Electronic and Embedded Systems
University of Sri Jayewardenepura

SMART NUTRITION SUPPLIER FOR CROPS

PROJECT REPORT

AS2019963
W.M.Virajith

Acknowledgments

I want to express my sincere appreciation to Dr. M. D. R. Perera, Senior Lecturer in Computer Science, Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Gangodawila, Nugegoda, Sri Lanka, for his kind advice, assistance, and helpful recommendations. I pledge to support the embedded system project wholeheartedly.

Content

Purpose

For Agricultural crops, It is necessary to supply the correct amount of water and suitable amount of nutritions, such as Nitrogen, Potassium, Phosphorus etc.

In the practical world, there are no much options to optimize this activity with the technology.

Therefore, I decided to create this Smart nutrition supply system to grow the crops in optimal environment.

Introduction

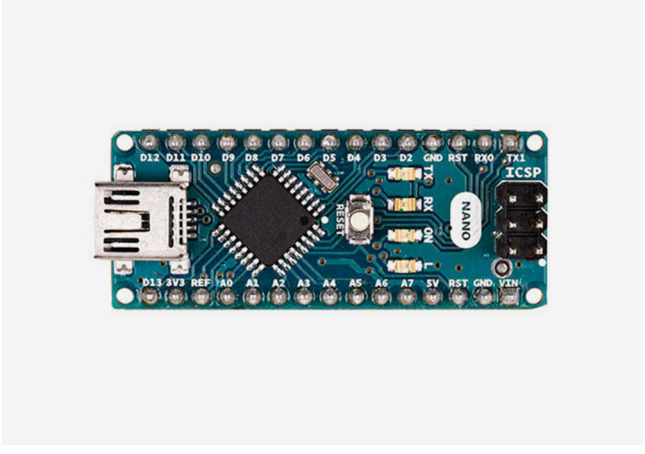
In this project, the system identifies the moisture of the soil, percentage of the Nitrogen, Potassium, Phosphorus and supply them to the soil directly when the elements are in low level.

Initially the moisture sensor measures the moisture of the soil and then the color sensor will measures the N, P, K levels of the soil.

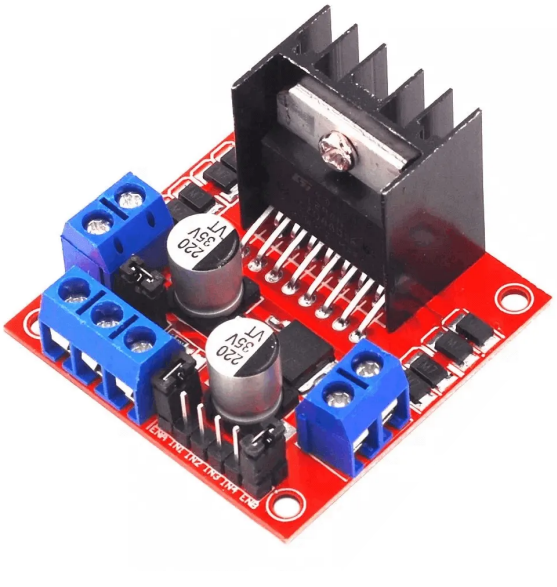
After measure

Components

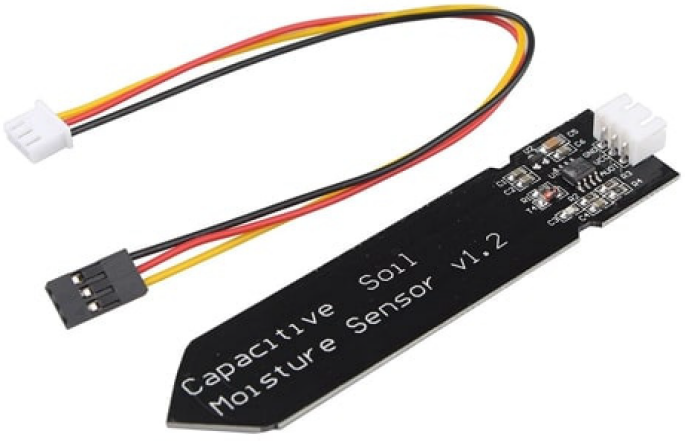
Following components are used to develop the system.



Arduino NANO

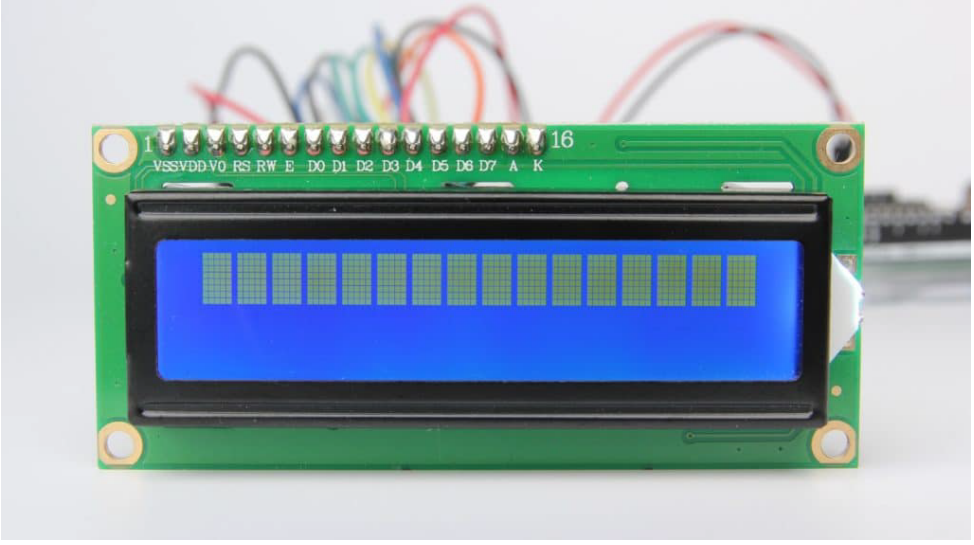


Motor driver



Moisture sensor

Color sensor

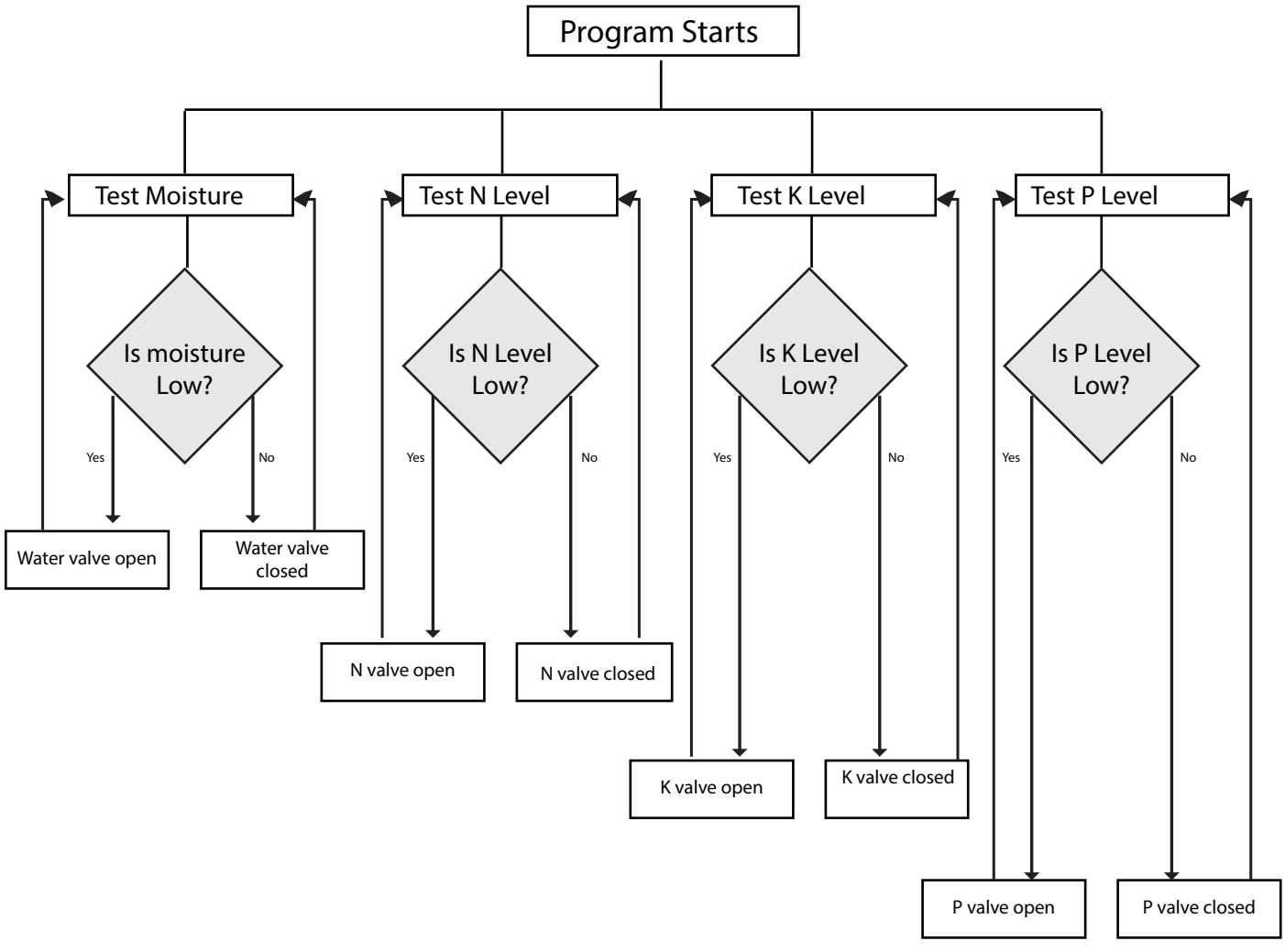


LCD Screen

LCD Screen

Design Overview

Flow Chart



Code

```
/* Soil moisture sensor O/P pin */
const int sensor_pin = A1;

/* Mortor In pins*/
#define waterOut 2
#define NOut 10
#define POut 11
#define KOut 12

// Define color sensor pins

#define S0 4
#define S1 5
#define S2 6
#define S3 7
#define sensorOut 8

// Color senser Calibration Values
int redMin = 20; // Red minimum value
int redMax = 203; // Red maximum value
int greenMin = 23; // Green minimum value
int greenMax = 312; // Green maximum value
int blueMin = 20; // Blue minimum value
```

```
// Variables for Color Pulse Width Measurements
int redPW = 0;
int greenPW = 0;
int bluePW = 0;

// Variables for final Color values
int redValue;
int greenValue;
int blueValue;

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// set the LCD address to 0x27 for a 16 chars and 2
line display
LiquidCrystal_I2C lcd(0x27, 20, 4);
void setup() {
  lcd.init();
  // Print a message to the LCD.
  lcd.backlight();
  lcd.setCursor(2, 1);
  lcd.print("SMART NUTRITION");
  lcd.setCursor(3, 2);
  lcd.print("SUPPLY SYSTEM");
  lcd.setCursor(0, 3);
  delay(2000);
  lcd.clear();
  lcd.setCursor(2, 1);
  lcd.print("loading");
  delay(2000);
  lcd.clear();
```

```
// Set S0 - S3 as outputs
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(S3, OUTPUT);

  // Set Sensor output as input
  pinMode(sensorOut, INPUT);

  // Set Frequency scaling to 20%
  digitalWrite(S0, HIGH);
  digitalWrite(S1, LOW);

  // Setup Serial Monitor
  Serial.begin(9600);
}

void loop() {

  //Moisture sensor
  float moisture_percentage;
  int sensor_analog;
  sensor_analog = analogRead(sensor_pin);
  moisture_percentage = ( 100 - ( (sensor_analog /
1023.00) * 100 ) );
  Serial.print("Moisture Percentage = ");
  Serial.print(moisture_percentage);
  Serial.print(" % \n\n");
  delay(1000);
  lcd.setCursor(13, 0);
  lcd.print(moisture_percentage);
  lcd.setCursor(18, 0);
  lcd.print("%");
```

```
/*Water pump function*/
  if (moisture_percentage < 25) {
    digitalWrite(2, HIGH);
    Serial.print("Water pump ON\n");

    //LCD Output - water pump status
    lcd.setCursor(1, 0);
    lcd.print("W pump - ON");

  }
  else {
    digitalWrite(2, LOW);
    Serial.print("Water Pump OFF\n");
  }

  //LCD Output - water pump status
  lcd.setCursor(0, 0);
  lcd.print("W Pump - OFF");



  // Read Red value
  redPW = getRedPW();
  // Map to value from 0-255
  redValue = map(redPW, redMin, redMax, 255, 0);
  // Delay to stabilize sensor
  delay(200);

  // Read Green value
  greenPW = getGreenPW();
  // Map to value from 0-255
  greenValue = map(greenPW, greenMin, greenMax,
255, 0);
  // Delay to stabilize sensor
  delay(200);
```

```
  // Print output to Serial Monitor
  Serial.print("Red = ");
  Serial.print(redValue);
  Serial.print(" - Green = ");
  Serial.print(greenValue);
  Serial.print(" - Blue = ");
  Serial.println(blueValue);

  /*N pump function*/
  if ((246 < redValue) && (redValue < 249) && (149 <
blueValue) && (blueValue < 197) && (187 < greenVal-
ue) && (greenValue < 217)) {
    digitalWrite(10, HIGH);
    Serial.print("N pump ON\n");

    //LCD Output - N pump status
    lcd.setCursor(0, 1);
    lcd.print("N pump - ON");
    lcd.setCursor(13, 1);
    lcd.print("LOW");


  }

  else {
    digitalWrite(10, LOW);
    Serial.print("N Pump OFF\n");

    //LCD Output - N pump status
    lcd.setCursor(0, 1);
    lcd.print("N Pump - OFF");
    lcd.setCursor(13, 1);
    lcd.print("HIGH");

  }
```

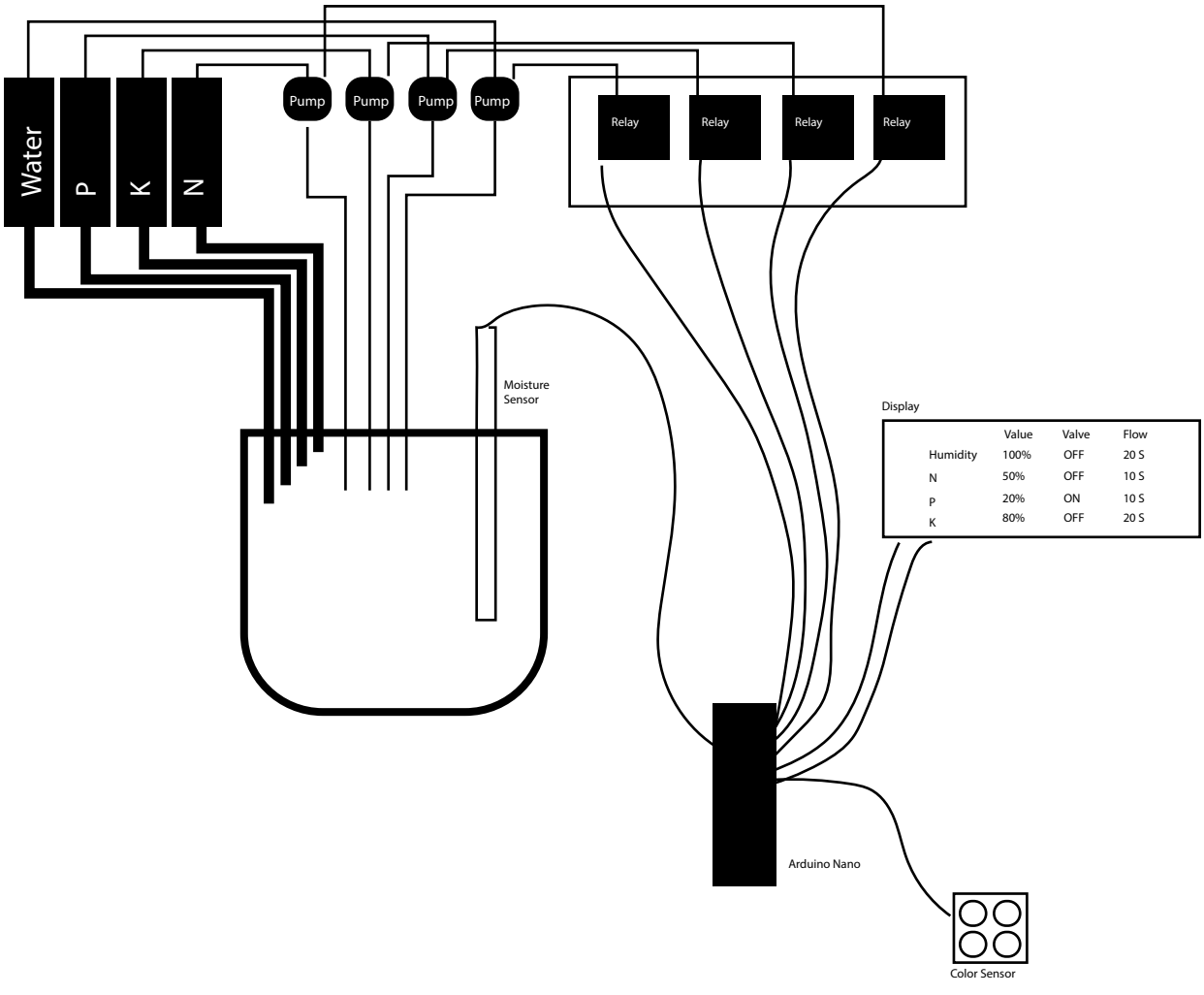
```
  // Print output to Serial Monitor
  Serial.print("Red = ");
  Serial.print(redValue);
  Serial.print(" - Green = ");
  Serial.print(greenValue);
  Serial.print(" - Blue = ");
  Serial.println(blueValue);

  /*N pump function*/
  if ((246 < redValue) && (redValue < 249) && (149 <
blueValue) && (blueValue < 197) && (187 < greenVal-
ue) && (greenValue < 217)) {
    digitalWrite(10, HIGH);
    Serial.print("N pump ON\n");

    //LCD Output - N pump status
    lcd.setCursor(0, 1);
    lcd.print("N pump - ON");
    lcd.setCursor(13, 1);
    lcd.print("LOW");


  }

  else {
    digitalWrite(10, LOW);
    Serial.print("N Pump OFF\n");

    //LCD Output - N pump status
    lcd.setCursor(0, 1);
    lcd.print("N Pump - OFF");
    lcd.setCursor(13, 1);
    lcd.print("HIGH");

  }
```

```
  else {
    digitalWrite(5, HIGH);
    Serial.print("P pump HIGH\n");

    //LCD Output - P pump status
    lcd.setCursor(0, 2);
    lcd.print("P Pump - OFF");
    lcd.setCursor(13, 2);
    lcd.print("HIGH");
  }

  /*K pump function*/
  if ((167 < redValue) && (redValue < 208) && (165 <
blueValue) && (blueValue < 206) && (199 < greenVal-
ue) && (greenValue < 225)) {
    digitalWrite(12, HIGH);
    Serial.print("K pump ON\n");

    //LCD Output - K pump status
    lcd.setCursor(0, 3);
    lcd.print("K Pump - ON");
    lcd.setCursor(13, 3);
    lcd.print("LOW");
  }

  else {
    digitalWrite(12, LOW);
    Serial.print("K pump OFF\n");
    //LCD Output - K pump status
    lcd.setCursor(0, 3);
    lcd.print("K Pump - OFF");
    lcd.setCursor(13, 3);
    lcd.print("HIGH");
  }
}
```

```
// Function to read Red Pulse Widths
int getRedPW() {
  // Set sensor to read Red only
  digitalWrite(S2, LOW);
  digitalWrite(S3, LOW);
  // Define integer to represent Pulse Width
  int PW;
  // Read the output Pulse Width
  PW = pulseIn(sensorOut, LOW);
  // Return the value
  return PW;

}

// Function to read Green Pulse Widths
int getGreenPW() {

  // Set sensor to read Green only
  digitalWrite(S2, HIGH);
  digitalWrite(S3, HIGH);
  // Define integer to represent Pulse Width
  int PW;
  // Read the output Pulse Width
  PW = pulseIn(sensorOut, LOW);
  // Return the value
  return PW;

}
```

```
// Function to read Blue Pulse Widths
int getBluePW() {
  // Set sensor to read Blue only
  digitalWrite(S2, LOW);
  digitalWrite(S3, HIGH);
  // Define integer to represent Pulse Width
  int PW;
  // Read the output Pulse Width
  PW = pulseIn(sensorOut, LOW);
  // Return the value
  return PW;
}
```

Block Diagram



Issues

Accuracy of the sensors are not so accurate.
So the sensors have to calibrate often to get a successful output.

References

https://arduino-tutorials.net/

https://docs.arduino.cc/tutorials/

https://www.tutorialspoint.com/arduino/index.htm