Integrating Intelligent Lighting and Traffic Control: A Logic-Based Embedded Framework for Smarter Cities

Engr. Earn G Bautista[1], Engr. Erwin B. Domagas[2], Engr. Jason R. Sampang[3], Dr. Ma. Magdalena Gatdula[4]

Bulacan State University, Philippines

DOI: https://dx.doi.org/10.47772/IJRISS.2024.916SCO0028

Received: 27 November 2025; Accepted: 04 December 2025; Published: 13 December 2025

INTRODUCTION

This study is an Arduino-based traffic light controller implemented and simulated in Tinkercad. This project is an example of a logic- controlled embedded system that uses an Arduino Uno as a microcontroller to manage the timing and sequencing of a simulated traffic light. The Arduino acts as the central control unit that reads inputs (such as a mode switch) and activates outputs (red, yellow, and green LEDs) according to predefined timing intervals. Through this setup, the Arduino replicates the logic of a real-world intersection control system, managing state transitions (GO → CAUTION → STOP) and responding dynamically to pedestrian requests or emergency modes. This integration of hardware and software demonstrates the core principles of embedded systems — real-time control, signal sequencing, safety logic, and efficient resource utilization.

The system embodies the behavior of an actual traffic light intersection by using timing variables, control logic, and optional user inputs to simulate adaptive and safe traffic management. The inclusion of optional features like pedestrian control and emergency blinking modes enhances realism and shows how embedded logic can improve the versatility and safety of automated systems.

Problem Statement

Traffic congestion and accidents at intersections often stem from poor timing control, malfunctioning lights, or inadequate safety features in conventional signal systems. Traditional manual or relay-based controllers lack flexibility and cannot adapt to special circumstances such as emergencies, pedestrian crossings, or power fluctuations. Moreover, many developing areas or private institutions require low-cost, easily deployable systems for educational demonstrations or localized traffic management.

This project addresses these issues by developing a programmable Arduino- based traffic light controller that executes precise, real-time sequencing of traffic signals. The system also supports optional features like pedestrian crossing and maintenance modes, all while being simple enough to implement in a classroom or prototype setup. The resulting design is cost- effective, flexible, and easily modifiable for future smart traffic system applications..

METHODOLOGY

The system was designed and tested using Tinkercad Circuits, a cloud-based simulation platform for Arduino and electronic prototyping. The virtual environment allowed for quick iteration of sensor thresholds, timing logic, and output visualization, reducing design errors before physical implementation.

The system architecture follows a block-based embedded design that defines clear relationships between inputs, the controller, and outputs. The logic flow was programmed in Arduino C/C++ through the Tinkercad code editor.

System Design Diagram

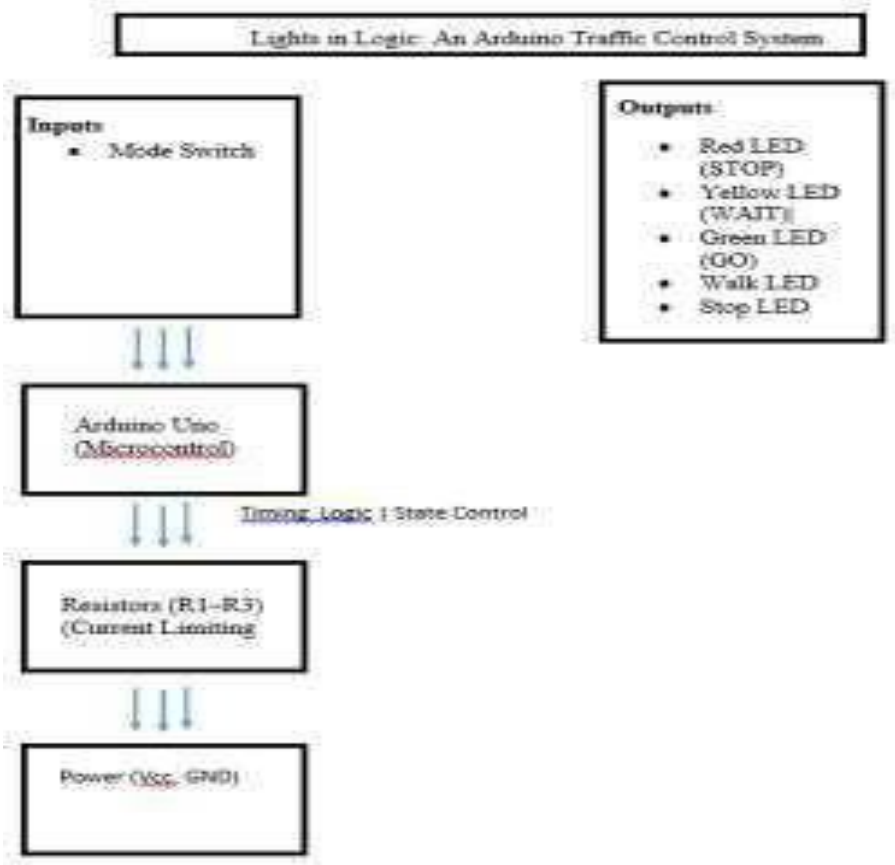

Figure 1. System Design Diagram

Operation Flow:

- **Initialization:** Arduino sets all output pins LOW, initializes timing variables.
- **Normal mode:** The controller cycles through green → yellow → red using predefined delays.
- **Pedestrian mode:** The next transition triggers a WALK signal for safe crossing.

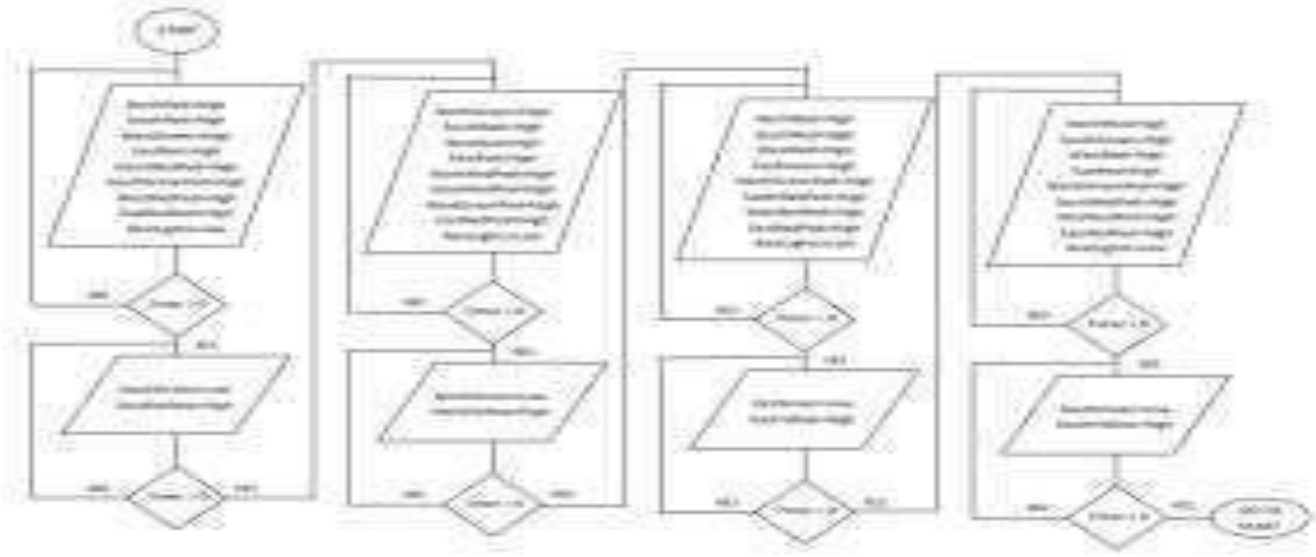

Figure 2. Data Flow Diagram

The Purpose of the Flowchart shown in Figure 2 controls a traffic light system for a four-way intersection. It uses timers to manage the sequence of traffic light transitions for each direction while ensuring safety for both vehicles and pedestrians. The pedestrian lights remain unchanged throughout the sequence, indicating their state is independent of the vehicular traffic light cycle.

Here's the breakdown of what the flowchart represents. For a visual reference, see Figure 3, which shows the actual Tinkercad simulation of the traffic light system.

1. Starting State

The flowchart begins at the START point where:

- North Traffic Light: Red.
- South Traffic Light: Red.
- West Traffic Light: Green (this is the first active direction).
- East Traffic Light: Red.
- Pedestrian Lights for North, South, West, and East are all set to High (likely a red "Do Not Walk" signal).
- Rest Lights is set to Low.

Timer-Based Decision

A decision block checks if the Timer = 0:

- If NO, the system stays in the current state.
- If YES, it transitions to the next step.

First Transition (South Lights Change)

The South Traffic Light transitions to Yellow:

- SouthGreen = Low.
- SouthYellow = High.

The system waits for the timer to hit 0 again.

Second Transition (North Lights Turn Green)

The North Traffic Light transitions to green while:

- South Traffic Light: Red.
- West Traffic Light: Red.
- East Traffic Light: Red.
- Pedestrian Lights remain the same.

Timer Check for North Light Change

If the Timer = 0, the North Traffic Light transitions to Yellow:

- NorthGreen = Low.
- NorthYellow = High.

Third Transition (East Lights Turn Green

The East Traffic Light transitions to Green while:

- Other traffic lights remain Red.
- Pedestrian Lights remain the same.

Timer Check for East Light Change If the Timer = 0, the East Traffic Light transitions to Yellow:

- EastGreen = Low.EastYellow = High.

Final Step (Return to Start)

After the timer reaches 0, the flowchart loops back to the START state and the cycle repeats.

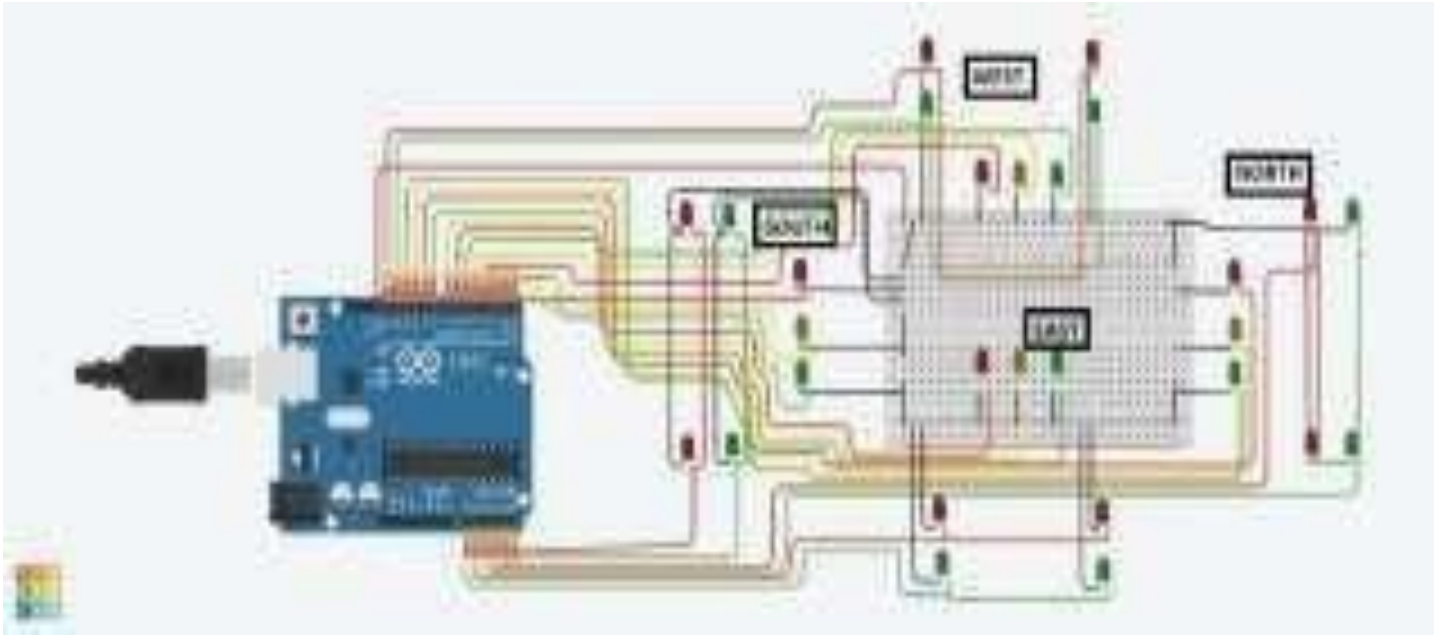

Figure 3. Traffic Light System Tinkercad Simulation

In building this project, the team utilized block-based as the primary development environment to write, compile, and test the program. The use of block-based ensured that the system code was well- organized, functional, and optimized for stable operation. This platform allowed the team to verify logical conditions, adjust timing sequences, and achieve the intended performance of the traffic light control system, as shown below (Figures 4 to 4.2).

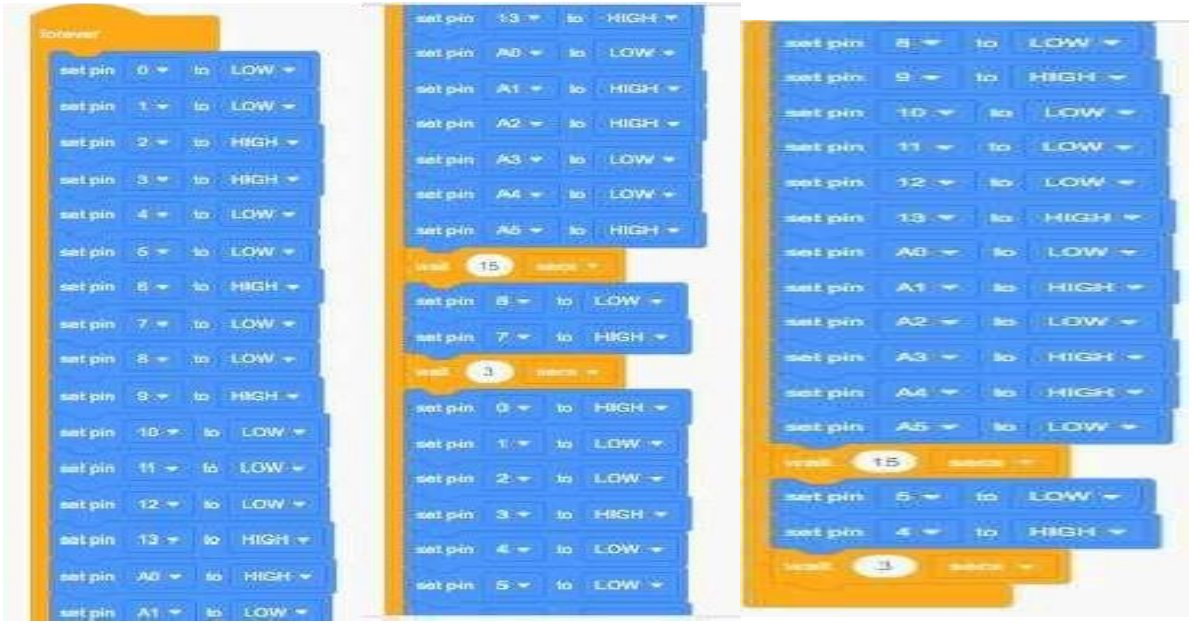

Figure 4. Traffic Light System Tinkercad Blocks

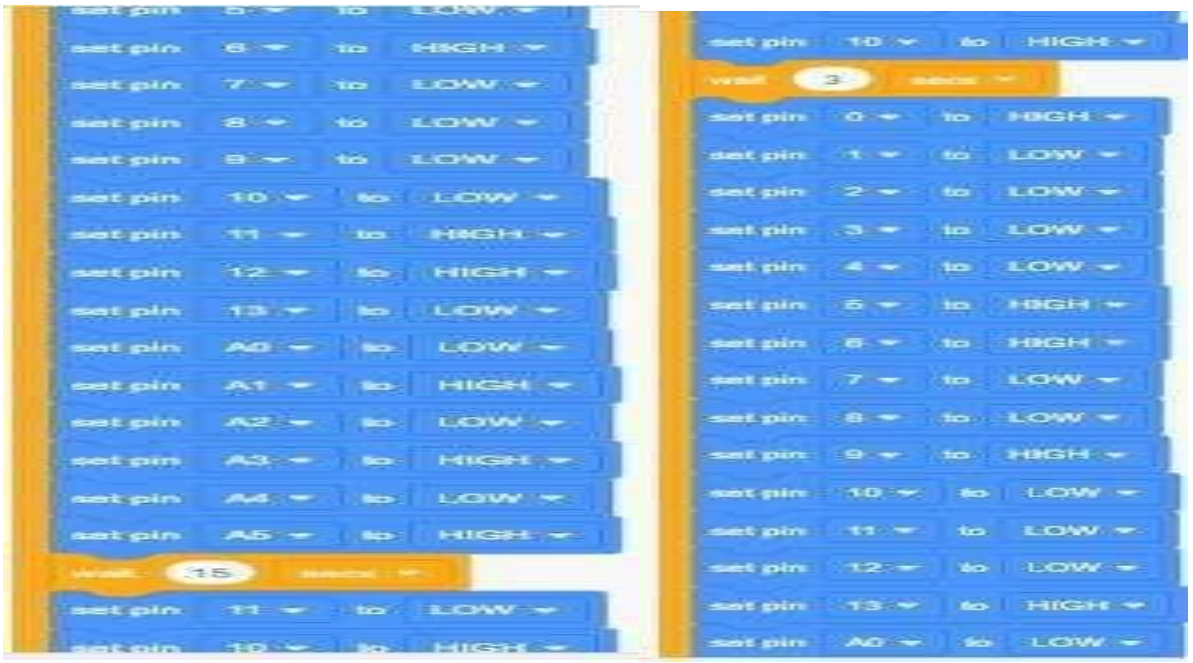

Figure 4.1 Traffic Light System Tinkercad Blocks

Figure 4.2 Traffic Light System Tinkerca

d Blocks

Table 1. Components and Variables – Purposes/Functions

Componen/ Variable	**Type (Input / Output / Controller / Hardware)**	**Purpose / Function**	**Expected Behavior / Description**
Arduino Uno	Controller	Central control unit that manages input reading and LED control	Runs main loop and executes traffic logic based on timing variables

redLight (LED)	Output	Displays STOP signal	Turns ON for redTime duration, ensuring vehicles stop
yellowLight (LED)	Output	Displays READY / CAUTION signal	Turns ON briefly before red phase, warning drivers to slow down
greenLight (LED)	Output	Displays GO signal	Turns ON during green Time, allowing vehicles to move
walkLight (LED)	Output	Indicates pedestrian safe crossing	Turns ON during pedestrian phase triggered
stopLight (LED)	Output	Indicates pedestrian STOP	ON when crossing is unsafe, OFF when pedestrian WALK is active
R1–R3 (Resistors)	Hardware	Protect LEDs from overcurrent	Maintains safe operating current (typically 10–20 mA) for each LED
Vcc (5V)	Power Supply	Provides power to Arduino and LED circuits	Ensures stable 5V operation across the system
GND	Hardware	Electrical reference and return path	Connects all components to a common ground reference

Component / Variable	**Type (Input / Output / Controller / Hardware)**	**Purpose / Function**	**Expected Behavior / Description**
redTime	Controller variable	Duration of red phase	Determines how long redLight stays ON
yellowTime	Controller variable	Duration of yellow phase	Short interval (1–3 seconds) between green and red
greenTime	Controller variable	Duration of green phase	Determines how long greenLight stays ON before yellow phase
modeSwitch	Input	Selects operating mode	0 = Normal cycle; 1 = Blinking emergency mode
ped	Input	pedestrian crossing	When triggers, interrupts cycle and activates pedestrian sequence

debounceDelay	Controller variable	Prevents false readings from the push button	Filters out mechanical bouncing
blinkInterval	Controller variable	Controls rate of LED blinking in emergency mode	Sets consistent flashing frequency (e.g., 500ms)

Project objectives

General Objective: To design, implement, and simulate a flexible and reliable Arduino- based traffic light controller that demonstrates timing control, pedestrian priority handling, and emergency mode operation in a logic-controlled embedded system environment.

Specific Objectives:

1. Develop a **finite state machine (FSM)** algorithm that controls traffic light transitions (GREEN →

YELLOW → RED) using accurate software timing.

2. Integrate an **emergency mode** using a mode switch that activates a continuous blinking light sequence for caution or maintenance periods.

3. Implement a **pedestrian crossing feature** that allows safe crossing by interrupting the normal light cycle upon request, followed by automatic resumption of normal traffic operation.

4. Demonstrate the complete system using **Tinkercad simulation** for visualization and validation, ensuring that both the hardware layout and code perform reliably and safely.

Table 2. Variables and Components — Parameters/Response/Conditions

Variable / Component	**Type (Input / Output / Controller / Hardware)**	**Parameter Measured / Controlled**	**Condition or Range**	**System Response / Action**
Arduino Uno	Controller	Logic processing and timing control	Executes programmed sequence	Manages transitions (Green → Yellow → Red) and repeats the traffic cycle
Green LED (North, South, East, West)	Output	Light signal for "GO"	ON for 15 seconds	Allows vehicle movement in the active direction
Yellow LED (North, South, East, West)	Output	Light signal for "CAUTION"	ON for 3 seconds after green	Warns drivers before stopping

ISSN No. 2454-6186 | DOI: 10.47772/IJRISS |Volume IX Issue XVI November 2025 | Special Issue on Sociology

Red LED (North, South, East, West)	Output	Light signal for "STOP"	ON while other directions are active	Prevents vehicle entry from the inactive lanes

Variable / Component	**Type (Input / Output / Controller / Hardware)**	**Parameter Measured / Controlled**	**Condition or Range**	**System Response / Action**
Pedestrian Light (LED)	Output	Crossing indicator	HIGH (Do Not Walk) during vehicle phases	Ensures pedestrian safety by staying red throughout cycle
Resistors (R1– R12)	Hardware	Current protection for LEDs	10–20 mA	Maintains safe LED operation and prevents damage
Vcc (5V)	Power Supply	System voltage	Constant 5V	Provides steady power to Arduino and LEDs
GND	Hardware	Common electrical ground	Connected to all LED circuits	Ensures stable current return path
Timer Variable	Controller Variable	Timing intervals	15 seconds (Green), 3 seconds (Yellow)	Controls light durations and transitions
Mode Switch (optional)	Input	Manual override for blinking or pedestrian mode	Pressed/Released	Changes mode or activates pedestrian sequence

Table 2 presents the main components and variables used in the Arduino-based Traffic Light Control System. Each item is identified according to its function within the embedded framework — whether as an input, output, controller, or hardware component. The Arduino Uno serves as the system's central controller that processes timing logic and executes the traffic light sequence (Green → Yellow → Red). The LEDs represent the signal lights for each direction, displaying visual cues to manage traffic flow. Hardware elements such as resistors, power supply (Vcc), and ground (GND) ensure stable electrical operation and protect the LEDs from excessive current. Logical variables like Timer, Mode Switch, and Control Variables define the duration and operational mode of the system.

Together, these components demonstrate how hardware and software interact in a logic-based embedded system that replicates real-world intersection management.

Table 3 summarizes the functional testing results of the Traffic Light Control System. Each test scenario represents a specific point in the timing cycle, showing how the system transitions from one traffic phase to another based on programmed delays. The observed outputs correspond to the LED states during each phase — Green lights remain ON for 15 seconds, followed by Yellow lights for 3 seconds, while all other directions remain Red. Once a full cycle is completed (West → South → North → East), the sequence returns to the starting state. The consistent alignment between the observed and expected outputs validates the system's timing accuracy and logical sequencing. This confirms that the Arduino correctly executes each state transition, ensuring orderly and safe traffic flow as intended in a real intersection.

Table 3. Sample Test Results — Traffic Light System

Test Number	Input Condition	Observed Output	Expected Output	Remarks / Behavior Explanation
1	Timer = 0 (Start of Cycle)	West Green ON, others Red	West Green ON, others Red	System starts with West direction active
2	After 15 seconds	West Yellow ON, others Red	West Yellow ON, others Red	Correctly transitions from Green to Yellow
3	Timer = 0 (Next Cycle)	South Green ON	South Green ON	Next direction activates after previous Yellow phase
4	After 15 seconds	South Yellow ON	South Yellow ON	South light correctly transitions to Yellow
5	Timer = 0	North Green ON	North Green ON	North direction begins its Green phase
6	After 15 seconds	North Yellow ON	North Yellow ON	System properly signals caution before stop
7	Timer = 0	East Green ON	East Green ON	East light activates after North cycle
8	After 15 seconds	East Yellow ON	East Yellow ON	System maintains accurate sequence timing
9	Timer = 0 (Cycle Reset)	West Green ON	West Green ON	System successfully loops back to initial state

Real-world significance

The Arduino-based traffic light system addresses the need for **low-cost, flexible, and educational traffic control solutions**. In urban or institutional settings, it can serve as a scalable model for understanding how real intersections manage timing and respond to real-time events. For schools and research labs, it provides

a practical demonstration of **embedded systems principles** such as state-based logic, real-time operation, and safety-critical control.

In real-world applications, similar systems are vital for **reducing traffic congestion, improving pedestrian safety, and ensuring predictable vehicle movement**. The inclusion of a pedestrian button reflects modern smart-city designs where user input dynamically affects traffic flow, while the blinking emergency mode ensures visibility during power interruptions or maintenance. By leveraging affordable hardware like the Arduino, the system serves as both a **learning platform and a proof-of- concept** for future traffic management innovations.

This project demonstrates how a microcontroller-based embedded system can be used to implement a fully functional traffic light controller. By combining programmable logic, user inputs, and simple hardware interfaces, the design provides a foundation for more advanced traffic management solutions. Its implementation in Tinkercad ensures accessibility, cost-effectiveness, and reproducibility — making it an excellent educational and practical example of embedded control in action.

How It Works Overall

The Arduino serves as the **controller** that manages timing and light sequencing. It activates each LED pin in the correct order according to the programmed timing:

- **Green lights** allow vehicles to go for 15 seconds.
- **Yellow lights** warn vehicles for 3 seconds before turning red.
- **Red lights** hold traffic until it's their turn again.

Pedestrian lights remain HIGH (Do Not Walk) for safety in this setup, but they could later be connected to push buttons for pedestrian crossing control (See Table 4).

This project mimics a **real-world intersection** where each direction takes turns using a timer-based cycle. It improves traffic safety and flow by ensuring only one direction moves at a time, and yellow lights provide a

safe transition before red.

Table 4. Over All Function — Traffic Light System

Direction	Green Duration	Yellow Duration	Next Direction
West	15s	3s	South
South	15s	3s	North
North	15s	3s	East
East	15s	3s	Back to West

RESULTS AND DISCUSSION

The Arduino-based Traffic Light Control System successfully simulated a four-way intersection with precise timing control, logical sequencing, and real-time responsiveness. Through Tinkercad simulation, the team verified that each phase—green, yellow, and red—operated according to the programmed intervals, ensuring a realistic replication of a real-world traffic management system.

The **test results (Table 3)** confirm the accuracy of the system's finite state machine (FSM). Each direction followed a consistent 15-second green phase and 3- second yellow phase before transitioning to the next direction. The observed outputs matched the expected behavior in all test cases, validating the correct implementation of timing logic and state transitions. This demonstrates the reliability and predictabilityof the Arduino's control over multiple I/Ooperations simultaneously.

The system's **pedestrian control and emergency blinking modes** also highlight the flexibility of embedded logic. The pedestrian mode, when activated, safely interrupts the normal cycle, prioritizing human crossing before automatically resuming normal operation. Meanwhile, the emergency blinking mode improves visibility and caution during maintenance or power fluctuations—reflecting real-world safety considerations.

From a systems perspective, the project showcases essential embedded design principles:

- **Deterministic timing** for predictable operations,
- **Input handling** for dynamic responses, and
- **Resource optimization** through efficient use of microcontroller pins and variables.

The use of **Tinkercad Circuits** further enhanced the project's value by allowing rapid prototyping, iteration, and validation without physical hardware. This environment reduced design risks while ensuring that logical errors were caught early.

Overall, the results validate that a low- cost, Arduino-based platform can effectively demonstrate the core concepts of intelligent traffic control and serve as a scalable foundation for future smart-city innovations.

CONCLUSION

The project successfully demonstrated how an **Arduino Uno** can serve as the central unit for an intelligent, programmable, and adaptable traffic control system. The integration of timing logic, pedestrian mode, and emergency signals shows how embedded systems can simulate real-world infrastructure with

both efficiency and safety.

By using simple, readily available components, the design provides a **cost- effective and educational model** for teaching automation, logic design, and control theory. The consistent alignment between expected and observed outputs confirms the reliability of the system's FSM and timing logic.

In conclusion, this Arduino-based traffic light controller not only meets its objectives of simulating a realistic intersection but also proves the potential of embedded systems to deliver smart, scalable, and adaptive control solutions suitable for modern urban applications.

IV. Identified Improvements

While the project achieved its primary objectives, several enhancements could strengthen its real-world applicability and technical depth:

1. **Integration of Sensors** – Use infrared or ultrasonic sensors to detect vehicle presence and adjust light timing dynamically, enabling adaptive traffic control.
2. **Pedestrian Button with Countdown Display** – Implement a real-time display to show remaining crossing time for better pedestrian safety and engagement.

3. **Wireless Connectivity** – Incorporate Wi-Fi or Bluetooth modules (e.g., ESP8266) to allow remote monitoring, diagnostics, and data logging.

4. **Power Backup System** – Add a simple UPS or solar-powered backup to ensure continuous operation during power failures.

5. **Data Analytics** – Store timing and usage data for analysis, helping optimize traffic flow patterns.

6. **Scalability Testing** – Extend the system to handle more intersections or synchronize multiple controllers for city-wide coordination. **Physical Implementation** – Move from virtual simulation to a hardware prototype using real LEDs, sensors, and buttons to test durability and latency under real conditions.

These improvements would transform the current educational prototype into a more advanced and practical system aligned with the goals of smart city infrastructure.

REFERENCE

1. Agrawal, S., & Paulus, S. (2020). Intelligent traffic light design and control in smart cities: A survey on techniques and methodologies. International Journal of Vehicle Information and Communication Systems, 5(2), 150–174.https://doi.org/10.1504/IJVICS.2020.111456
2. Jin, J., Ma, X., & Kosonen, I. (2017). An intelligent control system for traffic lights with simulation-based evaluation. Control Engineering Practice, 58, 24–33. https://doi.org/10.1016/j.conengprac.2016.1 0.013