FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Jimi Käyrä

OPTICAL HEART RATE MEASUREMENT WITH ARDUINO MKR1000

Bachelor's Thesis
Degree Programme in Computer Science and Engineering
June 2022

Käyrä J. (2022) Optical Heart Rate Measurement with Arduino MKR1000. University of Oulu, Degree Programme in Computer Science and Engineering, 25 p.

ABSTRACT

The primary goal of this thesis is to introduce the reader to the fundamentals of optical methods for cardiovascular monitoring. Opposed to traditional methods that rely on measuring the electrical activity of the heart with electrodes, optical methods utilize light and are thus considered indirect methods. The thesis begins with an introduction to photoplethysmography, a commonly used technique based on illuminating the skin and measuring the intensity of the reflected or transmitted light. In addition to presenting the operating principle of this method, some of the key issues and use cases are discussed; in particular, methods for estimating heart rate will be presented in more detail, including an example algorithm. Additionally, the core principles of the Arduino ecosystem and the WebSocket protocol will be considered.

In the latter part of this thesis, an implementation of an Internet of Things -capable optical heart rate meter based on the Arduino MKR1000 will be presented. Finally, the drawbacks and benefits of both photoplethysmography and the implemented system will be discussed in brief.

Keywords: photoplethysmography, cardiovascular monitoring, WebSocket, Internet of Things

Käyrä J. (2022) Optinen sykemittaus Arduino MKR1000 -laitteella. Oulun yliopisto, Tietotekniikan tutkinto-ohjelma, 25 s.

TIIVISTELMÄ

Tämän kandidaatintyön tavoitteena on esitellä lukijalle sydän- ja verisuonijärjestelmän toiminnan mittaamiseen käytettävien optisten menetelmien pääperiaatteita. Toisin kuin perinteiset menetelmät, jotka perustuvat sydämen sähköisen toiminnan mittaamiseen elektrodien avulla, optiset menetelmät hyödyntävät valoa ja ovat siten epäsuoria menetelmiä. Työssä käsitellään yleisesti käytössä olevaa fotopletysmografiaa, joka perustuu ihon valaisemiseen ja heijastuneen tai läpäisseen valon intensiteetin mittaamiseen. Fotopletysmografian toimintaperiaatteen esittelemisen lisäksi käsitellään joitakin tärkeimpiä käyttökohteita ja menetelmän haasteita. Tarkemmin käsitellään fotopletysmografiaan perustuvan sykemittauksen toimintaperiaate ja esitellään esimerkkialgoritmi. Tämän jälkeen käsitellään pääpiirteittäin Arduino-ekosysteemiä ja WebSocket-protokollaa.

Työn jälkimmäisessä osassa esitellään esineiden Internet -käyttöön soveltuva toteutus optisesta sykemittarista. Toteutukseen käytetään Arduino MKR1000 -kehitysalustaa ja hyödynnetään WebSocket-protokollaa. Lopuksi pohditaan lyhyesti sekä fotopletysmografian että toteutetun sykemittarin ansioita, haasteita ja jatkokehitysmahdollisuuksia.

Avainsanat: esineiden Internet, fotopletysmografia, kardiovaskulaaritoiminnan seuranta, WebSocket

TABLE OF CONTENTS

FOREWORD

I would like to thank Dr. Ella Peltonen for supervising this thesis and providing insightful feedback. Additionally, I would like to extend my thanks to my family and friends for supporting me throughout my studies.

Oulu, June 6th, 2022

Jimi Käyrä

LIST OF ABBREVIATIONS AND SYMBOLS

AC	alternating current
ADC	analog-to-digital converter
API	application programming interface
BP	blood pressure
CSS	Cascading Style Sheets
deoxy-Hb	deoxygenated hemoglobin
DC	direct current
DOM	Document Object Model
ECG	electrocardiography, electrocardiogram
HTML	HyperText Markup Language
HR	heart rate
IoT	Internet of Things
IR	infrared
LED	light-emitting diode
LS	light source
MA	motion artifact
OHRM	optical heart rate monitoring
oxy-Hb	oxygenated hemoglobin
PD	photodetector
PP	peak-to-peak
PPI	peak-to-peak interval
PPG	photoplethysmography, photoplethysmogram
UI	user interface

A_i	absorbance of the ith layer
c_i	concentration of the ith layer
d_i	optical path length in the ith layer
e	Euler's number
f_s	sampling rate
I_i	light intensity in the ith layer
I_in	intensity of incident light
I_trans	intensity of transmitted light
L	threshold value
M	threshold value
n	number of layers
N	window size
N_cd	length of cooldown period
p_i	sampling instant of the ith peak
$\mathrm{PPI_s}$	peak-to-peak interval in samples
$\mathrm{PPI_t}$	peak-to-peak interval in seconds
$x[n]$	nth sample
ε_i	extinction coefficient of the ith layer

1. INTRODUCTION

In the recent years, wearable sensors have become commonplace and widely available. Featured on everyday devices such as smartwatches, they are utilized both by athletes and ordinary individuals for health monitoring. As a result, the need for reliable yet convenient methods for measuring various health-related metrics has emerged. Optical methods are often used since they are convenient and possess numerous advantages over traditional methods. In particular, heart rate (HR) is considered an important metric; it provides a good glimpse of an individual's overall health while still being easy to measure and interpret.

Additionally, various health-related cloud services have been gaining in popularity. Since health monitoring devices often feature Internet of Things (IoT) connectivity, the data produced by them can be distributed to the cloud for further analysis. By utilizing machine learning approaches, it has become possible to derive useful physiological information that can then be utilized for improving the user's health, including disease prevention and detection.

In this thesis, some methods for physiological monitoring with optical methods will be considered. The primary aim is to introduce the reader to the fundamentals of optical methods for cardiovascular monitoring by conducting a brief literature review on the operating principles, use cases and key issues of these methods; in particular, optical heart rate monitoring (OHRM) will be considered. Additionally, concerning wearable IoT devices, the fundamentals of the WebSocket protocol and the Arduino platform used for prototyping purposes will be presented. With this foundation, an IoT-based optical heart rate monitor implementation utilizing the Arduino MKR1000 device will be presented in Chapter 3, demonstrating these concepts in practice.

2. RELATED WORK

This chapter introduces the reader to photopletysmography (PPG), an optical and non-invasive method for obtaining various cardiovascular metrics. Additionally, the core principles of the WebSocket protocol and the Arduino platform will be presented.

2.1. Photopletysmography

Traditionally, electrocardiography (ECG) has been viewed as the gold standard for measuring various heart-related metrics [1], meaning that it is considered superior to other methods specifically in terms of measurement accuracy and is often used as a point of comparison. This is due to the fact that ECG is a direct technique based on monitoring the electrical activity of the heart with bioelectrodes placed on the skin [2]. However, requiring electrode placement, ECG limits the mobility of the user and can be cumbersome to use, particularly in wearable applications [2].

In order to overcome these limitations regarding flexibility and convenience, optical methods have been developed. While being non-invasive like ECG, these methods utilize light instead of electrical activity for measuring various cardiovascular metrics. As such, they are considered indirect methods since unlike ECG, they do not directly measure the electrical activity of the heart.

The technique considered in this chapter is PPG, a popular optical method that can be utilized for a variety of purposes ranging from HR estimation to disease detection. Opposed to ECG, PPG features simpler hardware implementation and is considered more cost effective [2].

2.1.1. Operating Principle

PPG is based on illuminating the skin and measuring changes occurring in the absorption of the light as a function of time [2]. Since light absorption depends on volumetric changes of the blood occurring in the tissue, the events of the cardiac cycle are reflected in the intensity of the light and measuring this provides various kinds of information about the cardiovascular system. As such, the measurement system consists of two fundamental components, a light source (LS) and a photodetector (PD) [2]. Often, an infrared (IR) or green light-emitting diode (LED) is utilized since longer wavelengths generally provide better tissue penetration, resulting in better signal quality [2].

In addition, the raw PPG signal obtained from the PD requires preprocessing. Due to a relatively non-complex waveform, simple processing often suffices for obtaining a sufficiently clear signal [3]. For this purpose, an amplifier and filters are typically utilized. A low-pass filter is used for removing the baseline and a high-pass filter is applied in order to eliminate any high-frequency noise present in the signal [4]. A microprocessor is then utilized for processing the filtered signal and deriving useful properties from it [4].

PPG sensors can be classified by how the measurement is implemented [2]. In transmittance mode, the light source (LS) and the photodetector (PD) are placed on

opposite sides of the body part utilized for the measurement [2]. The light then passes through the tissue and the non-absorbed part reaches the detector placed on the opposite side. In contrast, when reflectance mode is utilized, the light source and the detector are placed side by side [2]. The detector then receives the light that is reflected back from the tissue. Reflectance mode is often utilized if reliable signal cannot be obtained with transmittance mode, e.g., due to large thickness of the body part or the presence of bones. In addition, it is often considered more convenient and is utilized in smartwatches. These two modes have been illustrated in Figure 1.

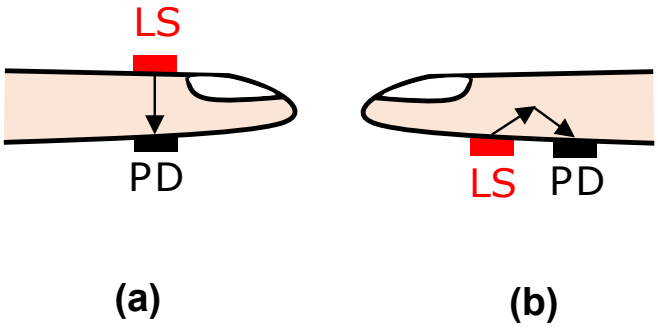

Figure 1. Illustration of (a) transmittance mode and (b) reflectance mode measurement on a finger.

Next, the physical foundation of the transmittance mode PPG measurement will be considered in brief. When light passes through a medium, the intensity of the transmitted light is given by Beer–Lambert's law [3]. According to it, the intensity I_{trans} of the transmitted light is

$$I_{\text{trans}} = I_{\text{in}}\mathrm{e}^{A} = I_{\text{in}}\mathrm{e}^{-dc\varepsilon}, \tag{1}$$

where I_{in} is the intensity of the incident light, e Euler's number and A absorbance [3]. The absorbance is dependent on the optical path length (d), the concentration of the medium (c) and the extinction coefficient (ε) which is an intrinsic property of the medium [3]. In the body part utilized for the measurement, the light passes through several layers of tissue that include the skin, muscle, bone and blood. Assuming that there are n layers in total with each layer having an absorbance of

$$A_i = -d_i c_i \varepsilon_i, \tag{2}$$

it can be seen by applying Equation 1 that the intensity of light passing through layer 1 is

$$I_1 = I_{\text{in}}\mathrm{e}^{A_1}. \tag{3}$$

Similarly, it can be deduced that the light intensity through layer 2 is

$$I_2 = I_1\mathrm{e}^{A_2} = I_{\text{in}}\mathrm{e}^{A_1}\mathrm{e}^{A_2} = I_{\text{in}}\mathrm{e}^{A_1+A_2}. \tag{4}$$

Continuing in this manner, it can be derived that the light intensity on the opposite side of the finger is then given by

$$I_n = I_{\text{in}} \mathrm{e}^{\sum_{i=1}^{n} A_i}. \tag{5}$$

The layers and the corresponding light intensities have been illustrated in Figure 2.

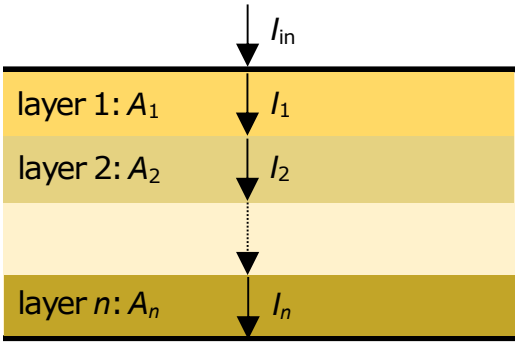

Figure 2. The n layers and their corresponding absorbances. In addition, the light intensities in each layer have been marked. I_{in} is the incident light intensity and I_n the intensity at the opposite side of the finger.

From this, it can be seen that the intensity of the transmitted light decays exponentially when the optical path length increases [3]. Since the pulsation of the heart results in periodic expansion and contraction of the arteries, periodic changes occur in the optical path length [5]. In this manner, the small blood volume changes caused by pulsation of the heart are reflected in the intensity of the transmitted light that is then measured and interpreted.

2.1.2. The PPG Waveform

The PPG waveform is comprised of two main components [6]. The large non-pulsatile direct current (DC) component corresponds to the baseline of the signal; it results from absorption of the light by static tissues, including bones, muscles and venous blood [6]. However, there is also some variation in the baseline due to low frequency fluctuations that can be caused by respiration and activity of the nervous system [3]. In contrast, the pulsatile, comparatively small alternating current (AC) component stems mainly from the small arterial blood volume changes in the tissue caused by pulsation of the heart as discussed in the previous subsection [6].

Since the AC component reflects volumetric changes of the arterial blood, numerous phases of the cardiac cycle can be identified from it. Some of these are highlighted in Figure 3. The first part of the waveform is the systolic component that mainly results from the pressure wave directly propagating from the left ventricle of the heart to the finger [7]. The highest point of this phase is the systolic peak that is reached when the blood volume in the tissue is maximal [3]. After this, the systolic phase ends in the dicrotic notch that corresponds to the closing of the aortic valve [8]. At this point, the diastolic phase begins. Primarily, it is the result of pressure waves that are reflected from smaller arteries located in the lower body and then reach the finger through the heart [7].

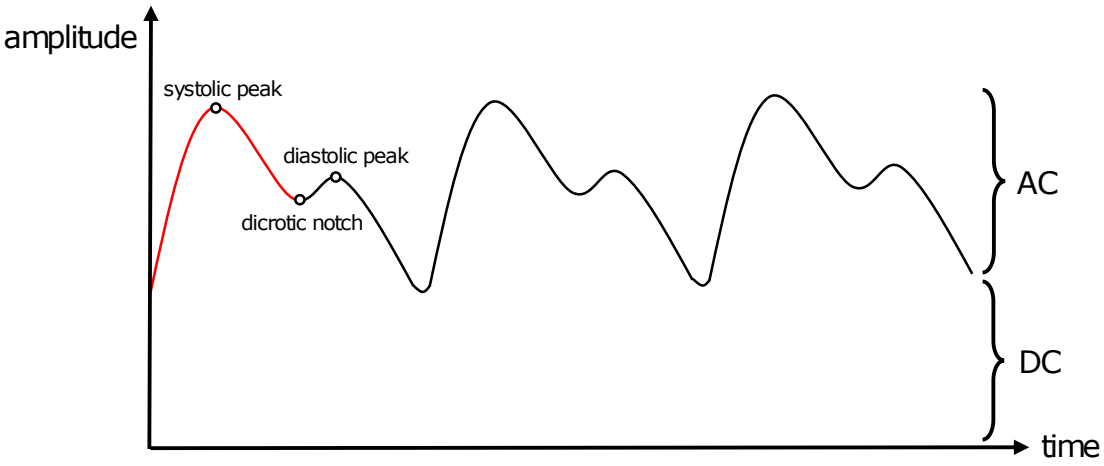

Figure 3. The PPG waveform and some of its components; the systolic phase of the first cycle has been emphasized with a red color. Note that the DC and AC components have not been drawn to scale for clarity.

2.1.3. Heart Rate Estimation

Reflecting the functioning of the cardiovascular system, there are numerous characteristics that can be derived from the PPG signal. Deriving heart rate (HR) will now be considered in more detail since unlike various other metrics, it does not require complex analysis of the PPG signal [9]. HR is also the metric that is utilized in the implementation part of this thesis.

In order to estimate HR from the PPG signal, one can utilize either the time domain or the frequency domain. In the time domain, HR estimation is typically based on finding the time intervals between certain parts of the periodic signal [10]. Perhaps the simplest approach is detecting peaks in the signal and finding the time differences between consecutive peaks. Often, the systolic peaks are utilized since they typically appear pronounced on the signal [10]. From the time intervals, one obtains the intervals between consecutive peaks called peak-to-peak intervals (PPI) that can then be utilized for HR estimation [10].

Numerous algorithms have been proposed for peak detection, ranging from relatively non-complex thresholding and derivative-based methods to methods that utilize more advanced tools, such as wavelets and neural networks [3]. While a simple algorithm can suffice for a high-quality PPG signal, more advanced algorithms are often required in practice. This is due to the fact that real-world PPG signals typically suffer from various kinds of noise and artifacts.

As an example of a peak detection algorithm, the mountaineer's method for peak detection proposed by Argüello-Prada [11] will now be considered briefly. In this method, the PPG pulse is modelled as a mountain with the systolic peak corresponding to the mountain top. The aim is to detect the systolic peaks and the sample $x[i]$ obtained at sampling instant i is considered to be a part of the systolic rising edge if it holds that

$$x[i] > x[i-1] \text{ when } i \geq 1. \tag{6}$$

A peak could then be detected at the point where Condition 6 no longer holds, i.e., when the derivative of the signal changes from positive to negative or zero. However, since noise could easily result in detecting erroneous peaks, a threshold value is utilized. The change in the derivative of the signal is only considered to indicate a systolic peak if Condition 6 has been met L times. This threshold value is updated dynamically based on the number of samples required to reach the systolic peak, enabling the detection of new peaks even when HR increases. The particular advantages of this method include ease of implementation and the fact that unlike several other methods, it does not depend on the signal amplitude [11].

In contrast, the frequency domain approach is generally based on calculating the power spectrum of the PPG signal in a time window [12]. From the resulting periodogram, the aim is to detect the fundamental frequency of the AC component that corresponds to the heart rate [12]. While one avoids the challenges associated with detecting peaks from the noisy signal in time domain, it can be at least equally difficult to discern the peak corresponding to the heart rate from the peaks caused by low-frequency noise present in the signal. In the approach proposed by Ahamed et al. [12], this is accomplished by utilizing multi-stage adaptive filtering and then singular spectrum analysis in the frequency domain for noise reduction. It is expected that this filtering suppresses the noise peaks sufficiently, making the peak corresponding to the HR easier to detect reliably.

2.1.4. Other Use Cases

In contrast to HR, other metrics that can be derived from the PPG signal generally require more complex analysis of the waveform [3]. Many of these metrics have important applications in a clinical setting and are based on the fact that blood circulation is modulated by various systems of the body [3].

A very common use case for PPG is measuring oxygen saturation (SpO_2). The measurement system is based on dual-wavelength PPG, typically featuring a red LED and an IR LED [4]. Since oxygenated (oxy-Hb) and deoxygenated (deoxy-Hb) hemoglobin absorb light at these wavelengths differently, the resulting light intensities can be used for estimating the amounts of oxy-Hb and deoxy-Hb present in the arterial blood [4]. SpO_2 can then be estimated from the relative amounts of oxy-Hb and deoxy-Hb [4].

Containing a respiratory component, the PPG signal can be also utilized for determining the breathing rate. There are various mechanisms on how respiration affects the PPG signal; one such mechanism is the variation of intrathoracic pressure that is then reflected in the PPG signal [2]. Since traditional methods for respiratory monitoring typically require wearing a chest band and a nasal cannula [2], a PPG-based method can greatly improve user comfort.

Serving as an indicator for hypertension, another research interest has been cuffless blood pressure (BP) measurement. While the relationship between BP and the PPG signal is not yet completely understood, deep learning models have already been succesfully utilized for BP estimation [13]. As an input to the models, either the raw PPG signal or some selected derived features, such as area under the curve or inflection points, are used [13].

Additionally, since the PPG waveform is modulated by individual differences, PPG can be utilized for assessing cardiovascular function and detecting diseases. One such metric is arterial stiffness that is indicated by the steepness of the systolic peak [2]. Furthermore, it has been shown that the second derivative of the PPG signal, indicating the acceleration of the blood, contains various kinds of useful information regarding the condition of the cardiovascular system [2]. For instance, it can be utilized for estimating the risk of coronary heart disease and determining the arterial stiffness index [2].

2.1.5. Issues

While photoplethysmography is considered to be better suited for wearable applications than ECG in many respects, it still suffers from several issues that complicate its use particularly in wearable devices.

Like many biosignals, the raw PPG signal is feeble. Due to this, it can be difficult to reliably discern the useful PPG signal from noise [10]. This issue can be exacerbated even further due to the fact that computationally intensive signal processing might be infeasible to implement in wearable devices that are typically resource-constrained, possibly resulting in a tradeoff between accuracy and resource consumption.

Concerning noise, the PPG signal is particularly susceptible to motion artifacts (MA) that are caused by voluntary or involuntary movements of the body [9]. Since these artifacts often overlap with the interesting parts of the signal in the frequency domain, basic filtering in the frequency domain is not applicable for eliminating the artifacts [9]. Instead, acceleration data is often measured alongside the PPG signal. Adaptive filtering can then be utilized for reducing the noise by using the accelerometer data as a reference signal [12].

In addition, the baseline of the signal can fluctuate due to events such as respiration and nervous activity [3]. Particularly in reflectance mode, the wander can also be caused by poor contact between the body part and the sensor. Since this can complicate the interpretation of the signal, adequate preprocessing is required in order to mitigate the baseline wander. Often, high-pass filtering is utilized for removing the fluctuation. However, similarly to MAs, the frequencies of the baseline wander often overlap with the frequencies of the AC component. High-pass filtering can thus result in distortion of the waveform [3].

Furthermore, numerous individual characteristics and environmental factors can easily alter both the shape and the amplitude of the resulting waveform. These factors include the user's age, gender, skin tone, condition of the vascular system and thickness of the subcutaneous fat layer [9]. For instance, low temperature or poor condition of the vascular system can cause hypoperfusion and result in lower signal amplitude [3]. However, as discussed previously, this can be also perceived as an advantage since these differences reveal information about the condition of the cardiovascular system. Overcoming these challenges requires careful algorithm planning that manages to take into account individual and environmental characteristics.

2.2. Arduino

Arduino[1] is an electronics platform originally developed for low-threshold prototyping. The Arduino ecosystem consists of Arduino boards and software. Although originally developed for prototyping purposes, Arduino devices have been gaining in popularity and are nowadays used for a variety of different purposes, ranging from teaching demonstrations to full-fledged projects. The key advantages of Arduino products include ease of use, low cost, cross-platform support and open-source platform [14].

The hardware side of the ecosystem consists of Arduino boards. They are essentially microcontroller boards that feature easily programmable interfaces, making it possible to develop prototypes without in-depth electronics knowledge. Some key features include USB connectivity and input/output pins that can be utilized for attaching sensors and actuators; two types of pins are featured. A digital pin has only two possible states (LOW and HIGH) whereas analog pins feature integer values from 0 to 1023. Since the boards contain a built-in analog-to-digital converter (ADC), the analog pins can also be used for sampling a signal provided by a sensor [15].

Concerning software, Arduino boards can be programmed with Arduino Integrated Development Environment (IDE), a tool featuring a code editor, an interface for progamming the device and debugging tools. For programming the device, the Arduino Programming Language built on top of C++ can be utilized. Furthermore, the structure of Arduino code is simple to understand since minimally, only two functions, `setup()` and `loop()`, are required [16]. The function `setup()` is ran once in the beginning of the program while `loop()` is ran repeatedly [16].

Additionally, a wide variety of different libraries providing extra functionality are available. For instance, there are libraries providing wrappers for wireless communication and tools for data processing. The Arduino IDE can be utilized for installing the libraries and utilizing them in the Arduino code.

2.2.1. MKR1000

In addition to the general features presented above, the Arduino MKR1000 utilized in the implementation part of this thesis features built-in Wi-Fi connectivity. This enables interacting with the device through the Internet and makes the board well-suited for IoT applications. Additionally, it features a powerful SAMD21-based processor [17].

Furthermore, the MKR1000 is compatible with Arduino IoT Cloud, an online platform for creating IoT projects. It features tools for configuring the Arduino board and automatically creating code for programming the board [18]. For displaying the data provided by the device, a visual interface called dashboard is featured [18]. For example, it could be utilized for monitoring temperature with an Arduino device and displaying this data on another device.

[1] `http://www.arduino.cc`

2.3. WebSocket

WebSocket is a full-duplex communication protocol facilitating bidirectional message exchange between a client and a server [19]. Compared to traditional half-duplex Hypertext Transfer Protocol (HTTP), WebSocket has less overhead and it eliminates the need to poll for new data from the server. Due to these advantages, it is especially well-suited for real-time web applications that require continuous and low-latency data transfer between a server and clients. Instead of initiating multiple Transmission Control Protocol (TCP) connections by periodic polling, only a single TCP connection is utilized.

An example of message exchange over WebSocket protocol has been presented in Figure 4. Initially, the client sends a WebSocket handshake request to the server. The handshake request is essentially a HTTP GET request that contains relevant headers, including an Upgrade header that invites the server to switch to the WebSocket protocol [19]. The server then provides a WebSocket handshake response containing the HTTP 101 Switching Protocols status code, indicating that the server is switching to the WebSocket protocol as requested by the client [19]. After the connection has been established, the client and server can send messages back and forth with minimal overhead. Finally, the connection can be closed by either party.

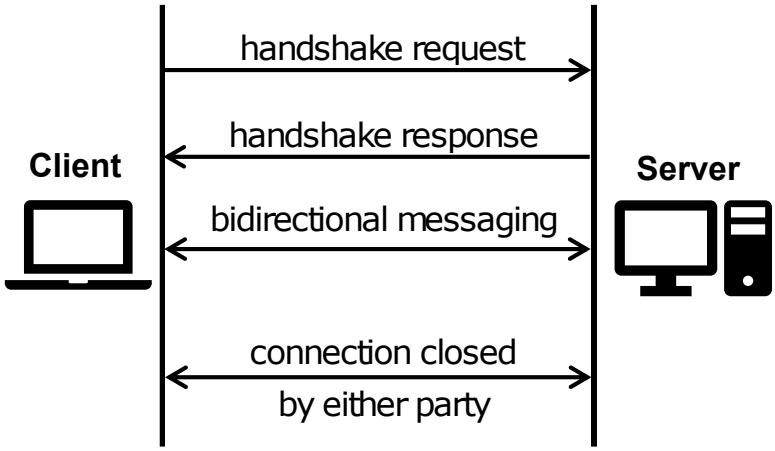

Figure 4. The messages exchanged between a client and a server during a WebSocket connection.

3. IMPLEMENTATION

In this chapter, the structure and functionality of the implemented heart rate measurement system is presented. The aim was to implement a system that demonstrates the main concepts of OHRM and IoT communication utilizing the WebSocket protocol.

The Arduino MKR1000 is utilized for implementing the OHRM. Furthermore, it connects to a Wi-Fi network and functions as a WebSocket server, sending the HR data to connected clients in the local network. The system architecture consisting of a Wi-Fi router, the MKR1000 with a pulse sensor attached and connected clients is depicted in Figure 5.

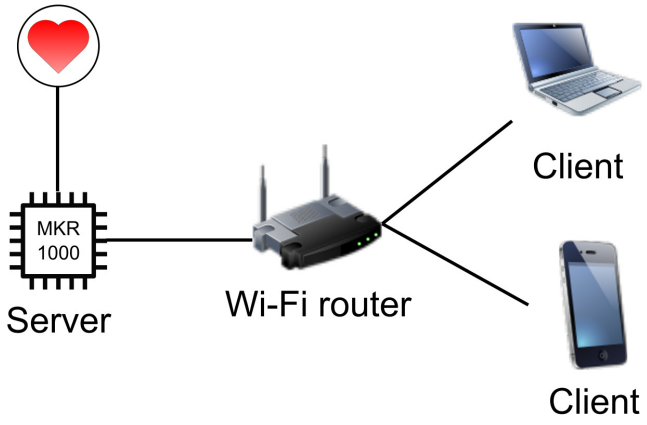

Figure 5. The architecture of the implemented system.

Next, the hardware and software implementation of the OHRM system is presented. Finally, the implementation of the client-server -functionality and the integration of the OHRM system into this will be considered.

3.1. Heart Rate Measurement

The implementation of the heart rate measurement system was divided into three parts. First, the utilized hardware will be discussed. Next, a method for detecting the peaks in the obtained signal and estimating the HR will be presented. Finally, the presented method will be implemented in software.

3.1.1. Hardware

Considering the hardware implementation of the measurement system, the initial plan was to utilize only basic electrical components and use Vishay's TCRT5000 which combines an IR LED and and an IR receiver into a single module. The simple circuit depicted in Figure 6 was built for this purpose. However, this approach was discarded since even though the PPG waveform could be identified, the resulting signal was very noisy. In particular, the signal contained high baseline drift and was extremely sensitive

to finger placement to the point that HR estimation did not seem feasible. Additionally, one of the components was rendered useless in an electrical mishap.

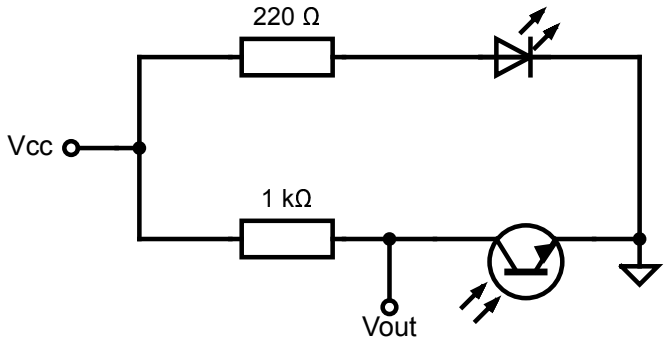

Figure 6. The circuit diagram of the initial system.

Instead of building a more complex circuit with hardware-based filtering and amplifiers, it was decided to utilize a sensor designed specifically for this purpose. The Sparkfun SEN-11574 was chosen for its ease of use and due to the fact that it features built-in signal amplification and noise cancellation [20], likely reducing the need for filtering in the software. Furthermore, being a reflective-mode sensor, it enables utilizing a wider variety of measurement sites.

The SEN-11574 features three header pins: one for supply voltage (Vdd), another for the signal output and a ground pin (GND) [20]. Since the input voltage range is 3 ... 5.5 V [20], the Arduino's +3.3 V supply (Vcc) is utilized directly for powering the sensor. The signal output is fed to one of the Arduino's analog inputs (A0), enabling the MKR1000 to sample the signal with its ADC. The system is shown in Figure 7.

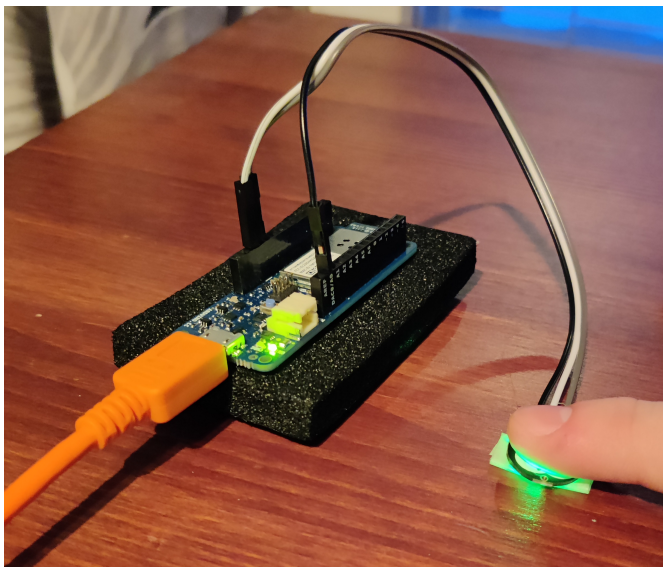

Figure 7. The system in action; a finger is placed on the SEN-11574.

3.1.2. Peak Detection and HR Estimation

Having decided on the hardware implementation, the software implementation for analyzing the data provided by the sensor and estimating the HR from it was considered next. The method used for estimating the HR is based on detecting the systolic peaks of the PPG signal. This method was chosen since the systolic peaks appear pronounced on the signal provided by the pulse sensor, likely making them easier to detect reliably than other parts of the waveform. Utilizing the sampling intervals of the detected peaks, one can obtain the PP intervals and use them to calculate the corresponding HR estimate.

For peak detection, a naive thresholding approach was first considered. In this approach, a peak would be detected when the signal crosses a certain predetermined threshold. However, this would likely result in detecting multiple occurrences of the same peak. Furthermore, a fixed threshold would not account for variations in the signal amplitude caused by differences in finger positioning and the individual characteristics discussed in Chapter 2.

As a more dynamic approach, a simplified version of the mountaineer's method discussed in Chapter 2 was considered. In this method, a peak is detected when the derivative of the signal changes sign from plus to minus. However, remaining noise in the signal could result in detecting the diastolic peak and false peaks.

In the method utilized in this thesis, parts from these two approaches were combined. The sample $x[n]$ obtained at a sampling instant $n = 1, 2, \ldots$ is considered a peak candidate if it holds that

$$x[n] < x[n-1], \tag{7}$$

$$x[n] \geq M, \tag{8}$$

where M is a fixed threshold. Condition 7 aims at detecting a falling edge of the signal while Condition 8 aims at preventing the detection of erroneous peaks. Furthermore, a cooldown period N_{cd} is utilized to prevent noise peaks and detecting the same peak multiple times. A sample $x[n]$ fulfilling Conditions 7 and 8 is only considered a peak if none of the last N_{cd} samples have been classified as a peak.

After detecting two successive peaks at sampling instants p_i and p_{i+1}, the corresponding PP interval is found to be

$$\mathrm{PPI_s} = p_{i+1} - p_i \tag{9}$$

samples. With a sampling rate of f_{s}, this corresponds to a time interval

$$\mathrm{PPI_t} = \frac{1}{f_{\mathrm{s}}}(p_{i+1} - p_i). \tag{10}$$

From this, it can be deduced that the corresponding heart rate estimate is

$$\mathrm{HR} = 60/\mathrm{PPI_t} = \frac{60 f_{\mathrm{s}}}{p_{i+1} - p_i}. \tag{11}$$

With the aim of increasing stability, averaging is utilized. The moving average of the last N PP intervals is calculated and this is then used for obtaining the HR estimate sent to the server and eventually shown to the user.

3.1.3. Software

The algorithm described in the previous subsection was implemented in the Arduino MKR1000. This approach was chosen since the algorithm is computationally lightweight and one avoids the delays that would be associated with running the algorithm in another device. Additionally, this enabled utilizing the Arduino's built-in LED for visualizing the heartbeat. However, performing the processing on another device could have been considered if a more computationally intensive algorithm would have been implemented. The implemented algorithm is depicted in Algorithm 1.

Algorithm 1. HR estimation

Input : Parameters f_s, N_cd, N and M
Output: Heart rate estimate HR every time a new peak is detected

$cd \leftarrow 0$
$count \leftarrow 0$
$previous \leftarrow 0$
$i \leftarrow 0$
PPI $\leftarrow$ `zeros`(N)
every $1/f_\mathrm{s}$ **do**
  $x \leftarrow$ `readSensor` ()
  $count \leftarrow count + 1$
  if $cd > 0$ **then**
    $cd \leftarrow cd - 1$
  end

  if $x \geq M$ **and** $x < previous$ **and** $cd = 0$ **then**
    PPI$[i] \leftarrow count$
    $i \leftarrow (i + 1)$ mod N
    $count \leftarrow 0$
    $cd \leftarrow N_\mathrm{cd}$
    HR $\leftarrow f_\mathrm{s} \cdot 60$ / (`sum` (**PPI**)$/N$)
  end
  $previous \leftarrow x$
end

During the software implementation, the values for parameters M, N_cd and N were chosen experimentally. It was noted that $M = 515$ resulted in reliable peak detection while still tolerating some variability in the signal amplitude. Additionally, the cooldown period $N_\mathrm{cd} = 20$ seemed sufficient for preventing the detection of erroneous peaks. The sampling rate $f_\mathrm{s} = 50$ Hz was initially chosen in order to enable easier visual inspection of the signal in the Arduino serial monitor. However, it also seemed to suffice for HR estimation due to the low frequency of the heart rate. Finally,

$N = 5$ seemed to provide a good balance between stability of the HR estimate and the time delay associated with averaging. An example result of the peak detection algorithm is presented in Figure 8.

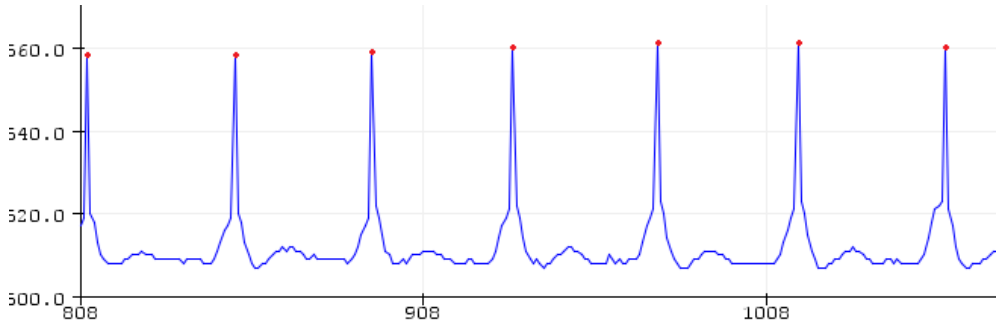

Figure 8. The detected systolic peaks shown accentuated on the Arduino serial monitor with parameters $M = 515$, $N_{\mathrm{cd}} = 20$ and $f_{\mathrm{s}} = 50$ Hz. The red dots marking the peaks were added for clarity.

3.2. Networking

In this section, the implementation of the network connectivity and the integration of the HR algorithm into this will be considered. The implementation was divided into two parts that will be considered separately (server and client).

3.2.1. Server

The central part of the system is the server that handles connection requests and sends the obtained HR data to connected clients. Initially, an approach that involved sending a message to clients every time a peak is detected was attempted. Even though this would have enabled the clients to display individual heartbeats in near real-time, it was noted that it was highly unreliable due to variation in network latency. Instead, it was decided to obtain the HR estimate on the Arduino and send this reading to clients if a change is detected.

Several alternatives for implementing the WebSocket server were considered. While the library WiFi101 can be used for utilizing WiFi on the MKR1000, another library is needed for server implementation. The library NINA-Websocket was utilized since it provides a simple WebSocket implementation sufficing for the needs of this thesis.

However, when integrating Algorithm 1 into the server implementation, issues were faced in setting the sampling rate. The server code in the main loop may take longer to execute and vary in execution time, affecting the timing and thus the sampling rate. In order to ensure that sampling is performed roughly at the frequency f_{s}, the sampling was moved into a hardware timer provided by the Arduino. Concretely, the Timer5 library was utilized since it provides higher level abstractions of the timers.

3.2.2. Client

Connected to the server in the local network, the client periodically receives data provided by the server. This HR data is then made available to the user and also presented graphically.

Different methods for implementing the client service were considered. Initially, it was planned to implement the client service as a mobile application. However, this would have limited the compatibility of the service to mobile devices running Android.

In order to ensure better interoperability and easier implementation, a web-based solution was decided on. The implemented client runs in the browser and utilizes the JavaScript WebSocket Application Programming Interface (API) to communicate with the server. Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS) are utilized for the user interface (UI) design.

Concretely, the implementation of the system is based on the callbacks provided by the WebSocket API. When new data becomes available or the state of the connection changes, JavaScript is utilized to modify the HTML Document Object Model (DOM) and thus display the information to the user.

As demonstrated in Figure 9, the UI of the system consists of a connection status indicator, a pulsating heart shape displaying the newest HR estimate and a graph visualizing the last 10 HR estimates. The graph was implemented with Chart.js, an open source JavaScript library for data visualization. Furthermore, the pulsating heart was accomplished with a CSS animation and JavaScript was utilized for retriggering the animation.

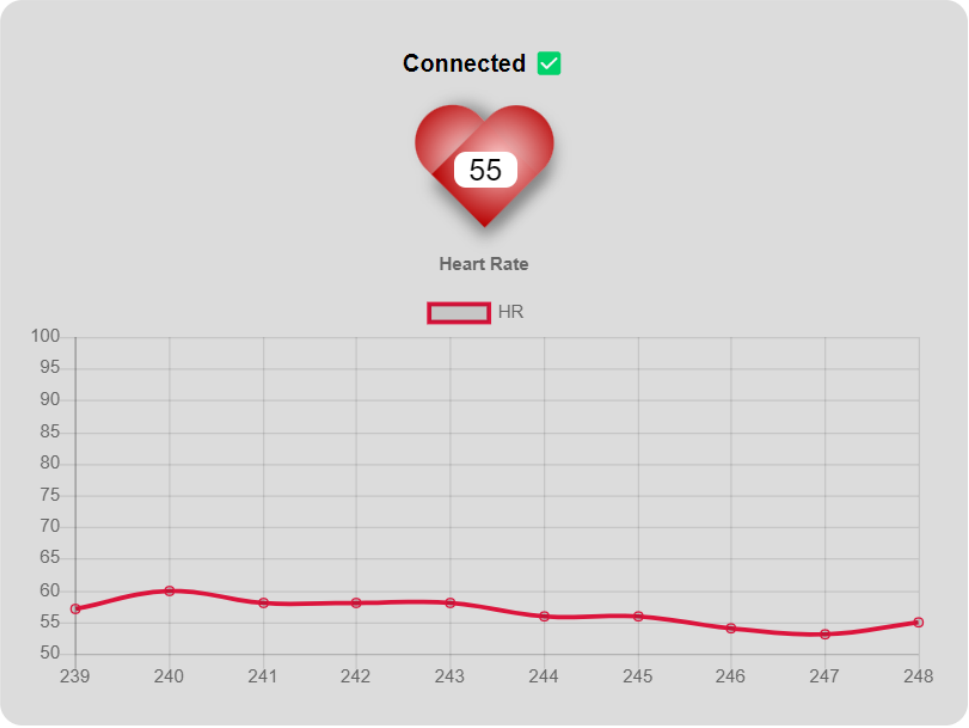

Figure 9. A screenshot of the browser client.

4. DISCUSSION

In the literature review section of this thesis, it was noted that PPG is a multidimensional and powerful tool for health monitoring with promising future prospects; it was surprising to note how many metrics can be reliably extracted from the seemingly simple waveform. However, many unresolved issues still remain regarding signal quality and reliability. Furthermore, integration of the measurement system into wearable devices poses various additional challenges for signal processing and noise removal. Since adequate signal quality forms the basis for extracting reliable information from the signal, these challenges must be addressed with careful preprocessing and algorithm design.

In the implementation part, a simple optical heart rate meter was implemented and it was shown that basic HR estimation can be accomplished without special hardware or complex algorithms. However, being a simple prototype, several challenges would need to be addressed in order to be able to integrate the implementation into a concrete wearable device with sufficient accuracy.

To begin with, several measures could be taken for improving the accuracy of the system. Instead of the approach based on a fixed threshold and the derivative, the mountaineer's method presented in Chapter 2 could have been utilized directly. Since it was noted that positioning of the finger had a large impact on the PPG amplitude, such an amplitude-independent method could have greatly improved the accuracy of the measurement. Furthermore, there is currently no means for detecting finger presence, resulting in erroneous HR data when a finger is not placed on the sensor.

On the other hand, owing to the various challenges associated with the implemented system, it was possible to demonstrate many of the PPG issues that were presented in Chapter 2 in practice. For instance, it was noted that testing the system with cold hands resulted in lower signal amplitude and distortion in the waveform, causing the peak detection algorithm to fail. Additionally, it was noted that the contact between the finger and the sensor had a large impact on the signal quality, demonstrating the challenges of a peak detection algorithm based on a fixed threshold value.

Concerning the IoT functionality of the system, the implementation is simple yet not very scalable. Since the resource-constrained MKR1000 functions as a server, performance problems can arise when the number of connected clients increases. As a more scalable solution, sending the HR data to a cloud server could be considered; for instance, the Arduino IoT Cloud could be utilized. This would also facilitate easier access to the data, enabling further distribution and analysis of the data. Machine learning approaches could then be utilized for deriving metrics from the gathered data as discussed in the introduction.

5. SUMMARY

The aim of this thesis was to introduce the reader to PPG, a relatively simple yet versatile optical technique for physiological monitoring. It was demonstrated that PPG is a well-suited technique especially for wearable devices and some of the main use cases and issues were discussed; in particular, methods for estimating HR from the PPG signal were presented in more detail. Additionally, the core principles and features of the WebSocket protocol and the Arduino ecosystem were presented, including the IoT-suited Arduino MKR1000.

In the latter part of this thesis, the theoretical part was applied by implementing an IoT-capable optical heart rate meter based on the Arduino MKR1000. It was shown that simple PPG-based heart rate measurement can be implemented with relatively simple hardware and a non-complex algorithm. However, it was also noted that since the signal is modulated by individual differences and typically contains noise originating from various sources, careful preprocessing and a more complex algorithm would be required for improving the reliability of the system.

6. REFERENCES

[1] Cleland J., Cox L., Papini G. & Sartor F. (2018) Methodological Shortcomings of Wrist-Worn Heart Rate Monitors Validations. Journal of Medical Internet Research 20(7). DOI: http://doi.org/10.2196/10108.

[2] Castaneda D., Esparza A., Ghamari M., Soltanpur C. & Nazeran H. (2018) A review on wearable photoplethysmography sensors and their potential future applications in health care. International Journal of Biosensors and Bioelectronics 4(4). DOI: http://doi.org/10.15406/ijbsbe.2018.04.00125.

[3] Kim S., Park J., Seok H. & Shin H. (2021) Photoplethysmogram Analysis and Applications: An Integrative Review. Frontiers in Physiology 12(7). DOI: http://doi.org/10.3389/fphys.2021.808451.

[4] Tamura T. (2019) Current progress of photoplethysmography and SPO2 for health monitoring. Biomedical Engineering Letters 9(1), pp. 21–36. DOI: http://doi.org/10.1007/s13534-019-00097-w.

[5] Lee C., Lee M. & Shin H. (2011) Relations between ac-dc components and optical path length in photoplethysmography. Journal of Biomedical Optics 16(7), p. 077012. DOI: http://doi.org/10.1117/1.3600769.

[6] Maeda Y., Sekine M., Tamura T. & Yoshida M. (2014) Wearable Photoplethysmographic Sensors—Past and Present. Electronics 3(2), pp. 282–302. DOI: http://doi.org/10.3390/electronics3020282.

[7] Elgendi M. (2012) On the Analysis of Fingertip Photoplethysmogram Signals. Current Cardiology Reviews 8(1), pp. 14–25. DOI: http://doi.org/10.2174/157340312801215782.

[8] Fernández J., Fullana J., Gaudric J., Ghigo A., Khelifa I., Lagrée P. & Politi M. (2016) The dicrotic notch analyzed by a numerical model. Computers in biology and medicine 72, pp. 54–64. DOI: http://doi.org/10.1016/j.compbiomed.2016.03.005.

[9] Ajmal, Boonya-ananta T., Branan K., Coté G., Fine J., McShane M., Ramella-Roman J. & Rodriguez A. (2021) Sources of Inaccuracy in Photoplethysmography for Continuous Cardiovascular Monitoring. Biosensors 11(4), p. 126. DOI: http://doi.org/10.3390/bios11040126.

[10] Pankiewicz B. & Wójcikowski M. (2020) Photoplethysmographic Time-Domain Heart Rate Measurement Algorithm for Resource-Constrained Wearable Devices and its Implementation. Sensors 20(6), p. 1783. DOI: http://doi.org/10.3390/s20061783.

[11] Argüello-Prada E. (2019) The mountaineer's method for peak detection in photoplethysmographic signals. Revista Facultad de Ingeniería 90, pp. 42–50. DOI: http://doi.org/10.17533/udea.redin.n90a06.

[12] Islam M., Zabir I., Ahamed T., Yasar T., Shahnaz C. & Fattah S. (2017) A Time-Frequency Domain Approach of Heart Rate Estimation From Photoplethysmographic (PPG) Signal. Biomedical Signal Processing and Control 36, pp. 146–154. DOI: http://doi.org/10.1016/j.bspc.2017.03.020.

[13] El-Hajj C. & Kyriacou P. (2021) Cuffless blood pressure estimation from PPG signals and its derivatives using deep learning models. Biomedical Signal Processing and Control 70. DOI: http://doi.org/10.1016/j.bspc.2021.102984.

[14] What is Arduino? URL: http://www.arduino.cc/en/Guide/Introduction. Arduino. Accessed 20.4.2022.

[15] Introduction to the Arduino Board. URL: http://www.arduino.cc/en/reference/board. Arduino. Accessed 27.4.2022.

[16] Using Functions in a Sketch. URL: http://docs.arduino.cc/learn/programming/functions. Arduino. Accessed 20.4.2022.

[17] Arduino MKR 1000 WiFi. URL: http://docs.arduino.cc/hardware/mkr-1000-wifi. Arduino. Accessed 9.4.2022.

[18] An intro to the Arduino IoT Cloud. URL: http://docs.arduino.cc/learn/starting-guide/arduino-iot-cloud. Arduino. Accessed 20.4.2022.

[19] The WebSocket Protocol. URL: http://datatracker.ietf.org/doc/html/rfc6455. Internet Engineering Task Force (IETF). Accessed 9.4.2022.

[20] Pulse Sensor SEN-11574. URL: http://www.sparkfun.com/products/11574. SparkFun Electronics. Accessed 4.2.2022.