Department of Chemical Engineering
University of California, Santa Barbara

CH E 166. Mechatronics and Instrumentation for Chemical Engineer

Design of an Arduino-Based Real-Time Sleep Apnea Alarm System Using Heartbeat, Passive Infrared, and Microphone Sound Detection Sensors

Sean Shen, Yulun Wu

June 6, 2024

1 Abstract

Sleep apnea, particularly obstructive sleep apnea (OSA), is a prevalent condition characterized by repetitive interruptions in breathing during sleep, which can lead to serious health complications if left untreated. To address this, we have developed an innovative alarm system designed to monitor individuals at risk for sleep apnea and alert caregivers or family members during critical episodes. The system integrates three key sensors: a heart rate sensor, a microphone sensor for detecting snoring sounds, and a passive infrared (PIR) sensor to monitor physical movement. These sensors work in concert to detect signs of apnea, such as abrupt changes in heart rate, cessation of snoring, and lack of movement, which may indicate airway obstruction. The alarm system is configured to activate based on predefined thresholds, providing real-time alerts that facilitate immediate intervention. Our design also highlights the need for further improvements, including the incorporation of data storage capabilities and enhanced sensor specificity to overcome current limitations and improve diagnostic accuracy. This system not only aims to enhance the safety and health monitoring of individuals with sleep apnea but also serves as a critical tool in preventing the severe health emergencies associated with the disorder.

2 Introduction

2.1 Significance of Setting up Alarm for Sleep Apnea

Sleep apnea encompasses a spectrum of breathing disruptions during sleep, attributed to various factors including relaxation of throat muscles, excessive throat tissue, or structural airway irregularities. Among its types, obstructive sleep apnea (OSA) stands out as the most prevalent, characterized by airway obstruction due to relaxed throat muscles. This obstruction leads to diminished airflow or complete cessation of breathing, often accompanied by symptoms like loud snoring and gasping[**1**]. Left untreated, OSA can precipitate severe complications such as heart attacks, strokes, or fatal suffocation, particularly posing risks for older individuals or those with pre-existing respiratory conditions. Thus, proactive measures, including heightened awareness and emergency response protocols such as alarm systems to notify families in case of sleep apnea-related emergencies, are imperative in averting potential tragedies and safeguarding vulnerable individuals' well-being.

2.2 Parameters to Set Up the Alarm System

The design of the alarm system needs to be based on several parameters to ensure accurate and precise setup, minimizing the risk of errors in the system.

- When designing the alarm system, prioritizing heart rate as the primary parameter is essential. Monitoring airflow during sleep isn't practical, making heart rate the most direct indicator of potential suffocation. Research suggests that during suffocation episodes, a person's heart rate sharply increases within the first 60 to 90 seconds[**2**], making it a reliable trigger for alarm activation.
- Sound detection is another critical parameter to consider when designing the alarm system. Loud snoring, a common symptom of sleep apnea, indicates that the person

is breathing. If breathing ceases for an extended period, the snoring stops. The alarm system should detect this absence of snoring and trigger an alert.

- Motion detection is also crucial. Lack of movement may indicate potential suffocation, making it essential for the motion sensor to monitor the person's movements to assess danger. Additionally, in the event of a false alarm, the individual can simply wave a hand at the sensor to immediately deactivate the alarm.

All of these parameters should be integrated to effectively monitor the person's sleep, ensuring that the alarm triggers accurately and promptly.

2.3 The Principle of Arduino Sensors

Microphone sensor captures sound from its environment and converts it into electric signals for the Arduino microcontroller. At the heart of the sensor is an electret microphone, which functions by generating a small voltage in response to sound pressure. This analog signal is typically weak and requires amplification, which is achieved through an onboard amplifier circuit, such as the LM393 or LM386. The amplified signal is then sent to one of the Arduino's analog inputs. Here, the Arduino's onboard analog-to-digital converter (ADC) translates the analog sound signal into a digital value that the microcontroller can process and analyze[**3**]. Under some coding, these digital values to monitor sound levels, detect noise patterns, or activate specific responses, such as turning on an LED or triggering an alarm, based on the characteristics of the detected sound. This setup allows for a variety of applications, from noise monitoring systems to interactive sound-sensitive projects.

The heart rate sensor used for this design is an optical heart rate sensor operates on the principle of photoplethysmography (PPG)[**4**]. This method involves emitting light, typically IR, into the skin and measuring the amount of light that pass through to the sensor's photodetector. The whole heart rate sensor have two parts, a sensor part which contains an IR LED and a photo diode, and a control part that consist of an OP-Amp IC that help to connect the signal to the microcontroler. When there is a heartbeat, the blood flow in the capillaries will vary the amount of blood and also the amount of light that the LED shoots to the photo diode, result in variable light signals. These variations are converted into an electrical signal, which is then processed to determine the heart rate. The sensor's ability to monitor these changes continuously allows for real-time heart rate monitoring.

The passive infrared (PIR) sensor also utilized IR as its principle of detecting. The PIR sensor contains two slots, each crafted from a material that is sensitive to infrared (IR) radiation. While the accompanying lens primarily aids in directing IR radiation to these slots, it is not actively involved in detection. Typically, both slots perceive the same level of IR radiation emitted from the surroundings, such as walls or the outdoors, when the sensor is in a dormant state. However, when a warm object, like a human or animal, passes by, it initially intercepts the IR radiation to one slot before the other, creating a positive differential change in IR levels between the two slots. As the object moves away, this effect reverses, resulting in a negative differential change. These fluctuations generate detectable signal pulses, which the sensor uses to identify movement within its range[**5**].

3 Method

The alarm system is divided into two main sections: the monitoring section and the action section. The monitoring section integrates all sensors and an LCD screen, connected in parallel to an Arduino microcontroller. Each sensor is connected to the 5V power supply, ground, and corresponding analog or digital pins. To prevent overloading, resistors ranging from 200 to 1,000 Ω are used with the LCD screen. Meanwhile, the action section, which includes a buzzer and an LED, is configured in series but operates parallel to the monitoring section. This setup ensures that the activation loop triggers the alarm appropriately when specified conditions are met. The complete circuit configuration and the theoretical set up of the alarm system is illustrated in Figure 1, Figure 2 and Figure 3.

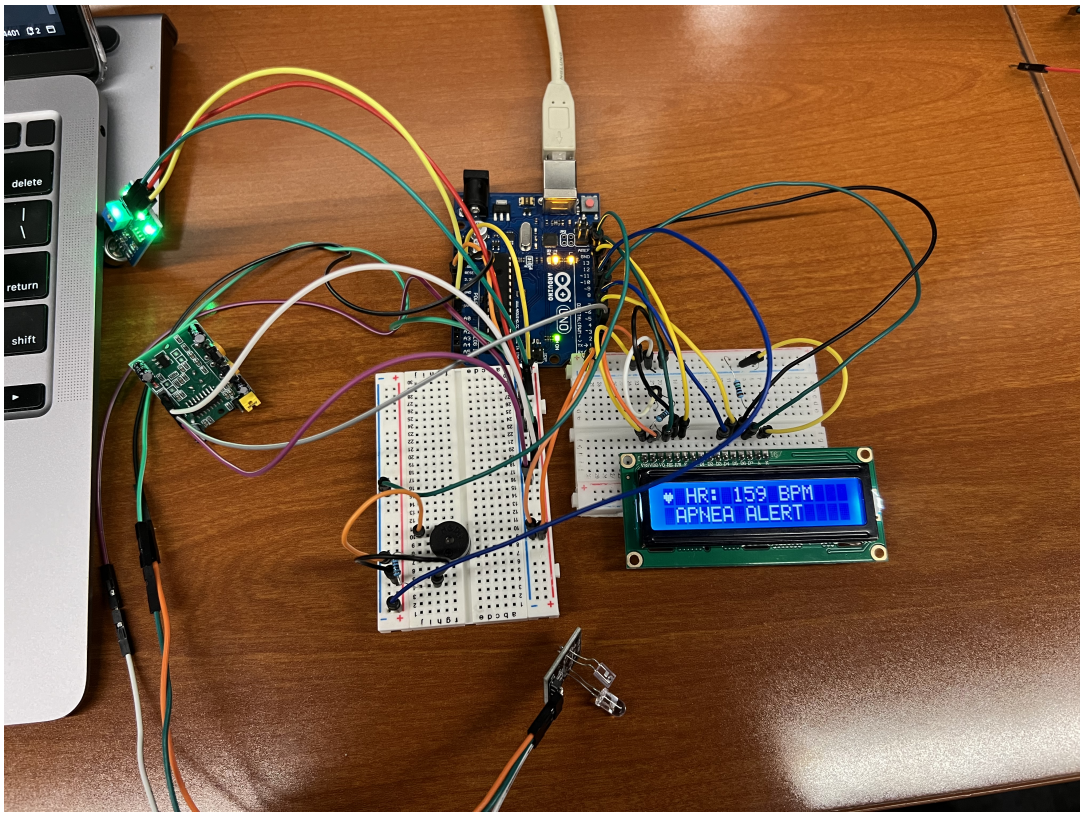

Figure 1: the picture of the actual set up of the circuit

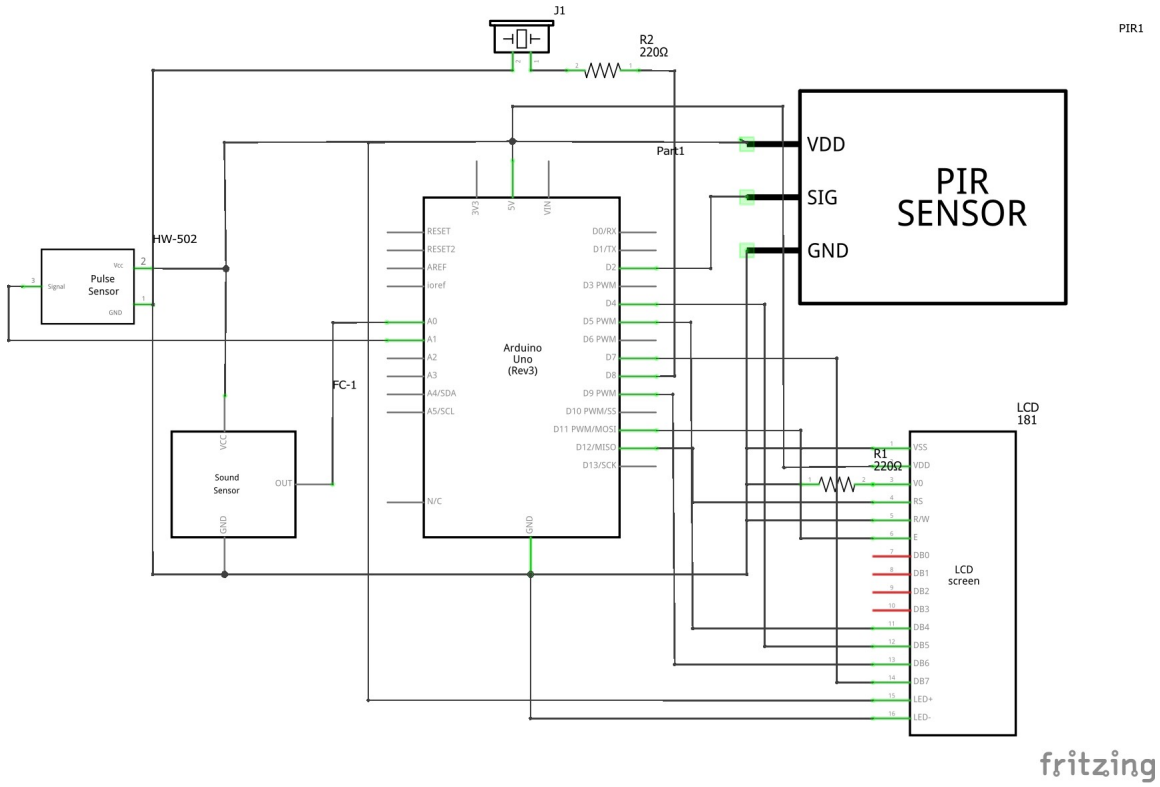

Figure 2: The circuit configuration for a sleep apnea detection alarm system is shown. It features an Arduino Uno as the central controller, interfaced with a PIR sensor for motion detection, a pulse sensor for monitoring heart rate, and a sound sensor for detecting snoring. The circuit also includes an LCD screen for displaying real-time sensor data and system status. All components are connected as shown, with appropriate power supply lines, grounding, and signal connections to ensure proper functionality and integration of the system.

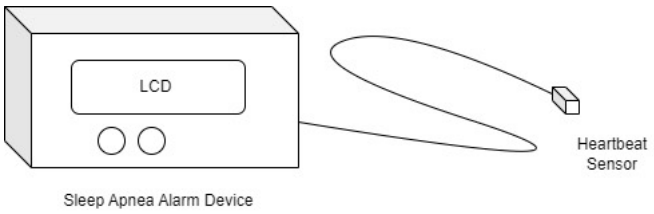

Figure 3: the theoretical set up of the alarm system

After setting up the circuit, the subsequent step involves programming the alarm system. Calibration for heart rate was necessary since the signals received from the heart rate sensor did not directly correspond to the actual heart rate. We conducted calibration by measuring multiple heart rates of our own and raw signals received by Arduino on the same time under dim light condition since the heartbeat sensor is sensitive to visible light. We determined that there is a linear relationship between the raw signal from the sensor and the heart rate, allowing us to map the signal range from 0 to 1023 to a heart rate range of 40 to 180 BPM. The calibration curve can be found in the appendix. To ensure the alarm system functions correctly, multiple conditional loops were implemented to activate the buzzer and LED when specific conditions are met: a high heart rate is detected, the

microphone sensor picks up minimal sound (indicating the absence of snoring), and the PIR sensor detects no movement. The real-time heart rate is displayed on the LCD screen with a 10-millisecond delay. The complete code for the circuit is provided in Appendix A.

4 Results and Discussion

4.1 Testing the Performance of the Sensors

To evaluate the performance of each sensor, multiple tests have been conducted. We tested the range of the PIR sensor and found that it can detect movement up to 4.5 meters away and within a 135-degree field in front of the sensor. Within this range, the PIR sensor is highly sensitive, capable of detecting even slight human movements. The performance of the heart rate sensor was also assessed, with the results displayed in Figure 4 and Figure 5.

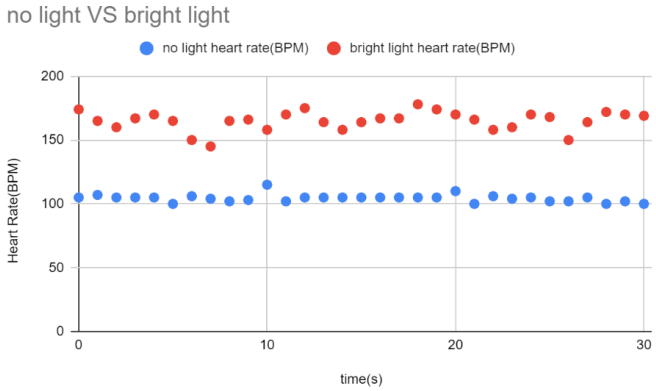

Figure 4: Heart rate measurements taken by the alarm system over time under conditions of no light and bright light.

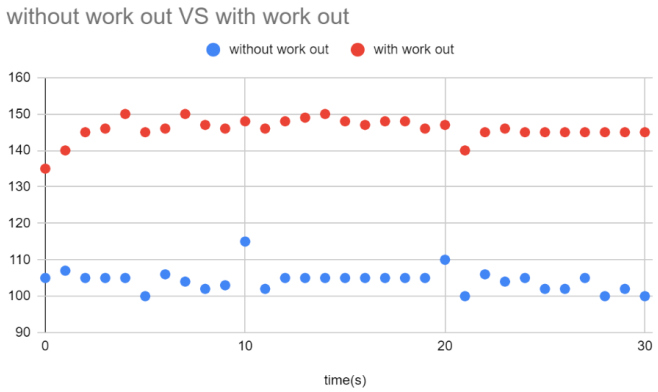

Figure 5: Heart rate measurements taken by the alarm system over time under conditions of before and after exercise.

We conducted tests to record heart rates using the alarm system under both no light and bright light conditions. The results indicate that heart rate readings are higher and show greater variability under bright light conditions. This may be due to interference from surrounding light, which affects the electrical signal on the photodiode, sending additional signals to the Arduino microcontroller and consequently altering the heart rate measurements. Under no light conditions, the heart rate readings remained relatively stable, averaging around 100 BPM, which is within the normal range for humans. Additionally, we measured heart rates before and after a 30-minute workout session. The data clearly demonstrated an increase in heart rate due to exercise, confirming that the heart rate monitor functions as intended under no light condition.

We also measured the frequency range detectable by the microphone sensor. Utilizing a YouTube video that emits sound frequencies ranging from 20 Hz to 20 kHz, we determined that the microphone sensor can effectively detect sounds from 80 Hz to over 1500 Hz. It is important to note that while the sensor is capable of detecting higher frequencies, we chose not to test these limits to avoid risking damage to the sensor. Additionally, the frequency range for human snoring typically varies between 652 and 1500 Hz [**6**], indicating that the microphone sensor is well-suited for snoring detection.

4.2 Limitation and Improvement

Given that this design represents a basic blueprint, there are several ways it could be enhanced to overcome its limitations. A primary limitation is that the heart rate sensor in the alarm system only records real-time data without storing historical data. Consequently, we must rely on setting a threshold for heart rate high enough to trigger the alarm. However, this approach may lead to inaccuracies, as elevated heart rates can also be caused by other factors such as nightmares. A more precise method to detect sleep apnea would be to identify sharp increases in heart rate over a short period. Unfortunately, the current Arduino setup does not allow for comparing current heart rate data with previous data due to its inability to store past readings. To address this issue, integrating an SD card module into the Arduino board would enable data storage, allowing for more accurate and reliable sleep apnea detection.

Additionally, the heart rate sensor's sensitivity, coupled with its calibration based solely on experimental data, may contribute to inaccuracies in readings, potentially affecting the system's performance. To enhance accuracy, incorporating an ECG sensor, which measures the electrocardiogram of an individual, can provide more precise insights into heart behavior during sleep apnea episodes. Furthermore, the current microphone sensor has limitations; it merely detects the presence of sound without distinguishing its source. This could compromise safety in scenarios where suffocation occurs but ambient noise prevents the alarm from triggering. To address this, further development is necessary to refine the microphone's sensitivity, allowing it to specifically detect snoring sounds and reduce interference from other noises. This would ensure more reliable operation of the alarm system under various conditions.

5 Conclusions

In conclusion, while the alarm system presents a foundational framework capable of detecting critical symptoms of sleep apnea and trigger the alarm system accordingly, its

effectiveness can be significantly bolstered by addressing its current limitations. Future developments should focus on enhancing data storage capabilities, improving sensor accuracy, and refining detection algorithms. Such improvements will ensure that the system not only alerts effectively during an apnea episode but also records pertinent data for medical review, ultimately safeguarding the well-being of individuals at risk for sleep apnea.

References

[1] Gellhorn, E. *Cardiovascular Reactions in Asphyxia and the Postasphyxial State*. Am. J. Cardiol. **1964**, *14*(4), 400-410. https://doi.org/10.1016/0002-8703(64)90400-4.

[2] Li, D.; Mabrouk, O. S.; Liu, T.; Tian, F.; Xu, G.; Rengifo, S.; Choi, S. J.; Mathur, A.; Crooks, C. P.; Kennedy, R. T.; Wang, M. M.; Ghanbari, H.; Borjigin, J. *Asphyxia-activated corticocardiac signaling accelerates onset of cardiac arrest*. Proc. Natl. Acad. Sci. U. S. A. **2015**, *112*(16), E2073-E2082. https://doi.org/10.1073/pnas.1423936112.

[3] Campbell, S. *How to Use Microphones on the Arduino*. Circuit Basics, June 7, 2023. Available online: http://www.circuitbasics.com/how-to-use-microphones-on-the-arduino/.

[4] Datta, K. S.; Kumar, L. M.; Kiran, M.; Arun, M. P. S.; Kumar, R. G. V. V. D. S. P. *Heartbeat Sensor Using Arduino*. Pramana Res. J. **2020**, *10*(4), 102.

[5] Lady, A. *Pir Motion Sensor*. Adafruit Learning System. Available online: https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work?gclid=EAIaIQobChMIvvPAqJLJhgMVoBGtBh3GKgBJEAQYASAAEgIyYfD_BwE (accessed June 7, 2024).

[6] Lee, G. S.; Lee, L. A.; Wang, C. Y.; et al. *The Frequency and Energy of Snoring Sounds Are Associated with Common Carotid Artery Intima-Media Thickness in Obstructive Sleep Apnea Patients*. Sci Rep. **2016**, *6*, 30559. https://doi.org/10.1038/srep30559.

A Supplemental Graph

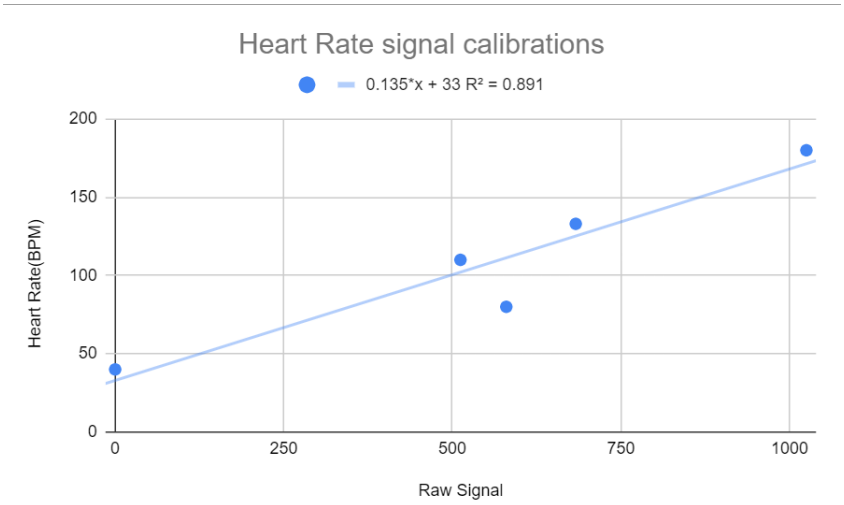

Figure A1: Calibration curve of the heart rate and the raw signal received from the heart rate sensor

Arduino Code for the alarm design

```
#include <LiquidCrystal.h>

// Custom character
byte Character[8] = {
  0b00000,
  0b00000,
  0b01010,
  0b11111,
  0b11111,
  0b01110,
  0b00100,
  0b00000
};

const int heartbeatPin = A0;
const int soundPin = A1;
const int pirPin = 2;
const int buzzerPin = 8;

int heartbeatThreshold = 200; // Example threshold for low heart rate (BPM)
int soundThreshold = 100;     // Example threshold for sound level
int apneaDuration = 10000;    // Time (ms) to consider as potential apnea event

unsigned long apneaStartTime = 0;
bool inApnea = false;
bool buzzerActive = false;

const int numReadings = 5;    // Number of readings to average
int readings[numReadings];    // Array to store the readings
int readIndex = 0;            // Current index in the array
int total = 0;                // Total of readings
int average = 0;              // Average heart rate

// Change the pins for the LCD to avoid conflict with the PIR sensor on pin 2
LiquidCrystal lcd(12, 11, 5, 4, 9, 7); // RS, E, D4, D5, D6, D7

void setup() {
  Serial.begin(9600); // Ensure the Serial Monitor matches this baud rate
  pinMode(heartbeatPin, INPUT);
  pinMode(soundPin, INPUT);
  pinMode(pirPin, INPUT);
  pinMode(buzzerPin, OUTPUT);

  // Create custom character
  lcd.createChar(0, Character);

  // LCD Initialization
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("Initializing...");
  delay(2000); // Wait for 2 seconds

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.write(byte(0)); // Display custom character
  lcd.print(" HR: ");
```

```
  // Initialize readings array
  for (int i = 0; i < numReadings; i++) {
    readings[i] = 0;
  }
}

void loop() {
  // Read the raw value from the heartbeat sensor
  int heartbeatRaw = analogRead(heartbeatPin);

  // Read the sound sensor value
  int soundValue = analogRead(soundPin);

  // Read the PIR sensor state
  int pirState = digitalRead(pirPin);

  // Print data for the Serial Plotter
  Serial.print("HeartbeatRaw:");
  Serial.println(heartbeatRaw);

  // Print data for the Serial Monitor
  int heartbeatBPM = map(heartbeatRaw, 0, 1023, 40, 180);
  Serial.print("Heartbeat BPM: ");
  Serial.print(heartbeatBPM);
  Serial.print(", Sound: ");
  Serial.print(soundValue);
  Serial.print(", PIR: ");
  Serial.println(pirState);

  // Update the readings array
  total = total - readings[readIndex];
  readings[readIndex] = heartbeatBPM;
  total = total + readings[readIndex];
  readIndex = (readIndex + 1) % numReadings;
  average = total / numReadings;

  // Display the average heart rate on the LCD
  lcd.setCursor(6, 0);
  lcd.print(average);
  lcd.print(" BPM ");

  // Check conditions for sleep apnea
  if (pirState == LOW && soundValue < soundThreshold && average < heartbeatThreshold) { // All conditions met
    if (!inApnea) {
      apneaStartTime = millis();
      inApnea = true;
    } else if (millis() - apneaStartTime > apneaDuration) {
      digitalWrite(buzzerPin, HIGH); // Trigger alarm
      buzzerActive = true;
      lcd.setCursor(0, 1);
      lcd.print(" APNEA ALERT ");
    }
  } else {
    inApnea = false;
  }
```