Group 1: Vladislav Korkka, Juhani Kangas, Luukas Laitinen

uPulse

Project Report

First-year Hardware Project

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Communication and Technology

10 May 2024

Abstract

This project aimed to create a heart rate monitoring device, which is easy to use and has all the features required for the highest level grade.

uPulse uses a photoplethysmography (PPG) sensor to measure heart rate and heart rate variability (HRV). The system includes a Raspberry Pi Pico for the frontend measuring and processing and a Raspberry Pi 4 for the backend operations such as saving results to history. Communication between frontend and backend is handled with Mosquitto, which is an MQTT broker.

The project involved developing algorithms for heart rate detection and HRV calculations. uPulse analyzes HRV data locally and also sends data to Kubios Cloud for detailed analysis.

Future improvements are planned to make uPulse even better. These improvements include enhancing heart rate detection accuracy and adding a durable casing, which also looks good. Despite some minor issues, the project met its goals. This project fulfilled its educational goals and also provided a foundation for future enhancements and potential real-world applications.

Version history

Ver	Description	Date	Author(s)
1.0	Added Introduction and Midway Summary	14.4.2024	Vladislav Korkka, Juhani Kangas, Luukas Laitinen
2.0	Edited the introduction, Added Parts: Theoretical background, Methods and material, Implementation, Conclusions and Final summary.	9.5.2024	Vladislav Korkka, Juhani Kangas, Luukas Laitinen
2.1	Final editing and added Abstract	10.5.2024	Vladislav Korkka, Juhani Kangas, Luukas Laitinen

Contents

1 Introduction

In the Hardware 2 course, the assignment was to develop a heart rate variability (HRV) analysis system using a Raspberry Pi Pico. heart rate variability (HRV) analysis is not merely about counting heartbeats, it involves analyzing the variations in heart rate over time. These variations provide insights into stress levels and overall cardiovascular health.

The primary challenge addressed by this project is the complexity and inaccessibility of traditional HRV monitoring devices, which are often expensive and not user-friendly for non-professionals. By developing a simple and easy to use HRV analysis device, individuals can take steps to monitor and manage their health using uPulse.

This project is a great practical demonstration of how skills learned in university can be applied to create real and actually useful technological solutions. The goal of this project was to improve our hardware and software development skills, while also gaining experience in solving real-world challenges.

This report details the process undertaken in developing this device from start to finish. It presents the encountered challenges and the practical implications of the final product.

2 Theoretical Background

Heart rate (HR) is the number of times the heart beats per minute (BPM). Heart rate variability (HRV) is the variation in the time intervals between heartbeats, measured in milliseconds (ms). HRV shows how the sympathetic and parasympathetic nervous systems interact, which gives valuable information about how the body regulates itself and how healthy the heart is. Most common HRV metrics are the root mean square of successive differences (RMSSD) and the standard deviation of NN intervals (SDNN). Higher HRV typically indicates better cardiovascular health and resilience to stress, lower HRV is usually associated with stress, fatigue, and potential health issues. (1), (2)

The normal range for heart rate and HRV values varies among individuals due to factors such as age, physical condition and genetics. The normal resting heart rate for adults ranges from 60 to 100 BPM. RMSSD values usually range from 20 to 50 ms in healthy adults, with athletes potentially having higher values. SDNN values above 100 ms are generally indicative of good cardiovascular health, while values below 50 ms can signal potential issues.

Several physiological and environmental factors affect heart rate and HRV. Heart rate regulation is mostly controlled by the autonomic nervous system (ANS), which consists of the sympathetic and parasympathetic nervous systems. Hormones like adrenaline and cortisol, which are released during stress or physical exertion, also significantly impact heart rate and HRV. Additionally, heart rate and HRV vary throughout the day, usually being lower during sleep and higher when awake.

uPulse features the Crowtail Pulse Sensor v2.0, which uses photoplethysmography (PPG) to measure heart rate and HRV. PPG is an optical technique that detects blood volume changes in the microvascular bed of tissue. (3), (4) The PPG sensor used in the uPulse operates by emitting light into the skin using an LED. Blood absorbs more light than the surrounding tissues, so with each heartbeat, blood volume increases, absorbing more light and reflecting less back to the sensor. Between heartbeats, blood volume decreases, reflecting more light. The photodetector captures the intensity of the reflected light, producing a waveform corresponding to the cardiac cycle. This PPG signal allows the device to measure heart rate by identifying peaks in the waveform and to calculate HRV by analyzing the intervals between these peaks, known as inter-beat intervals (IBIs).

3 Methods and Material

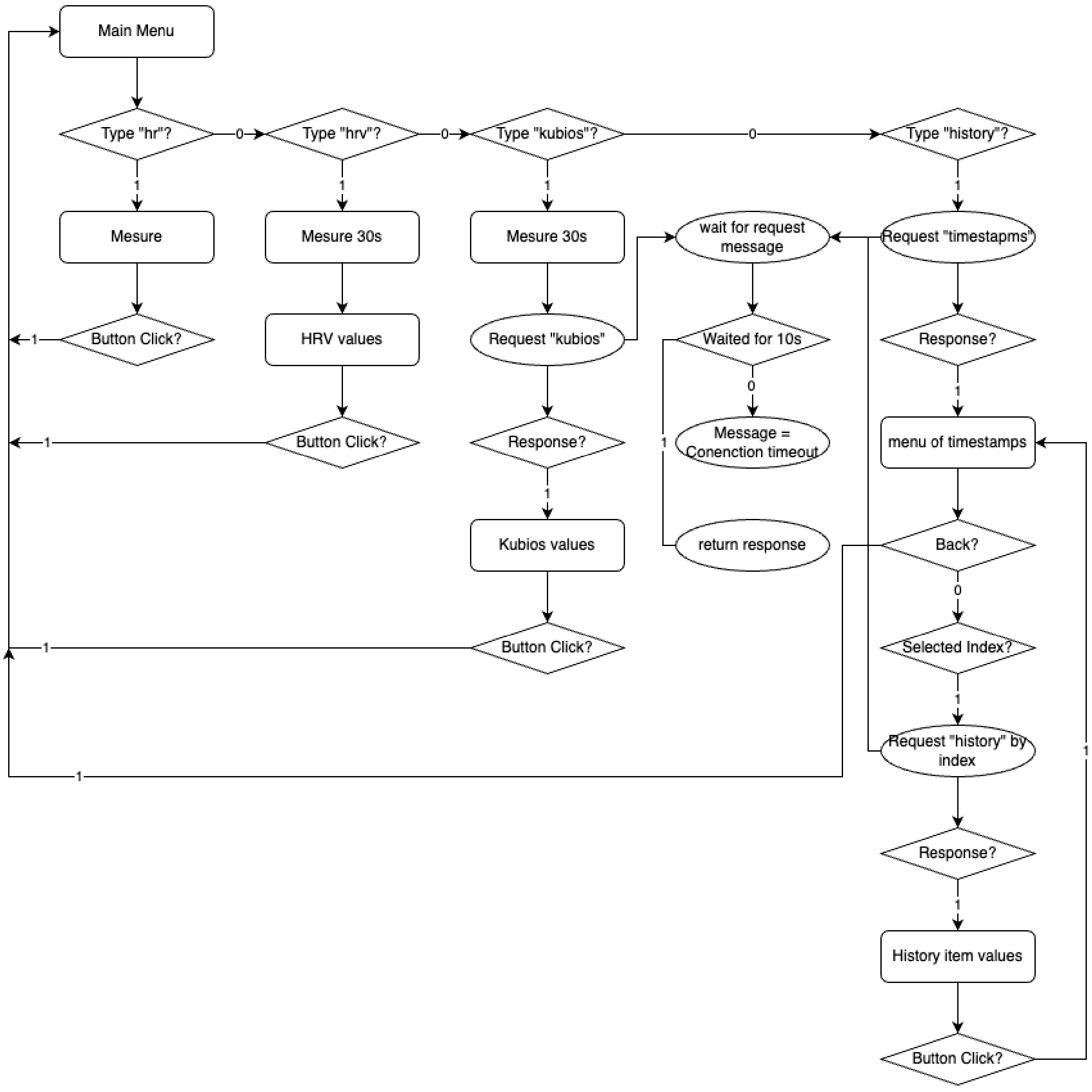
The uPulse system operates through a well-coordinated setup involving four key devices: a Raspberry Pi Pico Wireless, Raspberry Pi 4, Crowtail pulse sensor, and an Asus RT-AX53U (AX1800) Dual-band WiFi router. The frontend is managed by the Raspberry Pi Pico, which utilizes MicroPython to process inputs from the user interface. Backend operations are handled by the Raspberry Pi 4, operating on Python, which manages data for Kubios and stores it on an SD card.

The Crowtail pulse sensor, a crucial component for heart rate detection, is directly connected to the Raspberry Pi Pico, enabling immediate data acquisition. To verify the accuracy of the uPulse's heart rate detection algorithm, we conducted comparative tests with the Apple Watch Series 5. Both devices were placed on the same body part simultaneously to ensure that the heart rate readings could be accurately compared under the same physiological conditions .

The Raspberry Pi 4 is linked to an Asus RT-AX53U router, facilitating a crucial connection that enables communication between the Raspberry Pi Pico and the Raspberry Pi 4. By utilizing MQTT calls, a lightweight messaging protocol, the Raspberry Pi Pico transmits instructions and data to the Raspberry Pi 4. This configuration ensures that the system operates both synchronously and efficiently, optimizing performance and significantly enhancing the overall reliability of the uPulse system.

4 Implementation

Logical Representation of uPulse code structure.



System Overview

The system is designed to monitor heart rate and heart rate variability (HRV) through a wearable device, which uses onboard sensors to capture physiological signals and provides a user-friendly interface via an OLED display. Data is processed locally for immediate feedback or through Kubios service and transmitted securely to a backend server for more comprehensive analysis and storage.

System Architecture

1. Hardware:
    - Wearable Sensors: Capture photoplethysmogram (PPG) signals and send them to the microcontroller.
    - Microcontroller: Processes the sensor signals and drives the OLED display to provide a user interface.
    - Input Controls: Rotary encoder and buttons for menu navigation.
2. Software:
    - Main Module (main.py): The central coordination point that integrates input handling, display management, and functional module execution.
    - Data Processing (calculate.py): Provides algorithms to analyze PPG signals and compute heart rate and HRV metrics.
    - Display (functions.py): Manages the OLED display output to present data and navigation menus.
    - Network (Request.py): Manages data communication via MQTT, allowing the system to interact with external servers.
    - Backend (backend.py): Connects to external analysis services, stores data securely, and provides historical data.

Algorithms

1. Heart Rate Calculation:
    - Signal Filtering: Raw PPG signals are pre-processed to remove noise and artifacts.

- Peak Detection: Detects peaks to determine heartbeats, and calculates pulse-to-pulse intervals (PPIs).
- Heart Rate Computation: Converts the PPI into beats per minute (BPM).

2. HRV Measurement:
    - Time-Domain Analysis: Calculates metrics like SDNN (Standard Deviation of NN intervals) and RMSSD (Root Mean Square of Successive Differences).
    - Frequency-Domain Analysis: Performs spectral analysis to assess autonomic nervous system activity.

Data Collection and Processing

1. Data Collection:
    - Data was collected using onboard sensors over several sessions lasting 30 seconds each, capturing the raw PPG signals continuously.
    - The captured data was stored for later analysis in JSON format.
2. Processing:
    - Local Processing: Initial heart rate and HRV calculations were performed on-device using the calculate.py algorithms for immediate feedback.
    - Backend Analysis: The backend server provided more advanced analysis via external services (Kubios Cloud). Data was sent through secure MQTT communication, and results were retrieved and stored in a JSON file for historical review.

This architecture ensures an efficient, scalable system that provides immediate user feedback while enabling comprehensive, server-based analysis for in-depth health insights.

5 Group Work Summary

5.1 Midway Summary

Juhani Kangas:

- Single-handedly completed all project group exercises for week 1,2 and 3
- Assumed responsibility for a portion of the project report.
- Conducted extensive research on integrating MQTT with MicroPython, Raspberry Pi, and router systems.

Vladislav Korkka:

- User Interface: A menu system that allows users to navigate through various options.
- Heart Rate Monitoring Logic: This includes the programming required to calculate heart rate from the data collected via the LED sensor.
- Input Handling: Implementation of encoder button inputs to interact with the device through physical buttons.
- Visual Feedback: Creation of heart beat animations and menu selection animations displayed on a small 128x64 pixel OLED screen.
- Project Report: Writing own part of the project midway summary

Luukas Laitinen:

- Writing the project report
- Setting up MQTT broker connection
- Exploring Kubios to understand its functionalities and how it can be integrated into our HRV analysis system

In the weeks ahead, the focus will be on delving into research and implementing Kubios and MQTT, while simultaneously refining the existing code. Additionally, there is an aim to address the persistent challenge of securing a steady heartbeat signal from the fingertip. Despite best efforts to boost sensitivity in pulse detection, reliably capturing the faint pulse remains an ongoing struggle.

5.2 Final Summary

Juhani Kangas:

- Single-handedly completed all project group exercises for week 1 , 2 , 3 and 4.
- Assumed responsibility for a portion of the project report, presentation PowerPoint and project presentation.
- Conducted extensive research on integrating MQTT with MicroPython, Raspberry Pi, and router systems.
- Helped with the implementation of MQTT, Kubios and backend .

Vladislav Korkka:

- Frontend logic and implementation.
- Backend testing, and additional improvements.
- Project report: Implementation and own part of final summary.

Luukas Laitinen:

- Implemented Kubios API request, result history and MQTT connection between frontend and backend.
- Portion of the final project report and project presentation.

Work distribution table.

	Frontend	Backend	Project exercises	MQTT	Documents & presentation
Juhani	0	1	3	1	1
Vladislav	3	1	0	2	1
Luukas	0	2	0	2	1

0 = Did not participate. 1=Contributed 2=Did majority 3=Did all

Our group (Group 1) efficiently distributed tasks and coordinated well, ensuring that each member was assigned to the role where they fit best. This approach led to each member delivering excellent results in their respective responsibilities. Our only weak point was the timing of the project creation. We had the entire frontend built even before Hardware 2 course started, but we only began working on the MQTT, Kubios and backend a few days before the project presentation.

6 Conclusions

The state of uPulse is excellent, meeting all the highest level requirements (Grade 5) as outlined in the "Project Evaluation Criteria." We faced minimal issues, except for the router, which stopped working multiple times without a clear reason. Despite restarting it and verifying the settings on the router's configuration page, the problem persisted until it inexplicably resolved itself the next day.

uPulse has a few limitations because of its simplicity. It stores only up to 1200 history items, and its heart rate sensor's accuracy is around 98%, which might not suffice for extremely precise measurements. The device's lack of a casing also makes it quite brittle.

Looking ahead, we plan to enhance functionality so that analysis starts only when a steady pulse is detected and pauses if the user removes their finger, resuming once the finger is replaced on the sensor. Additionally, we aim to make a casing with a 3D printer once available, which will make uPulse easier to handle and more visually appealing.

References

1. Heart rate [Internet]. Wikipedia; [cited 2024 May 9]. Available from: https://en.wikipedia.org/wiki/Heart_rate.
2. Heart rate variability [Internet]. Wikipedia; [cited 2024 May 9]. Available from: https://en.wikipedia.org/wiki/Heart_rate_variability.
3. Photoplethysmography [Internet]. Medical News; [cited 2024 May 9]. Available from: https://www.news-medical.net/health/Photoplethysmography-(PPG).aspx
4. Crowtail - Pulse Sensor [Internet]. Electrow. 2022 Jul 29 [cited 2024 May 9]. Available from: https://www.elecrow.com/wiki/index.php?title=Crowtail-_Pulse_Sensor.