CARITAS UNIVERSITY AMORJI-NIKE, EMENE, ENUGU STATE
Caritas Journal of Engineering Technology
CJET, Volume 4, Issue 1 (2025)
Article's History: Received:18th December, 2024 Revised:25th January 2025 Accepted: 21st February, 2025

Affordable Micro-Controller Arduino-Based Board Experimental Training Kit Implementation in Nigerian Universities

Ifeagwu E. N.
Department of Electrical and Electronic Engineering,
Federal University Otuoke, Bayelsa State,
ifeagwuen@fuotuoke.edu.ng

Ezema D.C.
State University of Medical and Applied Sciences (SUMAS), Igbo-Eno, Enugu State, Nigeria.
donatus.ezema@sumas.edu.ng

Abstract

Engineering and technology education in Nigeria faces challenges, including limited access to practical learning tools and high-cost laboratory equipment. To bridge this gap, this paper proposes the development of an affordable, Arduino-based experimental training kit tailored for Nigerian universities. The kit aims to provide hands-on experience in embedded systems, electronics, and programming, fostering practical skills and innovation among students. The training kit is designed with cost-effective components, modularity, and scalability, ensuring it can support a wide range of experiments, from basic circuit design to advanced microcontroller-based projects. The Arduino platform was selected for its affordability, open-source ecosystem, and ease of integration with various sensors and actuators. To validate its effectiveness, the training kit was piloted in selected Nigerian universities, where it demonstrated improved student engagement, better understanding of theoretical concepts, and enhanced problem-solving abilities. Additionally, the kit's affordability makes it accessible to institutions with limited budgets, promoting inclusive education. This study highlights the potential of low-cost, locally adaptable solutions in addressing the practical learning challenges in Nigerian higher education, emphasizing the importance of integrating such tools into engineering curricula to produce industry-ready graduates.

Keywords: AT mega 328, Arduino Uno, Internet of Things (IoT), Micro-controller,

1.0 Introduction

Engineering and technology education in Nigerian universities can benefit greatly from practical, hands-on learning experiences. However, the high cost of laboratory equipment often limits access to essential tools for students to develop their skills. This has created a gap between theoretical knowledge and practical application, particularly in fields like electronics, robotics, and IoT (Internet of Things) (Kopetz H., 2021)

An open-source platform for electronic prototyping, called Arduino is used to create interactive electronic devices and electronic projects (Monk S., 2020). Arduino is made up of a hardware microcontroller (programmable circuit board) and software called an IDE (Integrated Development Environment) that runs on your computer and is used to write and upload computer code to the board (Muhammed A., 2014). A range of microprocessors and controllers are used in the creation of Arduino boards (Ifeagwu, et al., 2015). A collection

of analogue output/input (I/O) pins on the board may interface with different expansion boards (shields), breadboards (for prototyping), and other circuits (Dogan I., 2018). The training kit integrates hardware modules and comprehensive software support, enabling students to perform practical experiments, learn programming, and understand real-world applications (Douglas V, 2018). The modular design allows for flexibility and scalability, accommodating various levels of education and experimentation (Edward, 2015). The solution also includes detailed instructional materials and tutorials to facilitate hands-on learning.

An affordable Arduino-based experimental training kit can bridge this gap, offering a cost-effective, versatile, and scalable solution for university laboratories since there was no alternative way to bridge this gap, this strategy concentrated on putting Arduino as the only way (Ifeagwu, et al., 2015). Arduino, a widely used open-source microcontroller platform, is ideal for learning electronics, programming, and embedded systems (Floyd T., 2018). By designing a kit tailored to the Nigerian education system, universities can empower students to engage in real-world problem-solving and innovation.

For good reason, the Arduino platform has grown in popularity among those who are new to electronics. In contrast to the majority of earlier programmable circuit boards, the Arduino may be updated with new code using a USB cable instead of a separate piece of hardware known as a programmer (Katz P. 2019). Furthermore, the Arduino IDE makes programming simpler by using a condensed form of C++. Last but not least, Arduino offers a standardized form factor that separates the microcontroller's functions into more manageable packaging(Holliday and Resmith,2020). Figure 1 displays an Arduino board diagram.

Figure 1: Diagram of an Arduino Board (Monk S., 2020)

A boot loader that makes uploading programs to the on-chip flash memory easier is pre-programmed into Arduino microcontrollers. The Arduino Uno comes pre-installed with the Optiboot bootloader. Program code is transferred to boards over a serial link to a different computer (Wakerly, 2017). To convert between transistor-transistor logic (TTL) level signals and RS-232 logic levels, certain serial Arduino boards have a level shifter circuit. The FTDI FT 232 and other USB-to-serial adapter chips are used to implement the Universal Serial Bus (USB) programming protocol on Arduino boards nowadays. Some boards, including later-model Uno boards, use an AVR chip instead of the FTDI chip (Ifeagwu and Adebayo, 2024). This chip has USB-to-serial firmware that can be reprogrammed via its own ICSP header. Other versions, such as the unauthorized Boarduino and the Arduino Mini, use Bluetooth, a removable USB-to-serial adaptor board or cable, or other techniques. Standard AVR in-system programming (ISP) programming is utilized in place of the Arduino IDE when utilizing conventional microcontroller tools (Monk, 2020).

The board is equipped with sets of digital and analogue input/output (I/O) pins that may be interfaced with various expansion boards (shields) and other circuits (Anthony, et al., 2019). Its pin diagram is shown in Figure 2 (Anthony, et al., 2019). The board has 14 digital I/O pins (out of which 6 are capable of PWM), and 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts, Uses a 16 MHz quartz crystal oscillator, a power jack, an ICSP header and an RST button (Katzp P., 2019).

The word "uno" means "one" in Italian and was chosen to mark the initial release of Arduino Software (Douglas V., 2018). The Uno board is the first in a series of USB-based Arduino boards; it and version 1.0 of the Arduino IDE were the reference versions of Arduino, which have now evolved to newer releases. The ATmega328 on the board comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer (Predko M, 2017). The Arduino Uno has several facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual COM port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A Software Serial library allows for serial communication on any of Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify the use of the I2C bus.

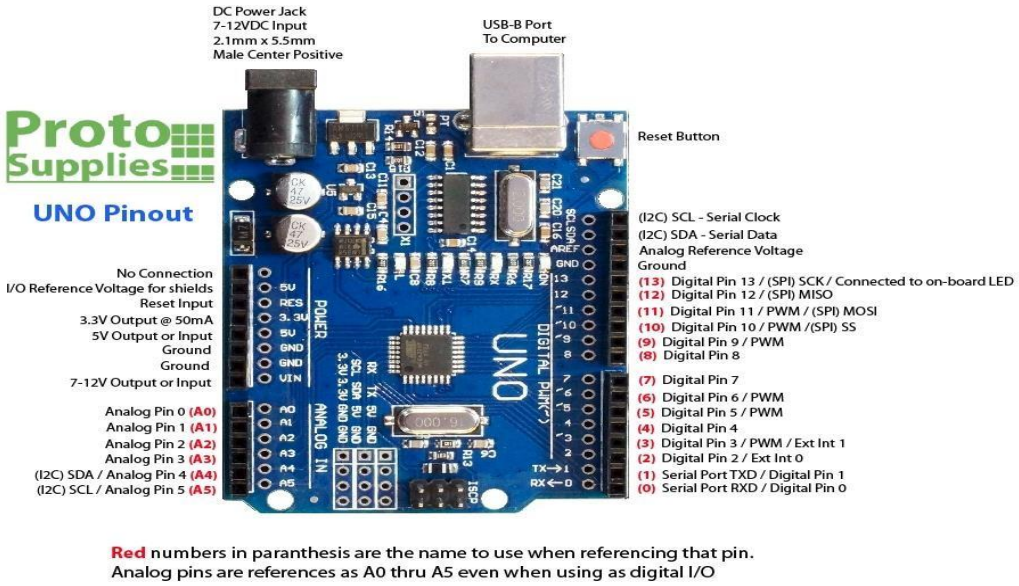

Figure 2: The Arduino Uno Board Layout (Anthony, et al., 2019)

2.0 Materials and Method

2.1 Materials

The materials used for the work contain specifically:

(1) Power Supply section: It transmits power to the entire circuit.
(2) Liquid Crystal Display: This decodes the data from the Arduino and displays the values and signals from the kit
(3) SMS unit: This is where the SIM module (SIM 900A) is used for experiments like the Intruder alert system where a user is alerted for potential security threats. A SIM module is used for this experiment.
(4) Button unit: The button helps to perform LED manipulation and controls how an LED works.
(5) Sensor: A sensor (flame sensor) is used for a fire detection system.
(6) Keypad unit: It is for calculation purposes.
(7) Switching unit: This is the unit where LDR, light light-dependent resistor is used to experiment with a dark activated circuit where low resistance triggers the circuit to switch on. It can also serve as a sensor.
(8) LED unit: It is an indicator and is used for LED MANIPULATION or BUTTON EXPERIMENT.
(9) Sound unit: The sound unit helps to output sound signal from the experimental kit with the help of the buzzer as a sound device
(10) Breadboard: It is the interconnection of other components to the kit.

(11) OP-AMP: It amplifies voltage into the kit
(12) Arduino uno (control unit): It is the major contributing unit of the system.
(13) Laptop (computer) for writing projects and design of the circuit diagram.

(14) Device programmer for burning the program.

(15) Proteus Software for real-time simulation.

(15) Electronic components like transistors, capacitors, resistors.

2.2 Method

Figure 3 shows the circuit diagrams of buttons interfaced to Arduino for LED Control. In the circuit, a button is utilized to turn on and off a collection of LEDs. Some of the LEDs are off when the button is pressed, but the green ones are on. The LEDs change states when the button is released; those that are ON will turn OFF, and those that are OFF will turn ON. The circuit in Figure 3 depicts a simple button-controlled array of LEDs. The power supply to Arduino could be either from the DC power jack (7 - 12V), the USB connector (5V), or the V_{in} pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board. The button will serve as an input source to the ATmega328 controller, hence the pin where it is connected will be defined as an input pin; while the pins where the array of LEDs are connected will be defined as output port.

The Arduino board receives signals from the single push, single throw button. When the button is not pressed, the connection will initially set pin 13 of the Arduino controller to ground. However, as soon as the button is touched, the Arduino receives a 5V source, and the controller will detect that the button is pressed. Depending on the condition of pin 13, the appropriate code is now written to turn the LEDs on or off.

Arduino board can be interfaced with several displays including seven-segment, LCD and also to Dot matrix displays using shift registers. In this paper, we interfaced an LCD to Arduino Uno. The interconnectivity components are Arduino Uno, 16x2 LCD and 5k potentiometer. The Potentiometer was used for the adjustment of the LCD. The block diagram for this interface is shown in Figure 3. This is because the Arduino board can write to the LCD and as well read from it.

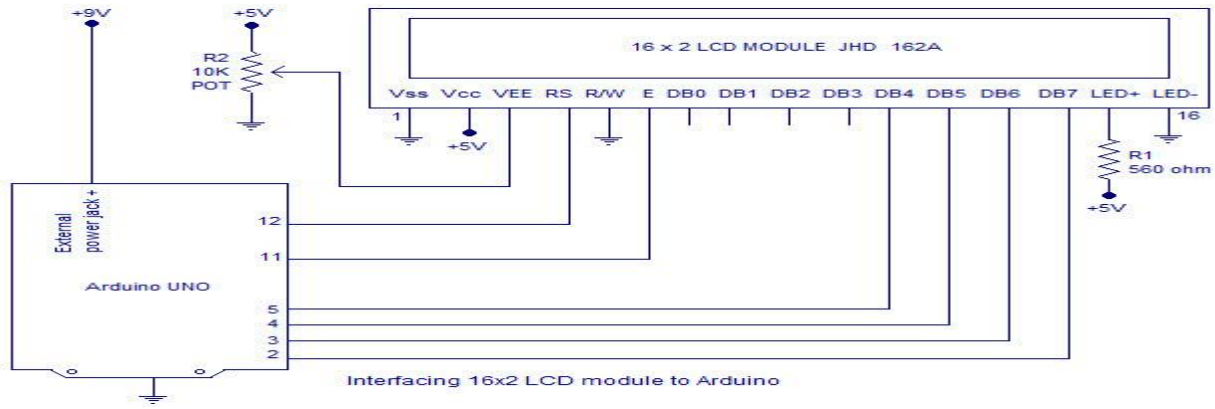

Figure 3: Circuit diagram of LCD interfaced to Arduino Uno.

2.1.4 Software Design of the System

The program for the system of operation was developed in Arduino Programming language. The sample of the program is shown in section 3.1.4.1.

2.1.4.1 Programme Code for the Experimental Kit

```
#include <LiquidCrystal.h> //to include the LCD library code

LiquidCrystallcd(2,3,4,5,6,7); //LiquidCrystal lcd(RS,EN,D4,D5,D6,D7)

 definition for LCD

int BUTTON_1_PIN =A0;  int BUTTON_2_PIN =A1 ;

int BUTTON_3_PIN = A2; int BUTTON_4_PIN=A3;

int LED_1_PIN=8;  int LED_2_PIN=9; int

LED_3_PIN= 10;  int LED_4_PIN = 11;  int

LED_5_PIN = 12;  int LED_6_PIN= 13;  int

LED_7_PIN = 1;  int LED_8_PIN= 0;

intBUTTON_1_PIN_CONTAINER=0;int

BUTTON_2_PIN_CONTAINER=0:int

BUTTON_3_PIN_CONTAINER=0;int BUTTON_4_PIN_CONTAINER =0;

void setup() {// INNITIALIZING... pinMode(BUTTON_1_PIN, INPUT); //setting the BUTTON_1_PIN as input pin pinMode(BUTTON_2_PIN, INPUT); //setting the BUTTON_2_PIN as input pin pinMode(BUTTON_3_PIN, INPUT); //setting the BUTTON_3_PIN as input pin pinMode(BUTTON_4_PIN, INPUT); //setting the BUTTON_4_PIN as input pin pinMode(LED_1_PIN, OUTPUT); //setting the LED_1_PIN as OUTPUT pin pinMode(LED_2_PIN, OUTPUT); //setting the LED_2_PIN as OUTPUT pin pinMode(LED_3_PIN, OUTPUT); //setting the LED_3_PIN as OUTPUT pin pinMode(LED_4_PIN, OUTPUT); //setting the LED_4_PIN as OUTPUT pin pinMode(LED_5_PIN, OUTPUT); //setting the LED_5_PIN as OUTPUT pin pinMode(LED_6_PIN, OUTPUT); //setting the LED_6_PIN as OUTPUT pin pinMode(LED_7_PIN, OUTPUT); //setting the LED_7_PIN as OUTPUT pin pinMode(LED_8_PIN, OUTPUT); //setting the LED_8_PIN as OUTPUT pin digitalWrite(LED_1_PIN, LOW); digitalWrite(LED_2_PIN, LOW); digitalWrite(LED_3_PIN, LOW); digitalWrite(LED_4_PIN, LOW); digitalWrite(LED_5_PIN, LOW); digitalWrite(LED_6_PIN, LOW); digitalWrite(LED_7_PIN, LOW); digitalWrite(LED_8_PIN, LOW); digitalWrite(BUTTON_1_PIN, HIGH); digitalWrite(BUTTON_2_PIN, HIGH); digitalWrite(BUTTON_3_PIN, HIGH); digitalWrite(BUTTON_4_PIN, HIGH);

  lcd.begin(16, 2); //indicating the type of lcd in use lcd.setCursor(2,0);

  lcd.print("ARDUINO BASED"); lcd.setCursor(0,1); lcd.print("TRAINING KIT");

  delay(2000);}
```

2.2 Arduino Uno Testbed Environment

The Arduino IDE, shown in Figure 4, provides a near-complete environment for most Arduino-based work. The top menu bar has the standard options, including “File” (new, load save, etc.), “Edit” (font, copy, paste, etc.), “Sketch” (for compiling and programming), “Tools” (useful options for testing projects), and “Help”. The middle section of the IDE is a simple text editor where you can enter the program code. The bottom section of the IDE is dedicated to an output window that is used to see the status of the compilation, how much memory has been used, any errors that were found in the program, and various other useful messages.

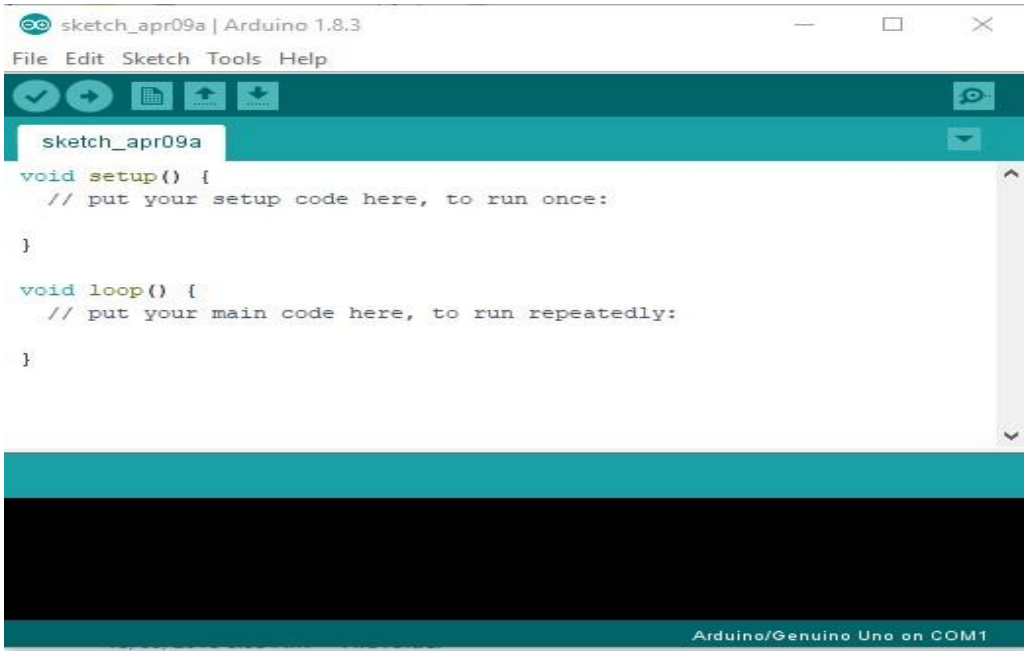

Figure 4: The Arduino IDE in its Default State

The Arduino development environment main program has two functions already defined. The setup and loop function. The setup function is where the code that should run once is written. Setup codes like defining the input and output pins, initializations etc. are written inside the setup function. The loop function should contain the codes that are meant to run continuously in the project. Examples of such codes could be a continuous reading of sensor inputs to know when the state changes, displaying outputs in the selected display device etc.

3.0 Results and Discussion

3.1 Result

The circuit diagram in Figure 5 was developed in a Proteus environment using Proteus 8.5 software. Due to the many interconnections of wires involved, a referencing style of Proteus design was done. The different units developed were integrated to form the circuit below. These units include the button unit, LCD unit, LOR unit, LED unit, keypad unit, switching unit, alarm unit and GSM unit.

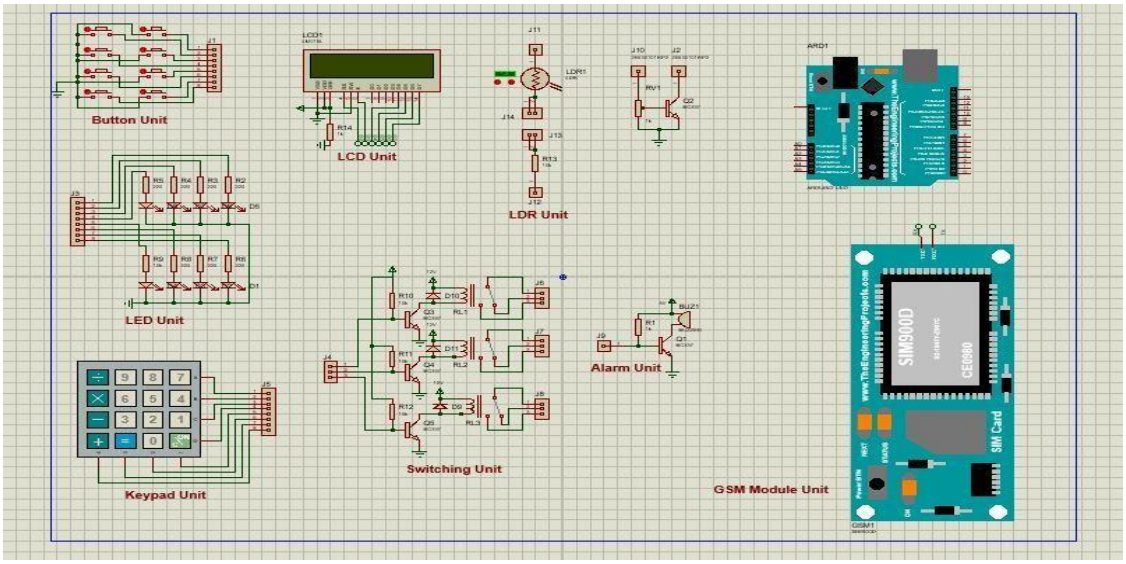

Figure 5: Complete Circuit diagram

3.2: Discussion

The constructed system after integration and testing is shown in Figure 6.

Figure 6: Developed System

3.2.1 How Arduino Experimental Kit Works

The Arduino Uno, an open-source electronics platform built on user-friendly hardware and software, is used in the Arduino experimental kit. Arduino boards may read inputs, such as a light on a sensor, a finger on a button, or a tweet, and convert them into outputs, such as turning on an LED, turning on a motor, or posting something online. By providing the microcontroller on your board with a set of instructions, you may instruct it on what to do. The Arduino software (IDE), which is based on processing, and the Arduino programming language, which is based on wiring, are used to do this. The experimental Kit is an easy-to-use learning tool that allows novices to conduct electronic projects before putting them into practice in real-time.

3.3 Result of Testing of the System

Testing was done once the unit modules were joined to create a system. The system operated as intended and there were no issues as shown in Table 1.

Table 1: Result of Testing

S/N	Description	Quantity	Functionality
1	Arduino uno	1	Functional
2	Transistor Bc 337	2	Functional
3	Buzzer	1	Functional
4	LCD 16 X 2	1	Functional
5	Bread Board	1	Functional
7	SIM 900A	1	Functional
8	Button	8	Functional
9	Keypad	1	Functional
10	Power jack	1	Functional

11	Relay	3	Functional
12	LM 358	1	Functional
13	Resistor(1K)	3	Functional
14	Power Adapter	1	Functional
15	PCB	1	Functional
16	Packaging	1	Functional
17	Connecting Wire	1	Functional
18	Dc to Dc Converter	1	Functional
19	Male and female Connectors	4	Functional
20	LED	10	Functional
21	LDR	1	Functional
22	Resistor (220 ohms)	8	Functional
23	Power Connector	1	Functional

4.0 Conclusion

An Arduino-based experimental kit is a simple tool that novices can use to complete a variety of electronic tasks. It is versatile. After operation, the system may communicate with GSM phones thanks to its GSM module. It can communicate with laptops and other computing devices thanks to a port. Making the system user-friendly is made possible by the LCD that was interfaced with it. It informs the user of the operational state.

Nonetheless, the project was created to help academics and students comprehend how an Arduino board functions. A researcher who creates an Arduino control program should be able to observe the physical or real-time effect. This will significantly increase the motivation and enthusiasm required to advance the research.

References

Anthony N I, Kelvin N N, Emmanuel U C (2019) “Intelligent System Design of Microcontroller-based Real-Time Process Control Trainer”, International Journal of Recent Trends in Engineering and Research (IJRTER), Vol. 05, Issue 04, Pp 1-10.

Chalford P.T., (2012), “The Concise Book of real time systems”, Time system Corporation, Pittsburgh, Pp 45.

Dogan Ibrahim (2018), “Microcontroller-Based Temperature Monitoring and Control”, Elsevier Science & Technology Books, Maryland USA, Pp. 63, 87, 107-129.

Douglas V.H. (2018), “Microcontrollers and Interfacing: “Programming Hardware” McGraw Hill Inc, New York, Pp.270-344.

Edward H, (2015) “Electrical and Electronics Technology”, Pearson Education Ltd, India, PP 634.

Floyd T.F. (2018), "Digital Fundamental", 6th Edition, Prentice-Hall International Inc, New Delhi, Pp 4-7.

Holliday D. and Resmick Robert (2020), "Fundamentals of Microcontrollers", Don Peters Press Ltd, Fulmar, P 88.

Ifeagwu, E N., Adebayo, Adeniyi D. (2024). "The Design and Analysis of a Micro Controller Based Fire Alarm System with Water Sprinkler". International Journal of Engineering and Mathematical Intelligence (IJEMI), Vol8, (1), Pg1-10.

Ifeagwu E.N, Edeko F.O., Emagbetere J.O. (2015) 'Analysis of Least Mean Square Adaptive Beamforming Algorithm of the Adaptive Antenna for improving the performance of the CDMA20001x base mobile radio network" "International Journal on Recent and Innovation Trends in Computing and Communication(IJRITCC), Vol. 3, Issue5,2015, Pg 3296-3299.

Kopetz H. (2021), "Real-Time Systems: Design Principles for Distributed Embedded Applications", Kluwer Academic Publishers, Dordrecht, Pp 6-11.

Katz P.(2019), "Digital Controls Using Microcontrollers", Prentice Hall International Inc, New Delhi, Pp 15-43.

Muhammad Ali Mazidi (2014.), "The 8051 Microcontroller and Embedded Systems", Prentice Hall, New Jersey USA,

Monk S. (2020), "Programming Aduino: Getting Started with Sketches", McGraw-Hill Publishing Companies, New York, Pp. 23-34.

Predko Myke, (2017) "Handbook of Microcontrollers", McGraw Hill, New York, USA, Pp 1-17.

Wakerly John F. (2017), "Digital Designs: Principles and Practices", 4th Edition, PHI Learning Private Limited, New Delhi, Pp 5-6.