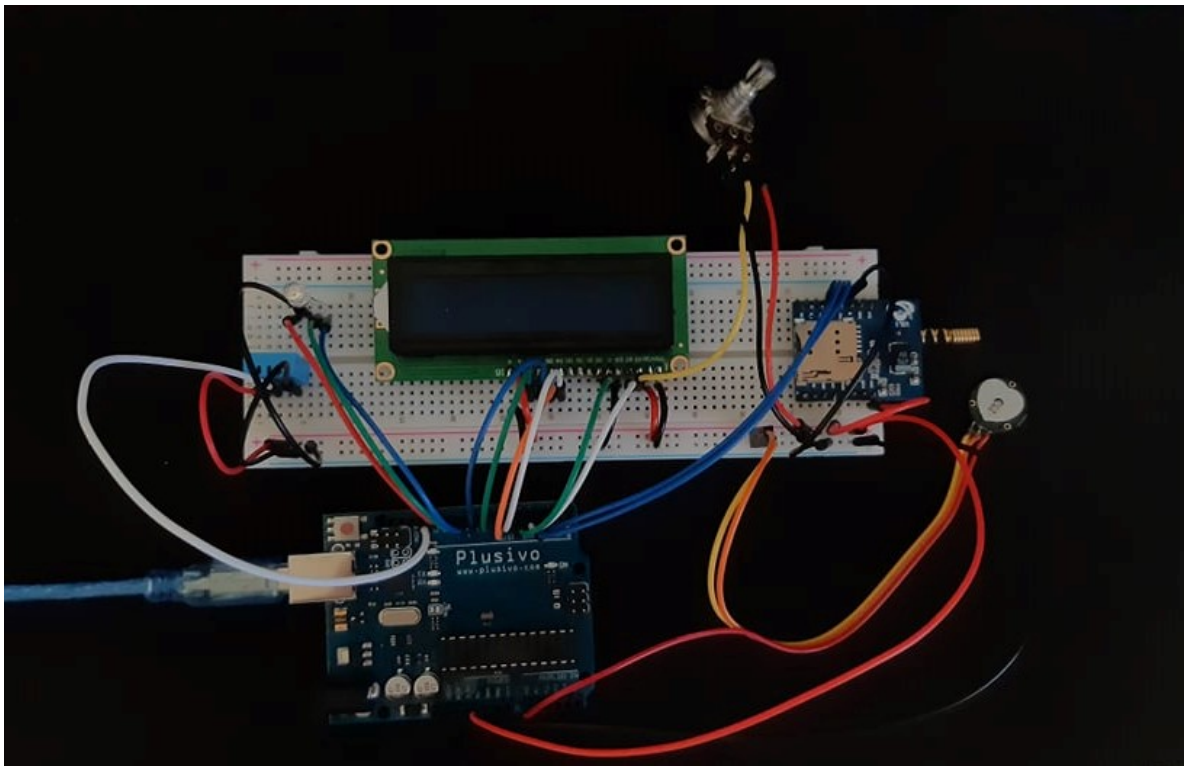
Portable MediKit

Luca Andrei Iulian 3E3

January 17, 2022



1 Purpose

This project has it's mission to solve a real life problem in the medical field. Lets say we have a person under medical observation, neither a nurse or a relative it's available to keep an eye on the patient so we need a portable device easy to use that will notify us in case of any emergency. We can use it in the next situations:

1. In a hospital in a very busy period when no nurse is available for a few patients.
2. At home, when the ill person is under medical observation and a family member have to take care of him and can't be near his proximity all the time.

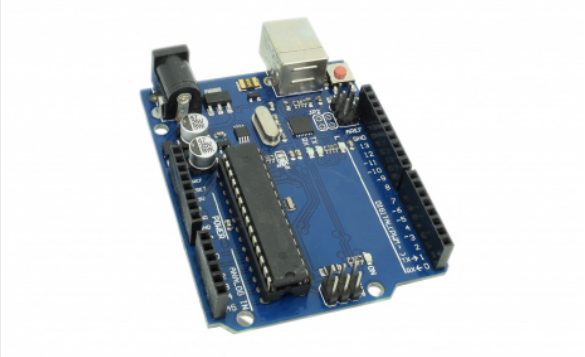
Our device is equiped with a:

- 16x2 LED, perfect for debugging or checking up the value returned by the sensors on the spot
- Temperature and Humidity Sensor
- Pulse Sensor
- RGB Led that will show that something it's wrong lighting the red color also trigering the GSM module that will send an SMS/E-mail
- GSM Module

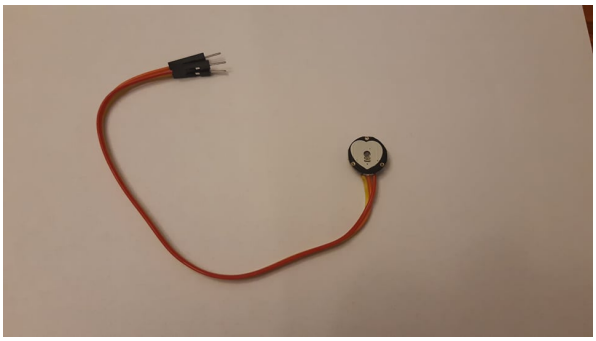
2 Requirements

2.1 Parts Required

- 1 Arduino UNO (or any other clone of it)

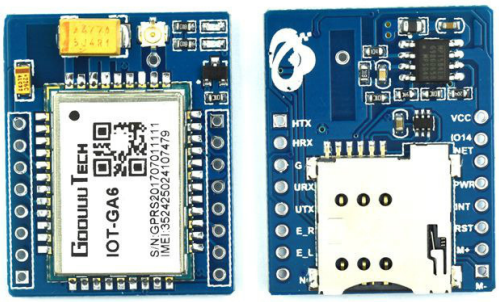


- 1 Pulse/Oxymeter Sensor

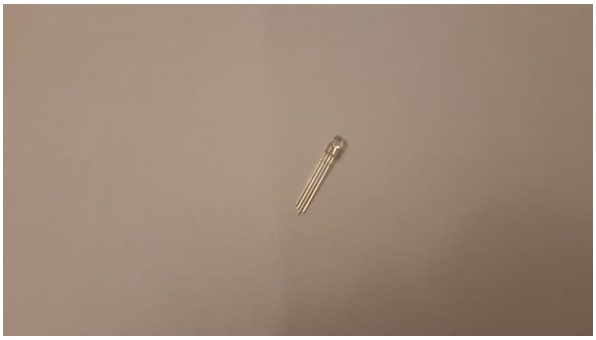


- 1 LCD 16x2

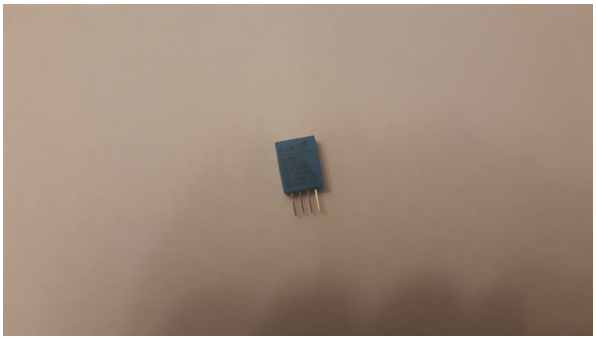
- 1 GSM and GPRS Module GA6-B



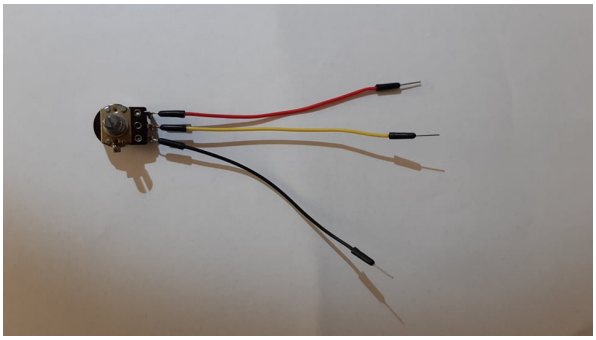
- 1 RGB Led



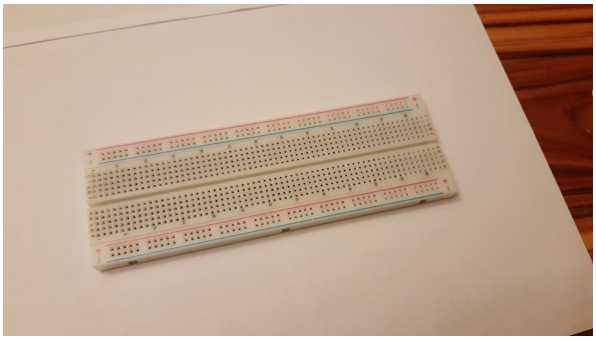
- 1 DHT11 Temperature and Humidity Sensor



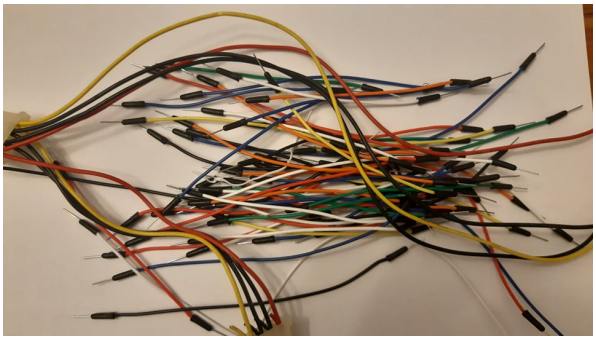
- 1 Potentiometer (20k Ohm or 10k it's fine)



- 1 Breadboard



- Hook-Up Wires



2.2 Costs

The total cost of this device including the arduino clone would be around 26 us dollars.

3 Brief Description

The patient will have the pulse sensor attached to one of his fingers and will permanently monitor his situation along the ambiental temperature and humidity. With the use of the GSM and GPRS model we will send a notification to the person responsible for the wellbeing of the patient and send the current values received from the sensors. On the LED screen and RGB led attached to breadboard we can monitor on the spot the actual situation.

4 The Making of the project and Connections

Firstly, you will have to glue the pins of LED Monitor with your letcon. Then connect:

- Pin 1,16 and 5 are Ground
- Pin 2 and 15 are for 5V Power
- Pin 3 is for the Potentiometer
- Pin 5 goes to pin 5 digital of Arduino
- Pin 11 goes to pin 6 digital of Arduino
- Pin 12 goes to pin 7 digital of Arduino
- Pin 13 goes to pin 9 digital of Arduino
- Pin 14 goes to pin 8 digital of Arduino

Then, I added the Potentiometer which is very easy to install and you can controll the contrast of your LCD screen. One pin goes to GND, one to power and the middle one goes to pin 3 of your screen as I also said up.

For the RGB led which will show the overall status of your sensors I used:

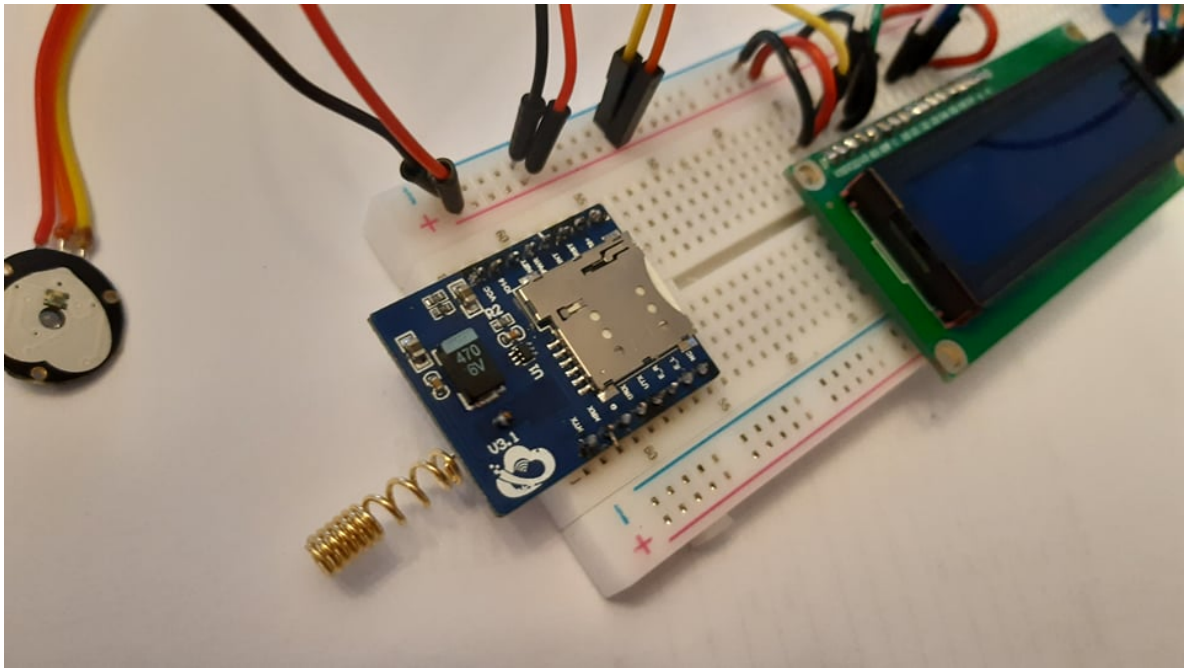
- Pin 12 of Arduino for the RED colour
- Pin 11 of Arduino for the GREEN colour
- Pin 10 of Arduino for the BLUE colour

The Temperature and Humidity even if it has 4 pins attached to the breadboard, only 3 of them we will use. One for GND, one for Power and the one closest to the power one we will attach to Pin 13 Digital of the Arduino.

The Pulse sensor also needs one pin to GND, one pin to 5V and the other one to A0 which is the first analog pin of arduino, through this pin we will receive the raw data from the sensor.

Lastly but not least, the GSM module, after you glued all his 14 pins in order to connect it to the breadbaord, this one needs:

- G pin to GND of Arduino
- VCC pin connected to 5V of Arduino
- UNX pin connected to pin 2 of Arduino
- UTX pin connected to pin 3 of Arduino



4.1 Personal Tips for Implementation

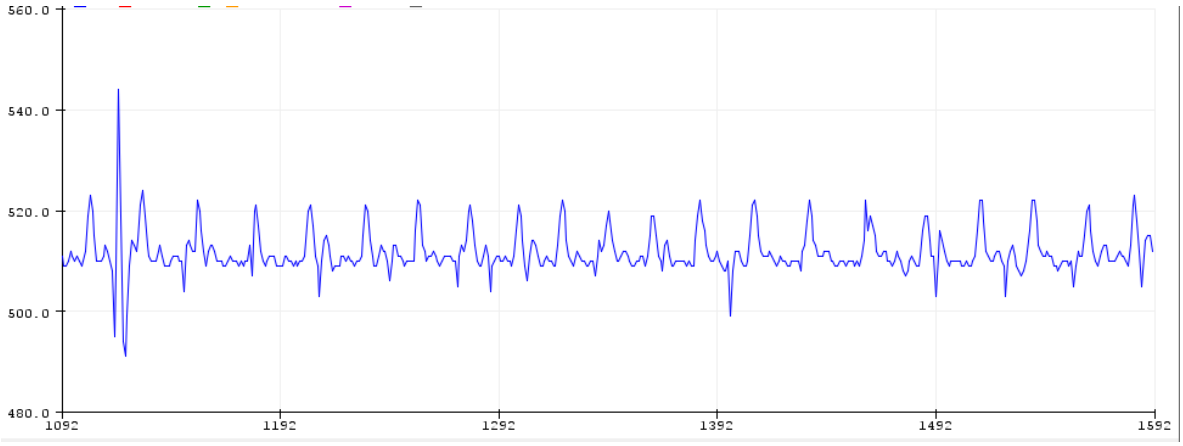
- Being an analog sensor, the pulse sensor is very sensible.That's what analog is about, taking weak signal and amplifying it. So, the signal will get disturbed by the GSM module especially when it will send an SMS. You can solve this by protecting the signal wire so that it doesn't get disturbed so easily, sadly I didn't have time left to do this but if you try to recreate this project I recommend you doing this.

- When I tried to put every pieces together to work the hard reality hit me, I didn't have enough amperage, the usb port of laptops only give 500 mA and this wasn't enough for my lcd screen and gsm module xD. Take a classic usb wire plug it in a phone charger and after splitting the wire put the + wire (red one) in your breadboard so you can charge the high consumers sepparately. From a classic phone charger you will have the same voltage but whith 2A which should be enough and everything will run better.

  WARNING Be careful so that your pieces doesn't receive voltage from both the arduino and the phone charger, this can burn your sensors.

- Keep the connections clean because you will have many devices attached to your arduino and it's important to see as clear as possible.

- Especially for cheap pulse sensors, where the raw data received from the sensor is not very precise, I've tried multiple methods to find the BPM as accurate as possible and this is the best one I've found. First plot the raw data and set a treshold based on where you see the spikes when your heart beats, mine is 515. Then you will observe that for every pulse the sensor sents around 10 values above your treshold, so if you increment a value for every pulse the actual value will be 10x bigger than the actual real times your heart beats. Finally, I recommend you to use millis() so you can count how many beats you received in a minute, you can stop when it reached 30 seconds so you can receive the results faster and then simply multiply by 2 your result but preferably let it at 60 seconds for the best accurate results.



- For the GSM Module, remove the PIN approval for the SIM card from your phone settings before putting it in the module. I spent about three hours thinking it's something wrong with my code or the module it's broken :)

5 The Code

```
#include <SoftwareSerial.h>
#include <PulseSensorPlayground.h>
#include <LiquidCrystal.h>
#include "DHT.h"
#define DHTTYPE DHT11
#define DHTPIN 13 //TH sensor

int redPin = 12;
int greenPin = 11;
int bTemperature, bHumidity = 0;
int threshold = 513; // Determine which signal to "count as a beat" and which to ignore.
int bpm = 0;
int calculatedBPM;
int Signal;
String mood;
float t,h = 0;
const int pulsePin = 0;// PulseSensor connected to ANALOG PIN 0
const int rs = 4, en = 5, d4 = 6, d5 = 7, d6 = 9, d7 = 8;
unsigned long myTime;
unsigned long readTH;
unsigned long THOccurance;
unsigned long lastOccurance;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //Pins of the LED 16X2 Screen
SoftwareSerial sim(3, 2);
DHT dht(DHTPIN, DHTTYPE);

//Left for debugging sim module
void updateSerial()
{
    while (Serial.available()){
      sim.write(Serial.read());
    }
    while(sim.available()){
      Serial.write(sim.read());
    }
```

```
}

void sendSMS(int mode){
    sim.println("AT+CMGF=1");
    sim.println("AT+CMGS=\"+40747295501\"");
    switch(mode){
        case 0:
          sim.println("Check-Up Message");
          break;
        case 1:
          sim.println("DANGER!");
          sim.println("Your patient is in danger or the pulse sensor is not working prop
          break;
        case 2:
          sim.println("DANGER!");
          sim.println("The temperatue it's too extreme!");
          break;
        case 3:
          sim.println("DANGER!");
          sim.println("The humidity it's not in normal parameters!");
          break;
    }
    sim.print("Temperature: ");
    sim.println(t);
    sim.print("Humidity: ");
    sim.println(h);
    sim.print("BPM: ");
    sim.println(calculatedBPM);
    sim.print("Overall mood: ");
    sim.println(mood);
    sim.println((char)26);
}

void checkUp(){
    if (calculatedBPM!=0 && (calculatedBPM < 40 || calculatedBPM > 150)){
        digitalWrite(redPin, HIGH);
        digitalWrite(greenPin, LOW);
        mood = "Red";
        sendSMS(1);
    }
    if ((calculatedBPM >= 40 && calculatedBPM <= 150) || calculatedBPM == 0){
        digitalWrite(greenPin, HIGH);
        digitalWrite(redPin, LOW);
        mood = "Green";
        sendSMS(0);
    }
    if (t < 16 || t > 25){
        mood = "Red";
        sendSMS(2);
    }
    if(h > 50 || h < 30){
        mood = "Red";
        sendSMS(3);
    }
}
```

```
void showTH(){
    lcd.setCursor(0, 0);
    lcd.print("T:");
    lcd.setCursor(2, 0);
    lcd.print(t);
    lcd.setCursor(8, 0);
    lcd.print("H:");
    lcd.setCursor(10, 0);
    lcd.print(h);
//  ALTERNATIVE (but blocking) WAY TO DISPLAY T&H
//    for (int positionCounter = 0; positionCounter < 32; positionCounter++) {
//      lcd.scrollDisplayLeft();
//      delay(400);
//    }
//    for (int positionCounter = 0; positionCounter < 48; positionCounter++) {
//    lcd.scrollDisplayRight();
//    delay(400);
//    }
//    for (int positionCounter = 0; positionCounter < 16; positionCounter++) {
//    lcd.scrollDisplayLeft();
//    delay(200);
//    }
}

void showBPM(){
    lcd.setCursor(0, 1);
    lcd.print("BPM: ");
    lcd.setCursor(5, 1);
    lcd.print(calculatedBPM);
}

void setup() {
    Serial.begin(9600);
    Serial.println(F("DHT11 test!"));
    sim.begin(115200);
    lcd.begin(16, 2);
    dht.begin();
    pinMode(redPin, OUTPUT);
    pinMode(greenPin, OUTPUT);
}

void loop() {
    myTime = millis() - lastOccurance;
    readTH = millis() - THOccurance;
    Signal = analogRead(pulsePin);
    h = dht.readHumidity();          // read humidity
    t = dht.readTemperature();       // read temperature
    if(Signal > threshold){
       bpm++;
    }
    if(myTime > 30000){
       calculatedBPM = bpm/10*2;
       showBPM();
       lastOccurance += 30000;
       bpm = 0;
    }
```

```
    //Good for testing sensor in plotter
    Serial.println(Signal);
    if(readTH > 65000){
      checkUp();
      THOccurance +=65000;
      showTH();
    }
    delay(10);
}
```

6 The Circuit