International Journal of Advances in Applied Sciences (IJAAS)
Vol. 9, No. 2, June 2020, pp. 77~84
ISSN: 2252-8814, DOI: 10.11591/ijaas.v9.i2.pp77-84

Design, simulation and implementation of an Arduino microcontroller based automatic water level controller with I2C LCD display

Akinwole OO
Department of Electrical/Electronic Engineering, Federal Polytechnic, Nigeria

Article Info

Article history:

Received May 8, 2019
Revised Feb 11, 2020
Accepted Mar 14, 2020

Keywords:

Arduino microcontroller
Float switch
I2C LCD
Simulation
Water level

ABSTRACT

The paper explains utilization of Arduino Microcontroller to automatically control level of water in a tank. From a well spelt out algorithms, flowchart was drawn, from which Codes were written and compiled on Arduino IDE. IF statements were copiously used. Proteus was used to simulate the design while the project was implemented on breadboard. Liquid Crystal Display function displays the level of water on the 16 × 2 LCD thus: Very Low, Low, High and Very High. An option of I2C LCD codes was written thus providing advantage of using only two analog input A4 and A5 pins instead of 4 to 8 pins in other configurations thereby allowing other pins dedicated for other tasks. The design recommends improvements in the area of sump control so that its low water level could disable pump thus preventing it from running dry. Protection devices like circuit breaker overload and phase failure relays are recommended in order to prolong the life of the Water pump. It is believed that the design will go in long way in educating power electronic engineers in the arts of design using Arduino; also, a mass production of the device will accentuate Small and Medium Enterprises SMEs in developing countries with its concomitant economic advantages.

Corresponding Author:

Akinwole OO,
Department of Electrical/Electronic Engineering,
Federal Polytechnic,
Ado-Ikare Rd, Ado-Ekiti, Nigeria.
Email: oyeakin2003@yahoo.com

1. INTRODUCTION

One of the results of modern age is the increase in devices that work in automatic fashion. Daily man tends to find a way to simplify his activities thereby solving problems. Many times are wasted operating systems manually; also cost of employing operators can eat deep into the company finance. The design, apart from solving this problem, also delves into trending Arduino design methodology. Appliance or equipment manufacturers and designers alike have keyed-in into this in order to remain relevant. Automatic systems reduce number of workers to attend to industrial process; the resultant is the drastic reduction in wages and losses which will undoubtedly dovetail into more profits. Thus automatic system supports lean manufacturing where customers are satisfied at low and affordable cost [1]. The aforementioned is made possible by continuous improvement in design techniques and ingenuous usage of electronics components and building blocks. Microcontrollers offer better solution, being computers on single chips; enable production of embedded smart systems which are prevalent everywhere today [2]. A designer just need to master it, learn their instruction sets and write codes that make them work.

This project will allow the level of water in a tank to be maintained automatically, that is, unattended to; it uses Arduino microcontroller, codes were written in order for it to perform as water level controller.

2. METHODOLOGY

The tasks to be performed were listed while flowchart for the design was drawn. Input pins hosting three float switches were connected to Arduino pins 1 to 3 using pull down 10k resistors. Four LEDs level indicators indicating "Water Level Very High, High, Low, and Very Low" were connected to Arduino pins 9, 8,7and 6 respectively. IF statements were in the codes. The project was simulated using Proteus while the implementation was done on breadboard. An LCD displaying water levels was included using six Arduino pins. An option of I2C LCD was included thus enabling only two analog pins A4 and A5 to be engaged; where A4 is connected to SDA and A5 to SCL , that is, i2C data and clock respectively [3].

2.1. System's flowchart

Algorithms that explain the sequence of operations and steps for the design were written thus enabling the diagrammatical representation of flow of process through the agency of flowcharting [4]. This approach simplifies the writing of system's codes (Figure 1).

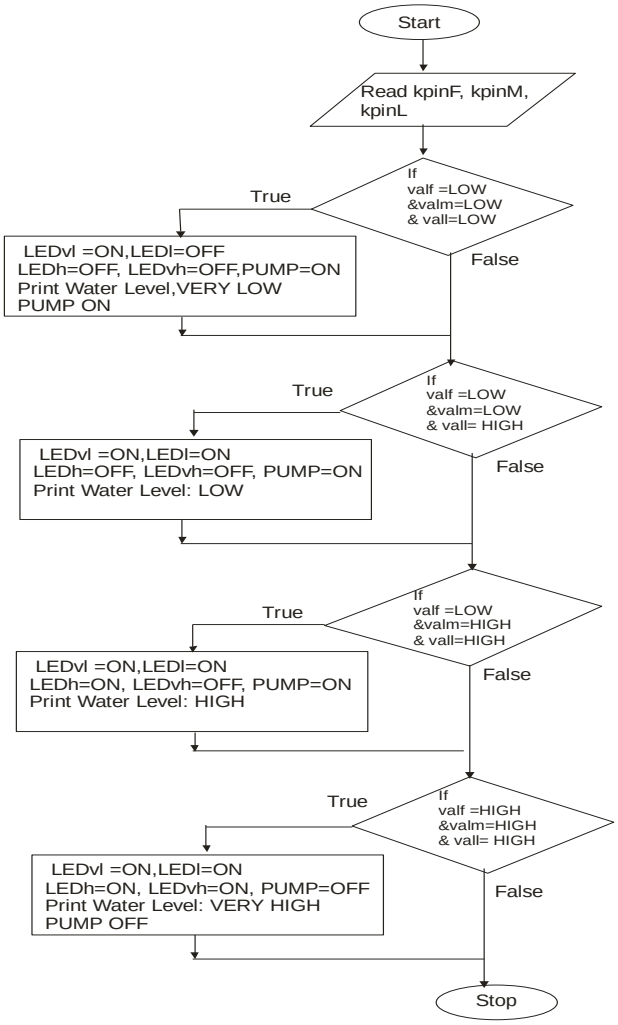

Figure 1. Flow chart

Step 1: Read the digital values (HIGH, +5V or LOW, 0V) at microcontroller pins kpinF, kpinM and kpinL and store their values at valf, valM and vall respectively. (F, M and L represent Full, Medium and Low respectively).

Step 2: If valf, valm and vall have LOW (0V) values, Light Emitting Diodes LEDs LEDvl ON (lit), LEDl OFF, LEDh OFF, LEDvh OFF; Water Pump Switched ON, Print on LCD "Water Level: VERY LOW PUMP ON" else go to step 3. (vl, l, h, vh represent very low, low, high and very high respectively).

Step 3: If valf and valm are LOW (0V) values and vall HIGH, Light Emitting Diodes LEDs LEDvl ON (lit), LEDl ON, LEDh OFF, LEDvh OFF; Water Pump ON, Print on LCD "Water Level: LOW PUMP ON" else go to step 4.

Step 4: If valf LOW while valm and vall HIGH (+5V), Light Emitting Diodes LEDs LEDvl ON (lit), LEDl ON, LEDh ON, LEDvh OFF; Water Pump ON, Print on LCD "Water Level: HIGH PUMP ON" else go to step 5.

Step 5: If valf, valm and vall are HIGH (+5V), Light Emitting Diodes LEDs LEDvl ON (lit), LEDl ON, LEDh ON, LEDvh ON; Water Pump switch OFF, Print on LCD "Water Level: VERY HIGH PUMP OFF".

2.2. Arduino uno microcontroller

A Microcontroller can be referred to as a computer on a chip. It parades input and output pins. Unlike a Microprocessor, it includes processor with memory location for storing programs written in C language [5].

Arduino Uno board, Figure 2, has an Atmega 328 on it as the main Microcontroller. Using this unit does not require extra money for a Programmer. Codes or sketches and compilations are done on its Integrated Development Environment IDE This advantage makes prototyping of embedded system easier by engineers and electronic systems enthusiasts. The board has six (A0 to A5) analog inputs to handle analog signals and fourteen I/O pins for input and output functions [4].

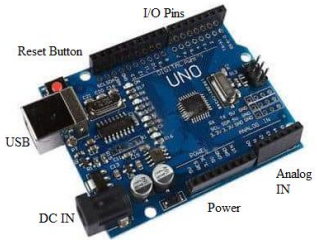

Figure 2. Arduino uno board

2.3. Design of a relay controlling switch

Transistor works as a switch when it is operated at its extreme regions, that is, at cut-off and saturation [6] At saturation, Common-Emitter Voltage, V_{CE} =0; and Collector Current $I_C = \frac{V_{CC}}{R_C}$ As shown in Figure 3.

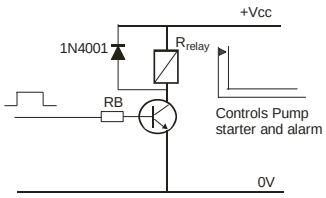

Figure 3. A transistor switch

While at cut-off, Collector Supply V_{CC} is equal to Collector Emitter Voltage V_{CE} (V_{CE} = V_{CC}) and I_C = 0.

Using Kirchhoff's Voltage Law in the collector circuit (1),

$$V_{CC} = I_C R_{relay} + V_{CE} \quad (1)$$

Using a switching and linear amplification transistor 2N2222,

$I_{C\ max}$ = 800mA at I_C= 150mA, V_{CE} = 10V while $h_{FE\ min}$ =100 and h_{FEmax} = 300 [7]
V_{HIGH} = V_{CC} = 9V, R_{relay} = 400 **Ω**, h_{FE}= 100, Relay operating voltage = 9V
By using 300**Ω**, $I_C = \frac{V_{CC}}{R_{relay}} = \frac{9}{400}$= 22.5mA
Base current, $I_B = \frac{I_C}{hFE} = \frac{22.5mA}{100}$ = 0.23mA
For saturation, I_B> 0.23mA.
Let I_B = 2.5mA.
Arduino V_{HIGH} = 5V, V_{BE} = 0.7V
$R_B = \frac{V_{HIGH}-V_{BE}}{I_B} = \frac{5-0.7}{2.5mA}\frac{4.3}{3mA}$ = 1720**Ω**
Preferred Value = 1.5k**Ω**

2.4. The system's codes

The system's codes below were punctuated with comments thereby given clear understandings of the application of Arduino microcontroller's syntax in one hand and providence of needed assistance in future software improvements. In order to write to the LCD, 4 bits mode was adopted in which LCD's D4-D7 are engaged (figure 4). The syntax is LiquidCrystal lcd (RS, EN, D4, D5, D6, D7), in this case: LiquidCrystal lcd (4,5,10,11,12,13) are connected to RS, EN, D4, D5, D6 and D7 respectively [4].

```
#include <LiquidCrystal.h>
#include <Wire.h>

LiquidCrystallcd (4,5,10,11,12,13);

int kpinF=1; // pin 1 is set as water full sensor pin
int kpinM=2;// pin 2 is set as water medium level sensor pin
int kpinL=3;// pin 3 is set as water low level sensor pin
int MotorBuzzer=0; // pin 0 is set as water pump and buzzer drive pin
int LEDvh=9; // pin 9 is set as Very High water level indicator pin
int LEDh=8; // pin 8 is set as High water level indicator pin
int LEDl=7; // pin 7 is set as Low water level indicator pin.
int LEDvl=6;//pin 6 is set as Very Low water level indicator pin.

int valf=0;//The value of the memory location sensing Full water level initialized to
zero
int valm=0;// The value of the memory location sensing Medium water level initialized
to                zero
int vall=0;// The value of the memory location sensing Low water level initialized to
zero

void setup()
{
lcd.begin(16,2); // for 16 X 2 LCD module
lcd.setCursor(0,0);
lcd.print("AUTO WATER LEVEL");
lcd.setCursor(0,1);
lcd.print(" CONTROLLER");
delay(1000);
pinMode(kpinF, INPUT);//kpinF set as Input pin
pinMode(kpinM, INPUT);//kpinM set as input pin
pinMode(kpinL, INPUT);// kpinL set as input pin
pinMode(LEDvh, OUTPUT);//LEDvh set as output pin
pinMode(LEDh, OUTPUT);// LEDh set as output pin
pinMode(LEDl, OUTPUT);// LEDl set as output pin
pinMode(LEDvl, OUTPUT);// LEDvl set as output pin
pinMode(MotorBuzzer, OUTPUT);// MotorBuzzer set as output pin
}
void loop()
{
valf=digitalRead(kpinF);// Read kpinF and store the digital value in valf
valm=digitalRead(kpinM); // Read kpinM and store the digital value in valm
vall=digitalRead(kpinL);// Read kpinL and store the digital value in vall

if((valf==LOW)&&(valm==LOW)&&(vall==LOW))
{
digitalWrite(LEDvl, HIGH);
digitalWrite(LEDl, LOW);
digitalWrite(LEDh, LOW);
digitalWrite(LEDvh, LOW);
```

```
digitalWrite(MotorBuzzer, HIGH);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("WATER LEVEL");
lcd.setCursor(0,1);
lcd.print("VERY LOW PUMP ON");
delay(500);
}
if((valf==LOW)&&(valm==LOW)&&(vall==HIGH))
{
digitalWrite(LEDvl, HIGH);
digitalWrite(LEDl, HIGH);
digitalWrite(LEDh, LOW);
digitalWrite(LEDvh, LOW);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("WATER LEVEL");
lcd.setCursor(0,1);
lcd.print("LOW");
delay(500);
}
if((valf==LOW)&&(valm==HIGH)&&(vall==HIGH))
{
digitalWrite(LEDvl, HIGH);
digitalWrite(LEDl, HIGH);
digitalWrite(LEDh, HIGH);
digitalWrite(LEDvh, LOW);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("WATER LEVEL");
lcd.setCursor(0,1);
lcd.print("HIGH");
delay(500);
}
if((valf==HIGH)&&(valm==HIGH)&&(vall==HIGH))
{
digitalWrite(LEDvl, HIGH);
digitalWrite(LEDl, HIGH);
digitalWrite(LEDh, HIGH);
digitalWrite(LEDvh, HIGH);
digitalWrite(MotorBuzzer, LOW);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("WATER LEVEL");
lcd.setCursor(0,1);
lcd.print("VHIGH PUMP OFF");
delay(500);
}
}
```

2.5. Codes using I2C

In order to employ i2C configuration and benefit from its gain, the above code can be modified by including its code using #include<LIquidCrystal_I2C.h> and LiquidCrystal_I2C lcd(0x3F,2,1,0,4,5,6,7); 0x3F is the address of the i2C module [4]. Other syntax should be noted. The void () loop codes which the Arduino executes continuously remain the same as in the section above.

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>

LiquidCrystal_I2C lcd(0x3F,2,1,0,4,5,6,7);

int kpinF=1; // pin 1 is set as water full sensor pin
int kpinM=2;// pin 2 is set as water medium level sensor pin
int kpinL=3;// pin 3 is set as water low level sensor pin
int MotorBuzzer=0; // pin 0 is set as water pump and buzzer drive pin
int LEDvh=9; // pin 9 is set as Very High water level indicator pin
int LEDh=8; // pin 8 is set as High water level indicator pin
int LEDl=7; // pin 7 is set as Low water level indicator pin.
int LEDvl=6;//pin 6 is set as Very Low water level indicator pin.

int valf=0;//The value of the memory location sensing Full water level initialized to
zero
int valm=0;// The value of the memory location sensing Medium water level initialized
to              zero
```

```
int vall=0;// The value of the memory location sensing Low water level initialized to
zero

void setup()
{
lcd.begin(16,2); // for 16 X 2 LCD module
lcd.setBacklightPin(3,POSITIVE);
lcd.setBacklight(HIGH);
lcd.setCursor(0,0);
lcd.print("AUTO WATER LEVEL");
lcd.setCursor(0,1);
lcd.print(" CONTROLLER");
delay(1000);

pinMode(kpinF, INPUT);//kpinF set as Input pin
pinMode(kpinM, INPUT);//kpinM set as input pin
pinMode(kpinL, INPUT);// kpinL set as input pin
pinMode(LEDvh, OUTPUT);//LEDvh set as output pin
pinMode(LEDh, OUTPUT);// LEDh set as output pin
pinMode(LEDl, OUTPUT);// LEDl set as output pin
pinMode(LEDvl, OUTPUT);// LEDvl set as output pin
pinMode(MotorBuzzer, OUTPUT);// MotorBuzzer set as output pin
}
```

2.6. Simulation using proteus

The project was wired in Proteus [8] for simulation, (Figure 4). Proteus is a simulation and design software developed by labcenter electronics [9]. Single pole single throw switch was employed to represent float switches sensing low, medium and full water levels. They drive kpinL, kpinM and kpinF respectively. The pins are pull down using three 10kΩ resistors. A closed switch applies +5V across its respective pin.

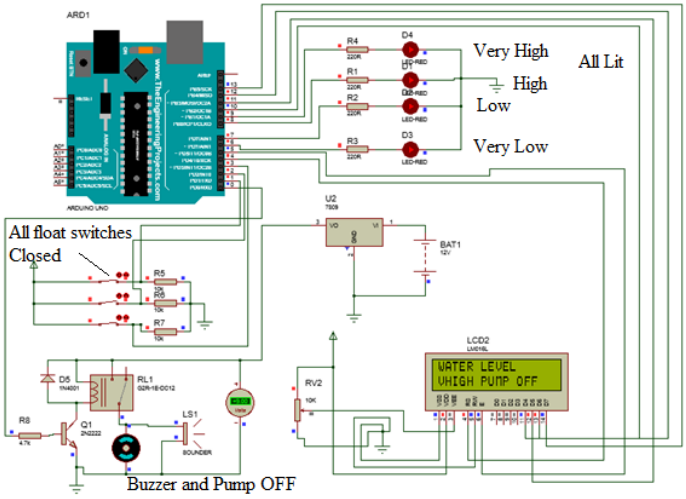

Figure 4. Simulation in proteus environment with general LCD connection

3. THE COMPLETE CIRCUIT

Water sensors sensing Low, Medium and Full levels of water tank are connected to Arduino pins 1, 2 and 3 respectively. Pin 0 drives the water pump and buzzer while Light Emitting Diodes indicating Very Low (vl), Low(l), High(h) and Very High(vh) are connected to pins 6, 7, 8, and 9 respectively.

Current flowing in the LEDs are minimized by connecting 220Ω resistors in series with them; As their cathodes are grounded (0V), the arrangement is active High, that is, when the conditions for any of the LED to be illuminated is met, its anode terminal connected to an Arduino pin goes High, thereby allowing current to flow through it, as shown in Figure 5.

Input Pins (Sensors) are tied down using 10KΩ resistors; so +5VDC is connected through the float switches to each input pin. When the water level is below any switch, its contacts becomes opened, its Arduino pin sees 0V but as water reaches that float device the switch closes thus presenting +5V to its Arduino pin. The values in the pins are stored in valf (full), valm (medium) and vall (low) memory location.

digitalRead functions are used to read the values, if all the values are LOW, it means the water level is Very Low, Pin 0 that drives the water pump goes HIGH (+5V)

The HIGH level drives a transistor switch [10] using a switching transistor 2N2222, Its Vcc is connected to the output terminal of a +9V fixed voltage regulator, LM7809. The base resistor 4.7Ω provides a potential difference in the base circuit which allows current to flow into the transistor. A transistor connected in common emitter configuration act as an Inverter [11], that is, +5V at the base produces 0V at the collector. Since a 9V relay with operating coil having ohmic (DC) resistance of 400Ω is the load, a 9V is effectively across the coil thus cause it to operate thereby closing its normally open contact which is wired in series with Electric pump control circuit. A piezoelectric device wired across the relay produces sound.
A power diode 1N4001 acting as a suppression diode [11] connected across the relay protects the transistor against the back emf.

As the level of the water in the tank rises with val, valm and valf become HIGH, all the LEDs are lit, Arduino pin 0 becomes 0V, the transistor switch cuts off, relay de-energizes, hitherto closed relay contact becomes opened thereby disengaging power into the water pump control circuit. The buzzer goes off. The Water Pump is automatically switched off.

The LCD circuit uses the 12C module to reduce complexity of display circuit. Instead of using six Arduino I/O pins 4, 5, 10, 11, 12 and 13; only two analog pins A4 and A5 are used. They are connected to SDA and SCK respectively. The I2C module is soldered to the back of the 16 x 2 LCD unit. (Figure 6)

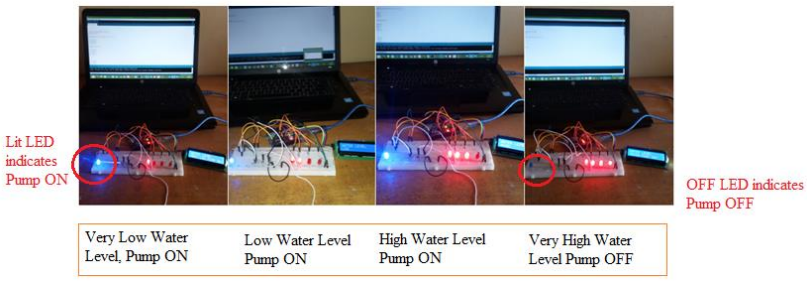

Figure 5. Implementation on breadboard. water float switches are realised by closing successive contacts using wire jumpers.

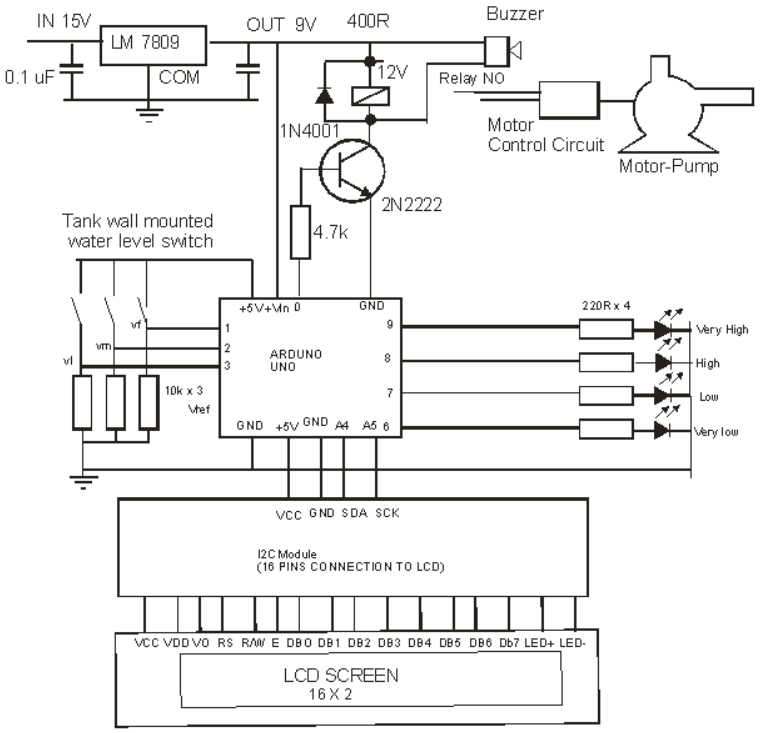

Figure 6. Working circuit for i2c lcd connection: the i2c backpack was soldered to the display while only 2 arduino pins were used.

4. RESULTS

The result has helped to solve problems of manual operations of water pumping system. It has also deviated from usage of PIC microcontrollers into trending Arduino platform. The result of implementation is shown in Figure 5; the HIGH (+5V) driving the buzzer and motor pump was represented by a blue LED, so when it is lit, the pump and buzzer were ON and vice versa, float switches were realized by manual closing of a set of four contacts. At Very Low water level, one LED was lit, at Low water level, two LEDs were ON, at High tank level, three LEDs were lit and at Very High tank level all the four LEDs were in ON positions; at this juncture, the water pump went off, indicated by the OFF blue LED.

5. CONCLUSION

The paper has succeeded in highlighting the simplicity of using Arduino Microcontroller in the designs of Power Electronic systems. In order to prolong the life of the pump, a Direct- on-Line starter could be engaged in starting it. In this arrangement, an overload relay would be wired into the motor main circuit thus preventing overloads from burning its stator windings. The following recommendations are made:

1. The design can be improved by having means of sensing low water level in the sump to prevent the pump from running dry.
2. An inclusion of a well selected circuit breaker will protect the motor main circuit from short circuit.
3. To further protect the water pump, an overload relay and a phase failure relay can be included in the starter unit to protect it from single phasing, wrong phase sequence and power supply imbalance.

REFERENCES

[1] Reliable Plant, "Benefits of automation in lean manufacturing," Retrieved from: fromwww.reliableplant.com/lean-manufacturing-automation, Accessed online 2019.
[2] Bates, M.P., "Programming 8-bit PIC microcontrollers in C with interactive hardware simulation," NewnessUSA, 2008.
[3] Adafruit Learning System. I2C/SPI LCD Backpack. Retrieved online 2018 from www.learnadafruit.com/i2c-spi-lcd-backpack
[4] Akinwole, OO, Oladimeji, TT. "Design and implementation of Arduino microcontroller based automatic lighting control with I2C LCD Display," *J. Electr Electron Syst*, vol. 7, pp. 258, 2018.
[5] Arduino Tutorialpoint, pp. 24-35, 2016, Retrieved online from www.tutorialpont.com, 2018.
[6] Electronics Hub., "Transistor as Switch," Retrieved from: www.electronicshub.org/transistor-as-switch/, Sep 2018.
[7] Philips Semiconductors Datasheet on 2N2222 Switching Transistor, Retrieved from: www.semiconductors.philips.com, 2017.
[8] Labcenter Electronics, Proteus User manual, Intelligent Schematic Input System, www.ele.uva.es, 2002
[9] Narasimha, R. "Proteus PCB Design and Simulation Software-Introduction", www.circuittoday.com,2017
[10] Akinwole, OO, "Design and implementation of an electric cooker control as a means of preventing domestic fire incidence" *Journal of Engineering Reaearch and Reports*, vol. 4, no. 3, pp. 1-9, 2019.
[11] Horowitz, P., Hill, W., *The Art of Electroni.cs*, Cambridge University Press, 1989.