VIT-AP
UNIVERSITY

ENGINEERING CLINICS - 2

AQUA CONTROLLER - Wireless pump control for Field Irrigation

Using Aurdino and GSM

Project Review - 2

(Working Prototype)

Guide

Prof Dr. Visalakshi Annep

Group Members:

21BCE9584- **GUDURU DASTAGIRI**	21BEC7211 - **MUDIMI CHARAN**
21BCE9671 – **KARUMANCHI BHAVYA SRI**	21BCE9149 – **NERALLA PAVAN**
21BCE9561 - **MOLLETI MONIKA SAI**	21BCE9406 - **AKURATHI VARSHITHA**

The Mobile-Controlled Water Pump System for Farmers is designed to address the challenges faced by farmers in remote areas, where fields are far from their homes and power supply is limited. The project aims to enable farmers to remotely control the water pumps in their fields using their mobile phones. This system provides convenience and efficiency in managing irrigation, ensuring optimal water usage, and reducing the need for manual intervention.

Hardware Components:

- Arduino Uno or similar microcontroller board
- GSM module for mobile communication
- 16x2 LCD display
- Relay module for controlling the water pump
- Power supply
- Beard bord
- Jumper wires
- Water pump

Arduino Uno:

Arduino Uno is a microcontroller board based on the ATmega328P chip. It is one of the most widely used and beginner-friendly boards in the Arduino family. The board comes with digital and analog input/output pins that allow you to connect various sensors, actuators, and other electronic components to create interactive projects. It also has a USB interface, power jack, and an onboard voltage regulator.

Key features of Arduino Uno:

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Digital I/O Pins: 14 (of which 6 can be used as PWM outputs)
- Analog Input Pins: 6
- Flash Memory: 32 KB
- SRAM: 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz

Arduino Uno is programmed using the Arduino IDE, which is an open-source development environment that simplifies the process of writing, compiling, and uploading code to the board.

GSM Module:

GSM modules enable communication over the GSM network. They allow you to send and receive SMS messages, make phone calls, and access the internet (limited capabilities compared to modern smartphones). GSM modules usually come in the form of small boards that can be easily integrated into various projects.

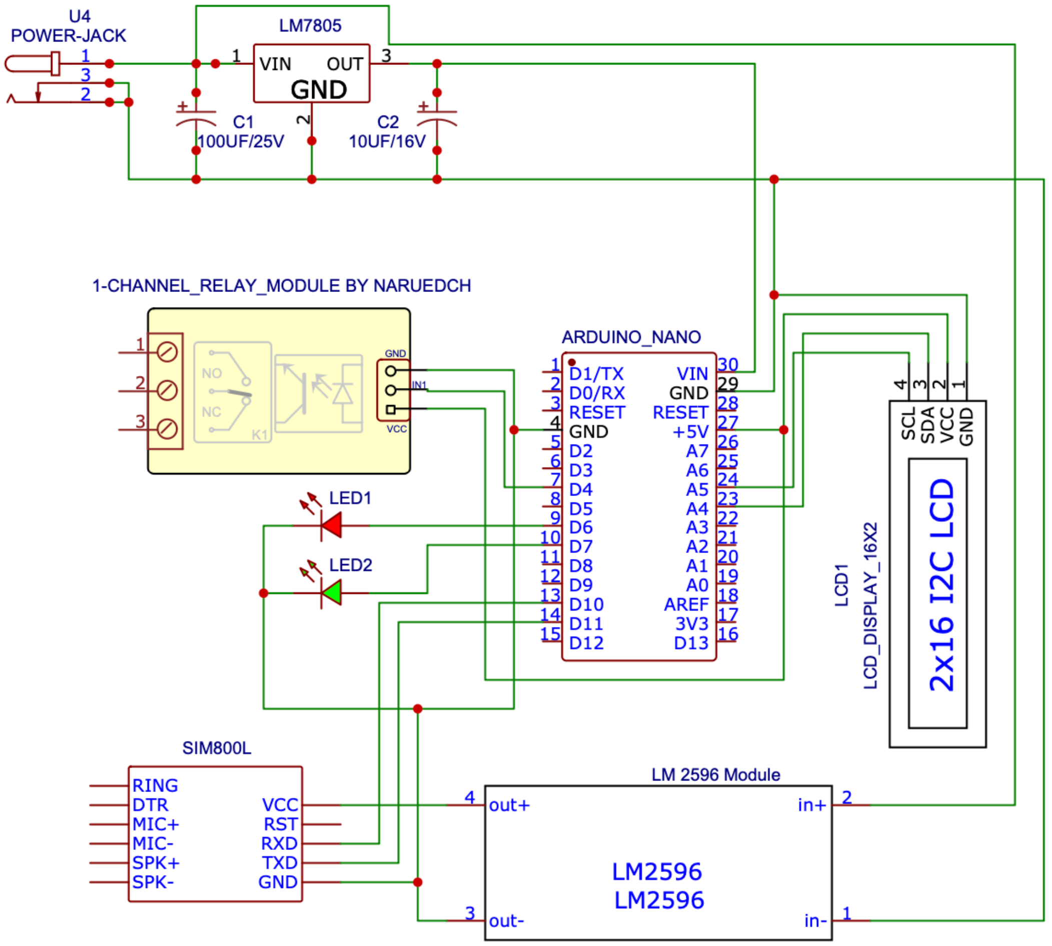
Pump Control Relay:

An electronic switch used to turn the water pump ON or OFF based on commands received from the

GSM module.

Hardware Setup:

Our project deals with Arduino uno which is the main component works like a mini CPU and GSM module allow you to send and receive SMS messages, make phone calls, and access the internet (limited capabilities compared to modern smartphones).Connect the GSM module to the Arduino Uno using the UART (serial) interface. Usually, you'll connect TX (transmit) from the module to RX (receive) on the Arduino and vice versa. If the pump operates on a different voltage or requires a higher current than the Arduino can provide, use a relay module to control the pump's power supply. Connect the relay module to one of the Arduino's digital pins. Connect the pump or the motor driver to the relay module or directly to the Arduino, depending on the voltage and current requirements. Ensure all connections are secure, and double-check the wiring before proceeding.

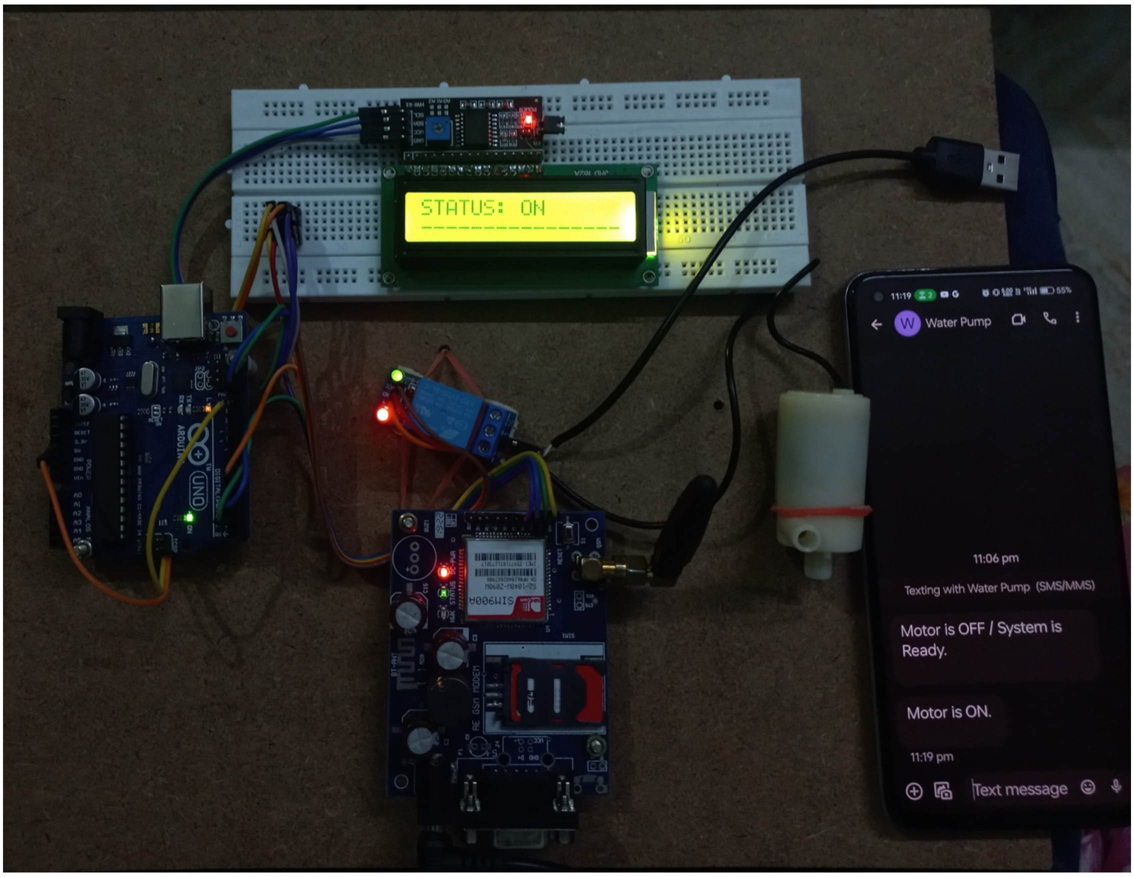


Software Setup

1. Begin by installing the Arduino IDE on your computer.
2. Connect the Arduino board to your computer using the USB cable.
3. Launch the Arduino IDE and paste the provided code for the Mobile-Controlled Water Pump System.
4. Verify the code for any potential errors to ensure smooth operation.
5. Upload the verified code to the Arduino board, allowing it to take control of the system.
6. Ensure that the SIM card is securely inserted into the GSM module and has been activated correctly.
7. Power on the system and double-check that the GSM module establishes a connection with the mobile network

Operations:

1. Once the setup is on, it tests the eeporm and performs the required diagnosis, And sends a message that “motor is on/ System is Ready”.
2. When a farmer makes a call from their mobile phone to the GSM module, the system verifies the number (It only allows authenticated users).and turn On the pump and an acknowledgement will be sent.



Motor is OFF / System is Ready.

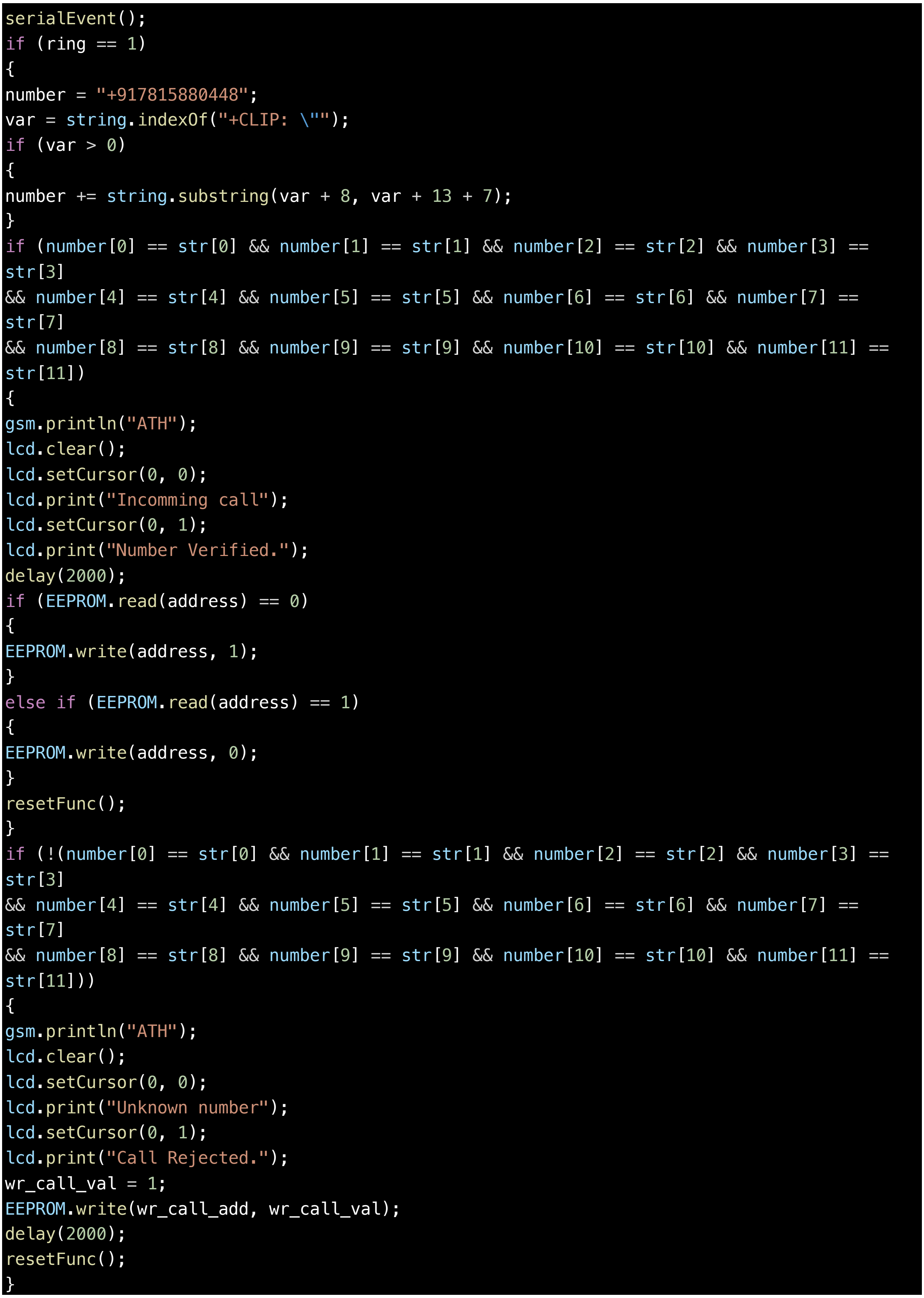
Motor is ON.

3. The Arduino board processes the incoming calls sequences and extracts the command to turn the water pump ON or OFF.
4. Based on the received command, the Arduino board controls the relay module, which in turn switches the water pump ON or OFF.
5. The status of the water pump (ON or OFF) is displayed on the LCD display.

Code:

```
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>
#include <EEPROM.h>
SoftwareSerial gsm(2,3); // RX, TX
LiquidCrystal_I2C lcd(0x27, 16, 2);

int address = 0;
int wr_call_add = 1;
int f_address = 2;
int eeprom_dead = 3;
int wr_call_val = 0;
int f_value = 70;
int value = 0;
int ring = 0;
int i = 0;
int var = 0;
int eeprom_tst = 0;
int eeprom_tst_ack = 0;
int tst_var = 100;
int wr_call_tst = 100;
int eeprom_dead_val = 0;
String number = "";
String string = "";
const int output = 5;
const int red = 7;
const int buzzer = 4;
const int green = 6;
boolean wait = true;
boolean at_flag = 1;
boolean net_flag = 1;
char str[]="+917815880448";
void setup()
{
Serial.begin(9600);
lcd.init();
lcd.backlight();
gsm.begin(9600);
pinMode(output, OUTPUT);
pinMode(red, OUTPUT);
pinMode(buzzer, OUTPUT);
pinMode(green, OUTPUT);
if (EEPROM.read(f_address) != f_value)
{
EEPROM.write(f_address, f_value);
EEPROM.write(address, value);
EEPROM.write(wr_call_add, wr_call_val);
eeprom_dead_val = 0;
EEPROM.write(eeprom_dead, eeprom_dead_val);
}
if (EEPROM.read(eeprom_dead) == 1)
{
while (true)
{
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("EEPROM Error.");
lcd.setCursor(0, 1);
lcd.print("System Disabled.");
delay(1500);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Reload the code");
lcd.setCursor(0, 1);
lcd.print("with new address");
delay(1500);
}
}
if (EEPROM.read(address) == 0)
{
digitalWrite(output, HIGH);
digitalWrite(red, HIGH);
digitalWrite(buzzer, LOW);
delay(2000);
digitalWrite(green, LOW);
gsm_init();
lcd.clear();
if (EEPROM.read(wr_call_add) == 0)
{
lcd.setCursor(0, 0);
lcd.print("Sending SMS");
lcd.setCursor(0, 1);
lcd.print("Acknowledgement.");
delay(1000);
gsm.println("AT+CMGF=1");
delay(500);
gsm.print("AT+CMGS=");
gsm.print("\"");
gsm.print(str);
gsm.println("\"");
delay(1000);
gsm.println("Motor is OFF / System is Ready.");
delay(100);
gsm.println((char)26);
}
if (EEPROM.read(wr_call_add) == 1)
{
wr_call_val = 0;
EEPROM.write(wr_call_add, wr_call_val);
}
}
if (EEPROM.read(address) == 1)
{
eeprom_test();
output_begin();
}
}
void(* resetFunc) (void) = 0;
void loop()
{
```



```
serialEvent();
if (ring == 1)
{
number = "+917815880448";
var = string.indexOf("+CLIP: \"");
if (var > 0)
{
number += string.substring(var + 8, var + 13 + 7);
}
if (number[0] == str[0] && number[1] == str[1] && number[2] == str[2] && number[3] ==
str[3]
&& number[4] == str[4] && number[5] == str[5] && number[6] == str[6] && number[7] ==
str[7]
&& number[8] == str[8] && number[9] == str[9] && number[10] == str[10] && number[11] ==
str[11])
{
gsm.println("ATH");
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Incomming call");
lcd.setCursor(0, 1);
lcd.print("Number Verified.");
delay(2000);
if (EEPROM.read(address) == 0)
{
EEPROM.write(address, 1);
}
else if (EEPROM.read(address) == 1)
{
EEPROM.write(address, 0);
}
resetFunc();
}
if (!(number[0] == str[0] && number[1] == str[1] && number[2] == str[2] && number[3] ==
str[3]
&& number[4] == str[4] && number[5] == str[5] && number[6] == str[6] && number[7] ==
str[7]
&& number[8] == str[8] && number[9] == str[9] && number[10] == str[10] && number[11] ==
str[11]))
{
gsm.println("ATH");
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Unknown number");
lcd.setCursor(0, 1);
lcd.print("Call Rejected.");
wr_call_val = 1;
EEPROM.write(wr_call_add, wr_call_val);
delay(2000);
resetFunc();
}
```

```
}
if (EEPROM.read(address) == 0)
{
serialEvent();
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("System Standby");
lcd.setCursor(0, 1);
lcd.print("Status: OFF");
delay(1000);
}
else if (EEPROM.read(address) == 1)
{
lcd.setCursor(0, 0);
lcd.print("STATUS: ON");
lcd.setCursor(0, 1);
lcd.print("----------------");
}
}
void gsm_init()
{
lcd.print("System booting....");
lcd.setCursor(0, 1);
lcd.print("initiating......");
delay(1500);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Checking Module");
lcd.setCursor(0, 1);
lcd.print("Connectivity....");
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Module");
lcd.setCursor(0, 1);
lcd.print("Connection: OK");
delay(1500);
eeprom_test();
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("System is Ready");
lcd.setCursor(0, 1);
lcd.print("----------------");
delay(1500);
}
void serialEvent()
{
while (gsm.available())
{
char read_char = gsm.read();
string += read_char;
i++;
```

```
if (string[i - 4] == 'R' && string[i - 3] == 'I' && string[i - 2] == 'N' && string[i - 1]
== 'G' )
{
ring = 1;
}
}
}
void output_begin()
{
digitalWrite(red, LOW);
digitalWrite(output,LOW);
digitalWrite(buzzer, HIGH);
delay(2000);
digitalWrite(buzzer, LOW);
digitalWrite(green, HIGH);
if (EEPROM.read(wr_call_add) == 0)
{
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Sending SMS");
lcd.setCursor(0, 1);
lcd.print("Acknowledgement.");
delay(1000);
gsm.println("AT+CMGF=1");
delay(500);
gsm.print("AT+CMGS=");
gsm.print("\"");
gsm.print(str);
gsm.println("\"");
delay(1000);
gsm.println("Motor is ON.");
delay(100);
gsm.println((char)26);
}
if (EEPROM.read(wr_call_add) == 1)
{
wr_call_val = 0;
EEPROM.write(wr_call_add, wr_call_val);
}
lcd.clear();
}
void eeprom_test()
{
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Testing EEPROM");
lcd.setCursor(0, 1);
lcd.print("Memory.");
delay(1500);
eeprom_tst = EEPROM.read(address);
eeprom_tst_ack = EEPROM.read(wr_call_add);
```

```
EEPROM.write(address, tst_var);
EEPROM.write(wr_call_add, wr_call_tst);
if (EEPROM.read(address) == tst_var && EEPROM.read(wr_call_add) == wr_call_tst)
{
EEPROM.write(address, eeprom_tst);
EEPROM.write(wr_call_add, eeprom_tst_ack);
if (EEPROM.read(address) != eeprom_tst || EEPROM.read(wr_call_add) != eeprom_tst_ack)
{
digitalWrite(output,HIGH);
digitalWrite(red, HIGH);
digitalWrite(buzzer, LOW);
delay(2000);
digitalWrite(green, LOW);
eeprom_dead_val = 1;
EEPROM.write(eeprom_dead, eeprom_dead_val);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("EEPROM Test is");
lcd.setCursor(0, 1);
lcd.print("Unsuccessful.");
delay(1500);
}
else if (EEPROM.read(address) == eeprom_tst && EEPROM.read(wr_call_add) ==
eeprom_tst_ack)
{
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("EEPROM Test is");
lcd.setCursor(0, 1);
lcd.print("Successful.");
delay(1500);
}
}
else if (EEPROM.read(address) != tst_var || EEPROM.read(wr_call_add) != wr_call_tst)
{
digitalWrite(output,HIGH);
digitalWrite(red, HIGH);
digitalWrite(buzzer, LOW);
delay(2000);
digitalWrite(green, LOW);
eeprom_dead_val = 1;
EEPROM.write(eeprom_dead, eeprom_dead_val);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("EEPROM Test is");
lcd.setCursor(0, 1);
lcd.print("Unsuccessful.");
delay(1500);
}
}
void eeprom_sms()
```

```
{
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Sending SMS");
lcd.setCursor(0, 1);
lcd.print("Acknowledgement");
delay(1000);
gsm.println("AT+CMGF=1");
delay(500);
gsm.print("AT+CMGS=");
gsm.print("\"");
gsm.print(str);
gsm.println("\"");
delay(1000);
gsm.println("EEPROM error. System disabled. Please reload the code with new address.");
delay(100);
gsm.println((char)26);
{
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("EEPROM Error.");
lcd.setCursor(0, 1);
lcd.print("System Disabled.");
delay(1500);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Reload the code");
lcd.setCursor(0, 1);
lcd.print("with new address");
delay(1500)
```