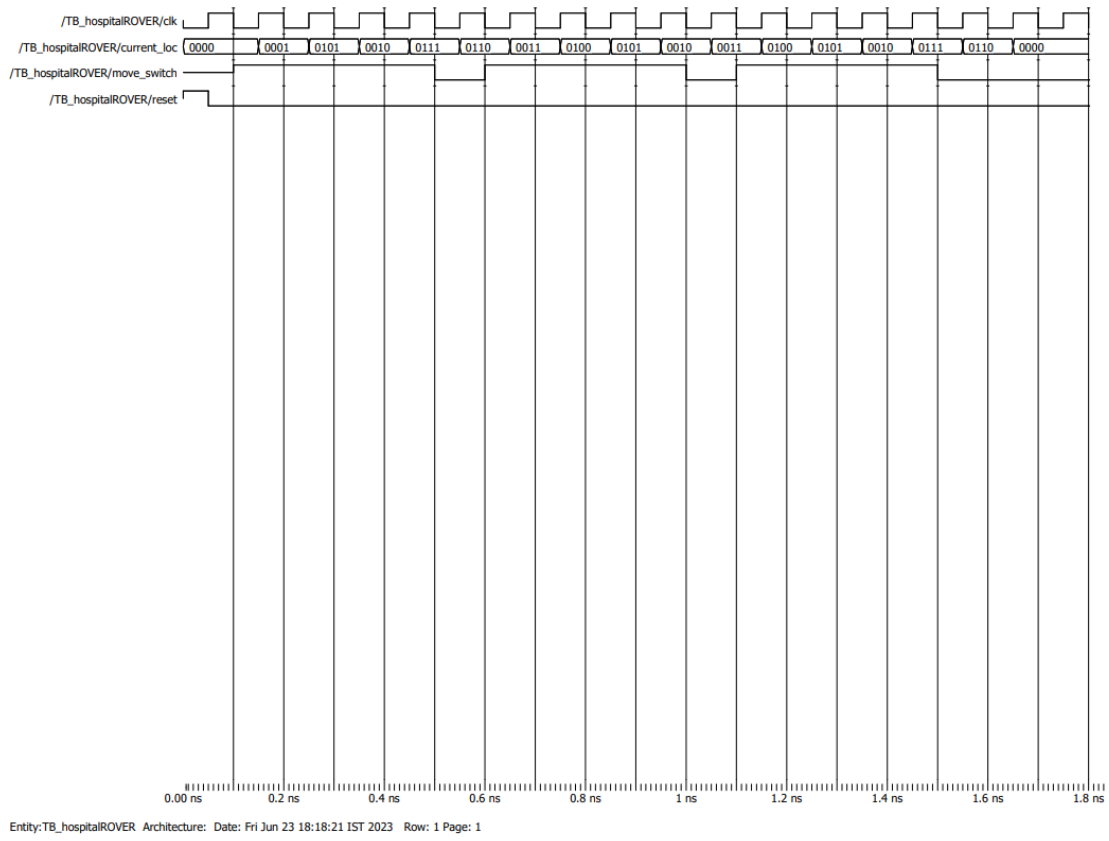

Figure 2:Functional Verification using ModelSim Altera

Results and Observations

The area and power estimations of the hospital rover FSM are essential in assessing the performance and viability of the FSM. Through analysis using the tools provided in Intel Quartus Prime lite software the Area and Power estimates were found.

- Area estimate:
  - The hardware resources required on the Cyclone V FPGA board are measured as part of the area estimate. It comes out to be $\frac{6}{32070}$ ALM. This implies that the FSM only utilises a very small fraction of the available resources.
- Power estimate:
  - The amount of power consumed for proper operation of the FSM is estimated in this analysis. This is calculates using clock frequency, toggle rate and circuit capacitance. The power estimate is around 10.50mW(milliWatts)

Conclusion:

The design and functional verification of the hospital rover FSM have been completed successfully, the functional simulation shows that the FSM performs the required functions effectively and efficiently. It provides reliable information on the location of the rover between the 8 rooms of the hospital.

Future improvements:

The ability to add custom traversal locations and expand the adaptability of the FSM. The ability to track more locations are two of the many possible improvements for the FSM. These improvements would make the FSM more widely usable and find wider applications for it.

References:

Intel Verilog HDL basics

Sequence detector example