



**INDUSTRIAL INSTRUMENTATION LABORATORY**

**(UEE15B04)**

*A LAB WORK REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF*

**BACHELOR OF TECHNOLOGY**

IN

**ELECTRONICS AND INSTRUMENTATION ENGINEERING**

On the topic

**Heart pulse sensor using Arduino**

UNDER THE SUPERVISION OF

**DR. Rupam Gupta Roy**

PROFESSOR, EIE DEPARTMENT, NIT AGARTALA

**ACKNOWLEDGEMENT**

I would like to express my special thanks of gratitude to **Dr. Rupam Gupta Roy** (Professor) who gave us the golden opportunity to do this wonderful report on the subject “**INDUSTRIAL INSTRUMENTATION”** and for their continuous efforts to help us grasp this subject and also for generating interest in it. We would like also to thank our lab technicians who has helped us to master the experimental knowledge and also to confirm theoretical knowledge from the same.

**SEMESTER- 5TH**

PRITHIKA SAHA(22UEI147)

**INDEX**

1.INTRODUCTION………………….PG 1

2.APPARATUS REQUIRED………..PG 2

3.HEART PULSE SENSOR………….PG 3

4.ARDUINO UNO…………………..PG 4

5.JUMPER WIRES…………………..PG 5

6.OLED DISPLAY…………………….PG 6

7.SCHEMATIC DIAGRAM…………PG 7

8.CIRCUIT DIAGRAM………………PG 8

9.DISPLAY RESULT………….……….PG 9

10.EXPLANATION……………….…PG 10

11.CONCLUSION…………………..PG 11

**INTRODUCTION**

A heart pulse sensor paired with an Arduino enables real-time monitoring of heart rate, creating a valuable tool for projects in health tracking, biofeedback, and wearable tech. This system functions by detecting blood flow through the fingertip or earlobe, where each pulse alters light absorption. By connecting a pulse sensor to an Arduino, the analog signal from the sensor can be captured, processed, and displayed as beats per minute (BPM), providing accurate heart rate data in a simple, accessible format.

Arduino’s versatility makes it easy to program and analyze the pulse data. Using an open-source pulse sensor library, we can measure raw signals and detect each heartbeat, allowing for visual outputs like LEDs, serial monitors, or even smartphone integration. This project introduces users to essential electronics and programming concepts, such as reading analog values, filtering noise, and setting detection thresholds.

Beyond technical skills, the project highlights the potential for low-cost, custom health devices, suitable for educational projects, DIY wearables, and health research. With minimal components and straightforward code, building a heart rate monitor becomes a hands-on learning experience, bridging the gap between technology and health monitoring in a fun, practical way.

**apparatus required**

1.HEART PULSE SENSOR

2.ARDUINO UNO

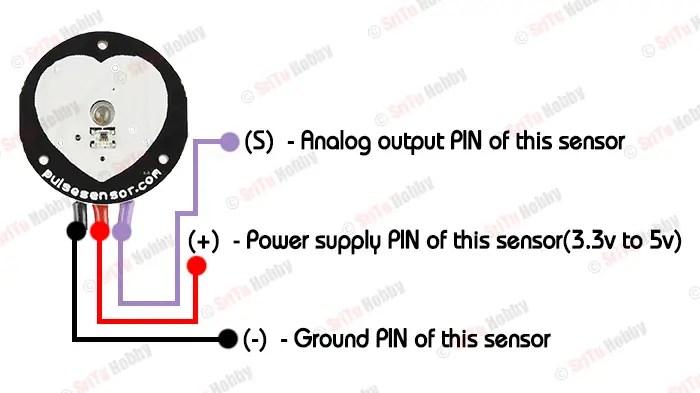
3.JUMPER WIRES

4.OLED DISPLAY

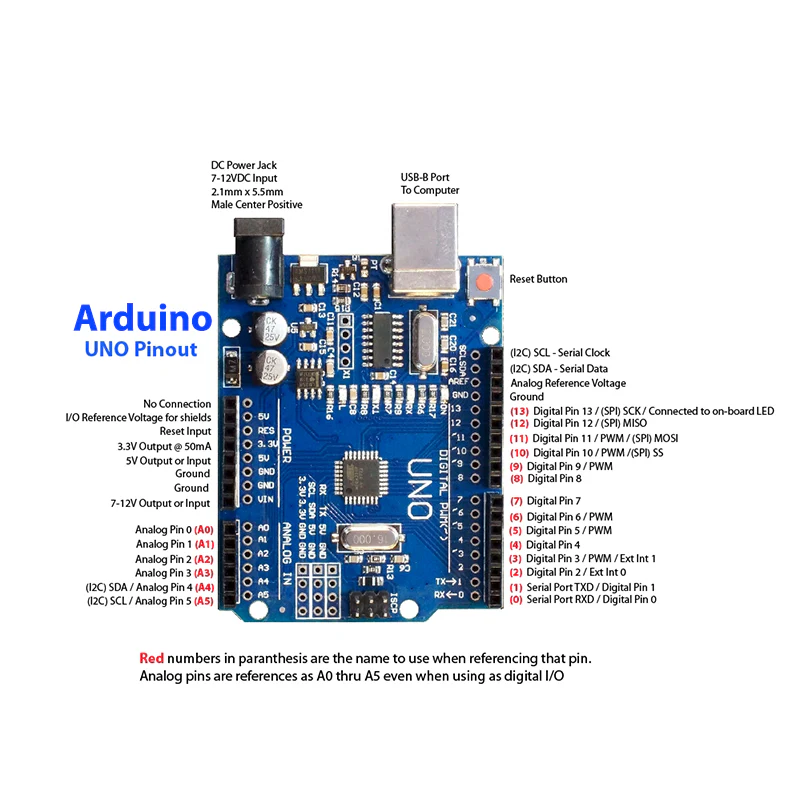
**heart pulse sensor**

This sensor is created mainly using two components. That is, the ADPS-9008 light photosensor and the one green LED. Next, on the backside of this sensor, we can see additional components with the LED. Among these, we can see resistors, capacitors, op-amp, and one reverse protection diode.

The heart pulse sensor operates on a principle called **photoplethysmography (PPG)**, which detects blood volume changes in the microvascular tissue. This sensor includes an LED that emits light (usually green or infrared) and a photodetector positioned to measure the light reflected or transmitted through the tissue. When the heart pumps, blood volume increases, absorbing more light and decreasing the light reaching the photodetector. Between beats, blood volume decreases, allowing more light to reach the detector. These fluctuations are converted into an analog signal that corresponds to the heartbeat, which can be processed to determine the pulse rate in beats per minute (BPM).



**arduino uno**

The Arduino Uno operates as a microcontroller platform based on the ATmega328P chip, which handles processing, inputs, and outputs to interact with various sensors, displays, and devices. The board includes 14 digital input/output pins (6 can be used for PWM), 6 analog inputs, and power pins, enabling it to interface with a wide range of electronic components. The Uno operates by executing code uploaded from the Arduino IDE via USB. When powered, it continuously runs the code in a loop, reading data from inputs (like sensors), processing it, and controlling outputs (like LEDs or motors). Its simplicity and versatility make the Uno ideal for prototyping and educational projects.

**jumper wires**

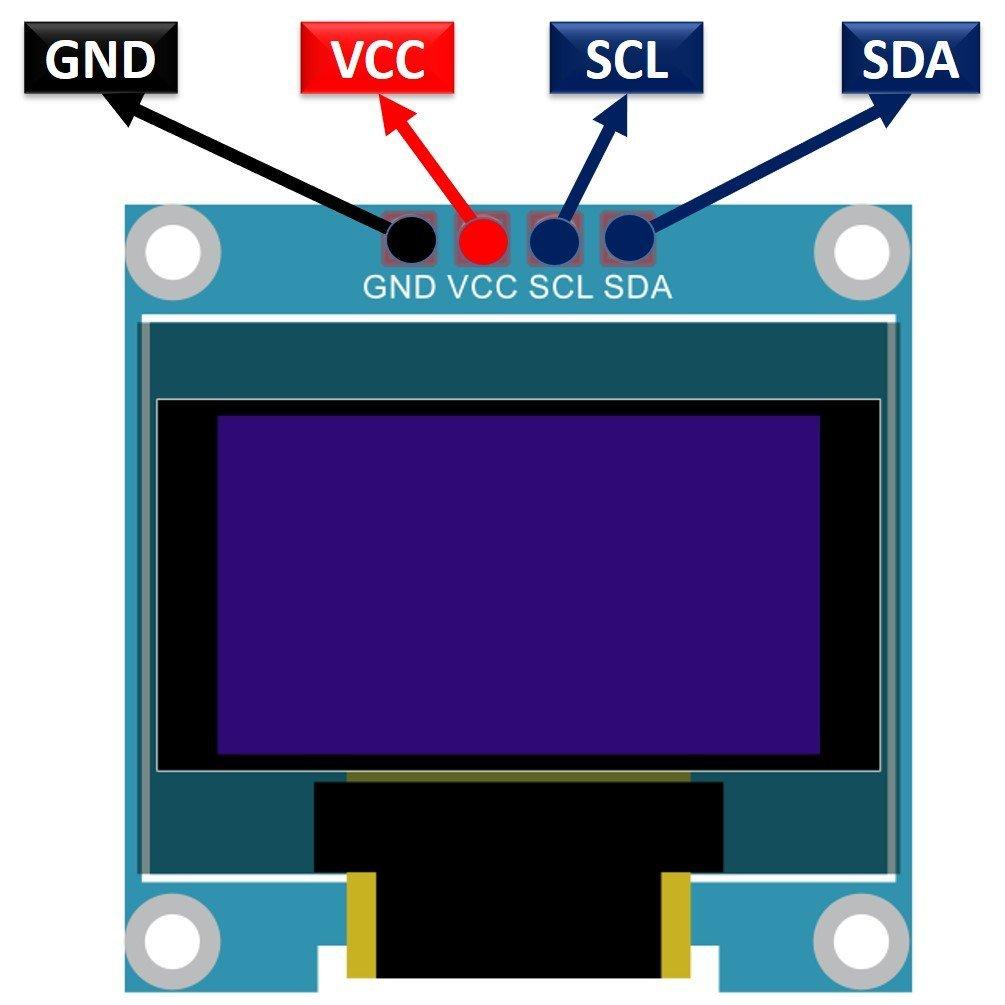
Jumper wires are essential connectors in prototyping and electronic projects, enabling quick and flexible connections between components on a breadboard or directly to an Arduino or other microcontrollers. They come in three types: **male-to-male**, **female-to-female**, and **male-to-female**, each suited for specific types of connections depending on the pins involved.

Inside each jumper wire is a small conductive wire, usually copper, which transmits electrical signals. The outer insulation prevents short circuits and provides durability. The beauty of jumper wires lies in their simplicity and reusability; they allow you to easily experiment with connections, test circuits, and make adjustments without soldering.

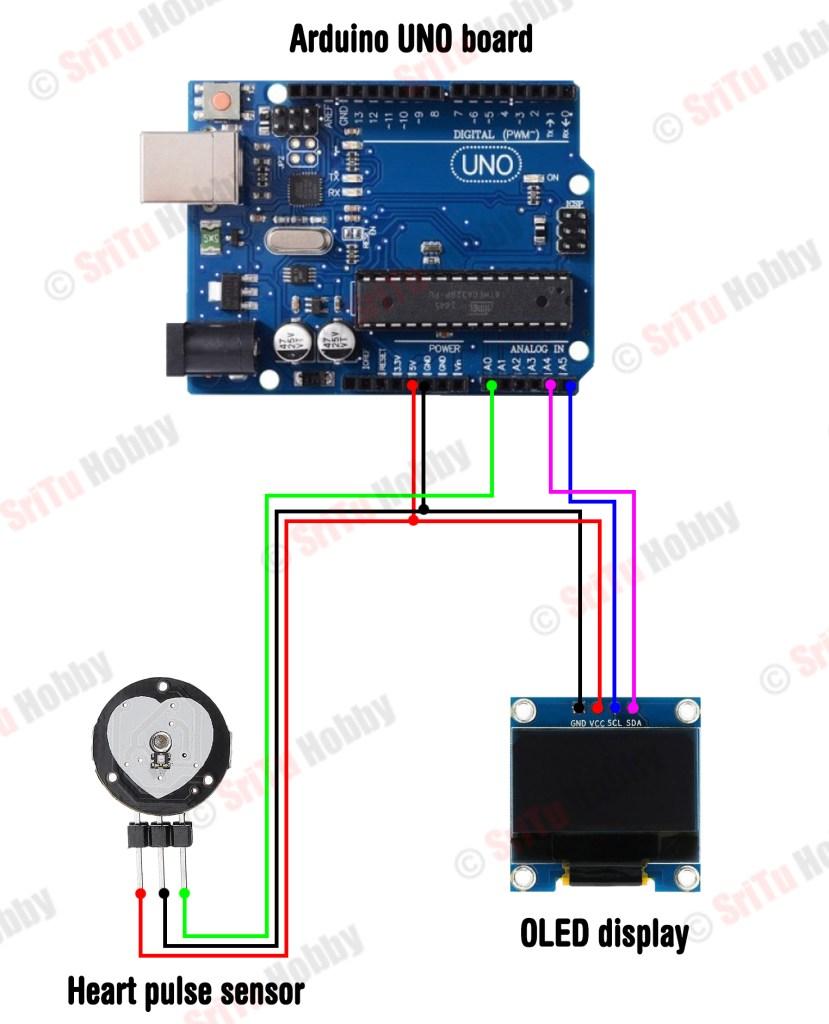


**oled display**

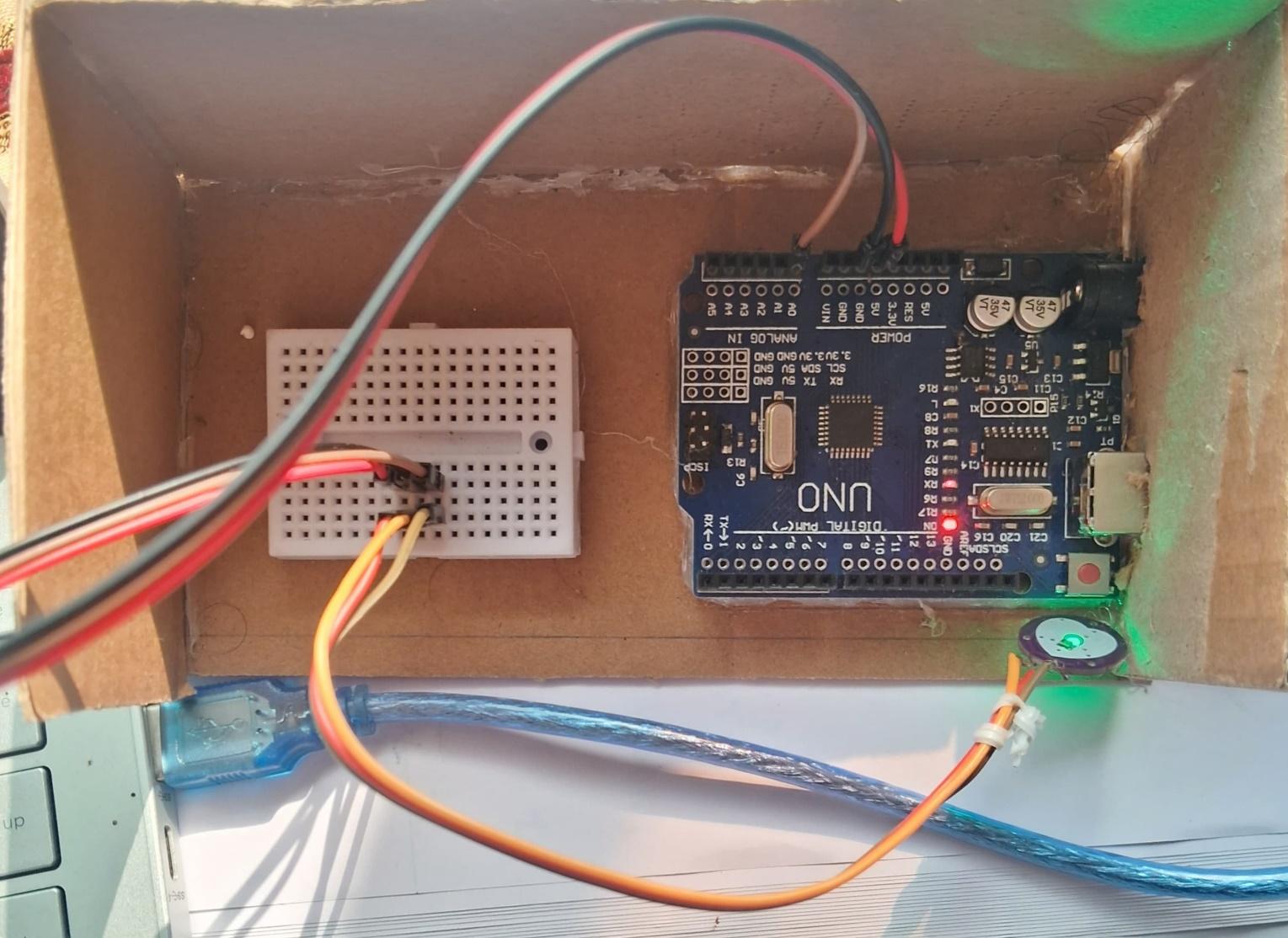
An OLED (Organic Light Emitting Diode) display is a thin, lightweight, and energy-efficient screen technology that creates bright, high-contrast visuals. Unlike traditional LCDs, which require a backlight, OLED displays are made of organic compounds that emit light individually when an electric current passes through them. This means each pixel on an OLED screen lights up independently, allowing for true black levels, as pixels can completely turn off in dark areas. This self-illumination mechanism not only enhances contrast and color vibrancy but also reduces power consumption when displaying dark images, making OLED displays popular in wearable devices, smartphones, and modern screens.



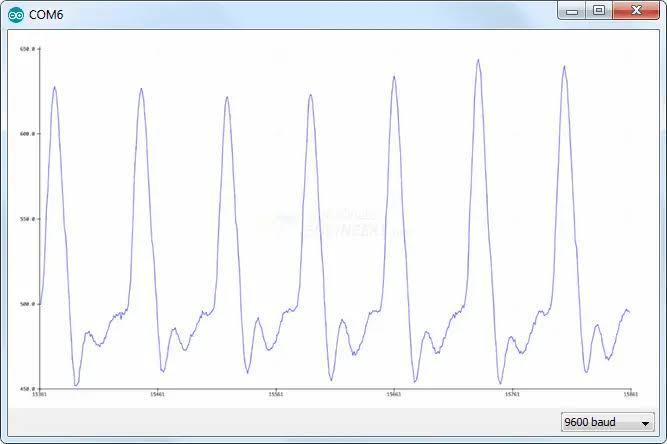
**SCHEMATIC DIAGRAM**



**CIRCUIT DIAGRAM**



**DISPLAY RESULT**



**Explanation:**

**Step 1: Connect the Pulse Sensor**

1. Place the Pulse Sensor on the breadboard.
2. Connect the VCC pin of the Pulse Sensor to the 5V pin on the Arduino.
3. Connect the GND pin of the Pulse Sensor to the GND pin on the Arduino.
4. Connect the Signal pin of the Pulse Sensor to the A0 (analog input) pin on the Arduino.

**Step 2: Arduino Code**

1. Install the PulseSensor Playground Library:
2. Upload the Code

**Step 3: Testing**

1. Upload the code to the Arduino.
2. Open the Serial Monitor from the Arduino IDE to view your heart rate readings.
3. Place the sensor on your fingertip or earlobe, where the pulse can be detected easily.
4. You should see the LED blinking in sync with your heartbeat, and the BPM will display in the Serial Monitor.

**conclusion**

In this project, we successfully created a heart pulse monitoring system using an Arduino and a pulse sensor module. This setup demonstrated how biomedical signals like heart rate can be measured with simple, affordable components. By capturing the heartbeat data and calculating beats per minute (BPM), we could monitor pulse rate in real-time.

Through this project, we learned about interfacing biomedical sensors with microcontrollers, data acquisition, and signal processing. Adjustments to sensor placement and calibration helped us understand the importance of stable sensor connections and threshold settings for accurate readings.

This project can be expanded upon by adding features like data logging, wireless data transmission, or integrating an LCD screen to display readings directly. Overall, it provides a solid foundation for understanding pulse monitoring technology and serves as a stepping stone toward more complex health monitoring systems.