A MINIPROJECT REPORT ON

HEART RATE VARIABILITY ANALYSIS

Submitted in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Electronics and Biomedical Engineering

Of

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

By

ABHIJITH A (MDL22EBE005)

ANIRUDH VIJAYAN E K (MDL22EBE014)

HANNA ANSAR KOLOTH (MDL22EBE034)

JOEL JACOB (MDL22EBE040)

Under the supervision of

Dr. SAJITHA S

DEPARTMENT OF BIOMEDICAL ENGINEERING

MODEL ENGINEERING COLLEGE

KOCHI-682021

http:/www.mec.ac.in

APRIL 2025

DEPARTMENT OF BIOMEDICAL ENGINEERING
MODEL ENGINEERING COLLEGE
KOCHI-682021

CERTIFICATE

*This is to certify that, the report titled "**HEART RATE VARIABILITY ANALYSIS** " is a bonafide account of the **EBD 334: MINI PROJECT** presented by **JOEL JACOB**(MDL22EBE040) Sixth Semester B. Tech in Electronics and Biomedical, in partial fulfilment of the requirements for the award of the Bachelor's degree, **B. Tech in Electronics and Biomedical Engineering** from APJ Abdul Kalam Technological University during the academic year 2024-2025.*

Guide	Project Coordinator	Project Coordinator
Dr. Sajitha S	**Ms. Vijayalakshmi.K**	**Ms. Geethu Gilbert**
(Assistant Professor)	(Assistant Professor)	(Assistant professor)

Head of Department

Dr. Minimol B
(Associate Professor)

ACKNOWLEDGEMENT

First and foremost, we thank God Almighty for his blessings without which we would not have been able to complete our work in time. We express our sincere gratitude to **Dr. Mini M G**, Principal, Govt. Model Engineering College, for giving us this opportunity. We express our gratitude to **Dr. Minimol B**, HOD, Department of Biomedical Engineering, for her constant support and valuable advice. We also extend our sincere gratitude to our project coordinators, **Ms. Vijayalakshmi K**, Assistant Professor and **Ms. Geethu Gilbert**, Assistant Professor, Department of Biomedical Engineering, for allowing us to use the resources available and giving us valuable suggestions. We hereby express our overwhelming thanks to our guide, **Dr. Sajitha S**, Assistant Professor, for guiding us throughout this venture giving us crucial suggestions, and correcting our mistakes all along. We would also thank all other faculty members who encouraged and helped us to complete this mini project successfully. Last but not least we would like to thank all fellow classmates who gave us constant support and encouraged us to complete this venture on time.

ABSTRACT

This project presents the development of a cost-effective and compact device for acquiring and analysing 5-minute ECG data to evaluate Heart Rate Variability (HRV). The device uses an AD8232 ECG sensor and an Arduino Uno to record the electrical activity of the heart. Data is processed offline using Python to extract HRV metrics that reflect the balance and activity of the autonomic nervous system (ANS). HRV is a well-established marker of physiological well-being and is sensitive to stress, fatigue, and various cardiovascular and metabolic conditions. The system is designed to be non-invasive, easy to use, and suitable for repeated measurements, making it ideal for both clinical and personal health applications. The ability to detect shifts in HRV patterns can support early identification of health issues, aid in stress monitoring, and contribute to long-term wellness tracking. With minimal hardware requirements and open-source software, the device is low-cost and accessible, and it holds promise for future development into a portable, standalone HRV assessment tool.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

HRV	Heart Rate Variability
ECG	Electrocardiogram
SDNN	Standard Deviation of NN interval
RMSSD	Root Mean Square of Successive Differences
pNN50	Percentage of RR intervals differing by more than 50ms
LF	Low Frequency
HF	High Frequency
VLF	Very Low Frequency
FFT	Fast Fourier Transform
PSD	Power Spectral Density
PNS	Parasympathetic Nervous System
SNS	Sympathetic Nervous System
AR	Autoregressive Modelling
TVAR	Time-varying Autoregressive Model
RLD	Right Leg Drive (Electrode)
ADC	Analog to Digital Convertor

CHAPTER 1

INTRODUCTION

This project focuses on the development of a low-cost, offline Heart Rate Variability (HRV) monitoring system based on a 5-minute ECG signal. The system is built using an AD8232 ECG sensor interfaced with an Arduino Uno microcontroller for ECG acquisition. The ECG data is transferred to a laptop where it is processed using Python-based algorithms for signal enhancement, R-peak detection, and HRV analysis. The analysis is conducted offline and includes computation of RR intervals and HRV metrics, which are used to interpret autonomic nervous system activity. This system provides a simplified, non-invasive, and accessible solution for monitoring cardiovascular health outside clinical settings.

1.1 Motivation for the project

Traditional HRV monitoring requires sophisticated, high-cost medical equipment typically available only in clinical or research environments. This restricts access for individuals, fitness enthusiasts, and those in remote or under-resourced areas. Athletes and yoga practitioners can benefit from routine HRV monitoring to track physical recovery and stress levels, while others may use it for general wellness or chronic disease management. This project is driven by the need for a stable, affordable, and repeatable ECG acquisition method that makes HRV monitoring more widely available. The proposed system can bridge the gap between clinical setups and personalized health tracking by offering a low-cost and practical alternative for 5-minute ECG-based HRV analysis.

1.2 Problems in Traditional HRV Analysis

Traditionally, HRV analysis has been performed using high-end medical equipment in hospitals or specialized labs. However, these systems are expensive, making HRV monitoring inaccessible to individuals and small research facilities. Moreover, real-time HRV monitoring is often limited to clinical environments, preventing continuous health tracking outside of medical settings. Noise interference, motion artifacts, and baseline drift also present challenges in acquiring clean ECG signals for accurate HRV analysis.

1.3 Proposed Solution: ECG Acquisition and HRV Monitoring System

Advancements in embedded systems and low-cost sensor technologies have enabled the development of an affordable and portable HRV monitoring system using the AD8232 ECG sensor. By integrating this sensor with an Arduino microcontroller and processing the data using Python, a cost-effective, real-time HRV analysis system can be created. This project focuses on real-time signal acquisition, pre-processing, R-peak detection, and HRV computation using various methods such as time-domain, frequency-domain, and non-linear analysis.

To enhance signal quality, filtering techniques are applied, including a bandpass filter (0.5–40 Hz) to remove baseline drift and high-frequency noise, and a notch filter (50 Hz) to eliminate power line interference. Adaptive thresholding is used for R-peak detection. Power spectral analysis is performed using Welch’s method to estimate LF and HF power, allowing assessment of autonomic nervous system function. Data visualization includes ECG waveforms, RR intervals, and HRV metrics, providing real-time insights. To improve accuracy and usability, the project incorporates multiple phases of optimization. The R-peak detection algorithm is fine-tuned for high sensitivity and specificity. HRV metrics such as SDNN, RMSSD, and LF/HF ratio are validated against Kubios HRV software. The system undergoes iterative testing to optimize filtering parameters and peak detection accuracy. Additional improvements include automated report generation, interactive visualization, and real-time monitoring. Future advancements may include integrating wireless communication for remote monitoring, implementing machine learning algorithms for automated classification of heart conditions, and developing a mobile application for user-friendly ECG tracking.

1.4 Heart Rate Variability (HRV)

HRV is a well-established physiological marker reflecting autonomic nervous system function and overall cardiovascular health. It is defined as the variation in time intervals between consecutive heartbeats (RR intervals). HRV analysis provides insights into heart function, stress levels, and overall well-being, making it valuable for assessing conditions such as arrhythmias, myocardial infarction, and stress-related disorders.

HRV can be analysed using three primary methods:

Time-domain analysis: Includes metrics like SDNN (standard deviation of NN intervals) and RMSSD (root mean square of successive differences) to measure heart rate fluctuations.

Frequency-domain analysis: Uses power spectral density estimation to divide HRV into low-frequency (LF: 0.04-0.15 Hz) and high-frequency (HF: 0.15-0.4 Hz) components, which reflect sympathetic and parasympathetic activity, respectively. A higher LF/HF ratio indicates increased sympathetic dominance, often linked to stress and cardiovascular risk.

Non-linear analysis: Includes Poincaré plots to visualize HRV dynamics and detect abnormalities in heart rate regulation.

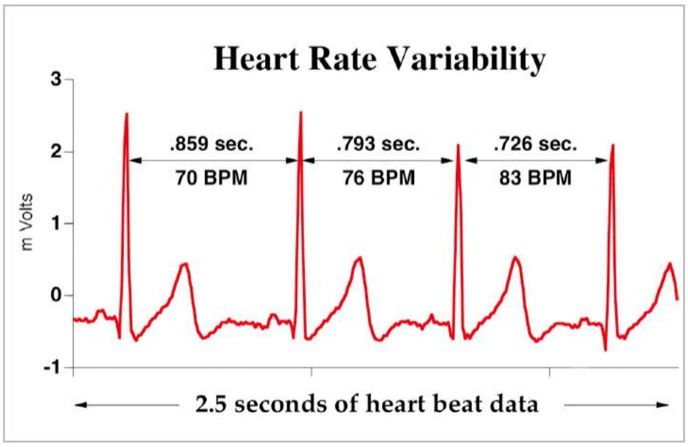


Fig 1.4 Heart Rate Variability (HRV)

1.5 Electrocardiogram (ECG) Analysis and HRV Monitoring

This project aims to bridge the gap between clinical HRV monitoring and personal health tracking by developing a real-time ECG acquisition and analysis system. The hardware setup includes the AD8232 ECG sensor interfaced with an Arduino Uno for continuous signal acquisition. Data is collected at a 250 Hz sampling rate for high-resolution ECG signals. Pre-processing techniques, including bandpass filtering and baseline correction, enhance signal clarity. RR intervals are extracted from detected R-peaks, and HRV parameters are computed and analysed in real-time.

By leveraging affordable hardware and efficient signal processing techniques, this system allows users to monitor their heart health effectively. Applications extend beyond personal wellness tracking to include stress analysis, athletic training, and telemedicine. Insights derived from HRV can support early diagnosis and risk assessment for cardiovascular diseases, making the system a valuable tool for both research and healthcare applications.

CHAPTER 2

LITERATURE SURVEY

1. **Tarvainen, Mika P., Stefanos D. Georgiadis, Perttu O. Ranta-Aho, and Pasi A. Karjalainen. "Time-varying analysis of heart rate variability signals with a Kalman smoother algorithm."** ***Physiological measurement*** **27, no. 3 (2006): 225.**

Heart Rate Variability (HRV) analysis is a valuable tool for assessing autonomic nervous system function. Traditional spectral analysis methods like Short-Time Fourier Transform (STFT) and Welch's method are widely used but struggle with nonstationary signals. Wavelet transforms offer multi-resolution analysis but are computationally complex, while time-frequency distributions such as the Wigner distribution suffer from interpretation challenges due to cross-term interference.

Autoregressive (AR) models have emerged as a superior alternative for HRV spectral estimation. These models provide higher frequency resolution and enable the decomposition of HRV signals into distinct spectral components. Studies have demonstrated their effectiveness in capturing transient physiological changes more accurately than nonparametric methods, particularly in tracking the dynamic variations of low-frequency (LF) and high-frequency (HF) components. Despite their advantages, AR models face challenges in real-time HRV analysis due to difficulties in estimating time-varying parameters. Adaptive filters like the Least Mean Square (LMS) and Recursive Least Squares (RLS) methods have been used for tracking changes, but they suffer from lag errors as they rely only on past data. The Kalman filter improves recursive estimation but still exhibits lag, limiting its application in rapidly fluctuating HRV conditions.

The Kalman smoother was introduced to overcome these issues by incorporating both past and future observations, significantly reducing lag errors. This results in more accurate spectral estimation and better detection of physiological changes, such as those occurring during an orthostatic test. Research has shown that the Kalman smoother outperforms other adaptive filtering techniques by providing clearer HRV variations over time. A key advantage of the Kalman smoother is its ability to decompose HRV spectra into distinct LF and HF components, improving the understanding of autonomic regulation. Additionally, error propagation techniques have been introduced to quantify uncertainties in spectrum estimates, increasing the reliability of HRV analysis for clinical and research applications.

Overall, HRV spectral analysis has evolved toward more adaptive and precise methodologies. The Kalman smoother method is a significant advancement, offering improved accuracy in tracking HRV dynamics under nonstationary conditions. Its potential applications in real-time physiological monitoring and clinical diagnostics make it a promising tool for future cardiovascular health assessments.

2. **Tarvainen, Mika P., J -P. Niskanen, J. A. Lipponen, P. O. Ranta-Aho, and P. A. Karjalainen. "Kubios HRV—a software for advanced heart rate variability analysis." In *4th European Conference of the International Federation for Medical and Biological Engineering: ECIFMBE 2008 23–27 November 2008 Antwerp, Belgium*, pp. 1022-1025. Springer Berlin Heidelberg, 2009.**

Heart Rate Variability (HRV) analysis is essential for understanding autonomic nervous system regulation, particularly in distinguishing sympathetic and parasympathetic influences. Traditional HRV analysis methods include time-domain, frequency-domain, and nonlinear approaches. Time-domain methods involve statistical measures such as the standard deviation of RR intervals (SDNN) and the root mean square of successive differences (RMSSD). These measures provide insight into both short-term and long-term variations in heart rate. Frequency-domain methods, such as Welch's periodogram and autoregressive (AR) modelling, estimate the power spectral density of HRV signals. These methods decompose HRV into distinct frequency bands, including very low frequency (VLF), low frequency (LF), and high frequency (HF). The LF/HF ratio is often used as an index of sympatho-vagal balance, reflecting the interplay between sympathetic and parasympathetic activity. AR modelling allows for spectral decomposition, which can improve the accuracy of band power estimation. However, incorrect decomposition may lead to distorted results, necessitating careful implementation.

Recent advancements in HRV analysis include nonlinear methods, such as Poincaré plots, entropy measures, and detrended fluctuation analysis (DFA). These techniques capture complex dynamics in HRV that traditional time- and frequency-domain methods may overlook. The introduction of software like Kubios HRV has facilitated advanced HRV analysis by integrating multiple analytical techniques into a user-friendly interface. Kubios HRV provides tools for artifact correction, trend removal, and comprehensive HRV parameter calculation, making it a widely adopted solution in research and clinical applications.

As HRV analysis continues to evolve, combining traditional and advanced methods enhances its applicability in physiological monitoring. Software tools with adaptable algorithms allow researchers to explore HRV in greater depth, providing more accurate insights into autonomic function. Future developments may focus on real-time HRV assessment and integration with wearable technology for continuous health monitoring.

3. **Abhishekh, Hulegar A., Palgun Nisarga, Ravikiran Kisan, Adoor Meghana, Sajish Chandran, Trichur Raju, and Talakad N. Sathyaprabha. "Influence of age and gender on autonomic regulation of heart."** ***Journal of clinical monitoring and computing*** **27 (2013): 259-264.**

Heart Rate Variability (HRV) is a widely used non-invasive method for assessing autonomic regulation of the heart. It is influenced by multiple factors, including age and gender. Research suggests that HRV decreases with age due to a decline in parasympathetic activity and a relative increase in sympathetic dominance. The loss of vagal tone with aging has been observed in several studies, indicating a progressive reduction in autonomic control of the heart. Additionally, HRV has been linked to various physiological and pathological conditions, including cardiovascular diseases, stress, and psychiatric disorders.

Gender differences in HRV have also been extensively studied. Females generally exhibit higher parasympathetic activity, as reflected in increased high-frequency (HF) power, while males show higher sympathetic activity, indicated by greater low-frequency (LF) power and LF/HF ratio. The cardioprotective effect in females has been hypothesized to result from oestrogen's role in enhancing vagal modulation. Hormonal influences, particularly oestrogen and its interaction with autonomic function, contribute to these differences. Studies have also indicated that females have a lower risk of cardiovascular diseases compared to males due to this autonomic balance.

Most earlier studies assessing the effects of age and gender on autonomic function relied on 24-hour HRV recordings. However, frequency-domain measures have been found to be more accurate in short-term HRV analysis. Short-term HRV recordings have provided insights into autonomic regulation and have been used in studies analysing HRV in relation to various factors, including stress, lifestyle, and disease conditions. The findings consistently demonstrate a decline in HRV with aging and gender-based variations in autonomic function.

Understanding the impact of age and gender on HRV is crucial for assessing cardiovascular health and predicting potential risks. The shift towards increased sympathetic dominance with age and the gender-specific autonomic differences provide insights into individual susceptibility to cardiovascular diseases. Further studies, particularly those incorporating hormonal assessments and real-time HRV monitoring, could help clarify the mechanisms underlying these autonomic changes and their implications for long-term health.

CHAPTER 3

RELATED THEORY

3.1 HEART RATE VARIABILITY (HRV):

Heart Rate Variability (HRV) refers to the variation in time intervals between consecutive heartbeats, also known as R-R intervals (or inter-beat intervals, IBI). It is a key indicator of the autonomic nervous system (ANS) function and cardiac health. HRV is influenced by the balance between the sympathetic nervous system (SNS), which accelerates heart rate, and the parasympathetic nervous system (PNS), which slows it down.

The human heart does not beat at a constant interval; instead, there are small fluctuations in beat-to-beat timing due to autonomic regulation. This variability is caused by:

Respiratory Sinus Arrhythmia (RSA): During inhalation, heart rate increases (sympathetic activation), and during exhalation, it decreases (parasympathetic activation).

Baroreceptor Reflex: Sensors in the aorta and carotid arteries help regulate blood pressure and heart rate through autonomic feedback mechanisms.

Neurohormonal Influence: Hormones like adrenaline and cortisol affect HRV, particularly under stress.

A high HRV generally indicates good cardiovascular health and adaptability, while a low HRV is associated with stress, fatigue, and increased risk of cardiovascular diseases.

3.1.1 ECG SIGNAL ACQUISITION AND PRE-PROCESSING

Electrocardiogram (ECG) signals are recorded using the AD8232 module, which is connected to an Arduino Uno. The Arduino continuously samples the ECG signal and transmits the data via a serial connection to a laptop for further analysis.

Band pass filtering (Butterworth filter)

To remove noise and unwanted frequency components from the ECG signal, a Butterworth bandpass filter is applied. The bandpass filter allows frequencies in the range of **0.5 Hz to 35 Hz** to pass through, eliminating baseline wander and high-frequency noise.

Implementation: scipy.signal.butter() and scipy.signal.filtfilt()

$$H(s) = \frac{1}{\sqrt{1 + (\frac{f}{f_c})^{2n}}}$$

Moving Average Smoothing

A moving average filter is applied to smooth the ECG signal further, reducing noise and enhancing peak detection.

Implementation**:** numpy.convolve()

$$y[n] = \frac{1}{N} \sum_{k=0}^{N-1} x[n-k]$$

R-peak detection

The R-peaks in the ECG signal are identified using the find_peaks() function from SciPy. The method sets a height threshold and a minimum distance between consecutive peaks to detect the most prominent R-peaks.

Implementation: using scipy.signal.find_peaks ()

3.1.2 HRV MEASUREMENT AND ANALYSIS

Time-Domain Analysis

Time-domain methods measure the statistical variations in R-R intervals over a period. Some common parameters include:

Mean RR interval: The average time between successive R-peaks.

$$\text{Mean RR} = \frac{1}{N}\sum_{i=1}^{N} RR_i$$

Mean Heart Rate (HR):

$$\text{HR (bpm)} = \frac{60000}{\text{Mean RR (ms)}}$$

SDNN (Standard Deviation of NN Intervals): Measures overall HRV and reflects both sympathetic and parasympathetic activity.

$$\text{SDNN} = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N}\left(\text{RR}_j - \overline{\text{RR}}\right)^2}$$

RMSSD (Root Mean Square of Successive Differences): Represents short-term HRV, mainly controlled by the parasympathetic nervous system.

$$\text{RMSSD} = \sqrt{\frac{1}{N-1}\sum_{j=1}^{N-1}\left(\text{RR}_{j+1} - \text{RR}_j\right)^2}$$

pNN50: Percentage of successive R-R intervals differing by more than 50ms, indicating vagal (parasympathetic) influence.

$$\text{pNN50} = \frac{\text{NN50}}{N-1} \times 100\%$$

Frequency Domain Analysis

Frequency-domain analysis decomposes HRV into different frequency components using **Welch's Periodogram Method**.

Welch's Power Spectral Density (PSD) Estimation:

Implementation: scipy.signal.welch()

$$P_{xx}(f) = \frac{1}{L}\sum_{i=1}^{L} |X_i(f)|^2$$

Low Frequency (LF) (0.04 - 0.15 Hz): Reflects both sympathetic and parasympathetic activity, often linked to baroreceptor function.

High Frequency (HF) (0.15 - 0.40 Hz): Represents parasympathetic activity and respiratory influence.

$$\text{Power} = \int_{f_1}^{f_2} PSD(f)df$$

LF/HF Ratio: Used to estimate the balance between sympathetic and parasympathetic modulation.

$$\text{LF/HF Ratio} = \frac{\text{LF Power}}{\text{HF Power}}$$

Non-linear HRV Analysis

The heart is regulated by a complex interplay of physiological systems, particularly the autonomic nervous system (ANS), which governs involuntary functions such as heart rate, respiration, and blood pressure. These interactions often lead to non-linear and chaotic patterns in heart rate dynamics that may not be fully captured by traditional linear methods like time-domain and frequency-domain analysis. Therefore, non-linear analysis methods have been introduced to better understand the underlying complexity and adaptability of cardiac regulation.

SD1(Short-term variability): Measures the dispersion of points perpendicular to the line of identity (the 45° diagonal line in the plot). SD1 primarily reflects parasympathetic (vagal) activity and is closely related to short-term fluctuations in heart rate. It is mathematically equivalent to RMSSD, a time-domain feature.

$$SD1 = \sqrt{\frac{1}{2}\mathrm{Var}(RR_{n+1} - RR_n)}$$

SD2(Long-term variability): Measures the dispersion of points along the line of identity. SD2 captures both sympathetic and parasympathetic influences and reflects the longer-term variability in heart rate dynamics.

$$SD2 = \sqrt{2\mathrm{Var}(RR_n) - \frac{1}{2}\mathrm{Var}(RR_{n+1} - RR_n)}$$

SD1/SD2 ratio: This ratio serves as an index of autonomic balance. A lower ratio suggests sympathetic dominance, while a higher ratio indicates greater parasympathetic activity. In clinical research, changes in this ratio have been associated with conditions like anxiety, depression, diabetes, and cardiovascular diseases.

Poincaré Plot Analysis: A Poincaré plot is a non-linear tool used to visualize heart rate variability. It is a scatter plot in which each RR interval is plotted against the next successive RR interval. The resulting shape provides qualitative insights into the autonomic regulation of the heart. A tighter, elliptical cluster suggests reduced variability, often associated with stress or disease. A wider, more dispersed shape indicates higher variability, reflecting healthy autonomic function.

While time-domain and frequency-domain metrics provide valuable statistical and spectral insights, non-linear analysis uniquely captures the complexity, unpredictability, and fractal-like nature of heart rate behaviour. These methods can detect subtle changes in autonomic regulation that linear methods may overlook. They are especially useful in detecting early signs of autonomic dysfunction, differentiating between healthy and pathological states, assessing physiological adaptability and stress resilience, enhancing diagnosis in cases where traditional HRV metrics appear normal

Therefore, integrating non-linear analysis with conventional HRV metrics offers a more comprehensive understanding of heart rhythm dynamics and overall autonomic nervous system function.

3.2 HARDWARE

The hardware components of this project are carefully selected to enable real-time ECG signal acquisition and heart rate variability (HRV) analysis. The system consists of an AD8232 ECG sensor module for capturing bio-potential signals, an Arduino Uno for analog-to-digital conversion and serial data transmission, and a laptop for signal processing using Python-based algorithms. The AD8232 provides a clean and amplified ECG signal, while the Arduino ensures seamless data acquisition and transmission. The laptop processes the signal using filtering techniques, peak detection, and HRV computation. Together, these components form a compact and efficient system for analysing cardiac activity.

3.2.1 ARDUINO UNO

Memory – AVR CPU up to 16 MHz, 32KB Flash, 2KB SRAM, 1KB EEPROM.
Arduino UNO is an open-source microcontroller board based on the Microchip ATmega328P.
The board has 14 digital I/O pins,6 analog I/O pins and is programmable with Arduino IDE, via a type B USB cable.

Fig 3.2.1 Arduino UNO

3.2.2 AD8232 ECG MODULE

Analog type output

Operating voltage: 3.3V DC

Low current consumption: 170 uA

Noise rejection at 60Hz: 80dB

High gain (G = 100), with DC current blocking

Integrated Right Leg Amplifier (RLD)

RFI filtering

Shutdown pin

Electrode input: Mini Plug 3.5mm

Configurations: 2 or 3 electrodes

Fig 3.2.2 AD8232 ECG MODULE

3.2.3 LAPTOP INTERFACING

A laptop is used for real-time processing and HRV analysis. The ECG data, acquired by the Arduino and ADS1115, is transmitted to the laptop via serial communication for further processing using Python-based algorithms. The system supports real-time data visualization, allowing for continuous monitoring of both ECG and HRV signals. Signal processing techniques such as QRS detection and RR interval extraction are implemented to enable accurate HRV analysis. Additionally, the setup is compatible with machine learning models, allowing for future integration aimed at cardiac health prediction.

Fig 3.2.3 Laptop

CHAPTER 4

METHODOLOGY AND IMPLEMENTATION

This chapter describes the methodology and implementation of Heart Rate Variability (HRV) analysis using an AD8232 ECG module, Arduino Uno, and a laptop. The acquired ECG signal is processed using Python to extract meaningful HRV metrics using various signal processing and mathematical techniques. The process includes signal acquisition, pre-processing, feature extraction, and analysis in both time and frequency domains, along with nonlinear HRV analysis.

4.1 BLOCK DIAGRAM

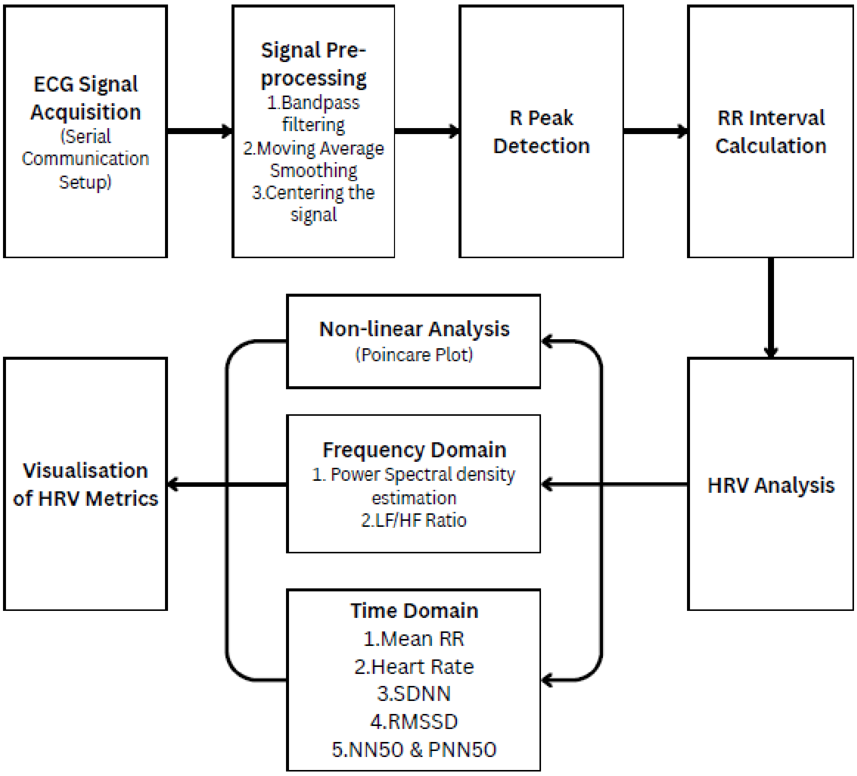


Fig 4.1 Block Diagram

4.1.1 ECG SIGNAL ACQUISITION

Serial Communication Setup

The ECG data is acquired using an Arduino Uno connected to an AD8232 ECG module. The Arduino transmits the raw ECG signal via a serial connection at a baud rate of 9600. The data acquisition is managed using the Python `serial` library.

The code initializes the serial communication by connecting to the specified COM port. The system collects ECG data for a defined duration (300 seconds) with a sampling frequency of 250 Hz. The received data is validated, ensuring only numerical values are stored in an array. If communication issues arise, appropriate error handling mechanisms provide warnings or terminate execution.

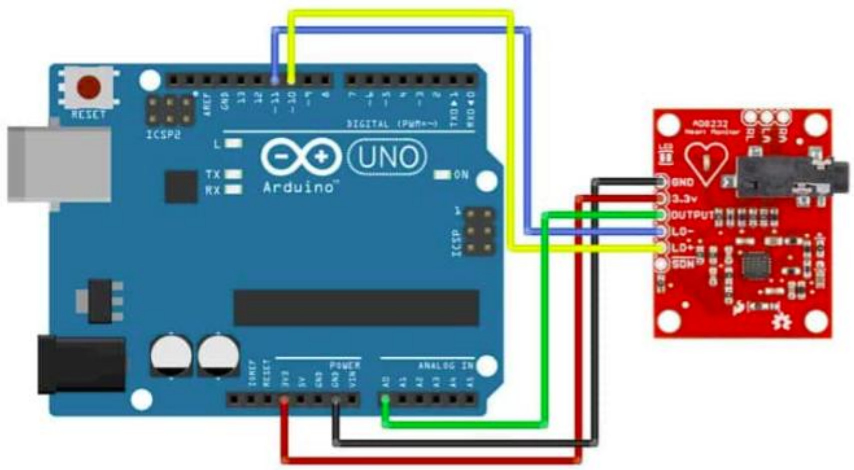


Fig 4.1.1 Interfacing of AD8232 with Arduino UNO

4.1.2 SIGNAL PRE-PROCESSING

1. Bandpass Filtering

A Butterworth bandpass filter (0.5 Hz – 35 Hz) is applied to remove unwanted noise and baseline wander from the raw ECG signal. The butter_bandpass function designs the filter using a 4th-order Butterworth filter, ensuring effective noise suppression. To maintain signal integrity, the filter is applied using filtfilt for zero-phase filtering, preventing any phase distortion. However, if the signal length is too short, lfilter is used as an alternative filtering approach to ensure proper signal processing. This ensures no phase distortion, preserving the alignment of QRS complexes.

2. Moving Average Smoothing

A moving average filter (window size = 7 samples) smoothens the signal to reduce minor fluctuations and enhance peak detection accuracy. This operation serves as a low-pass filter, smoothing out transient noise while preserving the prominent peaks necessary for accurate R-peak detection.

3.Centering the Signal

The mean value of the ECG signal is subtracted to centre it around zero, improving the robustness of peak detection and feature extraction. Centering improves the stability and reliability of peak detection algorithms. It normalizes the baseline and minimizes bias caused by signal drift or sensor offset.

4.1.3 R PEAK DETECTION

A height threshold set to 40% of the maximum signal amplitude. A minimum distance of 0.3 seconds (75 samples at 250 Hz) between peaks. The detected peaks correspond to R-peaks in the ECG, used for further HRV analysis.

4.1.4 RR INTERVAL CALCULATION

Computing RR Intervals

The RR intervals are derived as the time differences between consecutive R-peaks. These intervals are expressed in milliseconds. RR intervals (samples) are computed using `np.diff(r_peaks)`. They are converted to milliseconds using:

```
rr_intervals_ms = (rr_intervals_samples / fs) * 1000.
```

RR Interval Plotting

The RR intervals are plotted over time to visualize variations and trends, which are crucial for HRV analysis.

4.1.5 TIME DOMAIN HRV ANALYSIS

Various statistical measures are extracted from the RR intervals to quantify HRV. The Mean RR Interval (ms) represents the average time between successive R-peaks, while the Mean Heart Rate (bpm) is calculated as 60000 divided by the mean RR interval. The Min and Max Heart Rate (bpm) indicate the lowest and highest heart rates derived from RR intervals. SDNN (Standard Deviation of RR Intervals) serves as a key indicator of HRV, reflecting autonomic nervous system activity. RMSSD (Root Mean Square of Successive Differences) quantifies short-term HRV by measuring the variability between successive RR intervals. Lastly, NN50 and PNN50 represent the count and percentage of successive RR intervals that differ by more than 50 milliseconds, providing further insights into heart rate dynamics.

4.1.6 FREQUENCY DOMAIN HRV ANALYSIS

Interpolation for Spectral Analysis

Since RR intervals are unevenly spaced, interpolation is performed using `interp1d` to create an evenly sampled signal before spectral analysis.

Power Spectral Density (PSD) Estimation

The Welch method is used to estimate the power spectral density of RR intervals. The signal is resampled at 4 Hz to ensure adequate frequency resolution. The power is calculated in two frequency bands:

Low Frequency (LF: 0.04 – 0.15 Hz) – Indicates sympathetic and parasympathetic activity.

High Frequency (HF: 0.15 – 0.40 Hz) – Represents parasympathetic activity.

The LF/HF Ratio is computed as a measure of autonomic balance.

4.1.7 NON-LINEAR HRV ANALYSIS (Poincaré plot)

A Poincaré plot visualizes RR interval variability. RR(n) vs RR(n+1) pairs are plotted. SD1 (short-term variability) and SD2 (long-term variability) are calculated. The SD2/SD1 ratio quantifies autonomic balance. An ellipse is overlaid to represent variability, with its major and minor axes corresponding to SD1 and SD2.

4.1.8 VISUALISATION OF HRV METRICS

The processed ECG and HRV features are visualized through multiple plots, each serving a distinct purpose. The Raw ECG Signal displays the original data before any processing, while the Filtered ECG Signal illustrates the effect of noise removal. The R-Peak Detection Plot highlights the detected R-peaks within the ECG signal. The RR Interval Distribution represents variations in RR intervals using a histogram, whereas the Power Spectrum Plot shows the distribution of HRV power across different frequency bands. Finally, the Poincaré Plot visualizes nonlinear HRV dynamics, providing insights into heart rate variability.

4.2 CIRCUIT DIAGRAM

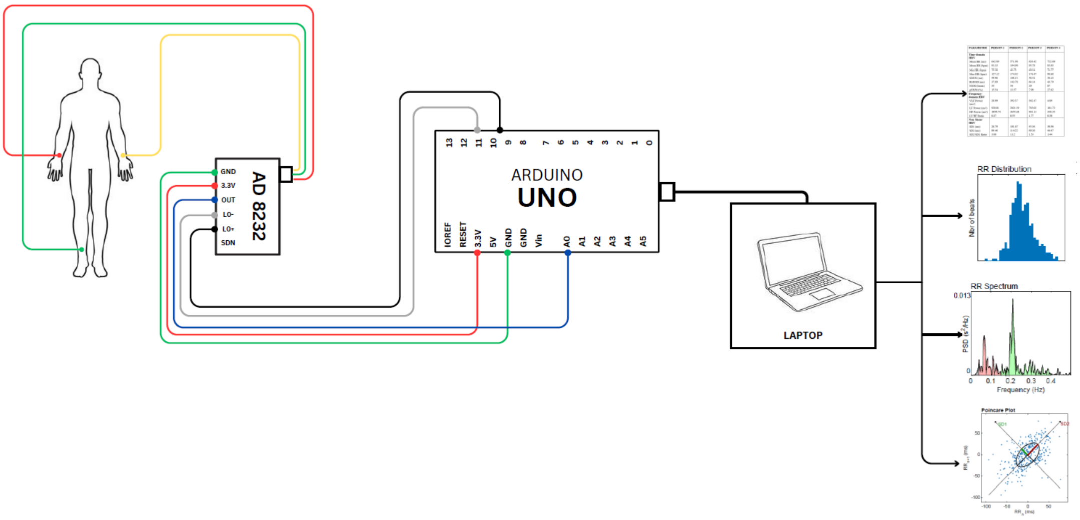


Fig 4.2 Circuit Diagram

In this project, the AD8232 module is used for real-time ECG signal acquisition. The module is connected to an Arduino Uno, which reads the analog ECG signal and transmits it to a laptop over a serial connection. The data is then captured and processed using Python. The raw signal often contains noise from various sources such as muscle movements, electrical interference, or baseline wander due to respiration. To address this, the code first applies a Butterworth bandpass filter between 0.5 and 35 Hz, which isolates the frequency range relevant to cardiac activity while removing low-frequency drift and high-frequency noise. This filtered signal is then smoothed using a moving average technique, which helps in reducing minor fluctuations and enhancing the prominence of R-peaks, which are the sharpest and most distinct parts of the ECG waveform. The signal is also centered by subtracting its mean to ensure it oscillates around zero, which improves the accuracy of peak detection.

R-peak detection is then performed using a peak-finding algorithm, where a dynamic threshold based on signal amplitude is used to ensure only significant peaks are selected, and a minimum distance is enforced between peaks to avoid detecting false positives. Once the R-peaks are identified, the time difference between successive peaks is calculated to obtain RR intervals, which serve as the foundation for heart rate variability (HRV) analysis. Several time-domain HRV metrics are computed, including the average heart rate, the standard deviation of RR intervals, and measures like RMSSD and pNN50, which reflect short-term variations in heart rate and are commonly used to assess parasympathetic activity.

For frequency-domain analysis, the RR intervals are interpolated to create an evenly spaced time series suitable for spectral analysis. Power spectral density is then estimated using Welch's method, which allows quantifying the distribution of power across different frequency bands. These bands include low-frequency and high-frequency components, which are associated with sympathetic and parasympathetic nervous system activity, respectively. The code calculates the absolute and relative power in each band and the LF/HF ratio, which is often used as an indicator of autonomic balance.

In addition to linear methods, the project incorporates nonlinear analysis through the generation of Poincaré plots. These plots visually represent the relationship between consecutive RR intervals, offering insight into the heart's beat-to-beat variability. From these plots, two geometrical features are extracted: SD1, representing short-term variability, and SD2, representing long-term variability. The ratio of SD1 to SD2 provides additional context on the autonomic control of heart function. Nonlinear methods like these are especially valuable because they can capture patterns and dynamics in heart rate that may not be visible through traditional linear analysis, making them a powerful complement in HRV research.

4.2 ALGORITHM AND FLOWCHART

The software implementation utilizes Arduino IDE for programming the Arduino Uno to acquire ECG signals from the AD8232 sensor and transmit data via serial communication. Visual Studio Code is used for Python-based signal processing and HRV analysis, leveraging libraries like NumPy, SciPy, and Matplotlib. The algorithm involves ECG acquisition, pre-processing, R-peak detection, RR interval calculation, and HRV analysis in time, frequency, and nonlinear domains. This integration ensures efficient real-time data collection, processing, and visualization for comprehensive heart rate variability analysis.

4.3.1 Algorithm for ECG Acquisition and Signal Pre-processing

4.3.1.1 ECG Signal Acquisition using Arduino

Algorithm

Step 1: Start

Step 2: Start serial communication at 9600 baud rate to transmit ECG data to the computer.

Step 3: Read the ECG Signal from analog pin from A0 of the Arduino

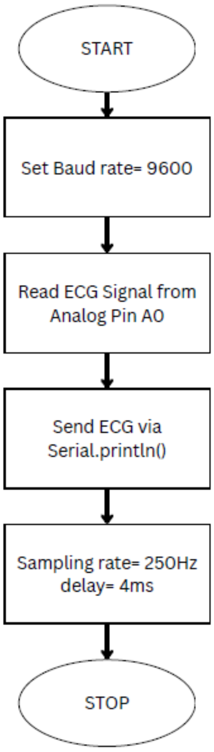
Step 4: Convert the analog signal (0–1023 ADC value) into digital format.

Step 5: Send the ECG signal data via Serial.println() for real-time monitoring.

Step 6: Maintain a sampling rate of 250 Hz by adding a 4 ms delay (delay (4) ;).

Step 7: Stop

Flowchart



4.3.1.2 ECG Signal Acquisition and Pre-processing using Python

Algorithm

Step 1: ECG Data Acquisition

Step 2: Initialise Serial Communication with Arduino at 9600 baud rate.
Step 3: Define parameters:

Serial port (COM4 or other, as required)

Sampling frequency (fs=250 Hz)

Total duration (DURATION= 300 sec)

Step 4: Read ECG Signal from Serial Input

Continuously read data from analog pin A0 of Arduino.

Convert the raw signal (0–1023 ADC values) into an ECG waveform.

Store the data into an array for further processing.

Stop acquisition when the required number of samples is reached.

Step 5: Close the Serial Connection once data collection is complete.

Step 6: Pre-processing (Noise Reduction)

Step 7: Apply Bandpass filter (0.5-35 Hz)

Step 8: Apply Moving Average filter (compute 7-point moving average to Smoothen the signal)

Step 9: Centre the ECG Signal (subtract the mean to remove baseline wandering)

Step 10: R-Peak Detection

Step 11: Compute threshold (40% of the maximum amplitude).

Step 12: Set minimum distance between consecutive peaks (0.3 sec = 75 samples).

Step 13: Use find_peaks () function to detect R-peaks.

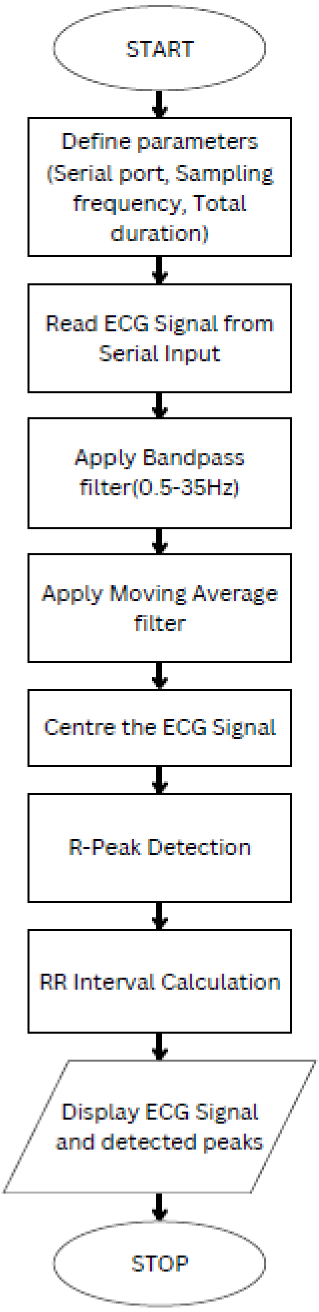
Step 14: RR Interval Calculation

Step 15: Calculate time difference between successive R-peaks.

Step 16: Convert RR intervals into milliseconds ((RR_samples / fs) * 1000).

Step 17: Plot ECG Signal and detected R peak

Flowchart


START
Define parameters (Serial port, Sampling frequency, Total duration)
Read ECG Signal from Serial Input
Apply Bandpass filter(0.5-35Hz)
Apply Moving Average filter
Centre the ECG Signal
R-Peak Detection
RR Interval Calculation
Display ECG Signal and detected peaks
STOP

4.3.1.3 Algorithm and flowchart for HRV Analysis

Algorithm

Step 1: Compute the mean RR interval by averaging all RR intervals.

Step 2: Calculate the mean heart rate by dividing 60000 by the mean RR interval.

Step 3: Determine the minimum heart rate using the longest RR interval and the maximum heart rate using the shortest RR interval.

Step 4: Compute SDNN by calculating the standard deviation of RR intervals.

Step 5: Compute RMSSD by taking the square root of the mean of squared successive RR interval differences.

Step 6: Count NN50 as the number of successive RR interval differences greater than 50 s.

Step 7: Compute pNN50 by dividing NN50 by the total number of RR intervals and multiplying by 100.

Step 8: Interpolate RR intervals using cubic interpolation to get evenly spaced data for frequency-domain analysis.

Step 9: Compute the power spectral density using Welch’s method.

Step 10: Extract power in the very low frequency (VLF), low frequency (LF), and high frequency (HF) bands by integrating the power spectral density over the respective frequency ranges.

Step 11: Calculate the LF/HF ratio by dividing LF power by HF power.

Step 12: Extract successive RR interval pairs for Poincaré plot analysis.

Step 13: Compute SD1 by finding the standard deviation of successive RR interval differences and scaling it appropriately.

Step 14: Compute SD2 by calculating the variance of RR intervals, adjusting for SD1, and taking the square root.

Step 15: Compute the SD2/SD1 ratio by dividing SD2 by SD1.

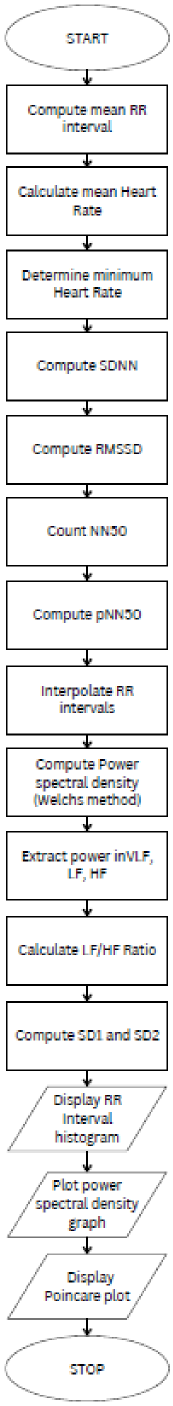
Step 16: Plot the RR interval histogram, showing the distribution of RR intervals.

Step 17: Plot the power spectral density graph, highlighting VLF, LF, and HF regions.

Step 18: Generate the Poincaré plot by plotting RR intervals against their successive values.

Step 19: Overlay SD1 and SD2 ellipses on the Poincaré plot to visualize short-term and long-term HRV variations.

Flowchart


START
Compute mean RR interval
Calculate mean Heart Rate
Determine minimum Heart Rate
Compute SDNN
Compute RMSSD
Count NN50
Compute pNN50
Interpolate RR intervals
Compute Power spectral density (Welchs method)
Extract power inVLF, LF, HF
Calculate LF/HF Ratio
Compute SD1 and SD2
Display RR Interval histogram
Plot power spectral density graph
Display Poincare plot
STOP

CHAPTER 5

RESULTS AND CONCLUSION

In this chapter, the results of the Electrocardiogram (ECG) and Heart Rate Variability (HRV) analysis are presented. The data was obtained using the AD8232 ECG sensor module in conjunction with an Arduino Uno microcontroller. This setup enabled the real-time acquisition of ECG signals, which were then processed for HRV analysis. The experimental setup is shown below, followed by the presentation and discussion of the obtained results.

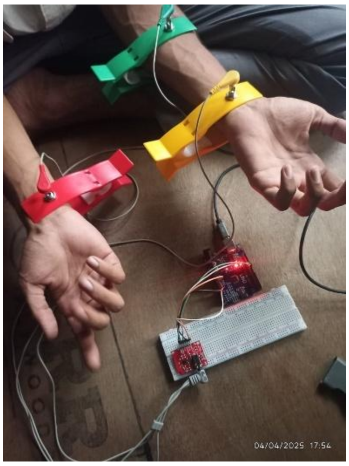


Fig 5 Experimental Setup

5.1ECG DATA ACQUISTION

The ECG signal was successfully acquired from the AD8232 module using an Arduino Uno and transmitted to a laptop. The sampling rate was set at 250 Hz, ensuring sufficient resolution for HRV analysis. Raw ECG data was collected for 300 seconds (5 minutes), providing enough beats for a meaningful HRV evaluation. The data acquisition was effective, and the collected ECG signal contained clear QRS complexes needed for HRV computation.

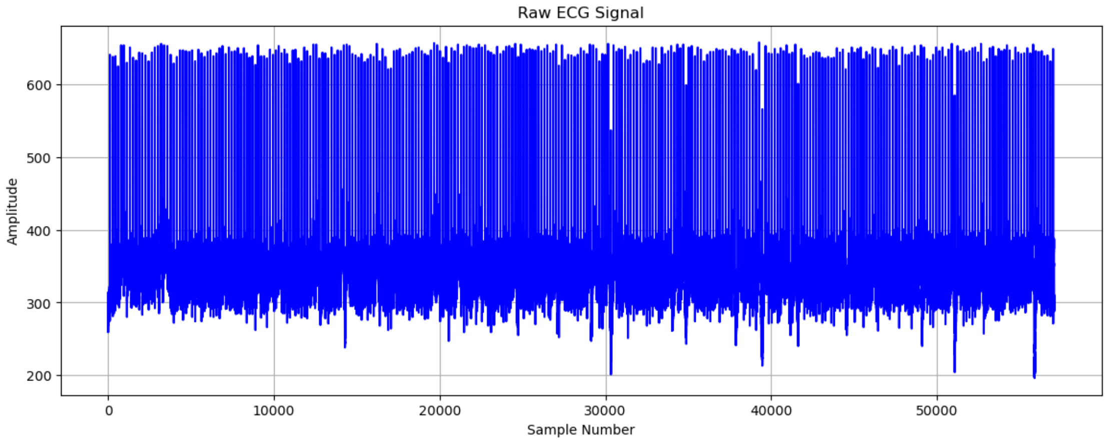


Fig 5.1 Raw ECG Signal

5.2PRE-PROCESSING OF THE SIGNAL

R-peaks were detected using a thresholding method based on 40% of the max amplitude. The RR intervals (time between R-peaks) were extracted, forming the basis for HRV analysis. R-peaks were successfully detected, and RR intervals were extracted accurately, serving as input for both time-domain and frequency-domain HRV analysis.

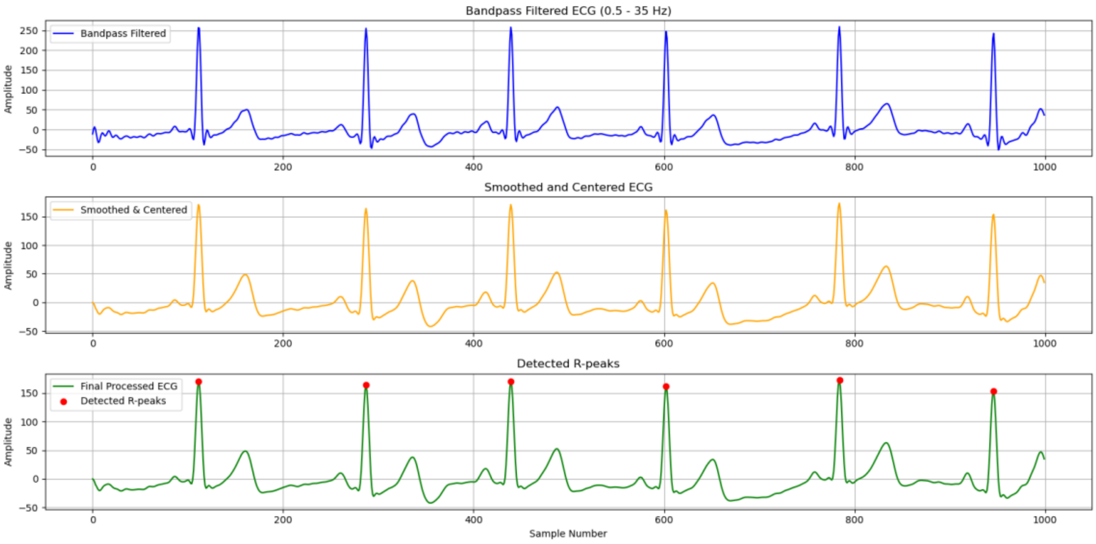


Fig 5.2.1 Pre-processed Signals

RR Intervals (ms):

[704. 608. 652. 728. 648. 740. 764. 748. 744. 732. 720. 772. 784. 772.

720. 760. 764. 712. 788. 792. 748. 752. 780. 712. 728. 752. 692. 748.

792. 740. 724. 752. 688. 740. 736. 720. 716. 768. 772. 692. 744. 724.

720. 732. 684. 656. 704. 724. 648. 712. 744. 684. 728. 748. 708. 664.
700. 700. 620. 668. 700. 640. 720. 728. 692. 684. 756. 764. 696. 736.
748. 676. 724. 808. 732. 752. 756. 712. 700. 656. 712. 764. 736. 716.
740. 720. 708. 752. 764. 680. 688. 712. 660. 676. 720. 728. 684. 776.
804. 732. 748. 772. 720. 740. 768. 700. 748. 772. 736. 700. 724. 724.
640. 676. 764. 784. 784. 680. 684. 748. 764. 716. 660. 724. 776. 760.
688. 704. 772. 780. 748. 676. 708. 724. 736. 688. 656. 696. 728. 716.
664. 696. 724. 736. 732. 696. 712. 724. 684. 684. 732. 744. 768. 764.
776. 756. 732. 752. 764. 696. 764. 780. 744. 720. 700. 676. 680. 632.
680. 632. 664. 800. 692. 640. 764. 768. 836. 780. 764. 712. 700. 764.
776. 700. 756. 760. 720. 760. 736. 692. 724. 700. 640. 712. 756. 680.
768. 760. 776. 724. 740. 776. 732. 736. 756. 756. 712. 712. 740. 732.
688. 720. 728. 748. 784. 736. 772. 784. 756. 720. 740. 744. 680. 724.
704. 656. 716. 748. 676. 732. 736. 700. 740. 732. 704. 740. 736. 700.
720. 724. 656. 712. 748. 704. 684. 736. 740. 684. 716. 732. 684. 716.
736. 720. 672. 660. 688. 704. 700. 720. 776. 784. 724. 748. 768. 788.
716. 736. 768. 748. 728. 752. 752. 688. 740. 780. 752. 712. 764. 768.
684. 712. 732. 704. 684. 740. 720. 664. 708. 716. 652. 728. 760. 736.
676. 704. 684. 704. 612. 660. 708. 668. 676. 720. 732. 748. 728. 688.
668. 692. 696. 664. 720. 724. 656.]

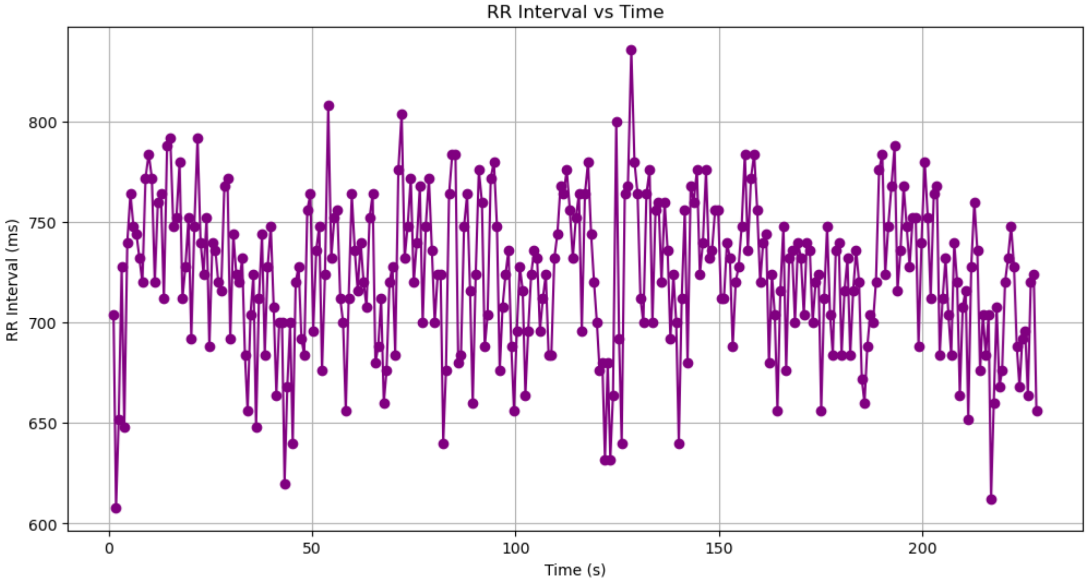


Fig 5.2.2 Plot of RR Interval vs Time

5.3TIME-DOMAIN HRV METRICS

Metric	**Value**	**Interpretation**
Mean RR Interval	722.60 ms	Normal resting range: **800-1000 ms**
Mean HR	83.03 bpm	Typical resting HR: **60-80 bpm**
SDNN	38.43 ms	Measures overall HRV, normal: **50-100 ms**
RMSSD	43.79 ms	Reflects parasympathetic activity, higher is better
PNN50	27.62 %	Indicates vagal tone, normal: **>15%**

Table 5.3 Time-domain HRV Metrics

<u>Time-Domain HRV Results:</u>

Mean RR (ms): 722.60

Mean HR (bpm): 83.03

Min HR (bpm): 71.77

Max HR (bpm): 98.68

SDNN (ms): 38.43

RMSSD (ms): 43.79

NN50 (beats): 87

pNN50 (%): 27.62

Conclusions:

The results indicate a mean HR of 83.03 bpm, slightly above the normal resting HR range (60-80 bpm). The SDNN value of 38.43 ms is below the expected range of 50-100 ms, suggesting lower HRV. However, the pNN50 value of 27.62% indicates good vagal tone, which is a positive marker for autonomic nervous system balance.

5.4FREQUENCY- DOMAIN HRV METRICS

Band	**Power (ms^2)**	**Interpolation**
LF (0.04-0.15 Hz)	101.72	Reflects both **sympathetic and parasympathetic control**
HF (0.15-0.4 Hz)	338.35	Represents **parasympathetic (vagal) activity**
LF/HF Ratio	0.30	Balance between **sympathetic and parasympathetic**

Table 5.4 Frequency- Domain HRV Metrics

Frequency-Domain HRV Results:

LF Power: 101.72 ms^2

HF Power: 338.35 ms^2

LF/HF Ratio: 0.30

Conclusions:

The low LF/HF ratio (0.30) suggests a dominance of parasympathetic activity, which is generally associated with relaxation and reduced stress. HF power is significantly higher than LF, indicating strong vagal tone and good autonomic control. Overall, these results suggest a well-regulated autonomic nervous system with a dominance of parasympathetic control, which is beneficial for cardiovascular health.

5.5NON-LINEAR HRV METRICS

Metric	Value	Interpretation
SD1	30.96 ms	Reflects **short-term** HRV (parasympathetic activity)
SD2	44.67 ms	Represents **long-term** HRV (both autonomic branches)
SD2/SD1 ratio	1.44	Higher values indicate **increased HRV adaptability**

Table 5.5 Non-linear HRV Metrics

Nonlinear HRV Results (Poincare):

SD1: 30.96 ms

SD2: 44.67 ms

SD2/SD1 Ratio: 1.44

Conclusions:

The SD1 value (30.96 ms) indicates a moderate level of short-term HRV, suggesting balanced vagal activity. The SD2 value (44.67 ms) represents overall HRV, showing the combined influence of both sympathetic and parasympathetic activity. The SD2/SD1 ratio (1.44) falls within a normal range, suggesting good autonomic adaptability and cardiovascular regulation. These results indicate a healthy autonomic nervous system with a good balance between short-term and long-term HRV regulation.

5.6PLOT OF RR INTERVAL DISTRIBUTION

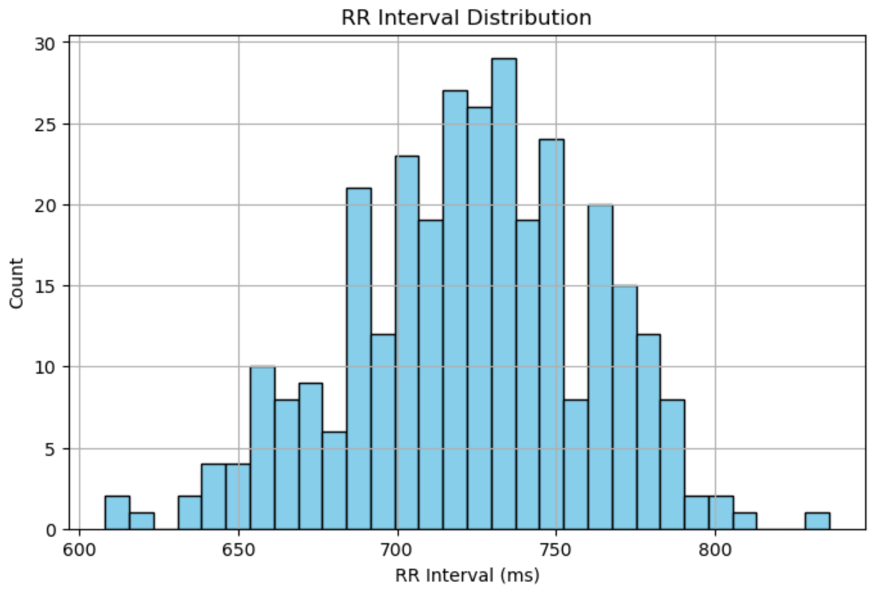


Fig 5.6 Plot of RR Interval Distribution

5.7PLOT OF RR INTERVAL SPECTRUM

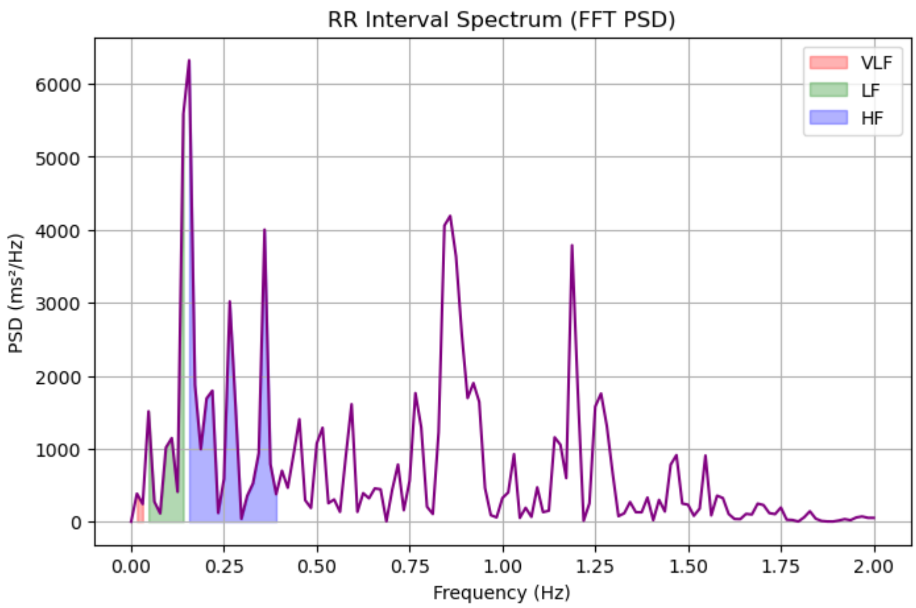


Fig 5.7 Plot of RR Interval Spectrum

5.8 PLOT OF POINCARE PLOT

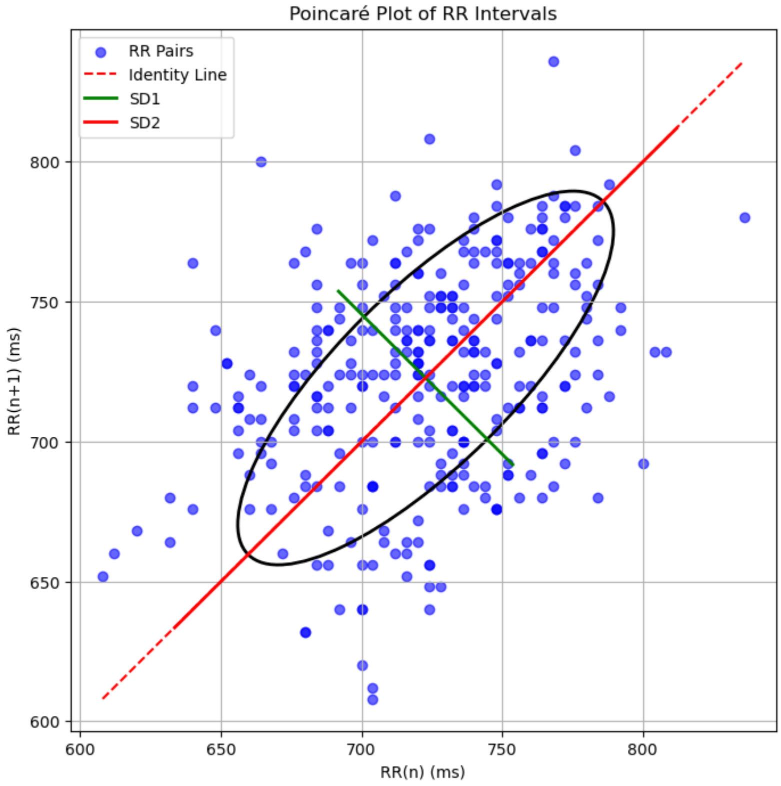


Fig 5.8 Poincare Plot

Conclusion:

PARAMETER	PERSON 1	PERSON 2	PERSON 3	PERSON 4
Time-domain HRV				
Mean RR (ms)	642.89	571.98	626.42	722.60
Mean HR (bpm)	93.33	104.90	95.78	83.03
Min HR (bpm)	75.38	43.73	49.34	71.77
Max HR (bpm)	127.12	174.42	176.47	98.68
SDNN (ms)	59.96	108.13	58.54	38.43
RMSSD (ms)	37.89	143.78	64.14	43.79
NN50 (beats)	55	54	29	87
pNN50 (%)	15.54	13.57	7.99	27.62
Frequency-domain HRV				
VLF Power (ms^2)	28.99	392.37	362.47	4.89
LF Power (ms^2)	920.01	2021.39	785.02	101.72
HF Power (ms^2)	1059.74	3655.68	443.11	338.35
LF/HF Ratio	0.87	0.55	1.77	0.30
Non- linear HRV				
SD1 (ms)	26.79	101.67	45.36	30.96
SD2 (ms)	80.46	114.22	69.26	44.67
SD2/SD1 Ratio	3.00	1.12	1.53	1.44

Table 5.9

The results presented were obtained from four different individuals, using a custom-designed device developed for real-time ECG acquisition and heart rate variability (HRV) analysis. The system effectively captured ECG signals and computed HRV metrics across time-domain, frequency-domain, and nonlinear (Poincaré) methods. The collected values were compared to observe inter-individual differences in heart rate behaviour and autonomic function. The device, being low-cost and non-invasive, proves to be a practical and efficient solution for personal or clinical HRV monitoring, demonstrating its potential for broader healthcare applications and future developments.

CHAPTER 6

FUTURE SCOPE

This project successfully developed an efficient system for real-time acquisition of ECG signals over a duration of five minutes and performed comprehensive heart rate variability (HRV) analysis across time-domain, frequency-domain, and nonlinear domains. In the future, this work can be extended by developing a compact, handheld unit capable of collecting ECG data from a larger number of patients in various environments, including clinical, remote, or home settings. The portable system could be integrated with wireless communication and cloud storage to allow continuous monitoring and remote access to patient data. Moreover, the collected HRV features can be used to train machine learning models for automatic classification of cardiovascular diseases such as arrhythmia, atrial fibrillation, or stress-related conditions. By combining real-time data acquisition, advanced signal processing, and intelligent diagnostics, this project has the potential to evolve into a powerful tool for preventive healthcare and early disease detection.

REFERENCES

[1] Tarvainen, Mika P., Stefanos D. Georgiadis, Perttu O. Ranta-Aho, and Pasi A. Karjalainen. "Time-varying analysis of heart rate variability signals with a Kalman smoother algorithm." *Physiological measurement* 27, no. 3 (2006): 225.

[2] Tarvainen, Mika P., J -P. Niskanen, J. A. Lipponen, P. O. Ranta-Aho, and P. A. Karjalainen. "Kubios HRV—a software for advanced heart rate variability analysis." In *4th European Conference of the International Federation for Medical and Biological Engineering: ECIFMBE 2008 23–27 November 2008 Antwerp, Belgium*, pp. 1022-1025. Springer Berlin Heidelberg, 2009.

[3] Abhishekh, Hulegar A., Palgun Nisarga, Ravikiran Kisan, Adoor Meghana, Sajish Chandran, Trichur Raju, and Talakad N. Sathyaprabha. "Influence of age and gender on autonomic regulation of heart." *Journal of clinical monitoring and computing* 27 (2013): 259-264.

[4] van Ravenswaaij-Arts, Conny MA, Louis AA Kollee, Jeroen CW Hopman, Gerard BA Stoelinga, and Herman P. van Geijn. "Heart rate variability." *Annals of internal medicine* 118, no. 6 (1993): 436-447.

APPENDIX I

Arduino Code for ECG Acquisition

```
void setup () {
  Serial.begin(9600);
}

void loop () {
  int ecg_value = analogRead(A0);
  Serial.println(ecg_value);
  delay(4);  // Sampling at 250 Hz
}
```

Python Code for ECG Acquisition, Pre-processing, R peak detection and HRV Analysis

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import butter, filtfilt, lfilter, find_peaks
import serial
import time

# --------------------------- Serial Data Acquisition --------------------------- #
# Serial Connection to Arduino
SERIAL_PORT = "COM4"  # Change this
BAUD_RATE = 9600
DURATION = 300 # seconds
fs = 250  # Hz

try:
    ser = serial.Serial(SERIAL_PORT, BAUD_RATE, timeout=1)
    print(" Connected to Arduino!")
except Exception as e:
    print(f" Error connecting to Arduino: {e}")
    exit()
```

```
#  Data Collection
total_samples = fs * DURATION
ecg_data = []

print(" Collecting ECG data...")
start_time = time.time()
while len(ecg_data) < total_samples:
    try:
        line = ser.readline().decode('utf-8').strip()
        if line.isdigit():
            ecg_data.append(int(line))
    except Exception as e:
        print(f" Serial Read Error: {e}")
    if time.time() - start_time > DURATION:
        print("Timeout reached.")
        break

ser.close()
print(" Data collection complete!")
ecg_data = np.array(ecg_data)

# --------------------------- Preprocessing --------------------------- #
#  Bandpass Filter Function
def butter_bandpass(lowcut, highcut, fs, order=4):
    nyquist = 0.5 * fs
    low = lowcut / nyquist
    high = highcut / nyquist
    b, a = butter(order, [low, high], btype='band')
    return b, a

def apply_bandpass_filter(data, lowcut=0.5, highcut=35.0, fs=250, order=4):
    if len(data) == 0:
        raise ValueError("ECG data is empty! Check serial communication.")

    b, a = butter_bandpass(lowcut, highcut, fs, order=order)

    if len(data) < max(len(a), len(b)) * 3:
        print(" Data too short for filtfilt, using lfilter instead.")
        return lfilter(b, a, data)  # Use lfilter for short signals

    return filtfilt(b, a, data)


# Moving Average Smoothing
def moving_average(data, window_size=7):
    return np.convolve(data, np.ones(window_size) / window_size, mode='same')

# Detect R-Peaks on smoothed signal
```

```
def detect_r_peaks_direct(ecg_signal, fs=250, height_ratio=0.4, distance_sec=0.3):
    threshold = height_ratio * np.max(ecg_signal)
    min_distance = int(distance_sec * fs)
    peaks, _ = find_peaks(ecg_signal, height=threshold, distance=min_distance)
    return peaks

# --------------------------- Processing Pipeline --------------------------- #
#  Step 1: Bandpass Filtering
ecg_filtered = apply_bandpass_filter(ecg_data, lowcut=0.5, highcut=35.0, fs=fs,
order=4)

# Step 2: Moving Average Smoothing
ecg_smoothed = moving_average(ecg_filtered, window_size=7)

#  Step 3: Centering
ecg_centered = ecg_smoothed - np.mean(ecg_smoothed)

#  Step 4: R-Peak Detection
r_peaks = detect_r_peaks_direct(ecg_centered, fs=fs, height_ratio=0.4,
distance_sec=0.3)  # You can tune these values

# --------------------------- Plotting --------------------------- #
#  Plot raw ECG
plt.figure(figsize=(14, 5))
plt.plot(ecg_data, color='blue')
plt.title('Raw ECG Signal')
plt.xlabel('Sample Number')
plt.ylabel('Amplitude')
plt.grid(True)
plt.show()

#  Optional: Show different stages for understanding
plt.figure(figsize=(16, 8))

plt.subplot(3, 1, 1)
plt.plot(ecg_filtered[:1000], color='blue', label='Bandpass Filtered')
plt.title("Bandpass Filtered ECG (0.5 - 35 Hz)")
plt.ylabel("Amplitude")
plt.legend()
plt.grid(True)

plt.subplot(3, 1, 2)
plt.plot(ecg_centered[:1000], color='orange', label='Smoothed & Centered')
plt.title("Smoothed and Centered ECG")
plt.ylabel("Amplitude")
plt.legend()
plt.grid(True)
```

```
plt.subplot(3, 1, 3)
plt.plot(ecg_centered[:1000], color='green', label='Final Processed ECG')
plt.scatter(r_peaks[r_peaks < 1000], ecg_centered[r_peaks[r_peaks < 1000]],
color='red', label='Detected R-peaks', zorder=5)
plt.title("Detected R-peaks")
plt.xlabel("Sample Number")
plt.ylabel("Amplitude")
plt.legend()
plt.grid(True)

plt.tight_layout()
plt.show()

# --------------------------- RR Interval Calculation & Plotting ------------------------- #
#  Calculate RR intervals in samples
rr_intervals_samples = np.diff(r_peaks)  # Difference between successive R-peaks

# Convert to milliseconds
rr_intervals_ms = (rr_intervals_samples / fs) * 1000  # fs = 250 Hz

print ("\n RR Intervals (ms):")
print(rr_intervals_ms)

# Time axis for RR intervals (use time of second R-peak onward)
rr_times = r_peaks[1:] / fs  # Convert to seconds for plotting

# RR Interval Plot
plt.figure(figsize=(12, 6))
plt.plot(rr_times, rr_intervals_ms, marker='o', linestyle='-', color='purple')
plt.title("RR Interval vs Time")
plt.xlabel("Time (s)")
plt.ylabel("RR Interval (ms)")
plt.grid(True)
plt.show()

#-------------------------------TIME-DOMAIN HRV METRICS-----------------------#

# Mean RR Interval (ms)
mean_rr = np.mean(rr_intervals_ms)

# Mean HR (bpm) = 60000 / Mean RR
mean_hr = 60000 / mean_rr

# Min and Max HR (bpm)
min_hr = 60000 / np.max(rr_intervals_ms)
max_hr = 60000 / np.min(rr_intervals_ms)
```

```
# SDNN (Standard deviation of RR intervals)
sdnn = np.std(rr_intervals_ms)

# RMSSD (Root Mean Square of Successive Differences)
rmssd = np.sqrt(np.mean(np.diff(rr_intervals_ms)**2))

# NN50 (number of successive RR intervals differing by more than 50ms)
nn50 = np.sum(np.abs(np.diff(rr_intervals_ms)) > 50)

# pNN50 (percentage of NN50)
pnn50 = (nn50 / len(rr_intervals_ms)) * 100

# Print Time-domain results
print("\n Time-Domain HRV Results:")
print(f"Mean RR (ms): {mean_rr:.2f}")
print(f"Mean HR (bpm): {mean_hr:.2f}")
print(f"Min HR (bpm): {min_hr:.2f}")
print(f"Max HR (bpm): {max_hr:.2f}")
print(f"SDNN (ms): {sdnn:.2f}")
print(f"RMSSD (ms): {rmssd:.2f}")
print(f"NN50 (beats): {nn50}")
print(f"pNN50 (%): {pnn50:.2f}")
```

#--FREQUENCY-DOMAIN HRV VIA FFT(POWER SPECTRAL DENSITY--#

```
from scipy.signal import welch

# Interpolate RR intervals to get evenly sampled signal for FFT
from scipy.interpolate import interp1d

# Interpolation to get continuous RR signal
times = np.cumsum(rr_intervals_ms) / 1000  # convert to seconds
interp_func = interp1d(times, rr_intervals_ms, kind='cubic')
time_interp = np.linspace(times[0], times[-1], len(times))
rr_interp = interp_func(time_interp)

# Compute Power Spectrum using Welch's method
frequencies, psd = welch(rr_interp, fs=4.0, nperseg=256)  # 4Hz interpolation

# Frequency bands (Hz)
vlf_band = (0.003, 0.04)
lf_band = (0.04, 0.15)
hf_band = (0.15, 0.40)

# Band powers
vlf_power = np.trapz(psd[(frequencies >= vlf_band[0]) & (frequencies <
vlf_band[1])],
```

```
                  frequencies[(frequencies >= vlf_band[0]) & (frequencies <
vlf_band[1])])
lf_power = np.trapz(psd[(frequencies >= lf_band[0]) & (frequencies < lf_band[1])],
                  frequencies[(frequencies >= lf_band[0]) & (frequencies < lf_band[1])])
hf_power = np.trapz(psd[(frequencies >= hf_band[0]) & (frequencies < hf_band[1])],
                  frequencies[(frequencies >= hf_band[0]) & (frequencies < hf_band[1])])

# LF/HF Ratio
lf_hf_ratio = lf_power / hf_power

# Print Frequency-domain results
print("\n Frequency-Domain HRV Results:")
print(f"VLF Power: {vlf_power:.2f} ms²")
print(f"LF Power: {lf_power:.2f} ms²")
print(f"HF Power: {hf_power:.2f} ms²")
print(f"LF/HF Ratio: {lf_hf_ratio:.2f}")

#---------------NON-LINEAR HRV------------------#

# Poincare Plot Metrics
rr_diff = np.diff(rr_intervals_ms)
sd1 = np.sqrt(np.var(rr_diff) / 2)
sd2 = np.sqrt(2 * np.var(rr_intervals_ms) - (np.var(rr_diff) / 2))
sd_ratio = sd2 / sd1

print("\n Nonlinear HRV Results (Poincare):")
print(f"SD1: {sd1:.2f} ms")
print(f"SD2: {sd2:.2f} ms")
print(f"SD2/SD1 Ratio: {sd_ratio:.2f}")

#---------------------------------------------plots-------------------------------#
# RR Interval Histogram
plt.figure(figsize=(8, 5))
plt.hist(rr_intervals_ms, bins=30, color='skyblue', edgecolor='black')
plt.title("RR Interval Distribution")
plt.xlabel("RR Interval (ms)")
plt.ylabel("Count")
plt.grid(True)
plt.show()

# RR Spectrum (FFT-based)
plt.figure(figsize=(8, 5))
plt.plot(frequencies, psd, color='purple')
plt.fill_between(frequencies, psd, where=((frequencies >= vlf_band[0]) &
(frequencies < vlf_band[1])), color='red', alpha=0.3, label='VLF')
plt.fill_between(frequencies, psd, where=((frequencies >= lf_band[0]) & (frequencies
< lf_band[1])), color='green', alpha=0.3, label='LF')
```

```
plt.fill_between(frequencies, psd, where=((frequencies >= hf_band[0]) & (frequencies
< hf_band[1])), color='blue', alpha=0.3, label='HF')
plt.title("RR Interval Spectrum (FFT PSD)")
plt.xlabel("Frequency (Hz)")
plt.ylabel("PSD (ms²/Hz)")
plt.legend()
plt.grid(True)
plt.show()

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patches

# Extract RR interval pairs for Poincare Plot
rr_n = rr_intervals_ms[:-1]   # All RR intervals except the last
rr_n1 = rr_intervals_ms[1:]   # All RR intervals except the first

# Calculate SD1 & SD2
sd1 = np.sqrt(0.5 * np.var(rr_n1 - rr_n))
sd2 = np.sqrt(2 * np.var(rr_n1 + rr_n))

# Scatter plot for Poincare
plt.figure(figsize=(8, 8))
plt.scatter(rr_n, rr_n1, alpha=0.6, color='blue', label='RR Pairs')

# Mean of RR for centering ellipse
mean_rr_n = np.mean(rr_n)
mean_rr_n1 = np.mean(rr_n1)

# Plot identity line
plt.plot([min(rr_n), max(rr_n)], [min(rr_n), max(rr_n)], 'r--', label='Identity Line')

# Add SD1 and SD2 lines
plt.plot([mean_rr_n - sd1, mean_rr_n + sd1], [mean_rr_n1 + sd1, mean_rr_n1 - sd1],
'g-', linewidth=2, label="SD1")
plt.plot([mean_rr_n - sd2, mean_rr_n + sd2], [mean_rr_n1 - sd2, mean_rr_n1 + sd2],
'r-', linewidth=2, label="SD2")

# Draw the ellipse representing SD1 and SD2
ellipse = patches.Ellipse((mean_rr_n, mean_rr_n1), width=2*sd2, height=2*sd1,
angle=45,
                          edgecolor='black', facecolor='none', linestyle='-', linewidth=2)
plt.gca().add_patch(ellipse)

# Labels and legend
plt.xlabel("RR(n) (ms)")
plt.ylabel("RR(n+1) (ms)")
```

```
plt.title("Poincaré Plot of RR Intervals")
plt.legend()
plt.grid()
plt.show()

# Print SD1 and SD2 values
print(f"SD1: {sd1:.2f} ms")
print(f"SD2: {sd2:.2f} ms")
print(f"SD2/SD1 Ratio: {sd2/sd1:.3f}")
```

APPENDIX II

AD8232 Data Sheet

SPECIFICATIONS

V_S = 3 V, V_{REF} = 1.5 V, V_{CM} = 1.5 V, T_A = 25°C, operating temperature (T_{OPR}) = −40°C to +105°C for the W grade, FR = low, SDN = high, and AC/$\overline{DC}$ = low, unless otherwise noted.

Table 1.

Parameter	Symbol	Test Conditions/ Comments	A Grade Min	A Grade Typ	A Grade Max	W Grade Min	W Grade Typ	W Grade Max	Unit
INSTRUMENTATION AMPLIFIER									
Common-Mode Rejection Ratio, DC to 60 Hz	CMRR	V_{CM} = 0.35 V to 2.85 V, V_{DIFF} = 0 V	80	86					dB
		T_{OPR}				80			dB
		V_{CM} = 0.35 V to 2.85 V, V_{DIFF} = ±0.3 V		80			80		dB
Power Supply Rejection Ratio	PSRR	V_S = 2.0 V to 3.5 V	76	90					dB
		T_{OPR}				76	90		dB
Offset Voltage (RTI)	V_{OS}								
Instrumentation Amplifier Inputs				3	8				mV
		T_{OPR}					3	8	mV
DC Blocking Input[1]				5	50		5	50	μV
Average Offset Drift									
Instrumentation Amplifier Inputs				10					μV/°C
		T_{OPR}					10		μV/°C
DC Blocking Input[1]				0.05			0.05		μV/°C
Input Bias Current	I_B			50	200		50	200	pA
		T_A = 0°C to 70°C		1					nA
		T_{OPR}						3.75	nA
Input Offset Current	I_{OS}			25	100		25	100	pA
		T_A = 0°C to 70°C		1					nA
		T_{OPR}						1	nA
Input Impedance									
Differential				10‖7.5			10‖7.5		GΩ‖pF
Common Mode				5‖15			5‖15		GΩ‖pF
Input Voltage Noise (RTI)									
Spectral Noise Density		f = 1 kHz		100			100		nV/√Hz
Peak-to-Peak Voltage Noise		f = 0.1 Hz to 10 Hz		12			12		μV p-p
		f = 0.5 Hz to 40 Hz		14			14		μV p-p
Input Voltage Range		T_A = 0°C to 70°C	0.2		$+V_S$				V
		T_{OPR}				0.2		$+V_S$	V
DC Differential Input Range	V_{DIFF}		−300		+300				mV
		T_{OPR}				−300		+300	mV
Output									
Output Swing		R_L = 50 kΩ	0.1		$+V_S$ − 0.1				V
		T_{OPR}				0.1		$+V_S$ − 0.1	V
Short-Circuit Current	I_{OUT}			6.3			6.3		mA
Gain	A_V			100			100		V/V
Gain Error		V_{DIFF} = 0 V		0.4			0.4		%
		V_{DIFF} = −300 mV to +300 mV		1	3.5			3.5	%
		T_{OPR}						7.9	%
Average Gain Drift		T_A = 0°C to 70°C		12					ppm/°C
		T_{OPR}					12		ppm/°C
Bandwidth	BW			2			2		kHz
RFI Filter Cutoff (Each Input)				1			1		MHz

Rev. D \| Page 4 of 32

Parameter	Symbol	Test Conditions/ Comments	A Grade Min	A Grade Typ	A Grade Max	W Grade Min	W Grade Typ	W Grade Max	Unit
OPERATIONAL AMPLIFIER (A1)									
Offset Voltage	V_{OS}			1	5				mV
		T_{OPR}					1	5	mV
Average TC		T_A = 0°C to 70°C		5					µV/°C
		T_{OPR}					5		µV/°C
Input Bias Current	I_B			100			100		pA
		T_A = 0°C to 70°C		1					nA
		T_{OPR}					2.5		nA
Input Offset Current	I_{OS}			100			100		pA
		T_A = 0°C to 70°C		1					nA
		T_{OPR}					1		nA
Input Voltage Range			0.1		$+V_S - 0.1$	0.1		$+V_S - 0.1$	V
Common-Mode Rejection Ratio	CMRR	V_{CM} = 0.5 V to 2.5 V		100			100		dB
Power Supply Rejection Ratio	PSRR			100			100		dB
Large Signal Voltage Gain	A_{VO}			110			110		dB
Output Voltage Range		R_L = 50 kΩ	0.1		$+V_S - 0.1$				V
		T_{OPR}				0.1		$+V_S - 0.1$	V
Short-Circuit Current Limit	I_{OUT}			12			12		mA
Gain Bandwidth Product	GBP			100			100		kHz
Slew Rate	SR			0.02			0.02		V/µs
Voltage Noise Density (RTI)	e_n	f = 1 kHz		60			60		nV/√Hz
Peak-to-Peak Voltage Noise (RTI)	$e_{n\,p\text{-}p}$	f = 0.1 Hz to 10 Hz		6			6		µV p-p
		f = 0.5 Hz to 40 Hz		8			8		µV p-p
RIGHT LEG DRIVE AMPLIFIER (A2)									
Output Swing		R_L = 50 kΩ	0.1		$+V_S - 0.1$	0.1		$+V_S - 0.1$	V
Short-Circuit Current	I_{OUT}			11			11		mA
Integrator Input Resistor			120	150	180	120	150	180	kΩ
Gain Bandwidth Product	GDP			100			100		kHz
REFERENCE BUFFER (A3)									
Offset Error	V_{OS}	R_L > 50 kΩ		1			1		mV
Input Bias Current	I_B			100			100		pA
Short-Circuit Current Limit	I_{OUT}			12			12		mA
Voltage Range		R_L = 50 kΩ	0.1		$+V_S - 0.7$				V
		T_{OPR}				0.1		$+V_S - 0.7$	V
DC LEADS OFF COMPARATORS									
Threshold Voltage				$+V_S -$ 0.5			$+V_S -$ 0.5		V
Hysteresis				60			60		mV
Propagation Delay				0.5			0.5		µs
AC LEADS OFF DETECTOR									
Square Wave Frequency	f_{AC}		50	100	175	50	100	175	kHz
Square Wave Amplitude	I_{AC}			200			200		nA p-p
Impedance Threshold		Between +IN and −IN	10	20		10	20		MΩ
Detection Delay				110			110		µs
FAST RESTORE CIRCUIT									
Switches		S1 and S2							
On Resistance	R_{ON}		8	10	12	8	10	12	kΩ
Off Leakage				100			100		pA
Window Comparator									
Threshold Voltage		From either rail		50			50		mV
Propagation Delay				2			2		µs
Switch Timing Characteristics									
Feedback Recovery Switch On Time	t_{SW1}			110			110		ms
Filter Recovery Switch On Time	t_{SW2}			55			55		ms
Fast Restore Reset	t_{RST}			2			2		µs

Rev. D | Page 5 of 32

Parameter	Symbol	Test Conditions/ Comments	A Grade			W Grade			Unit
			Min	Typ	Max	Min	Typ	Max	
LOGIC INTERFACE									
Input Characteristics									
Input Voltage (AC/$\overline{DC}$ and FR)									
Low	V_{IL}			1.24			1.24		V
High	V_{IH}			1.35			1.35		V
Input Voltage ($\overline{SDN}$)									
Low	V_{IL}			2.1			2.1		V
High	V_{IH}			0.5			0.5		V
Output Characteristics		LOD+ and LOD− terminals							
Output Voltage									
Low	V_{OL}			0.05			0.05		V
High	V_{OH}			2.95			2.95		V
SYSTEM SPECIFICATIONS									
Quiescent Supply Current				170	230		170	230	µA
		T_A = 0°C to 70°C		210					µA
		T_{OPR}						270	µA
Shutdown Current				40	500		40	500	nA
		T_A = 0°C to 70°C		100					nA
		T_{OPR}						612	nA
Supply Range			2.0		3.5	2.0		3.5	V
Specified Temperature Range			0		+70	−40		+105	°C
Operational Temperature Range			−40		+85	−40		+105	°C

[1] Offset referred to the input of the instrumentation amplifier inputs. See the Input Referred Offsets section for additional information.

ABSOLUTE MAXIMUM RATINGS

Table 2.

Parameter	Rating
Supply Voltage	3.6 V
Output Short-Circuit Current Duration	Indefinite
Maximum Voltage, Any Terminal[1]	$+V_S + 0.3$ V
Minimum Voltage, Any Terminal[1]	−0.3 V
Storage Temperature Range	−65°C to +125°C
Operating Temperature Range	
AD8232ACPZ	−40°C to +85°C
AD8232WACSZ	−40°C to +105°C
Maximum Junction Temperature	140°C
θ_{JA} Thermal Impedance[2]	48°C/W
θ_{JC} Thermal Impedance	4.4°C/W
ESD Rating	
Human Body Model (HBM)	8 kV
Charged Device Model (FICDM)	1.25 kV
Machine Model (MM)	200 V

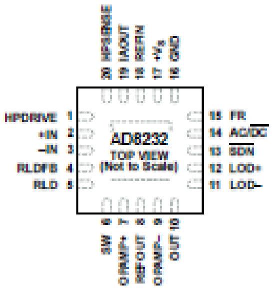
[1] This level or the maximum specified supply voltage, whichever is the lesser, indicates the superior voltage limit for any terminal. If input voltages beyond the specified minimum or maximum voltages are expected, place resistors in series with the inputs to limit the current to less than 5 mA.
[2] θ_{JA} is specified for a device in free air on a 4-layer JEDEC board.

Stresses at or above those listed under Absolute Maximum Ratings may cause permanent damage to the product. This is a stress rating only; functional operation of the product at these or any other conditions above those indicated in the operational section of this specification is not implied. Operation beyond the maximum operating conditions for extended periods may affect product reliability.

ESD CAUTION

ESD (electrostatic discharge) sensitive device. Charged devices and circuit boards can discharge without detection. Although this product features patented or proprietary protection circuitry, damage may occur on devices subjected to high energy ESD. Therefore, proper ESD precautions should be taken to avoid performance degradation or loss of functionality.

PIN CONFIGURATION AND FUNCTION DESCRIPTIONS



NOTES
1. CONNECT THE EXPOSED PAD TO GND OR LEAVE UNCONNECTED.

Figure 2. Pin Configuration

Table 3. Pin Function Descriptions

Pin No.	Mnemonic	Description
1	HPDRIVE	High-Pass Driver Output. Connect HPDRIVE to the capacitor in the first high-pass filter. The AD8232 drives this pin to keep HPSENSE at the same level as the reference voltage.
2	+IN	Instrumentation Amplifier Positive Input. +IN is typically connected to the left arm (LA) electrode.
3	–IN	Instrumentation Amplifier Negative Input. –IN is typically connected to the right arm (RA) electrode.
4	RLDFB	Right Leg Drive Feedback Input. RLDFB is the feedback terminal for the right leg drive circuit.
5	RLD	Right Leg Drive Output. Connect the driven electrode (typically, right leg) to the RLD pin.
6	SW	Fast Restore Switch Terminal. Connect this terminal to the output of the second high-pass filter.
7	OPAMP+	Operational Amplifier Noninverting Input.
8	REFOUT	Reference Buffer Output. The instrumentation amplifier output is referenced to this potential. Use REFOUT as a virtual ground for any point in the circuit that needs a signal reference.
9	OPAMP–	Operational Amplifier Inverting Input.
10	OUT	Operational Amplifier Output. The fully conditioned heart rate signal is present at this output. OUT can be connected to the input of an ADC.
11	LOD–	Leads Off Comparator Output. In dc leads off detection mode, LOD– is high when the electrode to –IN is disconnected, and it is low when connected. In ac leads off detection mode, LOD– is always low.
12	LOD+	Leads Off Comparator Output. In dc leads off detection mode, LOD+ is high when the +IN electrode is disconnected, and it is low when connected. In ac leads off detection mode, LOD+ is high when either the –IN or +IN electrode is disconnected, and it is low when both electrodes are connected.
13	SDN	Shutdown Control Input. Drive SDN low to enter the low power shutdown mode.
14	AC/DC	Leads Off Mode Control Input. Drive the AC/DC pin low for dc leads off mode. Drive the AC/DC pin high for ac leads off mode.
15	FR	Fast Restore Control Input. Drive FR high to enable fast recovery mode; otherwise, drive it low.
16	GND	Power Supply Ground.
17	+V_S	Power Supply Terminal.
18	REFIN	Reference Buffer Input. Use REFIN, a high impedance input terminal, to set the level of the reference buffer.
19	IAOUT	Instrumentation Amplifier Output Terminal.
20	HPSENSE	High-Pass Sense Input for Instrumentation Amplifier. Connect HPSENSE to the junction of R and C that sets the corner frequency of the dc blocking circuit.
	EP	Exposed Pad. Connect the exposed pad to GND or leave it unconnected.

User Manual
SKU: A000066

Description

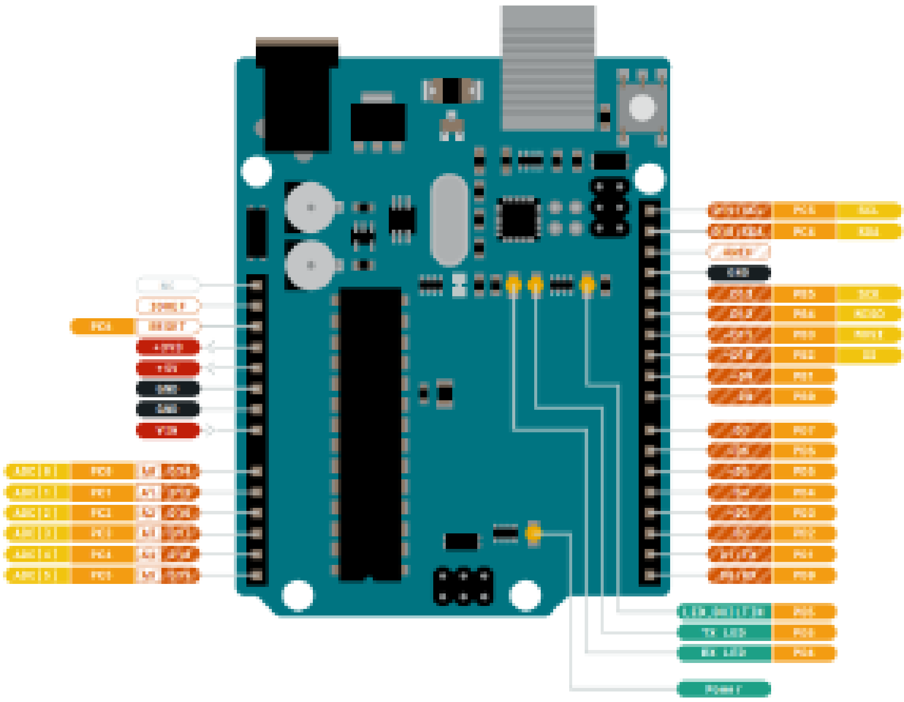
The Arduino® UNO R3 is the perfect board to get familiar with electronics and coding. This versatile development board is equipped with the well-known ATmega328P and the ATMega 16U2 Processor.

This board will give you a great first experience within the world of Arduino.

Target areas:

Maker, Introduction, Industries

5 Connector Pinouts



Pinout

5.1 JANALOG

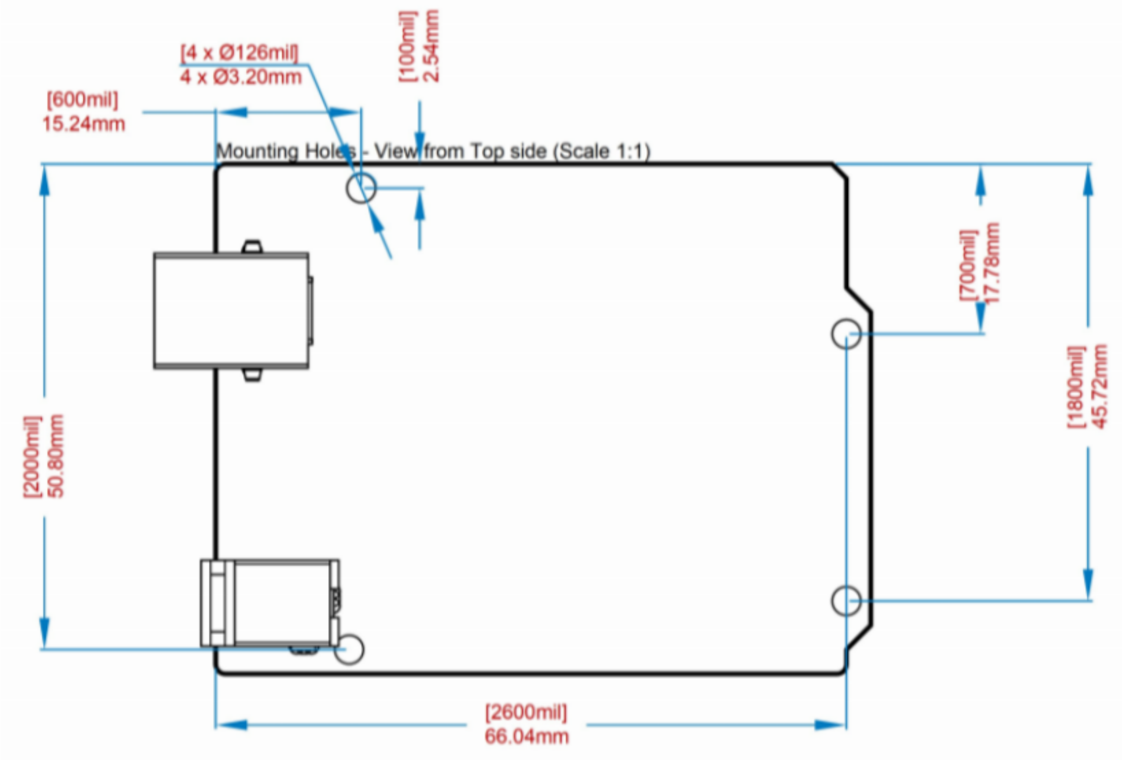
Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

5.2 JDIGITAL

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

5.3 Mechanical Information

5.4 Board Outline & Mounting Holes



Board outline