Module Title: Advanced Digital Design

Module Code:

Assignment 2:

Designing a Finite State Machine Using VHDL

Student Registration:

Supervisor: Mr. G.H. Rohan O. Dharmarathna

Submission Date: 15th of Jan 2021

**CONTENT**

[1 LIST OF TABLES 2](#_Toc61542315)

[2 LIST OF FIGURES 2](#_Toc61542316)

[3 INTRODUCTION 3](#_Toc61542317)

[3.1 Design brief of the FSM 3](#_Toc61542318)

[4 BACKGROUND 5](#_Toc61542319)

[4.1 Mealy State Machine 5](#_Toc61542320)

[4.2 Moore State Machine 5](#_Toc61542321)

[4.3 All Possible State Transitions of the Design 6](#_Toc61542322)

[4.4 Clock Signal 7](#_Toc61542323)

[4.5 Common Applications of FSM 7](#_Toc61542324)

[5 VHDL CODES 8](#_Toc61542325)

[5.1 VHDL Codes of the Designed FSM 8](#_Toc61542326)

[5.2 Testbench Code of the Designed FSM 10](#_Toc61542327)

[6 SIMULATION 13](#_Toc61542328)

[6.1 Simulation Results for Start\_st State 14](#_Toc61542329)

[6.2 Simulation Results for State Change (Start\_st to Task1) 15](#_Toc61542330)

[6.3 Simulation Results for State Change (Task1 to Task2) 16](#_Toc61542331)

[6.4 Simulation Results for State Change (Task2 to Termin) 17](#_Toc61542332)

[6.5 Simulation Results for State Change (Termin to Start\_st) 18](#_Toc61542333)

[6.6 Simulation Results for State Change (Start\_st to Task2) 19](#_Toc61542334)

[6.7 Simulation Results for State Change (Task2 to Task 1) 20](#_Toc61542335)

[6.8 Simulation Results When No Input Condition is Satisfied 21](#_Toc61542336)

[7 CONCLUSION 22](#_Toc61542337)

[8 REFERENCES 23](#_Toc61542338)

# LIST OF TABLES

[Table 1: Inputs, states and outputs of the designed FSM 3](#_Toc61542346)

[Table 2: Outputs of the design 4](#_Toc61542347)

[Table 3: All possible state transitions of the designed FSM 6](#_Toc61542348)

[Table 4: VHDL code of the designed FSM. 8](#_Toc61542349)

[Table 5: Test bench code for designed FSM 10](#_Toc61542350)

# LIST OF FIGURES

[Figure 1: Block diagram of the FSM 3](#_Toc61542354)

[Figure 2: State diagram of the FSM 4](#_Toc61542355)

[Figure 3: Mealy state machine block diagram 5](#_Toc61542356)

[Figure 4: Moore state machine block diagram 5](#_Toc61542357)

[Figure 5: Full simulation results of the FSM 13](#_Toc61542358)

[Figure 6: Simulation waveform for Start\_st state 14](#_Toc61542359)

[Figure 7: Simulation waveform for State Change (Start to Task1) 15](#_Toc61542360)

[Figure 8: Simulation waveform for state change (Task1 to Task2) 16](#_Toc61542361)

[Figure 9: Simulation waveform for state change (Task2 to Termin) 17](#_Toc61542362)

[Figure 10: Simulation waveform for state change (Termin to Start\_st) 18](#_Toc61542363)

[Figure 11: Simulation waveform for state change (Start\_st to Task2) 19](#_Toc61542364)

[Figure 12: Simulation Waveform for state change (Task2 to Task 1) 20](#_Toc61542365)

[Figure 13: Simulation waveform when no input condition is satisfied 21](#_Toc61542366)

# INTRODUCTION

A finite-state machine (FSM) or simply a state machine is mathematical tool which is used for computation. In digital electronics, Finite State Machine is used for designing synchronous sequential circuits when it has finite number of states.

## Design brief of the FSM

Here in this assignment we are going to implement finite state machine in VHDL based on the parameters given in the assignment and create a test bench in VHDL and simulate its behavior for all the possible transitions. This design consists 4 inputs, 4 states and 3 outputs. They are,

Table 1: Inputs, states and outputs of the designed FSM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Inputs** | a | b | c |  |
| **States** | Start\_st | Task1 | Task2 | Termin |
| **Outputs** | t | u | v |  |

Below *Figure 1* illustrate the block diagram of the FSM that we are going to design.

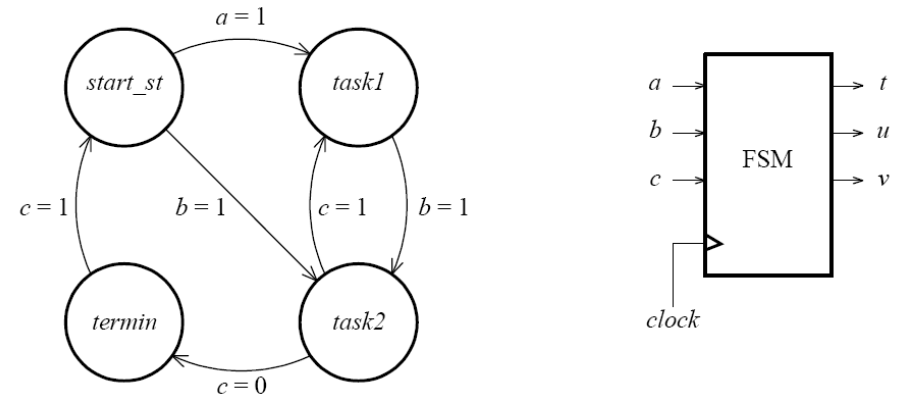


Figure 1: Block diagram of the FSM

Here in this design, states are changed based on the inputs and the present state. Below *Figure 2* illustrate the state diagram of the FSM.

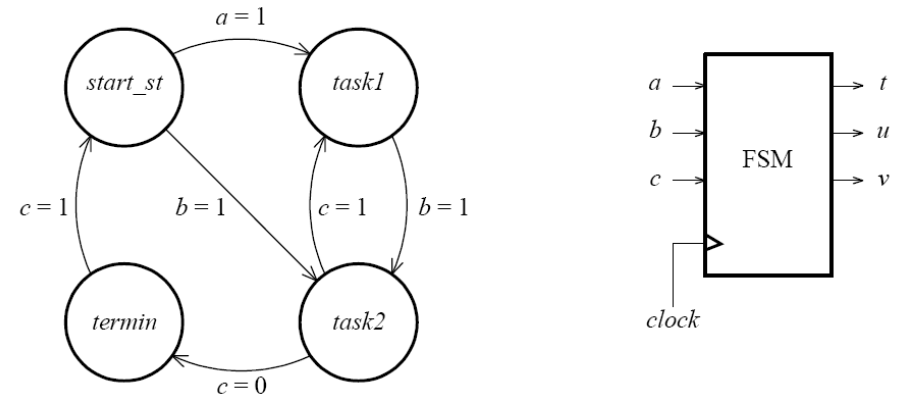


Figure 2: State diagram of the FSM

For every positive transition of the clock signal, state will be changed based on the input as per the above diagram. The outputs of the design will be change based on the state. Below *Table 2* illustrate the outputs of the design.

Table 2: Outputs of the design

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **t** | **u** | **v** |
| Start\_st | 0 | 0 | 0 |
| Task1 | 0 | 1 | 0 |
| Task2 | 1 | 1 | 0 |
| Termin | 1 | 1 | 1 |

# BACKGROUND

A finite state machine (FSM) or a synchronous sequential circuit with finite number of states, change their states for every positive or negative transition of the clock pulse with respect to the input. There are two types of FSM based on the input. They are Mealy State Machine and Moore State Machine.

## Mealy State Machine

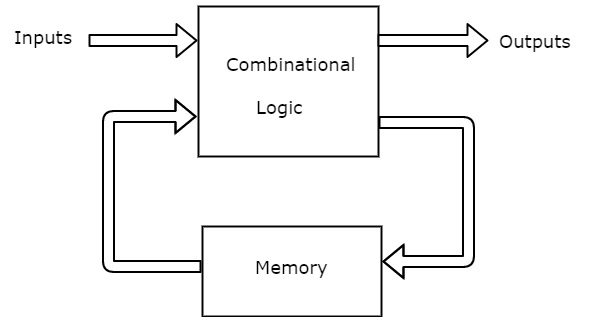
If outputs of the FSM depend on present input and present state, this state machine will be considered as Mealy State Machine. The below *Figure 3* illustrate the block diagram of Mealy state machine. 

Figure 3: Mealy state machine block diagram

## Moore State Machine

If outputs of the FSM depend only on the present state, this state machine will be considered as Moore State Machine. The below *Figure 4* illustrate the block diagram of Moore state machine.

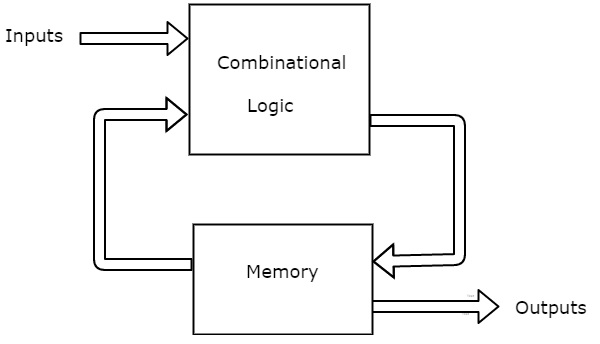


Figure 4: Moore state machine block diagram

Since this FSM design depend on both present state and present output, we have considered this design as a Mealy type state machine.

## All Possible State Transitions of the Design

As per the all the above studies and the parameters of the design given in the assignment, state transitions were designed using VHDL. Below *Table 3* illustrate the all the possible transitions with respect to the input.

Table 3: All possible state transitions of the designed FSM

|  |  |  |
| --- | --- | --- |
| **When Present state at** | **If the Input** | **Then Next state** |
| Start\_st | a=1 | Task1 |
|  | b=1 | Task2 |
|  | a or b not equal to 1 | Start\_st |
| Task1 | b=1 | Task2 |
|  | b not equal to 1 | Task1 |
| Task2 | c=1 | Task1 |
|  | c=0 | Termin |
|  | c not equal to 1 or 0 | Task2 |
| Termin | c=1 | Start\_st |
|  | c not equal to 1 | Termin |

Here in the VHDL coding of the design, we have considered only one input at certain clock signal and the other two input signals were neglected while taking decision to select the next state.

After completing designing and implementing the FSM in VHDL using ISE Design Suite Xilinx software, simulation part of the assignment was carried out using a test bench in order to verify the implementation for all the behaviors of the system based on the above *Table 3*.

## Clock Signal

synchronous sequential circuits change their states for every positive or negative transition of the clock pulse. However, in this design we have only used the rising edge of the clock signal. The period of the clock pulse is used as 10ns in the test bench and it can be change based on the requirements.

## Common Applications of FSM

FSM are commonly used in,

* vending machines
* video games
* traffic lights
* controllers in CPU
* text parsing
* analysis of protocol
* recognition of speech
* language processing

# VHDL CODES

## VHDL Codes of the Designed FSM

Table 4: VHDL code of the designed FSM.

|  |
| --- |
|  |
|  |

## Testbench Code of the Designed FSM

In this assignment, we have created a Testbench using VHDL to simulate the implemented FSM using Isim software in order to verify the implementation for all the transitions. Below *Table 5* illustrates the Testbench code of the designed FSM.

Table 5: Test bench code for designed FSM

|  |
| --- |
|  |
|  |
|  |

# SIMULATION

Here we have divided the simulation results to 8 sub categories based on the state transitions. They are

1. Start\_st state
2. Start\_st state to Task1
3. Task1 to Task2
4. Task2 to Termin
5. Termin to Start\_st
6. Start\_st to Task2
7. Task2 to Task1
8. When no input condition satisfied

Below *Figure 5* illustrate the full simulation results with all the above transitions.

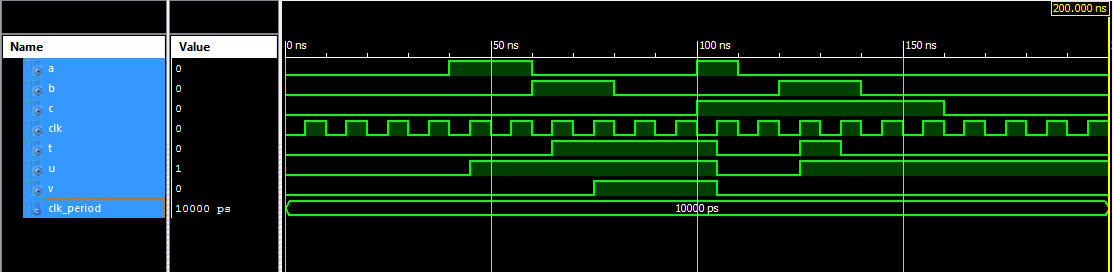


Figure 5: Full simulation results of the FSM

## Simulation Results for Start\_st State

Below figure illustrate the simulation results for Start\_st state.

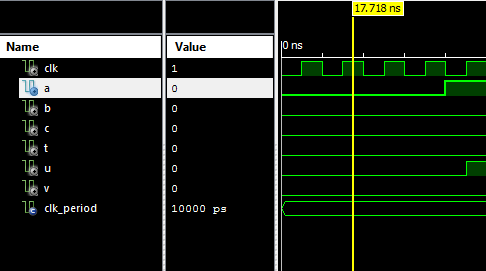


Figure 6: Simulation waveform for Start\_st state

**Comment:** As per the simulation results at Start\_st state, we can observe that,0ns to 40ns every input is 0 and output is 000 (t=0, u=0 and v=0). That means output is stayed at Start\_st state until the correct input appear. Therefore, we can verify behavior of the FSM at Start\_st.

## Simulation Results for State Change (Start\_st to Task1)

Below figure illustrate the simulation results for state change (Start\_st to Task1)

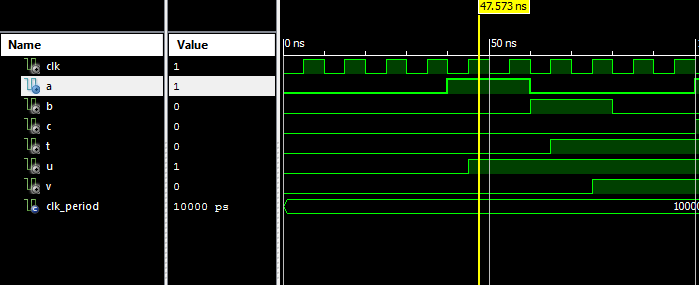


Figure 7: Simulation waveform for State Change (Start to Task1)

**Comment:** As per the simulation results shown in above *Figure 7*, we can observe that, input a become 1 at 40ns and the b and c both equals 0 (a=1, b=0, c=0). Since FSM only consider the input a=1 at this point, output changes to 010 (t=0, u=1, v=0) at 45ns with the rising edge of the clock signal. That means output change from Start\_st (000) to Task1 (010). Therefore, we can verify the behavior of the FSM when state transition from Start\_st to Task1.

## Simulation Results for State Change (Task1 to Task2)

Below figure illustrate the simulation results for state change (Task1 to Task2).

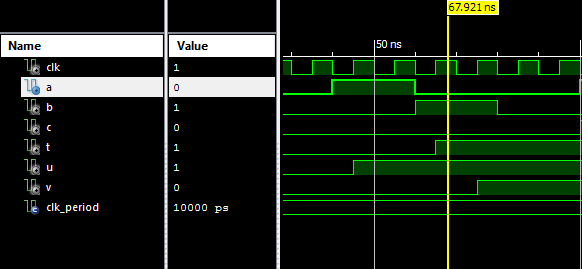


Figure 8: Simulation waveform for state change (Task1 to Task2)

**Comment:** As per the simulation results shown in above *Figure 8*, we can observe that, at 65ns input a=0, b=1 and the state before 60ns is Task1 state. According to the design, FSM only considers the input b=1 at this point. Therefore, output changes to 110 (t=1, u=1, v=0) at 65ns with the rising edge of the clock signal. That means output change from Task1 (010) to Task2 (110). Therefore, we can verify the behavior of the FSM when state transition from Task1 to Task2.

## Simulation Results for State Change (Task2 to Termin)

Below figure illustrate the simulation results for state change (Task2 to Termin).

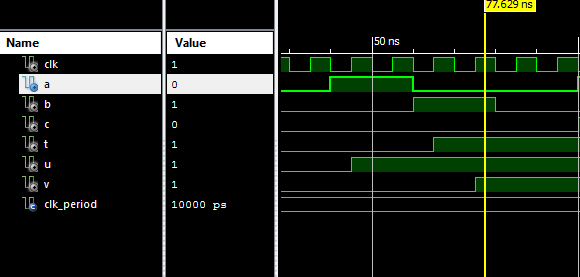


Figure 9: Simulation waveform for state change (Task2 to Termin)

**Comment:** As per the simulation results shown in above Figure 9, we can observe that, at 75ns input a=0, b=1 and c=0. The state before 75ns is Task2 state. According to the design, FSM only considers the input c=0 at this point. Therefore, output changes to 111 (t=1, u=1, v=1) at 75ns with the rising edge of the clock signal. That means output change from Task2 (110) to Termin (111) and stay at Termin until 105ns. Therefore, based on this results we can verify the behavior of the FSM when state transition from Task2 to Termin.

## Simulation Results for State Change (Termin to Start\_st)

Below figure illustrate the results for state change (Termin to Start).

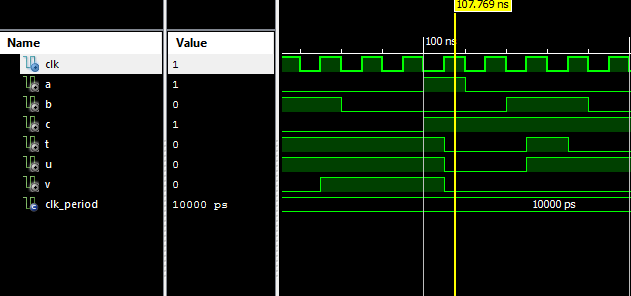


Figure 10: Simulation waveform for state change (Termin to Start\_st)

**Comment:** As per the simulation results shown in above *Figure 10*, we can observe that, at 105ns input a=1, b=0 and c=1. The state before 75ns is Termin state. According to the design, FSM only considers the input c=1 at this point. Therefore, output changes to 000 (t=0, u=0, v=0) at 105ns with the rising edge of the clock signal. That means output change from Termin (111) to Start\_st (000) and stay at Start\_st until 125ns. Therefore, based on this results we can verify the behavior of the FSM when state transition from Termin to Start\_st.

## Simulation Results for State Change (Start\_st to Task2)

Below figure illustrate the simulation results for state change (Start\_st to Task2).

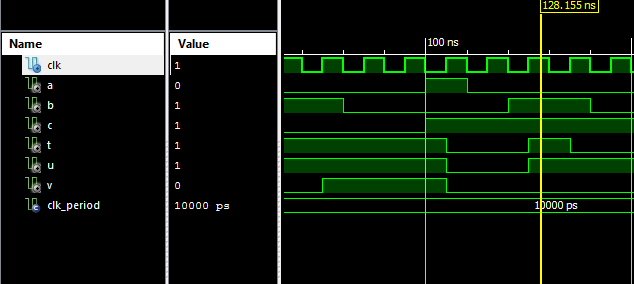


Figure 11: Simulation waveform for state change (Start\_st to Task2)

**Comment:** As per the simulation results shown in above *Figure 11*, we can observe that, at 125ns input a=0, b=1 and c=1. The state before 125ns is Start\_st state. According to the design, FSM only considers the input b=1 at this point. Therefore, output changes to 110 (t=1, u=1, v=0) at 125ns with the rising edge of the clock signal. That means output change from Start\_st (000) to Task2 (110) and stay at Task2 until 135ns. Therefore, based on this results we can verify the behavior of the FSM when state transition from Start\_st to Task2.

## Simulation Results for State Change (Task2 to Task 1)

Below figure illustrate the simulation results for state change (Task2 to Task 1).

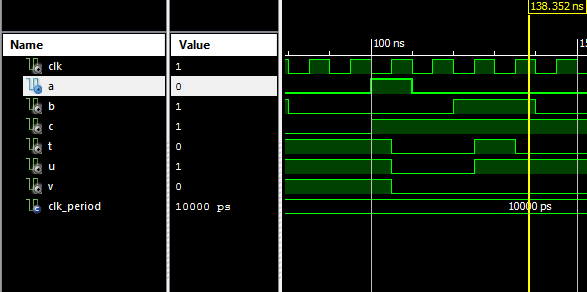


Figure 12: Simulation Waveform for state change (Task2 to Task 1)

**Comment:** As per the simulation results shown in above *Figure 12*, we can observe that, at 135ns input a=0, b=1 and c=1. The state before 135ns is Task2 state. According to the design, FSM only considers the input c=1 at this point. Therefore, output changes to 010 (t=0, u=1, v=0) at 135ns with the rising edge of the clock signal. That means output change from Task2 (110) to Task1 (010). Therefore, based on this results we can verify the behavior of the FSM when state transition from Task2 to Task1.

## Simulation Results When No Input Condition is Satisfied

Below figure illustrate the simulation results when no input condition is satisfied.

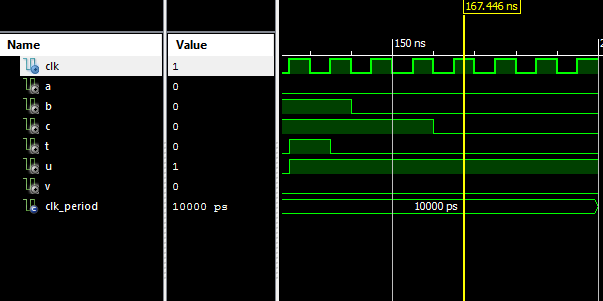


Figure 13: Simulation waveform when no input condition is satisfied

**Comment:** As per the simulation results shown in above *Figure 13*, we can observe that, after 135ns output does not changed. That means output stay at Task1 (010) until the simulation is over. As per the design conditions, present state will be the next state until right input applied to the FSM. The input b=1 is the only condition that can change the state from Task1 (010) to Task2 (110). Since b does not become 1 after the 140ns, FSM will stay at Task1 (101) until the end of the simulation. Therefore, based on this results we can verify the behavior of the FSM when no input condition is satisfied.

According to the all above simulation results from *Figure 5* to *Figure 13*, we can verify that the implemented FSM perform all the state transitions as expected in the assignment.

# CONCLUSION

As per the assignment requirements, we have implemented a Finite State Machine (FSM) in VHDL using Xilinx (ISE design suite software) and simulate its behavior using Isim Software. All the expected state transitions were achieved from the design and all the simulation results were explained based on the input and state transition. By completing this design, we have gained the knowledge about

* Synchronous sequential circuits
* Finite state machine (FSM)
* Mealy machine and Moore machine
* VHDL coding in Xilinx
* Simulating a VHDL design Isim

While implementing this system I have understand that the RESET input needed to be included in order to get a more realistic design and there are more other ways to implemented a FSM except the method that I used. Therefore, those methods are also needed to experimented in order to find the more efficient design. However, since this assignment is conducted in limited timeline, those can be experimented as a further work.

# REFERENCES

1. CircuitVerse. 2021. Finite State Machines. [online] Available at: <https://learn.circuitverse.org/docs/finitestatemachine.html> [Accessed 10 January 2021].
2. Tutorialspoint.com. 2021. Digital Circuits - Finite State Machines - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/digital\_circuits/digital\_circuits\_finite\_state\_machines.htm> [Accessed 14 January 2021].
3. En.wikipedia.org. 2021. Finite-State Machine. [online] Available at: <https://en.wikipedia.org/wiki/Finite-state\_machine> [Accessed 14 January 2021].
4. ElProCus - Electronic Projects for Engineering Students. 2021. Finite State Machine (FSM) : Types, Properties, Design And Applications. [online] Available at: <https://www.elprocus.com/finite-state-machine-mealy-state-machine-and-moore-state-machine/#:~:text=The%20finite%20state%20machine%20applications,as%20well%20as%20network%20protocols.> [Accessed 14 January 2021].