Journal of IoT in Social, Mobile, Analytics, and Cloud (ISSN: 2582-1369)
www.irojournals.com/iroismac/

Advanced Traffic Light Controller using FPGA and ARDUINO

Mandeep Singh Narula[1], Amulya Ayush[2], Madhav Kumar[3]

Department of ECE, Jaypee Institute of Information Technology, Noida, India

E-mail: [1]mandeep.narula@jiit.ac.in, [2]amulyaayush2002@gmail.com, [3]ksharma82794@gmail.com

Abstract

This study presents a novel approach to develop a traffic light controller using FPGA and Arduino platforms. The objective is to design a robust, adaptable, and efficient traffic management system. Methodology involves hardware integration of FPGA for real-time processing and Arduino for interfacing with sensors and actuators. The proposed system utilizes FPGA's parallel processing capabilities for rapid decision-making and Arduino's flexibility for sensor data acquisition and actuator control. Performance evaluation includes simulation and real-world testing to assess reliability, response time, and scalability. Results demonstrate the effectiveness of the proposed controller in optimizing traffic flow and improving intersection safety.

Keywords: FPGA, Vivado, Arduino, Traffic Light

1. Introduction

Traffic management is a pressing concern in densely populated areas, and a reliable, efficient, and adaptable traffic light system is essential for enhancing road safety and optimizing traffic flow [1-3]. By implementing this work, the research aims to leverage the capabilities of FPGA technology to design a sophisticated controller that can respond dynamically to varying traffic conditions. Reconfigurable semiconductor devices known as Field-Programmable Gate Arrays (FPGAs) provide enormous flexibility in digital circuit design. FPGAs allow for the dynamic modification of signal timings and real-time processing of sensor data in traffic light control. Their capacity for parallel processing improves efficiency and responsiveness, which is essential for maximizing traffic flow. Additionally, FPGAs make

DOI: https://doi.org/10.36548/jismac.2024.2.006
Received: 02.04.2024, received in revised form: 24.04.2024, accepted: 18.05.2024, published: 30.05.2024

it easier to integrate several communication protocols, enabling smooth connection with different traffic management systems. FPGAs are essential to current traffic light controllers because they can apply sophisticated algorithms and adjust to changing traffic patterns, making transportation networks safer and more effective [4-5].

The main advantage of this traffic light controller is its capability to prioritize emergency vehicles such as ambulances and fire services by dynamically adjusting traffic light signals. This feature ensures swift passage for emergency vehicles, minimizing response times during critical situations and potentially saving lives. However, a limitation lies in the frequency and range of this adjustment mechanism. While the controller can effectively manage traffic flow to accommodate emergency vehicles within a certain proximity to the intersection, its effectiveness may diminish at greater distances or in areas with high traffic density. Therefore, the system's ability to prioritize emergency vehicles may be limited to specific geographic areas or scenarios where traffic conditions allow for timely adjustments. Despite this constraint, the controller's capacity to enhance emergency response efficiency remains a valuable asset in urban traffic management systems, prioritizing public safety and well-being.

State Table and state diagram is shown in Table 1 and Figure. 1 respectively. The problem formulation for this work on a traffic light controller using FPGA on Vivado involves designing and implementing a system that simulates and controls traffic lights using LEDs. This work aims to address the challenges of efficient traffic management and signalization.

Table 1. State Table

PRESENT STATE ABC	**INPUT**	NEXT **STATE A'B'C'**	**M1 RYG**	**M2 RY G**	**T RY G**	**S RY G**
001 001	TMG' TMG	001 010	001	001	100	100
010	TY' TY	010 011	001	010	100	100
011	TTG' TTG	011 100	001	100	001	100
100	TY' TY	100 101	010	100	010	100
101	TSG'	101	100	100	100	001

	TSG	110				
110	TY' TY	110 001	100	100	100	010
111	-	000	000	000	000	000

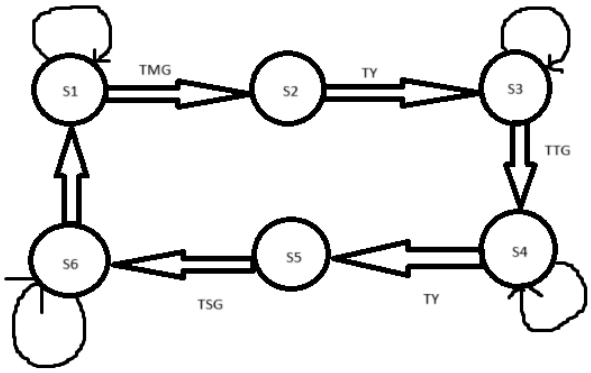

Figure 1. Sate Diagram

The primary objectives include:

1. **Traffic Simulation:** Develop a realistic traffic simulation model that emulates various traffic scenarios, considering different road intersections and traffic flow patterns.

2. **LED Control:** Implement a robust control system using FPGA and Vivado to manage the LEDs representing traffic lights. This involves designing logic circuits to control the sequencing of lights based on the simulated traffic conditions.

3. **Timing and Synchronization:** Optimize the timing of traffic light transitions to ensure smooth traffic flow and minimize congestion. Synchronize traffic lights at different intersections to enhance overall traffic coordination.

4. **User Interface:** Create a user-friendly interface, possibly through buttons or switches, to allow manual control or override of the traffic lights. This can be useful for testing and scenarios that require manual intervention.

5. **Fault Tolerance:** Incorporate mechanisms to handle potential faults or malfunctions in the system, such as sensor errors or power fluctuations. Ensure that the system can recover gracefully from unexpected situations.

6. **Power Efficiency:** Consider power consumption efficiency in the FPGA design to promote sustainability and reduce energy costs over prolonged operation periods.
7. **Real-time Monitoring:** Implement a real-time monitoring system to gather data on traffic conditions, allowing for analysis and optimization of traffic light timings based on actual usage patterns.
8. **Documentation and Reporting:** Provide comprehensive documentation, including design specifications, implementation details, and user manuals. Include a reporting mechanism to analyze the system's performance and effectiveness.

By addressing these aspects, this work aims to contribute to the field of intelligent traffic management systems by creating a functional and efficient traffic light controller using FPGA technology and Vivado design tools.

2. Description of Hardware

Schematic diagram is shown in Figure. 2. The main components are:

2.1 Arduino

Arduino and ESP32 are two outstanding platforms in the field of embedded systems and DIY electronics that are revolutionizing the development of projects and prototypes for hobbyists, and professionals. The platform offers rich hardware and software ecosystems that enable users to create a wide range of innovative applications, from simple flashing LED projects to complex Internet of Things (IoT) devices. The versatility and accessibility of Arduino make it an ideal platform for prototyping and implementing various projects in various fields. Some of the most common applications of Arduino are Home Automation, IoT Devices, Educational Tools, Robotics, Wearable Technology etc.

Arduino is a physical processing tool based on a microcontroller board and integrated development environment board layout. Arduino open-source hardware technology is one of the best platforms available and are used to build a home automation system. This technology can be used to control applications such as lights and sensors by reading the inputs and turning

on the motor, LED etc. This can be achieved by sending instructions to the microcontroller[6-9].

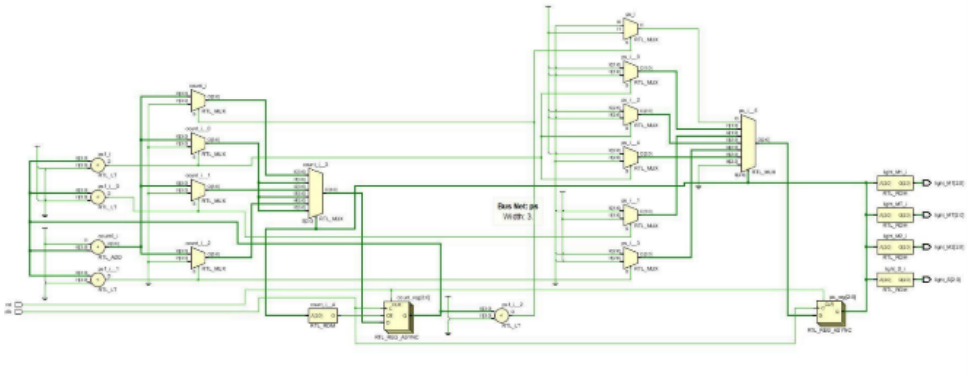

Figure 2. Schematic Diagram

2.2 ESP32

Enabling connectivity and versatility developed by Espressif Systems, the ESP32 represents a major advance in the field of embedded systems, especially in terms of wireless connectivity and computing power. Launched in 2016 as the successor to the ESP8266, the ESP32 combines a dual- core processor, Wi-Fi and Bluetooth connectivity, and a rich set of peripherals on a single chip,making it an attractive platform for IoT, networking and multimedia applications. .

ESP32 main features are:

1. **Dual-core Processor:** The ESP32 has a dual-core Xtensa LX6 microprocessor that enables parallel execution of tasks and efficient use of resources. This dual-core architecture improves performanceand responsiveness, especially in multitasking scenarios.

2. **Wireless:** ESP32 integrates Wi-Fi and Bluetooth modules, enabling seamless communication and networking capabilities. It supports multiple Wi-Fi protocols (802.11 b/g/n) and Bluetooth standards (BluetoothLowEnergy, BluetoothClassic), making it suitable for IoT, home automation and wearable technology applications.

3. **Abundant Peripherals:** The ESP32 offers a wide range of built-in peripherals, including GPIO pins, SPI, I2C, UART interfaces, analog-to-digital converters (ADC), digital-to- digital converters. analog converters (DAC) and pulse width modulation

(PWM) controllers. These peripherals facilitate interaction with external sensors, actuators, displays and communication modules, expanding the versatility and applicability of the platform.

4. **Ultra-Low Power Consumption:** The ESP32 includes power-saving features such as multiple sleep modes, dynamic voltage scaling and clock gate to enable ultra-low-power battery-powered and energy-efficient operation of IoT devices. This efficiency makes the ESP32 suitable for applications that require longer battery life and low power consumption.

5. **Rich Development Environment:** Espressif provides comprehensive development tools and resources to program and debug ESP32 based projects. ESP-IDF (Espressif IoT Development Framework) provides a robust SDK (Software Development Kit) that includes libraries, examples, and documentation for building ESP32 firmware and applications.

Some notable applications of ESP32 are IoT Gateways, Smart home systems, Industrial Automation, Sensor Sensor Networks, Audio Broadcasting and Multimedia etc.

2.3 Vivado

Vivado is a powerful and comprehensive development environment created by Xilinx for designing and programming FPGAs (Field-Programmable Gate Arrays) and SoCs (System on Chips). With its wide range of features and capabilities, Vivado has become an essential tool for engineers and developers working in the field of digital design and programmable logic. Vivado provides a user-friendly graphical interface that facilitates the entire design process, from creating a new project to generating programming files for the target FPGA or SoC. One of the standout features of Vivado is its support for both hardware description languages (HDLs) like Verilog and VHDL, as well as high-level synthesis languages like C and C++. This flexibility allows designers to choose the language that best suits their expertise and work requirements.

2.4 ZEDBoard

The ZedBoard is a development board designed to facilitate the prototyping and development of digital systems based on Xilinx Zynq-7000 All Programmable SoC (System

on Chip). The term FPGA stands for Field Programmable Gate Array, a type of integrated circuit that can be configured after manufacturing. The ZedBoard combines the flexibility of an FPGA with the processing power of an ARM Cortex-A9 dual-core processor, offering a versatile platform for a wide range of applications [10-12]. At the heart of the ZedBoard is the Xilinx Zynq-7000 SoC, which integrates programmable logic (PL) and processing system (PS) components on a single chip. The programmable logic consists of the FPGA fabric, allowing users to implement custom digital circuits tailored to their specific requirements. The processing system includes ARM Cortex-A9 cores, providing the ability to run embedded software.

2.5 LED's

Light-emitting diodes (LEDs) have become a ubiquitous and transformative technology in the field of illumination, revolutionizing the way we light up our world. These small, energy-efficient devices have replaced traditional incandescent and fluorescent lights in a wide range of applications, from household lighting to automotive headlights and electronic displays. The rise of LEDs can be attributed to their numerous advantages, including energy efficiency, longevity, and versatility [13-15].

2.6 LoRa

LoRa (Long Range) is a wireless communication technology that provides long-range, low-power connections for Internet of Things (IoT) and machine-to-machine (M2M) applications. Developed by Semtech Corporation, LoRa is based on beep spread spectrum modulation, which enables robust communication over long distances while consuming little power. LoRa operates in unlicensed industrial, scientific and medical (ISM) domains, allowing use in many environments without regulatory approval.

A key feature of LoRa technology is its ability to reach communication distances of up to several kilometres in urban environments and tens of kilometres in rural areas, making it suitable for applications such as smart cities, agriculture, industrial automation and asset tracking. LoRa uses spread spectrum modulation techniques to transmit data over a wide frequency range, allowing multiple devices to communicate simultaneously without interference. LoRa devices typically operate at lower GHz frequencies such as 433 MHz, 868 MHz, and 915 MHz, which offer better transmission characteristics compared to higher

frequency bands. LoRa networks consist of three main components: terminals, gateways and web servers. End devices such as sensors or actuators send data to gateways using LoRa modulation.

LoRa WAN (Long Range Wide Area Network) is a popular protocol used to implement LoRa networks, which provides standardized communication protocols and network architecture interoperability between devices and network infrastructure from different manufacturers. LoRaWAN supports a variety of network topologies, including star, mesh, and hybrid configurations, enabling flexible deployment in a variety of IoT applications.

One of the main advantages of LoRa technology is its long-range capability, which enables connectivity over large distances with minimal infrastructure requirements. This makes LoRa ideal for applications such as precision agriculture, where sensors installed in large fields or remote areas can send environmental data back to a central monitoring system without the need for extensive cabling or infrastructure.

Despite its many advantages, LoRa technology also has some limitations and challenges that must be considered. One challenge is limited data throughput, especially in applications that require high data rates or real-time communications. LoRa's focus on long-range communications and low-power operation costs data transmission, making it less suitable for applications such as high-definition video streaming or real-time voice communications. In addition, LoRa networks can experience latency due to the asynchronous nature of communications and the need for devices to compete for transmission time, which can affect the responsiveness of certain IoT applications.

3. Methodology

Traffic management is an important part of urban infrastructure and effective control systems are essential to maintain smooth traffic and ensure safety. This work proposes to develop a traffic light controller using STM32 microcontroller and Arduino with a special function to prioritize emergency vehicles such as ambulances. In addition, the system includes LoRa communication and ESP32-based LED modules for increased functionality and flexibility.

1. **System Architecture:** The system architecture consists of three main components: STM32 microcontroller, Arduino board and ESP32 modules. The STM32 acts as the main controller to control the traffic lights, while the Arduino board facilitates communication with sensors and peripherals.ESP32 modules with LoRa communication capabilities enable seamless coordination between traffic lights and emergency vehicles.

2. **Integration of Sensors:** Several sensors are integrated into the system to detect traffic flows and identify emergency vehicles. Inductive loop sensors built into the road detect the presence of vehicles at intersections. In addition, GPS modules installed in emergency vehicles provide real-time location information, which allows the system to anticipate their approach.

3. **Semaphore Control Algorithm:** The core of the system is the semaphore control algorithm implemented on the STM32 microcontroller. The algorithm dynamically adjusts traffic light timing based on real-time traffic conditions and prioritizes emergency vehicles upon detection. It uses a state machine-based approach to manage the sequential change of traffic lights at intersections, ensuring smooth and efficient traffic flow.

4. **Detection and Prioritization of Emergency Vehicles:** When the emergency vehicle approaches the intersection, the system receives its location information through the LoRa network. Once detected, the algorithm dynamically changes the sequence of traffic lights to create a green lane for the approaching vehicle, minimizing delays and ensuring that it passes quickly. This prioritization mechanism is critical to speeding up emergency response times and saving lives.

5. **LoRa Communication and ESP32 Integration:** The LoRa communication protocol enables long- range low-power wireless communication between the traffic light controller and ESP32 modules installed in emergency vehicles. Upon receiving a signal indicating the presence of an emergency vehicle, the ESP32 modules activate high-intensity LED displays that alert the surrounding traffic and ensure the safe passage of the vehicle.

6. **Simulation and Testing:** The system is extensively simulated and tested to confirm its performance in various traffic and emergency situations. Simulation software such as Proteus and real-world testing in controlled environments are used to assess system responsiveness, reliability and performance.

7. **Implementation and Integration:** Once the functionality of the system is verified, it can be implemented at intersections in urban areas. The integration with the existing traffic infrastructure is seamless, and the introduction of new control and communication modules requires minimal changes. Finally, the proposed traffic light controller combines STM32 microcontroller, Arduino, LoRa communication and ESP32 modules to create an efficient and adaptive traffic control system. By prioritizing emergency vehicles and using advanced communication technologies, the system improves traffic flow, shortens response times and improves overall road safety.

Flowchart is shown in Figure 3. Designing a traffic light controller using an FPGA (Field Programmable Gate Array) in Vivado involves several key steps and methodologies. The traffic light controller aims to simulate and control the behavior of traffic lights at an intersection using an FPGA. This will employ the Vivado software for FPGA programming and utilize LEDs to represent the traffic signals, identify specifications such as the number of traffic lights, intersection type (four-way, three-way, etc.), and timing requirements. The next step is to create a high-level block diagram outlining the major components of the traffic light controller system, specify the FPGA model and its characteristics, define the interface for input sensors and output LEDs. FPGA that meets the requirements in terms of logic elements, speed, and I/O capabilities should be chosen considering the factors such as cost and power consumption.

Hardware Implementation: Vivado's schematic editor is used to create a visual representation of the digital circuit, connect input sensors to appropriate FPGA pins, design the logic for the traffic light control, and implement the traffic light controller logic using VHDL (VHSIC

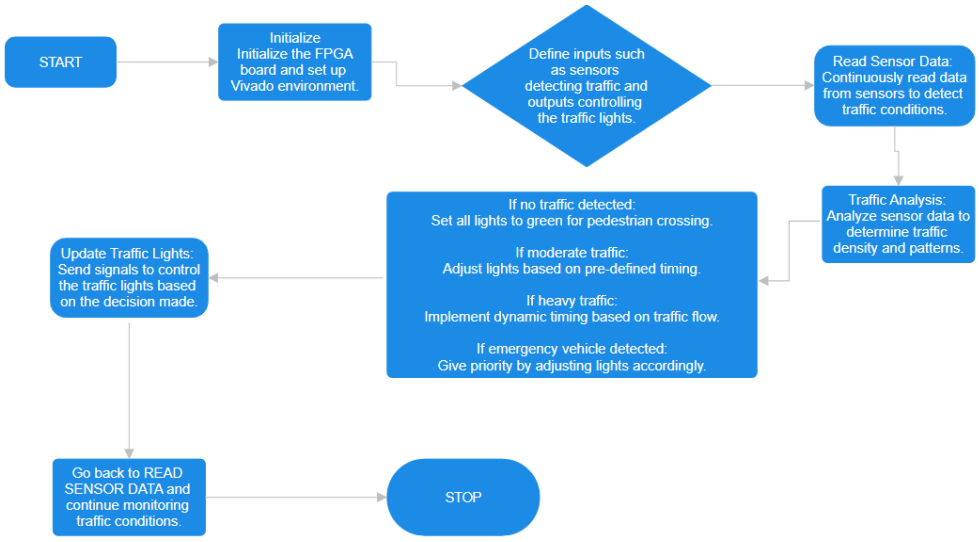

Figure 3. Flowchart

Hardware Description Language). The next step defines modules for traffic lights, timers, and input sensors. A comprehensive test bench is created to verify the functionality of individual modules and the overall traffic light controller. The VHDL code is validated under traffic scenarios and sensor inputs. The physical pins are assigned on the FPGA to corresponding signals in the VHDL code to ensure proper I/O standard compatibility.

In the traffic light controller project, communication between the FPGA and other components is typically established through various interfaces such as UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface), or I2C (Inter-Integrated Circuit). These interfaces enable data exchange between the FPGA and external devices such as sensors, actuators, and microcontrollers like Arduino. The FPGA sends control signals or receives sensor data through these interfaces, allowing for coordinated traffic light control based on real-time inputs. Additionally, protocols like RS-232 or RS-485 may be used for longer-distance communication. The FPGA processes incoming data, executes traffic light algorithms, and generates appropriate signals to control the traffic lights. Careful consideration is given to timing requirements, data rates, and protocol compatibility during interface design to ensure reliable communication. Through efficient communication, the FPGA orchestrates traffic flow while maintaining synchronization and responsiveness, crucial for effective traffic management in diverse scenarios.

4. Results

Simulation results, I/O port assignments, synthesis schematic, timing report, utilization report and power report is shown in Figure 4 to Figure 9 respectively.

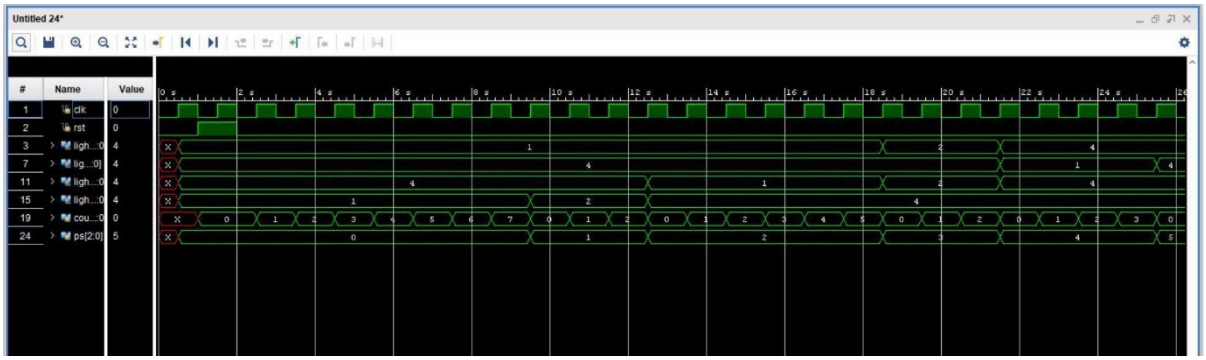

Figure 4. Simulated Waveform

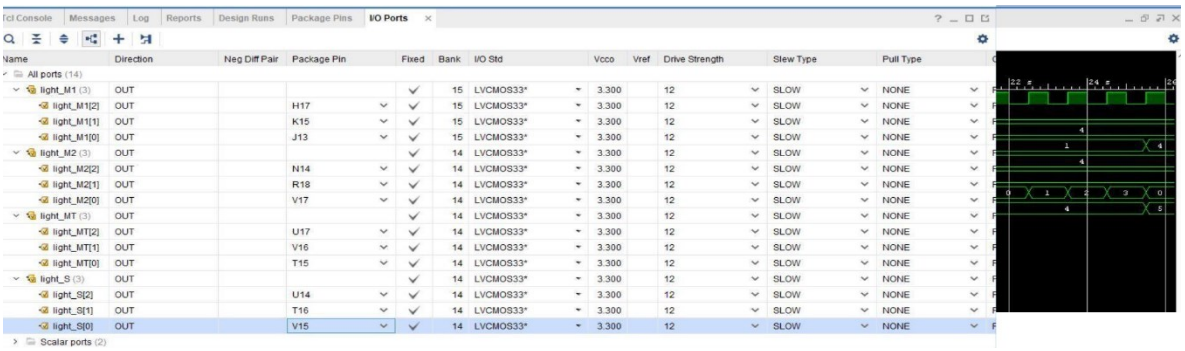

Figure 5. I/O Port Assignment

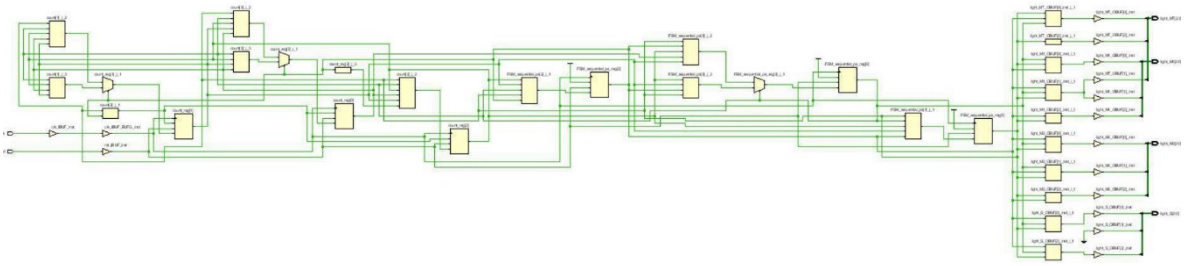

Figure 6. Schematic after Synthesis

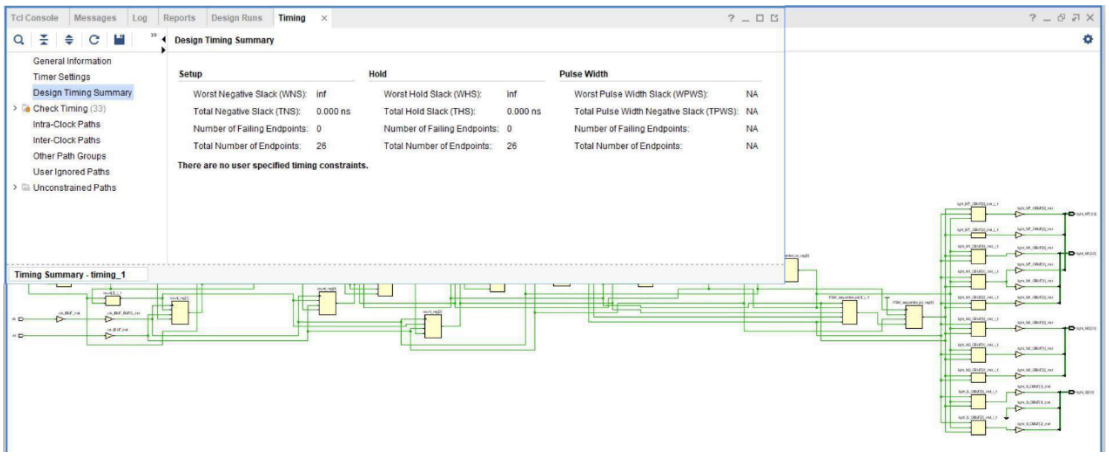

Figure 7. Timing Report after Synthesis

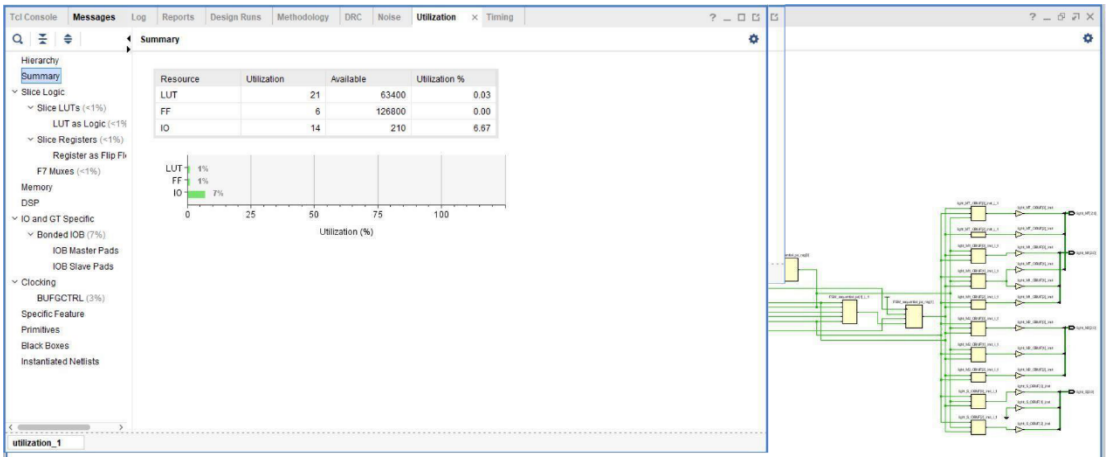

Figure 8. Utilization Report

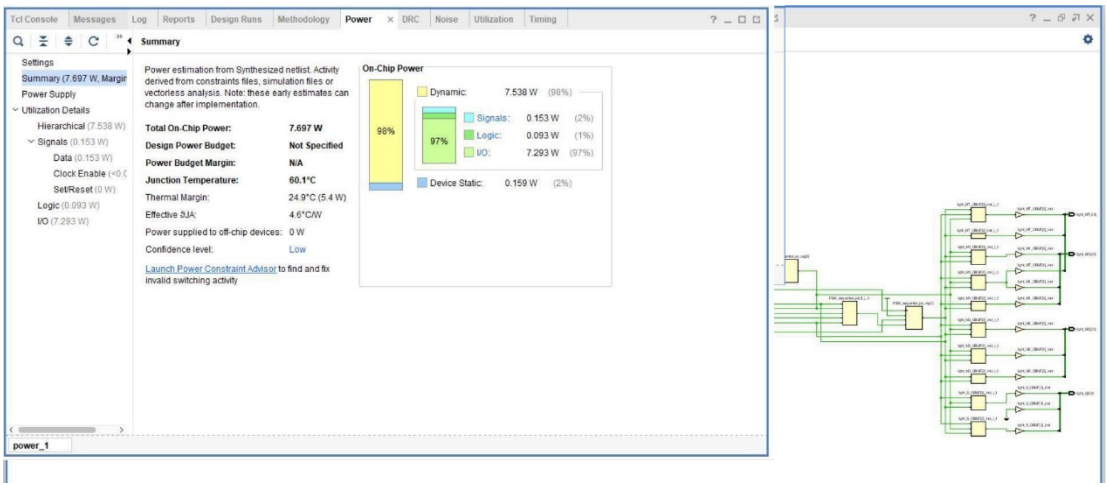

Figure 9. Power Report

5. Conclusion

In conclusion, the implementation of a traffic light controller using the FPGA ZedBoard and Vivado represents a significant advancement in the field of embedded systems and traffic management. It successfully leveraged the capabilities of the FPGA to create a robust and efficient control system for traffic lights, addressing key challenges in urban transportation. The utilization of the ZedBoard, equipped with a powerful Zynq-7000 SoC, provided a versatile platform for developing a real-time traffic light controller. The FPGA's parallel processing capabilities allows for the simultaneous handling of multiple inputs and outputs, enabling swift decision-making in response to changing traffic conditions. This not only enhances the overall efficiency of the traffic control system but also contributes to the optimization of road intersections. The design and synthesis process carried out in Vivado demonstrated the seamless integration of hardware description languages, such as Verilog or VHDL, with the development environment. This integration facilitated the creation of a complex yet efficient traffic light control algorithm, ensuring smooth transitions between signal phases and effective coordination of traffic flow. One notable achievement of the work lies in its emphasis on scalability and adaptability. The FPGA's reconfigurable nature allows for easy modifications to accommodate diverse intersection layouts, traffic patterns, and even future upgrades. This scalability ensures that the traffic light controller can be applied to various urban environments, contributing to the development of intelligent transportation systems. Furthermore, the successful implementation of the traffic light controller underscores the potential for FPGA based solutions in addressing real-world challenges. This work serves as a testament to the versatility of FPGA technology, showcasing its ability to handle complex control algorithms and real-time applications. In conclusion, the traffic light controller developed on the FPGA ZedBoard using Viva not only meets the immediate goal of efficient traffic management but also sets the stage for future innovations in intelligent transportation systems. The work opens avenues for further research and development in FPGA based embedded systems, with the potential to revolutionize urban traffic control and contribute to the advancement of smart cities.

References

[1] J. Pang, "Intelligent traffic light controller design using FPGA digest of technical papers," 2016 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 2016, pp. 449-452

[2] Pughat, A. Pritam, D. Sharma and S. Gupta, "Adaptive Traffic Light Controller for Vehicular Ad-Hoc Networks," 2019 International Conference on Signal Processing and Communication (ICSC), NOIDA, India, 2019, pp. 338-344

[3] S. S, S. M, C. V. Josphine, M. T. Kingslin, S. Sivarajan and C. L. K R, "IoT based on Smart Traffic Lights and Streetlight System," 2023 2nd International Conference on Edge Computing and Applications (ICECAA), Namakkal, India, 2023, pp. 1311-1316

[4] S. V. Lahade and S. R. Hirekhan, "Intelligent and adaptive Traffic Light Controller (IA-TLC) using FPGA," 2015 International Conference on Industrial Instrumentation and Control (ICIC), Pune, India, 2015, pp. 618-623

[5] S. Kaur, V. Varshitha, J. Siddharth, V. S. Tejus and H. Prasanna Kumar, "Adaptive Traffic Light Controller using FPGA," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2018, pp. 1612-1617

[6] M. S. Georgiou and A. Polycarpou, "Energy saving through replacement of traffic light systems," 2018 IEEE International Energy Conference (ENERGYCON), Limassol, Cyprus, 2018, pp. 1-5

[7] S. V. Kishore, V. Sreeja, V. Gupta, V. Videesha, I. B. K. Raju and K. M. Rao, "FPGA based traffic light controller," 2017 International Conference on Trends in Electronics and Informatics (ICEI), Tirunelveli, India, 2017, pp. 469-475

[8] H. Kulhandjian, W. Greives and M. Kulhandjian, "Smart Traffic Light Controller using Visible Light Communications," 2022 IEEE Vehicle Power and Propulsion Conference (VPPC), Merced, CA, USA, 2022, pp. 1-6

[9] M. Sutharsan, S. Rajakaruna, S. Y. Jayaweera, J. A. C. M. Jayaweera and S. Thayaparan, "Vision-Based Adaptive Traffic Light Controller for Single Intersection,"

2020 5th International Conference on Information Technology Research (ICITR), Moratuwa, Sri Lanka, 2020, pp. 1-6

[10] Devipriya, H. Girish, V. Srinivas, N. Adi Reddy and D. Rajkiran, "RTL Design and Logic Synthesis of Traffic Light Controller for 45nm Technology," 2022 3rd International Conference for Emerging Technology (INCET), Belgaum, India, 2022, pp. 1-5

[11] D. R. ALEKO and S. Djahel, "An IoT Enabled Traffic Light Controllers Synchronization Method for Road Traffic Congestion Mitigation," 2019 IEEE International Smart Cities Conference (ISC2), Casablanca, Morocco, 2019, pp. 709-715

[12] L. F. P. Oliveira, L. T. Manera and P. D. G. Luz, "Smart Traffic Light Controller System," 2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS), Granada, Spain, 2019, pp. 155-160

[13] S. Nath et al., "Design of an FPGA based intelligence traffic light controller with VHDL," 2012 International Conference on Radar, Communication and Computing (ICRCC), Tiruvannamalai, India, 2012, pp. 92-97

[14] M. Prabhu, A. Al Wardi, S. M. Hussain, S. Ghouse and A. V. Singh, "A Comprehensive Review of PLC based Intelligent Traffic Light Control System," 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2021, pp. 1-5

[15] Prashant Kumar Singh and P. Daniel, "Advanced real Traffic Light Controller system design using Cortex-M0 ip on FPGA," 2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies, Ramanathapuram, India, 2014, pp. 1023-1026