# **ΥΠΟΔΡΑΣΗ 3**

# **Προγραμματισμός Αισθητηρίων temperature και humidity**

# **Προγραμματισμός φώτων με έξυπνο ρελέ και μαγνητικής επαφής για πόρτα και παράθυρα και Σύστημα Συναγερμού**

/\* How to use the DHT-22 sensor with Arduino uno

Temperature and humidity sensor

Dev: Εσπερινό ΕΠΑΛ Ιεράπετρας 2020-2021 // Date: 20/05/2021 // \*/

//Libraries

#include <dht.h>

dht DHT;

/\* Use a photoresistor (or photocell) to turn on an LED (LIGHTS) in the dark

Dev: Εσπερινό ΕΠΑΛ Ιεράπετρας 2020-2021 // Date: 20/05/2021 // \*/

//Constants

const int pResistor = A0; // Photoresistor at Arduino analog pin A0

const int ledPin=9; // Led pin at Arduino pin 9

#define DHT22\_PIN 7 // DHT 22 (AM2302) - what pin we're connected to

//Variables

int value; // Store value from photoresistor (0-1023)

float hum; //Stores humidity value

float temp; //Stores temperature value

/\* Arduino Tutorial: How to use a magnetic contact switch

Dev: Εσπερινό ΕΠΑΛ Ιεράπετρας 2020-2021 // Date: 20/05/2021 // // Date: 31/05/2021 \*/

const int buzzer = 3;

const int sensor = 4;

byte dimmableled=5;

int state; // 0 close - 1 open wwitch

/\* PIR sensor tester

\*/

int inputPin = 2; // choose the input pin (for PIR sensor)

int pirState = LOW; // we start, assuming no motion detected

int val = 0; // variable for reading the pin status of photoresistor

void setup()

{

pinMode(ledPin, OUTPUT); // Set lepPin - 9 pin as an output//led ldr

pinMode(pResistor, INPUT);// Set pResistor - A0 pin as an input (optional)

pinMode(sensor, INPUT\_PULLUP);//magnetic contact

pinMode(dimmableled, OUTPUT);// dimmable led

pinMode(inputPin, INPUT); // pir

Serial.begin(9600);

}

void loop()

{

int chk = DHT.read22(DHT22\_PIN);

//Read data and store it to variables hum and temp

hum = DHT.humidity;

temp= DHT.temperature;

//Print temp and humidity values to serial monitor

Serial.print("Humidity: ");

Serial.print(hum);

Serial.print(" %, Temp: ");

Serial.print(temp);

Serial.println(" Celsius");

delay(2000); //Delay 2 sec.

value = analogRead(pResistor);

//You can change value "25"

if (value > 800){

digitalWrite(ledPin, LOW); //Turn led off

}

else{

digitalWrite(ledPin, HIGH); //Turn led on

}

delay(500); //Small delay}

// dimmable led with photoresistor

int reading=analogRead(pResistor);

int bright=reading/8;

delay(500);

analogWrite(dimmableled, bright);

state = digitalRead(sensor);//magnetic contact

/\*if (state == HIGH){

tone(buzzer, 400);

}

else{

noTone(buzzer);

}

delay(200);\*/

val = digitalRead(inputPin); // read input value

if (val == HIGH) { // check if the input is HIGH

if ((pirState == HIGH || state == HIGH)&&(state == HIGH)) {

// we have just turned on

Serial.println("Motion detected!");

// We only want to print on the output change, not state

pirState = LOW;

tone(buzzer, 400);

}

} else {

if ((pirState == LOW || state == LOW) && (state == LOW)){

// we have just turned of

Serial.println("Motion ended!");

// We only want to print on the output change, not state

noTone(buzzer);

}

delay(200);

pirState = LOW;

}

}

# **Προγραμματισμός Συστήματος Συναγερμού:**

/\*

\* Arduino Security and Alarm System

\*

\* by Esperino EPAL Ierapetras,

\*/

#include <LiquidCrystal.h> // includes the LiquidCrystal Library

#include <Keypad.h>

#define buzzer 8

#define trigPin 9

#define echoPin 10

long duration;

int distance, initialDistance, currentDistance, i;

int screenOffMsg =0;

String password="1234";

String tempPassword;

boolean activated = false; // State of the alarm

boolean isActivated;

boolean activateAlarm = false;

boolean alarmActivated = false;

boolean enteredPassword; // State of the entered password to stop the alarm

boolean passChangeMode = false;

boolean passChanged = false;

const byte ROWS = 4; //four rows

const byte COLS = 4; //four columns

char keypressed;

//define the cymbols on the buttons of the keypads

char keyMap[ROWS][COLS] = {

{'1','2','3','A'},

{'4','5','6','B'},

{'7','8','9','C'},

{'\*','0','#','D'}

};

byte rowPins[ROWS] = {14, 15, 16, 17}; //Row pinouts of the keypad

byte colPins[COLS] = {18, 19, 20, 21}; //Column pinouts of the keypad

Keypad myKeypad = Keypad( makeKeymap(keyMap), rowPins, colPins, ROWS, COLS);

LiquidCrystal lcd(1, 2, 4, 5, 6, 7); // Creates an LC object. Parameters: (rs, enable, d4, d5, d6, d7)

void setup() {

lcd.begin(16,2);

pinMode(buzzer, OUTPUT); // Set buzzer as an output

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

}

void loop() {

if (activateAlarm) {

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Alarm will be");

lcd.setCursor(0,1);

lcd.print("activated in");

int countdown = 9; // 9 seconds count down before activating the alarm

while (countdown != 0) {

lcd.setCursor(13,1);

lcd.print(countdown);

countdown--;

tone(buzzer, 700, 100);

delay(1000);

}

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Alarm Activated!");

initialDistance = getDistance();

activateAlarm = false;

alarmActivated = true;

}

if (alarmActivated == true){

currentDistance = getDistance() + 10;

if ( currentDistance < initialDistance) {

tone(buzzer, 1000); // Send 1KHz sound signal

lcd.clear();

enterPassword();

}

}

if (!alarmActivated) {

if (screenOffMsg == 0 ){

lcd.clear();

lcd.setCursor(0,0);

lcd.print("A - Activate");

lcd.setCursor(0,1);

lcd.print("B - Change Pass");

screenOffMsg = 1;

}

keypressed = myKeypad.getKey();

if (keypressed =='A'){ //If A is pressed, activate the alarm

tone(buzzer, 1000, 200);

activateAlarm = true;

}

else if (keypressed =='B') {

lcd.clear();

int i=1;

tone(buzzer, 2000, 100);

tempPassword = "";

lcd.setCursor(0,0);

lcd.print("Current Password");

lcd.setCursor(0,1);

lcd.print(">");

passChangeMode = true;

passChanged = true;

while(passChanged) {

keypressed = myKeypad.getKey();

if (keypressed != NO\_KEY){

if (keypressed == '0' || keypressed == '1' || keypressed == '2' || keypressed == '3' ||

keypressed == '4' || keypressed == '5' || keypressed == '6' || keypressed == '7' ||

keypressed == '8' || keypressed == '9' ) {

tempPassword += keypressed;

lcd.setCursor(i,1);

lcd.print("\*");

i++;

tone(buzzer, 2000, 100);

}

}

if (i > 5 || keypressed == '#') {

tempPassword = "";

i=1;

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Current Password");

lcd.setCursor(0,1);

lcd.print(">");

}

if ( keypressed == '\*') {

i=1;

tone(buzzer, 2000, 100);

if (password == tempPassword) {

tempPassword="";

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Set New Password");

lcd.setCursor(0,1);

lcd.print(">");

while(passChangeMode) {

keypressed = myKeypad.getKey();

if (keypressed != NO\_KEY){

if (keypressed == '0' || keypressed == '1' || keypressed == '2' || keypressed == '3' ||

keypressed == '4' || keypressed == '5' || keypressed == '6' || keypressed == '7' ||

keypressed == '8' || keypressed == '9' ) {

tempPassword += keypressed;

lcd.setCursor(i,1);

lcd.print("\*");

i++;

tone(buzzer, 2000, 100);

}

}

if (i > 5 || keypressed == '#') {

tempPassword = "";

i=1;

tone(buzzer, 2000, 100);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Set New Password");

lcd.setCursor(0,1);

lcd.print(">");

}

if ( keypressed == '\*') {

i=1;

tone(buzzer, 2000, 100);

password = tempPassword;

passChangeMode = false;

passChanged = false;

screenOffMsg = 0;

}

}

}

}

}

}

}

}

void enterPassword() {

int k=5;

tempPassword = "";

activated = true;

lcd.clear();

lcd.setCursor(0,0);

lcd.print(" \*\*\* ALARM \*\*\* ");

lcd.setCursor(0,1);

lcd.print("Pass>");

while(activated) {

keypressed = myKeypad.getKey();

if (keypressed != NO\_KEY){

if (keypressed == '0' || keypressed == '1' || keypressed == '2' || keypressed == '3' ||

keypressed == '4' || keypressed == '5' || keypressed == '6' || keypressed == '7' ||

keypressed == '8' || keypressed == '9' ) {

tempPassword += keypressed;

lcd.setCursor(k,1);

lcd.print("\*");

k++;

}

}

if (k > 9 || keypressed == '#') {

tempPassword = "";

k=5;

lcd.clear();

lcd.setCursor(0,0);

lcd.print(" \*\*\* ALARM \*\*\* ");

lcd.setCursor(0,1);

lcd.print("Pass>");

}

if ( keypressed == '\*') {

if ( tempPassword == password ) {

activated = false;

alarmActivated = false;

noTone(buzzer);

screenOffMsg = 0;

}

else if (tempPassword != password) {

lcd.setCursor(0,1);

lcd.print("Wrong! Try Again");

delay(2000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print(" \*\*\* ALARM \*\*\* ");

lcd.setCursor(0,1);

lcd.print("Pass>");

}

}

}

}

// Custom function for the Ultrasonic sensor

long getDistance(){

//int i=10;

//while( i<=10 ) {

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance = duration\*0.034/2;

//sumDistance += distance;

//}

//int averageDistance= sumDistance/10;

return distance;

}