DESIGN OF HIGH SPEED SEQUENCE DETECTOR USING VERILOG

[1]Dr. John Paul Pulipati, Principal, [2.] P.Anil Kumar, [3] B.Manjula,
[4] P.Venkatapathi, [5]P.Anitha
[1]principal@mrce. in,Malla Reddy College of Engineering
[2] Assistant Professor, Dept. of ECE,Malla Reddy College of Engineering
[3]Assistant Professor, Dept. of ECE,Malla Reddy College of Engineering
[4]Assistant Professor, Dept. of ECE,Malla Reddy College of Engineering
[5]Assistant Professor, Dept. of ECE, Malla Reddy College of Engineering

Abstract— There is a enormous usage of sequence detectors in digital circuits as it is the basic function and it became essential in most of the digital systems counting ALU, microprocessors and DSP. Sequential circuit's works on a clock cycle which may be synchronous or asynchronous. Sequential circuits use current inputs and previous inputs by storing the information and putting back into the circuit on the next clock cycle. This paper presents the high speed Sequence Detector in Verilog, which is a sequential state machine used to detect consecutive bits in a binary string. The flip-flops help to detect the pattern in the given string. The Sequence Detector gives for some particular sequence of inputs and outputs, whenever the desired sequence has found. And this paper shows a great vision on the design analysis of sequence detector using Verilog. The delay (1.045ns) minimized. The proposed architecture of sequence detector is synthesized in Xilinx ISE14.7.

Keywords: Sequence detector, sequential circuits, flip-flop, delay, Verilog, Mealy machine and Xilinx ISE14.7.

Introduction

FSM is a mathematical model of computation. It is an abstract machine that can be in exactly one of a finite number of states at any given time. The FSM can change from one state to another in response to some external inputs; the change from one state to another is called a transition. An FSM is defined by a list of its states, its initial state, and the conditions for each transition. Finite state machines are of two types - deterministic finite state machines and non-deterministic finite state machines. A deterministic finite-state machine can be constructed equivalent to any non-deterministic one.

The behavior of state machines can be observed in many devices in modern society that perform a predetermined sequence of actions depending on a sequence of events with which they are presented. Simple examples are vending machines, which dispense products when the proper combinationofcoins is deposited, elevators, whose sequence of stops is determined by the floors requested by riders, traffic lights, which change sequence when cars are waiting, and combination locks, which require the input of combination numbers in the proper order. The finite state machine has less computational power than some other models of computation such as the Turing machine.

II. Proposed work

A sequence detector is a sequential state machine. Sequential circuit's works with respect to a clock cycle which may be synchronous or asynchronous. The figure shows a basic diagram block of sequence detector. Sequential circuits use current inputs and previous inputs by storing the data and putting back into the circuit on the

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.38

next clock cycle.

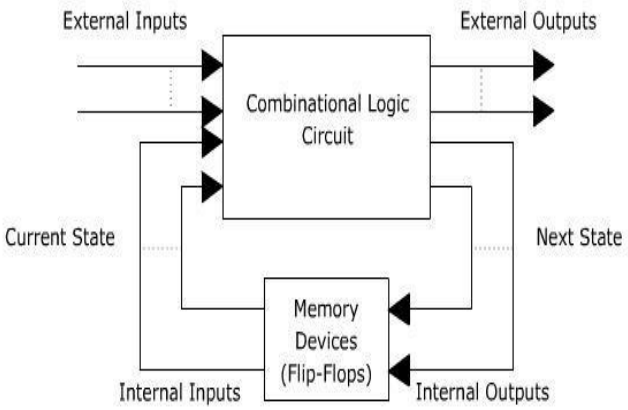


Figure 1: Block Diagram of Sequence Detector.

Finite State Machine (FSM):

A FSM is a model it is used to design sequential logic circuits. It is considered as an abstract machine that can be in one of a finite number of states. The machine is accessible in only one state at a time; the state it is in at any given time is called the current state. It can change81 from one state to another state when the triggering event or condition is introduced into the machine, this is called a transition. A specific FSM is well-defined by a list of its states, and the triggering condition for each transition. It can be implemented using models like Mealy machine and Moore machine. For this expt., we will use Mealy machine model implementation.

Types Of Sequence Detector:

There exist basically two types of sequence detectors.

That are:

1) Overlapping Sequence Detector.
2) Non- Overlapping Sequence Detector.

Overlapping Sequence Detector:

In a sequence detector that allows overlap, the final bits of one sequence can be the start of another sequence.

Non-Overlapping Sequence Detector:

The sequence detector with no overlap allowed resets itself to the start state when the sequence has been detected.

Flip-flop: A flip-flop or latch is a circuit that has two stable states and it can be used to store state data. A flip-flop is a bistable multivibrator. The circuit can be made to change state by signals applied to one or more Control inputs and will have one or two outputs, one for the normal value and one for the complement value of the stored bit. Memory elements in any sequential circuit are usually flip-flops.

Sequence detector:

Suppose a sequence detector is to be designed to detect a sequence 1101.

Then the state diagram will be:

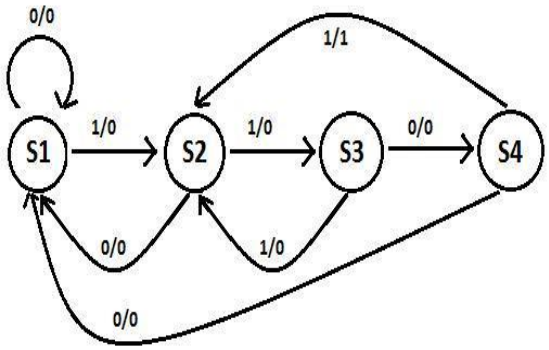
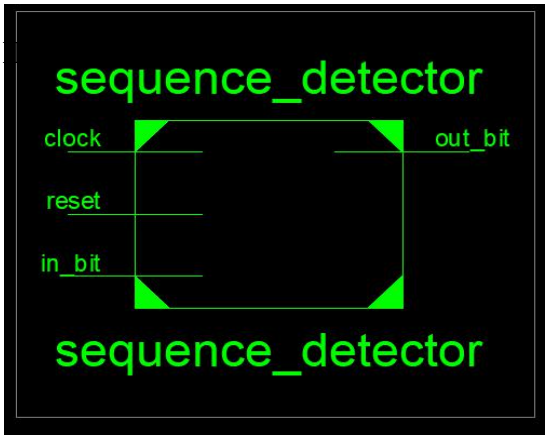


Figure 2: State Diagram of Sequence Detector.

Note that this state diagram is considering overlap i.e. if we have input 1101101 we will have output0001001.

For same input, non-overlap case will have output 0001000. Either cases are correct but we will consider only overlap case henceforth.



ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.38

Figure 3 : Top Module of Sequence Detector.

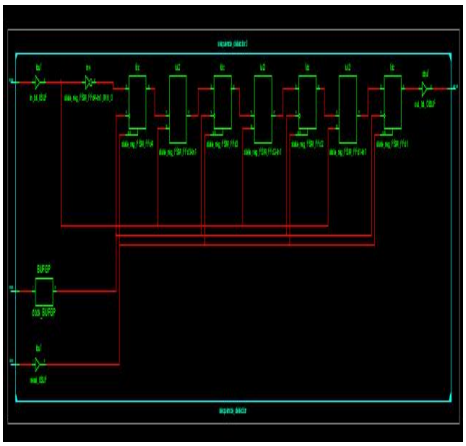


Figure 4 : Technology Schematic of Sequence Detector

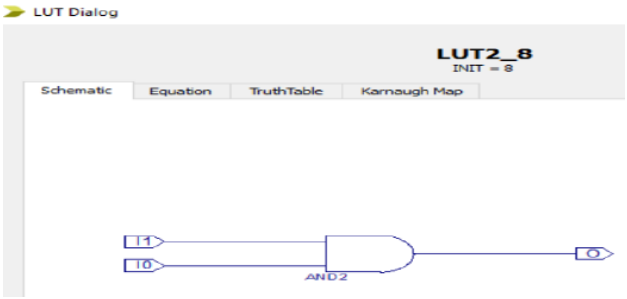


Figure 5 : LUT Schematic of Sequence Detector.

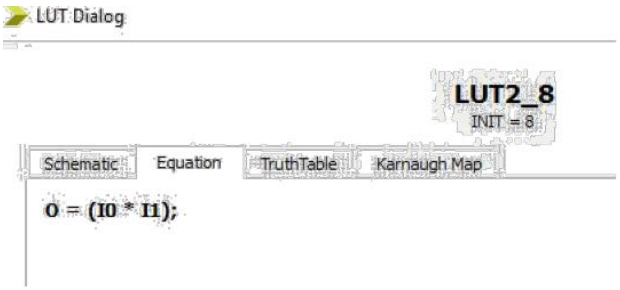


Figure 6 : LUT Equation of Sequence Detector

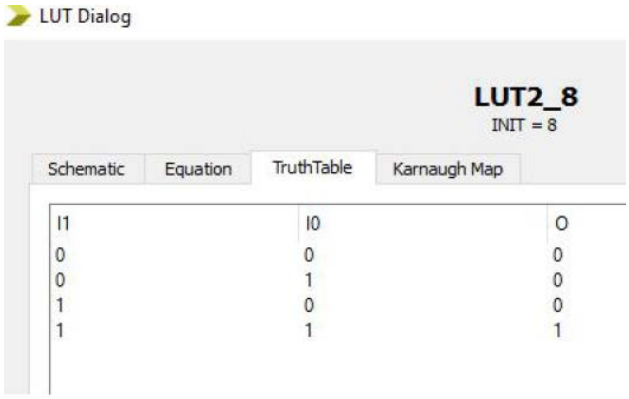


Figure 7 : LUT Truth table of Sequence Detector.

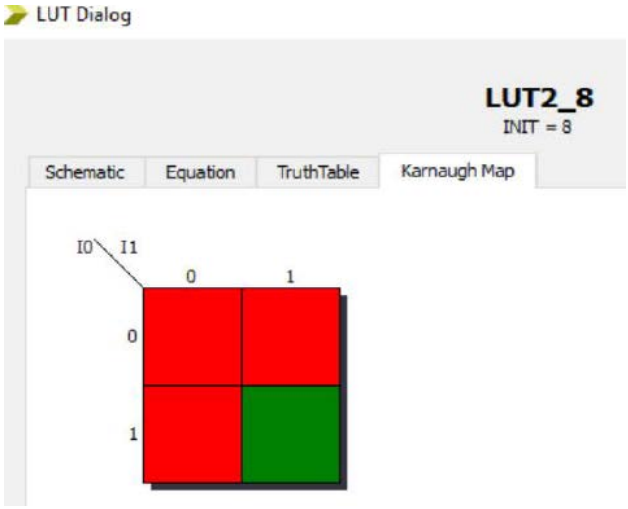


Figure 8 : LUT karnaugh map of Sequence Detector.

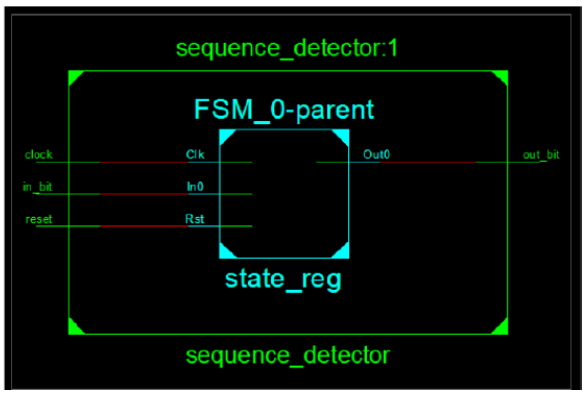


Figure 9 : Top RTL Schematic of Sequence Detector

ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-12, 2017
DOI:10.21276/ijcesr.2017.4.12.38

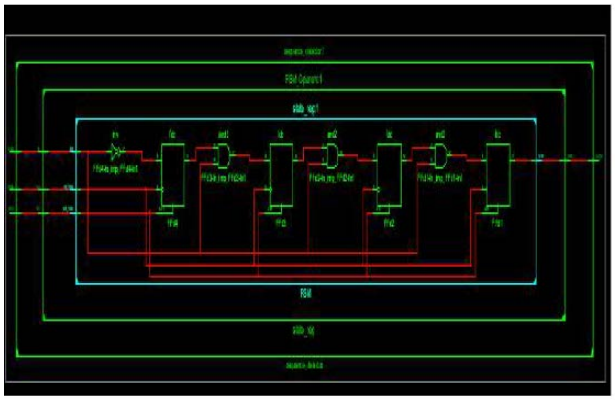


Figure 10 : RTL Schematic of Sequence Detector

Final Register Report:

Macro Statistics

Registers : 4

Flip-Flops: 4

Cell Usage :

BELS : 4

INV : 1

LUT2 : 3

FlipFlops/Latches : 4

FDC : 4

Clock Buffers : 1

BUFGP : 1

IO Buffers : 3

IBUF : 2

OBUF : 1

Timing Summary:

Speed Grade: -1

Minimum period: 1.045ns (Maximum Frequency:

956.938MHz)

Minimum input arrival time before clock: 1.744ns

Maximum output required time after clock: 3.259ns

Timing Detail:

Timing constraint: Default period analysis for Clock 'clock'

Clock period: 1.045ns (frequency: 956.938MHz)

Total number of paths / destination ports: 3 / 3

Delay: 1.045ns (Levels of Logic = 1)

Source: state_reg_FSM_FFd2 (FF)

Destination: state_reg_FSM_FFd1 (FF)

Source Clock: clock rising

Destination Clock: clock rising

Total 1.045ns (0.565ns logic, 0.480ns route)

(54.1% logic, 45.9% route)

Total memory usage is 4568248 kilobytes

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	4	44800	0%
Number of Slice LUTs	4	44800	0%
Number of fully used LUT-FF pairs	0	8	0%
Number of bonded IOBs	4	640	0%
Number of BUFG/BUFGCTRLs	1	32	3%

Table 1: Device Utilization of Sequence Detector.

IV. Simulation result

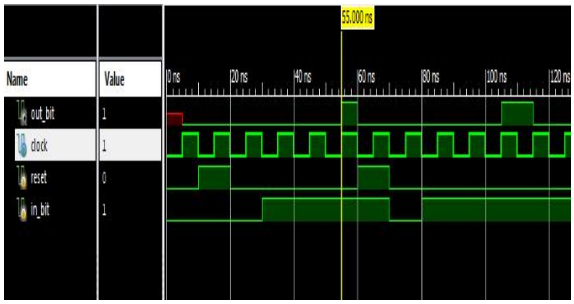


Figure 11: Simulation Result of the Sequence Detector.

The results are obtained by simulating the verilog code in Xilinx ISE 14.7.

V. Conclusion

The proposed design of sequence detector is synthesized and simulated in Xilinx ISE 14.7. The source code is written in Verilog. As we know Delay is the major factor in VLSI design that limits the performance of any circuit. This paper concentrates more on speed by presenting a simple approach to reduce the delay of sequence detector architecture, which helps in increasing the computational level of calculations. This proposed sequence detector has delay 1.045ns. Sequence detector has extensive variety of applications such as design of Ring counter, Serial Adder, Schmitt trigger.

References

1. IEEE Standard 1364-2001, IEEE Standard Description Language based on the Verilog Hardware Description Language, 2001.

2. Mindek, M., “Finite State Automata and Image Recognition” DATESO 2004, pp 132-143 (2004), ISBN: 80-248-0457-3.

3. G. Navarro, R. Baeza-Yates, “Improving an Algorithm for Approximate String Matching.”,Algorithmica, 30(4) 2001.

4. S.V.Rama Subramanian and Kamala Krithivasan,“Finite Automata and Digital Images”, IJPRAI, Vol. 14, No. 4, pp. 501-524, 2000.

5. John C.Martin , ”Introduction to Languages and Theory of Computation”,Third edition, Tata McGraw Hill,2009.