International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 06 Issue: 12 | December - 2022 Impact Factor: 7.185 ISSN: 2582-3930

IOT BASED TEMPERATURE AND HEART RATE MONITORING SYSTEM

HEALTH MONITORING SYSTEM

Name: M.SARAVANAKUMAR

AUTHOR Name: M.SARAVANAKUMAR

Designation of 1st Author:THIRUTHANGAL-626130

Name of Department of :COMPUTER SCIENCE

Name of organization: INFORMATION TECHNOLOGY

City: SIVAKASI

Country: INDIA

Introduction

The main objective of the project is to monitor the patient pulse rate in a graphical view. The pulse sensor can be attached in the patient finger, which helps to find the current pulse rate of the patient. The pulse rate will get view to the display with the pulse rate and graphical view to the doctor. The changes in the pulse rate can be directly view in the monitor. If the patient pulse rate decrease to below 65 or increase above 80, the doctor should check the status of patient. Each second the patient should be monitor by some person, thus the range of pulse can be checked. In the absence of person, if the pulse rate comes below the range or increases it will create a huge problem for the patient. To overcome this problem, the GSM get attached to the Arduino board and thus when the pulse get increase or decrease it will trigger the GSM to SMS the doctor. For each bed the GSM number will be differ, thus the doctor can came to know the respective patient and the doctor can take necessary step for the patient. The temperature of the patient is monitored by using Mlx90614 Contactless IR Infrared Temperature Sensor Module. If the temperature increases, the GSM triggers to send a current temperature as SMS to the respective doctor as an alert intimation. This process will fulfill the requirement of both doctor and patient in an effective manner.

SYSTEM CONFIGURATION

HARDWARE REQUIREMENT

Processor	:	Dual core
RAM	:	2 GB DDR II
Hard Disk	:	320 GB
Board	:	Arduino UNO (ATmega328 microcontroller)
Sensor	:	Heart Pulse Sensor, Contactless Temperature
SMS Doctor	:	GSM Module

SOFTWARE REQUIREMENT

Operating System	:	Windows 7
Language used	:	Arduino IDE

About Software

Arduino

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board – you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

Arduino Board

Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjuction with abreadboard and some wire. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

- **GND (3)**: Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- **5V (4) & 3.3V (5)**: As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- **Analog (6)**: The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- **Digital (7)**: Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).

- **PWM (8)**: You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).
- **AREF (9)**: Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Reset Button

Just like the original Nintendo, the Arduino has a reset button **(10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' **(11)**. This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

TX RX LEDs

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear – once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs **(12)**. These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit **(13)**. Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This

information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

Voltage Regulator

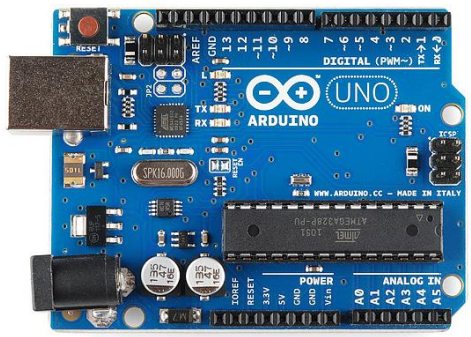
The voltage regulator **(14)** is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says – it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

The Arduino Family

Arduino makes several different boards, each with different capabilities. In addition, part of being open source hardware means that others can modify and produce derivatives of Arduino boards that provide even more form factors and functionality. If you're not sure which one is right for your project, check this guide for some helpful hints.

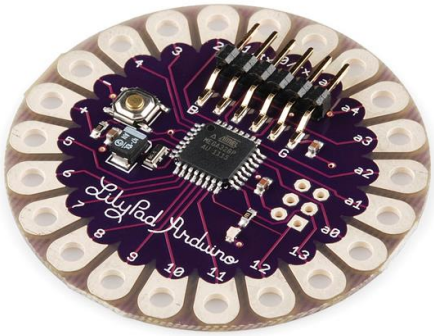
1. **Arduino Uno (R3)**

The Uno is a great choice for your first Arduino. It's got everything you need to get started, and nothing you don't. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a USB connection, a power jack, a reset button and more. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.



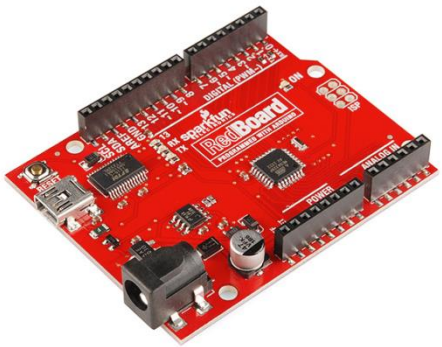
2. LilyPad Arduino

LilyPad is a wearable e-textile technology developed by Leah Buechley and cooperatively designed by Leah and SparkFun. Each LilyPad was creatively designed with large connecting pads and a flat back to allow them to be sewn into clothing with conductive thread. The LilyPad also has its own family of input, output, power, and sensor boards that are also built specifically for e-textiles. They're even washable!



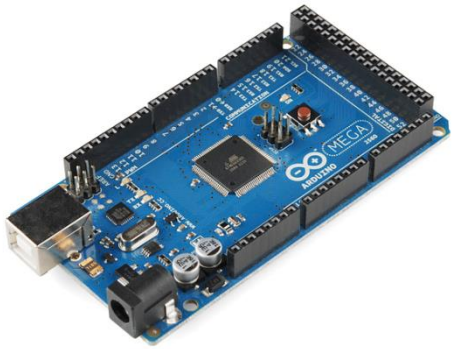
3. RedBoard

The RedBoard can be programmed over a USB Mini-B cable using the Arduino IDE. It'll work on Windows 8 without having to change your security settings (we used signed drivers, unlike the UNO). It's more stable due to the USB/FTDI chip we used, plus it's completely flat on the back, making it easier to embed in your projects. Just plug in the board, select "Arduino UNO" from the board menu and you're ready to upload code. You can power the RedBoard over USB or through the barrel jack. The on-board power regulator can handle anything from 7 to 15VDC.



4. Arduino Mega (R3)

The Arduino Mega is like the UNO's big brother. It has lots (*54!*) of digital input/output pins (14 can be used as PWM outputs), 16 analog inputs, a USB connection, a power jack, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The large number of pins makes this board very handy for projects that require a bunch of digital inputs or outputs (like lots of LEDs or buttons).



5. Arduino Leonardo

The Leonardo is Arduino's first development board to use one microcontroller with built-in USB. The code libraries are available which allow the board to emulate a computer keyboard, mouse, and more.

Features of Arduino

- **Cross-platform**

The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

- **Simple, clear programming environment**

The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well.

Open source and extensible software

The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based.

System Analysis

Existing System

In existing system each patient should be monitor by a person, thus the changes in pulse can be analyzed by the person. If the pulse rate decrease or increase during the absence of person, it will become a risky task for the patient. The doctor can't view the status of all patients at the same time. If the patient was not taken by the respective step during the changes in pulse will lead to bad situation. If the doctor checks the patient immediately the pulse rate came down will save their life. In this covid19 pandemic time, monitoring the temperature of the patient will be a risky task due to spreading of virus.

Disadvantage of Existing System

- The doctor can't know the status of the patient at all time.
- The absence of monitoring will lead to a risky task for the patient.
- If the pulse rate increases or decreases, the doctor can't get the intimation.
- The doctor won't get any intimation regarding the increase in temperature of the patient in hospital.

Proposed System

In proposed system no need to monitor the pulse rate of the patient by a person. Here if the pulse rate increase or decrease from the normal level, the doctor can get intimation through SMS. Each bed of the patient will be attached with the pulse sensor and GSM module. The pulse sensor helps in fining the current pulse rate of the patient and if the pulse rate of the rate decrease below 65 or increase above 80, the GSM gets trigger and the call will be received by the respective doctor. There will be a possibility of occurring the changes in pulse rate at any time and thus at any time the doctor can get intimation and thus immediate step can be taken for the respective patient. The Mlx90614 Contactless IR Infrared Temperature Sensor is used to find the patient temperature in smart manner. As soon as the temperature increases, the GSM triggers to send a SMS with current temperature value.

Advantage of Proposed System

- The doctor can get an intimation from the respective patient and thus from anywhere the status of the patient can be known by the doctor.
- No need to have a person to monitor the pulse rate of the patient and thus the absence of person won't affect the patient.
- The rate of pulse can be easily viewed through monitor display from the pulse sensor attached to the patient finger.
- The temperature of the patient and COVID19 patient will be monitored in an automatic manner using contactless IR Temperature sensor. The patient no need to wear the sensor, without any contact it starts to predict temperature.

Project Description

The main objective of the project is to help the doctor to monitor the pulse rate of the doctor in an interactive manner. The pulse rate of the patient may decrease or increase above the normal at any time and thus monitoring at all time will be a risky task. To overcome this problem the pulse sensor can be attached to the finger of the patient, which will monitor the pulse rate of the patient. If the pulse rate comes below the normal value, the GSM attached to the circuit will get trigger and the SMS will send to the respective doctor. The SMS will contain the patient bed number and current pulse rate, which help the doctor to take immediate step to rectify the corresponding patient.

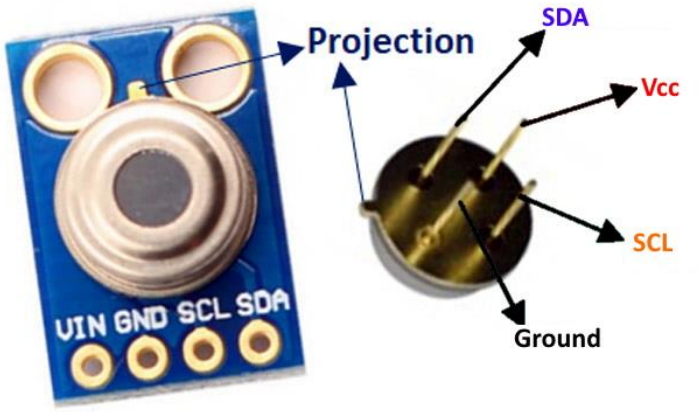
Module Description

Predict Patient Pulse

The pulse rate of the patient can be predicted by placing the pulse sensor on the finger of the patient. The pulse rate will be viewed through the monitor designed by processing. The rate of pulse and the range of flow will be viewed through the monitor.

Contactless Infrared (IR) Digital Temperature Sensor

The MLX90614 is a **Contactless Infrared (IR) Digital Temperature Sensor** that can be used to measure the temperature of a particular object ranging from -70° C to 382.2°C. The sensor uses IR rays to measure the temperature of the object without any physical contact and communicates to the microcontroller using the I2C protocol.



MLX90614 Pinout Configuration

Pin No.	Pin Name	Description
1	Vdd (Power supply)	Vdd can be used to power the sensor, typically using 5V
2	Ground	The metal can also act as ground
3	SDA – Serial Data	Serial data pin used for I2C Communication
4	SCL – Serial Clock	Serial Clock Pin used for I2C Communication

MLX90614 Temperature Sensor Specifications

- Operating Voltage: 3.6V to 5V (available in 3V and 5V version)
- Supply Current: 1.5mA
- Object Temperature Range: -70° C to 382.2°C
- Ambient Temperature Range: -40° C to 125°C
- Accuracy: 0.02°C
- Field of View: 80°

- Distance between object and sensor: 2cm-5cm (approx.)

Working Principle of MLX90614

As mentioned earlier, the MLX90614 sensor can measure the temperature of an object without any physical contact with it. This is made possible with a law called **Stefan-Boltzmann Law**, which states that all objects and living beings emit IR Energy and the intensity of this emitted IR energy will be directly proportional to the temperature of that object or living being. So the MLX90614 sensor calculates the temperature of an object by measuring the amount of IR energy emitted from it.

Use of MLX90614 Thermometer Sensor

The MLX90614 Temperature sensor is manufactured by a company called **Melexis**. The sensor is factory calibrated and hence it acts like a **plug and play sensor module** for speeding up development processes.

The MLX90614 consists of two devices embedded as a single sensor, one device acts as a sensing unit and the other device acts as a processing unit. The sensing unit an **Infrared Thermopile Detector** called **MLX81101** which senses the temperature and the processing unit is a **Single Conditioning ASSP** called MLX90302 which converts the signal from the sensor to digital value and communicates using I2C protocol. The MLX90302 has a low noise amplifier, 17-bit ADC and a powerful DSP which helps the sensor to have high accuracy and resolution.

The sensor requires no external components and can be directly interfaced with a microcontroller like Arduino. As you can see above the power pins (Vdd and Gnd) can be directly used to power the sensor, typically 5V can be used, but there are other versions of this sensor which can operate on 3.3V and 7V as well. The capacitor C1 is optional and is used to filter noise and provide optimum EMC. The signal pins (SCL and SDA) for used for I2C communication and can be connected directly to microcontroller operating on 5V logic.

Intimate Doctor by SMS

If the pulse rate of the patient decreases below 60 or increase above 80, the GSM attached to the board get triggered. Each patient will have a unique SIM number, therefore the SMS from the respective patient can be easily detect by the doctor. The SMS will contain the details of the bed number and current pulse rate of the patient.

Receive SMS

The doctor can receive the pulse rate of the patient with the bed details to his mobile and after receiving the immediate action can be taken by the doctor to save the life of the patient. The doctor no need to believe on some other person during the absence, as per the pulse rate the intimation can be easily fetch by the doctor.

Problem Definition

The main problem faced by the doctor is to monitor the patient during the absence. If the in-charge misses to monitor, it will became a huge problem for the patient. During the time of absence of in-charge, if the pulse rate of the patient increases or decreases it will create a serious for the patient. To overcome this problem, the heart pulse sensor can be fixed to the patient which will help to monitor the pulse rate of the patient and if it went below the normal range, the GSM attached to the circuit will get trigger and the SMS

will send to the respective doctor. The SMS will contain the details of the patient bed number and pulse rate to the doctor mobile. The doctor can take necessary step to solve the issue of the patient.

Circuit Description

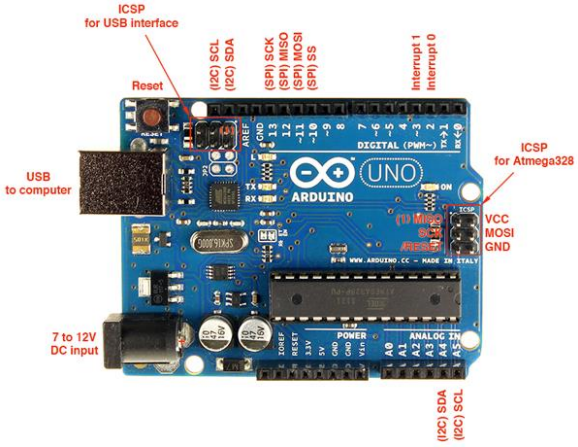
Arduino UNO

Arduino is an open source embedded development platform which includes a simple development board and is used easily for writing, programming and uploading codes. **Freeduino Uno** is a board which is designed to be a cost effective alternative for the official arduino board and is highly compatible with arduino compatible shields, tools and the arduino ide which can be used anywhere.

Freeduino Uno can be programmed directly through a USB connection to a PC through the Arduino IDE, which is a simple and easy to use program development environment for writing, compiling and uploading codes. Freeduino Uno offers onboard 5V and 3V3 voltage regulators and provides an empty hole next to each IO and power pin header for easy expansion.

Features

- The board is built on a high quality fr-4(1. 6 mm) double sided PCB with a green solder mask and a clear and legible legend
- Our products have a great platform to learn embedded programming
- These are compatible with all arduino compatible shields and the arduino ide
- It comes with an atmega328 with a pre-burnt arduino boot loader
- It has a standard 6x1 avr isp header
- All i/o pins connected by a female header, perfect for prototyping



GSM Module SIM 900

The SIM800 modem has a SIM800 GSM chip and RS232 interface while enables easy connection with the computer or laptop using the USB to Serial connector or to the microcontroller using the RS232 to TTL converter. Once user connect the SIM800 modem using the USB to RS232 connector, user need to find the correct COM port from the Device Manger of the USB to Serial Adapter. Then user can open Putty or any other terminal software and open an connection to that COM port at 9600 baud rate, which is the default baud rate of this modem. Once a serial connection is open through the computer or your microcontroller user can start sending the AT commands. When user send AT commands for example: "AT " user should receive back a reply from the SIM800 modem saying "OK" or other response depending on the command send.

Connecting with GSM

- The SIM800 requires huge power, so use a 12V DC 2 to 3 Amps adapter, any wrong adapter might cause damage also to the modem.
- On the board there are Tx(D) and Rx(D) pins, near the 5V and GND pins, don't ever connect anything to them or send any command to them, they are not the TTL pins for serial communication, they are for upgrading the firmware, and if user connect them to the microcontroller or computer, the firmware will get damaged and the module will stop working.

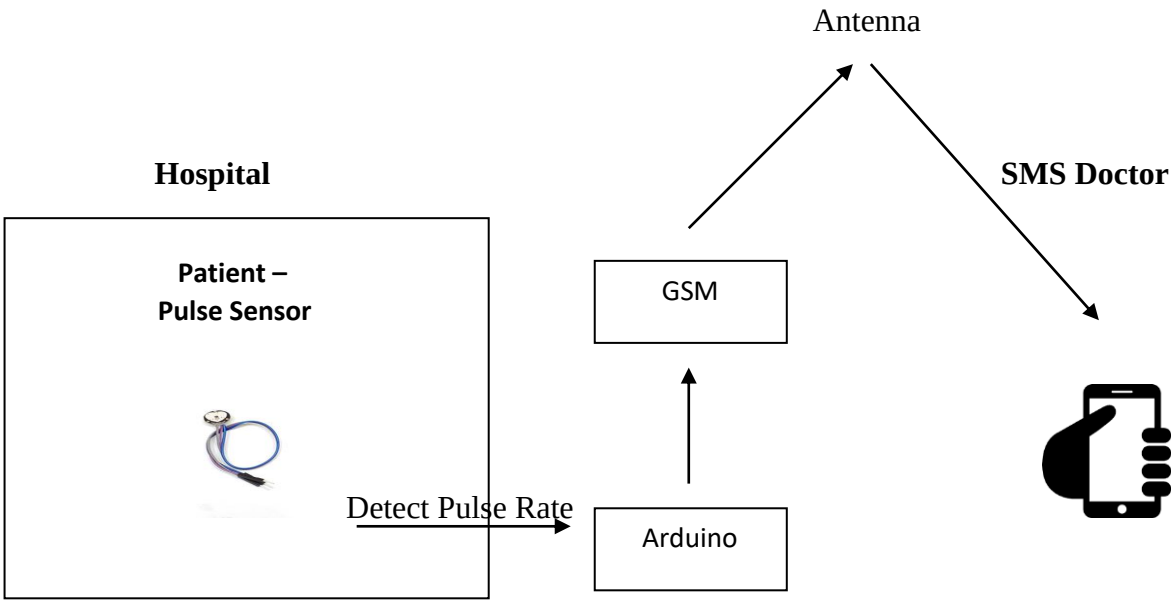
Pulse Rate Sensor

Pulse Sensor Amped is a greatly improved version of the original Pulse Sensor, a plug-and-play heart-rate sensor for Arduino and Arduino compatibles. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects.

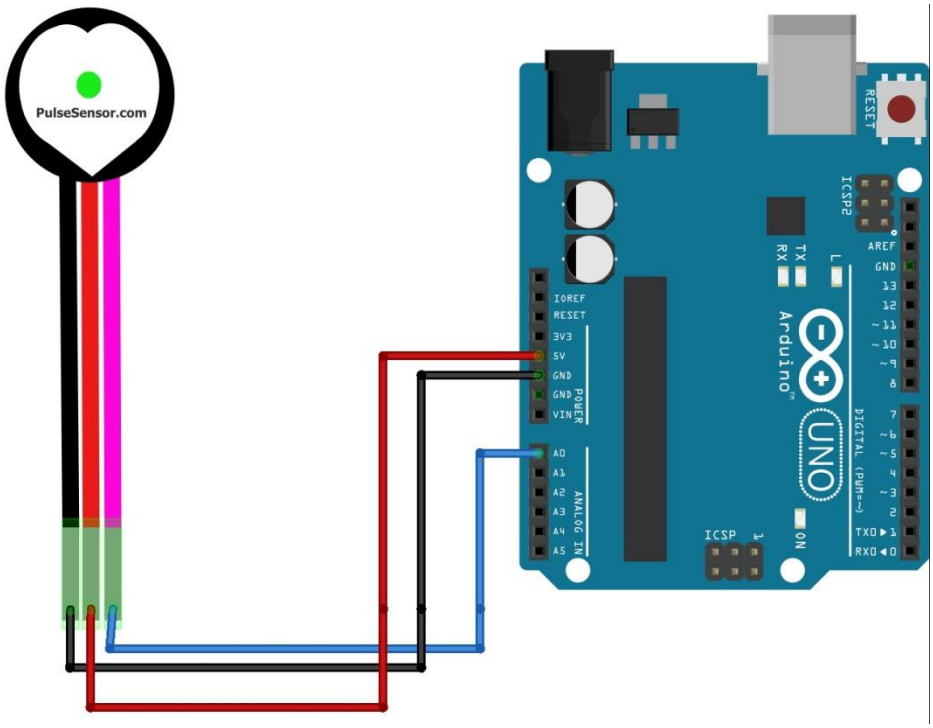
Pulse Sensor Amped adds amplification and noise cancellation circuitry to the hardware. It's noticeably faster and easier to get reliable pulse readings. Pulse Sensor Amped works with either a 3V and 5V Arduino.

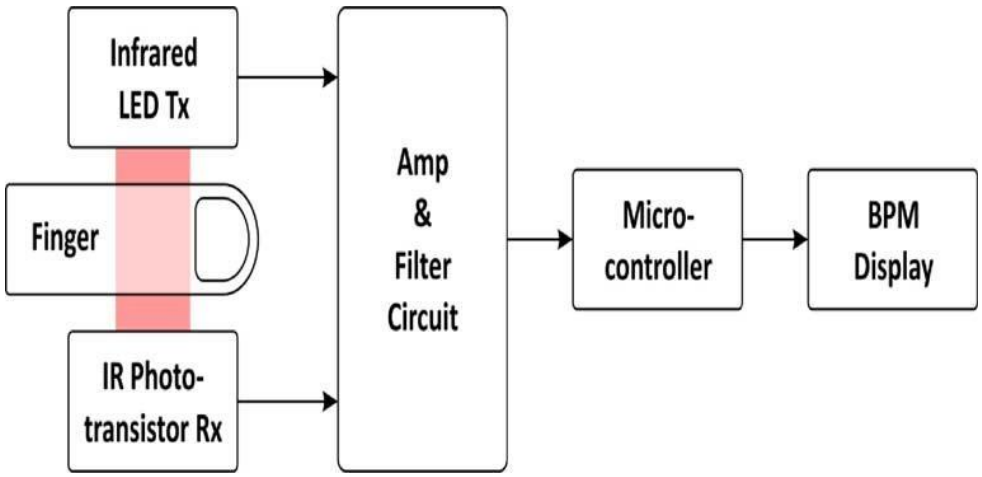
Power: 3-5V
Diameter: 16mm
Magnification: 330
LED Wavelength: 609nm

Block Diagram

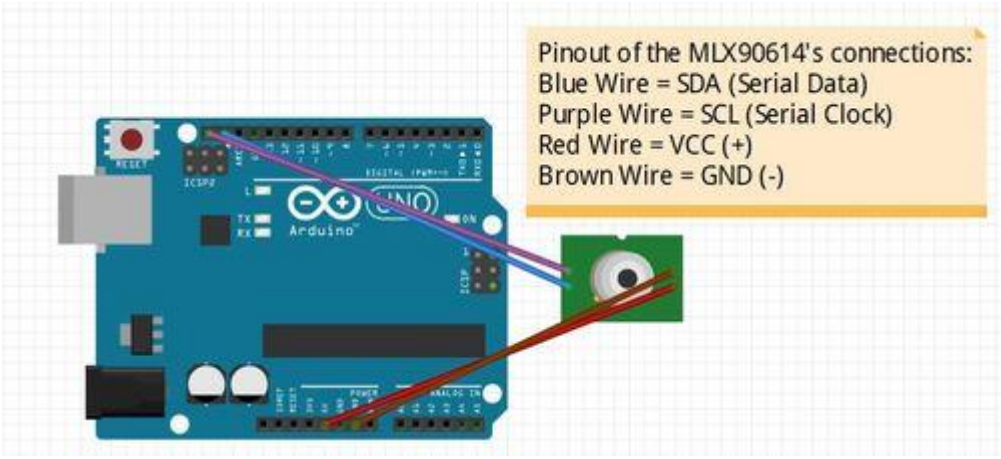


Circuit Connection

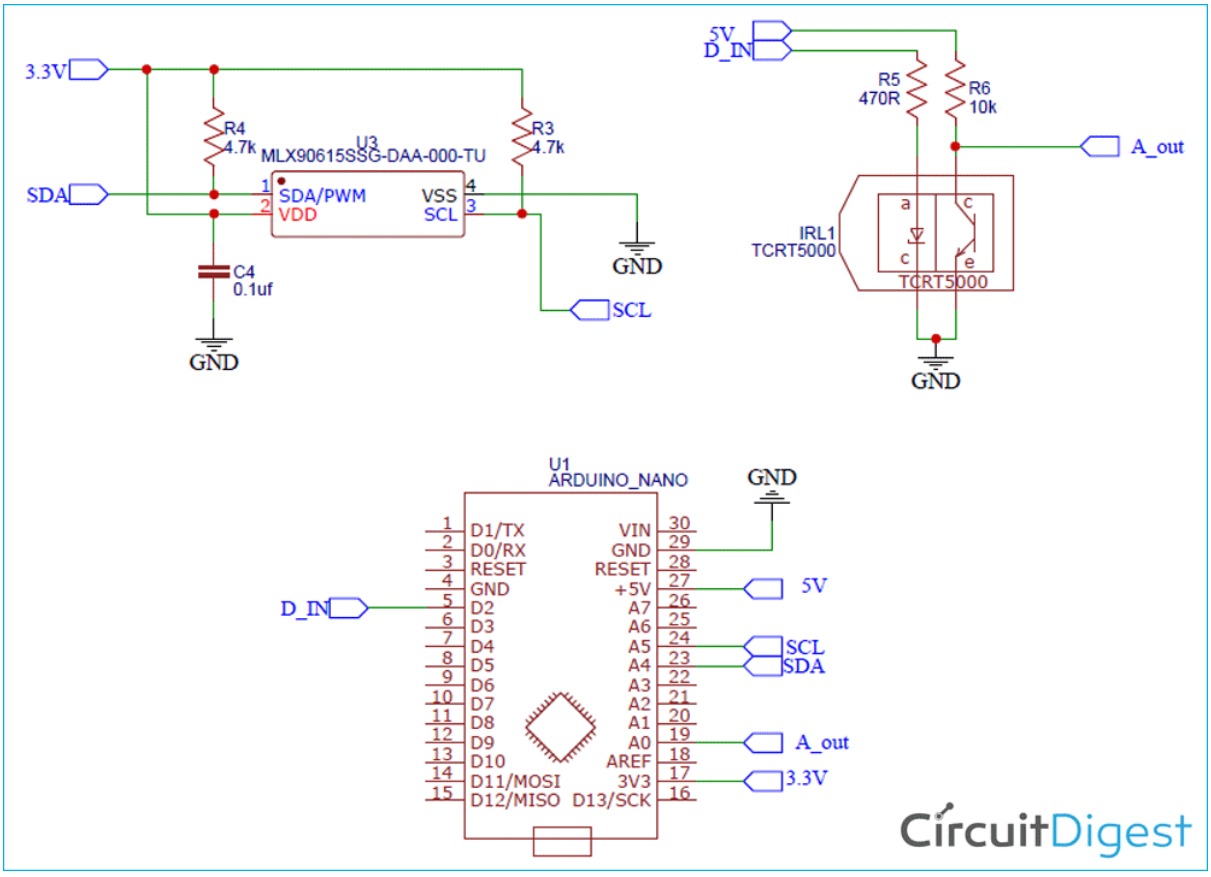


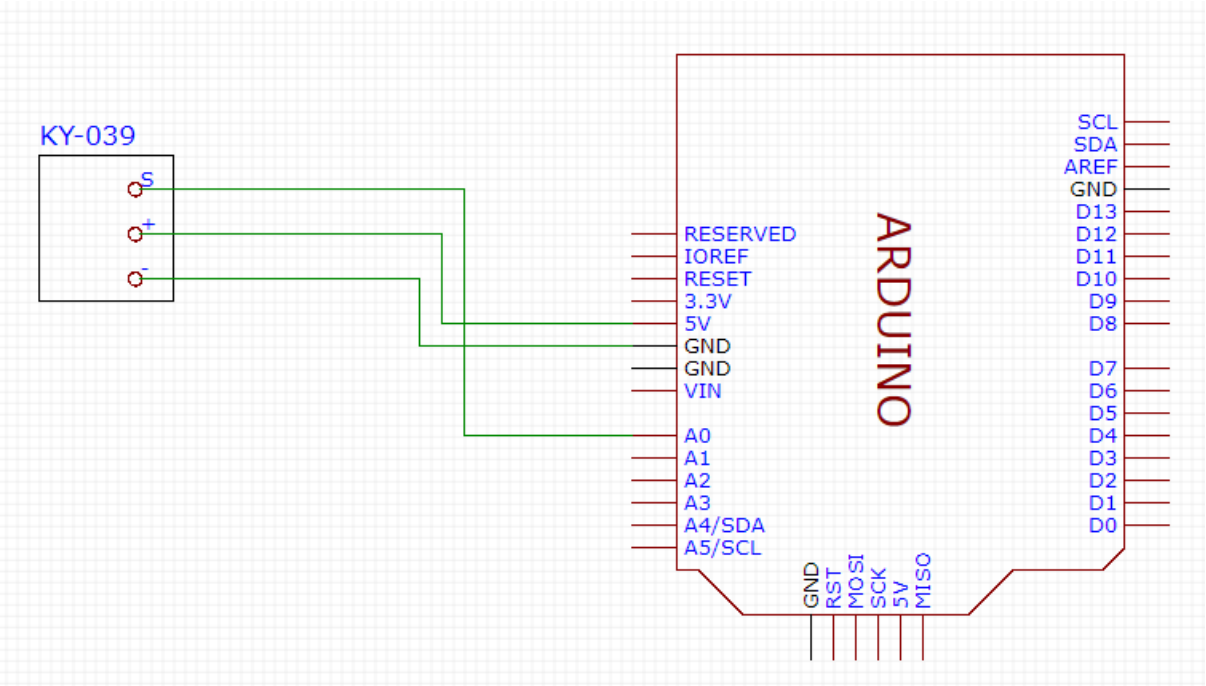


Mlx90614 Contactless IR Infrared Temperature Sensor

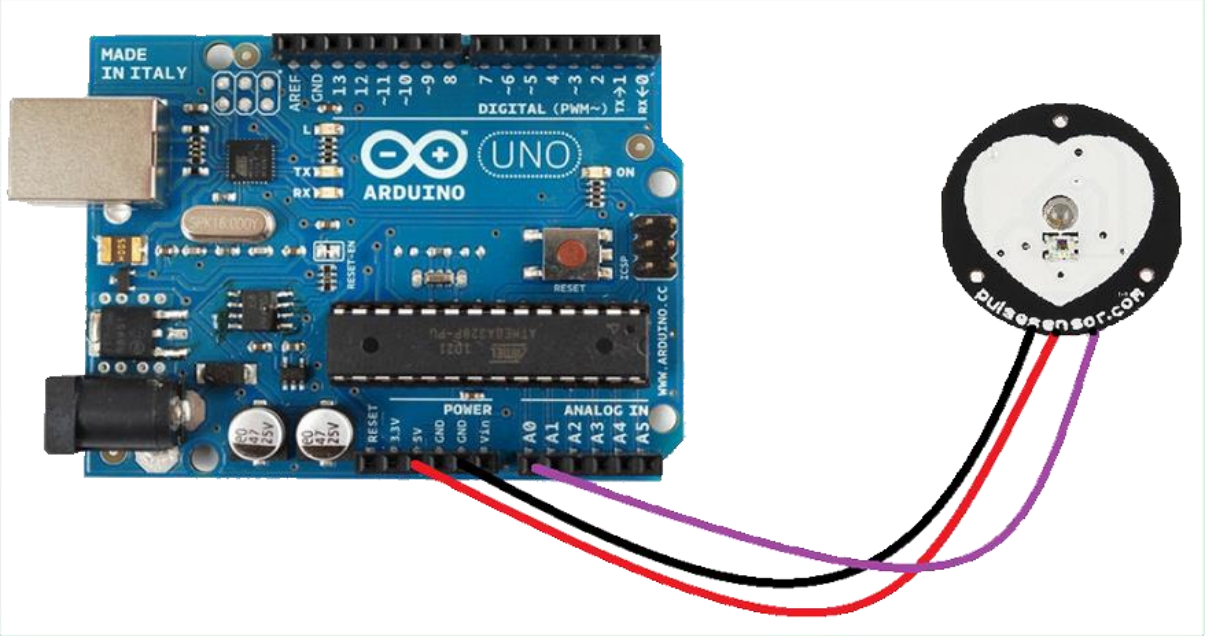


Input Design





Output Design



Pseudo code

```
#include <Wire.h>

#include <Adafruit_MLX90614.h>

Adafruit_MLX90614 mlx = Adafruit_MLX90614();

int PulseSensorPurplePin = 0;        // Pulse Sensor PURPLE WIRE connected to ANALOG PIN 0

int LED13 = 13;   //  The on-board Arduion LED

#include <SoftwareSerial.h>

SoftwareSerial mySerial(6,7);


int Signal;                // holds the incoming raw data. Signal value can range from 0-1024

int Threshold = 550;            // Determine which Signal to "count as a beat", and which to ingore.


void setup() {

Serial.begin(9600);

Serial.println("Adafruit MLX90614 test");

mlx.begin();

}
```

Find Temperature

```
void loop()
 {
Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempC());
Serial.print("*C\tObject = "); Serial.print(mlx.readObjectTempC()); Serial.println("*C");
Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempF());
Serial.print("*F\tObject = "); Serial.print(mlx.readObjectTempF()); Serial.println("*F");
Serial.println();
delay(500);

Signal = analogRead(PulseSensorPurplePin);  // Read the PulseSensor's value.
// Assign this value to the "Signal" variable.

Serial.println(Signal);                    // Send the Signal value to Serial Plotter.


if(Signal > Threshold){                         // If the signal is above "550", then "turn-on" Arduino's on-Board
LED.
digitalWrite(LED13,HIGH);
} else {
digitalWrite(LED13,LOW);                //  Else, the sigal must be below "550", so "turn-off" this LED.
}
```

Call GSM to make Call

```
if (mlx.readObjectTempC() > 31)
{
mySerial.begin(9600);
delay(1000);
mySerial.println("ATD8681819168;");
Serial.println("Calling through GSM Modem");
delay(1000);
```

GSM to send SMS

```
mySerial.println("AT+CMGF=1");
delay(1000);
mySerial.println("AT+CMGS=\"+918681819168\"\r");
delay(1000);
mySerial.println("Temperature Increses Alert - Take Necessary Steps");
delay(100);
mySerial.println((char)26);
delay(1000);
}
delay(10);
```

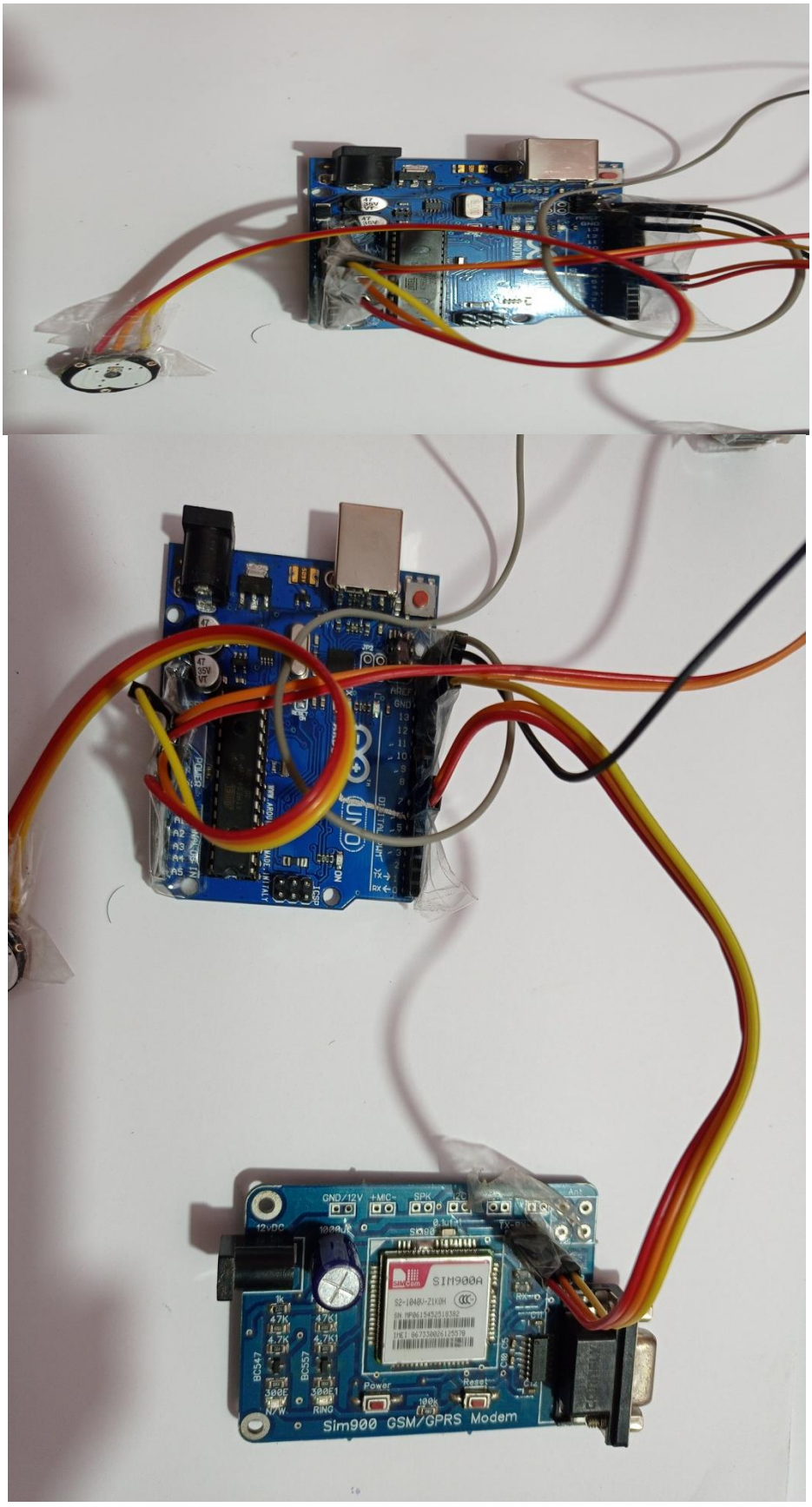
```
}
```

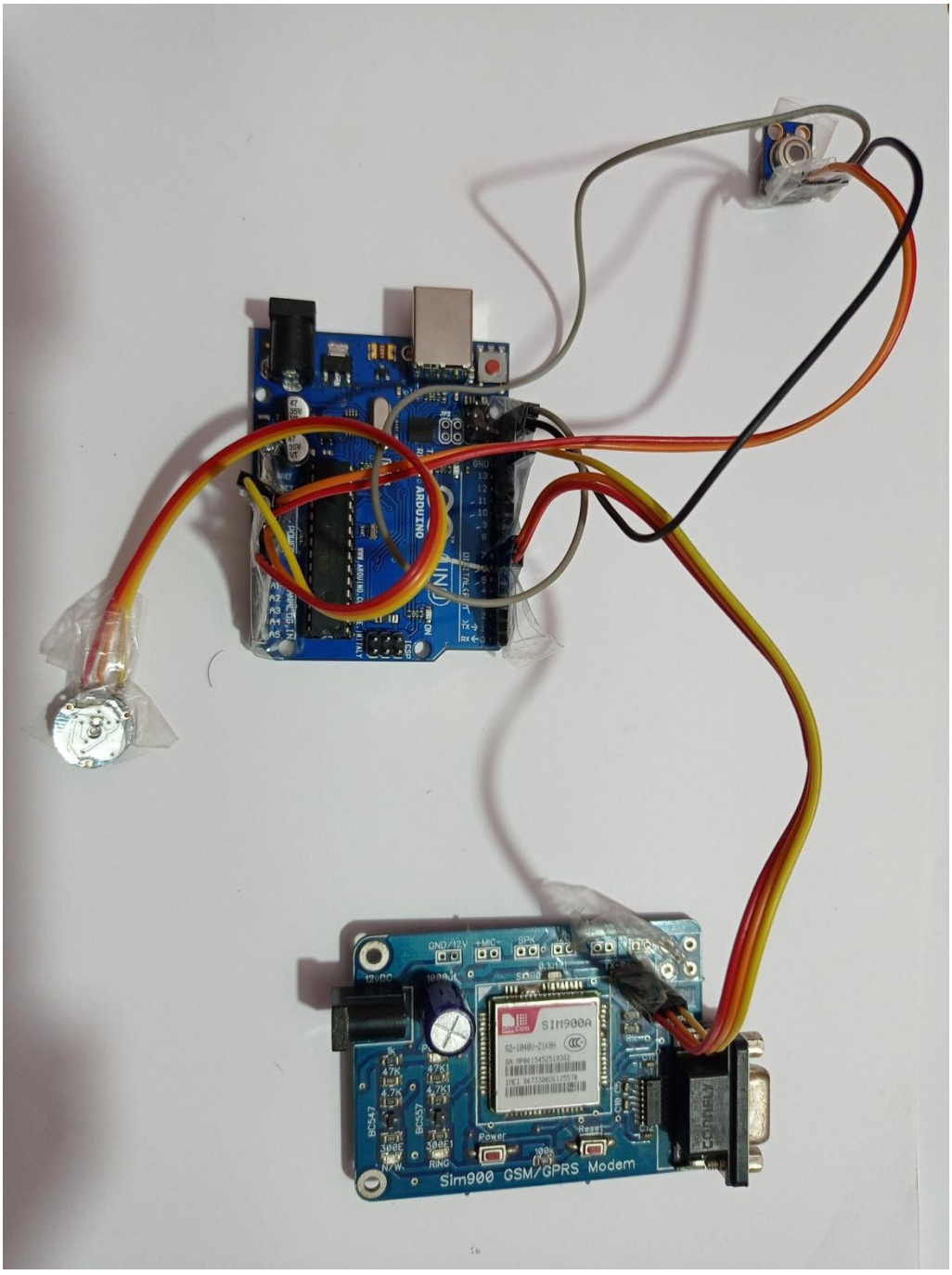
Adafruit Library Code for Temperature Prediction

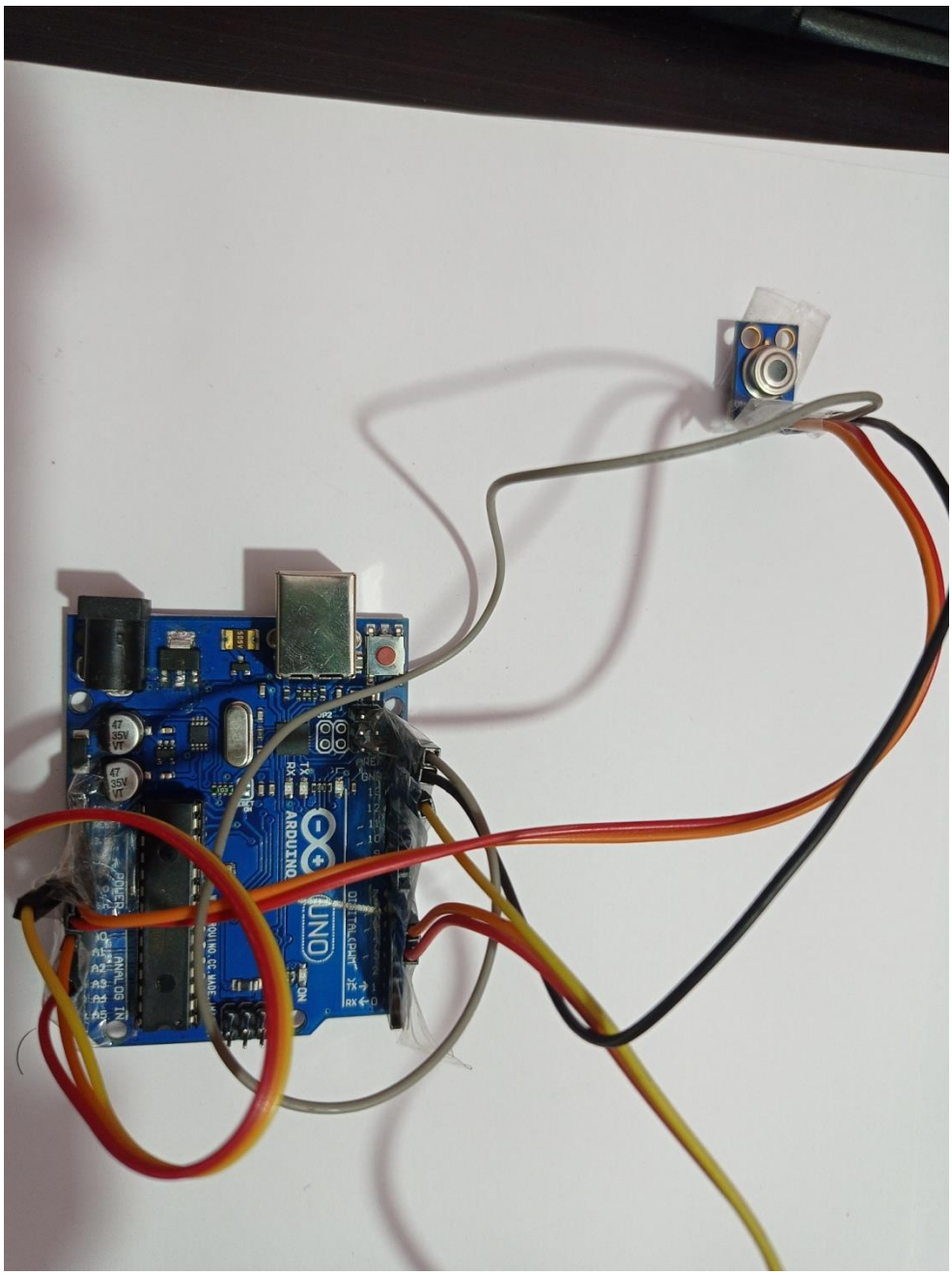
```
#include "Adafruit_MLX90614.h"

/**
 * @brief Construct a new Adafruit_MLX90614::Adafruit_MLX90614 object
 *
 * @param i2caddr The I2C address to use. Defaults to 0x5A
 */
Adafruit_MLX90614::Adafruit_MLX90614(uint8_t i2caddr) { _addr = i2caddr; }

/**
 * @brief Begin the I2C connection
 *
 * @return bool  Always returns true
 */
bool Adafruit_MLX90614::begin(void) {
  Wire.begin();

  /*
  for (uint8_t i=0; i<0x20; i++) {
    Serial.print(i); Serial.print(" = ");
    Serial.println(read16(i), HEX);
```

```
  }
  */
  return true;
}

/**
 * @brief Read the raw value from the emissivity register
 *
 * @return uint16_t The unscaled emissivity value
 */
uint16_t Adafruit_MLX90614::readEmissivityReg(void) {
  return read16(MLX90614_EMISS);
}
/**
 * @brief Write the raw unscaled emissivity value to the emissivity register
 *
 * @param ereg The unscaled emissivity value
 */
void Adafruit_MLX90614::writeEmissivityReg(uint16_t ereg) {
  write16(MLX90614_EMISS, 0); // erase
  delay(10);
  write16(MLX90614_EMISS, ereg);
  delay(10);
```

```
}

/**
 * @brief Read the emissivity value from the sensor's register and scale
 *
 * @return double The emissivity value, ranging from 0.1 - 1.0
 */
double Adafruit_MLX90614::readEmissivity(void) {
  uint16_t ereg = read16(MLX90614_EMISS);
  return ((double)ereg) / 65535.0;
}
/**
 * @brief Set the emissivity value
 *
 * @param emissivity The emissivity value to use, between 0.1 and 1.0
 */
void Adafruit_MLX90614::writeEmissivity(double emissivity) {
  uint16_t ereg = (uint16_t)(0xffff * emissivity);

  writeEmissivityReg(ereg);
}
```


SIM900A
Sim900 GSM/GPRS Modem


ARDUINO
SIM900A
Power
Reset
Sim900 GSM/GPRS Modem


ARDUINO
UNO

Conclusion

The patient can be monitor at all time by the doctor. The doctor can get an intimation as a SMS, when the pulse of the respective pateint reaches too low or high. The doctor can take immediate action during the changes in pulse rate for the respective person from anywhere, which will save the life of the patient. The concept of monitoring the patient can be implemented in a IoT concept and therefore during the absence of nurse or any other person won't affect the patient. The doctor can easily monitor the patient health from anywhere which will save the life of the patient in a immediate time duration.

BIBLIOGRAPHY

REFERRED BOOKS

[1] Embedded C written by Michael Pont published by Pearson on March 28, 2014.

[2] AVR Microcontroller and Embedded Systems: Using Assembly and C written by Muhammad Ali Mazidi

[3] Programming Fundamentals of Embedded Software: Where C and Assembly Meet written by Lewis.

[4] Database connectivity Embedded Software Development With C written by Haring D.D. Et.Al, Kai, Qian on 2013

REFERRED WEBSITES

[1] www.arduino.cc

[2] www.arduinoclassroom.com

[3] www.instructables.con

[4] www.practicalarduino.com