Table of contents

Figure list

Tabular list

1 Introduction

In this practical exercise, the objective is to design a sequence detector that triggers an output flag for a duration of two clock periods upon detecting the input sequence '11001'. Importantly, this detector should maintain continuous scanning of the input, even when the flag is active, in order to identify overlapping sequences like "110011001". The sequence detector wil be an synchronous Finite State Machines (FSMs), which will be complemented by the creation of FSM diagrams and truth tables. Additionally, the system will be equipped with an active-low reset.

Two approaches are considered:

Approach 1: Using a single FSM, which is a basic method suitable for simple cases but may not scale well for longer flag durations or increased overlap length.

Approach 2: Employing interacting FSMs, with FSM1 for sequence detection and FSM2 for flag management. Both FSMs operate in parallel, driven by the same clock signal, ensuring timely interaction.

2 Design Description

2.1 Approach 1 Design Description

In the first approach, we utilize a single Finite State Machine (FSM), which represents the more basic of the two proposed solutions. This FSM is responsible for both detecting the input sequence "11001" and raising the flag for a duration of two clock periods upon detection. To optimize this design, a Mealy machine is chosen due to its ability to achieve the desired functionality with fewer states.

In (Table: 1), the Mealy logic for the target input sequence "11001" is outlined, along with the corresponding states necessary to achieve this sequence. The table alternates between input values of 0 and 1 to demonstrate some of the current and next states.

In (Figure: 1), you can see a complete representation of the Mealy machine logic, showing all the states and how they change. The machine starts at state A and ends up at state F, where the flag stays up for two clock periods. This happens as the machine transitions from state E to F and can go from F back to A or to C.

Table 1: Truth table in Approach 1

Input	Current state	Next state	Output
0	A	A	0
1	A	B	0
0	B	A	0
1	B	C	0
0	C	D	0
1	D	B	0
0	D	E	0
1	E	F	1
0	F	C	1
1	F	A	1

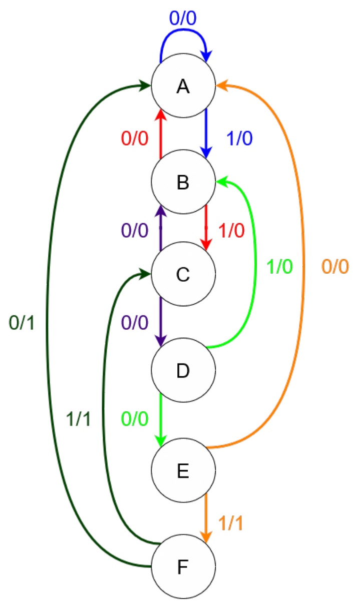


Figure 1: Finite State Machine of Approach 1

2.2 Approach 2 Design Description

In the second approach, we employ a dual Finite State Machine (FSM) strategy to achieve the desired functionality. FSM1 is dedicated to recognizing the input sequence, while FSM2 handles the task of raising and maintaining the flag Z.

For FSM1, i have chosen a Moore machine as the ideal model. Moore machines generate their outputs solely based on the current state, independent of the input. This selection aligns perfectly with the designs objectives since FSM1's primary role is to identify the target sequence (*Difference Between Mealy Machine and Moore Machine* 2023).

FSM2's role involves managing the flag Z by considering both the state of FSM1 and the clock signal. In this case, a Mealy machine is the more appropriate choice. Mealy machines allow the output to depend on both the current state and the input, making them well-suited for this task. The key advantage of using a Mealy machine for FSM2 is its ability to respond immediately to state changes in FSM1 without waiting for the next clock cycle (*Difference Between Mealy Machine and Moore Machine* 2023). This responsiveness ensures that the flag Z is raised promptly when FSM1 detects the target sequence and is held high for the required two clock periods.

Both FSM1 and FSM2 operate synchronously, driven by the same clock signal, and are connected to the same reset signal, which is active low. This synchronized operation ensures that the two FSMs work in harmony to achieve the desired sequence detection and flag-raising behavior, even when overlapping sequences are present in the input stream.

(Table: 2) outlines the Moore logic for FSM1, which is responsible for detecting the input sequence ”11001” and the corresponding states required to achieve this sequence. The table demonstrates how the current and next states alternate between input values of 0 and 1.

In (Table: 3), you can se the Mealy logic utilized by FSM2, responsible for the operation of raising the flag Z for a duration of two clock periods.

(Figure: 2) provides a comprehensive representation of the second approach, depicting FSM1 and FSM2.

(Figure: 3) presents a block diagram of the representation, including all the inputs and outputs, the clock, and the reset signal.

Table 2: Truth table of FSM1 (detecting the sequence) in Approach 2

FSM1			
	Next state		
Current state	Input = 0	Input = 1	Output
S_0(000)	S_0(000)	S_1(001)	0
S_1(001)	S_0(000)	S_2(010)	0
S_2(010)	S_3(011)	S_1(001)	0
S_3(011)	S_4(100)	S_1(001)	0
S_4(100)	S_0(000)	S_5(101)	0
S_5(101)	S_0(000)	S_2(010)	1

Table 3: Truth table of FSM2 (holding the flag) in Approach 2

FSM2			
Input	Current state	Next state	Output
0	A	A	0
1	A	B	1
0	B	A	1

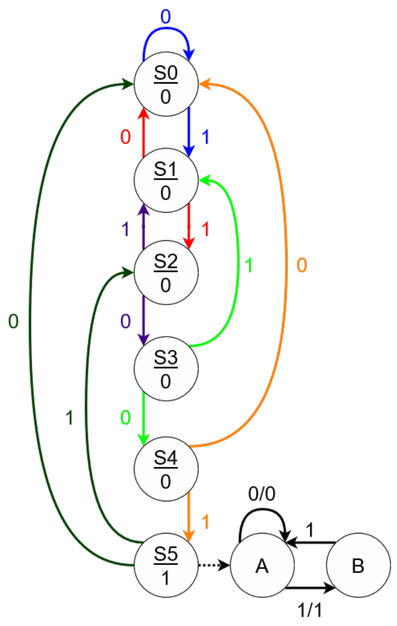


Figure 2: Finite State Machine of Approach 2

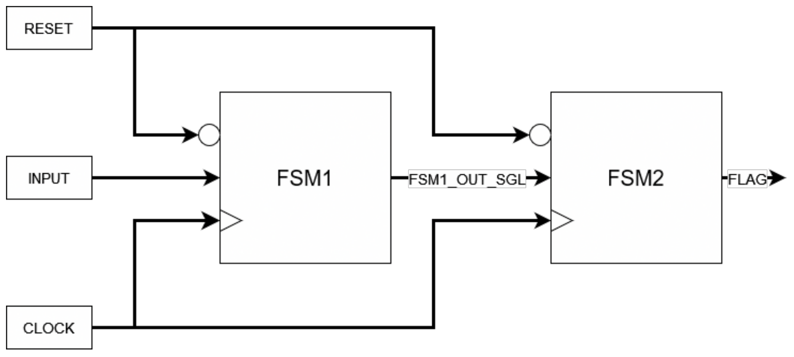


Figure 3: Block diagram of Approach 2

3 Simulation Results

The (Figure: 4) illustrates a simulation of Approach 2, which involves the following steps:

- Initial Testing: The simulation first tests all the states and raises a flag when the sequence ”11001” is encountered. This test occurs from 0 ns to 47 ns.
- Overlapping Sequence Detection: When testing all the states, we can test the detection of overlapping with the sequence ”10011001,” which results in the flag being raised twice. This test is displayed from 26 ns to 46 ns.
- Active Low Reset: The final test involves checking the behavior of the active low reset signal in the middle of the input sequences of ”11001.” This occurs at 46 ns to 60 ns, added 1 sec so we can see the rest activating.

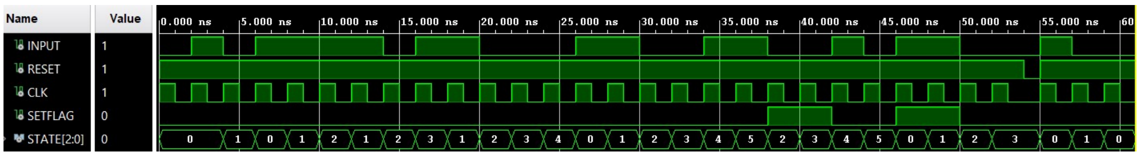


Figure 4: Simulation of part Approach 2

4 Synthesis Results

4.1 RTL

Shown below are the illustrations of the Register-Transfer Level (RTL) design for approach 2. This representation showcases the interconnected submodules employed at the highest level.

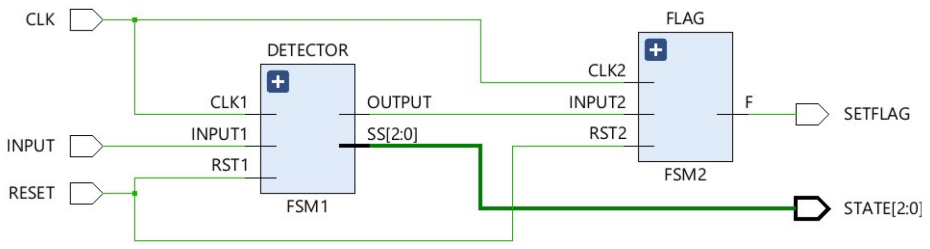


Figure 5: Top level RTL of part Approach 2

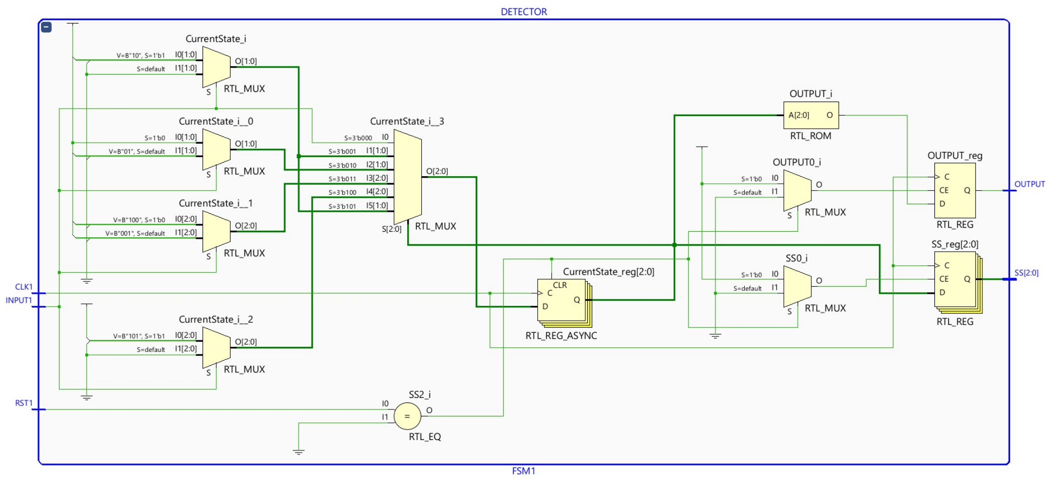


Figure 6: Detector RTL of part Approach 2

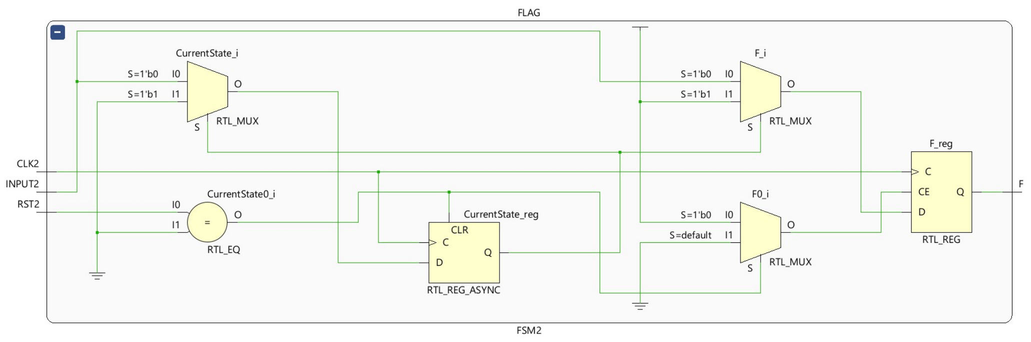


Figure 7: Flag RTL of part Approach 2

4.2 Schematic

Displayed below are the Synthesis schematics, illustrating the logical arrangement and interconnections within the design following the transformation from RTL code to gate-level logic.

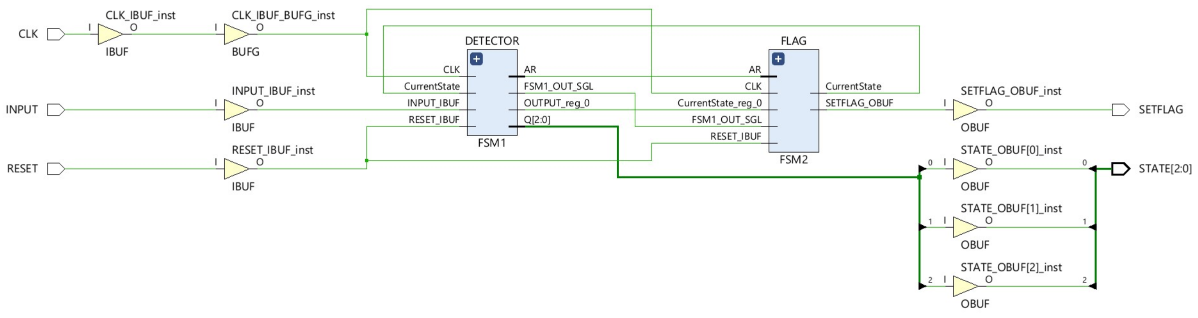


Figure 8: Top level Schematic of part Approach 2

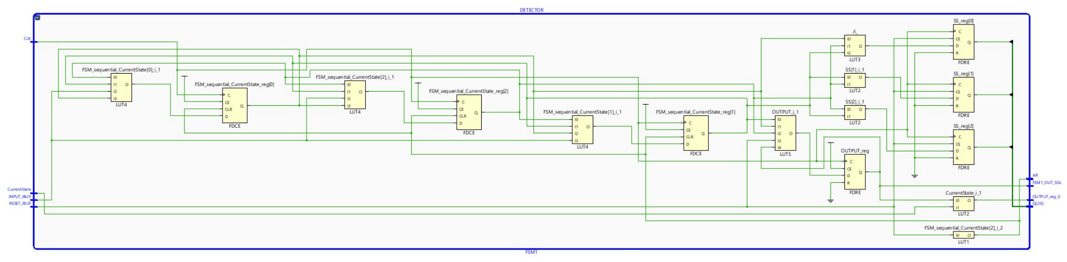


Figure 9: Detector Schematic of part Approach 2

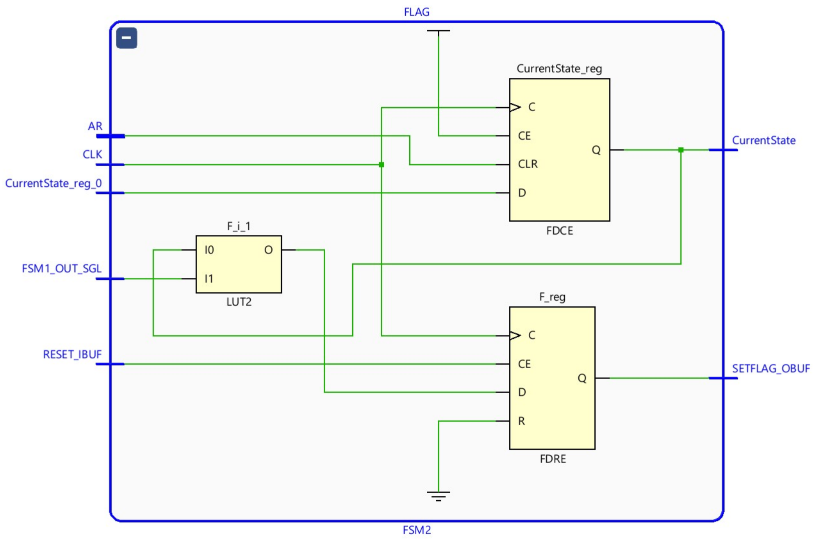
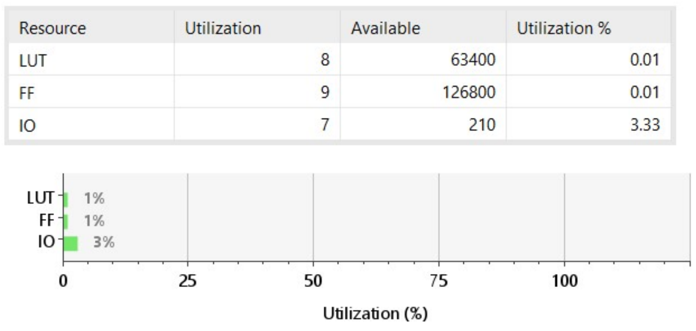


Figure 10: Flag Schematic of part Approach 2

4.3 Utilization

In (Figure: 11), we see the implementation of approach 2, showcasing the utilization of Lookup Tables (LUTs), Input/Output (IO) resources, and Flip-Flops (FF). This graphical representation provides valuable insights into the availability of these resources and their corresponding utilization percentages.



Resource	Utilization	Available	Utilization %
LUT	8	63400	0.01
FF	9	126800	0.01
IO	7	210	3.33

Figure 11: Utilization of part Approach 2

5 Discussion and Conclusion

In this practical, I successfully designed two approaches for a sequence detection system. This system triggers a flag for two consecutive clock cycles when receiving the input sequence ”11001,” and it effectively handles overlapping sequences. Additionally, I designed finite state diagrams for both methods, along with their corresponding truth tables.

Furthermore, I implemented and ran simulations for approach 2 using Vivado. Throughout the simulation, I conducted comprehensive testing of all state transitions to ensure they performed as intended. This thorough testing showcased the system’s ability to identify both non-overlapping and overlapping sequences, thus confirming its effectiveness.

One challenge I encountered during this practical was the design of Mealy and Moore state machines, as it had been a while since I last designed them. Nonetheless, implementing them in VHDL presented no difficulties, thanks to my prior experience with submodules and top-level design from previous practicals.

Overall, this practical was comparatively easier than the others, but it served as a valuable refresher for my understanding of Mealy and Moore finite state machines.

References

Difference Between Mealy Machine and Moore Machine (2023). BYJU'S Exam Prep. URL: https://byjus.com/gate/difference-between-mealy-machine-and-moore-machine/ (visited on 9th Sept. 2023).

Marking Criteria

The pracs in this course are marked to a specific criteria. This means you must demonstrate sufficient understanding and functionality and the marking will be done according to the rubric provided below. You must attempt each prac (i.e., a report must be received by the due date) to pass the course. All designs VHDL code used **must be your own work.** You are NOT permitted to use other VHDL code sources, unless directed to. Plagiarism is unacceptable and please read and understand the School Statement on Misconduct, available on the ITEE website at: http://www.itee.uq.edu.au/itee-student-misconduct-including-plagiarism . To avoid problems make sure that your VHDL code is the product of your work and do not let anybody else 'reuse' your code.

Marks	**Criteria**
Simulation	
0	Simulation not attempted or does not work
1	Simulation works only for non-overlapping sequence detection
2	Simulation works only for overlapping sequence detection
3	Simulation fully works for both overlapping and non-overlapping cases
4	Simulation fully works for both overlapping and non-overlapping cases with output retained for two clock periods
Report	
0	No evidence of content or work
1	Some content, insufficient explanation of circuit
2	Reasonable content, some explanation of circuit (both approaches are explained)
3	Good content, reasonable explanation of circuit (both approaches are explained)
4	Excellent content, good explanation of circuit (both approaches are explained)
Oral assessment	
0	No knowledge of the design **(total mark will be capped at 0%)**
1	Very little knowledge of the design **(total mark will be capped at 50%)**
2	Reasonable knowledge of the design
3	Good knowledge of the design.
4	Excellent knowledge of the design.
Total (12):	**Marker Initials:** **Date:**