Team name: Tu Dinh, Nhut Vo, Minh Nguyen

HeartBest Pulse Oximeter

**Project Report**

First-year Hardware Project

School of ICT

Metropolia University of Applied Sciences

22 March 2023

Abstract

|  |  |
| --- | --- |
| Abstract Authors:  Title:  Number of Pages:  Date:  Degree:  Programme:  Instructors: | Tu Dinh, Nhut Vo Quang, Minh Nguyen  Heartbest Pulse Oximeter  XX pages  X May 2024  Bachelor of Engineering  Information Technology  Saana Vallius |

The project aimed to build an embedded system for a heart rate monitoring device. The system gathered heart rate information from the pulse sensor using the photoplethysmography technique (PPG), employing local calculations to analyze Heart Rate Variability (HRV) in offline mode and utilizing Kubios Cloud for analysis in online mode.

For the implementation, the system constituted a combination of hardware components, including the Raspberry Pi Pico W, Crowtail pulse sensor, and OLED screen, as well as a software architecture that included algorithms, data structures, and network functionalities. In this project, MicroPython served as the programming language, with Thonny as the integrated development environment (IDE).

The project was successfully executed despite the team’s few challenges, for example, integrating each team member's code fragments into the main program. The final system filtered heart rate signals with high accuracy, and the software operated smoothly without any interruptions.

Overall, the project had provided valuable insights into the development of interactive embedded systems, enhancing students' understanding in this area.

**Version history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ver** | **Description** | **Date** | **Author(s)** |
| 1.0 | Created structure for the project report. Added instructions for what should be included in the different parts of the document. | 13.3.2023 | Saana Vallius |
| 1.1 | Added Instructions (REMOVE WHEN READY), minor editions. | 14.3.2023 | Sakari Lukkarinen |
| 1.2 | Highlighted all instructions with red font. | 19.3.2023 | Sakari Lukkarinen |
| 1.3 | Updated Appendix 2 based on comments. Updated table of contents. | 22.3.2023 | Saana Vallius |
| 2.0 | Added a section for group work summary. Changed the highlighted instructions to an easier color to handle. | 22.10.2023 | Saana Vallius |
| 3.0 | Updated 50% chapter 2: Theoretical background | 11.04.2023 | Tu Dinh |
| 4.0 | Updated group work midway summary | 13.04.2023 | Tu Dinh |
| 5.0 | Added introduction of the report | 14.04.2023 | Minh Nguyen |
| 6.0 | Updated 30% chapter 3: Methods and material | 14.04.2023 | Nhut Vo |
| 7.0 | Updated 100% chapter 2: Theoretical background | 22.04.2024 | Tu Dinh |
| 8.0 | Updated 100% chapter 3: Methods and material | 24.04.2024 | Nhut Vo |
| 9.1 | Updated the content of chapter 4: Implementation | 08.05.2024 | Nhut Vo, Tu Dinh |
| 9.2 | Updated report abstracts | 08.05.2024 | Nhut Vo |
| 9.3 | Updated final summary and conclusion | 08.05.2024 | Tu Dinh |

Contents

[1 Introduction 1](#_Toc148898689)

[2 Theoretical Background 1](#_Toc148898690)

[3 Methods and Material 6](#_Toc148898691)

[4 Implementation 8](#_Toc148898692)

[5 Group Work Summary 17](#_Toc148898693)

[5.1 Midway Summary 17](#_Toc148898694)

[5.2 Final Summary 18](#_Toc148898695)

[6 Conclusions 19](#_Toc148898696)

[References 19](#_Toc148898697)

Appendices

Appendix 1: Introducing Figures and Other Items

Appendix 2: Using Appendices

# Introduction

HeartBest Pulse Oximeter is the final project developed for the Hardware 2 course at Metropolia University of Applied Sciences. The project aims to introduce first-year IT students to basic hardware concepts by utilizing a system designed for measuring and analyzing heart rate.

The project scope includes developing a local algorithm for heart rate detection on the device, performing heart rate variability (HRV) analysis, establishing a connection to the Kubios Cloud HRV analysis service for detailed analysis, and displaying estimated stress and recovery status indexes on the device.

The documentation comprises five major chapters, offering a comprehensive overview of the project. Chapter 1 provides an overview of the project, outlining its purposes and structure. Chapter 2 dives into the theoretical background surrounding physiological phenomena relevant to heart rate measurements. Chapter 3 explores the technical properties employed in the project. Chapters 4 and 5 elaborate on the development process, its challenges, and results, respectively, followed by a conclusion and self-evaluation in Chapter 6.

The intended audience for the report includes professional lecturers and other students.

# Theoretical Background

The following section presents the scientific background behind the heart rate physiology and explains the signal utilized in the project. This section is divided into two parts. The first subsection introduces key medical terminologies such as "heart rate" and "heart rate variability," alongside their corresponding measurement norms and associated physiological phenomena. The second subchapter concentrates on the signal utilized in the project and elaborates on how the sensor detects this signal.

## Heart rate. Heart rate variability

Heart rate, also known as pulse rate, represents the frequency of heart contractions, which is typically measured by the unit of beats per minute (BPM) [1]. The contractions come from the ventricles in the lower chambers of the heart, sending pressure waves of blood through the arteries. These waves can be detected at the wrist or on a finger to measure the pulse rate. [2].

Heart rate variability (HRV) is a measure of the time variation between successive heartbeats. This beat-to-beat fluctuation is measured in milliseconds [3]. HRV is a valuable indicator for assessing physiological stress levels, primarily influenced by the sympathetic system, as well as monitoring recovery, which is controlled by the parasympathetic nervous system [4].

A typical resting heart rate varies from 60 to 100 beats per minute (bpm). Highly conditioned athletes may reach the resting heart rate as low as 40 bmp. However, such a minimum heart rate might be worrisome for an average individual. The maximum recommended heart rate is often calculated using the formula: 220 minus your current age [5].

## Physiological phenomena associated with heart rate and HRV

Heart rate is regulated by two divisions of autonomic nervous system: the parasympathetic nervous system (PNS) and the sympathetic nervous system (SNS). SNS is responsible for releasing hormones such as catecholamines, epinephrine and norepinephrine, which accelerate the heart rate and reduce heart rate variability. [6] On the other hand, PNS is known for exerting the opposite effect on the heart rate and HRV: it decreases heart rate while increasing heart rate variability.

Based on the indexes of SNS and PNS, the level of stress and recovery are indicated. During rest, a normal PNS index falls within the range of -2 to 2. During stress or intensive physical activity, this index can be much lower.

A diagram of parasympathetic tone

Description automatically generated

*Figure 2.1: PNS index [7]*

A similar interpretation can also apply to the SNS index. SNS values may vary widely, reaching up to level 5-35 during physical exercise or stressful condition [7].

A graph of a normal distribution

Description automatically generated with medium confidence

*Figure 2.2: SNS index [7]*

## Heart rate signal detection

Cardiac activity can be detected through various methods, including ECG (electrocardiograph), PW (pulse wave), PCG (phonocardiogram), or ACG (apex cardiogram). [8] In this project, PW method is used to estimate heart rate by tallying pulse occurrences in the large arteries. PW provides a simpler and straightforward means of accessing heart rate information, detectable through using a signal sensing technique called photoplethysmography (PPG).

Photoplethysmography is an optical technique used to measure the changes of blood volume within the microvascular tissue bed. PPG method typically involves placing two components against the skin: a light source (light emitting diode, or LED) and a photodetector. The light source emits the light, which is then captured by the photodetector. The intensity of the light indicates changes in arterial blood volume, leading to variations in the PPG signal.

There are two types of PPG sensors: transmissive and reflective. A transmissive sensor requires the LED and photodetector to be placed on the opposite sided of the measured skin surfaces. On the other hand, a reflective sensor has both the light source and photodetector on the same side, making them convenient because the sensor can be placed on any skin surface, such as the forehead, wrist, and hand palm. [9]

Diagram of a diagram of a light source

Description automatically generated

*Figure 2.3: Two types of PPG sensors [9]*

In this project, Crowtail Pulse Sensor v2.0 is utilized, employing a reflective sensor system. As can be seen from the Figure 2.2 below, the LED and photodetector are located right in the middle of the board. It is recommended to place the sensor on the earlobe or fingertip and sit in a relaxed posture to begin measurement. [10]

A red square with white heart and white text

Description automatically generatedA red circuit board with white connectors

Description automatically generated

*Figure 2.4: Crowtail Pulse Sensor v2.0 from the front and back [10]*

# Methods and Material

This chapter will primarily explain the materials used for the project. Firstly, the hardware part will be discussed, followed by the software aspect in the next section. Additionally, the techniques employed will be covered. Finally, the chapter will be concluded by summarizing the overall approach of the project.

## Hardware

### Main controller

To develop this project, Raspberry Pi Pico W is chosen to be the microcontroller board because it supports various peripherals and uses a 2.4 GHz wireless LAN connection. This connection is used to send HRV data to Kubios for further analysis.

A computer chip with many different colored wires

Description automatically generated with medium confidence

*Figure 3.1.1: Raspberry Pi Pico W pinouts*

### OLED display

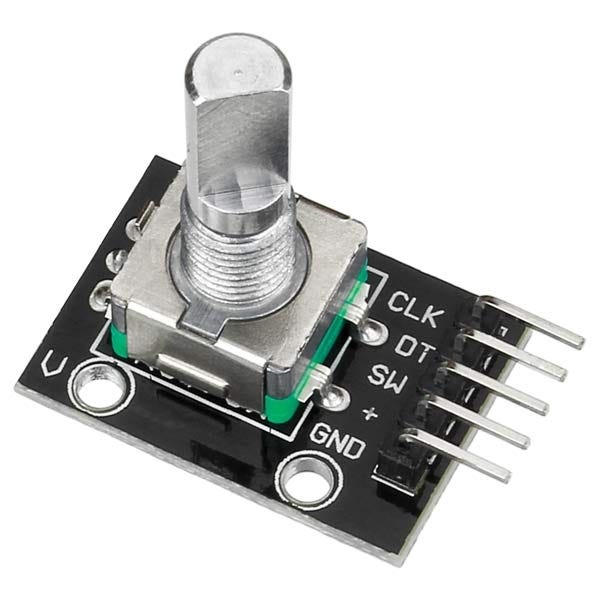
A monochrome digital SSD1306 OLED display which has 128x64 dimensions is chosen to display users’ selection from the menu as well as visualize information about the process and results of the heart rate measurement. By communicating via I2C, SSD1306 OLED provides clear, energy-efficient images with simple integration, making it perfect for compact devices with limited power and space.

A close up of a device

Description automatically generated

*Figure 3.1.2: OLED display*

### Rotary encoder



*Figure 3.1.3: Rotary encoder*

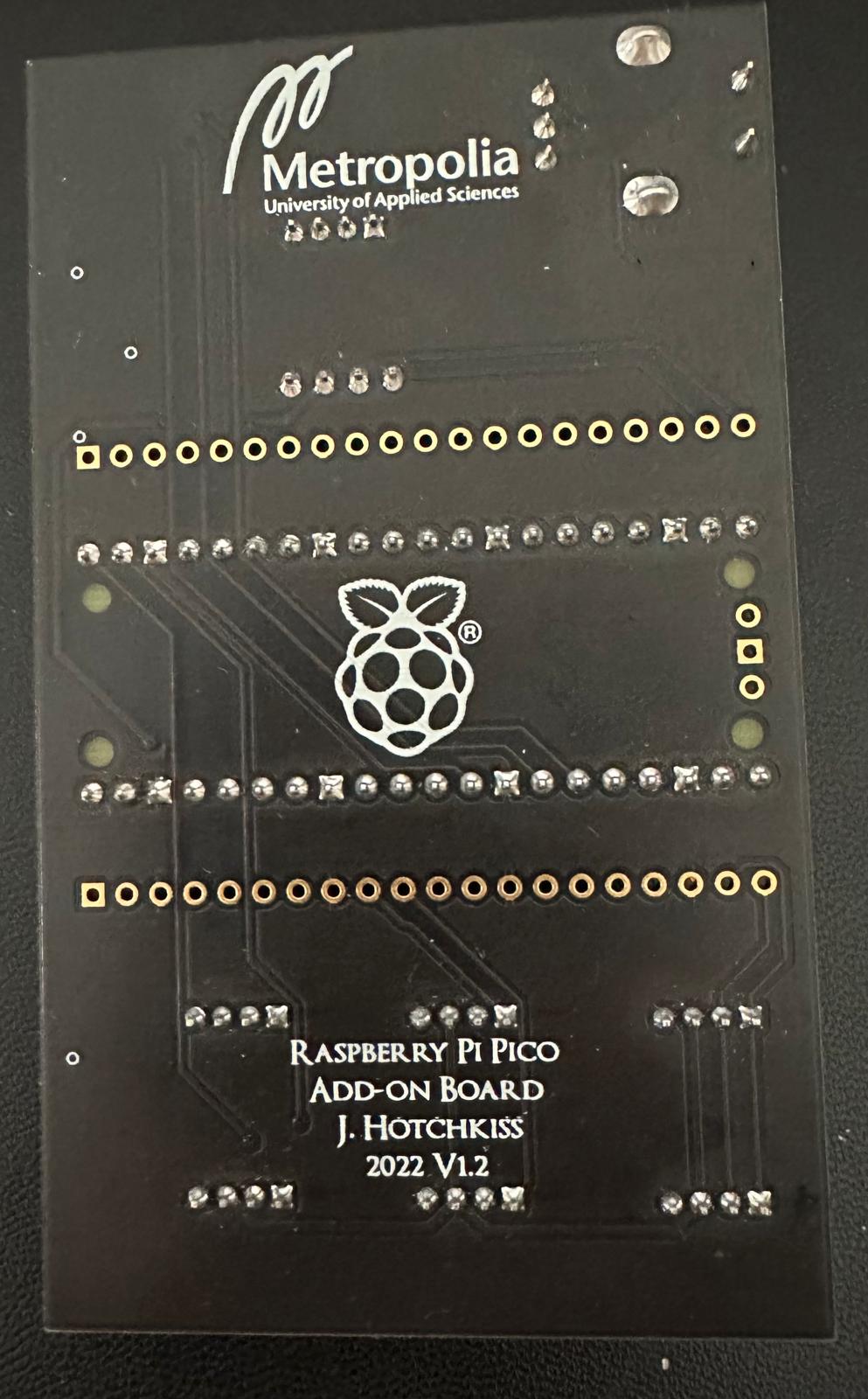
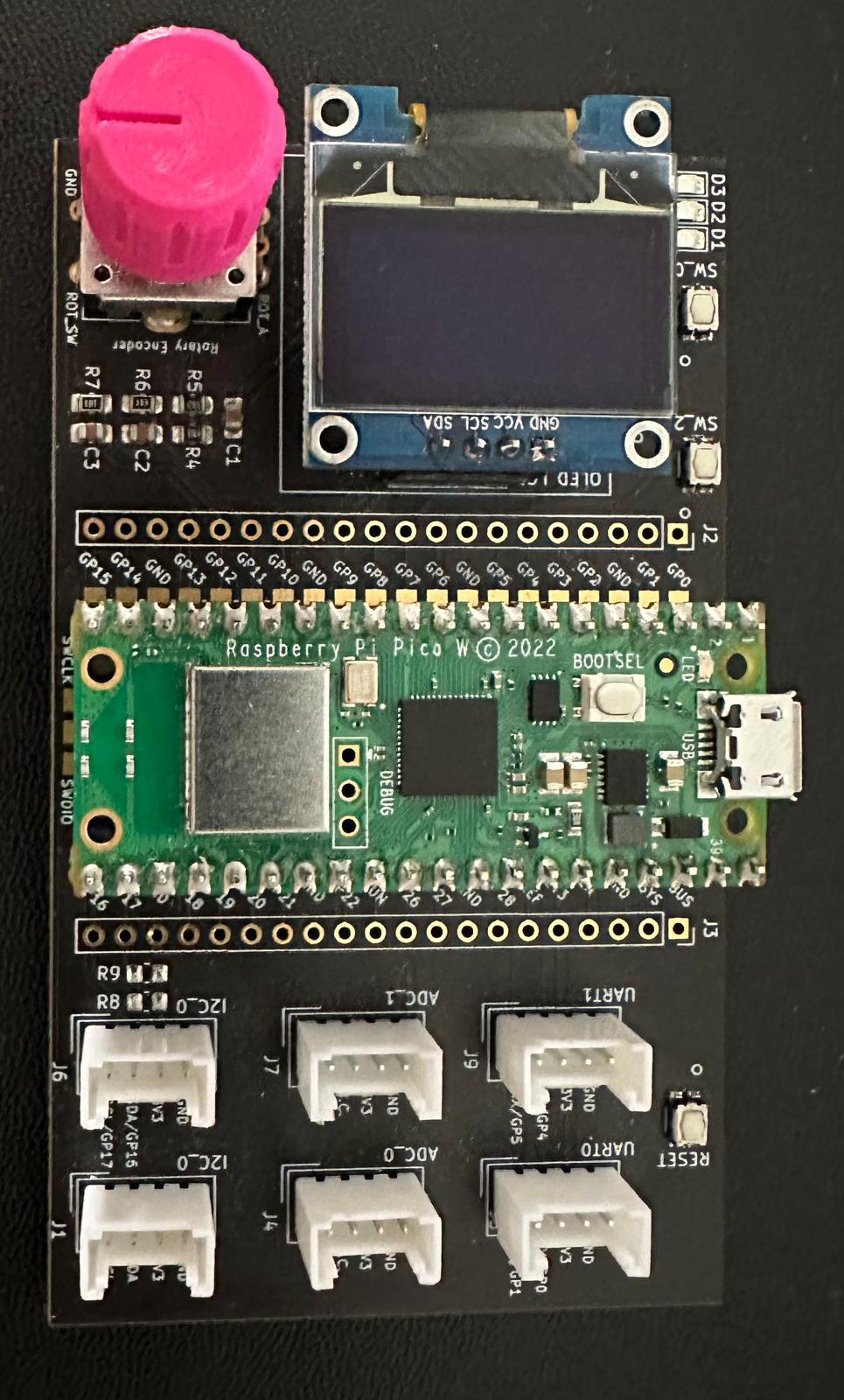
A controller is required to customize the menu options and interact with the program. With two functions: ROT A, ROT B and ROT Switch, the rotary encoder is the perfect combination for this project.

### Crowtail Pulse Sensor v2.0

Crowtail Pulse Sensor v2.0 uses the Grove connector to connect the analog inputs of the Raspberry Pi Pico. The electrical components of the heart rate sensor are attached to the back side of the sensor PCB. *(see figure 2.4)*

The component side also includes the Grove connector, which is used to connect the sensor to add-on board. The finger sensor is placed on the front side of the sensor PCB.

### Add-on Board and Grove connectors



*Figure 3.1.5: Front and back of add-on board and components*

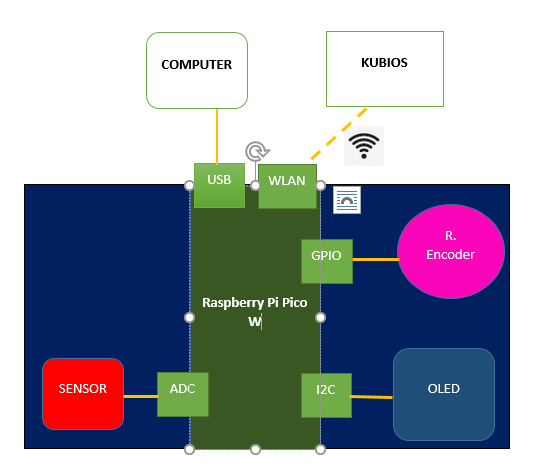
An add-on board was developed by Joseph Hotchkiss of Metropolia University of Applied Sciences to connect all the components and ensure they are electrically and inter-connectively compatible. The add-on board is designed to optimize space for components in the most reasonable way. The Grove connections used to connect the Crowtail Pulse sensor are also mounted on the circuit board.

## Software Requirements and Method

The Raspberry Pi Pico W is programmed using Thonny IDE and MicroPython. Thonny IDE, designed for embedded systems, makes it easy to download MicroPython firmware onto the Raspberry Pi Pico W and program it directly. For example, the MicroPython-specific ssd1306 library, which uses the FrameBuffer library, manages the OLED display. The project also makes use of other libraries, such as machine, array, network, socket, urequests, and ujson.

Moreover, the lecturers provided two files: fifo.py and piotimer.py. Fifo.py assures that logical data is stored in an interrupt-safe manner, and piotimer.py adds hardware timing classes.

## Operating Principle



*Figure 3.3: Visualisation of the hardware connection*

The system operates independently, allowing users to interact with the program via a rotary encoder. Interaction feedback is displayed on the OLED screen. Users can choose the operating by rotating or pressing the encoder.

Once measurement starts, the Crowtail optical sensor captures analog heart rate data, transmitted to the Raspberry Pi Pico. The microcontroller's ADC converts this to digital format and using that for calculations as well as operating programed systems.

# Implementation

The explanation in this section will clarify how the team had built the software architecture with logics, algorithms, and data to make the end-to-end system for the precise heart rate measurement with both offline and online connection.

## Software logic

With the purpose of building end-to-end system, which means it works like a standalone system when being powered up, the Raspberry Pi Pico W is the core part of the system. It collects data from the pulse sensor and processes the data based on each selected working option. In an overview, there are 4 working modes (based on the “Menu” class – our User Interface):

* **Measure HR**: collecting data, using algorithm to calculate the heart rate (beats per minute) and displaying the heart rate continuously every 5 seconds on the OLED.
* **BasicHRV**: collecting data, using medical calculation to analyse *meanppi, meanhr, sdnn, rmssd*, displaying those results on the OLED as well as connecting Pico to Wi-Fi and sending the results as a message over MQTT.
* **Kubios**: collecting data, sending the data to Kubios sever via internet connection for analysis and displaying the received results on the OLED as well as saving the results as txt file to the Pi Pico W.
* **History**: Accessing to the Pi Pico W memory to take the saved results from the last Kubios measurement and displaying it on OLED screen.

### Main program

The main system was logically and efficiently programmed. The flow chart below illustrates how the main system operates.

A diagram of a computer

Description automatically generated

*Figure 4.1: Main program’s logic*

When the device is powered up, it starts and shows the Welcome Text Screen. After 3 seconds, the Menu screen appears, where all the program's activities happen.

In this Menu state, we use a variable called "index" to control the selection of program options. We also use a boolean variable to manage inputs from the encoder: RotA=1, RotB=-1, press=0 (if boolean is true). Rotating the encoder changes the index value, indicating the chosen selection. Pressing the encoder sends a value of 0, indicating a selection action.

When in the Options state, the program executes the activities programmed for each corresponding option. There are 4 options in total as mentioned above: Measure HR, Basic HRV, Kubios and History. At this stage, the Boolean variable is set to false. Pressing the encoder now functions as a "go back" button, returning to the menu screen.

The logical algorithms ensure the system runs independently and stays in a loop without interruption.

### Option 1 from the Menu – Measure HR

In this option, a user can measure their heart rate continuously and stop whenever they wish. The heart rate detection algorithm is utilized here and will be discussed later in the report (section 4.2.1). No data will be saved after exiting this option.

The heart rate detection algorithm initiates whenever a “Heart\_ADC” class is invoked in the program. At the same time, an object named “tmr” of the “Piotimer” class is instantiated, configuring a hardware-based timer using RP2 PIO (Raspberry Pi Pico Programmable Input/Output) to generate hard interrupts. These interrupts are handled immediately by the Pico’s processor, making them suitable for the project application where accurate timing is essential. The timer is set to trigger periodically (mode=Piotimer.PERIODIC) at the specified frequency (freq=250). When “tmr” is triggered, it also calls the provided callback function that stores the heart rate signals in the FIFO memory. The raw digital signal values are then processed in our pulse signal filter methods and heart rate validation algorithm.

### Option 2 from the Menu – Basic HRV

When option 2 is selected, Raspberry Pi Pico initiates the collection of heart rate data from the sensor within 30 seconds. Using medical formulas provided by Kubios and algorithms developed by the team (which will be discussed later in section 4.2.2), basic information such as mean PPI, mean HR, SDNN, RMSSD are calculated and displayed on the OLED screen. At the same time, Raspberry Pi Pico establishes a connection to the group's Wi-Fi network and transmits this basic analysis as a message to a client laptop over MQTT (which was established in the Networks course).

### Option 3 from the Menu – Kubios

When a user selects option 3, the Raspberry Pi Pico will also collect data from the sensor for 30 seconds and encapsulate peak-to-peak intervals into a list. Next, the Pico connects to the group’s Wi-Fi network and log in to Kubios Cloud. After successfully logging into Kubios, the Pi Pico delivers the PPI list via the requests.post method. Once the Pi Pico retrieves the results from Kubios, it displays them on the OLED. Finally, the system generates a text file and saves it in the Pico for later use in Option 4: History.

### Option 4 from the Menu – History

This option allows users to access the Pi Pico's memory to search for a text file saved from the implementation of the Kubios Option. The latest data will be displayed on the OLED screen using the format created by the team.

## Developed algorithms

### Pulse signal detection and heart rate validation

The following table outlines the process of calculating heart rate (HR) from analog signals. The process is broken down into several steps, each contributing to the overall HR calculation. Each step is implemented as a method within “Heart\_ADC” class.

|  |  |
| --- | --- |
| **Steps** | **Purpose and implementation** |
| 1. filter\_raw\_AD() | Read the raw analog signal from the ADC and compute an average of 5 running analog-to-digital (AD) values. The calculation is meant to smoothen out the noise and fluctuations in the signal. |
| 1. find\_threshold() | Determine the minimum and maximum values of the filtered signal and calculate a threshold value. The threshold for the peak detection is set to 80% of this range. |
| 1. find\_peak() | Identify peaks in the filtered signal that exceed the threshold. Peaks in the filtered signal are detected by comparing each data point to the threshold value. Peaks are identified as points where the signal rises above the threshold, and the current signal value is higher than both the previous and next values. |
| 1. calculate\_bpm() | Calculate the heart rate (beats per minute, or BPM) from the detected peaks. The time difference between successive peaks, known as the peak-to-peak interval (PPI), is calculated. PPI is converted to milliseconds (ms) by multiplying with the inverse of the sampling frequency (f = 1/ 250 Hz). BPM is then computed using the formula: BPM = 60,000 / PPI. |
| 1. find\_averageHR() | Compute an average of 5 latest heart rates to remove unusual heart rate values and determine the final valid heart rate. |

*Table 1: Explanation of heart rate algorithm*

### Heart rate variability analysis

In addition to heart rate detection, our Raspberry Pi Pico is programmed to conduct local basic heart rate variability (HRV) computation. The basic HRV analysis will yield results for mean heart rate, mean PPI, SDNN, and RMSSD.

When a user chooses to initiate local HRV analysis on the Pico, the device first captures 30 seconds of data. After that, the heart rate and PPI values are saved for further analysis. Calculating the mean heart rate and mean PPI involves a straightforward process of averaging all provided values. The mathematical computations for SDNN and RMSSD values are more complex. Therefore, the following medical formulas provided by Kubios was used as a guideline for our algorithm.

A group of math equations

Description automatically generated

*Figure 4.2.1: Formulas provided by Kubios [12]*

Below are the steps how the program calculates SDNN - a standard deviation of normal PPI values:

* Calculate the mean of peak-to-peak-intervals (PPIs)
* Subtract the mean PPIs from each individual PPI, square the result, and sum up all the squared deviations
* Calculate the variance: Take the above squared deviations, divide by the total number of PPIs
* Calculate the standard deviation: take the square root of variance.

A computer screen shot of a program

Description automatically generated

*Figure 4.2.2: SDNN algorithm in the program*

The steps how the program calculates RMSSD - a square root of the squares of the successive differences of peak-to-peak-intervals:

* Calculate the differences between adjacent PPIs
* Square each difference above
* Calculate the mean of the squared differences
* Take the square root of the mean from the above result

A computer screen shot of a program

Description automatically generated

*Figure 4.2.3: RMSSD algorithm in the program*

# Group Work Summary

## Midway Summary

The project has been developed for three weeks. During the short period of time, the group has completed the following tasks:

### Programming

Using the exercises provided, a simple user interface on the OLED screen has been developed, which enable users to update the OLED display while receiving input from Thonny Shell console and interacting with Pico’s buttons.

In the OLED display exercise, each team member (Tu, Minh and Nhut) has tackled one exercise individually. However, for the exercise of week 2, which involved finding peaks and plotting with Thonny, the group collaborated on the program together for all the exercises.

To facilitate collaboration and version control, a GitLab repository was created for the project, with a pre-made pico-lib as a submodule. The group's remote repository was pushed to Tu's GitLab profile.

### Project background research and report writing

The introduction section of the report has been crafted by Minh. Tu is currently working on the theoretical background research and writing, with progress at 50%. Nhut is responsible for drafting the methods and materials section, also at 50% completion.

### Difficulties during the project implementation

The group encountered challenges in comprehending the FIFO module and its implementation in our upcoming programs. Similarly, the Week 2 exercise initially proved challenging due to our incomplete grasp of how the heart rate peaks and PPIs are determined. As the result, we collaborated closely on the programming tasks. Thanks to the assistance of other brilliant students, we successfully navigated through the difficulties posed by the Week 2 exercise.

## Final Summary

### A self-evaluation of the team’s overall performance

The project has been successfully implemented thanks to collaborative effort of the team. Despite facing challenges during the implementation process, including integrating code fragments, misinterpreting heart rate calculations, and addressing bugs in programming the press button, we successfully resolved all issues and delivered the final working product on schedule.

### The contribution from Tu Dinh

Tu conducted background research on heart rate topics and developed algorithms to filter, calculate, and analyze heart rate and other HRV measurements. Additionally, Tu authored the theoretical background section of the report and contributed to the implementation section, focusing on the program algorithm. Furthermore, Tu played a role in writing the introduction, group work summary, and conclusion of the report. On the presentation day, Tu prepared and presented her own work.

### The contribution from Nhut Vo

Nhut conducted background research on kubios connection, MQTT and building algorithms for main program as well as UI display for the program. In the report, Nhut is responsible for abstract, methods and materials, software logic. On the presentation day, Nhut prepared and presented his own work.

### The contribution from Minh Nguyen

Minh is responsible for the introduction of the report and the design of the program's welcome screen. Minh also participated in discussing ideas for the main program's logic and solving arising problems.

# Conclusions

In conclusion, the project of developing an embedded system for a heart rate monitoring device was a success. We implemented a system that collected heart rate information from a pulse sensor using PPG technique and employed local calculations to analyze HRV in both offline and online modes. The hardware components, including the Raspberry Pi Pico W, Crowtail pulse sensor, and OLED screen, were effectively integrated with a software architecture comprising algorithms, data structures, and network functionalities, developed using MicroPython and Thonny IDE.

Throughout the project, we encountered some challenges, such as integrating each team member's code fragments into the main program. Despite this, through effective communication and collaboration, we were able to overcome these obstacles and deliver a perfectly working product on time.

The final system still has some limitation, for example, the local SDNN and RMSSD calculations sometimes yield abnormal results. Furthermore, the user interface could be enhanced to be more appealing, offering a better experience for users interacting with the device.

Overall, the project has provided students with a solid foundation in knowledge and programming skills related to interactive embedded systems. It has been a fruitful experience, and the lessons learned on this project can be effectively applied in the future.

References

1. Heart rate [online]. 2024. Physiopedia. <https://www.physio-pedia.com/Heart_Rate>. Accessed 22 April 2024.
2. Harvard Health Publishing Staff. 2024. Heart rate variability: How it might indicate well-being. <https://www.health.harvard.edu/blog/heart-rate-variability-new-way-track-well-2017112212789>. Harvard Health Publishing. Accessed 22 April 2024.
3. Lukkarinen, Sakari. 2024. Hardware 2 project requirements. Organisation internal documentation. Metropolia University of Applied Sciences.
4. LeWine, Howard. 2023. What is a normal heart rate. <https://www.health.harvard.edu/heart-health/what-your-heart-rate-is-telling-you>. Harvard Health Publishing. Accessed 22 April 2024.
5. Heart rate [online]. 2024. UC Davis Health. <https://health.ucdavis.edu/sports-medicine/resources/heart-rate#:~:text=Heart%20rate%20is%20controlled%20by,to%20accelerate%20the%20heart%20rate>. Accessed 22 April 2024.
6. PNS AND SNS INDEXES IN EVALUATING AUTONOMIC FUNCTION [online]. 2024. Kubios. <https://www.kubios.com/hrv-ans-function/>. Accessed 22 April 2024.
7. He, Runjie; Shen, Lingyu; Wang, Zhuo; Wang, Guoqing; Qu, Hang; Hu, Xuehao & Min,Rui. 2023. Optical fiber sensors for heart rate monitoring: A review of mechanisms and applications. ScieneDirect.
8. Naeini, Emad Kasaeyan; Sarhaddi, Fatemeh; Azimi, Iman; Liljeberg, Pasi; Dutt, Nikil & Rahmani, Amir. 2023. A Deep Learning–based PPG Quality Assessment Approach for Heart Rate and Heart Rate Variability.
9. Biswas, Dwaipayan; Simões-Capela, Neide; Van Hoof, Chris & Van Helleputte, Nick. 2019. Heart Rate Estimation From Wrist-Worn Photoplethysmography: A Review. IEEE SENSORS JOURNAL.
10. Chow, Melanie. 2019. Using a rotary encoder on an Arduino. [Using a rotary encoder on an Arduino | by Melanie Chow | Medium](https://medium.com/@melaniechow/using-a-rotary-encoder-on-an-arduino-18f543ae7f78). Accessed 23 April 2024
11. Raspberry Pi Pico and Pico W [online]. 2024. Raspberry Pi. [Raspberry Pi Documentation - Raspberry Pi Pico and Pico W](https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html). Accessed 24 April 2024.
12. HRV ANALYSIS METHODS – HOW IS HRV CALCULATED. 2024. Kubios. <https://www.kubios.com/hrv-analysis-methods>. Accessed 8 May 2024