TITLE: DESIGN AND IMPLEMENTATION OF AUTOMATIC STREET LIGHT SYSTEM

COLLEGE OF ENGINEERING

COMPUTER ENGINEERING

GROUP 10

ICT 215: ROBOTICS I

BELLS UNIVERSITY OF TECHNOLOGY,OTA (BUT) – NEW HORIZON

GROUP 10

Ikeh Chidiebere Kingdavid 2023/12694

Ewuzie Chinecherem 2023/12230

Salako Ilyas COMPUTER ENGINEERING 2023/12630

Akindahunsi Akintayo COMPUTER ENGINEERING 2023/12711

Adeoye Jeremiah COMPUTER ENGINEERING 2023/12722

SUBMITTED TO AYUBA MUHAMMAD

TABLE OF CONTENTS

I. GROUP 10

- Title Page or Team Overview (This could be an introduction to the members or team goals).

II. INTRODUCTION

- Background of the Report
- Purpose and Scope of the Project

III. HOW TO BUILD THIS PROJECT

- Materials Required
- Step-by-Step Assembly Instructions

IV. PROJECT OVERVIEW

- Brief Summary of the Project Goals
- Challenges Addressed by the Project

V. PROJECT OBJECTIVES

- Primary Goals of the Project
- Secondary or Supporting Objectives

VI. PROJECT GUIDELINES

- Rules and Standards Followed
- Safety Considerations

VII. Key Components in the Circuit

- Description of Essential Components
- Technical Specifications

VIII. LIGHTING SYSTEMS

- Types of Lighting Systems Used
- Applications of the Systems in the Project

IX. Energy Efficiency

- Explanation of Energy-Saving Features
- Environmental Benefits

X. Cost Savings

- Financial Benefits of the Project

Forward

This project, titled **“Automatic Street Light System”**, is a group effort by Group
10 as part of the requirements for ICT 215 Robotics I. It reflects our attempt to use what we’ve learned in class to solve a practical problem using automation and technology.

The idea for this project came from the need to save energy and make streetlights work smarter without constant human involvement. Using components like the Arduino, an LDR, and an LCD, we created a system that automatically controls streetlights based on the light intensity around them.

We want to thank our instructor, **Mr. Mohammed**, for guiding us throughout this project. His advice helped us handle challenges and improve our understanding of the concepts involved.

This forward is just a brief introduction to our project, and we hope it shows how technology can be used to solve real-life problems in simple but effective ways.

Course Instructor: Mr Ayuba Muhammad.

Description

This project is all about the use of Proteus and Arduino software, it uses an LDR sensor to detect light levels and a transistor switch to turn the street light on in low light and off in high light, reducing energy consumption.

INTRODUCTION

WHAT IS AN AUTOMATIC STREET LIGHT?

An

automatic streetlight system is a lighting system designed to control streetlights automatically based on environmental conditions, such as ambient light levels or time of day. It eliminates the need for manual operation, improving energy efficiency, convenience, and reliability.

It consists of several key components.

HOW TO BUILD THIS PROJECT

To build this project, you

need; a The Proteus 8

Professional. b Arduino

software and Library. c

topic of choice.

PROJECT OVERVIEW

The aim of this project is to design and implement an **Automatic Streetlight System** using Proteus and Arduino. This system mimics the functionality of modern streetlights that automatically turn on at night and off during the day based on ambient light intensity. By utilizing a Light Dependent Resistor (LDR) to detect light levels, resistors, transistors, LEDs, and an Arduino microcontroller, the project provides a simple yet practical demonstration of how automation can optimize energy consumption and improve efficiency.

This project is particularly relevant in the context of energy-saving initiatives and smart city concepts, where automation reduces human intervention and ensures reliability.

PROJECT OBJECTIVES

Design and simulate functional robotic systems integrating sensors, actuators and microcontroller programming using Proteus, Tinkercad and Arduino.

Workflow: Proteus Simulation, Arduino Code Development, Hardware Integration, Debugging and Optimization, Documentation)

PROJECT GUIDELINES

Topic Selection:

- Choose a single topic from one of the predefined sections relevant toRobotics applications.
- Each student group must select one topic and one section only.● If you wish to propose a unique topic not listed here, submit a proposal for approval. Once approved, you may begin work on this topic.

Report Writing:

- Your group is required to produce a comprehensive project report in line with academic standards.
- Research and reference similar scholarly articles, journals, and relatedprojects to inform your work.
- Note: Plagiarism is strictly prohibited; directly copying content or usingAI-generated reports will result in penalties.

Team Collaboration:

Each department Engineering) is encouraged to approach the project based on departmental strengths.

T eams are expected to work collaboratively while being prepared to defend their contributions independently during presentations.

GitHub Repository and Team Contributions:

- All implemented projects must be published on GitHub, with each groupmember added as a contributor to the repository.
- Each group must have a designated team leader responsible for submitting the

GitHub link to the project upon completion.

- Each team member must fork the project repository to their personal

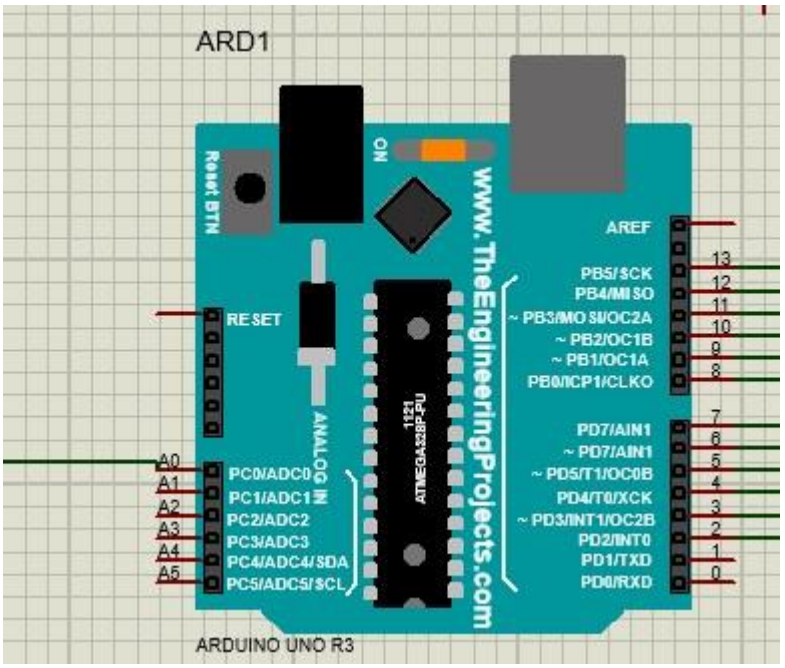
GitHub
accounts.

- The repository must include a well-organised README.md file, clearly explaining the project, setup instructions, and any dependencies.

Key Components in the Circuit

Arduino Uno (ARD1):

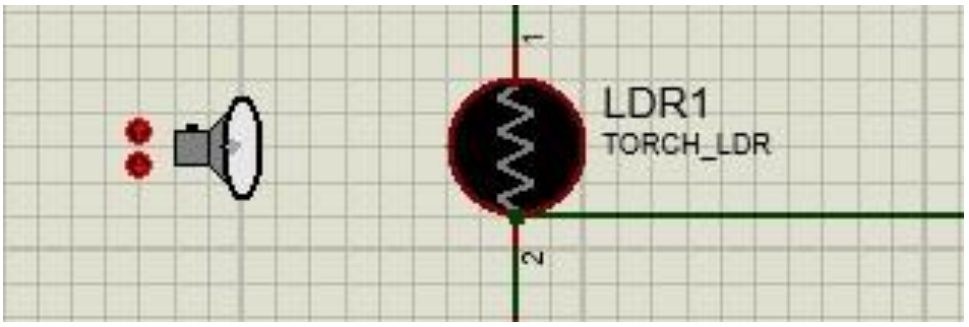
- The **Arduino Uno** is the central microcontroller unit that processes input from the light-dependent resistor (LDR) and controls the outputs to the LEDs and the LCD.
- The program uploaded to the Arduino defines how it responds to the light levels detected by the LDR.
- It receives analog input from the LDR on pin `A0` and controls digital output on its other pins (e.g., D2 to D7 for LEDs).



⍰⍰ Light Dependent Resistor ⍰LDR1⍰:

- The **LDR** is the primary sensor that detects ambient light levels. Its resistance decreases as the surrounding light intensity increases.

- Connected to a voltage divider circuit (with R1), it provides an analog voltage signal to pin `A0` of the Arduino. This signal corresponds to the detected light intensity.
- When the light level drops below a predefined threshold (e.g., at night), the Arduino triggers the LEDs to turn on.



Resistor R1 (10kΩ):

- **R1** works as a part of a voltage divider circuit with the LDR.
- This configuration ensures that the Arduino receives a proportional voltage signal based on the resistance of the LDR (which changes with light intensity).
- R1 stabilizes the voltage to make the sensor more reliable.

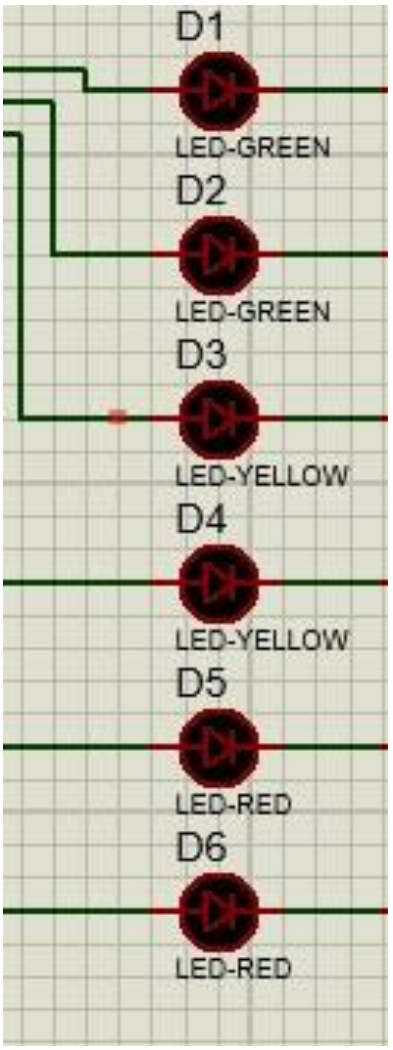
Light-Emitting Diodes (LEDs) – D1 to D6:

- The LEDs represent streetlights in this system.
- They are categorized as:
  - Green LEDs: Indicating sufficient light or daytime.

Yellow LEDs: Indicating transitional light levels (e.g., dusk or cloudy weather).

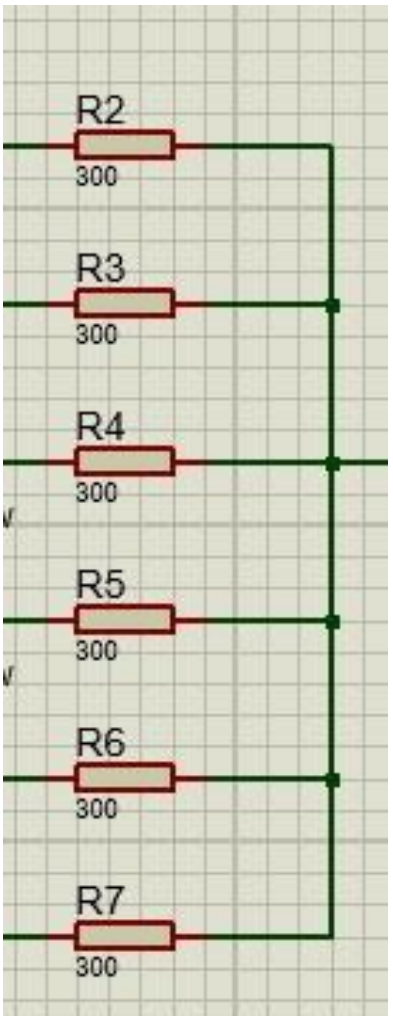
  - Red LEDs: Indicating low light or nighttime conditions where the streetlights are fully activated.

- Each LED is connected in series with a **current-limiting resistor** (R2 to R7) to prevent damage from excessive current.



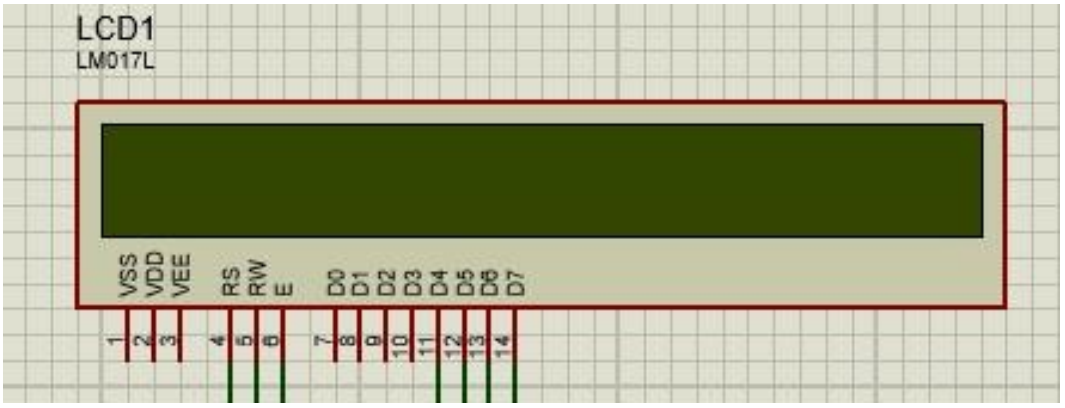
□□ Resistors □R2 to R7 □ 300Ω):

- Each LED has a **300Ω resistor** in series to limit the current flowing through it, ensuring the LEDs operate within safe limits.


R2
300
R3
300
R4
300
R5
300
R6
300
R7
300

16□2 LCD Display □LCD1□:

- The **LCD** provides a visual display of the system’s status, such as the current light intensity or the mode of operation (e.g., "Daytime," "Nighttime").
- It is connected to the Arduino's digital pins for data communication □D7 to D12□ and power pins for operation.
- The LCD’s display enhances user interaction and allows real-time monitoring.



IMPORTANCE OF AUTOMATION IN STREET LIGHTING SYSTEMS

1. Energy Efficiency

- **Reduced Power Consumption**:
  - Automated systems ensure that streetlights are turned on only when needed (e.g., during nighttime or low-light conditions) and off during the day.
  - Advanced features like dimming during off-peak hours further optimize energy usage.
- **Adoption of LED Technology**:
  - Automated systems often integrate energy-efficient LED lights, which consume significantly less power compared to traditional lighting systems like halogen or incandescent bulbs.

---

2. Cost Savings

- **Lower Electricity Bills**:
  - Automated operation reduces unnecessary lighting during the daytime or well-lit conditions, leading to significant cost savings in electricity bills.
- **Reduced Maintenance Costs**:
  -

Automation allows for proactive monitoring of the system, detecting faults (e.g., bulb failure) early, which reduces the cost and frequency of manual inspections.

---

3. Enhanced Convenience

- **Eliminates Manual Intervention**:
  - Automation removes the need for personnel to manually turn lights on/off, especially in large-scale deployments like urban areas, highways, and industrial complexes.
- **Remote Control**:
  - With IoT (Internet of Things) integration, modern systems allow remote monitoring and control of streetlights from a centralized location.

---

4. Improved Safety

- **Enhanced Visibility**:
  - Automated streetlights ensure that streets, highways, and public areas are well-lit during nighttime or adverse weather conditions, reducing the risk of accidents.
- **Crime Prevention**:
  - Adequate and timely lighting deters criminal activity, providing safer environments for pedestrians and drivers.

---

5. Environmental Sustainability

- **Energy Conservation**:
  - Automated systems reduce unnecessary energy usage, contributing to global efforts to conserve energy.
- **Integration with Renewable Energy**:
  - Many automated systems are solar-powered, reducing reliance on nonrenewable energy sources and lowering carbon emissions.

---

6. Smart City Integration

- **IoT and Smart Features**:
  - Automated streetlighting is a core component of smart cities, integrating with systems like traffic management, environmental monitoring, and emergency response.
- **Data Collection**:
  - These systems can collect data on energy usage, traffic patterns, and maintenance needs, aiding in urban planning and optimization.

---

7. Customization and Adaptability

- **Dynamic Adjustments**:
  - Automated systems can be programmed to adjust lighting intensity based on traffic levels, time of night, or specific events (e.g., festivals or emergencies).

- **Weather Sensitivity**:
  - Some systems can adapt to environmental changes, such as turning on earlier during foggy or stormy conditions.

---

8. Long-Term Durability

- **Optimized Operation**:
  - Automated systems prevent overuse of lights by operating only when necessary, prolonging the lifespan of components such as bulbs and sensors.

---

9. Reduced Light Pollution

- **Targeted Illumination**:
  - Automation allows precise control of when and where lights are active, minimizing unnecessary light emission into areas that don’t require it.
- **Dimming Technology**:
  - Lights can dim during low traffic hours, reducing light pollution while still maintaining safety.

---

10. Scalability

- **Easier Expansion**:
  - Automated systems are modular and can be easily expanded to accommodate growing cities, new neighborhoods, or additional features like motion sensors.

CONCEPTS AND WORKING PRINCIPLES OF THE AUTOMATIC STREET LIGHT

1. Concepts:

□□ Light Dependent Resistor □LDR□□

- The core concept revolves around the **LDR**, a photosensitive component whose resistance changes based on light intensity.
- When exposed to bright light, its resistance decreases; in darkness, its resistance increases.

□□ Voltage Divider Principle:

- The LDR is often used in a voltage divider circuit to produce a variable voltage output depending on the ambient light intensity.
- This output voltage is then used as a signal to trigger the system.

□□ Transistor as a Switch:

- A transistor (e.g., BC548□ acts as a switch, controlling the flow of current to the LEDs (streetlights) based on the input from the LDR circuit.

□□ Arduino Microcontroller □Optional):

- In more advanced designs, an Arduino is used to process the input signal from the LDR and execute logic to control the LEDs.
- It allows for more customization, such as dimming, delay times, or integrating additional features like motion sensors.

2. Working Principles:

□□ Daytime Mode □High Light Intensity):

- During the day, when there's sufficient ambient light, the LDR's resistance is low.
- In a voltage divider circuit, this results in a lower voltage at the base of the transistor or at the Arduino input pin.
- The transistor remains off (or the Arduino doesn't activate the output), so the LEDs remain off, saving energy.

□□ Nighttime Mode □Low Light Intensity):

- At night, the ambient light reduces, and the LDR’s resistance
- increases.

This results in a higher voltage at the base of the transistor or the Arduino input pin.

- The transistor turns on, allowing current to flow through the LEDs, or the Arduino activates the output pin, turning the LEDs on.

□□ Dynamic Transition:

- As the light intensity gradually changes (e.g., during sunset or sunrise), the system transitions smoothly between on and off states, thanks to the LDR’s sensitivity to light levels.

SYSTEM DESCRIPTION (how it works, when it activates, etc.)

When the automatic streetlight system activates (usually during low light or nighttime), here's the step-by-step explanation of how it works:

1. Sensing Low Light:

- **Light Dependent Resistor (LDR)**
  - As the ambient light intensity decreases (e.g., during sunset or in darkness), the **resistance of the LDR increases**.
  - This change in resistance alters the voltage output of the voltage divider circuit connected to the LDR.
- **Voltage Divider Output:**
  - The voltage across the LDR becomes higher due to its increased resistance.
  - This voltage acts as a signal to trigger the rest of the circuit.

2. Activating the Transistor or Arduino Logic:

- **Transistor as a Switch:**
  - the system uses a **transistor (BC548)**, the voltage from the LDR is applied to the base of the transistor.

- Once the base voltage crosses the threshold, the transistor switches on, allowing current to flow through the LEDs (streetlights).

- **Arduino-Based Control:**
  - the LDR's voltage is read as an **analog input**.
  - The Arduino compares the input voltage with a predefined threshold (set in the code).
  - If the voltage indicates low light, the Arduino sends a signal to its output pin, activating the LEDs.

3. Powering the LEDs:

- When the transistor or Arduino activates the circuit, the **LEDs light up**. This represents the streetlights turning on.
- The **power source (external supply)** provides the necessary current for the LEDs to function.

4. Continuous Monitoring:

- The system continuously monitors the light intensity through the
- LDR.

If the ambient light increases (e.g., during sunrise or when artificial light is introduced), the LDR’s resistance decreases.

- The reduced voltage causes the transistor to switch off (or the Arduino to deactivate the output), turning the LEDs off.

Summary of Activation Process:

□□ LDR senses low light levels.

□□ Voltage increases at the transistor base or Arduino input pin.

□□ Transistor turns on (or Arduino activates output), allowing current to flow.

□□ LEDs light up, simulating streetlights turning on.

APPLICATION AND POTENTIAL USE CASES OF THE AUTOMATIC STREET LIGHT SYSTEM

1. Residential and Housing Estates

- Used in gated communities, housing estates, and private
- neighborhoods.
-

Reduces energy consumption by activating lights only when needed.

Enhances security by ensuring well-lit surroundings during low light conditions.

2. Highways and Expressways

- Enhances visibility and safety for vehicles on highways at night.
- Reduces accidents caused by poor lighting or sudden darkness.
- Minimizes maintenance costs through automated operation.

3. Railways and Airports

- Used in railway platforms, stations, and airport runways for consistent and efficient lighting.
- Ensures safe navigation for passengers and personnel.

These are just some of the many, many use cases of the automatic street light system.

Benefits in These Use Cases:

Energy Efficiency: Reduces electricity wastage by operating only when required.

Cost Savings: Minimizes maintenance and manual operation costs.

Safety: Provides reliable lighting to prevent accidents and enhance security.

Environmentally Friendly: Often paired with solar power for sustainable use.

SOFTWARE USED

PROTEUS powerful design and simulation software that allows engineers and developers to create and test circuits before physically building them. It combines circuit simulation with PCB design tools, making it ideal for electronics projects.

Key Features of Proteus

Circuit Design and Simulation:

- Provides an intuitive environment for designing circuits.
- Simulates the behavior of electronic components to test
- functionality.

For our project, it was used to design and test the automatic streetlight circuit.

Microcontroller Integration:

- Allows simulation of microcontrollers like Arduino, PIC, and
- AVR.

You could load your Arduino code (in HEX format) into Proteus to simulate how your streetlight would respond in real-time.

Virtual Components:

- Includes libraries of virtual components like LEDs, resistors, transistors, and LDRs, which we used in our design.
- Enables testing how each component interacts before assembling them physically.

Debugging Tools:

- Detects issues in circuit functionality and displays errors, helping you identify and fix problems before building the hardware.

ARDUINO: open-source electronics platform based on easy-to-use hardware and software. It's mostly used for building electronic projects that require automation and control. In our project, Arduino played a vital role in automating the streetlight system.

Key Features of Arduino

Microcontroller Board:

- Your Arduino board acts as the "brain" of the project, processing inputs from the LDR and controlling the LEDs.
- You likely used an Arduino Uno, which is one of the most common boards for beginners and intermediate projects.

□□ Programming with Arduino IDE□

- The Arduino is programmed using the Arduino IDE □Integrated Development Environment), where we wrote a simple code to read LDR values and switch the LEDs on or off.

□□ Flexibility and Compatibility:

- Arduino is compatible with various sensors and components, making it ideal for our automatic streetlight project.
- It communicates seamlessly with the LDR to detect light intensity and respond accordingly.

□□ Power Efficiency:

- The Arduino's low power consumption made it a good fit for projects like ours that may operate continuously.

PROGRAMMING LOGIC (code description and it's purpose)

Structure and Detailed Code Description:

□□ Include Necessary Libraries

```
cpp
CopyEdit
#include <LiquidCrystal.h>
```

- **Purpose:** The `LiquidCrystal` library is included to manage communication with the 16×2 LCD display.
- **Functionality:** Provides functions to initialize and display messages on the LCD.

🖥️ LCD and LDR Setup

```
cpp
CopyEdit
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
#define LDR_PIN A0
```

- **LCD Initialization:**
  - Pins 7, 8, 9, 10, 11, and 12 are connected to the RS, Enable, D4, D5, D6, and D7 pins of the LCD, respectively.
  - The LCD is used to display real-time information such as the LDR value and the status of the lights.
- **LDR Connection:**
  - The LDR (Light Dependent Resistor) is connected to **Analog Pin A0**.

The LDR measures ambient light intensity, returning an analog value between 0 and 1023.

🔹 Define LED Pins

```
cpp
CopyEdit
const int ledPins[] = {2, 3, 4, 5, 6, 7};
const int ledCount = 6;
```

- **Purpose:** An array stores the digital pins connected to the LEDs, representing the streetlights.
- **ledCount:** Indicates the total number of LEDs used in the project for modularity.

🔹 Setup Function

```
cpp
CopyEdit
void setup() {
    pinMode(LDR_PIN, INPUT);
    for (int i = 0; i < ledCount; i++) {
        pinMode(ledPins[i], OUTPUT);
    }

    lcd.begin(16, 2);
    lcd.print("Street Light Sys");
    delay(2000);
    lcd.clear();
}
```

- **LDR Pin Setup:**Configured as an input to read light levels.
- **LED Pin Setup:**All LED pins are configured as outputs for turning the lights on or off.
- **LCD Initialization:**
  - `lcd.begin(16, 2)` → Sets the dimensions of the LCD (16 columns x 2 rows).
  - Displays the message "Street Light Sys" during startup for 2 seconds, then clears the screen.

🔁 Main Loop

```
cpp
CopyEdit
void loop() {
    int ldrValue = analogRead(LDR_PIN);
```

- **Read LDR Value:** The `analogRead` function retrieves the LDR value, which represents the light intensity.

🔁 Display LDR Value

```
cpp
CopyEdit
lcd.setCursor(0, 0);
lcd.print("LDR Value: ");
lcd.print(ldrValue);
lcd.print("   ");
```

- **Purpose:** The LCD displays the real-time LDR value on the first row.
- **Details:** The extra spaces (`"  "`) clear any previous digits to ensure clean updates.

🔦 Control LEDs Based on LDR Reading

```
cpp
CopyEdit
if (ldrValue < 500) {
    lcd.setCursor(0, 1);
    lcd.print("Lights: ON      ");
    for (int i = 0; i < ledCount; i++) {
        digitalWrite(ledPins[i], HIGH);
    }
} else {
    lcd.setCursor(0, 1);
    lcd.print("Lights: OFF     ");
    for (int i = 0; i < ledCount; i++) {
        digitalWrite(ledPins[i], LOW);
    }
}
```

- **Threshold Check:**
  - A threshold of `500` is used to distinguish between day (light) and night (darkness).
  - You can adjust this value based on environmental lighting conditions.
- **Lights ON Condition:**
  - If the LDR value is below 500 (darkness), "Lights: ON" is displayed on the second row of the LCD.
  - All LEDs in the array are turned on using `digitalWrite(pin, HIGH)`.
- **Lights OFF Condition:**
  - If the LDR value is above 500 (sufficient light), "Lights: OFF" is displayed on the second row.
  - All LEDs are turned off using `digitalWrite(pin, LOW)`.

⏱️ Delay for Stability

```
cpp
CopyEdit
delay(500);
```

- **Purpose:** Adds a 500ms delay to stabilize LDR readings and prevent flickering of the LEDs and LCD display.

Step-by-Step Circuit Description

breakdown of how the circuit components are connected:

Main Components

- **Microcontroller** Arduino board serves as the brain of the circuit.
- **LDR Light Dependent Resistor)** Connected to an analog input pin on the Arduino to measure ambient light levels.
- **LCD Liquid Crystal Display)** Displays the LDR reading and LED state ON/OFF.

LEDs Represent street lights, controlled by the Arduino.

Connections

LDR Connection:

- One terminal of the LDR is connected to the analog input pin (
- `A0`).

The other terminal is connected to **5V** through a pull-up resistor.

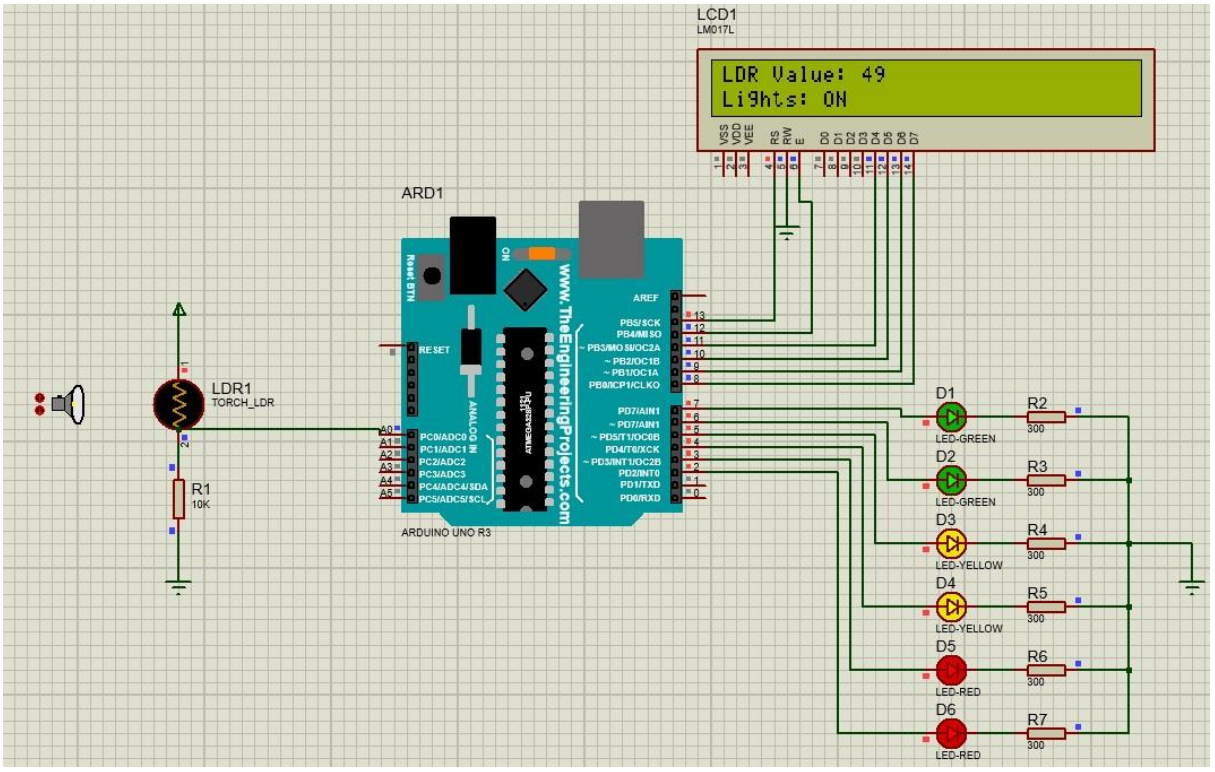
LCD Connection:

- The LCD pins (RS, Enable, D4–D7) are connected to digital pins 7, 8, 9, 10, 11, and 12 on the Arduino.
- The **VSS** (ground) and **VDD** (power) pins of the LCD are connected to the ground and 5V of the Arduino.
- **Contrast Pin** (`V0`): Connected to a potentiometer to adjust display contrast.

LEDs:

- Each LED is connected to a digital output pin (pins 2–7).
- The other terminal of each LED is connected to the ground through current-limiting resistors.

SIMULATION DIAGRAM FROM PROTEUS



ACTUAL CODES USED FOR THE MODEL

```
#include <LiquidCrystal.h>

// Initialize the LCD (RS, E, D4, D5, D6, D7)
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

#define LDR_PIN A0  // LDR connected to Analog Pin A0

// LED Pins
const int ledPins[] = {2, 3, 4, 5, 6, 7};  // LEDs connected
const int ledCount = 6; // Total number of LEDs

void setup() {
    pinMode(LDR_PIN, INPUT);

    // Initialize LED pins as output
    for (int i = 0; i < ledCount; i++) {
        pinMode(ledPins[i], OUTPUT);
```

```
    }

    // Initialize LCD
    lcd.begin(16, 2);
    lcd.print("Street Light Sys");
    delay(2000);
    lcd.clear();
}

void loop() {
    int ldrValue = analogRead(LDR_PIN);  // Read LDR value

    lcd.setCursor(0, 0);
    lcd.print("LDR Value: ");
    lcd.print(ldrValue);
    lcd.print("  "); // Clear extra digits

    if (ldrValue < 500) {  // Adjust threshold based on enviro
        lcd.setCursor(0, 1);
        lcd.print("Lights: ON      ");

        // Turn ON LEDs
        for (int i = 0; i < ledCount; i++) {
            digitalWrite(ledPins[i], HIGH);
        }
    } else {
        lcd.setCursor(0, 1);
        lcd.print("Lights: OFF     ");

        // Turn OFF LEDs
        for (int i = 0; i < ledCount; i++) {
            digitalWrite(ledPins[i], LOW);
        }
    }

    delay(500);  // Small delay for stable readings
}
```

WORKING OF THE SYSTEM

How the system detects light intensity: The system detects light intensity using a **Light Dependent Resistor (LDR)**, a sensor that changes it's resistance based on the light levels it recieves:

- **High Light Intensity (Daytime):** The LDR's resistance decreases significantly, allowing more current to flow through the circuit.
- **Low Light Intensity (Nighttime):** The LDR's resistance increases, reducing the current flow.
- The behavior of the system is determined by how this change in resistance influences the voltage across certain points in the circuit, which is monitored by the microcontroller (Arduino).

SYSTEM BEHAVIOR

Daytime (High Light Intensity):

- **LDR Resistance:** Low (minimal resistance).
- **Voltage Across LDR** Low.
- **LED Behavior:** Off (no need for artificial light).

Reason: The ambient light is enough to illuminate the area.

Nighttime (Low Light Intensity):

- **LDR Resistance:** High (maximum resistance).
- **Voltage Across LDR** High
- **LED Behavior:** The ambient light is too low to illuminate the area, so system compensates with the LED.

How the Arduino Monitors and Controls:

 Analog Read: The arduino reads the voltage across the LDR uing it's **analog pin**.

 Threshold Comparison: The voltage is compared to a preset threshold.

- If the voltage is **below the threshold**, it's daytime, and the LED
- remains OFF.

If the voltage is **above the threshold**, it's nighttime, and the LED is turned
ON.

 Digital write: The Arduino sends a signal to control the LED based on the result of the comparison.

This cycle runs continuously, ensuring the system responds dynamically to changing light conditions.

CHALLENGES FACED

 Sensor Calibration: The LDR sensor should be calibrated properly to avoid false triggering (e.g., street lights turning on when there's artificial light).

 Weather Conditions: Changes in weather, such as heavy rain or fog, may affect the performance of sensors and overall system reliability.

 System Reliability: The automatic street light system must be robust to handle long periods of use, especially under adverse environmental conditions.

Advantages:

- **Automatic Operation:** The system works without human intervention, saving time and effort.
- **Energy Saving**: The automatic switching reduces the chances of lights staying on unnecessarily during the day.
- **Cost Reduction**: By preventing wastage of energy, the system contributes to lowering the operational costs of street lighting.

RESULTS AND OBSERVATIONS

 Automatic Light Detection and Control:

The system successfully detects changes in light intensity using the LDR and automatically turns the LED on or off based on the ambient light level.

- Daytime: The LED remains off when the light intensity is high.
- Nighttime The LED turns on when the light intensisty drops.

Limitations

If the ambient light intensity fluctuates (it might be new to moving shadows) the LED may flicker.

REAL-WORLD APPLICATION

- HIght ways
- Urban areas

CONCLUSION

This project has shown how we can build a simple and effective automatic streetlight system using an LDR sensor. The system works by turning the lights on when it gets dark and turning them off during the day, which makes it easy to use and saves energy.

Using Proteus, I was able to simulate how the components like the LDR, LEDs, and LCD work together to make the system function properly. It's a good example of how automation can make everyday tasks more convenient and reliable.

Overall, this project has helped me understand how to apply basic electronics and programming skills to solve real-life problems. It's a great starting point for future projects, like adding more features or improving the design for use in larger systems.