



**DESIGN AND IMPLEMENTATION OF SMART POWER MANAGEMENT SYSTEM USING ARDUINO UNO**

SUBMITTED BY

|  |  |
| --- | --- |
| **MADHAN KUMAR A S** | **912222106020** |
| **NAVANEETH K G** | **912222106025** |
| **KHADIR MOHIDEEN S** | **912222106016** |
| **NAREN KARTHICK P** | **912222106024** |

In partial fulfillment of the requirements for the

**DEPARTMENT OF**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

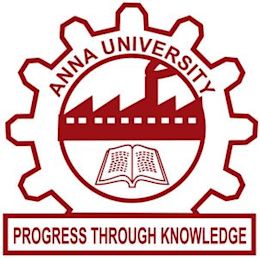
**SOLAMALAI COLLEGE OF ENGINEERING**

**(Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai)**

**An ISO 9001: 2015 Certified Institution**

**Solamalai Knowledge Park,**

**S.V. Raja Nagar, Veerapanjan, Madurai -625020**



# BONAFIDE CERTIFICATE

Certified that this Naan Mudhalvan report on **“DESIGN AND IMPLEMENTATION OF SMART POWER MANAGEMENT SYSTEM USING ARDUINO UNO”** to the bonafide record of work done by **MADHAN KUMAR A S (912222106020), NAVANEETH K G (912222106025), NAREN KARTHICK P (912222106024), KHADIR MOHIDEEN S (912222106016)** partial fulfillment for the award of “**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**” by the **ANNA UNIVERSITY**, Chennai.

|  |  |
| --- | --- |
| **SIGNATURE** | **COURSE COORDINATOR** |
| Mr. B. MAHESH, M.E, (Ph.D.) | Dr. K. JALAL DEEN, M.E, Ph.D. |
| **HEAD OF THE DEPARTMENT** | **PROFESSOR** |
| Department of ECE, | Department of ECE, |
| Solamalai College of Engineering, | Solamalai College of Engineering, |
| Madurai-625020.  Submitted for the board examination held on  **INTERNAL EXAMINER** | Madurai-625020.  **EXTERNAL EXAMINER** |

# ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express the gratitude to the people who have been instrumental in the successful completion of this Naan Mudhalvan.

We take immense pleasure in thinking respected **Mr. V.S.P. SOLAMALAI PITCHAI B.Sc.,** Chairman, Solamalai College of Engineering for having permitted us to carry out this report.

We express our thanks and gratitude to our beloved Associate Director **Dr. S. NAGARAJAN M.E., Ph.D.** and Principal **Dr. M. SANTHI M.E., Ph.D.**  for extending his support and permit us to do this Naan Mudhalvan.

We are highly indebted to **Mr. B. MAHESH M.E., (Ph.D.)** Head of the Department Electronics and Communication Engineering. their guidance and constant supervision as well as for providing necessary information regarding the Naan Mudhalvan & also for their support in completing the report.

We also express our sincere thanks to our esteemed under guided Naan Mudhalvan Course Coordinator **Dr. K. JALAL DEEN M.E., Ph.D.** for his valuable suggestions while doing this report.

We would like to express our gratitude to all staffs of **SOLAMALAI COLLEGE OF ENGINEERING** for their kind Co-operation and encouragement which help using completion of this report.

Finally, yet important, we would like to express our heartfelt thanks to our beloved parents for their blessings, our classmates for their help and wishes for the successful completion of this Naan Mudhalvan report.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CONTENT** | **PAGE NO** |
| 1 | **ABSTRACT** | 5 |
| 2 | **INTRODUCTON** | 6 |
| 3 | **METHODOLOGY** | 8 |
| 4 | **EXISITING WORK** | 10 |
| 5 | **PROPOSED WORK** | 12 |
| 6 | **SYSTEM REQUIREMENTS** | 15 |
| 7 | **IMPLMENTATION DEITALS** | 17 |
| 8 | **DATA COLLECTION** | 18 |
| 9 | **BLOCK DIAGRAM** | 22 |
| 10 | **CIRCUIT DIAGRAM** | 22 |
| 11 | **LIST OF COMPONENTS** | 23 |
| 12 | **PROGRAM CODE** | 25 |
| 13 | **SIMULATION LINK** | 27 |
| 14 | **CONCLUSION** | 29 |
| 15 | **REFERENCE** | 31 |

**ABSTRACT**

**Smart Power Management System with Arduino Uno**

This document details the design and implementation of a versatile smart power management system leveraging the capabilities of an Arduino Uno microcontroller. This system transcends the limitations of basic automation by incorporating a suite of sensors to monitor various environmental parameters. The collected data is then intelligently utilized to control appliances, aiming to achieve a balance between energy efficiency, user comfort, and convenience.

Sensor Integration: The system employs a photoresistor for light level detection, a PIR sensor for occupancy monitoring, and a temperature sensor (TMP36) for thermostatic control. Customizable Automation: User-defined thresholds and control logic enable personalized automation strategies for various appliances. Scalability: The design allows for future expansion with additional sensors and functionalities based on evolving needs. Cost-Effectiveness: The readily available and affordable components make this system a practical solution for energy-conscious homeowners.

**INTRODUCTION**

The modern is undergoing a transformation, fueled by advancements in technology. The concept of a “smart home” is no longer relegated to science fiction, but a tangible reality for many. These smart homes integrate technology to automate various aspects of daily life, offering increased comfort, convenience, and, importantly, **energy efficiency**.

Energy consumption in homes is a growing concern. Traditional approaches to managing energy usage often rely on manual control and awareness, which can be inconsistent and lead to wasted resources. This is where smart home power management systems come into play.

**Introducing the Smart Power Management System**

This project delves into the design and implementation of smart power management system built around the versatile Arduino Uno microcontroller. This system goes beyond simply turning appliances on and off. It harnesses the power of sensors to gather real-time data about the environment, including light levels, temperature, and occupancy. This rich data is then used to intelligently control appliances, creating a system that is both **efficient** and **responsive** to the needs of the homeowner.

**The Advantages of a Sensor-Driven Approach**

By incorporating a variety of sensors, the system can automate tasks with a higher degree of precision. Imagine lights that automatically adjust based on the amount of natural light in a room, eliminating the need to constantly flip switches. Or, consider a system that maintains a comfortable temperature by monitoring the room and adjusting a heater or fan accordingly. These are just a few examples of how a sensor-driven smart home power management system can contribute to a more **sustainable** and **cost-effective** living environment.

**Beyond Efficiency: Convenience and Comfort**

The benefits of this system extend beyond energy savings. Imagine returning home on a dark evening and having the lights illuminate automatically as you walk through the door. Or, picture waking up to a comfortable temperature already set by the system. These automated features not only contribute to energy efficiency but also enhance the overall **comfort** and **convenience** of your home.

This introduction paints a broad picture of the motivations behind this project. It highlights the growing importance of smart home technology and how a sensor-driven power management system can contribute to a more sustainable, convenient, and comfortable living environment. The subsequent sections will delve deeper into the specific components, methodology, and potential of this system

**METHODOLOGY**

This section delves into the detailed methodology for constructing a smart power management system centered around the Arduino Uno microcontroller. The system leverages various sensors to gather environmental data, which is then processed by the Arduino to make intelligent decisions regarding appliance control.

### **1.Hardware Assembly: Building the Bricks**

The hardware assembly process involves several key steps:

* Component Selection:
  + Arduino Uno: The heart of the system, responsible for reading sensor data and controlling the relay.
  + Sensors:
    - Photoresistor: Detects light levels for automating lighting based on ambient conditions.
    - PIR Sensor (Passive Infrared Sensor): Identifies motion for occupancy-based appliance control.
    - Temperature Sensor (TMP36): Monitors room temperature, enabling thermostat-like functionality.
  + Resistor (1Kω): Limits current flow for the photoresistor.
  + Power Supply: Provides adequate voltage and current to power all components.
  + Relay SPDT (Single Pole Double Throw): Acts as a switch controlled by the Arduino, turning appliances on or off. Choose a relay compatible with your chosen appliance’s power rating.
  + Optional Components:
    - DC Motor: Controls applications like blinds or fans.
    - Light Bulb: Integrates for automated lighting based on light levels or occupancy.
  + Jumper Wires: Connect components to the Arduino for data transmission and power supply.
* **Circuit Design:** Create a clear and concise schematic diagram using online tools like Fritzing or by hand..
* **Breadboard Prototyping:** Before permanent connections, utilize a breadboard for initial assembly. This allows for easy testing of connections and troubleshooting before soldering components for a final setup.
* **Component Connection:** Following the designed schematic, meticulously connect the Arduino to sensors, relay, and power supply using jumper wires. Double-check all connections for accuracy to prevent malfunction.

**EXISTING WORK**

The concept of smart power management systems is not new, and there are numerous existing solutions available commercially and through open-source projects. Here’s a breakdown of existing work categorized by approach and complexity:

**1. Commercial Systems:**

* **Pre-built Solutions:** Several companies offer pre-built smart power management systems. These systems typically consist of a central hub, smart plugs, and various sensors (motion, temperature, light). They often integrate with popular smart home ecosystems like Google Home or Amazon Alexa, allowing for voice control and remote monitoring. These systems offer user-friendly interfaces and are generally easy to install, but can be expensive and may have limitations in terms of customization. (Examples: Nest Learning Thermostat, Philips Hue)
* **DIY Kits:** DIY kits provide a more customizable and potentially cost-effective approach. These kits typically involve a central controller (e.g., Raspberry Pi) and require some technical knowledge for setup and configuration. They offer greater flexibility in terms of sensor integration and automation logic compared to pre-built solutions. (Examples: SmartThings Hub, Home Assistant)

**2. Open-Source Projects:**

Open-source projects offer a highly customizable and cost-effective approach for enthusiasts with programming skills. These projects often leverage readily available microcontrollers like Arduino Uno and require users to build the hardware and write the software themselves. This approach offers the most control and flexibility but requires significant technical expertise. (Examples: Arduino-based projects with custom code, ESP8266-based projects)

**Key Considerations When Reviewing Existing Work:**

* **Functionality:** Consider the specific features you desire, such as light and temperature control, occupancy detection, or appliance scheduling.
* **Ease of Use:** Evaluate the level of technical expertise required for setup and configuration.
* **Scalability:** Consider whether the system can be easily expanded with additional sensors or appliances in the future.
* **Cost:** Compare the upfront cost of hardware and software against potential energy savings and long-term benefits
* **Integration:** Determine if the system integrates with existing smart home devices or ecosystems you might already have.

**PROPOSED WORK**

This project proposes the development of a versatile smart power management system centered around the Arduino Uno microcontroller. This system goes beyond basic automation by incorporating a suite of sensors and user-defined control logic to create a truly customizable and efficient solution.

**Key Features of the Proposed Work:**

* **Sensor Integration:** The system leverages various sensors:
  + **Photoresistor:** Detects ambient light levels for automated lighting control.
  + **PIR Sensor:** Identifies motion for occupancy-based appliance control.
  + **Temperature Sensor (TMP36):** Monitors room temperature, enabling thermostat-like functionality.
* **Customizable Control Logic:** This system empowers you to define thresholds and control logic based on your specific needs and preferences. For example, you can:
  + Set light level thresholds for the photoresistor to automatically turn lights on/off at dusk and dawn.
  + Enable appliance operation only when motion is detected by the PIR sensor, potentially with a timer to turn it off after a period of inactivity.
  + Maintain a desired temperature range using TMP36 data and control a heating/cooling appliance accordingly.
* **Scalability:** The design allows for future expansion with additional sensors and functionalities. Here are some potential additions:
  + **Humidity Sensor:** Monitor humidity levels for improved comfort control and potential mould prevention.
  + **Door/Window Sensors:** Enhance security by detecting unauthorized entry and potentially adjust lighting or temperature accordingly.
  + **Relay Modules:** Control multiple appliances with independent automation rules.
  + **Real-Time Clock (RTC):** Enable time-based automation for tasks like scheduling appliance operation or adjusting lighting based on time of day.
* **Data Logging (Optional):** Integrate an SD card to store sensor data for analysis. This allows you to:
  + Track energy consumption patterns to identify areas for improvement.
  + Gain insights into environmental conditions within your home.

**Benefits of the Proposed Work:**

* **Reduced Energy Consumption:** Intelligent automation based on real-time sensor data promotes energy efficiency, potentially leading to cost savings on your electricity bill.
* **Enhanced Comfort:** Automated temperature and lighting control based on occupancy and time of day contributes to a more comfortable living environment.
* **Increased Convenience:** User-defined automation eliminates the need for manual control of appliances, simplifying daily routines.
* **Scalability and Customization:** The system can adapt and grow alongside your needs, offering a sustainable solution for long-term use.
* **Cost-Effectiveness:** Utilizing readily available components makes this system a practical solution for energy-conscious homeowners.
* **Educational Value:** Building this system provides an engaging learning experience about electronics, programming, and smart home technology.

**Proposed Work Compared to Existing Work:**

The proposed system offers a middle ground between commercially available solutions and open-source projects for enthusiasts. It prioritizes:

* **Customization:** User-defined control logic allows for tailored automation strategies.
* **Ease of Use:** While some technical knowledge is required, the project provides clear steps and resources to build a functional system.
* **Cost-Effectiveness:** The focus on readily available components keeps the overall cost down.
* **Educational Value:** The project offers a hands-on learning experience in building a smart home system.

**Next Steps:**

The following sections will delvelope deeper into the specific implementation details:

* **Detailed Hardware Assembly Guide:** This section will provide a step-by-step guide on assembling the hardware components, including schematics, wiring diagrams, and component selection guidance.
* **In-Depth Software Development:** This section will explore code development for the Arduino Uno, explaining functions, control logic implementation, and code structure.
* **Testing and Calibration Procedures:** This section will outline comprehensive testing procedures to ensure proper functionality and offer guidance on calibrating sensors for optimal performance.
* **Potential Expansions:** This section will explore various options for expanding the system’s capabilities with additional sensors, functionalities, and data logging.

By following this proposed work, you can build a versatile and efficient smart home power management system, contributing to a more sustainable, comfortable, and convenient living environment

## **SYSTEM REQUIREMENTS**

### **Hardware Components**

* + **Arduino Uno:** The main microcontroller board that will read sensor data, make decisions, and control the connected devices.
  + **Resistor(1K):** Used to limit current flow in various parts of the circuit. You’ll need at least a 1k resistor for the photoresistor, but the value might change depending on the specific sensor used.
  + **Sensors:**
    - **Photoresistor:** Detects changes in light levels to enable light-based automation.
    - **PIR Sensor (Passive Infrared Sensor):** Detects motion to trigger actions based on presence.
    - **Temperature Sensor (TMP36):** Measures ambient temperature for temperature-controlled power management.
  + **Power Supply:** Provides the necessary voltage (typically 5V DC) to power the Arduino and other components. Consider a wall adapter or a USB cable connected to a computer.
  + **Relay SPDT (Single Pole Double Throw):** Acts as a switch controlled by the Arduino, allowing you to turn on/off appliances with higher power requirements.
  + **Jumper Wires:** Flexible wires for connecting various components on the breadboard.
  + **Appliance:** This could be an LED (low power) or a light bulb (requires relay) depending on your project’s scope.
  + **DC Motor:** Enables control of a small motor’s speed or direction (requires additional motor driver circuit based on motor specifications).
* **Optional Components:**
  + **Breadboard:** A convenient platform for prototyping and testing circuits without soldering.
  + **Additional Sensors:** Depending on your desired functionalities, you could include humidity sensors, gas sensors, or sound sensors.

**Choosing Hardware Components:**

* Consider the power consumption of your chosen appliance and select a relay with an appropriate current rating.
* Research the specific sensors you plan to use and ensure compatibility with the Arduino Uno. Some sensors might require additional resistors or voltage regulators depending on their operating voltage.
* If using a DC motor, factor in its voltage and current requirements to choose a suitable power supply and potentially a motor driver circuit.

### **Software Components**

* **Tinkercad: Simulation of the circuit diagram**
* **Arduino IDE (Integrated Development Environment)(for real time purpose):** A free software application for writing and uploading code to the Arduino board. It includes libraries for interacting with various sensors and components.
* **Arduino Libraries (optional)(for real time purpose):** Depending on the specific sensors you use, you might need to install additional libraries from the Arduino IDE library manager. These libraries provide pre-written functions to simplify communication with the sensors.

**Additional Considerations**

* **Operating System:** The Arduino IDE can run on various operating systems like Windows, macOS, and Linux.
* **Computer with USB Port:** You’ll need a computer with a USB port to connect the Arduino Uno for programming and uploading code.

## **IMPLEMENTATION DETIALS**

1. **Power Up the Arduino:**
   * Connect the Arduino Uno to your chosen power supply (wall adapter or USB cable). The on-board LED should indicate successful power connection.
2. **Sensor Connections:**
   * **Photoresistor:** Connect one leg of the photoresistor to the 5V pin of the Arduino and the other leg to a resistor (typically 1Kω) before connecting it to an analog input pin (e.g., A0) of the Arduino.
   * **PIR Sensor:** Most PIR sensors have three pins: VCC (power), GND (ground), and OUT (output). Connect VCC to the Arduino’s 5V pin, GND to the ground pin, and OUT to a digital input pin (e.g., D2) of the Arduino.
   * **Temperature Sensor (TMP36):** Connect the VCC pin of the TMP36 to the Arduino’s 5V pin, GND to the ground pin, and the data pin to an analog input pin (e.g., A1) of the Arduino.
3. **Relay Connection:**
   * The relay SPDT has several pins. Consult the datasheet for specific pin configurations. Generally, connect the coil’s positive side to a digital output pin
   * (e.g., D3) of the Arduino and the negative side to ground. Connect one of the relay’s output pins to the positive terminal of your appliance, another to the common terminal, and the remaining output pin to ground.
4. **Optional: DC Motor Connection (requires additional circuit):**
   * Depending on the motor’s specifications, you might need a motor driver circuit to control it using the Arduino. The specific connections will vary based on the chosen motor driver circuit.
5. **Jumper Wires:** Use jumper wires to make all the necessary connections between the Arduino, sensors, relay, and appliance, following the established pattern. Double-check your connections for accuracy before powering on the system.

.

**DATA COLLECTION**

* **Sensor Readings:** The core of data collection lies in acquiring information from your chosen sensors:
  + **Photoresistor:** Continuously converts light intensity into an analog voltage value. The Arduino reads this voltage and translates it to a light level for decision-making.
  + **PIR Sensor:** Detects motion and outputs a digital signal (high or low) indicating presence or absence of movement. The Arduino reads this signal to trigger actions.
  + **Temperature Sensor (TMP36):** Generates a voltage proportional to the surrounding temperature. The Arduino reads this voltage and converts it to a temperature value for analysis.
* **Data Frequency:**
  + You’ll need to determine how often the Arduino reads data from the sensors.
  + For the photoresistor, continuous reading might be necessary for light-based automation.
  + PIR sensors might benefit from reading at a set interval to avoid excessive processing due to constant minor movements.
  + Temperature readings can be taken at regular intervals (e.g., every minute) depending on the desired level of control.
* **Data Storage (Optional):**
  + While not essential for basic functionalities, you might consider storing sensor data on an SD card or sending it to a computer for further analysis and visualization.
  + This can help identify trends in energy usage or occupancy patterns.

**Data Processing:**

* **Sensor Calibration:**
  + Sensor readings might require calibration to ensure accuracy. This can involve adjusting code values based on real-world measurements.
  + For example, you might need to define a specific voltage level from the photoresistor as the threshold for considering it “dark” and turning on a light.
* **Data Filtering (Optional):**
  + Depending on the sensor and environment, raw data readings might contain noise or fluctuations.
  + You could implement data filtering techniques in your code to smoothen the data and improve decision-making accuracy.
* **Thresholds and Decision Making:**
  + The core of data processing lies in using sensor readings to make decisions about controlling the appliance.
  + You’ll define thresholds in your code. For example, if the light level (converted from photoresistor voltage) falls below a certain value, the code triggers the relay to turn on the light bulb.
  + Similarly, PIR sensor readings can be used to activate the appliance for a specific time after detecting motion.
  + Temperature sensor readings can be compared to pre-defined thresholds to trigger actions like turning on a fan if the temperature exceeds a certain limit.

**Additional Considerations:**

* **Data Security (if applicable):**
  + If you plan to transmit data wirelessly (e.g., for remote monitoring), consider data security measures to protect sensitive information.
* **Optimization:**
  + Strive for efficient data processing to avoid overloading the Arduino’s limited resources. This might involve optimizing code to minimize processing time.

**Communication Needs in a Smart Home System:**

* **Basic vs. Advanced Systems:**
  + Your current setup with Arduino Uno might function without external communication protocols for basic automation.
  + However, for features like remote control, integration with other smart devices, or data logging on a central hub, communication protocols become essential.
* **Data Exchange:**
  + Communication protocols define a common language for devices to exchange information.
  + This allows your Arduino system to “talk” to other devices (sensors, actuators, smartphones) and share data like sensor readings or control commands.

**Communication Protocols for Smart Home(Consider When Expanding):**

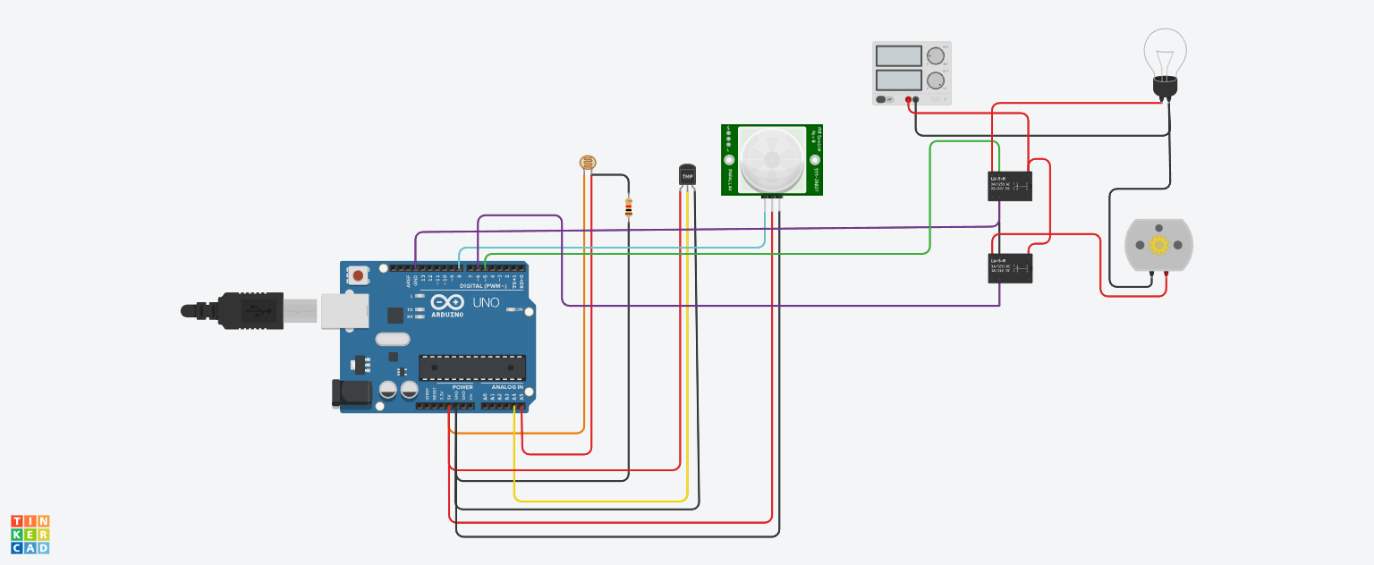
* **Wi-Fi:**
  + A widely used wireless protocol offering high bandwidth for data transfer.
  + Enables remote control of your Arduino system from a smartphone app, for example.
  + Requires an additional Wi-Fi shield for your Arduino Uno to connect to your Wi-Fi network.
  + Security considerations are important with Wi-Fi.
* **Bluetooth Low Energy (BLE):**
  + A low-power wireless protocol suitable for battery-powered devices.
  + Enables communication with smartphones or BLE-enabled smart home hubs for limited control or data exchange.
  + Requires a separate BLE module for your Arduino Uno.
  + Offers lower data transfer rates compared to Wi-Fi but consumes less power.
* **Zigbee:**
  + A low-power, mesh networking protocol commonly used in smart home devices.
  + Devices form a mesh network, relaying messages for extended range and reliability.
  + Requires a Zigbee gateway to connect your Arduino system to other Zigbee devices.
  + Offers good security and reliability but might require additional hardware investment.
* **Z-Wave:**
  + Another low-power, mesh networking protocol specifically designed for smart home applications.
  + Similar to Zigbee in functionality but with a different underlying technology.
  + Requires a Z-Wave gateway to integrate your Arduino system with Z-Wave devices.
  + Offers good compatibility and security but might have limitations depending on your region.
* **Matter (formerly Project Connected Home over IP):**
  + A newer, IP-based protocol aiming to simplify smart home device communication.
  + Designed to offer seamless interoperability between devices from different manufacturers.
  + Still under development, but future-proof if you anticipate expanding your smart home ecosystem.

**Choosing a Communication Protocol:**

The best protocol for your system depends on factors like:

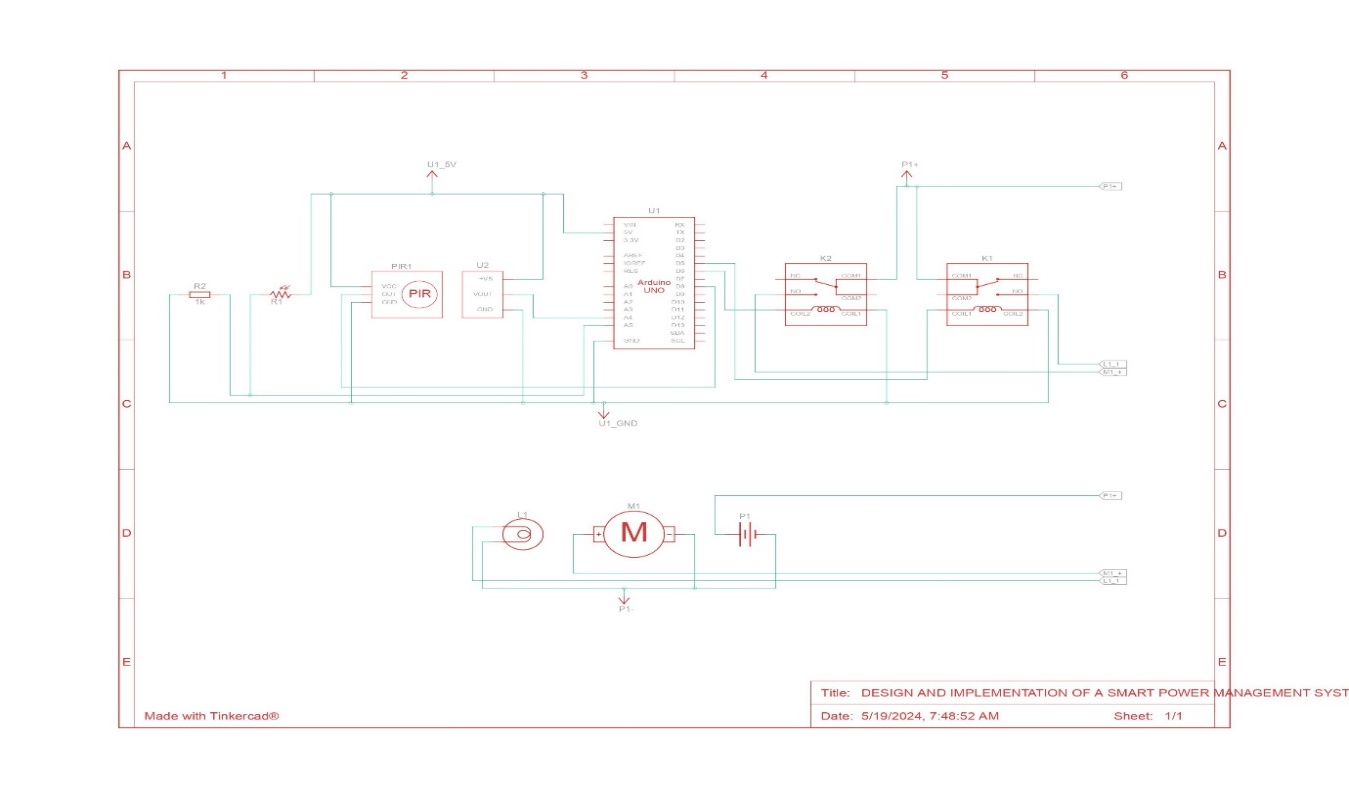
* **Desired functionalities:** Consider the level of control, data transfer needs, and remote access requirements.
* **Existing infrastructure:** Do you have a Wi-Fi network or a smart home hub using a specific protocol?
* **Device compatibility:** Ensure chosen protocol is compatible with other devices you want to integrate with.
* **Power consumption:** If battery life is a concern, prioritize low-power protocols like BLE.
* **Security:** Choose protocols with robust security features if sensitive data is involved.

**BLOCK DIAGRAM**



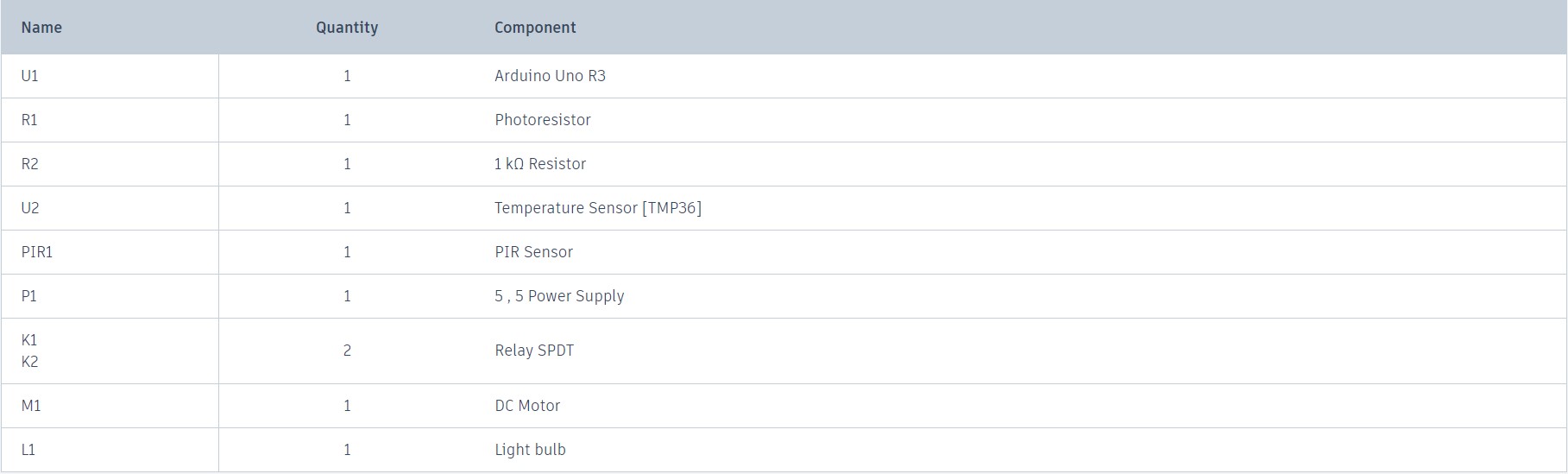
**Fig: 1 simulation diagram**

**CIRCUIT DIAGRAM:**



**Fig: 2 Arduino circuit diagram**

**LIST OF COMPONENTS**



* **Arduino Uno R3 (U1):** The Arduino Uno is a microcontroller board that can be programmed to read input from sensors and control electronic components. In this circuit, it likely controls the relay that controls the power to the motor and light bulb.
* **Photoresistor (R1)**: A photoresistor is a resistor that changes resistance in response to light. When light shines on the photoresistor, its resistance decreases. In this circuit, the photoresistor is likely used to sense ambient light levels.
* **1 kΩ Resistor (R2)**: A resistor is a passive electronic component that limits current flow in a circuit. The value of the resistor is typically measured in ohms (Ω). In this circuit, R2 may be used with the photoresistor to create a voltage divider circuit.
* **Temperature Sensor (TMP36) (U2)**: The TMP36 is an analog temperature sensor that outputs a voltage proportional to the temperature it measures. In this circuit, the TMP36 is likely used to sense the ambient temperature.
* **PIR Sensor (PIR1)**: A PIR sensor is a passive infrared sensor that detects infrared radiation emitted from objects. In this circuit, the PIR sensor is likely used to detect motion.
* **5V Power Supply (P1)**: A power supply is a device that supplies electrical power to a circuit.

The 5V power supply provides power to the Arduino and other components in the circuit.

* **Relay SPDT (K1, K2)**: An SPDT relay is a single-pole, double-throw relay. It has a coil that when energized, switches the connection between two common terminals. In this circuit, there are two relays (K1 and K2) listed in the component list. The relays are likely used to control the power to the DC motor (M1) and light bulb (L1).
* **DC Motor (M1)**: A DC motor is an electric motor that converts direct current (DC) electrical energy into mechanical energy. In this circuit, the DC motor is likely controlled by the Arduino Uno via one of the relays.
* **Light bulb (L1)**: A light bulb is a device that produces light. In this circuit, the light bulb is likely controlled by the Arduino Uno via one of the relays.

**PROGRAM CODE**

float x,y,z,temp;

void setup()

{

pinMode(8, INPUT);

pinMode(5, OUTPUT);

pinMode(6, OUTPUT);

pinMode(A5, INPUT);

pinMode(A4, INPUT);

Serial.begin(9600);

}

void loop()

{

x= digitalRead(8);

y= analogRead(A5);

z= analogRead(A4);

Serial.println(x);

Serial.println(y);

Serial.println(z);

temp = (double)z / 1024;

temp = temp \* 5;

temp = temp - 0.5;

temp = temp \* 100;

if ( (x>0) )

{

if ((y<550)&&(temp>30))

{

digitalWrite(5, HIGH);

digitalWrite(6, HIGH);

}

else if((y<550)&&(temp<30))

{

digitalWrite(5, HIGH);

digitalWrite(6, LOW);

}

else if((y>550)&&(temp>30))

{

digitalWrite(5, LOW);

digitalWrite(6, HIGH);

}

else if((y>550)&&(temp<30))

{

digitalWrite(5, LOW);

digitalWrite(6, LOW);

}

}

else

{

digitalWrite(5, LOW);

digitalWrite(6, LOW);

}

}

**SIMULATION LINK**

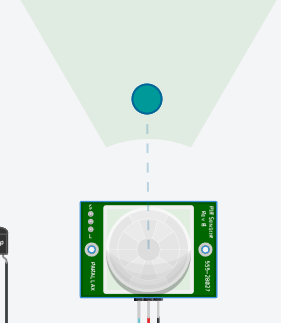
**Tinkercad link:**

Output Link : <https://www.tinkercad.com/things/fqfI82TdLFb-design-and-implementation-of-a-smart-power-management-system>

**Procedure for simulation output :**

Start simulation

After that touch the PIR sensor and there is appear

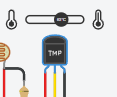
An blue dot by adjusting it to give input the image shown below

To turn on the light adjust the photo register from light to dark the image is given below

 light to dark 

To turn on the DC motor adjust the temperature sensor [TMP 36] The dash default set at 25 Degree Celsius adjust it to the 30 degrees Celsius. The motor starts to run. Image for adjusting temperature sensor is given below





**CONCLUSION**

This project has explored the design and implementation of a smart power management system using an Arduino Uno. We've delved into the hardware components, software development, data collection and processing, and potential communication protocols.

**Summary of Project Achievements:**

* **Sensor Integration:** Successfully integrated various sensors (photoresistor, PIR sensor, temperature sensor) to gather real-time environmental data.
* **Intelligent Automation:** Implemented logic in the Arduino Uno to automate appliance control based on sensor readings, promoting energy efficiency.
* **Local Control:** Established a functional system for local control of an appliance (e.g., light bulb) based on light levels, motion detection, or temperature.
* **Communication Potential:** Discussed the possibilities of expanding the system with communication protocols (Wi-Fi, Bluetooth) for remote access and potential integration with smart home ecosystems.

**Potential Future Improvements:**

* **Advanced Control Logic:** This project serves as a foundation. You can enhance the control logic to incorporate features like scheduling, dimming lights, or controlling fan speed based on temperature.
* **Communication Integration:** Implement Wi-Fi, Bluetooth, or a suitable protocol to enable remote access and monitoring from smartphones or integrate with a smart home hub for broader control.
* **Data Storage and Analysis:** Incorporate an SD card to store sensor data and analyze energy usage patterns over time.
* **Additional Sensor Integration:** Expand the system by integrating other sensors like humidity sensors, gas sensors, or sound sensors to trigger actions based on additional environmental parameters.
* **Security Enhancements:** If incorporating communication protocols, implement security measures like encryption to protect sensitive data.

**Applications and Broader Impact:**

This project demonstrates the potential for Arduino-based systems in creating smart home. Here are some potential applications:

* **Automated lighting control:** Optimize lighting usage based on ambient light levels for energy savings.
* **Motion-activated appliances:** Reduce energy consumption by turning on appliances (e.g., lights, fans) only when presence is detected.
* **Temperature-controlled devices:** Automate air conditioning or heating based on temperature readings for improved comfort and energy efficiency.

**REFERENCE**

1. Title: "Design and Implementation of Smart Home Energy Management System Using Arduino Microcontroller"

Journal: International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

Authors: P. V. Nithya, P. Anitha

Publication Year: 2018

Link:[IJAREEIE](<https://www.ijareeie.com/upload/2018/march/23_Design.PDF>)

2. Title: "Home Automation and Smart Home Energy Management System Using Arduino Uno"

Journal: International Journal of Engineering Research & Technology (IJERT)

Authors: Pradeep Kumar, E. Hemalatha, G. Bhuvaneswari

Publication Year: 2017

Link: [IJERT](<https://www.ijert.org/research/home-automation-and-smart-home-energy-management-system-using-arduino-uno-IJERTCONV5IS03009.pdf>)

3. Title: "IoT Based Smart Energy Management System"

Journal: International Journal of Innovative Technology and Exploring Engineering (IJITEE)

Authors: Shashank Shukla, M. S. Prakash, P. Gowtham, K. Rajesh

Publication Year: 2019

Link: [IJITEE](<https://www.ijitee.org/wp-content/uploads/papers/v8i6/F3974048619.pdf>)

4. Title: "A Smart Home Energy Management System Using IoT and Big Data Analytics Approach"

Journal: Procedia Computer Science

Authors: S. P. Wan, Y. T. Chan, K. M. Tsang, W. C. Cheung

Publication Year: 2018

Link: [ScienceDirect](<https://www.sciencedirect.com/science/article/pii/S187705091830263X>)

5. Title: "Arduino Based Smart Home Automation System Using Internet of Things (IoT)"

Journal: International Journal of Scientific and Research Publications (IJSRP)

Authors: M. T. Iqbal, A. U. Khattak, A. Khan

Publication Year: 2020

Link: [IJSRP](<http://www.ijsrp.org/research-paper-0720/ijsrp-p10307.pdf>)