“Motion Sensing Alarm With Keypad & Password – Security System ”

PROJECT REPORT

Submitted for CAL in B.Tech Microprocessor and Interfacing (CSE2006)

By

C MAHESHVAR- 17BCE1172
ROHIT SUBRAMANIAN - 17BCE1291

Slot: D1

Name of faculty: PROF. SASIPRIYA P.

(SCHOOL OF COMPUTER SCIENCE ENGINEERING: SCSE)

VIT UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ▪ CHENNAI
www.vit.ac.in

April 2019

1. ABSTRACT

Arduino, being an open-source microcontroller, offers engineers and non-engineers an efficient platform with tools to work on a wide range of digital projects. To exploit this technology and strengthen our fundamentals of such microcontrollers and interfacing we chose a project in which we aim to build a basic Arduino based security system which is interfaced with a keypad and buzzer. The following report walks through the entire development process and gives an overview of both the hardware and software development stages.

2. INTRODUCTION

A countless number of projects can be done with the help of an Arduino, from simple digital sensor connections and testing to more complex integrated IoT applications, it's all made possible with this simple open-source tool. Out of these several possible projects we chose to build a small scale implementation of a home security system using an arduino.

The Arduino micro-controller allows us to interact with our environment using sensors and actuators. For our project, we'll primarily focus on motion detection using a sensor and buzzer and its turning on or off using a 4x4 keypad . Whenever motion is detection , the buzzer starts beeping and the LED light is switched on . On entering the correct password on the keypad, the buzzer is stopped and the motion detection sensor is reset.

To develop this project, Arduino IDE environment will be used for Arduino programming .

Given below is the outline of the development stages:

-Assembly of physical structure of the circuit
-Arduino Programming
-Module integration and testing

3. COMPONENTS

- Arduino MEGA
- Motion Detection Sensor
- Buzzer
- 4x4 Keypad
- Breadboard and Wires
- LED lights
- Resistors and Capacitor
- Power Source (Batteries and Power Bank)

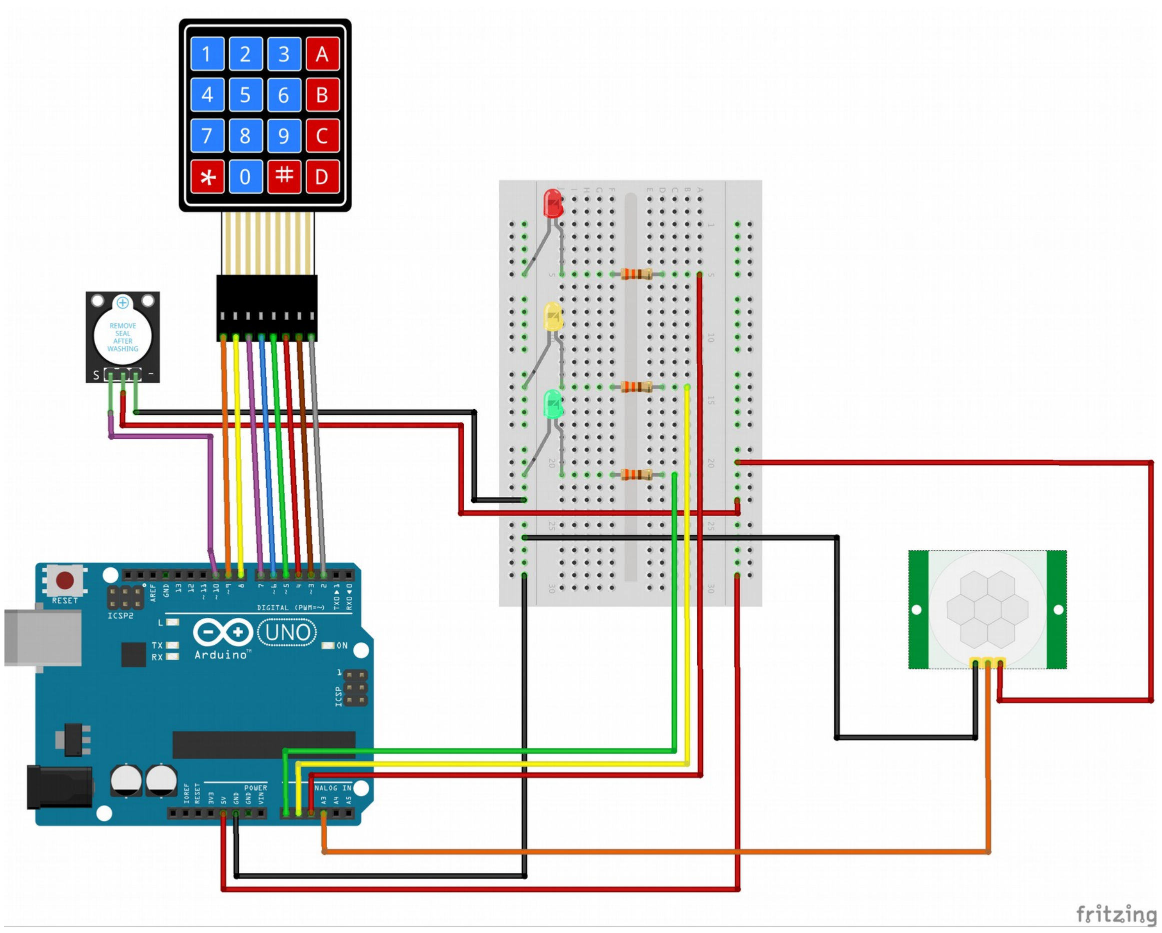
4. ARDUINO

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs (sensors) and turn it into an output (actuators). You can tell your board what to do by sending a set of instructions to the microcontroller on the board. The boards features serial communications interfaces are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or breadboards and other circuits.

The Arduino MEGA ADK is a microcontroller board based on the ATmega2560. It has a USB host interface to connect with Android based phones, based on the MAX3421e IC. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs,
4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

The Arduino integrated development environment (IDE) is a cross-platform application that is written in Java. It includes a code editor with features enhancing coding experience a provides a simple one-click mechanism to compile and upload programs to an Arduino board. The Arduino IDE supports the languages C/C++ using special rules of code structuring. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

5. SCHEMATICS



6. ARDUINO PROGRAMMING

In the arduino IDE, apart from declaring the pins we define two functions setup() and loop(). The setup() function is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup() function will only run once, after each powerup or reset of the Arduino board. After creating the setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing our program to change and respond. It is used to actively control the Arduino board.

In the setup() function, we configured all our pins as output pins and set the data rate in bits per second (baud) to 9600 for serial data transmission. Serial.begin(9600) opens the serial port and sets data rate to 9600 bps.

In the loop() function, we check for a condition to see if there is any incoming data using if(Serial.available>0). Serial.available gets the number of bytes (characters) available for reading from the serial port. This is data that's already arrived and stored in the serial receive buffer (which holds 64 bytes). If data is present then the incoming serial data is read using Serial.read(). Serial.write() then writes binary data to the serial port. This data is sent as a byte or series of bytes.

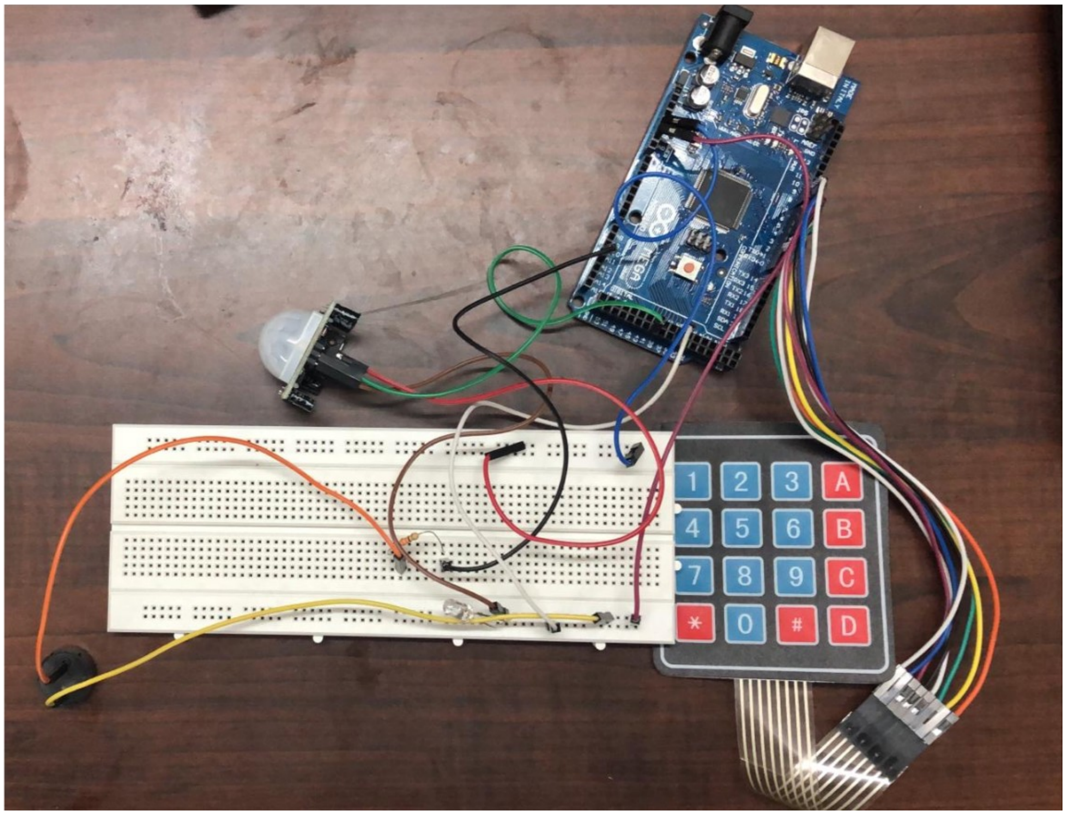
In Arduino, the analogWrite function allows you to generate a PWM (pulse-width modulation) wave in a pin. This function takes a value between 0 and 255 and doesn't work on all pins in Arduino. In Arduino Uno, it works on pins 3, 5, 6, 9, 10 and 11. The speed of the motor depends on value that was passed to the analogWrite function. If you pass 0, then the motor will stop and if you pass 255 then it will run at full speed. If you pass a value between 1 and 254, then the speed of the motor will vary accordingly.

The digitalWrite() function writes a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode(), its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH, 0V (ground) for LOW.

7. ARDUINO CODE

```
#include <Keypad.h>

const int buzzer = 39; //buzzer to arduino pin 39
int led = 24;              // the pin that the LED is atteched to
int sensor = 41;              // the pin that the sensor is atteched to
int state = LOW;              // by default, no motion detected
int val = 0;
int i=0,j=0,flag=0,k=0;
char password[4]={'1','2','3','4'};
char pass[4];

const byte ROWS = 4; //four rows
const byte COLS = 4; //four columns
//define the cymbols on the buttons of the keypads
char hexaKeys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {9, 8, 7, 6}; //connect to the row pinouts of the
keypad
byte colPins[COLS] = {5, 4, 3, 2}; //connect to the column pinouts of the
keypad

//initialize an instance of class NewKeypad
Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins,
colPins, ROWS, COLS);


void setup(){
  pinMode(buzzer, OUTPUT); // Set buzzer - pin 39 as an output
   pinMode(led, OUTPUT);      // initalize LED as an output
  pinMode(sensor, INPUT);    // initialize sensor as an input
  Serial.begin(9600);
}

void loop(){
  val = digitalRead(sensor);   // read sensor value
  if (val == HIGH) {           // check if the sensor is HIGH
    digitalWrite(led, HIGH);   // turn LED ON
  delay(100);                // delay 100 milliseconds

  if (state == LOW) {
    Serial.println("Motion detected!");
    state = HIGH;
  }
    while(k<=5)
    {
    tone(buzzer, 500); // Send 500Hz sound signal...
    delay(300);        // ...for 1 sec
    noTone(buzzer);     // Stop sound...
    delay(300);
    k=k+1;             // ...for 1sec
    }

    char customKey = customKeypad.getKey();  //Password
  if (customKey !=NO_KEY){
      pass[i]=customKey;
      i=i+1;
  }
 if(i==4 and j==0)
 {
   j=1;
   printo();
   return;
 }
 }
 else if(val == LOW) {
    digitalWrite(led, LOW); // turn LED OFF
    delay(200);             // delay 200 milliseconds

    if (state == HIGH){
      Serial.println("Motion stopped!");
      state = LOW;       // update variable state to LOW
  }
  }
   char customKey = customKeypad.getKey();

  if (customKey){
    Serial.println(customKey);
  }
}

void printo()
{
  Serial.println(pass);
for(i=0;i<4;i++)
{
  if(pass[i]!=password[i])
      flag=flag+1;
}
  if(flag==0)
     {
      Serial.println("RIGHT PASSWORD");
      digitalWrite(led, LOW);
      delay(5000);
      k=0;
      loop();
      j=0;
      i=0;
     }
  else
  {
    Serial.println("WRONG PASSWORD");
    k=0;
    loop();
    j=0;
    i=0;
  }
}
```

8. CIRCUIT



9. CONCLUSION

This project has successfully come through to implement a simple security system using a keypad. However this project can be improvised by using higher quality motion sensor and extra layers of protection such as biometric sensor (finger print sensor). Also this project can be connected with a bluetooth module in order to send a notification to a mobile app in case of anamolies detected

This work presents a low-cost, energy efficient circuit highlighting the operation of basic security operation systems used in homes or banks. A strong grasp of arduino programming and basic digital interfacing has been acquired through the implementation of this arduino project.