Design and Development of an Advanced Load Control System via Arduino

Safaa Najah Saud Al-Humairi

Faculty of Information Sciences and Engineering, Management and Science University, 40100 Shah Alam, Selangor, Malaysia
Email: safaa_najah@msu.edu.my
https://orcid.org/0000-0001-5978-7651

Abstract

Incorporating Internet of Things (IoT) technology into load control systems has changed how we acquire regulators and monitor devices, making it more convenient and safer. The proposed system focuses on designing an intelligent indicator and control system solution for demanding industrial environments where equipment experiences heavy-duty cycles. The Arduino microcontroller dictates myriad safety features and controls and monitors various devices' functions. This allows for easier mobile and Wi-Fi integration to remotely operate electronic devices, providing the system with an additional level of efficiency that increases safety in various applications such as agriculture, home automation, and industrial uses. Based on the multi-segment framework, the system processes input signals managed by an Arduino-based microcontroller that controls output devices such as motors, fans, and lights. It can be monitored 24/7 via a mode based on IoT (Internet of Things) technology to receive feedback and perform operations from anywhere in the world. This system supports system anomaly detection-e.g., motor stalls and overvoltage conditions. It can alert users if something malfunctions, improve visibility of day-to-day operations, and allow for swift responses. The results show that the designed system has improved the flexibility and usability of the operational load control using the Arduino microcontroller. It was also found that upon detecting a stall, the system feedback is sent as a notification to the user by the Blynk mobile app, thus preventing the motor operations from serious damage. Therefore, this system was able to offer sustainable solutions in contrast to overloading or overheating that may be faced by the system or the motors. It is also recommended that load frequency control (LFC) be implemented with Proportional-Integral-Derivative (PID) controllers and optimized techniques to make interconnected power systems more stable and efficient.

Keywords- Advanced load control system (ALCS), Internet of Things (IoT), Load Imbalance, Faulty Notification.

I. INTRODUCTION

Advanced Load Control System (ALCS) via Arduino is an indicator system designed and built for industrial machines that operate in heavy load zones [1]. It is constructed with the intrinsically safe features required for an industrial machine or motor operating in a heavy industrial environment [2]. The ALCS via Arduino has all the safety requirements for an industrial environment. Monitoring and control are the main entities of any field that can ensure effective performance. Hence, its importance is rising exponentially in this modern era. Industries worldwide have different parameters and measures to be placed within the limit [3]. Variations in these values may lead to the ceasing of performance or even the destruction of the equipment [4]. Hence, these will be monitored in real-time and controlled whenever needed. Accordingly, it can be characterized as a component that minimizes human interaction as much as possible and replaces it with automated electronic systems in various household processes [5]. For the development of such a system, Arduino is used as the main node for monitoring and controlling purposes. It can also collect data as it is designed to interact with the physical world. The system is placed in a network so that different devices and components can communicate and interact with each other, as well as end-users or other entities that do not appear. Other times, emergencies require prompt remediation with available staff–who don't necessarily have the level of expertise that your most experienced operators have failed to perform preventive maintenance [6]. The rise of the Internet of Things (IoT) has disrupted several industries, but one sector that is leaving a major impact is power distribution systems. IoT systems are pivotal in this regard when it comes to load distribution, real-time monitoring (including recording performance metrics), dynamic load balancing, and response time predictions for faulty components that may cause slowdowns or loss of data quality [7-10]. These IoT-based approaches can save both lives and assembly lines. The result has proven crucial for the stability, reliability, and efficiency of contemporary power systems, which are based on renewable energy resources and face increasing demands. Rathor and Saxena (2020) demonstrate the importance of IoT schemes in load balancing the energy management systems (EMS), revealing their capacity to amalgamate several energy sources and decentralize flow direction [11]. One of the key points they make is that coordinating all types of resources to share and trade energy with each other will be a crucial part of keeping supply and demand balanced, especially as renewable energy output can vary dramatically over short periods. Electric vehicles have the potential to complicate matters even further. In managing load distribution, Emamian et al.[12] illustrate how IoT systems play an important role by presenting a case study of an Intelligent Monitoring System (IMS), which uses their newly developed framework in photovoltaic plants as a data and device interoperating medium. This approach enables real-time monitoring and improves load distribution through fast alerts and data communication between different parts, ensuring more effective energy efficiency. Goudarzi et al. [13] explicate the essential role of IoT in load distribution control within smart grids, amplifying how these devices benefit and increase energy utilization and making them considerably more responsive and efficient. They emphasize that IoT technologies enable better control and monitoring of energy resources, in contrast to traditional centralized systems, leading to significant enhancements in current grid advancements, particularly in resiliency. Most equipment requires regular maintenance for optimal performance, but preventive maintenance is often the first task to be neglected when the situation is short-staffed and overwhelmed. It is quite easy to overlook regular maintenance when everything seems to be running smoothly, and many companies operate under the assumption that experienced workers will identify impending trouble before equipment fails [14]. Therefore, this system focuses on designing an IoT system that can wirelessly control various speeds over a wide range and prevent voltage overvoltage to the machine autonomously.

II. RELATED WORKS

Table 1 shows a summary of the previous works. The details of this summary were built up based on the procedure and results and a discussion of the related works, with the main focus on the highlighted advantages and lacking points.

TABLE I. LIST OF THE RELATED RESEARCH

Author & Year	Advantages / Remarkable Achievements	Lack Points	Remarks
Wanqing Xie, 2023 [15]	• Provides an ALCS utilizing Arduino in an ex vivo organ-on-a-chip model. • The system maintains disc health through optimized axial force measurements and nutrient delivery.	Limited capacity to explore cumulative effects of mechanical loading and nutrient interactions on disc health.	Significant advancement in mimicking in vivo conditions, but further research needed on long-term effects.
Doringin et al., 2019 [16]	• Addresses challenges of unbalanced three-phase electrical loads using Arduino, reducing electrical power losses through intelligent control mechanisms.	System's reliance on specific software and hardware configurations may limit adaptability in diverse environments.	Effective in balancing load currents but may require modifications for broader applicability.
Low et al., 2020 [17]	• Enhances feasibility of conducting complex multistressor experiments in marine research using a low-cost, modular Arduino system.	High costs of essential sensors and actuators may still pose a barrier to fully realizing the system's potential.	Affordable and user-friendly for researchers, though cost barriers persist for certain components.
Yumurtacı & Verim, 2020 [18]	• Emphasizes the educational value of an Arduino-based load control system, effectively teaching closed-loop control concepts through a liquid level control system.	Lacks extensive real-world application data, limiting its generalizability beyond educational settings.	Excellent pedagogical tool, but further validation needed for real-world applications.
Rikwan & Ma'arif, 2023 [19]	• Examines a PID control system implemented via Arduino for DC motor speed regulation, highlighting successful feedback encoder application and real-time data visualization.	Reliance on trial and error for parameter tuning may limit adaptability and efficiency in dynamic environments.	Enhances user interaction and control precision, but optimization techniques could improve adaptability.
Agung & Djuni, 2017 [20]	• Focuses on lighting control using Arduino, integrating real-time clock (RTC) functionality and SMS communication, with automated control based on environmental conditions.	Lacks scalability and does not address potential communication reliability issues, impacting overall effectiveness.	Effective for automated lighting control, but limited in scalability and communication reliability.

III. SYSTEM METHODOLOGY

A. Block Diagram

The ALCS via Arduino support block diagram (Figure 1) is separated into three segments. They consist of input, processor, and output. The system's first segment is the input source for the mechanism that compromises the android control switch, voltage, and current sensor. This input data and information is later sent to the second segment, processor development; in this system, an Arduino-based microcontroller is chosen for the processor development and integrated using the Arduino IDE software. The final segment will be the output source, which will be a compromise of the DC motor and lights as the load.

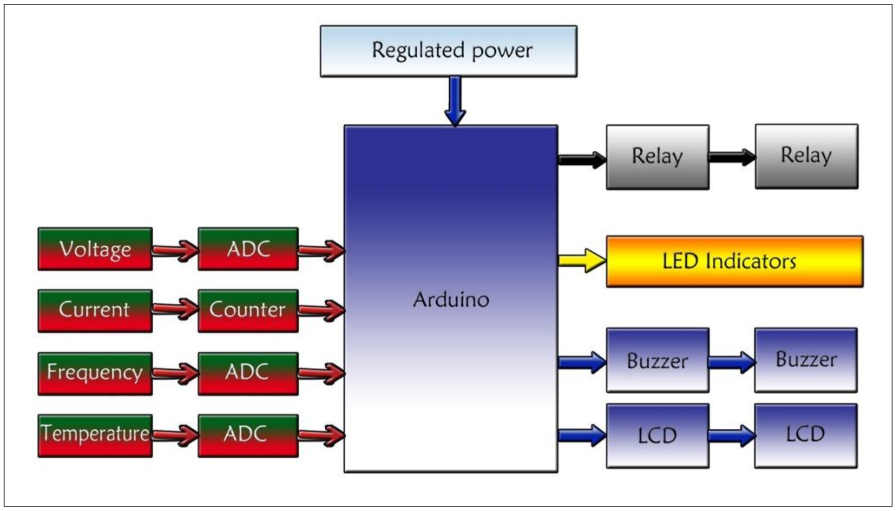

Figure 1, Block diagram of the ALCS

B. Flow Chart

Figure 2 shows the system's flow chart, which only shows the essential parts and their connections. The attached flowchart illustrates the operational sequence of an embedded system that regulates power supply distribution and oversees outlet nodes, featuring a dedicated subroutine for touch panel interaction. These include the main code flow and timer interrupt subroutine, which performs different system actions, which are as follows:

- System initialization, among the machine startups, is a mandatory step for hardware partial setup and software routines. This stage holds significant value as it confirms the precise creation of the system and its readiness to execute necessary tasks. After the initialization, it enters a loop and checks for commands given over GPRS (General Packet Radio Service).
- Blocking Receive Command By GPRS? Determines the system's response. If a command is identified, it proceeds to process and validate the single supported command (the match that logs action). This is crucial as it enables the system to comprehend and confirm the commands provided, preventing unauthenticated access.
- For the outlet or gateway node: After verifying the input command, the system forwards it to the concerned outlet. By doing this, we ensure that only the nodes responsible for their operations receive the commands.
- Next, we gather additional power information, and then the outlet node gathers data for step A of its operation. This data collection is crucial for monitoring the power consumption of each node in the system and for managing the overall power of the system.
- After collecting the data, we calculate the total power consumption of each branch by summing the total power used across all branches. These total power requirement numbers are critical in managing your nodes to achieve optimal power distribution.
- The system sends a protective value to the outlet node at each step, ensuring transient stability and safe operation. Which is a must for securing the nodes? Also, to keep the system stretched and stable.
- The system then checks the next branch decision block to determine whether any other branches need to be shut down. If not, it repeats the loop, gathering additional power data from a single outlet node. What if there were simply no more branches? That is when it gets to the fault at the outlet node. Is there a decision block that checks the output status of outlet nodes?
- Fault detection is a process that locates and corrects any failures present in a system. The outlet node state condition appears when a fault occurs, revealing the nodes' state and enabling operators to respond appropriately. The timer interrupt subroutine is also independent from the main program flow, and its purpose is to handle touch panel interactions.
- The timer develops and initiates touch control inputs through this interrupt, allowing them to function as a real-time touch interface without interfering with your program's primary functions. The subroutine is a randomly selected display of panel-touched signals, where the system detects actions cropped from touch-panel interactions.
- After retrieving the data, the analogue signals from the touch panel are converted into digital data using the Analog-to-Digital Converter (ADC). We should convert the data into an analogue format from a digital one to the best and purest extent possible. Subsequently, we calculate the actual contact position between an element and a panel within the converted data using a superimposed step signal detection method.

- The system receives this information from one end, enabling the induction of a flawless response. Once a subroutine completes the location calculation, it returns control to its calling main program.

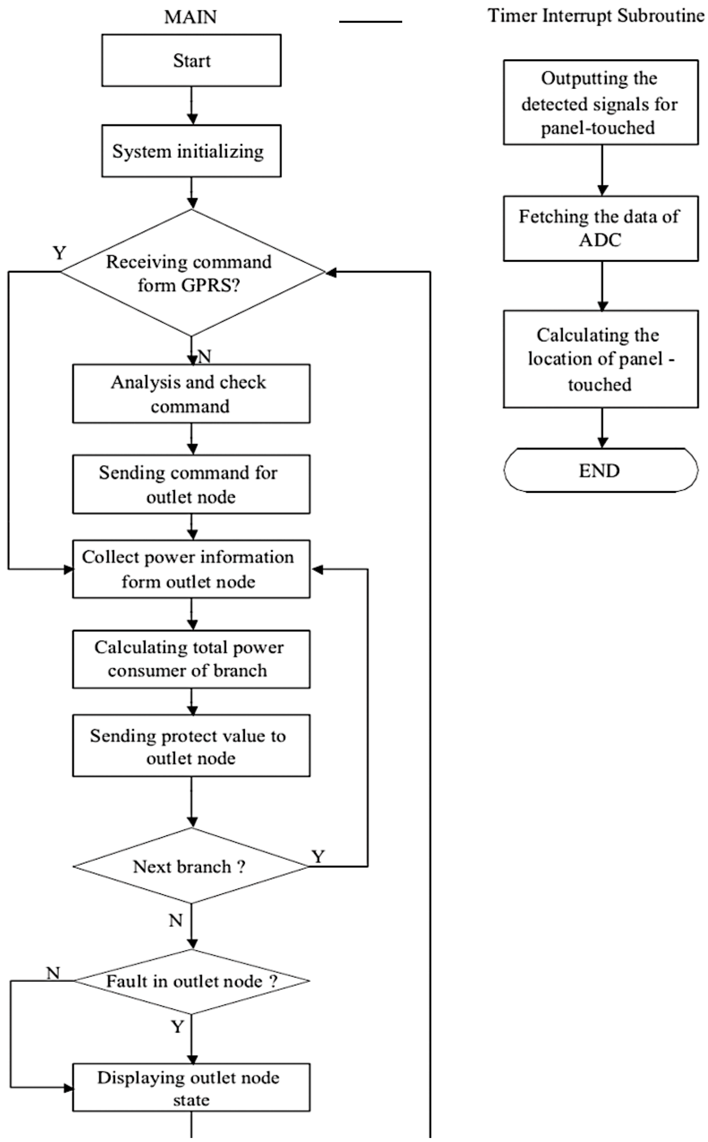

Figure 2, Flow chart of the system

C. Research hardware components

This circuit diagram represents the research utilizing an Arduino Uno microcontroller, an infrared (IR) receiver, and a series of transistors, relays, and diodes to control multiple loads (likely appliances or other electronic devices). The key components are as follows:

i. **Arduino Uno (Rev3):**
The Arduino Uno serves as the system's central control unit. It receives signals, processes them, and sends commands to control the loads. The pins in the diagram (D3 to D6) are configured as digital output pins to control the transistors.

ii. **TSOP1738 IR Receiver:**
The TSOP1738 IR receiver module detects infrared signals (likely from a remote control). It has three pins: VCC (5V), GND, and DATA. The DATA pin is connected to one of the Arduino's digital input pins (not clearly labeled in the diagram but likely to be one of the digital pins like D2), allowing the Arduino to receive the commands sent via IR remote control.

iii. **Resistors (R1 to R4):**
Resistors (each 1 kΩ) are connected between the Arduino's digital output pins (D3 to D6) and the base of the transistors (Q1 to Q4). These resistors limit the current flowing into the transistor's base, protecting the Arduino and the transistors from excessive current.

iv. **Transistors (Q1 to Q4, 2N2222):**
The 2N2222 NPN transistors act as switches. When a digital output pin from the Arduino sends a HIGH signal (approximately 5V), the corresponding transistor is activated (saturated), allowing current to flow from the collector to the emitter, energizing the relay.

v. **Relays (U1 to U4):**
Relays are electromagnetic switches that control the connection of the loads. Each relay is connected to a 12V power supply, which powers the relay coil when the corresponding transistor is activated. This action closes the relay's normally open (NO) contacts, allowing current to flow to the load.

vi. **Diodes (D1 to D4, 1N4001):**
The diodes are placed across the relay coils and are essential for protecting the transistors from the back EMF (Electromotive Force) generated when the relay coil is de-energized. The 1N4001 diodes are standard flyback diodes, which provide a path for the induced current, thereby preventing it from damaging the transistors.

vii. **Loads:**
The loads in this diagram are connected to the relay contacts. The specific nature of these loads needs to be detailed in the diagram, but they could be anything from lights to motors or other electrical devices that operate on the relay's voltage rating.

D. Working Principle

The system begins with receiving an infrared signal from a remote control, as shown in Figure 3. This signal is captured by the TSOP1738 IR receiver module, which is sensitive to the specific frequency of the remote's emitted IR signal. The TSOP1738 is connected to the Arduino Uno, where the DATA pin of the IR receiver communicates with one of the Arduino's digital input pins. When an IR signal is detected, the TSOP1738 converts this signal into a digital pulse, which is then fed into the Arduino for further processing.

The Arduino Uno, acting as the system's brain, processes the received signal according to its programmed logic. This logic could include decoding the IR signal to determine which button was pressed on the remote control and subsequently deciding which load needs to be activated or deactivated. Arduino uses its digital output pins (specifically D3 to D6 in this circuit) to send control signals based on the input processed by the IR receiver.

Once the Arduino determines which load needs to be controlled, it sends a HIGH signal (5V) to the corresponding digital output pin. This signal is routed through a resistor (to limit the current and protect the circuit) and into the base of a 2N2222 NPN transistor. The transistor acts as an electronic switch. When the base of the transistor receives the HIGH signal from the Arduino, it allows current to flow from its collector to its emitter, activating the transistor.

As the transistor turns on, current passes in a direction through the relay coil properly connected to your transistor's collector. Each of the circuit's relays uses a 12V power source to ensure a sufficient voltage is present whenever a transistor within them switches on and its coil requires energizing. Energizing the relay's coil closes a pair of electrical contacts, supplying power to its suitably connected load. This connection routes through the load, enabling it to function as desired. A flying back diode (IN4001) was utilized across each relay coil, which is considered a vital part of the circuit diagram, whereby these diodes control the shielding of the relay from the back-EMF caused when relays of coils lose power. Furthermore, these diodes also prevent any damage that may be caused to the transistors, which may lead to switch-signal voltage-carrying conditions. When the transistor is turned off, this magnetic field suddenly collapses and could produce a reverse voltage large enough to damage the transistor. When the magnetic field collapses, the flyback diode provides a path for this induced current to flow and poof away, keeping it from passing back into the transistor and destroying reliability in your circuit. Figure 4 shows the system's circuit simulation, which illustrates the relay control system using Arduino as a microcontroller and LCD display. The ULN2003A Darlington transistor relay was used to control the ON/OFF multi-relays that handle various loads connected to an AC power supply.

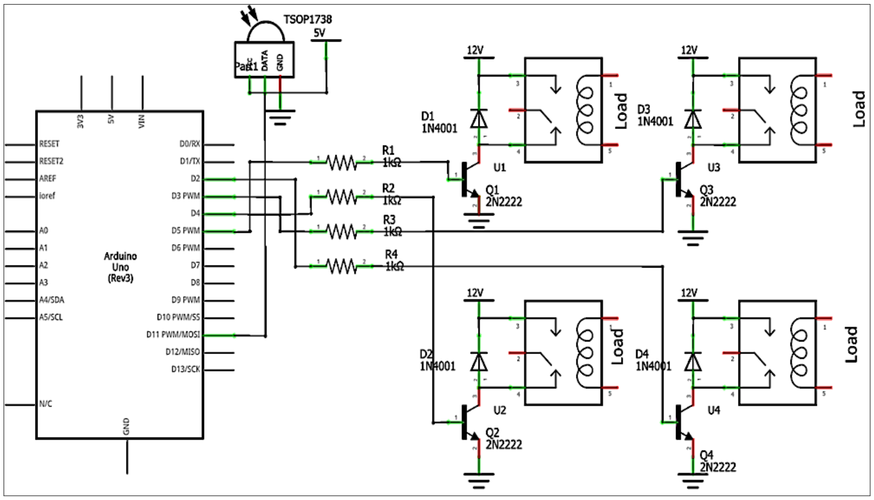

Figure 3, Circuit Diagram of the advanced load control system

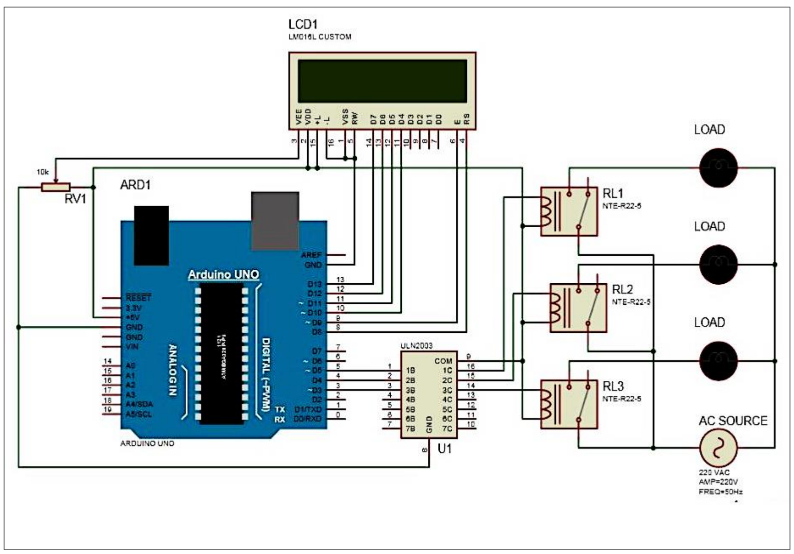

Figure 4, Circuit Simulation of Advanced Load Control System

E. Connection of components

The motor driver is an important part of controlling the motor in this system, as shown in Figure 5a, whereas the motor is connected to the microcontroller (Arduino) and powered by a 12V power supply without utilizing the 5V power supplied by the Arduino. This separate power source is necessary because the motors demand significantly more power than the Arduino can provide. By isolating the power supplies, the design prevents the Arduino from overloading and ensures that the motors receive the adequate power needed for optimal performance. The motor driver's output is then connected to the motors. It's vital to ensure the motors are compatible with the motor driver. If a motor with a higher power rating than the driver is used, it could damage the motor driver or cause it to malfunction. Proper matching of motor and driver specifications is crucial for the system's reliability and longevity.

ISSN: 2710-2165

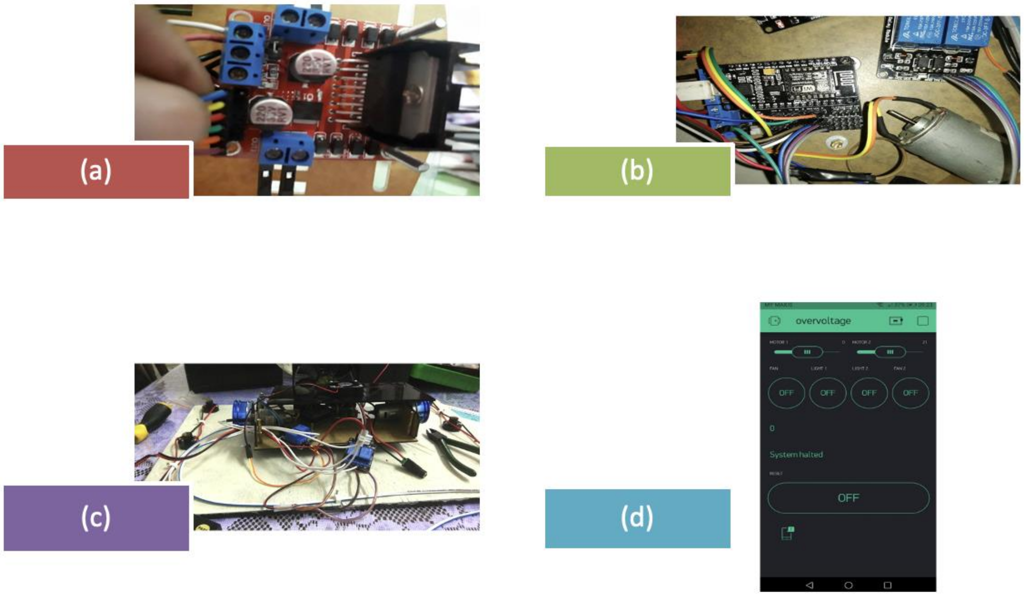

Figure 5, (a) Connection of the motor driver, (b) Connection of the relay, (c) Complete wiring of the prototype, and (d) Blynk Application.

The system employs four relays that are connected similarly, as shown in Figure 5. These relays control other loads, such as a 5V fan, 12V lights, and other devices. The relays act as switches, allowing the Arduino to control these loads by connecting or disconnecting them from the power supply. The relays provide isolation between the low-power control side (Arduino) and the high-power load side, ensuring the safe operation of the system.

The complete wiring of the prototype involves numerous connections, as indicated in the discussion. To ensure that each component receives the correct voltage and current, the system uses three separate power supplies:

a. A power supply for the Arduino microcontroller, typically 5V.
b. A dedicated 12V power supply for the motor driver.
c. A 12V rechargeable battery to power the remaining loads, such as the relays and connected devices.

Utilizing the multi-power supplies will help prevent the system from overloading and provide more stability. The wire mapping was carefully handled, and all the system connections were secured and in place. The system has been integrated with IoT monitoring via a mobile app called Blynk, which can provide full control of the running system wirelessly via Wi-Fi. The architecture of the implemented setup will provide a flexible technique to control the applied load and continuous monitoring.

IV. RESULTS AND DISCUSSION

A. System design

Figure 6 shows the final design we are prototyping, embodying a well-designed, highly-engineered gadget. The details in Figure 6(a-d), which showcase various aspects of the prototype from various perspectives, such as front, top, and side views, offer crucial insights into the functional design of a network wall time selector, providing a more realistic context than merely use-case scenarios alone. The prototype's overhead view reveals the interior layout, centered on what appears to be a motor or fan. This confirms its functional design. The casing itself clearly indicates that this core component performs most of the tasks necessary for the system's operation. A web of cables connects the surrounding components to each other. These might be sensors, switches, or indicators that support the functionality of the central unit. This front view of the prototype shows some external components, including a fan or impeller at the front of the face. This fan is likely to be used for air circulation or thrust generation and plays a vital role in the operation of the device. The front of this design displays precise coordination and perfect symmetry, conveying a strong sense of control over airflow or directional force. Side views provide more information about the prototype, with a particular focus on the integration of external components and cable routing design. This view emphasizes the balance of the design and delves into more intricacies, providing insights into the interconnections between each component. The connectivity capabilities are particularly notable, ensuring a seamless connection between all components without any clutter. The fact is essential as it enhances the functionality and utility of the prototype.

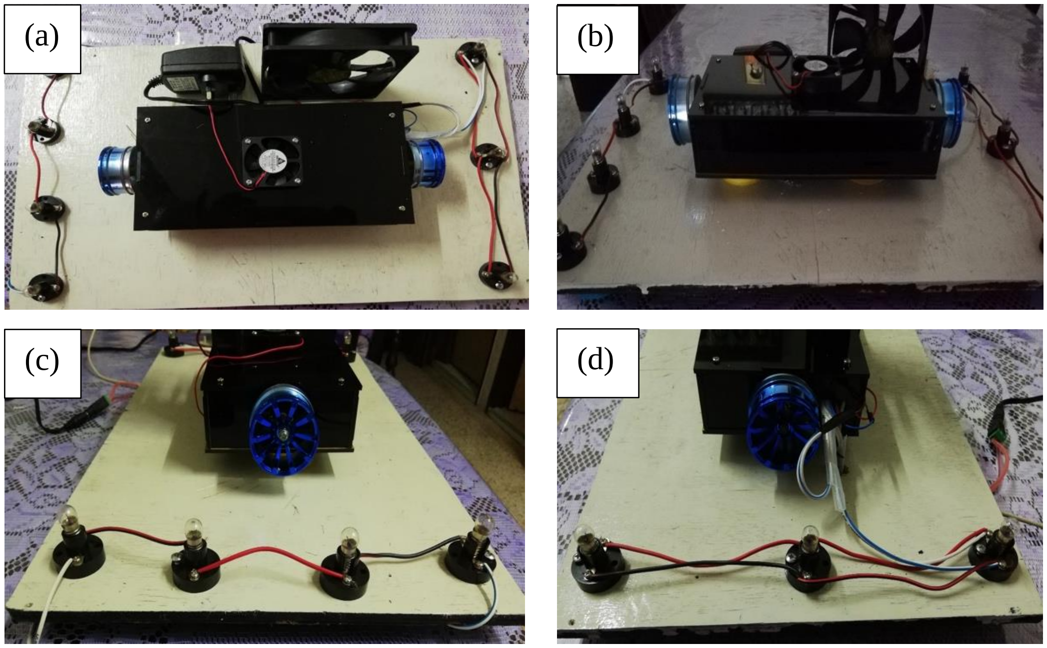

Figure 6, Prototype, (a) top view, (b) front view, (c,d) side view

B. System Functionality and Feedback

The system can be powered on, and its status is that the red light is closed (Figure 7a). This red light is an important sign that the system has been powered so that it is ready to execute into my Arduino. This light indicates that the electrical connections are in good shape and that the correct power supply works adequately. This is a pre-requisite step before you can go down to test other stages of the keepalive operation and ensure that the system gets enough power to work with. This system is built around the gear motor, which provides the mechanical movement required to arrive at this function. The motor driver is an interface between the Arduino Uno, which only sends weak signals and the stronger power requirements of a DC brushless motor.

Figure 7, (a) Indicator of the system turned ON, (b) the Indicator of the Wi-Fi Module Connected, (c) the motor driver connected to Arduino, (d) the motor working with adjustable speed, and (e) Other load functions.

Figure 7b highlights the connection of the ESP8266 Wi-Fi module to the Internet. The blue light in the image indicates that the Wi-Fi module has successfully connected to a Wi-Fi network, enabling communication between the Arduino and external devices, such as a smartphone or a computer. This connectivity is critical to the system, allowing for remote control and system monitoring through the Internet. The ESP8266 module is extensively used in IoT applications because it is low-cost and easy to implement with microcontrollers like Arduino. The blue light in this system indicates that the module is turned on and connected to a Wi-Fi network. By doing this, the module can communicate, exchange data, and work remotely from the system.

Figure 7c shows the motor driver connected to an Arduino, and the motor driver contains H-bridge circuits. A light shows whether the board is switched on and regulates the motor from signals coming through Arduino, which means that it is active. A motor driver board with LED lights shows that the driver is on. Further confirmation is needed to indicate that the driver is powered on and ready for instructions. The Arduino Uno sends a PWM (pulse width modulation) signal to the motor driver, which regulates how fast or in which direction the gear motor should rotate. The motor driver then decodes these signals to control the operation of the gear motor. This way, it would accurately tune the operation of the motor and, through that, advance in a controlled and predictable manner with our system. The system gets a lot of flexibility and usability because, finally, the speed and direction of the motor can be controlled through software programming in Arduino. In Figures 7d and e, the prototype's basic operational and safety attributes are described in detail, including the motor and control through the Blynk mobile app platform. This feature is in place to protect the system from damage when used on motors with faults, such as a stuck motor during use. The system should notice when this abnormal state occurs, circulate current to help hold the motor manually, or if something foreign happens unexpectedly while it is running.

The detection mechanism is likely based on monitoring the motor's current or voltage levels, which would spike if the motor were stuck and unable to rotate freely. Once this condition is identified, the system triggers an automatic safety response. Upon detecting the motor stall, the system immediately sends a notification to the connected mobile device via the Blynk application. As shown in

Figures 8a and b, this notification alerts the user that the motor is stuck and that an overvoltage condition may have occurred. Automatic notification is critical as it lets users be informed in real-time, even if they are not physically present near the device. Simultaneously, the system halts the motor's operation to prevent any potential damage resulting from continued operation under stalled conditions. This automatic shutdown protects both the motor and the overall system, ensuring that no further stress is placed on the components, which could lead to overheating or other failures.

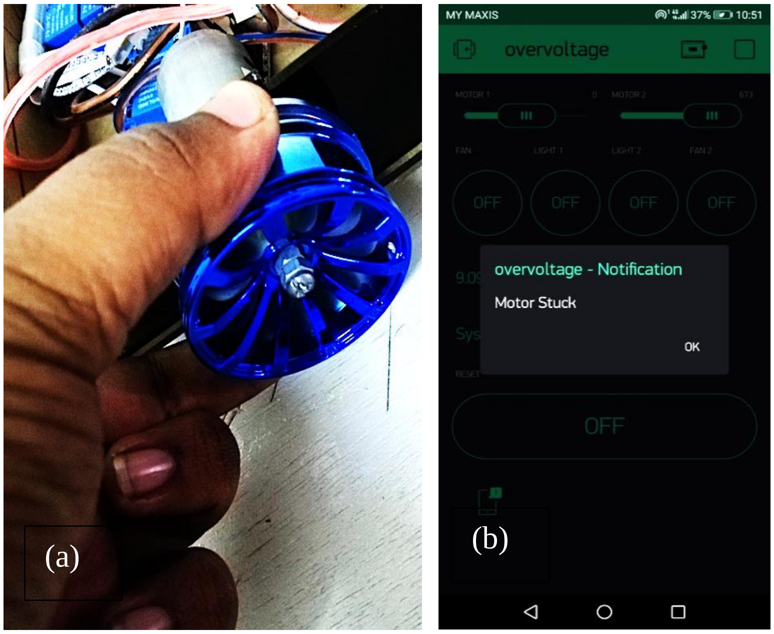

Figure 8, (a) Motor hold during operation and (b) Notification received

V. CONCLUSION

This paper proposed a multi-segment system framework to provide an ALCS with an incentive approach in the load distribution systems. The load control uses an Arduino microcontroller, and feedback is monitored via a Blynk mobile app as an IoT implementation approach. The result of the proposed system shows its effectiveness in controlling the load and providing an efficient load management system, as well as its usability and reliability in protecting the motors from overheating or overloading. Therefore, this Arduino-based system is considered a robust and sophisticated solution that is capable of monitoring and controlling various operational motors for agriculture and industrial automation as they potentially pave the way for improving productivity and automated long-term operational efficiency. Further recommendations to improve the performance or efficiency of the system include a model for load frequency control (LFC) over multiple areas to be incorporated to ensure that adequate power supply is always available to cater to changing demand. This further step is critical in maintaining a balance between generation and consumption and the appropriate use of the optimum control systems and conventional control systems.

ACKNOWLEDGEMENT

The author expresses gratitude to Management and Science University (MSU) for its facilities and support.

REFERENCES

[1] H. K. Kondaveeti, N. K. Kumaravelu, S. D. Vanambathina, S. E. Mathe, and S. Vappangi, "A systematic literature review on prototyping with Arduino: Applications, challenges, advantages, and limitations," Computer Science Review, vol. 40, p. 100364, 2021.
[2] J. P. Steiner, C. Baby, C. Buck, and D. F. Carmen, "Multiple location load control system," ed: Google Patents, 2011.
[3] A. Kammeyer and J. Nordholz, "Power Optimisation for a Verifiably Secure Separation Kernel," in 2023 2nd Asia Conference on Electrical, Power and Computer Engineering (EPCE), 2023: IEEE, pp. 164-171.
[4] A. La Monaca et al., "Surface integrity in metal machining-Part II: Functional performance," International Journal of Machine Tools and Manufacture, vol. 164, p. 103718, 2021.
[5] S. Aswathy, A. K. Tyagi, and S. Kumari, "The Future of Edge Computing with Blockchain Technology: Possibility of Threats, Opportunities, and Challenges," Recent Trends in Blockchain for Information Systems Security and Privacy, pp. 261-292, 2021.

[6] P. Goel, "Decision Support System for Improved Operations, Maintenance, and Safety: A Data-Driven Approach," Texas A&M University, 2020.
[7] S. Jadon, K. Pradyuman, U. Kalaria, K. Varsha, K. Gupta, and P. B. Honnavalli, "A comprehensive study of load balancing approaches in real-time multi-core systems for mixed real-time tasks," IEEE Access, 2024.
[8] O. Y. Abdulhammed, "Load balancing of IoT tasks in the cloud computing by using sparrow search algorithm," The Journal of Supercomputing, vol. 78, no. 3, pp. 3266-3287, 2022.
[9] F. Chahlaoui and H. Dahmouni, "A taxonomy of load balancing mechanisms in centralized and distributed SDN architectures," SN Computer Science, vol. 1, no. 5, p. 268, 2020.
[10] O. Laayati, M. Bouzi, and A. Chebak, "Smart energy management system: design of a monitoring and peak load forecasting system for an experimental open-pit mine," Applied System Innovation, vol. 5, no. 1, p. 18, 2022.
[11] S. K. Rathor and D. Saxena, "Energy Management System for Smart Grid: An Overview and Key Issues," International Journal of Energy Research, vol. 44, no. 6, pp. 4067-4109, 2020, doi: 10.1002/er.4883.
[12] M. Emamian, A. Eskandari, M. Aghaei, A. Nedaei, A. M. M. Sizkouhi, and J. Milimonfared, "Cloud Computing and IoT Based Intelligent Monitoring System for Photovoltaic Plants Using Machine Learning Techniques," Energies, vol. 15, no. 9, p. 3014, 2022, doi: 10.3390/en15093014.
[13] A. Goudarzi, F. Ghayoor, M. Waseem, S. Fahad, and I. Traoré, "A Survey on IoT-Enabled Smart Grids: Emerging, Applications, Challenges, and Outlook," Energies, vol. 15, no. 19, p. 6984, 2022, doi: 10.3390/en15196984.
[14] S. Chakraborty, "MOBILE BASED APPLIANCE CONTROL USING DTMF DECODER," Electronics and Telecommunication Engineering, Electronics Engineering, Kiit University, India, 2015.
[15] W. Xie et al., "Intervertebral Disc-on-a-Chip MF: A New Model for Mouse Disc Culture via Integrating Mechanical Loading and Dynamic Media Flow," Advanced Materials Technologies, vol. 8, no. 21, 2023, doi: 10.1002/admt.202300606.
[16] F. Doringin, M. J. Budiman, and S. Walukow, "Balance Automatic Control System Load 3 Phase," International Journal of Computer Applications, vol. 177, no. 9, pp. 33-37, 2019, doi: 10.5120/ijca2019919472.
[17] N. H. N. Low, C. A. Ng, and F. Micheli, "A Low-cost Modular Control System for Multistressor Experiments," Limnology and Oceanography Methods, vol. 18, no. 10, pp. 623-634, 2020, doi: 10.1002/lom3.10389.
[18] M. Yumurtacı and O. Verim, "Liquid Level Control With Different Control Methods Based on Matlab/Simulink and Arduino for the Control Systems Lesson," International Advanced Researches and Engineering Journal, vol. 4, no. 3, pp. 249-254, 2020, doi: 10.35860/iarej.750664.
[19] R. Rikwan and A. Ma'arif, "DC Motor Rotary Speed Control With Arduino UNO Based PID Control," vol. 1, no. 1, pp. 17-31, 2023, doi: 10.59247/csol.v1i1.6.
[20] I. R. Agung and I. D. D. H, "Design of Lighting Control With RTC Timer and SMS Using Microcontroller," Journal of Electrical Electronics and Informatics, vol. 1, no. 1, p. 24, 2017, doi: 10.24843/jeei.2017.v01.i01.p05.