VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama" Belagavi 560018, Karnataka

Mini Project work (18ECMP68) on

"Design and Implementation of Traffic Light Controller for T-Intersection on FPGA"

Submitted in the partial fulfilment of the requirement for the award of degree

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

AGNIVA GHOSH	**1EW20EC003**
GOWTHAM S	**1EW20EC023**
PUSHKAR D	**1EW20EC044**
SANJAY S K	**1EW20EC053**

Under the guidance of

Prof. MANJULA B B

Associate Professor , Dept of ECE, EWIT

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

EAST WEST INSTITUTE OF TECHNOLOGY

BANGALORE – 560091

2022– 2023

EAST WEST INSTITUTE OF TECHNOLOGY

No.63, Off, Magadi Main Road, Vishwananeedam Post, Bangalore-560091

Department of Electronics and Communication Engineering

CERTIFICATE

This is to certify that the project work report entitled **"Design and Implementation of Traffic Light Controller for T-Intersection on FPGA"** is a bonafide work carried out by SANJAY S K bearing USN IEW20EC053 in partial fulfilment of the award of degree of **Bachelor of Engineering in Electronics and Communication** of the Visvesvaraya Technological University, Belagavi during the year 2022-2023. The project work report has been approved as it satisfies the academic requirement in respect of project work prescribed for Bachelor of Engineering degree.

…………………………..	…………………………….	………………………………
Signature of Guide	**Signature of HOD**	**Signature of Principal**
Prof. Manjula B B	Dr . S G Hiremath	Dr . K Channakeshavalu
Assoc Prof., Dept of ECE	HOD, Dept of ECE	Principal,
EWIT, Bengaluru	EWIT, Bengaluru	EWIT, Bengaluru

External Viva

Name of the Examiners	**Signature with date**
1. ________________	________________
2. ________________	________________

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this project work. I would like to take this opportunities thank them all.

With proud gratitude I thank God almighty for all the blessing showered on us and for being able to Complete the project work successfully.

First and foremost, I would like to thank **Dr. K Channakeshavalu**, Principal, EWIT, Bangalore, for his moral support towards completing our project work.

I would like to thank, **Dr. S G Hiremath**, Professor and Head of the Department of ECE, EWIT, Bangalore, for his valuable suggestions and expert advice.

I deeply express my Sincere gratitude to our guide **Prof. Manjula B B**, Associate Professor, Department of ECE, EWIT Bangalore for her valuable guidance throughout the project work and guiding us to organize the report in a systematic manner.

I extended my thanks to the coordinators **Mr. Kotresh H M,** Associate professor, **Mrs. Manasa S**, Assistant professor, for their constant support , guidance and encouragement for project work.

I thank my parents and all the faculty members of department of electronics and communication Engineering for their constant support and encouragement.

Last, but not least ,I would like to thank our peers and friends who provided us with valuable support and suggestions.

DECLARATION

I here by declare that the project report entitled **"DESIGN AND IMPLEMENTATION OF TRAFFIC LIGHT CONTROLLER FOR T-INTERSECTION ON FPGA"** submitted to **EAST WEST INSTITUTE OF TECHNOLOGY** in partial fulfilment of a requirement for the award of degree of bachelor of engineering in Electronics and Communication of Visvesvaraya Technological University, Belagavi during academic year 2022-2023 is a record of bonafide work carried out by **SANJAY S K** bearing **USN 1EW20EC053** under the guidance of Prof. Manjula B B, Associate Professor, Dept of ECE, EWIT.

PLACE: BANGALORE

DATE:

NAME	**USN**	**SIGNATURE**
SANJAY S K	**(1EW20EC053)**	**……………………**

ABSTRACT

Traffic control systems are essential for managing traffic flow and ensuring road safety. In recent years, the implementation of traffic control systems has been heavily reliant on digital electronics and computer-based technology. Verilog is a hardware description language, is widely used for designing digital circuits and systems, including traffic control systems. In this paper, describes an overview of the implementation of a Verilog design for T-Intersection traffic light control. The design includes a traffic light controller that utilizes a state machine to regulate the flow of traffic. The Verilog code is synthesized using a design compiler, and the design is then simulated using a hardware description language simulator. The design flow provides an insight into the implementation of the traffic light controller using Verilog. The implementation shows that Verilog is a powerful and efficient tool for designing traffic control systems. It allows for easy simulation and testing of the design before implementation, enabling designers to optimize the design for performance and reliability. In conclusion, this paper highlights the importance of Verilog in the development of traffic control systems and its potential to contribute to the future of intelligent transportation systems.

KEYWORDS: FSM, Verilog HDL, T-Intersection, Combinational Logic Circuit, PCB, Microprocessor.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Traffic is the movement of vehicles, pedestrians, and other road users on a roadway. Managing traffic is essential to ensure safety and reduce congestion. A traffic controller is a device or system that regulates the flow of traffic at an intersection or along a roadway. Implementing a traffic controller involves designing and installing the system and ensuring that it is programmed and maintained properly. The implementation process involves analysing traffic patterns, selecting appropriate equipment and software, installing and configuring the system, and testing and validating its performance. Effective traffic management systems can create safer and more efficient transportation networks for communities.

Traffic Lights are used to control the vehicular traffic. In the modern era, everyone has different types of vehicles resulting in rise to the numbers of vehicles. That's why traffic lights are mandatory to avoid the traffic jams and accidents. There are three lights in the traffic signal, having different message for the drivers. Red light asks the driver to yield at the intersection, green light gives the driver free license to drive through the intersection whereas the yellow light alerts the driver to wait if the next light is red one or get ready to go / turn the engine ON if the green light is next. Apart from the traffic it is very necessary for the people to cross the roads at particular time interval. And this is only possible by controlling the traffic by giving some kind of signal. Analysing the traffic, estimating the delays to the areas is crucial part. Population can be predicted using GPS trekking and thus we can easily estimate the amount of time to be taken for delay. A perfect aligning of cars, bikes, cycles, trucks with orderly flow by giving right of way, this makes the process very systematic and even in the presence of heavy traffic accident rate goes down which is one of the biggest advantage. Timing and the delays of particular signal plays a vital role because it is very necessary for us to keep information about the amount of traffic which present in the local area. This gives us an idea about timing and delay requirement of every signal in the local area.

As we know the timing depends on traffic volume and its not necessary for us to have same traffic volume at each day so for that we can estimate average volume of traffic around the local area. Average can be made with consideration of 20 days for example we will analyse the traffic of 20 days then we will take an average of it and estimate delays and timings of it. It is very necessary factor for us to have adaptive mechanism. Apart from that we can estimate the traffic volume using GPS, which can give you volume prediction every day. For this we no need to take an average of particular days. GPS give us more correct prediction of volume of traffic than calculating an average of traffic of particular days. Thus, coordinating signal timing minimizes static timing and stopping of vehicles in the traffic to avoid the traffic jam.

1.1 MOTIVATION

Traffic control is a challenging problem in many cities. This is due to the large number of vehicles and the high dynamics of the traffic system. Poor traffic systems are the big reason for accidents, time losses. In this method of approach, it will reduce the waiting time of the vehicles at traffic signals. Verilog designing is hardware descriptive language, the name itself suggests that it deals with hardware designing and simulation. Basically, it becomes very difficult to mount the various electronic components on breadboard or PCB circuit. It also takes too much time for the simulation and sometimes many errors occur because of improper connection of components onto the circuit. And thus, to overcome this factor hardware descriptive language comes into conclusion. We can code the process using Verilog and we can mount it on a circuit or just upload it to the circuit accordingly so that circuit will work as according to the code we have written. HDL language is often used for sequential circuits like shift register, combinational logic circuit like adder, subtractor etc. Basically, it describes the digital systems like microprocessor or a memory. Whatever design that is described in HDL are independent, it has its unique state of work, very much easy to simulate, designing and debugging, and very useful than schematics, especially for large circuits thus, to overcome difficulties or problems to design the circuits manually with breadboard and PCB, use of Verilog designing in this complex world is increasing a way better.

1.2 PROBLEM STATEMENT

The problem that arises in traffic management is the need to efficiently manage the movement of vehicles, pedestrians, and other road users while ensuring their safety and reducing congestion. The implementation of a traffic controller system aims to address this problem by regulating the flow of traffic at intersections and along roadways. However, the challenge is in designing and installing an effective traffic controller system that can adapt to changing traffic conditions and is programmed and maintained properly to ensure optimal performance.

1.3 OBJECTIVES

- The primary objective of a traffic light controller is to regulate the movement of vehicles at intersection or along a roadway and to ensure efficient movement of vehicles and safety of pedestrians.
- The design can be optimised for delay and less power consumption.
- The area and complexity of the traffic light controller system can be reduced by implementing it on the FPGA Kit.

1.4 ADVANTAGES

1 Traffic control signals provide for an orderly movement of traffic.

2 They help in reducing the frequency of accidents of some special nature i.e., of Right-angle accidents.

3 They intercept heavy traffic to allow other traffic to cross the road intersection safely.

4 They provide authority to the drivers to move with confidence.

5 They control the speed of vehicles on main as well as on secondary roads.

6 They direct traffic on different routes without excessive congestion.

7 They provide economy over manual control at the intersection.

1.5 APPLICATIONS:

Automatic traffic light controller have several applications in the field of traffic management and transportation. Here are some examples:

1. Intersection Control: Automatic traffic light controllers are used to regulate the flow of traffic at intersections. They ensure smooth and safe movement of vehicles by assigning right-of-way to different lanes based on traffic conditions.
2. Traffic Optimization: These controllers analyse traffic patterns and adjust signal timings accordingly to optimize traffic flow. They can prioritize high-traffic directions or dynamically adapt to changing traffic conditions, reducing congestion and minimizing delays.
3. Pedestrian Safety: Automatic traffic light controllers incorporate pedestrian detection systems and manage pedestrian signals. They provide safe crossing times, implement special phases for pedestrians, and synchronize pedestrian movements with vehicular traffic to enhance pedestrian safety.
4. Adaptive Signal Control: These controllers utilize real-time data to adapt signal timings based on current traffic conditions. By dynamically adjusting signal cycles, they optimize traffic flow and reduce congestion, especially during peak hours or in response to incidents.

These applications demonstrate the importance of automatic traffic light controllers in improving traffic safety, optimizing traffic flow, and enhancing overall transportation efficiency.

CHAPTER 2

LITERATURE SURVEY

[1] Designing of a Traffic Signalling System at T-Intersection, Ramesh Surisetty. Int. Journal of Engineering Research and Application, ISSN: 2248-9622, Vol. 7, Issue 4, (Part -3) April 2017, pp.82-86: By studying the road traffic of the city we analyze that the major accident cause is collision of vehicles at the intersections. The collision may be rear shunt on approach to junction, right angled collision, principle right turn collision and pedestrian collision. These collisions can be avoided if proper design of signal is done so that the main objective of the dissertation is to provide better and safer movement of traffic through signal design at the intersection. The signal is designed as per IRC guidelines so that the signal can justify the proper movement of the traffic. By providing signals, there will be a reduction in conflicts. And there will be orderly movement of traffic. The signal design procedure involves a few important steps: (1) Three Phase design (2) Determining of amber time and clearance time (3) Determined optimum cycle length (4) Apportioning of green time (5) The presentation estimate of the above design.

[2] TRAFFIC LIGHT CONTROL SYSTEM USING VERILOG DESIGNING. Akshay Bidwai1, Abhiyash Hodge2, Hrishikesh Humnabakar3 1,2,3Electronics and telecommunication, Department, Vishwakarma Institute of Technology, Pune, India, ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-5, ISSUE-7, 2018 : Traffic light control system helps to conduct orderly flow of vehicles. There are many issues of obstacles, high level accidents which occur every day. So, traffic signal controllers prevent such occurrences. Verilog is hardware descriptive language (HDL) which is generally used to model electronic systems. That is, we can design a circuit and the function of circuit can be controlled by Verilog coding. Thus, design and verification can be done using Verilog designing. Similarly, traffic light signal controlling can be done using Verilog (hardware descriptive language). Bit Pattern to be used as RYG, RYG denotes (RED YELLOW GREEN). Transition from red light to green light requires more delay. That means the amount of time for the delay is more. Similarly transition from green light to yellow light requires moderate delay. Transition of yellow light to red light requires very small delay as compared to other delays.

[3] FPGA-Based Advanced Real Traffic Light Controller System Design WM El-Medany, MR Hussain, ISBN:978-1-4244-1347-8, DOI: 10.1109/IDAACS.2007.4488383: FPGA design implementation of a low cost 24-hour advanced traffic light controller system that was built as a term project of a VLSI design subject using VHDL. The system has been successfully tested and implemented in hardware using Xilinx Spartan 3 FPGA. The system has many advantages over the exciting TLC.

[4] International Journal of VLSI System Design and Communication Systems Volume.05, IssueNo.07, July-2017, Pages: 0657-0661: Traffic control is a challenging problem in many cities. This is due to the large number of vehicles and the high dynamics of the traffic system. Poor traffic systems are the big reason for accidents, time losses. In this it will reduce waiting time of the vehicles at traffic signals. Traffic Light Control (TLC) system also based on microcontroller and microprocessor. But the disadvantage of with microcontroller or microprocessor is that it works on fixed time, which is functioning according to the program that does not have the flexibility of modification on real time basis. In traffic light controller, density of traffic is sensed by using IR sensors throughout day and night, and accordingly time is allotted for users to pass. Other advantages of this system are: i) System senses emergency vehicles on the individual road moreover it gives priority to the traffic of that particular road where the emergency vehicles is sensed. ii) Finds out defaulter who crosses the red signal by capturing images using camera. In this, we are using FPGA with traffic sensors to control traffic according requirement means we can change the program if it require and thus reduces the waiting time. The hardware design has been developed using Verilog Hardware Description Language (HDL) programming. The output of system has been tested using Xilinx14.3.

[5] FPGA BASED TRAFFIC LIGHT CONTROLLER Lalita Choudhary, Krishna Balram Parihari, Kamlesh Kumhar, Jitendra Kumar Sao4: The traffic light sequence works on the specific switching of red, green and yellow lights in a particular way with stipulated time from. The normal function of traffic lights requires sophisticated control and coordination to ensure that traffic moves as smoothly and safely as possible and that pedestrians are protected when they cross the roads. This Traffic Light sequence Is generated using a specific switching mechanism which will help to control a traffic light system on road In a specified sequence. This paper focuses on the fact that the traffic lights can be varied in the day and night mode depending on the intensity of the traffic. It plays a vital role in supervising and running the metropolitan traffic and evades the possibilities of any unfortunate mishaps happening in and around the cities. It is a sequential machine to be scrutinized as per the requirements and programmed through a multistep development process. The methods that are used in this projects are proposing the circuit, write a code, simulate, synthesis and implement on the hardware. This paper presents the FPGA implemented low cost advance TLC system using chip scope pro and virtual input output. The TLC implemented is one of the real and complex signalling lights In kingdom of Bahrain, for pedestrian way included four roads and sensors and camera assisted motorway

CHAPTER 3

METHODOLOGY

3.1 Flow Chart

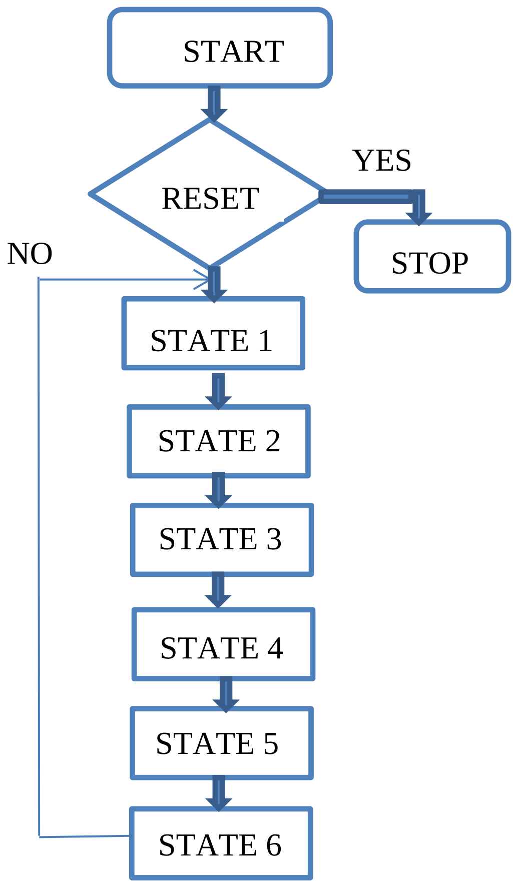


Figure 3.1 Flow Chart of the Design.

The above flow chart represents the working of Traffic Light Controller for T-Intersection. As the system is designed using finite state machine the control flows as shown in above figure 3.1.

The device will be in the initial state after a system reset. The Device adheres to the FSM as specified when reset is not set. It keeps the shift from one state to another going. This XILINX ISE 1.7 design of a traffic-light controller for a T-intersection is created using Verilog code, and an FPGA kit is used to construct it. The introduction of the delay between each state aids in the switching of the traffic light in a signal, allowing for easy vehicle passage without obstruction and delay.

3.2 Requirements

- **Software Requirements**

  Xilinx ISE 14.7 : Synthesis and Simulation.

- **Hardware Requirements**

  Spartan 6 FPPGA Kit : Implementation.

3.3 SYSTEM MODELING

- **Design Specification:**

The design flow depicted in Figure 3,2 is commonly utilized by HDL designers. Initially, specifications are written which describe the functionality, interface, and overall architecture of the digital circuit to be designed in an abstract manner. The architects do not need to consider how to implement this circuit at this stage. Next, a behavioural description is created to assess the design in terms of performance, functionality, Compliance with standards, and other high-level concerns, HDLs are commonly used for writing behavioural descriptions.

- **Design Entry:**

Design entry techniques include schematic-based, Hardware Description Language (HDL) based, and a combination of both, and the selection of a method depends on the design and designer. HDLs provide a level of abstraction that isolates designers from the hardware implementation details. Schematic-based entry, on the other hand, offer more visibility into the hardware and is the preferred choice for those who are hardware oriented. Another method, although rarely used, is state-machines, which is ideal for designers who vie the design as a sequence of states. However, the tools for state machine entry are limited. In the proposed system, HDL-based design entry is utilized.

- **Functional simulation**

Functional simulation is an essential step in the design flow process as it helps to catch any design errors or functional issues early on in the development cycle. By simulating the functionality of the design before actual implementation, designers can save valuable time and resources that would have been spent on costly design iterations. Moreover, functional simulation is an effective method for identifying and debugging complex system-level problems that are difficult to detect through other means.

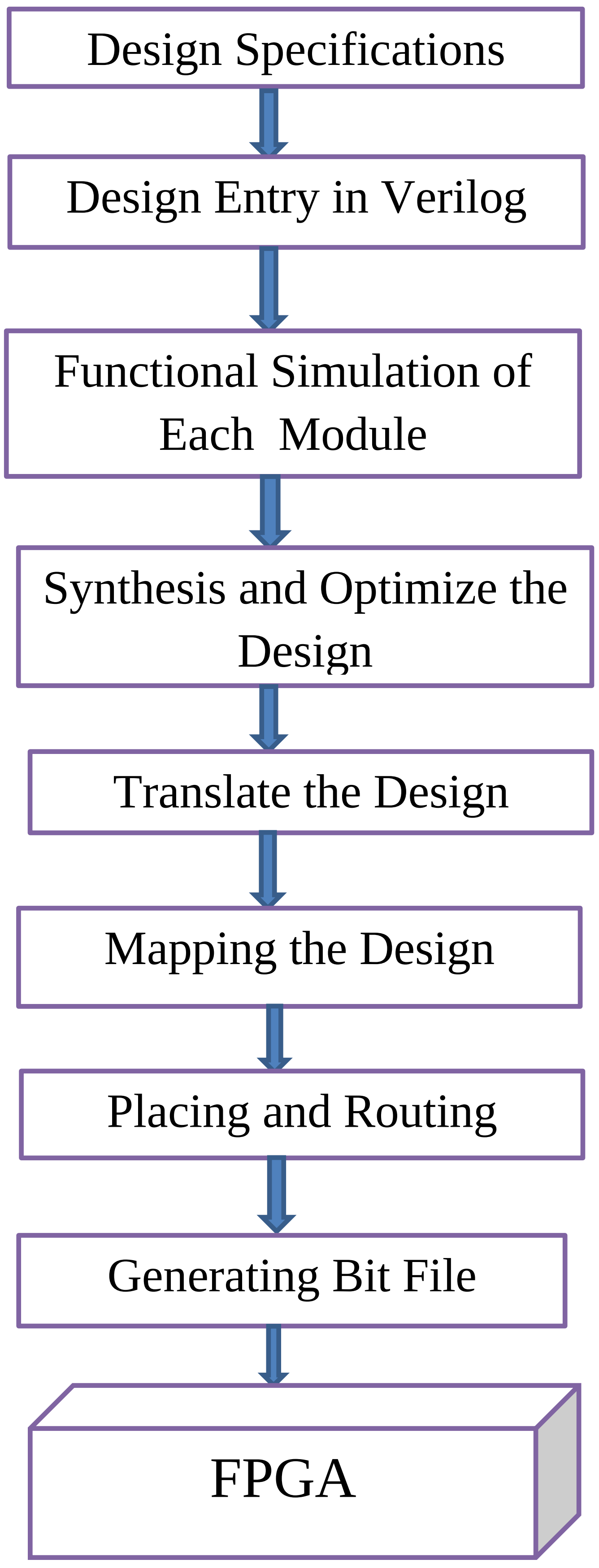


Figure 3.3 : System Modelling

- **Synthesize and optimize the Design**

The synthesis process translates Verilog code into a device net list format consisting of logical elements, such as gates and flip flops, for the complete circuit design. For designs with multiple sub-designs, such as implementing a processor with a CPU and RAM, the synthesis process generates a net list for each design element. During synthesis, the code syntax is checked and the design hierarchy is analysed to optimize the design for the E. selected architecture. The resulting net list or net lists are saved as an NGC file for Xilinx® E Synthesis Technology (XST) or a similar format for other synthesis tools.

- **Translate the Design**

The Translate process is an essential step in the design flow as it generates a complete and optimized design file that is ready for implementation. The NGD file contains a detailed description of the logic elements, their interconnections, and the physical constraints of the design. which enables the implementation tools to generate the bit stream file that configures the FPGA or ASIC, The UCF file, on the other hand, provides additional information that helps to ensure that the design meets the required timing constraints and fits within the available physical resources of the target device. By carefully defining the constraints, designers can ensure that the design will function correctly and meet the desired performance requirements

- **Mapping the design**

The Mapping process divides the circuit with logical elements into sub-blocks that can fit into the FPGA logic blocks. This process utilizes the Combinational Logic Blocks(CLB) and Input Output Blocks (IOB) of the target FPGA to fit the logic defined by the of NGD file, generating an NCD (Native Circuit Description) file that physically represents the design mapped to the FPGA components. The MAP program is used for this purpose. The Mapping process is an important step in the design flow as it ensures that the logic defined in the design file is mapped efficiently to the physical resources of the target FPGA, which can impact the design's performance and resource utilization. By dividing the design into sub-blocks that fit within the FPGA logic blocks, the Mapping process enables the implementation tools to generate an optimal configuration for the FPGA improving the design's performance and reducing implementation time

- **Place and Route**

The Place and Route (PAR) process places the sub-blocks generated by the Mapping process into logic blocks according to the specified constraints and establishes connections between them. For example, if a sub-block is placed in a logic block near an IO pin, it may improve the design's performance by reducing signal propagation time. The Place and Route process takes into account all the constraints and trade-offs to optimize the design for the target FPGA. The PAR tool takes the mapped NCD file as input and produces a completely routed NCD file as output, which includes routing information. The Place and Route process is a critical step in the design flow as it determines the actual physical placement of the logic elements in the target FPGA, which can have have a significant impact on the design's performance and resource utilization. By carefully optimizing the placement and routing of the logic elements, the PAR process ensures that the design meets the specified timing and resource requirements.

- **Generating bit file**

The bit stream file is generated as output from the Place and Route process, which ensures that the design is optimally placed and routed for the target FPGA. The bit stream file contains information such as the logical connections between the various components of the design, the placement of logic elements on the FPGA, and the configuration settings for the various components. Once the bit stream file is loaded onto the FPGA, the device is configured to perform the desired logic operations. The bit stream file can be used to program the FPGA at any time, allowing the design to be modified and updated as necessary. This makes FPGAs a popular choice for applications that require flexibility, speed, and low power consumption.

- **FPGA**

The BITGEN program is a critical part of the FPGA design flow, as it ensures that the design is properly configured for the target device. The BIT file contains all the necessary configuration data, including the logic connections, timing information, and device settings. The cable used to load the BIT file onto the FPGA is typically provided by the FPGA vendor and may vary depending on the specific device and interface requirements. Once the BIT file is loaded onto the FPGA, the device is ready to perform the desired logic operations. This final step of the design flow is critical to ensuring the successful implementation of the FPGA design.

CHAPTER 4

IMPLEMENTATION

The aim of the project is to Design and Implement a Traffic Light Controller system for T-intersection using Verilog on FPGA Kit.

- M1: Represents the vehicles moving from left to right.
- MT: Represents the main turn from the main road.
- M2: Represents the vehicles moving from right to left and from top to bottom.
- S : Represents the side road merging into the main road.

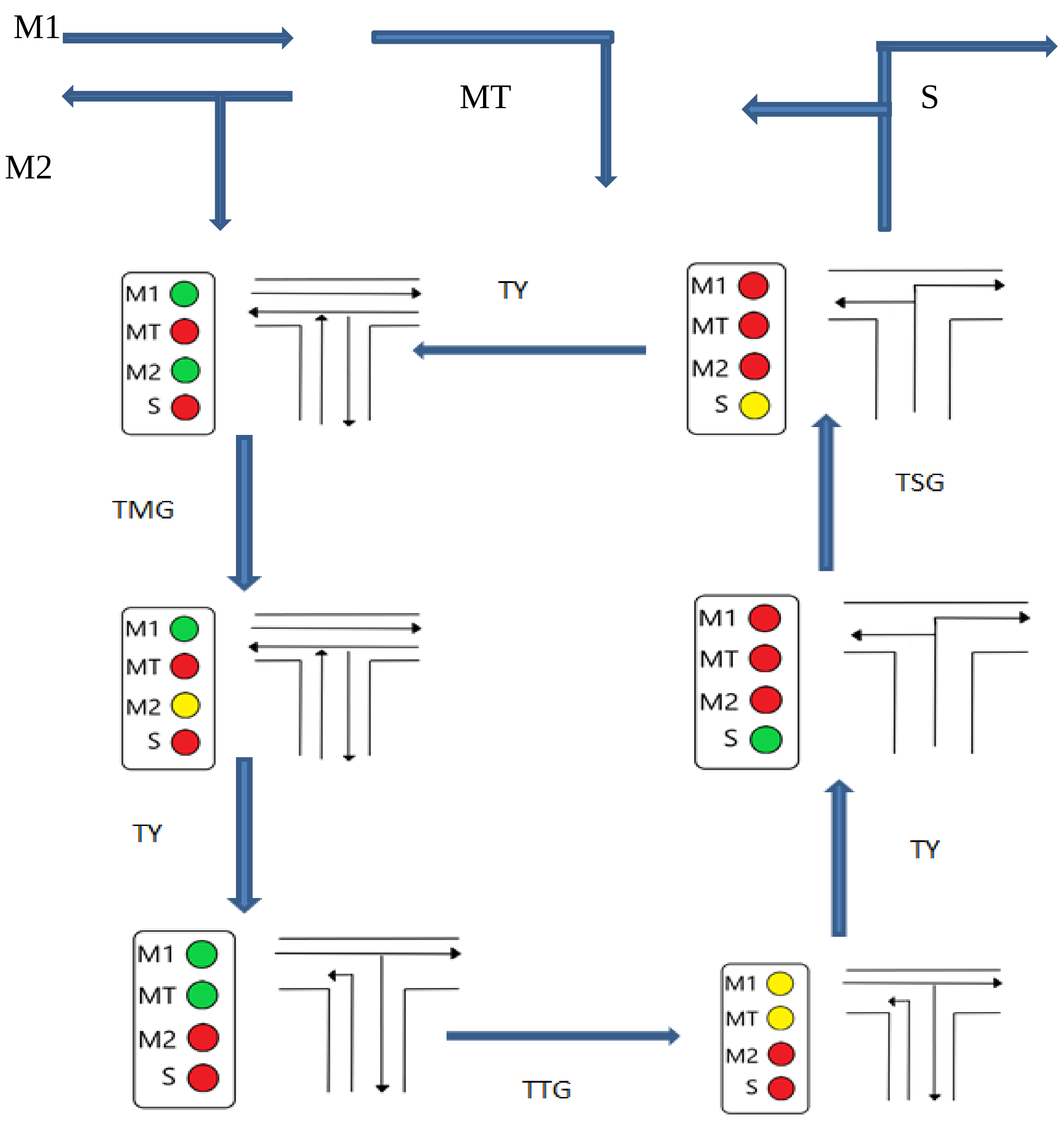


Figure 4.1 : Traffic light Controller switching of Led from one state to another.

Time Delays for changing from one state to another is considered as,

- TMG denotes Transition from one state to another state in main road i.e.. from state 1 to state 2.
- TY denotes Transition from one state to another state as a intermediate state i.e.. from state 2 to state 3.,state 4 to state 5,state 6 to state 1.
- TTG denotes Transition from one state to another state in main road and turning towards side road from main road i.e. from state 3 to state 4.
- TSG denotes Transition from one state to another state in side road towards main road i.e..from state 5 to state 6

Thus the cycle repeats again again.

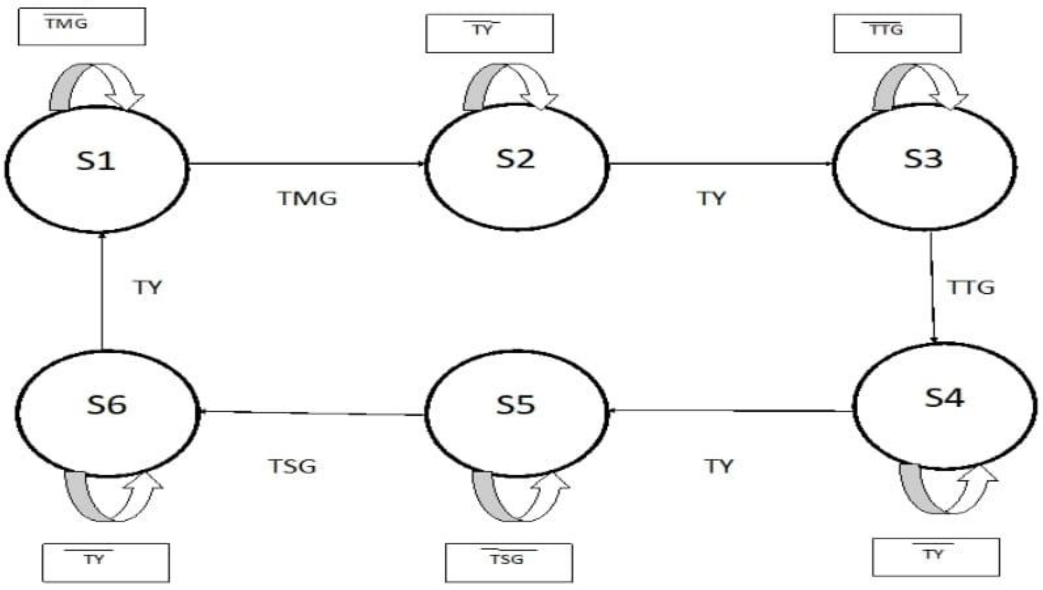


Figure 4.2: Finite State Machine (FSM) Transition from one state to another.

The Time Delays for each state are as follows:

- TMG-30 Seconds.
- TY - 4 Seconds.
- TTG-30 Seconds.
- TSG-20 Seconds.

This delay can be varied for different areas or zones

4.1 FINITE STATE MACHINE OVERVIEW:

Finite State Machines (FSMs) are at the heart of most digital design. The basic idea of an FSM is to store a sequence of different unique states and transition between them depending on the values of the inputs and the current state of the machine. The FSM can be of two types: Moore (where the output of the state machine is purely dependent on the state variables) and Mealy (where the output can depend on the current state variable values AND the input values).A two-step process implements the reactive behaviour: Implementing and optimizing the desired behaviour in a high-level, technology-independent representation of the decision process (called s-graph). Translating the s-graph into portable C code and using any available compiler to implement and optimize it in a specific, microcontroller-dependent instruction set. We deal with speed and size requirements at the s-graph level. The components of the cost function we consider the total number of vertices, which is related to the program code (ROM) size, the maximum distance between Begin and End vertices, which is related to the execution time, and the number of variables, which is proportional to the data size (RAM, including CPU registers).

Speed optimization makes the s-graph as flat as possible without exceeding a bound on the number of vertices. Driving the software synthesis procedure is a cost estimation done on the s-graph. We use appropriately chosen benchmarks to obtain a cost (code/data size and time) estimation for the various software constructs corresponding to s-graph vertices in various combinations. We can perform local optimization by collapsing groups of s-graph vertices with one entry point and two exit points into a single Test vertex, whose label we can obtain by function composition. We can estimate the potential gain of this collapsing as the reduction in code size and execution time due to the use of Boolean operations (used to compute the function) rather than tests and jumps .Finally, we can perform more global optimizations across multiple CFSMs, because composing CFSMs together can reduce the execution time and RAM occupation, thus eliminating the variables used for communication. Real-time operating system. The customized operating system for each microcontroller consists of a scheduler and drivers for the I/O channels. Hence it is extremely small and imposes little overhead, compared with standard operating systems for real-time applications. To correctly implement the CFSM behaviour, the operating system must satisfy the following constraints: Each transition of a task must be performed atomically; that is, the values of the input event buffers for that task must not change once it has been started. The operating system must reset consumed events before invoking a task again. The operating system must transfer all output events from a task by setting the signal or variable denoting the event presence only after it has updated the corresponding value.

CHAPTER 5

SOFTWARE TOOL

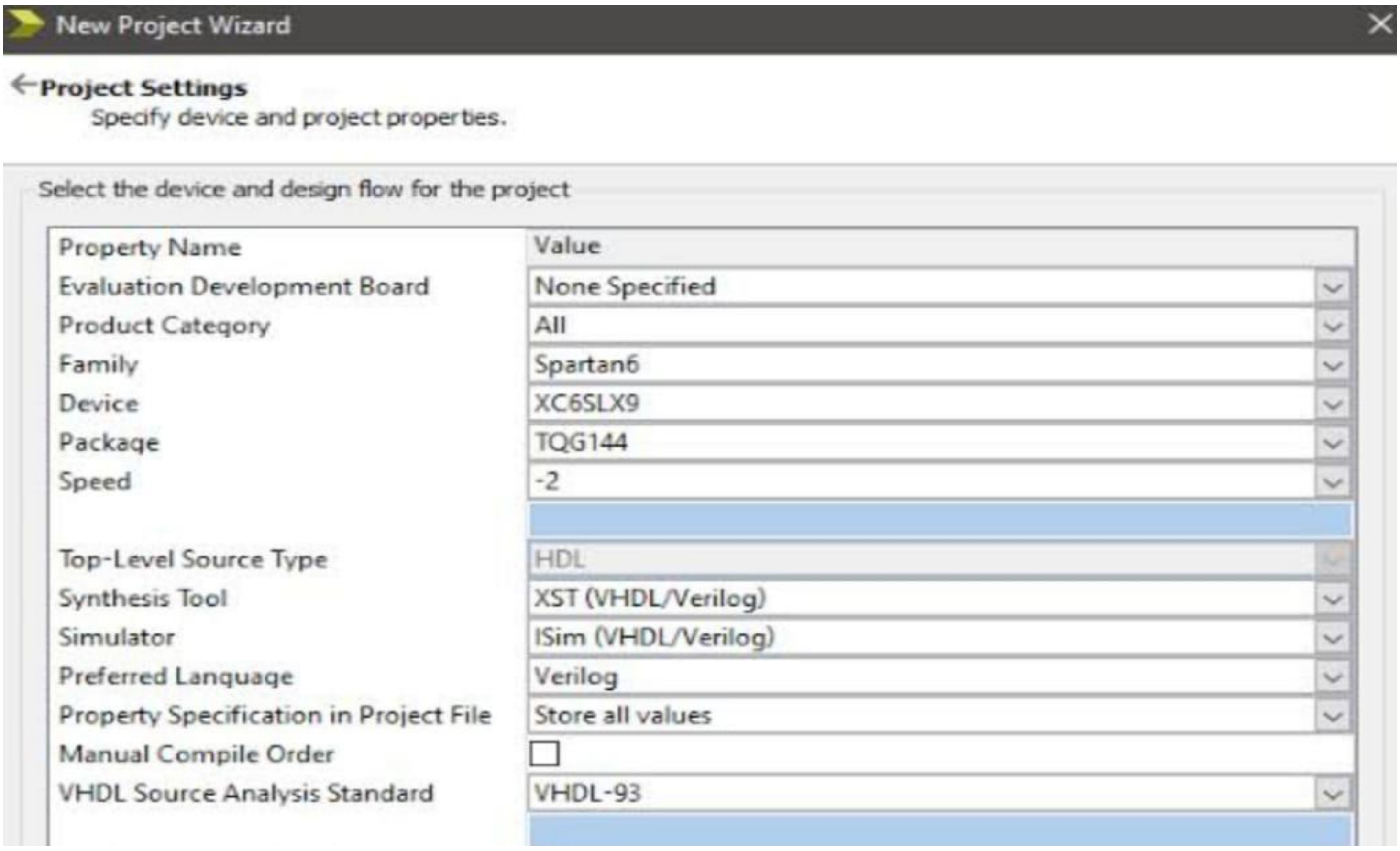
5.1 XILINX ISE 14.7

Figure 5.1 : Logo of XILINX ISE 14.7 DESIGN SUITE

Xilinx ISE (Integrated Synthesis Environment) is a discontinued software tool from Xilinx for synthesis and analysis of HDL designs, which primarily targets development of embedded firmware for Xilinx FPGA and CPLD integrated circuit (IC) product families. It was succeeded by Xilinx Vivado. Xilinx, Inc. released ISE® Design Suite 13.2, providing support for the 28nm 7 series families including the recently arrived Virtex®-7 VX485T device being demonstrated to customers .ISE Design Suite 13.2 enables Advance extensible Interface (AXI) interconnect support in CORE Generator™ system to build higher performance point-to-point architectures. Design teams building their own AXI compliant IP can now run simulations of the AXI interconnect protocol using the optional AXI BEM (bus functional model) verification IP to easily ensure all interface transactions are working properly.

In addition, this latest edition of ISE Design Suite provides an up to 25 percent Performance increase in designs targeting Virtex-7 2000T devices, the industry's largest density FPGAs built using Stacked Silicon Interconnect technology, The latest ISE software release also has enhancements to the Plan Ahead™ design and analysis tool, Providing partial reconfiguration support for Virtex-7 and Kintex™-7 FPGAs, and front to back, integrated project management environment for improved productivity in designs targeting Spartan®-3 FPGAs, Virtex-6 FPGAs.

5.2 DESIGN SPECIFICATION


New Project Wizard
Project Settings
Specify device and project properties.
Select the device and design flow for the project

Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan6
Device	XC6SLX9
Package	TQG144
Speed	-2
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store all values
Manual Compile Order	☐
VHDL Source Analysis Standard	VHDL-93

Figure 5.2 Design Specifications

In this project, we employed the Xilinx 13.2 synthesis tool, which is a software used to transform a high-level hardware description language (HDL) into a lower-level representation that can be implemented in hardware. The synthesis tool is optimized for the Spartan 6 specialization family, which is a type of field-programmable gate array (FPGA) device. Specifically, we used the XC8SLX9 device, which belongs to the Spartan 6 family and has a TQG144 package. The device has a speed grade of -3, which means it is optimized for high-performance applications. Lastly, we used Verilog HDL as the programming language for our design. This language design and is known for its hardware modelling capabilities specifications. By utilizing these tools and specifications, we were able to efficiently and accurately design our hardware system.

5.3 PROGRAMMING LANGUAGE

VERILOG HDL

Verilog is a Hardware Description Language (HDL). It is a language used for describing a digital system such as a network switch, a microprocessors , a memory, or a flipflop. We can describe any digital hardware by using HDL at any level. Designs described in HDL are independent of technology, very easy for designing and debugging are normally more useful than schematics, particularly for large circuits, Verilog standardized as IEEE 1364, is a hardware description language (HDL) used to model electronic systems. It is most commonly used in the design and verification of digital circuits at the register-transfer level of abstraction. It is also used in the verification of analog circuits and mixed-signal circuits, as well as in the design of genetic circuits. In 2009, the Verilog standard (IEEE 1364-2005) was merged into the System Verilog standard, creating IEEE Standard 1800-2009. Since then, Verilog is officially part of the System Verilog language. The current version is IEEE standard 1800-2017.

Hardware description languages such as Verilog are similar w software programming languages because they include ways of describing the propagation time and signal strengths (sensitivity). There are two types of assignment operators; a blocking assignment (=), and a non-blocking (<=) assignment, The non blocking assignment allows designers to describe a state-machine update without needing w declare and use temporary storage variables. Since these concepts are part of Verilog’s language semantics, designers could quickly write descriptions of large circuits in a relatively compact and concise form. At the time of Verilog's introduction (1984), Verilog represented a tremendous productivity improvement for circuit designers who were already using graphical schematic capture software and specially written software programs to document and simulate electronic circuits

The designers of Verilog wanted a language with syntax similar to the C Programming language, which was already widely used in engineering software development. Like C, Verilog is case-sensitive and has a basic pre-processor (though less sophisticated than that of ANSI C/C++). Its control flow keywords (if/else, for, while, case etc…) are equivalent, and its operator precedence is compatible with C. Syntactic differences include: required bit-widths for variable declarations, demarcation of procedural blocks (verilog uses begin/end instead of curly braces {}, and many other minor differences. Verilog requires that variables be given a definite size. In C these sizes are inferred from the ‘type' of the variable (for instance an integer type may be 8 bits).

Verilog supports a design at many different levels of of abstraction. Three of them are very important:

- **Behavioural Level:** This level describes a system by concurrent algorithms (Behavioural). Each algorithm itself is sequential, that means it consists of a set of instructions that are executed one after the other.
- **Register-Transfer Level:** Designs using the Register-Transfer Level specify the characteristics of a circuit by operations and the transfer of data between the registers .
- **Gate Level:** The operations are predefined logic primitives (AND, OR, NOT etc. gates). Using gate level modelling might not be a good idea for any level of logic desıgn.

A Verilog design consists of a hierarchy of modules. Modules encapsulate design hierarchy, and communicate with other modules through set of declared input, output and bidirectional ports. Internally, a module can contain any combination of the following: net/variable declarations (wire, reg, integer, etc..), concurrent and sequential statement blocks, and instances of other modules (sub-hierarchies). Sequential statements are placed inside a begin/end block and executed in sequential order within the block. However, the blocks themselves are executed concurrently, making Verilog a dataflow language

Verilog's concept of 'wire' consists of both signal values (4-state: "1, 0, floating, undefined") and signal strengths (strong, weak, etc..). This system allows abstract modelling of shared signal lines, where multiple sources drive a common net, When a wire has multiple drivers, he wire's (readable) value is resolved by a function of the source drivers and their strengths.

A subset of statements in the Verilog language is synthesizable. Verilog modules that conform to a synthesizable coding style, known as RTL (register-transfer level), can be physically realized by synthesis software. Synthesis software algorithmically transforms the (abstract) Verilog source into a net list, a logically equivalent description consisting only of elementary logic primitives (AND, OR, NOT, flip-flops, etc..) that are available in a specific FPGA or VLSI technology. Further manipulations to the net list. ultimately lead to a circuit fabrication blueprint (such as a photo mask set for an ASIC or a bit stream file for an FPGA).

CHAPTER 6

FPGA

- **FPGA (Field Programmable Gate Array):**

A Field Programmable Gate Array (FPGA) is an integrated circuit designed to be configured by a customer or designer after manufacturing- hence the term field-programmable. The FPGA configuration is generally specified using a hardware description language (HDL). Circuit diagrams were previously used to specify the configuration, but this is increasingly rare due to the advent of electronic design automation tools. FPGAS contain an array of programmable logic blocks, and a hierarchy of reconfigurable interconnects allowing blocks to be wired together. Logic blocks can be configured perform complex combinational functions, or act as simple logic gates like AND and XOR. In most FPGAs, logic blocks also include memory elements, which may be simple flip-flops or more complete blocks of memory. Many FPGAs can to performed implement in computer different logic software. functions, FPGAS allowing have a flexible remarkable reconfigurable role system software development simultaneously with hardware, enable system performance simulations at a very early phase of the development, and allow various system trials and design iterations before finalizing the system architecture.

The great advantage of the FPGA is that the chip is completely programmable and can be re-programmed. In this way it becomes a large logic circuit that can be configured according to a design, but if changes are required it can be re-programmed update, Thus, if circuit card or board is manufactured and contains an FPGA as part of the circuit, this is programmed during the manufacturing process, but can later be re-Programmed to reflect any changes, Thus, it is programmable in the field.

Due to their programmable nature, FPGAs are an ideal fit for many different markets, As the industry leader, Xilinx provides comprehensive solutions consisting of FPGA devices, advanced software, and configurable, ready-to-use IP cores for markets and applications such as, Targeted design platforms for Industrial, Scientific and Medical (ISM) enable higher degrees of flexibility, faster time-to-market, and lower overall non-recurring engineering costs (NRE) for a wide range of applications such as industrial imaging and surveillance, industrial applications and medical imaging equipment.

- **SPARTAN 6 FPGA KIT**

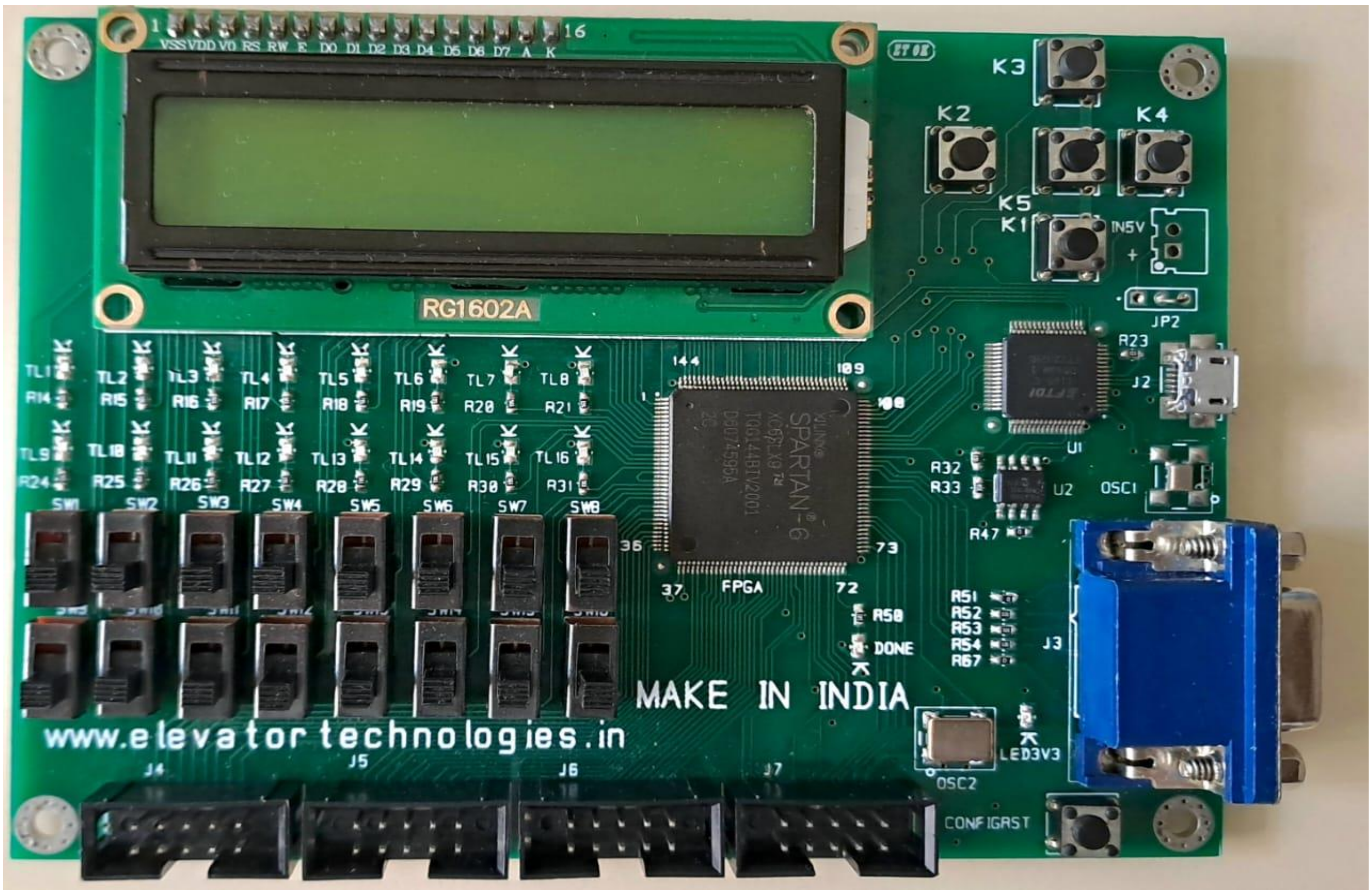


Figure 6.1 SPARTAN 6 FPGA KIT.

The Spartan 6 FPGA can be described as the best choice for balancing cost, space, performance and price. It has over 4 MBs of external non-volatile memory whereas it is also able to house a number of I/O devices and external connectors to various kinds of interfaces, making it the ideal choice for many and most digital applications. There are many reasons why people tend to go for Xilinx's Spartan 6 FPGA family, one of the most important ones being the fact that it offers great system integration capabilities as compared to the costs incurred, making it the perfect fit for high volume applications. On the other end of the spectrum, the simplicity of the circuit and absence of complex external components also makes it a great solution for

There are various factors which make Spartan 6 FPGAs the preferred choice when it comes to FPGA chips in various applications. Following are some of the defining characteristics of the Spartan 6 family.

1. **Power Consumption :**

   With the Xilinx Spartan 6 FPGA line, you get the option to choose from either 1.2V core or a 1.0V core. One of the benefits you get with Spartan 6 FPGAs is the hibernate power down mode that consumes zero power. It also has a suspend mode wherein the state and configuration is maintained, and enhanced control is achieved.

2. **System Integration:**

The Spartan 6 FPGA family has increased I/O connectivity capability thanks to a high in count t0 logic ratio, with over 40 I/O standards to help simplify the system design depending on the device and package size, the number of I/O pins may vary between 102 or 576. Integrated Endpoint Block is also available for PCI Express Designs.

3. **System Performance:**

The system includes up to eight low power serial transceivers operating at 3.2 GB/s as sell as an 800 MB/s DDR3 with an integrated memory controller. The use of adjustable 10 slew rates improves signal integrity whereas the data transfer rate lies at 1080 MB/s per differential V/O. they also have a Clock Management Tile to enhance system performance by promoting low noise and flexible clocking along with Digital Clock Managers and Phase Locked Loops to stop clock skew and promote low jitter clocking.

4. **Vast Applications :**

Apart from the uses mentioned before, Spartan 6 FPGAs are widely used and can be applied in industrial networks, high resolution graphics solutions, as well as in systems governing vehicle networking and connectivity.

5. **Cost Effectiveness:**

As mentioned before, the Spartan 6 FPGA family is the ideal fit if you are looking for ‘high-volume interface that allows for system I/O expansion at a low rate. The Micro Blaze processor soft IP eradicates the need for any external processing components. The FPGAs employ the use of multiple integrated blocks that are efficient in their function.

CHAPTER 7

RESULT

7.1 Simulation Result:

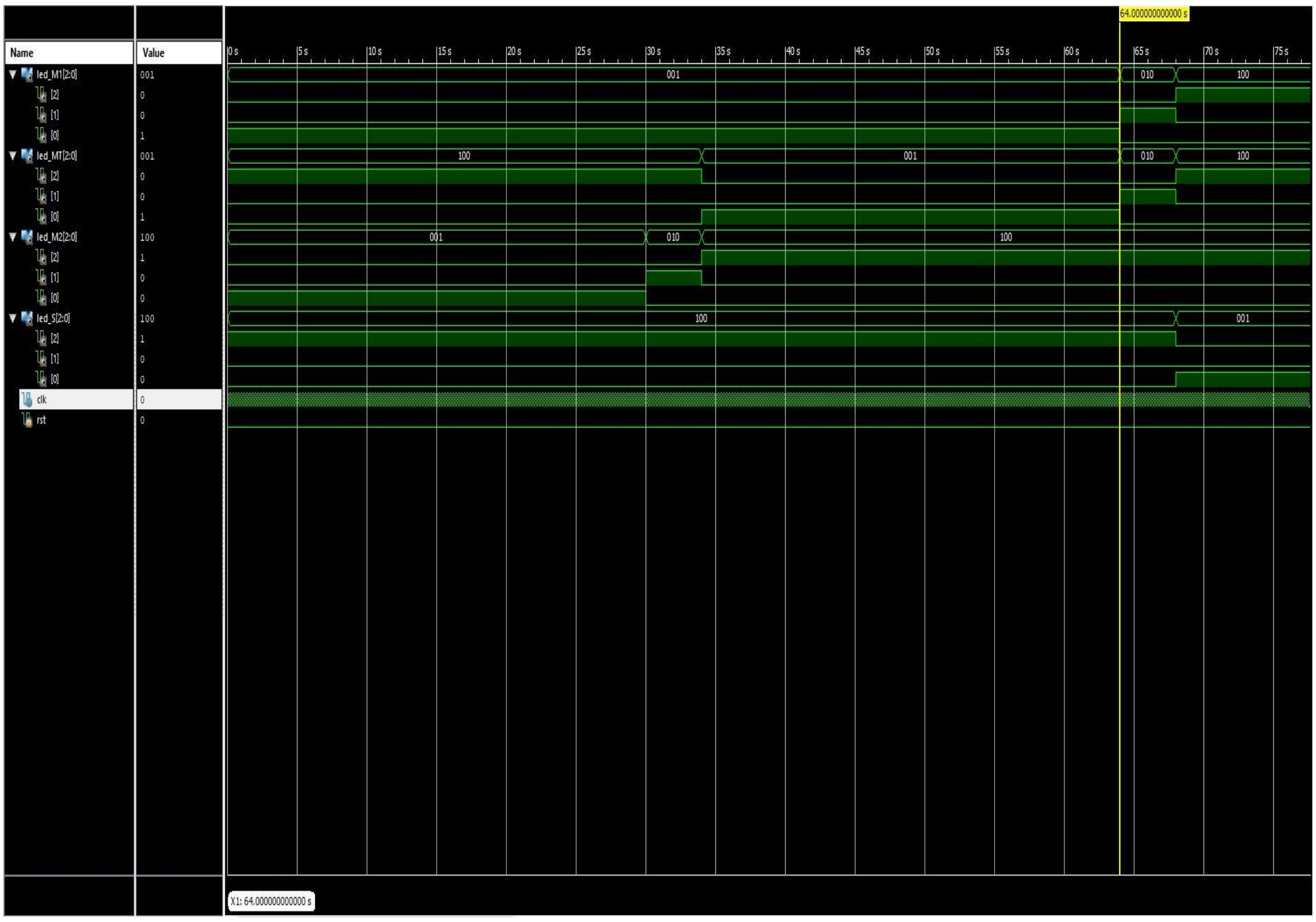


Figure 7.1 SIMULATION WAVEFORMS.

The Traffic Light Controller Design for T-Intersection was designed using Verilog. This design code was then synthesised and simulated to get waveforms as depicted in the Fig. 7.1. In the waveforms it is observed that every states transition as per the designed FSM with accurate time definitions.

7.2 RTL Schematic of Design

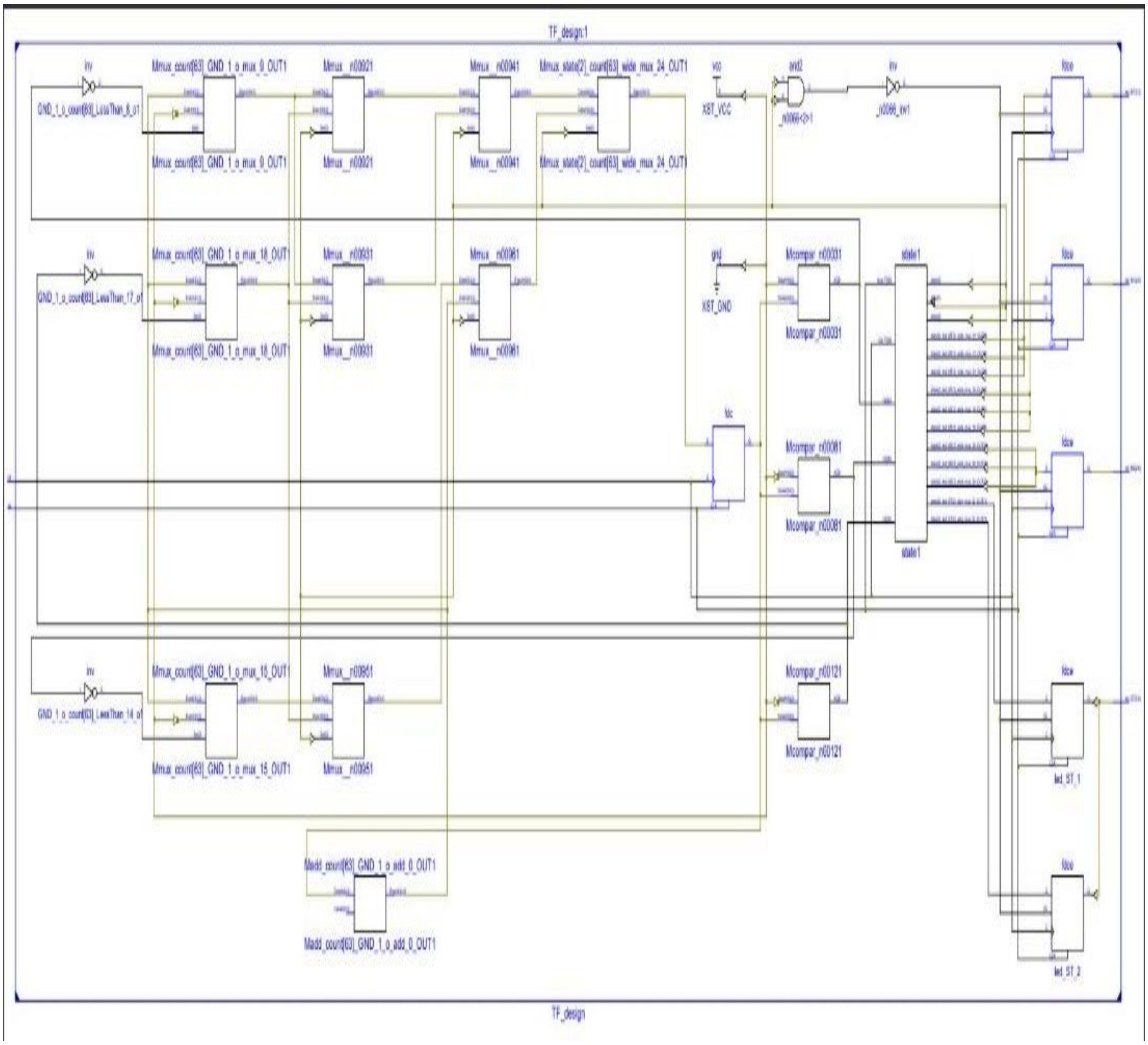

TF_design:1
TF_design
Mmux_count[63]_GND_1_o_mux_9_OUT1
Mmux_n00921
Mmux_n00941
Mmux_state[2]_count[63]_wide_mux_24_OUT1
Mmux_count[63]_GND_1_o_mux_18_OUT1
Mmux_n00931
Mmux_n00961
Mmux_count[63]_GND_1_o_mux_15_OUT1
Mmux_n00951
Madd_count[63]_GND_1_o_add_0_OUT1
Mcompar_n00031
Mcompar_n00081
Mcompar_n00121
XST_VCC
XST_GND
state1
led_ST_1
led_ST_2

Figure 7.2 RTL SCHEMATIC.

The Xilinx ISE 14.7 provides a feature of developing the RTL schematic for mapping and placing of individual devices as per the design. Fig 7.2 shows the RTL schematic developed for the Traffic Light Controller for a T-Intersection.

7.3 Top Module Of The Design

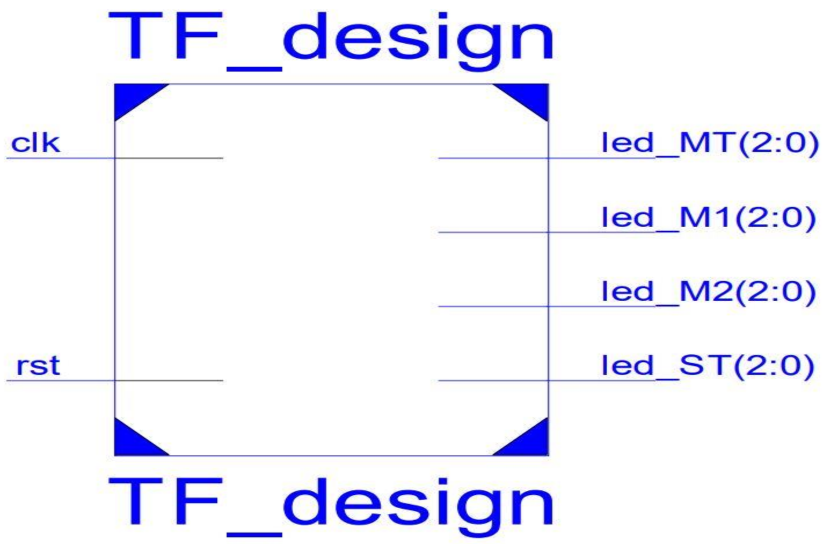
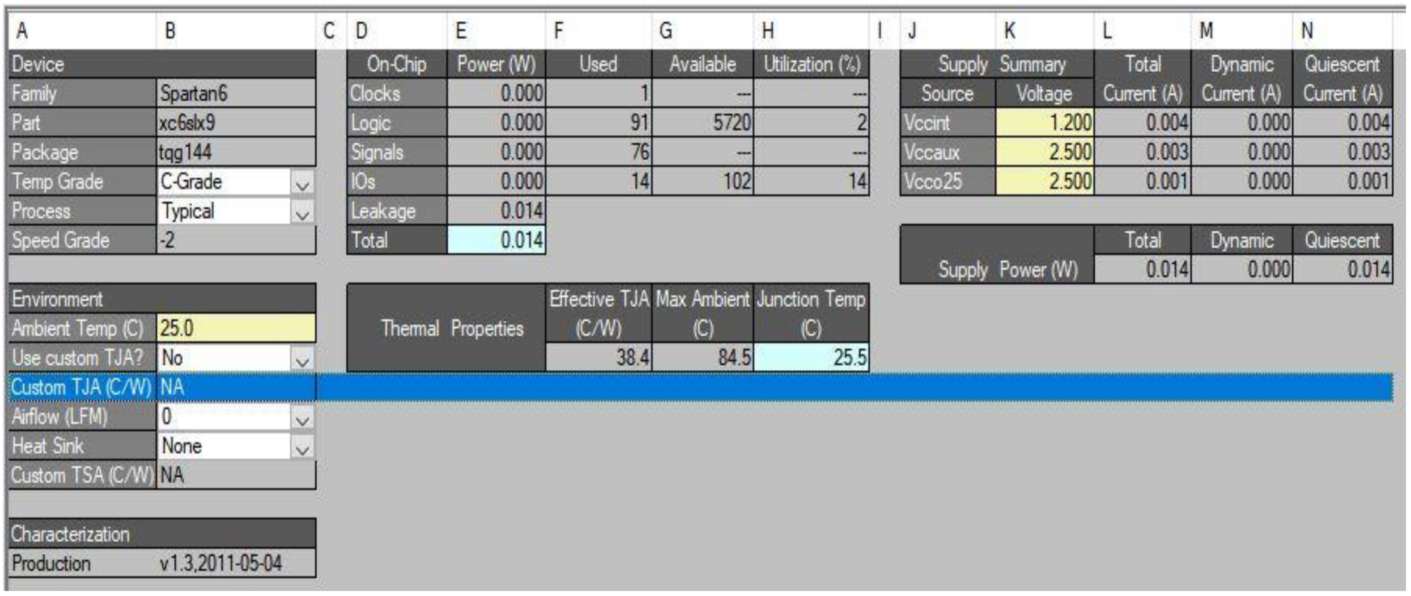


Figure 7.3 Top Module of the Design.

Fig 7.3 shows the Top module of the design for Traffic Light Controller; The top module is a crucial component that acts as the gateway for inputs and outputs as well as managing the internal operations of the processor. A processor is a device that executes a series of instructions to carry out specific tasks.

7.4 Power Analysis



Device	
Family	Spartan6
Part	xc6slx9
Package	tqg144
Temp Grade	C-Grade
Process	Typical
Speed Grade	-2

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	0
Heat Sink	None
Custom TSA (C/W)	NA

Characterization	
Production	v1.3,2011-05-04

On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.000	1	---	---
Logic	0.000	91	5720	2
Signals	0.000	76	---	---
IOs	0.000	14	102	14
Leakage	0.014			
Total	0.014			

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	38.4	84.5	25.5

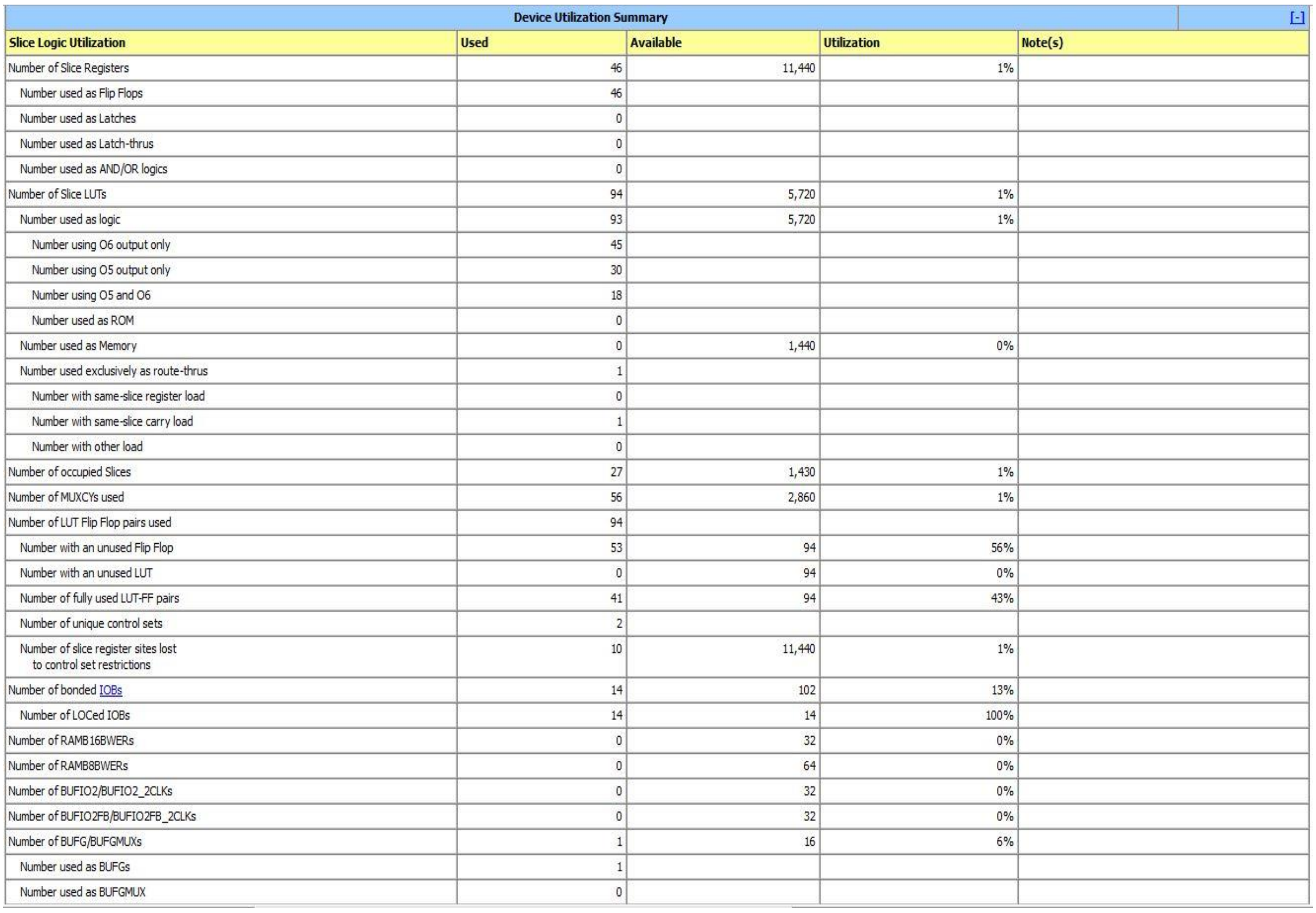
Supply Source	Summary Voltage	Total Current (A)	Dynamic Current (A)	Quiescent Current (A)
Vccint	1.200	0.004	0.000	0.004
Vccaux	2.500	0.003	0.000	0.003
Vcco25	2.500	0.001	0.000	0.001

	Total	Dynamic	Quiescent
Supply Power (W)	0.014	0.000	0.014

Figure 7.4 Power Report

Figure 7.4 gives an estimation of the power used by design.

7.5 Device Area Summary

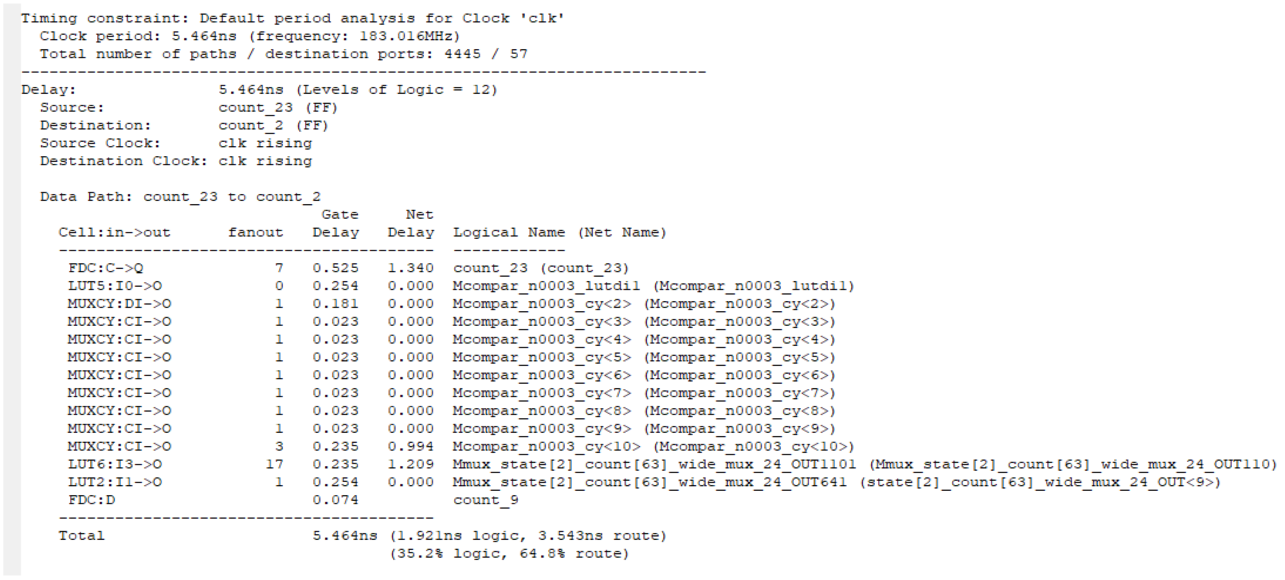


Device Utilization Summary				
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	46	11,440	1%	
Number used as Flip Flops	46			
Number used as Latches	0			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	94	5,720	1%	
Number used as logic	93	5,720	1%	
Number using O6 output only	45			
Number using O5 output only	30			
Number using O5 and O6	18			
Number used as ROM	0			
Number used as Memory	0	1,440	0%	
Number used exclusively as route-thrus	1			
Number with same-slice register load	0			
Number with same-slice carry load	1			
Number with other load	0			
Number of occupied Slices	27	1,430	1%	
Number of MUXCYs used	56	2,860	1%	
Number of LUT Flip Flop pairs used	94			
Number with an unused Flip Flop	53	94	56%	
Number with an unused LUT	0	94	0%	
Number of fully used LUT-FF pairs	41	94	43%	
Number of unique control sets	2			
Number of slice register sites lost to control set restrictions	10	11,440	1%	
Number of bonded IOBs	14	102	13%	
Number of LOCed IOBs	14	14	100%	
Number of RAMB16BWERs	0	32	0%	
Number of RAMB8BWERs	0	64	0%	
Number of BUFIO2/BUFIO2_2CLKs	0	32	0%	
Number of BUFIO2FB/BUFIO2FB_2CLKs	0	32	0%	
Number of BUFG/BUFGMUXs	1	16	6%	
Number used as BUFGs	1			
Number used as BUFGMUX	0			

Figure 7.5 Area Consumption from Design Summary.

Figure 7.5 gives an estimation of the area used by Traffic Light Controller Design for T-Intersection

7.6 Delay Analysis



```
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 5.464ns (frequency: 183.016MHz)
  Total number of paths / destination ports: 4445 / 57
-------------------------------------------------------------------------
Delay:               5.464ns (Levels of Logic = 12)
  Source:            count_23 (FF)
  Destination:       count_2 (FF)
  Source Clock:      clk rising
  Destination Clock: clk rising

  Data Path: count_23 to count_2
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDC:C->Q              7   0.525   1.340  count_23 (count_23)
     LUT5:I0->O            0   0.254   0.000  Mcompar_n0003_lutdil (Mcompar_n0003_lutdil)
     MUXCY:DI->O           1   0.181   0.000  Mcompar_n0003_cy<2> (Mcompar_n0003_cy<2>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<3> (Mcompar_n0003_cy<3>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<4> (Mcompar_n0003_cy<4>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<5> (Mcompar_n0003_cy<5>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<6> (Mcompar_n0003_cy<6>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<7> (Mcompar_n0003_cy<7>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<8> (Mcompar_n0003_cy<8>)
     MUXCY:CI->O           1   0.023   0.000  Mcompar_n0003_cy<9> (Mcompar_n0003_cy<9>)
     MUXCY:CI->O           3   0.235   0.994  Mcompar_n0003_cy<10> (Mcompar_n0003_cy<10>)
     LUT6:I3->O           17   0.235   1.209  Mmux_state[2]_count[63]_wide_mux_24_OUT1101 (Mmux_state[2]_count[63]_wide_mux_24_OUT110)
     LUT2:I1->O            1   0.254   0.000  Mmux_state[2]_count[63]_wide_mux_24_OUT641 (state[2]_count[63]_wide_mux_24_OUT<9>)
     FDC:D                     0.074          count_9
    ----------------------------------------
    Total                      5.464ns (1.921ns logic, 3.543ns route)
                                       (35.2% logic, 64.8% route)
```

Figure 7.6 Delay Analysis of the Design.

Figure 7.6 gives an estimation of Delay analysis for Design of Traffic Light Controller for T-Intersection on FPGA

7.7 FPGA IMPLEMENTATION

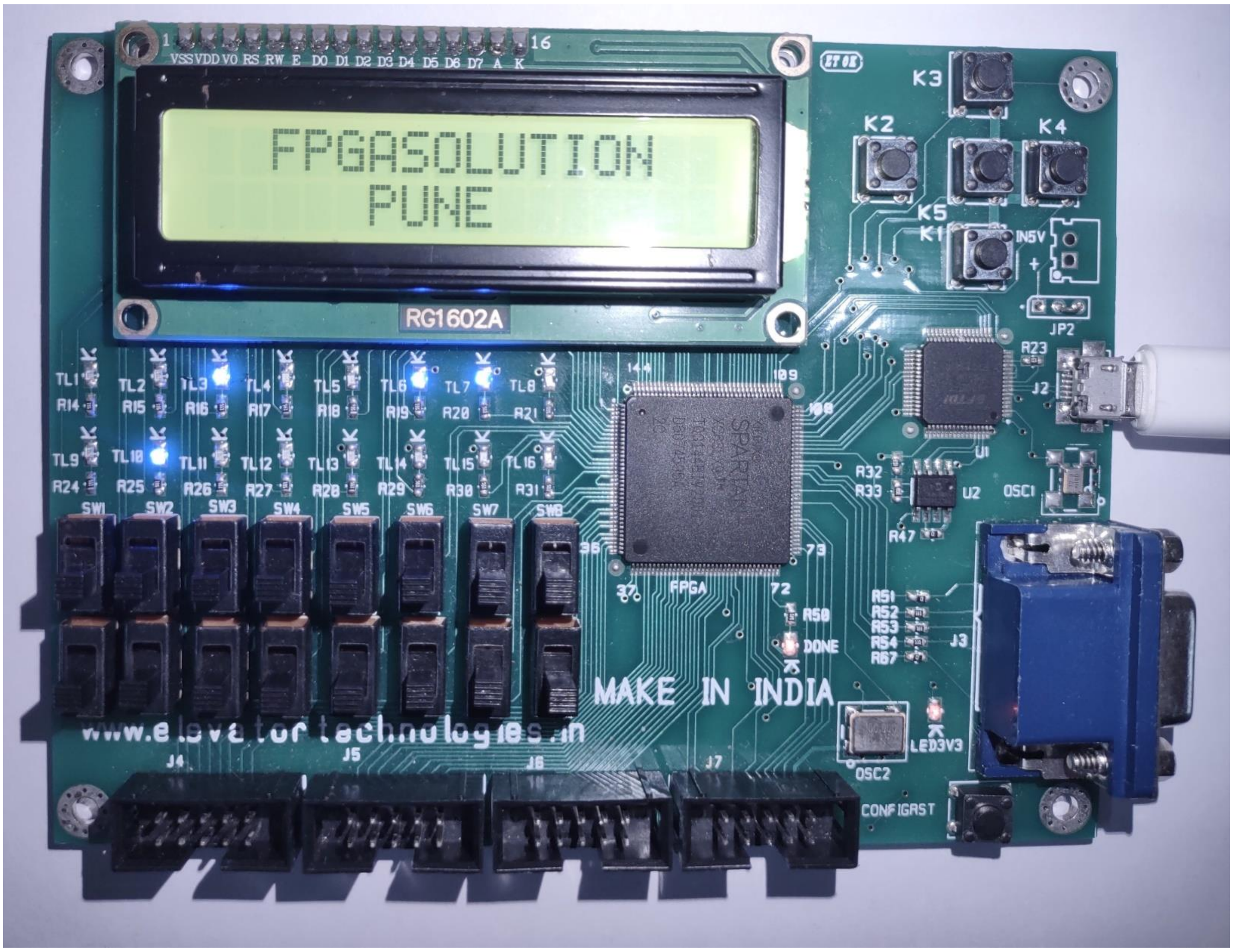


Figure 7.7 Hardware Simulation Result of State.

Figure 7.7 shows the implementation on the Spartan 6 FPGA board, The Traffic Light functionality is confirmed by simulation and FPGA Prototyping. After finishing valid functional simulation and post-route simulation, FPGA Prototyping can be performed. It is observed that the transition of states occur as per time definitions and specific set of outputs are given as desired.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

A Traffic Light controller for T-Intersection using Verilog was efficiently designed and synthesized in Xilinx ISE 14.7 and implemented on SPARTAN-6 FPGA kit with a power consumption of 0.014 Watt, and delay time of 5.464 nano seconds .Thus providing efficient solution for modern traffic control requirements. As the design is integrated in a small chip the size and complexity has reduced. At present the existing traffic light controller is embedded based which the system requires large space or area and more power consumption. By adopting this traffic light controller for T-Intersection we can reduce power and area.

Further the above Design can be added a feature in case of emergency situation. For example when an Ambulance stuck in a traffic the signal can be made traffic free this condition can be taken as a future work on this project.

REFERENCES

[1] Designing of a Traffic Signalling System at T-Intersection, Ramesh Surisetty. Int. Journal of Engineering Research and Application, ISSN: 2248-9622, Vol. 7, Issue 4, (Part -3) April 2017, pp.82- 86:

[2] TRAFFIC LIGHT CONTROL SYSTEM USING VERILOG DESIGNING. Akshay Bidwai1, Abhiyash Hodge2, Hrishikesh Humnabakar3 1,2,3Electronics and telecommunication, Department, Vishwakarma Institute of Technology, Pune, India, ISSN (PRINT): 2393-8374, (ONLINE): 2394- 0697, VOLUME-5, ISSUE-7, 2018 :

[3] FPGA-Based Advanced Real Traffic Light Controller System Design WM El-Medany, MR Hussain, ISBN:978-1-4244-1347-8, DOI: 10.1109/IDAACS.2007.4488383:

[4] International Journal of VLSI System Design and Communication Systems Volume.05, IssueNo.07, July-2017, Pages: 0657-0661:

[5] FPGA BASED TRAFFIC LIGHT CONTROLLER Lalita Choudhary, Krishna Balram Parihari, Kamlesh Kumhar, Jitendra Kumar Sao4:

[6] Quick Reference for Verilog HDL Rajeev Madhavan AMBIT Design Systems, Inc.