ingenieur wissenschaften htw saar

UNIVERSITÄT DES SAARLANDES htw saar

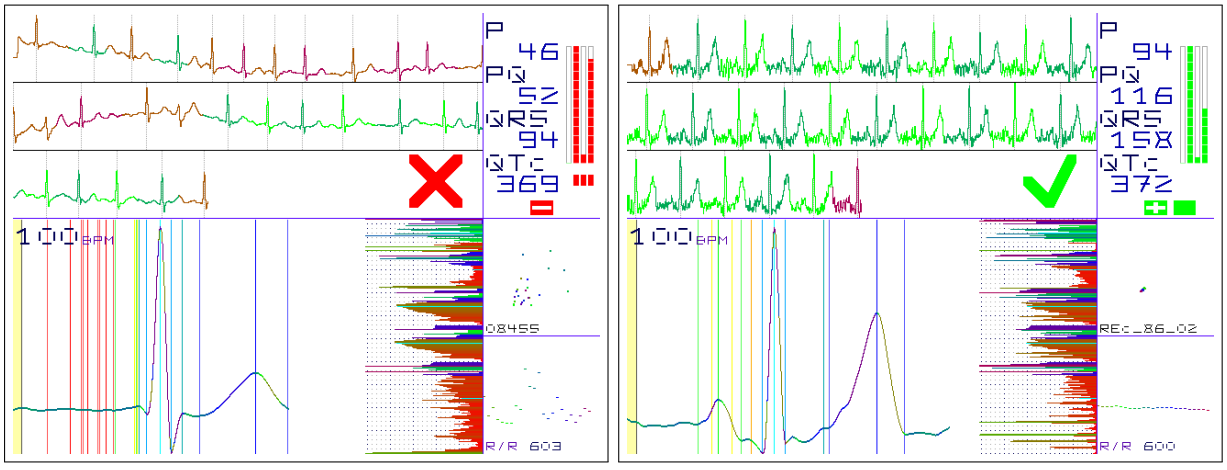
Systems Neuroscience & Neurotechnology Unit

Master's thesis

Development of an embedded device for real-time detection of atrial fibrillation and atrial flutter in single-channel ECG, using optimised classification based on a large training corpus

Eric Auer

Date: December 27, 2020



htw saar, Saarbrücken
Master Neural Engineering

Supervisors: Prof. Dr. Oliver Scholz
Prof. Dr. rer. nat. Dr. rer. med. habil. Daniel J. Strauss

« On ne voit bien qu'avec le cœur. L'essentiel est invisible pour les yeux. »
– Antoine de Saint-Exupéry, Le Petit Prince

To the millions of people around the world suffering from atrial fibrillation.

Abstract

Atrial fibrillation (A-Fib) and atrial flutter are widespread medical conditions of the heart. Loss of coordination between atrial and ventricular activities affects the smooth circulation of blood, causing an increased risk of blood clotting, which in turn elevates risk of pulmonary embolisms and cerebral infarction. However, the condition is not necessarily noticed by patients, for example through palpitations or tachycardia.

A custom embedded device developed for this thesis helps people to evaluate whether they are experiencing atrial fibrillation at a specific moment. The device measures single-channel ECG for less than one minute and instantly classifies it as either A-Fib, normal sinus rhythm (NSR) or undecided (low measurement quality or atypical ECG).

Building on an earlier proof of concept project work by the author, this thesis presents a fully integrated, custom device, using an advanced classification algorithm trained on thousands of short, annotated ECG fragments from the PTB-XL corpus. The algorithm uses morphological analysis of the averaged ECG shape, properties of the R/R interval distribution and spectral analysis of the ECG to create a feature vector used for classification. Analysis and raw ECG data can be transferred via Bluetooth at the user's discretion.

Zusammenfassung

Vorhofflimmern (und Vorhofflattern) sind häufig vorkommende Abweichungen in der Aktivität des Herzens. Durch die Verschlechterung der Koordination zwischen Vorhöfen und Herzkammern wird der Blutfluss beeinträchtigt und das Risiko der Bildung von Blutgerinseln steigt. Diese wiederum können gefährliche Lungenembolien oder Schlaganfälle verursachen. Zugleich sind sich Betroffene nicht notwendigerweise durch Symptome wie Herzklopfen oder Herzrasen ihres Vorhofflimmerns (A-Fib) bewusst.

Ein speziell für diese Masterarbeit entwickeltes Gerät hilft Menschen, zu überprüfen, ob sie zu einem bestimmten Zeitpunkt unter Vorhofflimmern leiden. Das Gerät misst für weniger als eine Minute das EKG der Nutzer (einkanalig) und zeigt sofort eine Klassifikation als entweder A-Fib, normaler Sinusrhythmus (NSR) oder weder noch (untypisches EKG oder zu schlechte Qualität der Messung) an.

Aufbauend auf einer vom Autor als Projektarbeit durchgeführten Machbarkeitsstudie, präsentiert diese Masterarbeit ein komplett eigenständig nutzbares Gerät, welches einen fortgeschrittenen Klassifikationsalgorithmus für die Aufgabe verwendet. Dieser wurde an Tausenden kurzer, annotierter EKG Fragmente aus dem PTB-XL Korpus trainiert und benutzt eine morphologische Analyse des gemittelten EKG Verlaufs, Eigenschaften der Verteilung der R/R Intervalle und spektrale Analysen des EKG, um einen Eigenschaftsvektor zur Klassifikation zu erzeugen. Analyse und Rohdaten des EKG können von den Benutzern optional zur Übertragung per Bluetooth freigegeben werden.

Declaration of authorship

I hereby certify that this master's thesis has been composed by me and is based on my own work, unless stated otherwise. No other person's work has been used without due acknowledgement in this thesis. All references and verbatim extracts have been quoted, and all sources of information, including graphs and data sets, have been specifically acknowledged.[1]

Eric Auer

Saarbrücken, December 27, 2020

[1] Source: `https://www.phil-fak.uni-duesseldorf.de/uploads/media/Declaration_of_Authorship.pdf`

[2] License based on Manuel Kohl's 2015 thesis template, available under the same Creative Commons license

Contents

1 Introduction **1**
1.1 Building on previous work . . . 1
1.2 Existing solutions . . . 2
1.3 Achievements in this thesis work . . . 3

2 Theoretical background **5**
2.1 Common ECG features in detection of atrial fibrillation . . . 5
2.2 Spectrum and phase . . . 7
2.3 Filtering for fast signal processing . . . 8
2.4 Principal component analysis and linear discriminant analysis . . . 9
2.5 Hardware choices . . . 12

3 Hardware design **15**
3.1 ECG analogue front-end . . . 15
3.2 User safety . . . 16
3.3 Central microcontroller / low power standby . . . 18
3.4 Space considerations . . . 19
3.5 Power source . . . 20
3.6 Power supply . . . 21
3.7 USB interface . . . 23
3.8 Display . . . 23
3.9 RGB LED . . . 25
3.10 Audio output . . . 25
3.11 Overall power consumption . . . 26
3.12 Layout considerations . . . 28
3.13 ECG electrodes and device housing . . . 30
3.14 Prototype assembly and errata . . . 33
3.15 Third generation device . . . 35
3.16 ESP32 pin assignments . . . 37

4 Software design **39**
4.1 Embedded system usage and training workflow . . . 39
4.2 ECG metrics and classification workflow . . . 41
4.3 Global classification . . . 44
4.4 Calibration and recording . . . 44
4.5 Custom digital filtering . . . 46
4.6 PTB-XL corpus excerpts . . . 47
4.7 Algorithm feature landscape . . . 50
4.8 Rule-based feature processing . . . 51
4.9 Principal component analysis . . . 53
4.10 Linear discriminant analysis . . . 54
4.11 Graphical user interface and result presentation . . . 56
4.12 Additional interactive user interface modalities . . . 58
4.13 Bluetooth access . . . 59

5 System evaluation **63**
5.1 Hardware evaluation . . . 63
5.2 Cost considerations . . . 65
5.3 Matching embedded hardware and efficient software . . . 66
5.4 Algorithm parameter tuning . . . 69
5.5 Classification performance . . . 69

5.6 Leave-1-out performance with PTB-XL 73
5.7 Performance with synthetic ECG 74

6 Conclusions **77**

List of tables **XIII**

List of figures **XIV**

References **XVI**

List of websites mentioned **XVIII**

A Appendix: hardware measurements, bill of materials, schematics **XXIII**
A.1 Signal quality and current measurements XXIII
A.2 Prototype bill of materials XXIV
A.3 Prototype circuit diagram XXIV
A.4 Third generation circuit diagram XXIV

B Appendix: software details **XXXI**
B.1 Additional classification plots XXXI
B.2 Digital filter details XXXII

C Appendix: software evaluation details **XXXVII**
C.1 Problematic analysis examples from old corpora XXXVII
C.2 PTB-XL A-Fib recordings classified as NSR XXXIX
C.3 PTB-XL NSR recordings classified as A-Fib XLI

D Appendix: Licensing of hardware designs and software **XLIX**
D.1 Software licenses XLIX
D.2 Hardware licenses L

List of abbreviations

AAA	Indicates battery size Micro (circa. $10 \times 44\,\mathrm{mm}$)
AC	Alternating Current
ADC	Analogue Digital Converter, to read analogue signals for digital processing
AFE	Analogue Front End, here an ADC with its analogue input circuits in one chip
BGA	Ball Grid Array, a chip package family with balls
BIOS	Basic Input Output System, fixed or firmware software module
BSD	Berkeley Software Distribution, a clone of the Unix OS
BT	Bluetooth, a wireless communication standard
CMRR	Common Mode Rejection Ratio
CMOS	Complementary Metal Oxide Semiconductor, digital circuit family
CPU	Central Processing Unit, computer processor
DAC	Digital Analogue Converter, to output analogue signals
DC	Direct Current, non-alternating
DIY	Do It Yourself, for example in context of home improvement
DOR	Diagnostic Odds Ratio, a statistical measure
DSP	Digital Signal Processing
ECG	Electrocardiography, measuring electrical activity of the heart
EMI	Electro-Magnetic Interference
ENIG	Electroless Nickel Immersion Gold, a PCB surface finnish
ENOB	Effective Number of Bits, describes ADC performance
ESD	Electro-Static Discharge, a risk for electronic circuits
FET	Field Effect Transistor, controlled by voltage, not current
FIFO	First In First Out, a buffer which can queue data
FIR	Finite Impulse Response, a type of digital filter in DSP
FHT	Fast Hartley Transform, a real-valued alternative to FFT
FFT	Fast Fourier Transform, transforms into the frequency domain
GPIO	General Purpose Input Output
HPF	High Pass Filter
HRV	Heart Rate Variability, natural variation of heartbeat durations
IDE	Integrated Development Environment (editor, compiler invocation etc.)
IIR	Infinite Impulse Response, another type of digital filter
IMU	Inertial Measurement Unit, accelerometer and gyroscope
I/O	Input / Output
IPS	In-Plane Switching, a LCD technology

LAN Local Area Network

LBBB Left Bundle Branch Block, a heart condition

LCD Liquid Crystal Display

LDA Linear Discriminant Analysis, a dimensionality reduction method

LDO Low Drop Out, a type of linear voltage regulator

LED Light Emitting Diode

LPF Low Pass Filter

MCC Matthews Correlation Coefficient, a statistical measure

MIT Massachusetts Institute of Technology, a university

MISO Master In Slave Out, (SPI) serial data line in said direction

MLCC Multi Layer Ceramic Capacitor: non-polarised, less effective under DC bias

MOSI Master Out Slave In, (SPI) serial data line in said direction

NPN Negative Positive Negative, type of bipolar transistor, controlled by current

NSR Normal Sinus Rhythm, regular heart rhythm

OAE Otoacoustic Emissions, weak sounds generated **by** the ear

OLED Organic LED, a display technology

OS Operating System, software essential for using a computer

OTA Over The Air, here: ability to wirelessly update firmware

PAC Premature Atrial Contraction, an abnormal heart beat

PCB Printed Circuit Board

PC Personal Computer

PCA Principal Component Analysis, a dimensionality reduction method

PCX PiCture eXchange, a classic graphics file format

PDF Portable Document Format, a standardised file format

PNC Premature Nodal Contraction, an abnormal heart beat

PNG Portable Network Graphics, a graphics file format

PNP Positive Negative Positive, type of bipolar transistor, controlled by current

PVC Premature Ventricular Contraction, an abnormal heart beat

QRS Sequence of 3 central spikes, Q, R and S, in the human ECG

RAM Random Access Memory, used as computer working memory

RBBB Right Bundle Branch Block, a heart condition

RGB Red Green Blue, common colour representation scheme

RISC Reduced Instruction Set Computing, a processor architecture

RLE Run Length Encoding, a simple data compression method

RMS Root Mean Square, a measure of magnitude

ROM	Read Only Memory, type of data storage
SDIO	Secure Digital Input Output, a bus system
SMD	Surface Mount Device, a component packaging paradigm
SNR	Signal to Noise Ratio
SoC	System on a Chip, containing CPU, RAM, I/O all on one chip
SPI	Serial Peripheral Interconnect, a serial interface standard
SVM	Support Vector Machine, a machine learning method
TGA	Truevision TGA (TARGA), a graphics file format
TFT	Thin-Film Transistor, a LCD technology
TRS	Tip Ring Sleeve, a family of plug / socket connectors
UART	Universal Asynchronous Receiver Transmitter, serial interface controller
UNIX	Name of an OS, no acronym. Linux belongs to the UNIX family
USB	Universal Serial Bus, a serial interface family
VAC	Volt AC, see AC
VSON	Very Small Outline No Leads, a chip package family
WCSP	Wafer Chip Scale Package, a chip package family
WLAN	Wireless LAN, also called WiFi

1 Introduction

1.1 Building on previous work

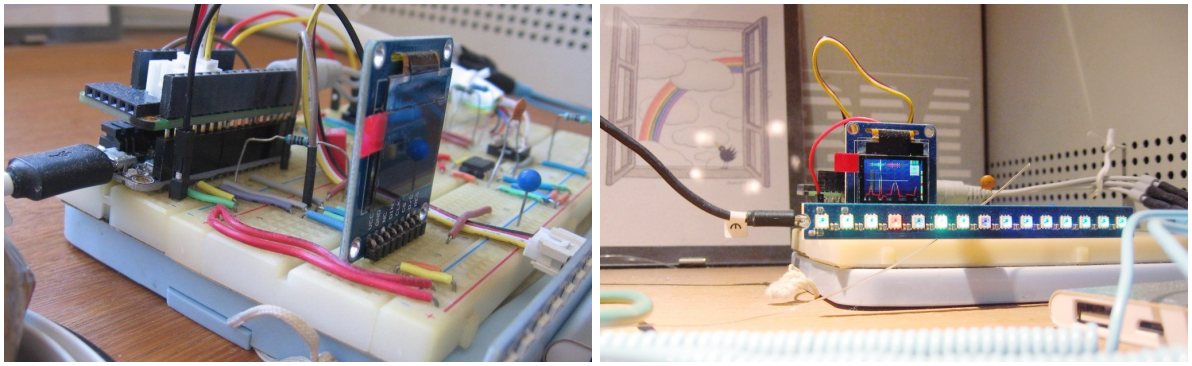


Figure 1.1: Left: Original ecghelper hardware: Stacked off-the-shelf modules, OLED, NeoPixels, for use with USB power bank and wristband electrodes. Right: Test run with a patient simulator

After having demonstrated the approach of detecting atrial fibrillation with a small, easy to use ECG device in an earlier project work [Auer 2020], this Master's thesis aims to evolve the original idea into a custom product prototype. While the original project used off-the-shelf modules, this thesis presents a compact, fully integrated device with new features. And while the original software provides a proof of concept, showing that a simple single channel ECG recording has sufficient quality to extract key metrics from an averaged ECG shape, the new version of the firmware written for this thesis is trained on a large corpus of short ECG recordings with known diagnosis, such as Normal Sinus Rhythm (NSR) or Atrial Fibrillation (A-Fib). This allows to improve the accuracy of the ECG classification significantly. For an ECG and physiology introduction, see [Menche 2007], pages 231ff (heart cycle) and 236ff (ECG).

As discussed in [Auer 2020] atrial fibrillation is a widespread medical condition with an inherent risk of triggering severe pulmonary embolism and cerebral infarction events. On the other hand, atrial fibrillation itself is not necessarily noticed by the patients. It may be noticed when it causes irregular heart rhythms, palpitations or tachycardia, but the underlying pathology can also stay without symptoms. In atrial fibrillation, the atria of the heart fail to pump blood in concordance with the larger ventricles, but as the latter still create a normal blood flow, patients may not have warning signs of their fibrillation events. However, as coordinated pumping from body to atria to ventricles to body is impeded, there is an increased risk of forming blood clots. When those leave the heart, they can end up blocking blood vessels in the heart, causing pulmonary embolism, or in the brain, causing cerebral infarctions ([Menche 2007], page 239).

The project report outlined an approach for detecting atrial fibrillation by measuring the electrical activity of the heart in a single-channel ECG. The user holds dry electrodes with both hands, thus providing the connected device with an Einthoven I signal. No special training for electrode placement is required and all hardware is fully reusable. As shown in the report, signal quality is sufficient to compute the average ECG shape for and extract metrics, which can subsequently be used to decide whether or not an atrial fibrillation event is ongoing. It is conceivable to use properties of the R/R intervals between beats and spectral properties of the recording to aid the classification, but the project software has a focus on the shape metrics. R/R properties are deliberately only shown, but not judged. Also, the only spectral metric is a weighted sum of the area under a global periodogram. For example [Moody 1983] does study A-Fib detection via R/R intervals.

1.2 Existing solutions

As long as data protection is not a requirement, widely available portable devices can be used to assess the health status of the user. For example, Science Daily reports[3] that the built-in camera and light of a smartphone can be sufficient to implement a simple photoplethysmography measurement which then (for example using R/R properties) can be used to detect atrial fibrillation.

Another smartphone-based method is presented in [Lahdenoja 2018], where the built-in accelerometer and gyroscope are used to detect atrial fibrillation, achieving an accuracy of 97.4% (sensitivity 93.8%, specificity 100%). As the abstract of [Perales 2020] announces, a systematic review of 43 studies found an ECG (single-lead, AliveCor device for use with smartphones and tablets) based 'sensitivity and specificity between 66.7% and 98.5% and 99.4% and 99.0%, respectively' (sic!), while photoplethysmographic measurements detected A-Fib with a sensitivity of 85.0% to 100% and a specificity of 93.5% to 99.0%. A similar, freely available review is [Duarte 2020].

According to a Cleveland Clinic study mentioned on their website on February 25, 2020[4], including ECG data can boost A-Fib detection rates using a smartwatch. Here, the user has to touch a button on the watch with the other hand to measure an Einthoven lead I ECG. Analysis of an ECG downloaded from the watch as PDF (portable document format) allowed detection of A-Fib 98% of the time, while the built-in app only detected 41% of the A-Fib. Other health wearables like the Fitbit now also have gotten approval to detect A-Fib based on ECG measurements[5].

An embedded solution mentioned in a May 5, 2020 item on Science Daily[6] relies on processing ECG data on a remote server. A necklace-shaped ECG is used to detect A-Fib. The user touches the necklace with both hands, or with palm and chest, to record 30 seconds of ECG data, which are then uploaded to a server for analysis.

One interesting ECG processing approach is announced in the abstract of [Hau 2020]. Stationary Wavelet Transform is applied to filtered ECG to produce features in the time-frequency domain, with a focus on Power Spectral Density and log energy entropy between 4 and 9 Hz, which are then assessed by an artificial neural network to classify the ECG as A-Fib or normal. The algorithm is designed to work on an embedded system in real-time speed. For training, the [Moody 1983] corpus of 23 long ($\approx$ 10 hours) ECG recordings from A-Fib patients has been used, which also has been one of the corpora used in [Auer 2020], but only in short excerpts.

Apart from data protection issues, one incentive for embedded ECG analysis can be to reduce the required upload bandwidth while monitoring patients for A-Fib. In [AlMusallam 2019], A-Fib is detected with 100% sensitivity and 96% specificity based on sending only R/R interval data and P-wave presence features to a remote server, drastically reducing the amount of uploaded data and thereby also energy consumption of the mobile device.

So, while availability of wearable ECG can be assumed and expected to improve A-Fib classification potential compared to purely R/R based approaches such as photoplethysmography, the added value of fully embedded processing is not getting as much attention as it could get.

Embedded systems do have sufficient computing power even when working on a limited energy budget and both the energy saving aspect and the protection of sensitive patient data make it worthwhile to explore the possibilities of a stand-alone embedded solution further. A device and

[3] `https://www.sciencedaily.com/releases/2018/08/180825081735.htm` August 25, 2018: A smartphone application can help in screening for atrial fibrillation

[4] `https://newsroom.clevelandclinic.org/2020/02/25/cleveland-clinic-studies-accuracy-of-apple-watch-4-for-atrial-fibrillation-detection/` Cleveland Clinic Studies Accuracy of Apple Watch 4 for Atrial Fibrillation Detection

[5] `https://www.BusinessWire.com/` September 14, 2020: Fitbit Receives Regulatory Clearance in Both the United States and Europe for ECG App to Identify Atrial Fibrillation (AFib)

[6] `https://www.sciencedaily.com/releases/2020/05/200505072158.htm` Novel necklace detects abnormal heart rhythm

algorithm to achieve this goal are presented in this thesis. While the user can optionally share data with other devices, the embedded system does not require that connection to perform recording, analysis and classification to detect A-Fib in single-channel ECG.

1.3 Achievements in this thesis work

The goals for this thesis have been described in a proposal in July and can be grouped into solving two main problems of the project:

First, the old system just combined pre-made modules on a breadboard, which had shortcomings in usability, flexibility and power consumption. There only was a tiny display and idle power consumption was high, while the used LED array was a lot larger and brighter than necessary.

Second, the old system analysed the recorded data by extracting easily understood metrics such as the amplitude of specific waves in the ECG and comparing them to expected values from literature. While there have been parameter adjustments based on test data, only a small corpus of a few hundred recordings of varying length was available. Often, the recordings lacked metadata about whether they were examples of atrial fibrillation, normal sinus rhythm or something else. For example, a set of recordings meant for biometric identification of people using their ECG is expected to contain few, another meant to show robust ECG beat detection in low quality telehealth ECG is expected to contain more cases of atrial fibrillation. Only dozens of known atrial fibrillation recordings were part of the project corpora.

To solve the first problem, I have developed a fully integrated new version of the `ecghelper` device with enhanced features. As compact, battery-powered device with a larger screen and electrode surfaces attached to the housing, it can be seen as a prototype of an actual product which patients can use at home to get an indication of their probable atrial fibrillation status. Added audio output capabilities improve usability and make the device usable for visually impaired people.

Should the user want to keep records of their measurements, or share them with medical experts, they can trigger a Bluetooth transfer mode to send analysis results and raw data to another suitable device, such as a computer or smartphone. At the same time, privacy risks are limited by keeping the user in control about when to activate the bluetooth connectivity.

This is a new software feature, as both old and new hardware are built around ESP32 controllers with built-in Bluetooth (see section 3.3). Even the new software could still be compiled for the old hardware, although the user interface will be very limited there. This makes it possible for others to continue development without having to build the custom new hardware first.

The current software, but also the circuit diagram and layout for the current hardware, will be made available as open source and open hardware after the Master's degree has been achieved. This will be beneficial both for the scientific audience and for maker communities, who can port the software to existing platforms such as the HeartyPatch[7] or develop yet better versions of the hardware and software. As the original inventor, I can also continue on the path towards production and sale of the third generation device, or offer consultancy services to others doing so.

Next to improved usability, another main design goal of the new hardware has been energy efficient operation. Careful design made it possible to create a battery powered device which can stay in standby for months, waking up as soon as the user touches the electrodes. It turns out that wireless transfers cause high power consumption, so the software keeps those short and optional. Efficient algorithms, low CPU clocks and careful use of output devices (buzzer, LED, display) makes it possible to perform many, in particular offline, measurement cycles without having to replace or recharge batteries.

[7] `https://heartypatch.protocentral.com/` and `https://www.crowdsupply.com/protocentral/heartypatch` – no CE marking, not available in Europe

Building the prototype predictably brought up further areas for improvements. Therefore, this thesis also presents a design for a third version of the device. This version has has a lower footprint, again significantly lower standby power consumption and additional sensors to make the device more pleasant and versatile to use. The upgraded power supply will also allow it to run longer with rechargeable batteries, thanks to a wider input voltage range. Also, the new design features battery monitoring. No prototype of the third version has been built yet.

While all versions have been designed with usability, reliability and safety for an actual product in mind, there have been no tests yet with current patients or healthy controls apart from the author himself and the occasional curious onlooker. Ongoing global health issues made it seem inappropriate to conduct experiments with groups of, in part vulnerable, people for this thesis.

This brings us to the second topic of improvements. To increase accuracy of the ECG analysis outcome over the heuristic approach of the project, this thesis makes use of the large, publicly available PTB-XL ECG corpus [Wagner 2020a, Wagner 2020b]. The corpus consists of almost 22000 annotated short, multi-channel ECG recordings. Extracting only the Einthoven lead I data for more than 6000 known normal sinus rhythm and more than 1500 known atrial fibrillation or atrial flutter ECG snippets provided more than enough training data for a basic, yet effective machine-learning approach to classify recordings, which is presented in section 4. Section 2 briefly introduces some of the underlying concepts and algorithms.

Instead of focusing on a set of ECG shape metrics, the current software uses ECG shape metrics, spectral properties of the overal recording, spectral and phase properties of beat-centred snippets and simple statistics about the R/R intervals to create a feature vector – a list of features describing a given ECG recording in a well-defined way.

Each recording (10 seconds in PTB-XL, up to 30 when using the `ecghelper` device interactively) is thereby projected onto a point in a high dimensionality space, using the feature vector as coordinate vector. Subsequently, zones in that space are marked as specific to the different outcomes. Normal sinus rhythm, atrial fibrillation or, importantly, recordings which cannot be put in either category.

A second classifier uses the corpus data to create a projection of features for any recording onto one single dimension, which can be seen as a NSR to A-Fib spectrum. Again, it is possible to classify recordings as ambiguous, this time by checking how close to the middle of the spectrum they are. Thirdly, some core ECG shape, in particular P-wave related, heuristics have been kept around as a third source of judgements. Combining all three methods allows the software to make an informed, plausible decision instead of blindly incorporating knowledge about **which** recordings of the training data snippets have which gold standard classification. Such overfitting would make the classifier less useful for novel data, while any `ecghelper` recording is new by definition.

To show that the software makes a decision based on **what** physiological state a given ECG snippet represents, it has been used to classify the recordings from the old project. None of them are part of the training data and their quality is often lower than the quality of the PTB-XL recordings. In the same way, recordings made with the `ecghelper` device and dry electrodes can be expected to have lower quality than those in PTB-XL. So applying the trained classifier on lower quality data from a completely different set can be seen as substitute for running a series of experiments with patients and controls using the `ecghelper` hardware while the latter is not an option at the moment.

The goals set out in this introduction have been achieved in the course of the thesis, which will first present some theoretical basics and the hardware and software design in the three following sections. Next, the performance of both hardware and software will be evaluated. The conclusions give an overview of the findings.

In the appendix, further details on hardware and software can be found, as well as additional measurements and result details from using the software to classify ECG recordings from the available ECG corpora.

2 Theoretical background

This section provides a quick overview of some of the known approaches, algorithms and established design choices used in the development of the `ecghelper2` algorithm. In particular, some mathematical concepts will be discussed in short introductions. References for further reading are given for those looking for more comprehensive presentations of selected topics.

As often, much of the actual implementation depends more on already available applications of the used concepts, rather on their theoretical basics. Signal processing and related lecture notes are helpful to understand **why** things work as they do, while just plugging known formulae as a 'black box' will often result in a working system, too. Knowing why and how a formula works will, however, enable the user to use it more effectively and avoid certain pitfalls and side effects.

2.1 Common ECG features in detection of atrial fibrillation

Over time, a variety of methods has been applied to detect atrial fibrillation (A-Fib) in ECG signals. For the purpose of this thesis, the focus will be on extracting features of medium complexity and on classification based on feature vectors and training data from a large corpus. Trained and empirically selected parameters in the embedded software will enable the `ecghelper2` to instantly classify a recording as A-Fib or Normal Sinus Rhythm (NSR) after a fast feature extraction step.

In [AlMusallam 2019], the authors suggest that combining p-wave absence and R/R irregularities can provide an efficient, low complexity to extract feature combination for detecting atrial fibrillation in low-power systems. They also state that this approach was not previously addressed in the literature and take care to select a low computational complexity QRS detection method to keep energy consumption even lower. As minimal complexity R/R irregularity feature, the difference between minimum and maximum R/R interval during a 10 s recording is used.

Note that the low-power ECG hardware used in this thesis already has QRS detection built-in (a modified Pan-Tompkins [Pan 1985] design) which would be an option to reduce computational load for the main embedded controller. However, this thesis chooses to do QRS detection in software, which gives more control and better time resolution. As the datasheet of the used MAX30003 chip explains, the temporal R/R (QRS) detector resolution of the hardware detection is lower than the available ECG sampling rates[8].

Interestingly, similar to the earlier proof of concept system in [Auer 2020], [AlMusallam 2019] use data from the MIT-BIH Arrhythmia and A-Fib Databases for training and evaluation, from which they extract 268 short (10 s) ECG excerpts containing A-Fib according to metadata.

Apart from using R/R based features, their algorithm also checks whether P-wave peaks can be found in the second half of each R/R interval, based on the observation that P-waves are likely to be absent during A-Fib. During A-Fib, they write, chaotic atrial activity emits irregular, low-amplitude fibrillatory 'f-waves' instead. The peaks are detected using a 5-tap derivative filter and sign changes from positive to negative (detection of local maxima) above a pre-set threshold. Not only peak presence, but also peak timing relative to the R peak is used as feature.

The goal of the authors could be described as getting maximum accuracy from the smallest possible number of CPU cycles. This is not the goal of the algorithm developed for this thesis, as a comfortable amount of processing power is available even on the embedded hardware design created for the thesis. Also, the users are assumed to measure their ECG only when they want to check their A-Fib status, not continuously.

[8] `https://datasheets.maximintegrated.com/en/ds/MAX30003.pdf`

So as long as the chosen algorithm is able to make a classification in nearly real-time based on less than a minute of recorded ECG data, with the available CPU speed, it can make use of additional ECG features to improve classification accuracy.

This differs from the scenario described in [Rincón 2012] where R/R variance and P-wave quality (correlation to a trained model) based A-Fib detection is implemented on the microcontroller of a Shimmer wearable wireless ECG sensor. Normally used only to coordinate data transfer between ECG analogue front-end and wireless peer, it has little memory and CPU power, so a low complexity algorithm has to be used to reach the goal of running the A-Fib detection directly on the sensor.

At higher CPU load but lower wireless bandwidth, transferring only the classification results, they conclude that sensor energy consumption is similar to that of merely streaming the raw, uncompressed ECG. Of course, energy will also be saved by the computer receiving the data, as classification has already taken place on the sensor.

For this thesis, wireless data transfer will be entirely optional. The embedded system will acquire the ECG recording, analyse and classify it and present the results using a built-in display and other user interface components for direct audiovisual feedback. Only if chosen by the user, data can then be transferred to other devices such as computers or smartphones using a wireless Bluetooth connection. This means that both data protection risks and energy consumption involved with wireless connectivity no longer apply as long as the users are satisfied with inspecting the results directly on the device. The hardware would also support WiFi transfers, but those are not used as they involve higher energy consumption (and radiation) and higher software complexity.

The dynamics of changing R/R intervals (the time between R peaks of adjacent heart beats) can be captured using a variety of statistical and other properties, a field known as Heart Rate Variability. While HRV based on long ECG measurements can capture for example slow, periodic changes, metrics based on short measurements can already be useful for A-Fib detection, as the example of [Rincón 2012] demonstrates. In this thesis, metrics inspired by [Nabian 2017] are being used. They can be computed directly from the R/R data, without having to resort to frequency transforms which would be of limited use for short recordings. In particular, the corpus data used in this thesis consists of short (10 s) recordings, each only containing the ECG of a small number of heartbeats, giving little room for analysis of periodic changes.

In particular in human analysis of ECG, it is common to interpret the morphology directly, not the spectrum or R/R statistics. Some examples can be found in [Menche 2007] (section 14.5) and in [Kramme 2011] (chapter 10). As the latter explains, amplitudes, timing, area, steepness and other morphological properties can be extracted by morphological analysis and used for diagnosis.

This book chapter also mentions that HRV can be based on a recording of 300 heartbeats. This is beyond the intended duration of ECG measurement sessions of `ecghelper2` and far beyond the length of the ECG excerpts available in most of the ECG corpora used here, so again, **spectral HRV** measures will not be used in this thesis.

Instead of processing ECG morphology of all individual heartbeats, the algorithm presented in this thesis will instead first create one averaged beat ECG from a selection of multiple, similar beats and then investigate morphological properties of that. Averaging can be used as a way to improve signal to noise ratio for periodical signals where noise is not phase-locked to the actual signal. The method is for example also used in the analysis of otoacoustic emissions (OAE), which are weak and therefore hard to measure directly.

Section 15.3 of [Kramme 2011] explains the principle, which improves signal to noise ratio by $\sqrt{N}$ by averaging over N events, phase-aligned to the signal – in OAE, this can be stimulus responses in fixed temporal relation to a repeated stimulus. Similar to OAE processing, the `ecghelper2` algorithm excludes deviant beats from averaging, a concept already used in [Auer 2020]. Note that when the signal itself has irregular properties, those will be attenuated by averaging. This actually

helps A-Fib detection. 'Proper' P-waves are expected to be in a more stable phase relation to the QRS complex than the more irregular activity in A-Fib, even though the latter may still look similar to P-waves when looking at individual beats.

2.2 Spectrum and phase

The computational power available in the hardware design of this thesis is easily sufficient for including some frequency domain features in the algorithm. As for example the abstract of [Hau 2020] shows, it is not uncommon to use spectral features in detection of atrial fibrillation. In particular, they expect A-Fib to involve oscillations at higher frequencies than the heart rate (in their case, 4 to 9 Hz) and with less fixed temporal relations relative to the normal heart cycle.

In other words, the phase relation between higher frequency components and the main heart cycle (for example 1 Hz at 60 beats per minute) is likely to vary over time. This makes the phase stability a possible feature as input for further classification.

To estimate phase stability, a complex spectrum or the sine and cosine components of a spectrum can be used. While a magnitude spectrum only shows which frequencies are present and how strong, the complex or sine and cosine spectrum also shows the phase angle. By summing up the complex spectral values for one frequency over several events (here: heart beats), the magnitude ('length') of the sum will be less than the sum of individual magnitudes unless all angles are the same. Signal phase angles correspond to vector directions of the spectrum in the complex (or sine and cosine) plane.

As phases vary more, the magnitude of the vector sum will decrease. So the ratio between magnitude of the sum of vectors and of sum of magnitudes of vectors will depend on how constant or stable phases are across beats. Low stability can either mean that phase is shifting in a regular pattern over time or that phases for the specific frequency are randomly changing from beat to beat. Both can be symptoms of a lack of synchronisation between global cardiac activity and the signal at that frequency. For example, fast, uncoordinated atrial activity, either regular or chaotic, but not time-locked to the heart cycle, should be visible as spectral components with low phase stability.

This thesis uses an implementation of Fast Hartley Transform (FHT) for the conversion of time domain data into spectra. It is a real-valued alternative to the Fast Fourier Transform (Discrete Fourier Transform) commonly used in signal processing, developed by Ralph Hartley in 1942 [Hartley 1942]. As it operates entirely in the real domain, the output are not complex amplitudes per positive and negative frequency, but sine and cosine amplitudes per positive frequency. The design using only real numbers makes the FHT suitable for efficient implementations on hardware such as the embedded infrastructure chosen for this thesis.

In the continuous form, while $(\mathcal{F}f)(y) = \frac{1}{\sqrt{2\pi}} \int f(x) e^{-iyx} dx$, $(\mathcal{H}f)(\omega) = \frac{1}{\sqrt{2\pi}} \int f(t) \mathbf{cas}(\omega t) dt$ with $\mathbf{cas}(x) = \mathbf{cos}(x) + \mathbf{sin}(x)$. So instead of using imaginary powers of e to represent periodicity as a complex rotating vector, a real-valued sum of sine and cosine (**cas**) is used. The fast, discrete implementations preserve that difference: The outcome of a FHT is real-valued, while the outcome of a FFT is complex-valued.

In fact, [Reeves 1990] has shown in a master's thesis that even 30 years ago, even a 512×512 **two-dimensional** FHT can be used to find the results of a two-dimensional FFT on a desktop computer of the time in 28 s. Chapter 2 of that thesis also gives an introduction to the mathematical foundations, differences between FHT and FFT and basics for an implementation, while the rest of the thesis is more geared towards maximising performance on the specific classic hardware.

Today, desktop computers have many times the performance of 1990, which makes it hard to use most recent implementations of FHT on an embedded system. They tend to have very specific optimisations for really large input and output data vectors, using properties of specific modern desktop

CPU, such as the ability to process short data vectors in parallel in a single instruction. Luckily, Scott Paine's 'am, a program for radiative transfer computations at microwave to submillimeter wavelengths' [Paine 2019] contains a more compact, generic, yet still optimised implementation with a suitable license. Having been developed at the Smithsonian Astrophysical Observatory, it 'is a work of the United States and may be used freely, with attribution and credit'. It has been stripped down a bit further for use in the `ecghelper2` software developed for this thesis, still maintaining high speed on embedded hardware.

2.3 Filtering for fast signal processing

A comprehensive book about digital signal processing, [Smith 1998], is available not only on paper, but also as a free website[9]. For those interested in a deeper look at spectral transforms (FFT etc.), chapters 8 to 12 and 31 will be of value.

In particular in the real-time processing part of the algorithm designed for this thesis, discrete-time digital filters are being used. This has been the case in [Auer 2020] as well, but filter designs have been improved and more filter instances are used in the updated algorithm.

Introduced in chapter 14 of [Smith 1998], digital filters can be characterised by impulse response, step response and frequency response. For the purpose of this thesis, filters are used to suppress certain frequencies or emphasise or dampen signal components in selected frequency ranges. Some of the used filters also have morphological goals (smoothing) or are used for pattern matching by means of correlation or convolution.

For pattern matching, the algorithm uses Finite Impulse Response (FIR) filters. A number of ECG samples are processed in a moving window style, advancing one sample at a time. To find longer patterns or, when using FIR filters for their frequency response, to have more fine-grained control over their effect on different frequency components in the input signal, longer windows have to be used, which in turn means higher computational load. It also means having to look back across more samples, causing increasing processing delays.

In the ECG example, this would mean that QRS complex detection would trigger after a fixed, long delay when the detection relies on one or more 'long' filters, looking at a large number of recent samples. To be able to give real-time feedback, the `ecghelper2` algorithm limits filter sizes. Note that any delay does not affect the ability to know when exactly QRS events happened. It only influences how much later (in terms of having to wait for additional ECG data) it will be known to the algorithm that the event has happened at the still well-known point in the past.

Unlike the algorithm used in [Auer 2020], the approach designed for this thesis can also make use of Infinite Impulse Response (IIR) filters (see section 14.6, filter classification of [Smith 1998]). Specifically, Pei Tseng notch filters designed with the help of the free open source Octave signal processing package[10] are used to remove signal artefacts of fixed, known frequencies (60 Hz and 187.25 Hz). According to the signal package website, [Pei 1996] is the reference paper about those biquad IIR filters.

Pei Tseng notch filters can be very narrow, but to cope with some variation of the artefact frequencies, parameter settings for less narrow filters have been deliberately chosen. Like all IIR filters, the notch filters have a recursive design, which can lead to some amount of ringing in the step response. The trade-off between time and frequency domain qualities of filters is discussed in sections 14.3 and 14.4 of [Smith 1998].

Regarding the FIR filters, suitable functions of the Octave signal package have been used to generate designs, which have then been converted to integer arithmetics by multiplying weights with powers

[9] `https://www.dspguide.com/`

[10] `https://octave.sourceforge.io/signal/overview.html`

of two. Within the current time window, each (integer) sample value can be multiplied by an integer weight and a sum of products can be accumulated. The final sum is then divided by the chosen power of two (with rounding) to produce a normalised integer filter output. Again using Octave signal package functions, the effect of this weight quantisation has been reviewed in terms of step, impulse and frequency response. In particular, the use of short filters and low quantisation factors will always lead to some compromises between filter performance and embedded-friendly computational and memory demands and filter delay. Note that IIR filters can be sensitive to quantisation ([Smith 1998] section 19.5) so, again reviewing actual filter responses, more fine-grained quantisation has been used for those.

2.4 Principal component analysis and linear discriminant analysis

The recent publication of the PTB-XL ECG corpus [Wagner 2020a, Wagner 2020b] earlier this year provides a wealth of data for machine learning oriented approaches to A-Fib detection as a classification task. To do trained classification, the `ecghelper2` algorithm generates a set of features from an ECG recording, which can be as short as the 10 s excerpts provided by PTB-XL. Those features capture properties of the ECG in a more generic way than thousands of raw sample amplitudes would. One example is the standard deviation of the R/R interval durations, another can be the amplitude or duration of a specific wave in the averaged ECG shape.

However, the features can not necessarily be mapped to A-Fib risks in obvious ways accessible to manual algorithm design. For example, the [Auer 2020] algorithm used knowledge from literature and empirical observations from corpus data to define what would make ECG shapes too deviant from NSR or likely to be A-Fib in terms of values of individual ECG morphology features.

The algorithm designed for this thesis uses far more features than at most a few dozen morphology metrics. A three digit number of features is generated to provide a significant abstraction from the raw ECG sample data, spanning a multidimensional space of ECG properties. Each ECG is represented as one vector pointing to a point in that space. The classification task can be interpreted as the task of knowing which areas in feature space correspond to which classification outcome, based on gold standard annotations. PTB-XL metadata already **tell** which recording shows A-Fib, NSR or other types of ECG.

Yet, the algorithm must not just blindly learn to enumerate all PTB-XL recordings which correspond to A-Fib. A good classification also has to generalise to novel input – in this thesis: recordings from completely different corpora. So it is good to limit the amount of input detail to push the analysis towards learning a more general concept of A-Fib, rather than a list of example recordings. One way to do this could be to reduce dimensionality of the feature space in a generic way first.

While the actual `ecghelper2` algorithm does not implement this step (instead, it deliberately reduces feature detail levels used in classification and ignores less useful features), one way to map a set of input vectors to a set of output vectors with a reduced number of dimensions is to devise a projection scheme which can sort the output dimensions by how much of the variance of the input data they capture. Then, the output vector can be truncated to contain only the most 'interesting' dimensions. A classification system can then focus on those new, fewer dimensions to make decisions.

A downside of such projections is that the output no longer can be seen in terms of the known input dimensions. Being based purely on input data, all input metadata are lost. The projection neither knows nor cares which input is A-Fib or NSR, nor which input is related to R/R, spectral or ECG morphology features. It just maximises how much of the differences between the inputs could be reconstructed after transforming, truncating to fewer dimensions and back-transforming the data to the original feature space.

In this thesis, Principal Component Analysis (PCA) has been used to explore whether it would be possible to capture most the message of all features in a smaller set of PCA-transformed features.

To learn about the mathematical concept of PCA, the **pattern recognition lecture notes by Ricardo Gutierrez-Osuna** (`https://psi.engr.tamu.edu/courses/`) can be recommended, in particular lecture 9 about Principal Components[11] and lecture 10 abound Linear Discriminant Analysis[12], which also introduces alternative LDA other than Fisher's original approach and other dimensionality reduction methods, as well as the review article [Jolliffe 2016].

While the pattern recognition course proceeds to discuss more advanced methods such as Support Vector Machines (SVM), the `ecghelper2` algorithm reaches good performance with LDA already, which is fast and straightforward to train (in Linux) and to apply (both on the embedded system for real-time use and in Linux for corpus processing). For those linguistically (or rather, phonetically) inclined, the speech processing lecture notes of the same author also are a good read.

As with the Fast Hartley Transform, most of the application in the actual `ecghelper2` software, in this case in the Octave script used to perform PCA and LDA as part of the training step, does not rely on deep understanding of the respective algorithms. It is sufficient to grasp the general ideas to be able to understand existing recipes and adapt them to the specific task at hand. As the Dutch would say: 'Vraag niet hoe het kan, profiteer ervan'. In the case of said Octave script (`pca_lda.m`), the 2011 recipe by Philipp Wagner on his `https://www.bytefish.de/` blog has been used[13]. His GitHub publications also include facial recognition code, for example, his research interests include eye tracking and feature extraction.

The steps for Principal Component Analysis are:

- For each feature for all recordings, subtract the mean over all instances (here: mean of that feature for all ECG recordings, no matter whether A-Fib or NSR)
- Divide the result by the standard deviation to form a Z-score
- Compute the covariance matrix of the Z-score normalised features (cov)
- Compute Eigenvectors and Eigenvalues for that matrix (eig)
- Sort both (together) to have the first principal component correspond to both first entries
- The Eigenvectors can now be used to project the data by multiplying the input vectors, after removing means, with the Eigenvectors, to get a vector of new coordinates
- Optionally, data can be reconstructed by multiplication with the transposed Eigenvector matrix and adding the means again. This can be used to evaluate how well the data would be preserved after dropping some of the dimensions after PCA

In this thesis, no reconstruction has been used: PCA only answered the question how well the gist of the input data would be preserved when processing only a subset of the first N principal components instead of the raw feature vectors with $M \gg N$ features, based on the comprehensive wikipedia article about PCA[14].

Given that the PTB-XL corpus already specifies which recordings concern A-Fib and which are NSR, Linear Discriminant Analysis is a method of dimensionality reduction which is very suitable for the task at hand. 'Bytefish' Philipp Wagner presents that algorithm based on the lecture notes by Ricardo Gutierrez-Osuna and, of course, the verbose wikipedia article about the topic[15]. Of course, it is no coincidence that this thesis recommends the same sources as Bytefish – it follows and supports his recommendations.

[11] `https://people.engr.tamu.edu/rgutier/lectures/pr/pr_l9.pdf`
[12] `https://people.engr.tamu.edu/rgutier/lectures/pr/pr_l10.pdf`
[13] `https://www.bytefish.de/blog/pca_lda_with_gnu_octave.html`
[14] `https://en.wikipedia.org/wiki/Principal_component_analysis`
[15] `https://en.wikipedia.org/wiki/Linear_discriminant_analysis`

Having access to a known gold standard regarding which ECG recording is to be classified how, the goal of the LDA is to find a separating plane in feature space which can be used for optimised class separation. Expressed in another way, LDA finds an axis in this space along which in-class feature vectors cluster as much as possible and the clusters for different classes differ as much as possible. The axis and plane are orthogonal and cross each other at the point where most feature vectors for one class are on one side and most feature vectors for the other class are on the other side, if only two classes are used.

In other words, the classification task can be expressed as projecting the feature vectors on the new axis and observing the sign of the difference between the real value of the separation plane and any projected feature vector on that axis. Like PCA, the LDA actually transforms the input vectors to output vectors with the same number of dimensions. The trick is that the first dimension now is sufficient to make the classification decision, with remaining dimensions covering remaining variability of the input data in a way which, to cite Bytefish, maximises between-class and minimises within-class scatter, clustering classes in the post-projection feature space.

To do this, LDA first has to calculate scatter between and within classes. Based on that, the compact and (when run on a desktop computer for the number of ECG recordings and features processed in this thesis) fast algorithm proceeds to find a projection. A sketch of the algorithm is:

- Subtract means from feature vectors, for each class separately (using the per-class means for each feature)
- For the within-class scatter, sum the dot products of the mean-free features for all within-class instances with their transpose:
  $S_{\mathbf{within}} = ((\mathbf{NSR} - \overline{NSR})^T \cdot (\mathbf{NSR} - \overline{NSR})) + ((\mathbf{FIB} - \overline{FIB})^T \cdot (\mathbf{FIB} - \overline{FIB}))$
- For the between-class scatter, consider how class means differ from global means (means of features for all feature vectors, ignoring class membership) as follows, $N_{\mathbf{class}}$ = size of **class**:
  $S_{\mathbf{between}} = (N_{\mathbf{NSR}} \cdot (\overline{NSR} - \overline{all})^T \cdot (\overline{NSR} - \overline{all})) + (N_{\mathbf{FIB}} \cdot (\overline{FIB} - \overline{all})^T \cdot (\overline{FIB} - \overline{all}))$
- Now solve $S_{\mathbf{within}} \cdot x = S_{\mathbf{between}}$ to find $\mathbf{x}$, using the Octave code: $S_{\mathbf{solved}} = S_{\mathbf{within}} \backslash S_{\mathbf{between}}$
- Compute Eigenvectors and -values for the result: `[eigvec, eigval] = eig`($S_{\mathbf{solved}}$)
- Rounding may have introduced some small imaginary components. Remove those by taking the absolute value of the Eigenvalues before sorting the Eigenvectors and Eigenvalues by magnitude of Eigenvalues
- The cumulative sum of the first N Eigenvalues, expressed as fraction of the sum over all Eigenvalues, shows that the first (one) dimension already covers almost the full classification task of clustering the feature vectors into A-Fib versus NSR
- As with PCA, the Eigenvectors can be used for projection and reconstruction. Note that the **global** mean has to be removed from or re-added to the raw data here. The **per-class** mean and normalisation steps are only used to find $S_{\mathbf{solved}}$ and the `eig()` of that. The projection itself is, importantly, agnostic of which class input vectors are.

For a more verbose description, refer to the Bytefish blog item, which demonstrates both PCA and LDA with a small, two-dimensional example data set and later a larger example data set about wine, presenting plots of raw, projected and reconstructed data and additional explanations. It does use somewhat more elaborate Octave constructs, so the actual code is a bit harder to read than the corresponding parts of `pca_lda.m` – which is much longer, processing the results further to plot ECG feature vector projections, generate C source code for the selected projection, etc.

The code aims to find a balance between using ready-made functions and staying transparent enough to show what is happening, in particular given that the result has to be source code to run on an embedded system.

In a more advanced environment such as Octave, which has immediate support for complex-valued vectors and matrices, more high-level representations would have been more easily accessible.

A simple standard C library for embedded systems does not offer that level of support for advanced mathematics. Also, the ESP32 hardware itself is fast only for simple arithmetics (addition, subtraction, multiplication) on integers and 32-bit floating point numbers. Even divisions are already noticeably slower. Higher precision floating point operations, trigonometric functions etc. are all slow.

Luckily, loops in FHT and other places in `ecghelper2` can often avoid repeated calls to trigonometric functions by iteratively computing sine and cosine of all integer multiples of one start angle.

2.5 Hardware choices

Based on experiences with and datasheets of the Maxim MAX30003 ECG analogue front-end and the ESP32 system on a chip, specifically the WROOM32 module used on the Adafruit HUZZAH32 ESP32 'feather' board, the hardware design for this master's thesis keeps the MAX30003 and the WROOM32 as core components, similar to [Auer 2020].

While the MAX30003 works with little power consumption even for full activity (up to 512 sps single-channel ECG recording with optional built-in QRS complex detection), it already contains all necessary analogue and digital functionality for ECG recording. The ESP32 only has to download the sample data via SPI and few external components for EMI filtering, safety etc. are required.

Importantly, the MAX30003 also has an ultra low power lead-on detection mode, where a tiny current between the ECG leads can be detected while the chip is in a low-power mode and used to generate an interrupt signal. This interrupt signal can then be used to wake up the ESP32, which has the ability to hibernate in low-power modes. Signals on I/O pins are one of the configurable ways to leave ESP32 standby and hibernation modes.

The `ecghelper2` design also keeps the CP2104 USB to serial port converter seen in the Adafruit HUZZAH32. The datasheet reveals that, unlike the officially recommended successor model, this chip easily tolerates 3.3 V logic levels on I/O pins even while powered down. So instead of searching for an USB to serial bridge with low power standby, the design choses to completely disconnect the power supply for the CP2104 until an USB bus voltage is sensed. In addition, the logics for letting 'modem' signals of the USB serial port control reset and boot modes of the ESP32, called 'autoreset' by Adafruit, are mimicked by `ecghelper2` (in a modified, lower component count way). This means that the toolchain can be used as if a normal HUZZAH32 would be connected instead of the actual `ecghelper2`.

The Adafruit module uses very few I/O signals, as it contains very few peripheral devices and leaves most pins free for external user circuity. One notable exception is that it has a LED connected to one I/O which conveniently also corresponds to one of the RGB (Red Green Blue) LED (Light Emitting Diode) signals of the `ecghelper2`. So the default boot loader can indicate boot status in the same way on both devices, before custom firmware for ECG processing and analysis takes over. There also is a reset button on both modules, although the use of a watchdog to reboot when software gets stuck can make it optional for the ECG.

A significant difference between HUZZAH32 and `ecghelper2` can be found in the power supply. Not designed towards energy saving, the former contains a simple linear low-dropout (LDO) voltage regulator and is usually powered from USB. Interestingly, it also has a built-in Li-Ion / Li-Poly battery controller. When a battery is controlled, it will be either charged (USB on) or feed the LDO, which is possible because this battery type will usually output slightly more than the required 3.3 V.

For the `ecghelper2`, however, a pair of easily available 1.5 V batteries will be used, so the voltage has to be boosted from at most 3.0 V to 3.3 V in battery mode. On the other hand, to improve battery life, the device should be able to run from up to 5 V USB power during debug and firmware upload sessions. This means that a buck / boost switching power supply should be used, which can convert voltages either up or down, as needed.

Ideal diodes pass power from either source to the supply, with minimal losses. Those are integrated circuits which simulate diodes by controlling FET switches with optimised conduction properties, in particular avoiding the dropout voltage present in normal diodes. The control circuit itself also uses some power and only has limited speed, but the latter is no issue for slow power source switching. The `ecghelper2` design can select ideal diodes with favourable conduction and internal power consumption properties.

Section 3 describes both the choice of power supply and power source, as well as many other aspects of the hardware design. One of the important properties of the power supply is the ability to keep the device in standby while consuming very little power. As soon as the user touches the ECG leads, the lead-on feature of the MAX30003 can the wake up the rest of the device. Thus, no power switch is necessary, improving usability and avoiding a mechanical component which could be subject to wear over time.

As the HUZZAH32 contains no ECG hardware, display, audio output or RGB LED, only a single LED under I/O control and one more which is directly connected to the Li-Ion charge controller to indicate battery status, the `ecghelper2` design is free to add any of those in suitable versions without affecting the (toolchain, source code, Arduino IDE[16] etc.) compatibility to the Adafruit board. Running the `ecghelper2` firmware on it can just send all signals meant for those peripherals and the ECG to the I/O pins where the peripherals would be on HUZZAH32.

Those could be acquired separately and wired to the HUZZAH32 with the help of a breadboard. As the [Auer 2020] system already connects a display, 'smart' LEDs and a MAX30003 'wing' board to the HUZZAH32 'feather', the new hardware and software have been designed to be distinguishable from the old system. That way, one version of the firmware can be written which automatically picks whichever peripherals are available. The mechanism does not really detect the individual devices. It only has to be able to find out whether it runs on the new hardware or rather on the old version.

Section 3 will describe the choices made for the display, RGB LED and audio hardware of the `ecghelper2` design. They all use standard components, carefully chosen for suitability in a low-power standby environment, so no special circuit design has been necessary. However, a proposed `ecghelper3` hardware will introduce additional circuity to reduce standby power consumption even further. It also adds more sensors and battery monitoring, to make the device more versatile and reliable **and** increase battery life in standby mode.

[16] integrated development environment, editor, compile shell, firmware upload tool etc

3 Hardware design

The `ecghelper` prototype developed for this thesis uses the same controller, ECG and USB chipset as the proof of concept breadboard system described in [Auer 2020]. However, the complete user interface got upgraded and the whole device is optimised to minimise energy consumption, for maximum battery life. The design includes mechanical aspects and built-in electrodes, as well as safety standards and EMI. This section concludes with a report of the prototype assembly, issues found and solved in the process and a proposal for an again improved third generation device design. The last subsection compares I/O pin assignments between all three generations, as differences will be important for the software, described in 4. The bill of materials for the prototype, full schematics for both the prototype and the proposed third generation device and some signal and current measurements can be found in appendix A.

The hardware design will be made available as open hardware under the Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) license[17]. Part of this hardware design is based on the Adafruit Feather ESP32 HUZZAH32 design by Limor Fried / Ladyada for Adafruit Industries[18], which is licensed under the CC BY-SA 3.0 License.

3.1 ECG analogue front-end

As in the system described in earlier work by the author [Auer 2020], the MAX30003[19] chip by Maxim Integrated will be used as integrated ECG analogue front-end (AFE). The ECG circuit design is based on the evaluation board used in [Auer 2020]. While using only 85 μW in active mode with up to 500 or 512 sps at 16 bit resolution, it also features an ultra low power lead-on detection mode with a total supply current below 5 μA for all supply voltages. In this mode, the AFE can generate an interrupt signal as soon as a current is detected between the ECG electrodes.

A 24-bit chip like the TI ADS1291[20] would have provided higher sampling rates and higher effective resolution (ENOB, effective number of bits), but would also have consumed more power and would have lacked the ability to generate interrupts while in a low-power mode. Also, according to for example [Pizzuti 1985], 500 sps and probably even 250 sps are sufficient as sampling rate for the purposes of this thesis.

The electrodes are internally connected to ground and, via a $> 20\,\mathrm{M\Omega}$ resistor, the supply voltage when the MAX30003 is idle in the special lead-on detect mode, so less than 0.2 μA will flow through the user's body initially. During ECG measurements (the highest two selectable sampling rates are 500 and 512 sps), it is possible to inject a lead-off detection current of 5, 10, 20, 50 or 100 nA which causes the signal to depart from a selectable input voltage range as soon as the connection to the user is broken, triggering a 'lead-off detection' status flag change. All detection events can also trigger interrupts through either of two extra pins.

Input impedance of the MAX30003 is high ($> 1.5\,\mathrm{G\Omega}$) and the chip features the complete analogue and digital processing chain needed to measure ECG. An analogue high-pass filter reduces DC drift. It is controlled by an external capacitor, for which the `ecghelper` design offers two choices, selectable by a solder jumper, either 0.4 Hz (recommended) or 0.04 Hz (less distortions, but less DC drift removal). A software compensation undoes part of the filter distortion and has to be adjusted (at compile time) to match the jumper setting. The chip also offers an IIR digital high-pass filter (0.4 Hz, first order Butterworth) and FIR digital low-pass filters (12-tap, 40, 100 or 140 Hz).

[17] `https://creativecommons.org/licenses/by-sa/4.0/`

[18] `https://github.com/adafruit/Adafruit-HUZZAH32-ESP32-Feather-PCB`

[19] `https://datasheets.maximintegrated.com/en/ds/MAX30003.pdf`

[20] `https://www.ti.com/product/ADS1291?qgpn=ads1291` also available as 2-channel versions ADS1292 and ADS1292R, the latter with respiratory impedance measurement

LPF frequency	SNR	ENOB	RMS noise	P-P noise
40 Hz	82.4 dB	13.4	0.34 μV	2.22 μV
100 Hz	79.1 dB	12.8	0.49 μV	3.24 μV
140 Hz	77.2 dB	12.5	0.61 μV	4.01 μV

Table 3.1: MAX30003 signal quality data, excerpted from the datasheet

As the reference recordings are usually filtered at 40 Hz and this, lowest, low-pass filter cut-off frequency setting removes most of the high-frequency noise in the signal, `ecghelper` normally uses that setting as well. A higher setting would allow more high-frequency components of the ECG to be analysed, unless software filters block them.

For convenient 2-lead recording without a reference or leg electrode, the `ecghelper` system uses the lead bias option of the MAX30003, which pulls both input electrodes towards the middle or centre voltage of the input amplifier, through selectable resistors between 50 MΩ and 200 MΩ. When strong DC offsets are present at the input, the chip can automatically change the high-pass filter slew current by a factor of almost 1800 to quickly re-centre signals. This feature is called fast recovery mode and can be both controlled and detected through configuration and status registers.

Table 3.1 lists the signal to noise ratio, effective number of (ADC) bits and noise at the highest selectable gain of 160 V/V (alternatives are 20, 40 and 80 V/V), for different settings of the low-pass filter cut-off frequency. It can be seen that the lowest cut-off frequency has the best SNR and lowest noise levels, providing more than 13 bits of effectively useful analogue to digital conversion resolution. Note that the chip internally uses 18 bits resolution and circa 32 ksps sampling rate (depending on the selected output sampling rate). It then uses digital filters to generate the data available to the system through a 32 sample FIFO buffer. Each sample is buffered with tag bits to indicate whether it is the last available one at that moment and whether a FIFO overflow or recovery mode event were ongoing while capturing it.

3.2 User safety

The design has surge-proof 49.9 kΩ or larger resistors between the rest of the circuit and any of the electrodes. The overvoltage protection of the power supply chip (see below) prevents voltages above 7 V to reach the MAX30003 even in fault conditions and the lowest impedances within the MAX30003 in normal operation are 20 MΩ to the supply voltage for one electrode and a direct connection to power supply ground for the other. A third electrode connected to the internal middle voltage of the input amplifier uses a 499 kΩ resistor between user and MAX30003. The only place where higher voltages exist in the device is between the display controller and the display panel (−12.5 V to 15.0 V) according to the controller datasheet. Even those are still within the extra-low voltage range.

The device is electrically isolated and, as long as the housing is not opened, has no connectors for external voltage sources. To conform to the safety requirements for type CF devices, currents through the user (both intentional patient auxiliary currents and unintentional patient leakage currents, both AC and DC) have to stay below 10 μA during normal operation and below 50 μA in single fault condition per IEC 60601-1[21] (see [Kramme 2011], chapter 5, pages 51 & 55).

This is the case for the main electrodes. Lead-on detection has an expected current flow below 165 nA, lead-off detection uses configurable currents of up to 100 nA and lead bias, which internally connects electrode inputs to middle voltage through selectable resistors of 50, 100 or 200 MΩ each involves even lower currents. The middle voltage is not output during lead-on detection mode. While

[21] See `https://www.flukebiomedical.com/blog/electrical-safety-standards-basic-testing` for a quick overview of applicable current limits

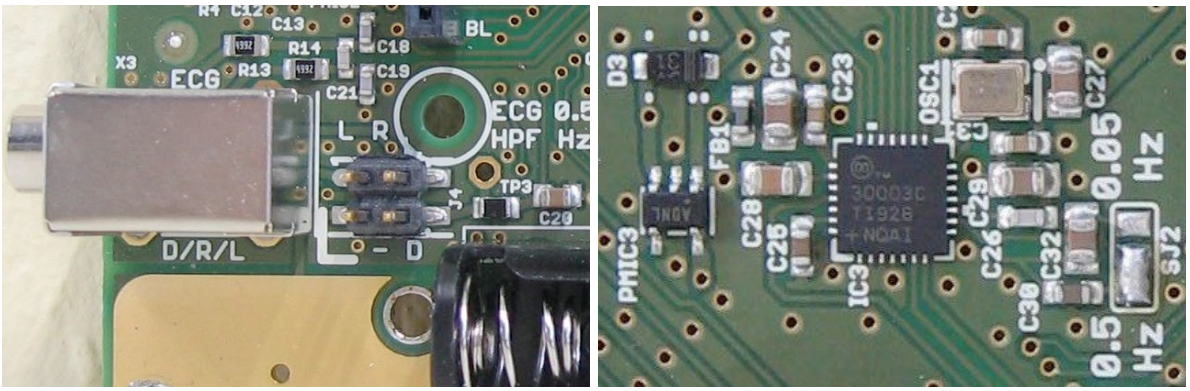


Figure 3.1: Left: ECG electrode circuit with safety resistors, external electrode connector & test header. Right: ECG AFE circuit, 1.8 V supply & low-power 32768 Hz oscillator

measuring, input impedances are high, so there is no current path between the middle voltage output and the other electrodes which would exceed the specified safety limits.

As the device is floating with respect to ground potential, there are no ground currents to be considered. All fault currents will be related to voltages present within the device itself.

The worst conceivable single fault would be having 3.3 V connected directly to one electrode while the other is connected to ground via 49.9 kΩ in lead-on detection mode. This would produce currents of up to 66 μA. By making sure in the layout that traces to the electrodes are never adjacent to or nearby supply voltage traces, this is reduced to 33 μA in the case of an internal short-circuit between the electrode input and the supply voltage inside the MAX30003 itself, while the other electrode output is connected to ground in lead-on detection mode. Because the fault current now has to travel through both 49.9 kΩ resistors to flow through the user, it is reduced to 33 μA. The circuit can be seen in figure 3.1.

Even a **double** fault with the improved layout, with 7 V error voltage, would still only cause 70 μA fault current. Opening the device will expose the user to at most 3.3 V, or at most 7 V if the power supply chip fails. The < 10 V handled by the audio circuit (see below) are not easily touched, as the circuit is under the display module and the battery holders form a mechanical obstacle. Also, the piezo driver voltage is only generated while sounds are produced. The display controller voltages are not exposed to touch. The only conceivable really dangerous situation would be the user connecting a faulty USB power supply to the device while touching circuits or ECG electrodes.

This obviously is as dangerous as directly touching the faulty USB power supply in the first place and not something which can be prevented in the `ecghelper` design. Serious accidents have happened with USB power supplies in the past, for example when people used their smartphones while charging them and a fault connected the output to 230 VAC. So it is important that the USB port (meant for firmware uploads and debugging) of the `ecghelper` is **not** accessible while the housing is closed. Even when opening the housing to replace batteries, it still should not be accessible, so the lids to access the batteries and to access the USB port should be different and it has to be made clear that users must not open the USB lid.

That said, it is not more dangerous to run the device with the USB port connected to a battery-powered, non-grounded laptop than it is to touch the USB port of that laptop, so engineers who are debugging the device are not exposed to extreme risks. Of course, most debugging can take place while **not** touching the electrodes anyway. The best option is to access ECG data only through the available wireless interfaces and to use the USB interface only for firmware updates.

The device does not involve radiation beyond the electromagnetic radiation of standard wireless interfaces. It does not contain dangerous light sources, but suitable light guides can make the light from signal LED more diffuse and more pleasant to look at. No dangerous amounts of heat are

produced either. A switching power supply chip with overtemperature protection is the only place where large currents are processed in a small space. The used microcontroller module has a metal lid as heat spreader and to reduce EMI.

To avoid mechanical risks, the housing should avoid too sharp edges. As there are no moving parts (apart from the reset button accessible though a hole in the housing or by opening the housing) there are no entrapment or similar risks either.

To prevent risks of fire, the power supply does not only have overtemperature and overcurrent / short-circuit protection, but also a fuse. Solder mask and layout lower the already low risk of direct battery short-circuit when metal parts touch the circuit board after opening the housing. Such short-circuits could cause the batteries to overheat and cause fires. Lithium based batteries would be more problematic than alkaline disposable or NiMH rechargeable batteries in such events.

3.3 Central microcontroller / low power standby

The ECG AFE lead-on interrupt is used to start the central controller, which will again be an ESP32[22]. Again, it will be used in the form of an ESP32-WROOM module, which contains a dual core ESP32 controller with wireless interfaces (one core is normally reserved for wireless communication there) and flash memory for the firmware, as well as a WiFi / Bluetooth antenna. The series also has SOLO modules with only one core and WROVER modules with additional RAM (Random Access Memory), but the ESP32 itself already contains 520 kB RAM, which is enough for recording and analysis of far more than 30 s of ECG data. The controller has a ROM BIOS (Read-Only Memory Basic Input Output System) for booting and basic functionality and the modules are available with 4, 8 or 16 MB of flash storage.

A WROOM module with the largest amount of flash storage is used. The `ecghelper` firmware takes 1310 kB of flash, 116 kB of static variables in RAM and a smaller amount of stack and dynamically allocated data in RAM in the final version, so neither RAM nor flash are bottlenecks. 320 kB of RAM can be used for WROOM32 applications, the rest is reserved for system tasks.

The ESP32 clocks at up to 240 MHz (default 80 MHz) and has fast hardware multiplication for both integer (32-bit) and single precision floating point data. Division is hardware-assisted, more advanced functions (e.g. sine, cosine) have to be calculated fully in software by the C library of the compiler. Espressif, who produces the ESP32, offers a toolchain with Arduino (`https://www.arduino.cc/`) integration, making development of embedded firmware for it convenient[23].

The ESP32 itself combines a powerful CPU, built-in Bluetooth and various interfaces with low power consumption in several available sleep states. While light sleep (idle) only stops program execution until the next event (such as a timer interrupt) and still uses 0.8 mA, deep sleep and hibernate modes offer power consumption levels of 15 μA and below[24]. Those still feature configurable wake up events, but do not preserve execution state. Advanced low power modes (up to 150 μA) even allow running limited computations on the built-in RTC (real time clock) coprocessor while in standby. Those are not used for `ecghelper`.

For simplicity, no state information is kept in deep sleep memory. Instead, the lead-on interrupt of the MAX30003 ECG AFE is used to trigger a full system boot, without preserving ESP32 state information. Basically, the `ecghelper` will use the ECG electrode connection of the user as a power switch. While performing a measurement, the AFE will be configured to use lead-off detection (selectable $5 - 100$ nA in AFE registers) to notice signal loss.

[22] `https://www.espressif.com/en/products/modules/esp32`

[23] Caveat – an empty `component.mk` file has to be present in the source directory to properly compile software consisting of multiple source code files, as opposed to just the main application `*.ino` C/C++ source code file

[24] See the ESP32 datasheet `https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf`

As soon as either the signal is lost or enough ECG data has been recorded, the device will either return to deep sleep or it will present the ECG analysis. In the latter case, if leads are still on at the end of a fixed presentation time period, Bluetooth communication will be enabled. This gives the users control over whether or not they want to allow wireless access to their data without requiring additional buttons or similar.

At the end of the result presentation period or, if Bluetooth mode is used, at the end (either by time-out or using a command sent wirelessly) of the Bluetooth session, the `ecghelper` will also return to deep sleep mode. Before doing so, the AFE will of course be configured to sleep in lead-on detect mode and all user interface components of `ecghelper` will be powered down. A watchdog timer reboots the device if it unexpectedly ends up in an infinite wait, in spite of using explicit timeouts wherever possible.

To conserve energy, all wireless communication is disabled early at boot. It is only optionally enabled after presenting analysis results. One disadvantage of this is that peak power consumption happens late in an online measurement cycle. If batteries are weak, the device may fail and reboot due to a power supply brown-out at this point. The brown-out detector is configurable via `sdkconfig.h` and is on with the lowest threshold ($\approx 2.4\,\mathrm{V}$) by default. Changing the detection level from 0 to 7 sets a higher threshold of $\approx 2.8\,\mathrm{V}$ for earlier resets when the power supply degrades[25]. Empirically, stability is best $> 2.4\,\mathrm{V}$ raw battery voltage.

However, it is expected that offline measurements without Bluetooth are far more common. The user is expected to check their ECG more often than they will find ECG states that they will want to store or share with others. Therefore, the boot process does not involve Bluetooth test activity, which would also require a peer to realistically test peak data streaming power consumption anyway.

3.4 Space considerations

A reset button is available in the design, but has been omitted for a proposed improved version to have no moving parts left which could be subject to wear. As mentioned, a watchdog timer keeps the firmware from getting stuck even in case of errors and it is always possible to reset the device by removing the batteries for a moment. Also, only the prototype design features several debug headers which allow both measuring bus signals and adding extra circuity for testing purposes. A proposed future version omits those headers to save space and adds extra sensors to the free space between display and circuit board.

The ESP32 is available on a convenient 1.27 mm pitch, 4-layer PCB (printed circuit board) module with integrated Bluetooth and WiFi antenna. This greatly simplifies the `ecghelper` design, which uses a 2-layer PCB. Where possible, all components are SMD (surface mount devices). The only exceptions are the battery holders, a pin header for the used display (both for stability and because it has a smaller footprint than a SMD header) and a 3.5 mm TRS (tip ring sleeve) mini jack plug socket which allows the connection of external ECG electrodes, such as those used for the breadboard system, or even sticky disposable electrodes for better ECG signal quality and hands-free use.

For the passive components, 0603 and larger parts have been selected to balance device size and ease of production. A proposed next generation of the `ecghelper` device uses more 0603 and fewer larger parts, to save even more space. This means more unlabelled parts, which is inconvenient for hand mounting, but causes no problems when using machine assembly.

The design also uses larger capacitors than strictly necessary in terms of physical size. This has the advantage that MLCC (multi-layer ceramic capacitors) still have more usable capacitance at the expected DC bias voltages, which improves voltage and signal stability in the circuit. The full bill of materials (BOM) can be found in appendix A.2.

[25] See the ESP32 technical reference manual, for voltages for all RTC_CNTL_DBROWN_OUT_THRES values: `https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf`

3.5 Power source

While the proof of concept project system used off-the-shelf components combined on a breadboard and powered either from a laptop or an USB power bank, the device needed for this thesis has to be working with easily available and small batteries to be most user friendly.

The ESP32 board of the project system includes infrastructure to manage a Li-Ion or Li-Poly battery. Those are available in various sizes, but not for example in supermarkets. So the device would have to ship with a suitable battery and the users may need assistance to get replacements.

In addition, having a rechargeable battery as fixed part of the design means the users would need a way to charge it while in the device. One common approach here are USB chargers, as USB ports are widespread and people often already have a charger, for example for their smartphones.

If the device were to support USB charging, users would be tempted to use it **while** charging. However, USB chargers are far from being safe and reliable enough to be connected, even indirectly, to humans who are recording their ECG.

So instead, the device has standard size batteries. Either disposable or rechargeable ones could be used, but charging takes place while the battery is not inside the device. Lithium button cells are relatively expensive and have limited capacities: 160, 200 or 550 mAh at 3.0 V are example capacities seen for 2025, 2035 and 2450 sizes on the Reichelt Elektronik website[26], which translates to 0.5, 0.6 and 1.7 Wh, respectively.

In addition, the device will draw high currents while transmitting data over Bluetooth. WiFi would draw even more power and the stronger electromagnetic radiation might be undesired, in ECG context and close to the user. As one design goal is to be able to share data with other devices, Bluetooth is a good option. Wired connections are avoided, so there are no risks to disturb measurements via ground loops or to cause unwanted currents through the user.

Interestingly, according to the user manual, the Kardia ECG device by AliveCor[27] uses 19 kHz audio to send data to a nearby smartphone or tablet, which allows it to run from low power button cells, but limits use to selected peer devices with sufficiently sensitive microphones. They also need special apps to implement the unusual, probably patented, communication method.

For maintenance purposes and firmware uploads, the device will provide USB connectivity compatible to the breadboard ESP32 system, but the USB port will only be accessible by opening the housing. For maximum flexibility, the power supply should support USB (5 V), Li-Ion and Li-Poly ($3.0 - 4.2$ V) as well as 2× alkaline (disposable) and 2× NiMH (rechargeable) input voltages. Regarding the voltage, Lithium button cells ($2.5 - 3.7$ V) would also be supported if they were able to deliver enough current. Alkaline AA or AAA size batteries have a nominal voltage of 1.5 V, NiMH ones available in the same sizes deliver 1.2 V, so the power supply should at least work with input voltages from $2.4 - 5.5$ V, to allow some tolerance in the USB voltage. Supporting lower input voltages would make the device more usable with rechargeable NiMH batteries, as their output voltage drops below 1.2 V unless they are in well-charged state.

Assuming 1700 mAh alkaline or 2500 mAh NiMH capacity for AA (Mignon) size and 650 mAh alkaline or 800 mAh NiMH capacity for AAA (Micro) as above, and the use of two batteries, 2.0 Wh are available from a pair of AAA batteries of either technology, or $5.0 - 6.0$ Wh when using AA size. Capacity assumptions are based on common values seen at Reichelt Elektronik.

Using 3 batteries of the same type would still be within the desired voltage range, but batteries are often sold in multiples of 2. With 4 alkalines, the voltage would be higher than with USB, requiring wider input voltage ranges for the power supply.

[26] `https://www.reichelt.de/`
[27] `https://www.alivecor.com/`

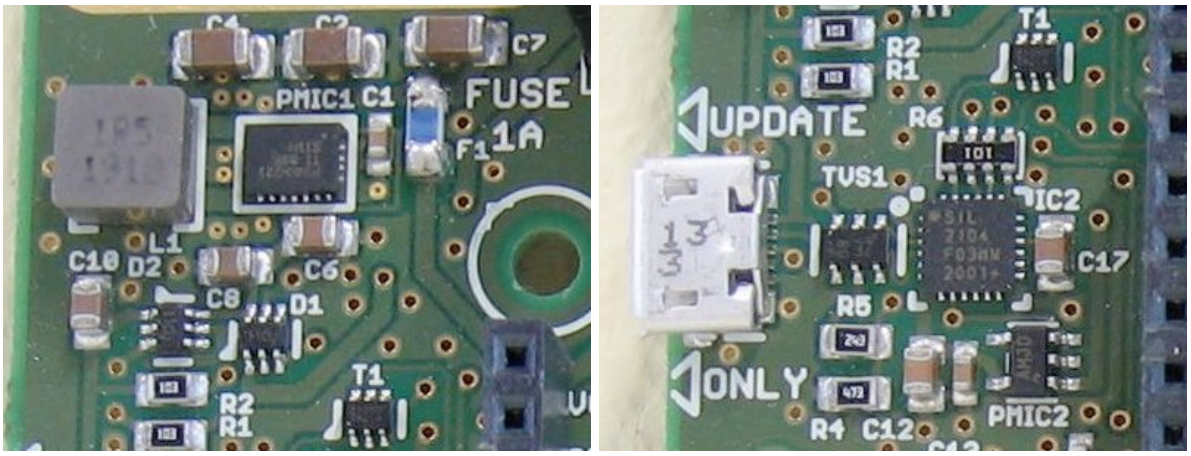


Figure 3.2: Left: Power supply (without T1) with fuse & ideal diodes (battery/USB selection). Right: USB (without R1, R2) with boot control, ESD protection & power gating

To make the device as small as possible, a pair of AAA batteries has been chosen to power it, so the energy budget are 2 Watt-hours until having to replace or recharge batteries. With careful choice of components, it is possible to keep standby at fractions of 1 mA for the expected battery voltages, corresponding to several months of standby operation. And even if the device were to draw 650 mA while active, it would still mean one hour of non-stop use. Given the experiences from the project phase, one hour is enough for dozens of measurements.

3.6 Power supply

To make the best use of the given energy budget, the `ecghelper` device needs a power supply which is efficient both at active loads and while in standby. Searching the TI Webench[28] website for suitable power supply ICs resulted in the choice for a low power buck / boost switching power supply. Normal linear voltage regulators would have required the input voltage to be above the 3.3 V required for the ESP32 and efficiency would have been low. Buck converters also require the input voltage to be at least as high as the output voltage, while boost converters require it to be lower. As battery voltage is lower, while USB voltage is higher than the required 3.3 V, buck/boost became a requirement.

Buck (step-down) converters would have been an option when requiring **three** batteries: Webench found several products with quiescent currents between 17 and 28 μA. LMR36503RS3-Q1 even advertises 4 μA, but for higher input voltages. The TPS62745 converter can only handle up to 22 μF of capacitive loads, making the 0.4 μA quiescent current device not suitable for `ecghelper` which would be very hard to keep below 22 μF of total bias capacitors – all details taken from the respective datasheets.

Using a required input voltage range of 2.5−5.5 V and at least the ability to provide 300 mA of output current, as well as high light load efficiency, as search criteria, Webench still presented too many candidates. So some additional requirements were chosen. At most 140 mm^2 area for the circuit, at most 32 mV ripple, at most 400 mA inductor ripple, at most 8 components needed, housing must not have less than 0.5 mm pin distance, excluding some 0.4 mm Ball Grid Array (BGA) and similar fine-pitch packages. Few options remained, all with 5.5 V maximum input voltage. Minimum input voltages varied from 1.3 to 2.3 V, as shown in table 3.2. Where maximum currents differ for buck and boost mode, the lower value is given.

[28] `https://webench.ti.com/power-designer/`

Device	V_{in}	I_q	**Efficiency**	I_{out} (versus input voltage)	**Package, required inductor and capacitors**
TPS630252	$\geq 2.3\,\mathrm{V}$	$35\,\mu\mathrm{A}$	$\approx 60\%$ $100\,\mu\mathrm{A}$, $> 88\% > 1\,\mathrm{mA}$	$\leq 2\,\mathrm{A}$ (...)	0.5 mm VQFN or 0.4 mm BGA, L & C see docs
TPS63001	$\geq 1.8\,\mathrm{V}$	$41\,\mu\mathrm{A}$	$> 63\% > 1\,\mathrm{mA}$	$\leq 0.8\,\mathrm{A}$ (...)	0.5 mm VSON, needs V_{ina} bootstrap, L & C see docs
TPS63021	$\geq 1.8\,\mathrm{V}$	$25\,\mu\mathrm{A}$	$\approx 65\%$ $100\,\mu\mathrm{A}$, $> 85\% > 1\,\mathrm{mA}$	$\leq 2\,\mathrm{A} > 2.6\,\mathrm{V}$ ($1.0\,\mathrm{A} > 2.0\,\mathrm{V}$, $0.6\,\mathrm{A} > 1.8\,\mathrm{V}$)	0.5 mm VSON, L typ. $1.5\,\mu\mathrm{H}$ ($1.0 - 2.2\,\mu\mathrm{H}$), $C_{\mathbf{in}} \geq 10\,\mu\mathrm{F}$, $C_{\mathbf{out}} \geq 66\,\mu\mathrm{F}$
TPS63802	$\geq 1.3\,\mathrm{V}$	$11\,\mu\mathrm{A}$	$\approx 75\%$ $100\,\mu\mathrm{A}$, $> 90\% > 1\,\mathrm{mA}$	$\leq 2\,\mathrm{A} > 2.3\,\mathrm{V}$ ($1.2\,\mathrm{A} > 1.8\,\mathrm{V}$, start $> 1.8\,\mathrm{V}$)	0.5 mm VSON or WCSP, L $0.47\,\mu\mathrm{H}$, $C_{\mathbf{in}} \geq 10\,\mu\mathrm{F}$, $C_{\mathbf{out}} \geq 22\,\mu\mathrm{F}$

Table 3.2: Comparison of low power TI buck/boost converters by minimum input voltage, quiescent and maximum current and efficiency at typical battery voltages and example currents

While the TPS63802[29] has the lowest quiescent current, works with smaller inductor and output capacitance values and has better performance at lower input voltages, it was not yet easily available when the prototype parts were ordered. So the TPS63021[30] is used for the `ecghelper` device described in this thesis, but a proposed improved third version (first version being the breadboard one) described later will use the TPS63802. The smaller inductor also helps to make the third version of the device significantly smaller.

TI Webench has been used to select suitable values for the external components, also taking EMI requirements into account. As mentioned, (physically) larger capacitors than strictly necessary have been chosen to improve power supply stability. Of course, all functional modules within the overall circuit also have local bypass capacitors as recommended in their respective datasheets.

While the converters feature output current limiting in case of short-circuits, a SMD fuse between the power supply and the rest of the circuit is part of the `ecghelper` designs. This allows separate testing of the power supply, keeps the batteries from being affected by persistent high currents in case of a short-circuit and protects the device from further damage.

The AFE requires three supply voltages. One for the interface, which is kept at 3.3 V to match the ESP32 and two for the analogue and digital ECG handling, which are both 1.8 V here. The AFE does support $1.1 - 2.0\,\mathrm{V}$ there, but input range and total harmonic distortion are worse at 1.1 V and power consumption is already very low at 1.8 V anyway. Also, the design simply follows the proven example of the evaluation board, which also uses 1.8 V.

To generate this voltage, a low drop-out (LDO) linear voltage regulator is used. The Maxim Integrated MAX1726[31] series, which is also used by the evaluation board, has a quiescent current below $5\,\mu\mathrm{A}$, typically $2\,\mu\mathrm{A}$, which makes it well suited for the `ecghelper` even in standby, waiting for a lead-on event. As the AFE consumes less than 0.1 mW even when active, it is better to use a LDO instead of a switching voltage converter. The Richtek RT9073[32] LDO offers even $1\,\mu\mathrm{A}$ quiescent current and higher (250 mA instead of 20 mA) maximum output load. However, the LDO from the evaluation board circuit is a proven choice and the I_q gain would only be $1\,\mu\mathrm{A}$ by switching to the Richtek LDO. In addition, the Maxim support has been quite helpful in clarifying ambiguities in the AFE datasheet, so it seemed appropriate to use their brand for the LDO as well.

[29] `https://www.ti.com/product/TPS63802`
[30] `https://www.ti.com/product/TPS63021`
[31] `https://www.maximintegrated.com/en/products/power/linear-regulators/MAX1726.html`
[32] `https://www.richtek.com/assets/product_file/RT9073/DS9073-06.pdf`

3.7 USB interface

For easy firmware uploads and debugging, the `ecghelper` features the same USB to serial converter as the Adafruit HUZZAH32 board used for the breadboard system, the Silicon Labs CP2104.[33]

This may seem like an odd choice given that the company recommends their CP2102N family for new designs. However, the CP2104 explicitly supports partial power down situations where no supply voltage is present for the CP2104 while the serial and status I/O pins, here connected to the ESP32, can still be on logic level voltages such as 3.3 V. The CP2102N[34] datasheet hints that this would still be acceptable, but also states $V_{io} + 2.5\,\mathrm{V}$ as the maximum allowed voltage for those pins.

Partial power down is an important feature for the `ecghelper` design. Without it, at least $100 - 200\,\mu\mathrm{A}$ CP2104 suspend mode supply current would have to be added to the standby power consumption of the device, according to the datasheet. The circuit design is based on the datasheet and the 'autoreset' circuit of the ESP32 board used in [Auer 2020], which makes it easy to initiate firmware uploads via the USB serial connection.

In contrast to the Adafruit HUZZAH32 ESP32 'feather' board, which has been used as reference, the `ecghelper2` design uses an integrated PNP (positive negative positive) transistor pair DDC114TU[35] by Diodes Inc. with built-in $10\,\mathrm{k}\Omega$ base resistors, allowing a smaller design with fewer individual components. The transistors are used for the auto reset feature also present in the evaluation board. When RTS and DTR have different logic levels, either RESET (actually the ENABLE pin of the ESP32) will be pulled down, or GPIO00 will be pulled down. The former powers down the ESP32, rebooting it as soon as the signal returns to logic high state. There is a R/C delay with $10\,\mathrm{k}\Omega$ and $1\,\mu\mathrm{F}$ connected to the pin. The latter signal is sampled when the ESP32 boots. A logic low state makes the ESP32 boot loader enter a serial firmware upload mode. A picture of the circuit, including power gating, is shown in figure 3.2, all schematics are shown in appendix A.3.

For maximum reliability, either the batteries or the USB port can be used to power the device, using TI LM66100[36] ideal diodes (quiescent current 150 nA) to select whichever has the higher input voltage at any moment. This makes sure that the main power supply is active when either USB or batteries are connected, avoiding partial power down and latch-up risks for components not designed for safe partial power down.

To keep standby power consumption low, the 3.3 V supply voltage for the CP2104 USB interface is only enabled when an USB bus voltage is sensed. This is done with help of a (slow) low-power Maxim Integrated MAX40203[37] ideal diode with enable input (quiescent current 300 nA). According to Maxim support, it is likely that the leakage current when disabled is as low as the reverse biased leakage current, which is 10 nA. This will become important for the third generation device (see section 3.15) display circuit described below. Ideal diodes are introduced in section 2.5

3.8 Display

The `ecghelper` device has to be able to give visual feedback to the user. While a simple LED would be sufficient to indicate a good (NSR, normal sinus rhythm) or bad (atrial fibrillation) outcome, it is desirable to be able to display the ECG while recording, the averaged ECG shape in analysis and other details of the signal analysis results.

[33] `https://www.silabs.com/interface/usb-bridges/classic/device.cp2104`
[34] `https://www.silabs.com/interface/usb-bridges/usbxpress/device.cp2102n-gqfn20`
[35] `https://www.diodes.com/assets/Datasheets/ds30345.pdf`
[36] `https://www.ti.com/product/LM66100`
[37] `https://www.maximintegrated.com/en/products/analog/analog-switches-multiplexers/MAX40203.html`

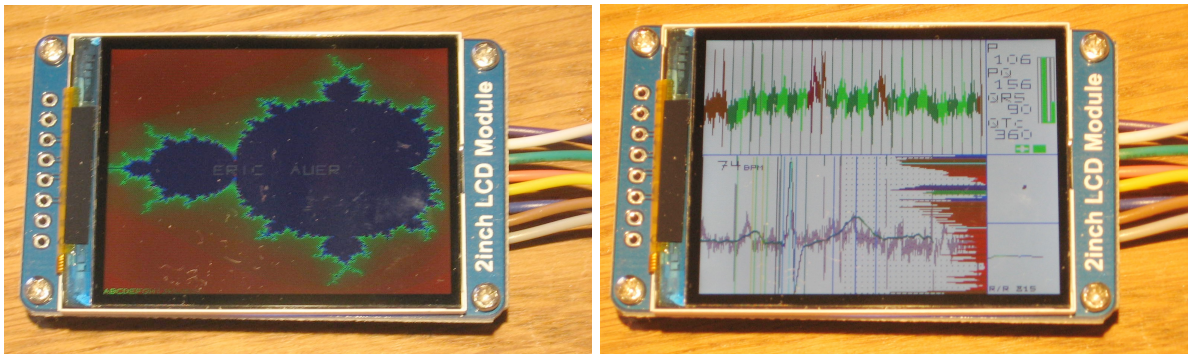


Figure 3.3: Experiments with the new, larger display in September 2020: A Mandelbrot fractal and an earlier version of the ecghelper user interface. The displays ship with cables to connect them to pin headers

Therefore, a relatively large, high resolution LCD module from Waveshare[38] has been selected for the device. It has a resolution of 240 × 320 pixels and a size of 2 inch (diagonal). Other options would have been the 128 × 160 pixel 1.8 inch or the 240 × 240 pixel 1.3 inch variants. However, the former would have a rather low resolution and the latter has a square shape which is impractical for ECG display. OLED displays have been considered, but were expensive at sufficient, yet still below 240 × 320 resolutions.

Unfortunately, the selected display was not available at the time from the local supplier `https://www.exp-tech.de/`, nor from the usually comprehensive supplies of Mouser, Digikey and Farnell, so displays for the prototype and the breadboard evaluation boards had to be ordered directly from Waveshare. While Exp Tech **plans** to sell the displays for less than 11 Euro each, postage, taxes and handling costs made the acquisition of the displays used for this thesis almost twice as expensive.

The controller used for the 320 × 240 and 240 × 240 pixel displays is a ST7789VW[39] with 40 µA standby current and 7.5 mA active current, counting all supply voltages, according to the datasheet. While it is possible to have lower I/O voltages, this is not implemented in the Waveshare modules and the ESP32 already uses 3.3 V, missing an opportunity to lower energy usage. The 160 × 128 pixel display would use a ST7735S with similar standby current, but only 2 mA active current, again according to the ST7735S datasheet. All voltages needed for the LCD itself are internally generated by the controllers and included in the power consumption considerations.

Interestingly, the main standby current leakage of the Waveshare modules is caused by a transistor circuit to control the backlight LED. Roughly 300 µA flow through a 10 kΩ pull-up resistor which keeps the backlight enabled when the backlight control pin of the module is left floating. By scratching a copper trace on the backside of the module, this feature can be removed, saving 300 µA of excess standby current. Now the backlight is only enabled when the control pin is actively pulled high by the ESP32. Luckily the schematics documentation[40] from Waveshare makes it easy to find the appropriate copper trace.

Omitting the pull-up resistor or using a FET instead of a bipolar transistor and a larger resistor value would be easy methods for Waveshare to improve their modules without having to change layout. The suggestion has been forwarded to Waveshare.

To achieve even lower standby currents, a proposed third generation of the `ecghelper` device – see section 3.15 – features yet another MAX40203 ideal diode with enable pin to enable the LCD power supply only when the backlight enable signal is active and TI SN74AUP3G34[41] low-power

[38] `https://www.waveshare.com/2inch-lcd-module.htm`
[39] `https://www.waveshare.com/w/upload/a/ae/ST7789_Datasheet.pdf`
[40] `https://www.waveshare.com/w/upload/e/ee/2inch_LCD_Module_SchDoc.pdf`
[41] `http://www.ti.com/product/sn74aup3g34?qgpn=sn74aup3g34`

triple buffer gates with partial power down support for the data and control signals to protect the display from latch-up while it is powered down. The static power consumption of those buffers is below 1 μA per chip, so the more complex circuit saves almost 38 μA in standby. Of course, this only works because the MAX40203 has a very low leakage current, far below 1 μA, while disabled, as mentioned.

While active, the displays use circa 32 mA according to own measurements. This suggests that the backlight LED runs at a nominal current of around 25 mA. It would be possible to use pulse-width modulation dimming in this generation of the `ecghelper`, as the display is a bit brighter than necessary. However, this would not work for the third generation, where the backlight control also controls the power supply of the controller.

3.9 RGB LED

The breadboard system used an array of NeoPixel[42] LED (see the documentation about the proof of concept `ecghelper` project system) for status and result display purposes. Those have already proven to be too bright and too power-hungry to use even in the USB powered breadboard system, which therefore only made use of between 0 and 12.5% of the available brightness.

In addition, a website[43] mentions that NeoPixel not only consume up to 34 mA or 53 mA per LED at full brightness, depending on version, but already 1 mA per LED even at zero brightness. Also, they require supply voltages of at least 3.5 or 3.7 V, depending on version, which would require additional power supply circuitry for `ecghelper`.

Clearly, NeoPixel are not suitable for the desired low power design. Instead, discrete Kingbright APHF1608SEEQBDZGKC[44] or APHF1608LSEEQBDZGKC[45] RGB LED are used. They are (almost too) tiny at 1.6×0.8 mm and feature sufficient brightness at low currents and low voltages: 15, 10 and 70 mcd at 2 mA and 1.8, 2.65, and 2.65 V respectively for the red, green and blue channels for the LSEEQ variant. The SEEQ variant offers higher brightness at 20 mA, but the same parameters at 2 mA, so either can be used. Both allow at least 20 mA continuous forward current.

To give a balanced spectrum while limiting power consumption, the red channels are used at 6 mA (1.9 V, 45 mcd, 220 Ω), the green channels at 1.5 mA (2.7 V, 50 mcd, 390 Ω) and the blue ones at 6 mA (2.9 V, 30 mcd, 68 Ω). This also means that it is most energy-efficient to use green to signal states which occur often.

The LED have common anodes and are connected to GPIO of the ESP32 through resistors to set suitable currents. The LED forward voltages are low enough to require no additional circuitry beyond that. The `ecghelper` contains two RGB LED to have enough flexibility in signalling, without using too many I/O pins to control them.

3.10 Audio output

One feature missing in the breadboard version was audio output. The ability to produce beep signals makes the device more user friendly and makes it easier to use for visually impaired users.

To generate beeps while keeping standby and active power consumption low, electromechanical speakers would be a problematic choice. Their coils need significant currents to drive and as inductive loads, they can be sources of power supply noise. Instead, `ecghelper` uses a relatively small PUI audio piezo transducer (SMT-1141-T-5-R[46]) with 5 V (max. 25 V) peak-peak rated voltage.

[42] `https://www.adafruit.com/category/168`
[43] `https://www.pjrc.com/how-much-current-do-ws2812-neopixel-leds-really-use/`
[44] `http://www.kingbrightusa.com/images/catalog/SPEC/APHF1608SEEQBDZGKC.pdf`
[45] `http://www.kingbrightusa.com/images/catalog/SPEC/APHF1608LSEEQBDZGKC.pdf`
[46] `https://www.puiaudio.com/products/SMT-1141-T-5-R`

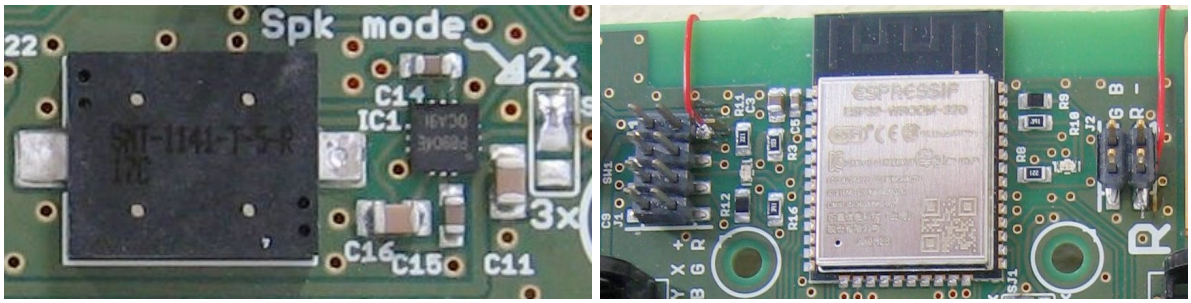


Figure 3.4: Left: Audio circuit with piezo transducer. Right: Core ESP32-WROOM-32D module with built-in antenna, tiny RGB LED on either side, debug & extension headers

Being capacitive loads, piezo transducers should not simply be wired to two GPIO of the ESP32. The charge present in the transducer could drive the GPIO beyond appropriate voltage levels, in particular when not switched at exactly the same time while flipping polarities.

On the positive side, the selected transducer has a rated current of less then 3 mA at resonant frequency (4 kHz) at the rated voltage, contributing to the overall low power design. The SPL at 10 cm distance is 73 dBA, which can be easily heard.

To be able to produce somewhat louder beeps at a larger range of frequencies, it is desirable to use higher drive voltages. Beeps at lower frequencies are easier to perceive for older users and generally sound more friendly according to the taste of the author.

Luckily, the Diodes Inc. PEM8904 18 V_{pp} output piezo sounder driver provides an integrated, low power solution to generate high drive voltages from a 3.3 V supply voltage. There are two versions: PAM8904[47], requiring at least 2.3 V supply voltage and using a 1 MHz voltage pump clock, and the newer PAM8904E[48], which already works with 1.5 V and uses a 100 kHz voltage pump clock. The `ecghelper` uses the latter, as lower pumping frequencies should mean less HF noise and because the no load current is lower ($\leq 220\,\mu$A depending on selected output voltage). More important than the no load current is the differential load current, which can be higher for the chosen PAM8904E, meaning it has been tested with more demanding piezo transducers of e.g. 47 nF instead of 15 nF for the PAM8904. Both chips have at least 40 mA short-circuit current.

A small disadvantage of the PAM8904E is that it takes up to 2 ms to activate, the PAM8904 taking less than 0.4 ms, but this can easily be compensated by starting to send control signals a bit earlier. Both chips have the very useful feature that they switch to 1 μA shutdown mode after 42 ms without a control signal. The ESP32 only needs a single GPIO to send square wave signals of suitable frequency to the chip, which can either use the supply voltage directly or double or triple it first with a built-in capacitive voltage pump.

The `ecghelper` design uses either 2× or 3× mode, selectable through a solder jumper. This means up to 18 V_{pp} at the transducer, as either of the pins of the transducer can be at almost 3×3.3 V, while the other pin is connected to ground. In testing, both modes were similar in beep volume, so the lower voltage mode should be used to conserve energy. The used transducer has 12 nF capacitance. A very rough estimate of the power consumption while producing beeps would be $8-15$ mA, based on the values in the datasheets.

3.11 Overall power consumption

The `ecghelper` has four main states of operation. Those are standby, ECG recording, result presentation and, if selected by the user, Bluetooth communication mode.

[47] `https://www.diodes.com/assets/Datasheets/products_inactive_data/PAM8904.pdf`
[48] `https://www.diodes.com/assets/Datasheets/PAM8904E.pdf`

Component	Standby mode	Offline operation	Bluetooth mode
MAX30003	$< 5\,\mu$A	$< 50\,\mu$A	$< 5\,\mu$A
AFE clock osc.	$10 - 15\,\mu$A	$10 - 15\,\mu$A	$10 - 15\,\mu$A
LDO 1.8 V for AFE	$2\,\mu$A	$2\,\mu$A	$2\,\mu$A
ESP32	$3 - 15\,\mu$A	$3 - 70$ mA	$100 - 260$ mA
ESP32 clock osc.	included	included	included
RGB LED	$0\,\mu$A	≤ 27 mA	≤ 27 mA
LCD Display	$< 40\,\mu$A	≈ 32 mA	≈ 32 mA
Audio	$< 5\,\mu$A	≤ 15 mA	≤ 15 mA
USB I/O, diodes	$0.6\,\mu$A	(< 20 mA, when used, here: $0.6\,\mu$A)	
Total current 3.3 V	$< 82.6\,\mu$A	< 146 mA	< 334 mA
Internal power	$< 273\,\mu$W	< 482 mW	< 1.102 W
Efficiency	$> 65\%$	$> 85\%$	$> 85\%$
DC/DC converter	$25\,\mu$A	$25\,\mu$A	$25\,\mu$A
Battery power	$< 495\,\mu$W	< 567 mW	< 1.297 W
Total current 3.0 V	$< 165\,\mu$A	< 189 mA	< 432 mA

Table 3.3: Expected power consumption in standby mode, while recording, processing or displaying ECG and analysis results, or in Bluetooth mode

In standby mode, the device has to be able to wake up at ECG electrode lead-on events. While recording ECG data, it would be possible to disable the display and use only the LED to provide feedback. The current firmware keeps the display on, to show a scope view of the incoming ECG while it is being recorded. In Bluetooth mode, receiving commands or waiting for them will use significantly less power than sending data. In that sense, Bluetooth mode has variable power demands. Recording and result presentation are similar in power consumption – both use peripherals and CPU, but no wireless I/O. Computation of the results is in part done while recording (filtering and QRS complex detection) and in part between recording and display (spectral analysis, ECG averaging, morphology metrics, feature vector based classification).

As the new analysis algorithm takes less than one second after the recording phase until results are ready, analysis is not considered a separate state in the power consumption estimates below. The ESP32 power consumption depends on CPU clock. At 240 MHz, up to 70 mA can be expected, while 31 mA are the maximum at 80 MHz according to the datasheet. The current firmware limits speed to 80 MHz for that reason. Even 40 MHz would be easily sufficient both for recording and processing. Activating Bluetooth can step up clock speeds.

Based on the estimates, the device could stay in standby for at least 4.8 months or do 3.5 hours of measurements before having to change batteries. In reality, power consumption should be lower. Modest CPU and peripheral device usage can achieve average currents far below the upper limit of the estimate given in table 3.3

On the other hand, the switching power supply cannot deliver enough current below a certain voltage, so the full capacity of the batteries cannot be used. In particular Bluetooth mode, having the highest power demands, will fail earlier. By using a converter chip with more favourable low voltage performance specs, batteries can last longer. However, this also causes more stress for rechargeable batteries. When the device keeps functioning until battery voltage is very low, the batteries could already be damaged from the too deep discharge.

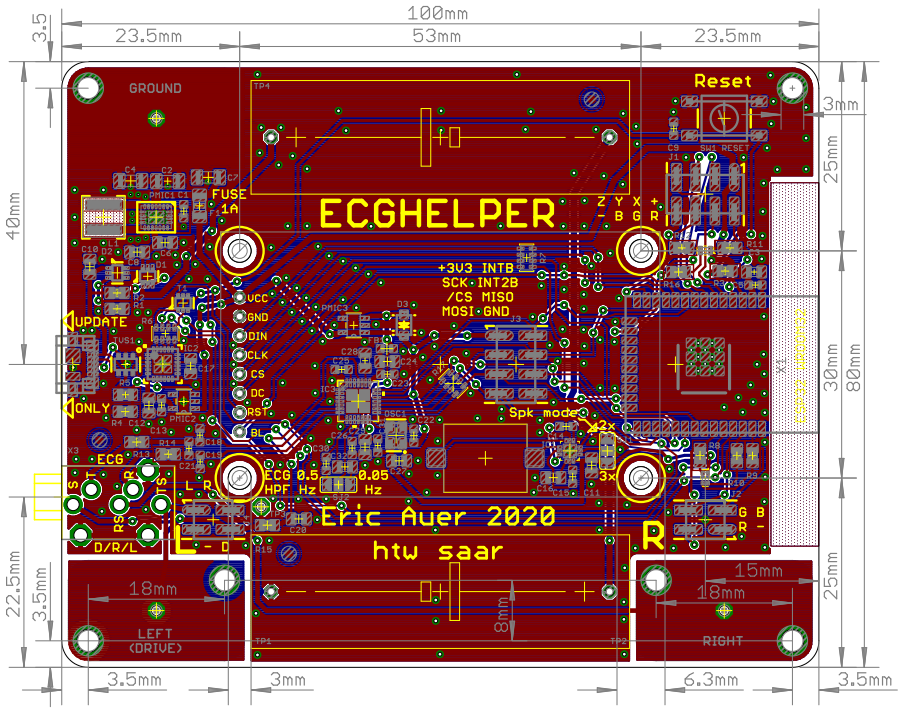


Figure 3.5: Layout of the ecghelper2 device, original, actual size, tstop layer hidden (use wire to connect RED1 & RED2 signals, see errata section 3.14)

3.12 Layout considerations

To be able to work on the layout with a freeware version of the EAGLE[49] layout editor (only for non-commercial and evaluation purposes), the `ecghelper` circuit board must use at most 2 copper layers (top and bottom) and must be at most 80×100 mm in size. As the display and batteries already are relatively large, the design starts with a board of exactly that size, with rounded edges.

The display is at the centre of the board, stacked on top of it and connected with a pin header. Screws and spacers keep the display at the right place. The battery holders are installed next to the long edges of the display. This leaves a H-shaped area of the board free to use for the remaining circuit. The area below the display is used for the actual ECG circuit and the audio circuit and transducer, giving both some protection from accidental touching, but also from EMI. The software avoids using audio while recording ECG, to not disturb measurements with the piezo drive signal.

Next to the display, the power supply and USB interface are placed on the left side and the ESP32 module is placed on the right side, partially covered by the display module. For good Bluetooth signal, there is a copper free zone at the right edge of the board, around the antenna of the ESP32 module. Note that the sheet electrodes also keep some vertical distance from the antenna to not interfere with the Bluetooth signal. Also, Bluetooth is disabled while measuring ECG, to avoid disturbing the measurement. Bluetooth is only optionally enabled after the measurement to share data and analysis results with other devices. Nearby the ESP32 module, at symmetrical locations, the two RGB LED for status signalling are located.

This leaves most edges and corners of the board free for the remaining task – electrical and mechanical connectivity. Two of the corners are used for screw connections to the sheet electrodes. The other corners also have screw holes, originally connected to circuit ground. As this was only useful for easy connection of measurement devices such as multimeters or oscilloscopes, a new version of the layout leaves the holes unconnected instead. The reset button is in the last corner.

[49] `https://en.wikipedia.org/wiki/EAGLE_(program)#License_model`

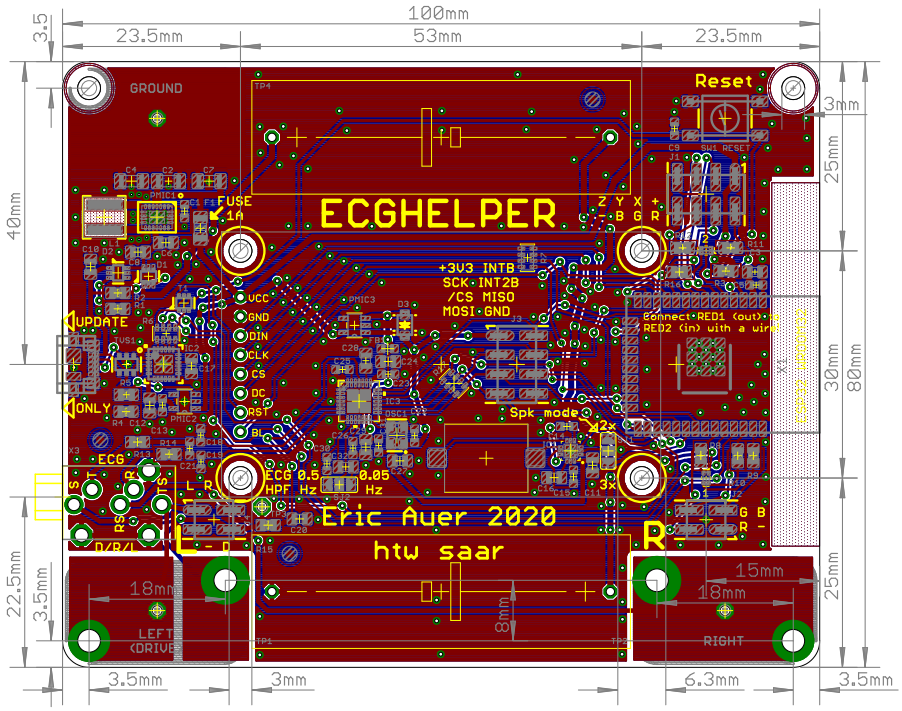


Figure 3.6: Layout of the ecghelper2 device, improved, actual size, tstop layer hidden (circuit diagram left unchanged – again use wire to connect RED1 & RED2 signals, see errata section 3.14)

While the right edge is reserved for ESP32 and Bluetooth, the left edge has the USB connector (only accessible by opening the side of the housing, not to be done by users) and a 3.5 mm TRS mini jack plug socket to connect external electrodes. The latter is accessible though a hole in the left side of the housing, without exposing other parts to dust etc.

The layout keeps most of the bottom copper layer as ground plane, as well as large parts of the top layer. Functional units are physically grouped on the board. Ground areas separate the units where appropriate. The main supply voltage is distributed mostly in a star topology, to reduce interference between functional units. The 1.8 V supply voltage for the MAX30003 is generated close to it, with a separate branch behind a ferrite bead for the analogue supply. Bypass capacitors are of course placed close to those chips which require them, including the power supply. The latter uses a shielded inductor, to reduce EMI, near a corner of the board, away from the ECG circuit.

As the ESP32 uses most power in the device, it could be an option to put the power supply close to that, further reducing noise and emissions, but it was acceptable to keep it close to the USB port, which also is one of the possible energy sources, for example while updating the firmware. Debugging and extension headers (for ECG electrodes, several GPIO and, only accessible while the display is not mounted, the MAX30003 signals) complete the design.

Care must be taken to not let the SPI header below the display touch the display module, which could cause short-circuits. The layout also had paste-in-hole soldering in mind, but this is **not** appropriate for most assembly workflows, so an updated version of the layout reverts that. The solder mask stencil ordered with the prototype PCB is the version for paste-in-hole.

When **not** using paste-in-hole, those stencil openings in the mask for the prototype design which would allow solder paste to be added to the vias for the display header, screw holes connecting to the sheet metal electrodes, battery holders and TRS mini jack plug socket **have to be covered before adding solder paste** if doing SMD and through-hole part soldering in separate steps.

The updated layout in figure 3.6 does not use paste-in-hole. In addition, it improves separation of middle voltage and left electrode screw hole terminals. Bus signal and ground layouts have been

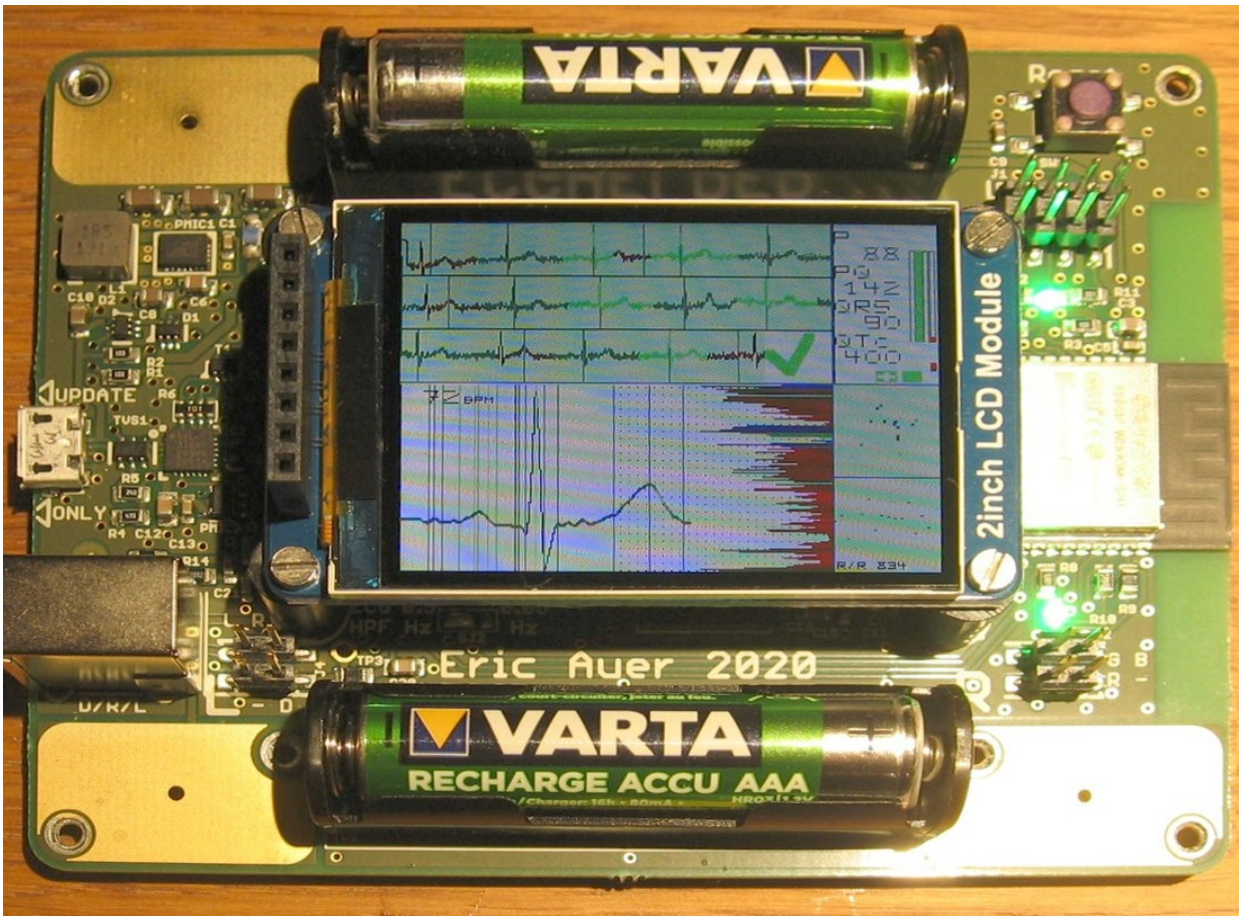


Figure 3.7: Early test of ecghelper2 with external wristband electrodes

improved. Almost all component placements are unchanged and the layout matches the original circuit diagram – including original flaws, which will be discussed in section 3.14.

3.13 ECG electrodes and device housing

To make the `ecghelper` fully integrated and easy to use, it must not use disposable electrodes. So instead, it uses sheet metal electrodes. As more specialised materials would not have been easily available, the device uses stainless steel. Stainless steel is well-tolerated by the body even certain implants have stainless steel parts in contact with body tissues ([Kramme 2011], chapter 57). Also, being a metal which does not rust, it has favourable polarisation properties. Using mixed metals such as copper for one electrode and zinc for the other, could create a large voltage offset disturbing the ECG measurement, as known from school experiments with potato, apple or lemon batteries. Metals such as nickel may cause adverse health effects such as allergies ([Menche 2007], page 223).

The original idea was to have electrodes of the size of the contact areas around the screw holes on the board, for making contact with the users thumbs and index fingers while holding the device. This would have allowed a very compact housing with cut-outs in those areas. On the other hand, the areas are somewhat small from an ergonomics point of view.

So instead, the entire device is fully enclosed in a housing made from plywood and acrylic glass. While plywood might seem somewhat improvised, it is easy to process and is not likely to pick up static electric charge. So this design avoids ESD issues. As the device is relatively large, using a 3D printer to make a custom housing in a simple cuboid shape would have seemed wasteful. Of course it is still important to make sure that sharp edges and corners are rounded off a bit.

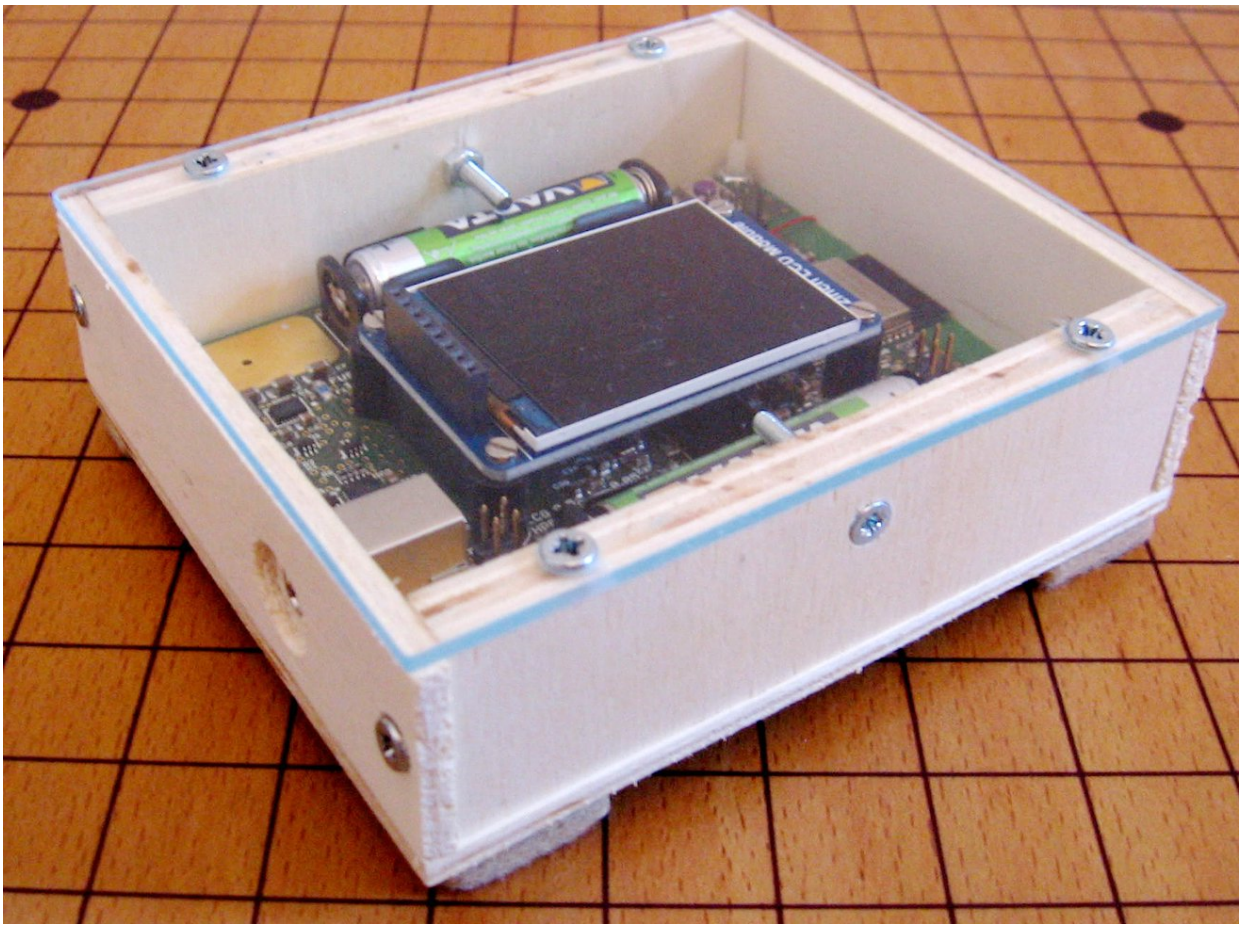


Figure 3.8: Plywood and acrylic glass test housing for the ecghelper2

To build the housing, 4 mm, but for the two longer walls 8 mm, plywood parts are used: 109×97 mm for the bottom and a same size acrylic glass part for the lid. An extra wood part can be used to cover the lid while transporting. Two parts, each 101×26 mm, are used for the long walls. The two short walls are 97×26 mm each. It is convenient to make the long walls by stacking two layers of 4 mm each and glue them together. The housing uses 12 countersunk-head screws (3×15 mm) and the electrodes are connected and fixed with 2 plastic and 4 metal (preferably stainless steel) M3 countersunk-head screws of 16 mm length. To secure the two batteries, two 20 mm M3 countersunk-head screws are used.

Each electrode screw uses M3 nuts on either side of the board, with **two exceptions**. The middle-voltage connection must use no – or a plastic – nut on top of the board, while the left electrode connection must use none on the bottom. In the improved layout and in the third generation device, the exception does not apply and all 12 nuts can be metal ones.

The housing is relatively tall, because a test socket header stands out of the top of the display module, making the SPI and other signals of the display easily accessible while the lid of the device is open. With a suitable cut-out in the acrylic glass lid, or by using a connector without test socket for the display, the device could be made more flat. That way, the lid for the device could already keep the batteries in place. In the prototype, a screw above each battery socket keeps the batteries from falling out in the event that the entire device is dropped on the ground by the user.

Having a housing around the entire device, without cut-outs to access internal electrodes, improves protection against dust and against making contact with other parts of the circuit while using the device. The only required opening is the one for the socket to connect (optional) external electrodes, which can be sealed so that dust can only reach the, electrically shielded, TRS mini jack plug socket

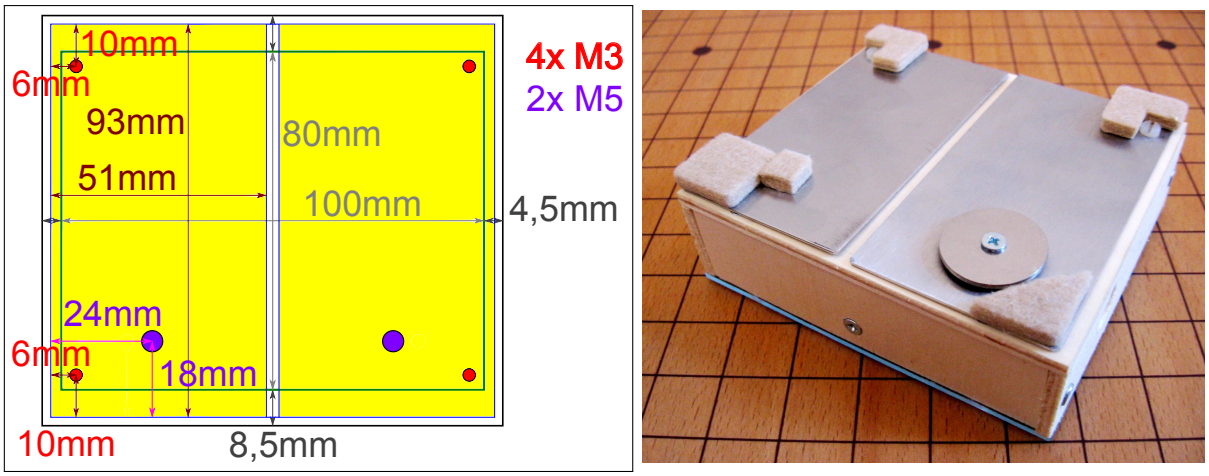


Figure 3.9: Stainless steel sheet metal electrodes at the bottom side of the housing, with a stainless steel washer for the reference (middle) voltage, dimensions and drill locations relative to the housing and photo of the actual prototype

itself. The sleeve connection of the socket can be touched by the user. This is equivalent to touching the middle voltage electrode / washer. The shield of the socket is connected to circuit ground, but can not be touched by the user while the housing is closed.

The electrical connection to the sheet electrodes is made through screws in two corners of the circuit board. This makes it possible to easily attach electrodes located at the bottom of the device. Outside the plywood housing, they are at sufficient distance from the Bluetooth antenna, so they can use almost the entire surface of the device bottom, one electrode each for left and right hand.

Users simply hold the device to conduct a measurement cycle. They either keep it in both hands at the end of the measurement until the firmware activates the Bluetooth connection. Or they opt against Bluetooth transfer by taking one hand away while results are shown, or by putting the device on a table in that phase.

Felt pads at the corners of the device bottom prevent scratches and short-circuits when laying down the device on a conductive surface. Otherwise, the short-circuit between the electrodes as false lead-on state would trigger measurement cycles, wasting battery power.

With batteries, housing and the relatively thick stainless steel electrodes made from sheet metal by the university workshop, the device has an overall weight of 235 grams, of which 75 for the two electrode plates at the bottom, each 51×93 mm in size. As described above, some screw holes are connected to circuit ground in the prototype, so isolated plastic screws have been used for those.

Exact electrode dimensions are shown in figure 3.9. Note that some holes have to be large enough for M5 screws, while only M3 screws pass through them. The M5 hole for the right electrode (when looking at the display – left electrode when looking at the bottom of the device or looking at the drawing!) simply lets another M3 screw with the same signal pass through the electrode, for mechanical stability. A washer can be used here if needed, or the right electrode could use 3 mm drilling for all holes. This is not the case in the prototype, to simplify electrode manufacturing by having two identical electrodes. One is used with one side up, the other one with the other side up.

In addition, one screw hole connects to the middle voltage of the ECG amplifier of the MAX30003. One large washer is screwed to that, allowing the user to make touch contact with that centre voltage if the lead biasing (internally connecting the main electrodes to centre voltage through large resistors) is not sufficient to keep DC offsets small. It is important that the washer also is made from stainless steel. A zinc plated washer would actually create a DC offset instead of reducing DC

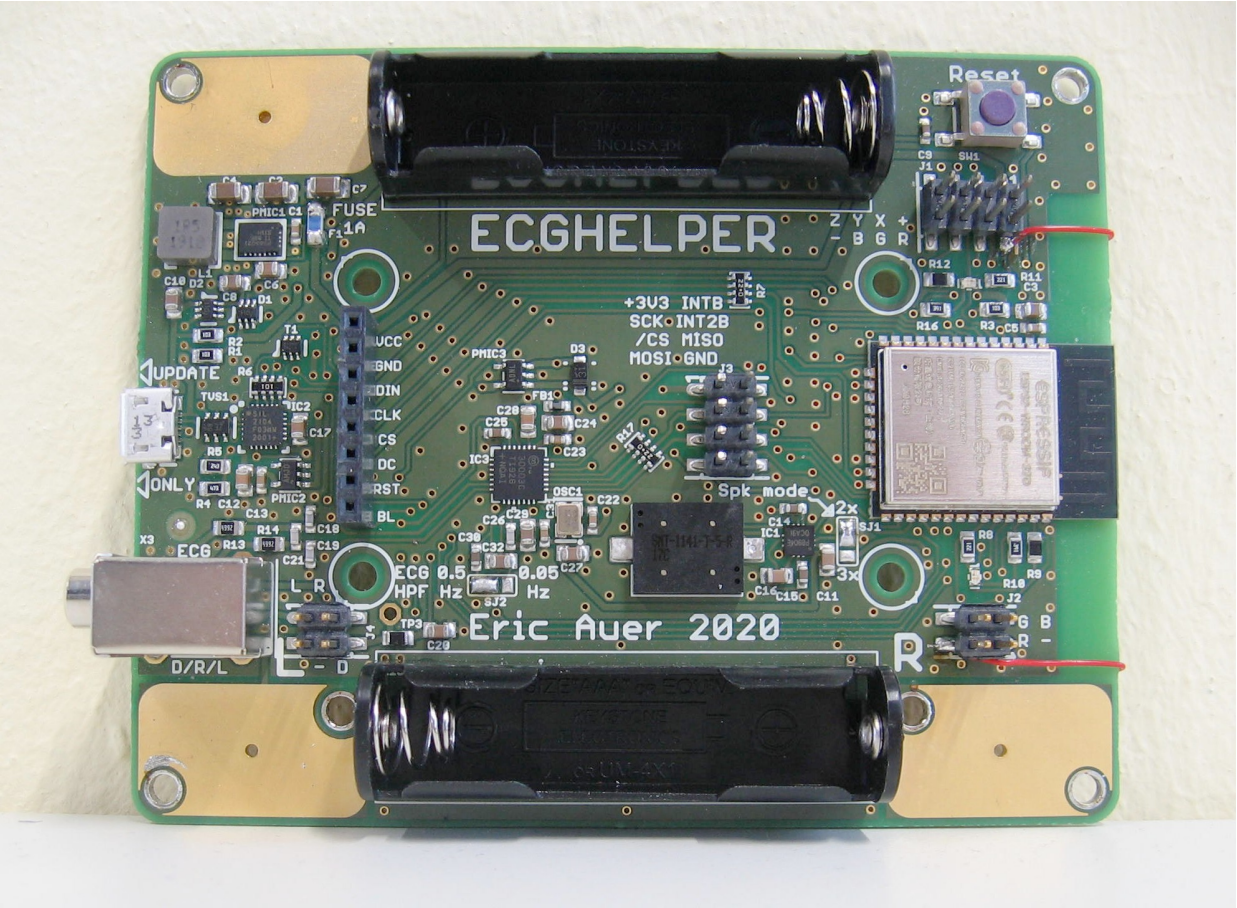


Figure 3.10: Bugfix: Connect RED1 and RED2 RGB LED signals, red wire routed below the board for aesthetic reasons and away from the Bluetooth antenna to avoid interference

offsets when the user touches it and the stainless steel main electrodes at the same time (similar to the lemon battery effect explained above).

The washer is connected to the screw hole at the bottom left side of the board, when looking at the display, which is further away from the corner. The hole closer to the corner is for the left electrode signal. This is why the sheet metal electrodes deliberately have too large hole diameters in the non-corner locations. A M3 screw can pass through the electrodes without making electrical contact. Of course, thin plastic washers have to be used to prevent the middle voltage electrode washer from touching the electrodes.

For testing, it is possible to glue aluminium tape to the bottom of the housing, instead of using steel sheets. Care has to be taken to use non-isolated tape. As the screws make contact on the side which will be touched by the user, the conductivity of the sticky side can be arbitrary. A suitable choice can be the kind of tape used for attic insulation. Of course, this can not be combined with a washer made from another material for the middle voltage, but as mentioned, touching that signal is optional for simple ECG measurements.

3.14 Prototype assembly and errata

The EAGLE design was sent to Aisler (`https://www.aisler.net/`) for manufacturing of boards and solder paste stencil. Components got ordered from Mouser (`https://www.mouser.de/`), with the exception of the displays, which had to be acquired directly from Waveshare (`https://www.waveshare.com/`). Mechanical components were either available at the university lab or bought from a local DIY (do-it-yourself) store.

As the design heavily relies on SMD components, although care had been taken to avoid parts with pitch below 0.5 mm, it was important to have good assembly tools.

Thanks to the support of Prof. Dr. Matthias Scherer and his colleague Klaus Stöß, it was possible to conduct the main SMD assembly step at their lab at the Trier University of Applied Sciences. Thank you! It would have been very hard to build a prototype without their professional equipment, including a SMD pick and place workplace[50], solder mask stencil handling and vapour-phase soldering. Otherwise, it might have been necessary to use a contract manufacturer.

Directly after the assembly, the prototype was ready for testing with the already prepared firmware. However, some bugs became apparent at that point. First, the electrode polarity differs from the breadboard setup, which is easy to compensate by either changing the MAX30003 polarity configuration bit or by sign-flipping the recorded amplitudes.

Next, it turned out that it was impossible to activate the red channel of one of the RGB LED. This is caused by an error in the circuit, as the used GPIO is input-only. As a workaround shown in figure 3.10, a wire can be soldered to the board which connects the red channels of both RGB LED to each other – at the GPIO level, so each LED still has separate resistors. This limits the possible light signal spectrum, as both RGB LED show the same red brightness now, but this is not seen as a serious shortcoming for the purposes of the `ecghelper` firmware. The input-only GPIO simply reflects the current (red) LED status now. Note that the wire is routed mostly away from the copper-free zone, to avoid Bluetooth interference. It can be placed below the board or below the display, for aesthetic reasons.

A more severe issue was a short-circuit in a prototype. Only back in Saarbrücken, it became apparent that the MAX30003 SPI bus debug header was exactly touching a pin of a component on the bottom side of the display module. By coincidence, the two signals were ground and the 3.3 V supply rail. So instead of just preventing data exchange between ESP32 and MAX30003, the short-circuit destroyed the SMD power supply fuse. Shortening the debug header pins slightly resolved the issue. It turned out that the SMD fuse is at an inconvenient location for replacement, so one of the few placement changes of the improved layout is to slightly move that fuse.

The audio feature was not working at first. It was necessary to configure the used GPIO as normal (not real-time clock managed) pin to make the PWM (pulse-width modulation) feature of the ESP32 able to send square waves of selectable duty cycle and frequency to the audio circuit. Duty cycle simply is 50% here, but it could be modified to manipulate the volume of the audible beeps.

The prototype board has built-in thumb electrode areas for testing, with ENIG (Electroless Nickel Immersion Gold) surface finish, which is an extremely thin layer of gold on top of a nickel layer which ensures flatness. Compared to standard solder (usually lead-free tin and silver based alloys with some other metals) coating, the surface is more flat. While gold could be a good electrode material, the bulk nickel layer is an allergen and easily exposed below the thin gold layer. Because of this and because the electrode areas on the board are quite small, further testing usually took place using the set of external wrist electrodes already acquired for the breadboard system.

Later, the housing was built, using materials from the local DIY store and equipment such as a PCB board cutter circular saw at the university lab. While waiting for the stainless steel electrodes from the university workshop (thanks to my supervisor for helping with ordering and logistics!) the housing could already be tested with the aluminium tape trick mentioned earlier.

At first, the stainless steel electrodes were impossible to use together with the centre voltage washer electrode. Measurements were okay as long as the washer would not be touched. It turned out that the washer had a zinc plated surface, causing a large DC offset when used together with the stainless steel electrodes. The MAX30003 went into recovery mode and the software immediately tried to

[50] Essemtec Expert Finepitch system with air suspended pick-and-place head, Microplacer for exact placement of fine pitch components etc.

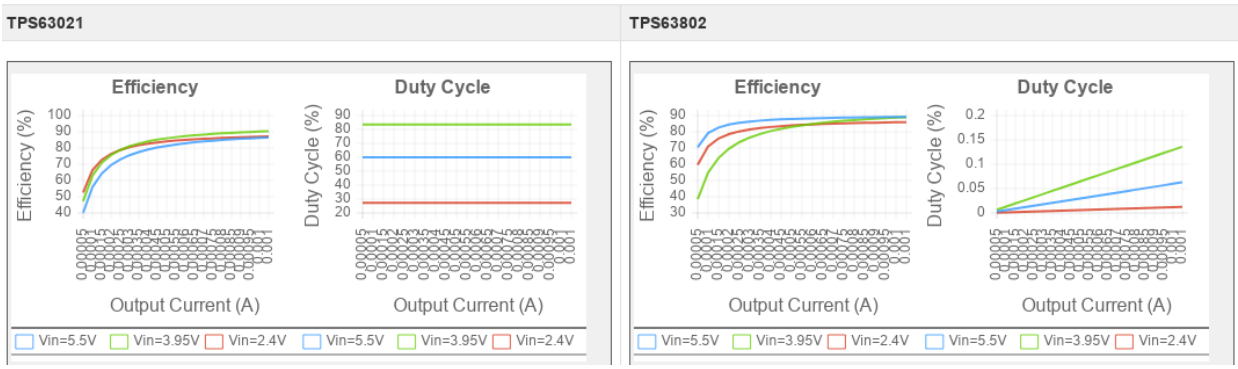


Figure 3.11: TI Webench comparison of TPS63021 to TPS63802 at very low loads (axis labels as generated by the Webench website)

restart the recording. The polarisation difference, when measured through a high impedance multimeter, was found to be as high as 0.6 V (one electrode +0.4 V when touching the other multimeter lead tip as reference, the other −0.2 V. Also, the screws used to connect the electrodes are probably zinc plated as well. As stainless steel washers were available at the local DIY store, the problem has been solved much faster than it took to find the source of the problem. The screws have not been replaced. They can be covered to prevent touching other conductors than the stainless steel ones.

The completed device now works as expected, apart from having worse battery life with rechargeables than would have been desired: NiMH battery voltages drop below the level at which the power supply can generate stable 3.3 V at high currents, such as those needed for Bluetooth transfers, relatively early. So, in particular when using Bluetooth mode, NiMH batteries have to be taken out of the device for recharging after fewer measurement cycles than expected. An obvious workaround is to use disposable alkaline batteries, which have higher cell voltages.

3.15 Third generation device

While the `ecghelper` device already is expected to have a low enough standby power consumption to offer several months of standby operation, it seems appropriate to investigate further optimisations.

A large part of the standby power consumption of the current prototype is expected to be caused by the display controller. To avoid that, the proposed next generation circuit uses an ideal diode with enable pin for the display controller supply voltage as a low power, low voltage drop switch. The control and data signals for the display are passed through buffers with partial power down support, to protect the controller from latch-up when the ESP32 sends signals while the supply voltage for the controller is disabled. As described in section 3.8, this reduces standby current draw from the 3.3 V rail by up to 38 µA.

The second way to reduce power consumption is to change to an even more energy efficient switching voltage converter chip. Using a TPS63802 based instead of a TPS63021 based supply, idle current is reduced by an additional 14 µA and the improved efficiency of 75% instead of 55% around 100 µA load will further reduce the power consumption of the overall device. In addition, the new design uses the open-drain power good signal to force a reset when the generated supply voltage drops below ≈ 3 V. This also is keeps the device in reset until the supply voltage has reached ≈ 3.15 V.

For improved battery awareness, the new design uses a TI TLV522[51] dual nanopower op-amp for battery sensing. With high impedance input voltage dividers and low-pass filtering, the entire circuit uses less than 2 µA to forward half of the input battery voltage (total and, by adding suitable wiring to the battery holder, single cell) to two GPIO of the ESP32, which has built-in 12-bit analogue

[51] `https://www.ti.com/lit/ds/symlink/tlv522.pdf`

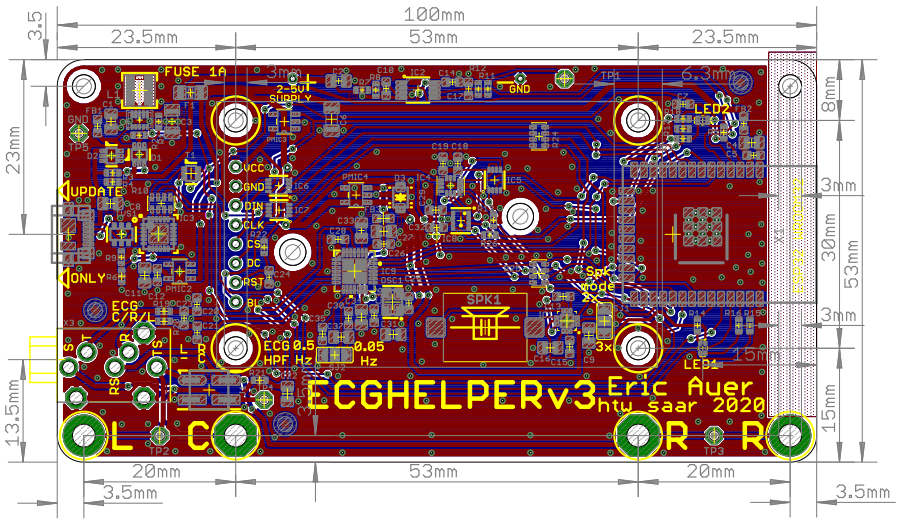


Figure 3.12: Layout of a proposed ecghelper3 device, actual size

digital converters (ADC). Without the extra circuit, the prototype presented in this thesis could only measure the voltage on the 3.3 V rail itself via ESP32 introspection A/D channels. The ESP32 also has a built-in Hall sensor – which could be used as trigger for hidden menus etc.

Most headers are only present in the prototype for debugging and testing purposes. Omitting them, as well as the reset button, helps to make the device smaller. As no contact areas beyond the screw holes will be needed, the electrode connection can also be designed in a more space efficient way, while still maintaining mechanical stability. Still using screws, not wires.

Apart from the area below the display module, a lot of board space is used by the battery holders. Using one single dual battery holder on the bottom side of the board instead of two individual holders on the top side allows making the layout significantly smaller. While this makes the device taller, it also serves as an extra placeholder between the electrodes and the Bluetooth antenna. Also, part of the extra height is compensated by omitting the display test header which stands out as tallest point in the prototype. Combining all changes, the new layout requires only a 100×53 mm board, less than two thirds of the original size. It also uses slightly smaller resistor form factors, which are typically unlabelled, but still easy to handle.

In spite of the reduced size, there still is enough space below the display module to add two extra sensors. One sensor is a Bosch BME280[52] sensor for temperature, humidity and air pressure. The other sensor is a Bosch BMX160[53] sensor for inertial measurements (accelerometer and gyroscope) and geomagnetic field (three-dimensional compass).

Both sensors have exceptionally low standby currents of $\leq 4\,\mu$A each. Active power consumption is ≤ 1.6 mA for each sensor, or much lower depending on which measurement modalities are active. Both SPI and I2C communication are supported. Here, SPI is used. The acceleration sensor also features a low power ($30\,\mu$A) step counter mode, which triggers interrupt signals on significant movements. This could also be used to wake up the `ecghelper` from standby mode.

The BMX160 and the MAX30003 share interrupt connections to two ESP32 GPIO. If one of the two is programmed to generate CMOS (here: push-pull) interrupt signals, the other must be programmed to **not** generate interrupt signals for the same GPIO. It is possible to let **both** chips generate interrupts on the same GPIO by programming both to use open-drain signals and program either the MAX30003 or the ESP32 to activate a built-in pull-up resistor for the signal. This

[52] https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/
[53] https://www.bosch-sensortec.com/products/motion-sensors/absolute-orientation-sensors/absolute-orientation-sensors-bmx160.html

creates a wired-or of two active-low interrupts, sharing one GPIO. The sensors also share MOSI and clock with the MAX30003, but share their own MISO line, separate from the MAX30003 MISO.

This reduces the risk of conflicts when reading from the SPI bus if chips are selected in parallel, at boot, or caused by software bugs. To have enough distinct chip select signals for all sensors, one additional triple buffer is used. Both Bosch sensors have internal pull-up resistors at their chip select inputs. The buffers allow connecting them to strapping pins of the ESP32 without changing the default weak pull-up or pull-down state of the ESP32 boot loader. At boot, the ESP32 samples logic levels from several strapping GPIO pins to select the desired boot mode, so it must be prevented that the pull-up resistors in the sensors change the settings away from the defaults of the ESP32.

While having more features, the smaller version of the device would use 29 μA less in standby on the 3.3 V side, 14 μA less idle current for the power supply itself and it would further reduce standby current by improving efficiency already at the lowest loads.

Having the ability to process environmental measurements makes it more worthwhile to have an ECG device with a large display at home and it can motivate the users to have their ECG analysed more often. The BME280 can be used to present room climate information to the user and to process this information along with the ECG, for example when uploading the combined data to a virtual diary via Bluetooth.

This in turn makes them more aware of the battery status, since it becomes less likely that the ECG device is just sitting in a drawer until the user feels a suspicious heart rhythm, only to notice that they first have to replace the depleted battery before they can record it.

The BMX160 serves a more physiological purpose. It can record user movements during ECG recording sessions, for example to estimate breathing rate. Alternatively, in particular when combined with external electrodes attached to the user for hands-free measurements, the sensor can be used to measure bodily activity while recording ECG data, or to measure movements which could relate to artefacts, such as walking or running, or to pathologies, such as muscle tremor. Also, as [Lahdenoja 2018] demonstrates, movement data can even be used as a basis for A-Fib detection.

As with the second generation prototype, the circuit diagram of the proposed third generation device can also be found in the hardware appendix A.4.

3.16 ESP32 pin assignments

To stay close to the original breadboard system and to simplify the adaptation of the firmware, the new `ecghelper2` board uses largely the same ESP32 GPIO mapping as the old system. Of course there are differences, but interestingly, it is possible to let the firmware detect on which hardware it is running and change settings accordingly[54].

Table 3.4 compares the GPIO pin assignments of all generations of the device. Note that because GPIO 36 cannot be used as output, red 1 and red 2 LED signals have to be connected both to GPIO 21 to work around the bug in the circuit to attempt using GPIO 36 for red 2. The third generation circuit uses different red 2 pins. The extension headers in the prototype also connect to all LED signals, ground and 3.3 V supply voltage.

The reset signal is shared by display and ESP32 and can be triggered via USB. The ESP32 cannot trigger the signal itself, but the display can also be reset by sending the appropriate command via SPI. Also note that GPIO 0, 2, 12 and 15 are strapping pins. Their logic levels are sampled by the ESP32 ROM at boot to configure initial states such as whether or not to enter serial firmware upload mode. While the breadboard device uses a 96 × 64 pixel OLED display with a different controller ([Auer 2020]), newer versions of `ecghelper` use a 320 × 240 pixel LCD.

[54] See the appendix A for the newer designs and [Auer 2020] for the breadboard system

Functional module or chip type	Signal name or description	Signal direction	Bread-board	Proto-type	3rd gen-eration
Boot Control	!Update Mode, P.U.	to ESP32	**0**	**0**	**0**
Boot Control	!Update Mode, P.D.	to ESP32	**2**	**2**	**2**
Boot Control	SDIO Mode 1, P.U.	to ESP32	**5**	**5**	**5**
Boot Control	!3.3 V Flash, P.D.	to ESP32	12	12	12
Boot Control	Boot Debug, P.U.	to ESP32	**15**	**15**	**15**
System Display	Display Clock	from ESP32	25	25	25
System Display	Display Data	from ESP32	26	26	26
System Display	Data/!Command	from ESP32	4	4	4
System Display	Display !Chip Select	from ESP32	17	17	17
System Display	Backlight Enable	from ESP32	(n/a)	32	32
MAX30003 ECG	SPI Clock	from ESP32	**5**	**5**	**5**
MAX30003 ECG	SPI MOSI	from ESP32	**18**	**18**	**18**
MAX30003 ECG	SPI MISO ECG	to ESP32	19	19	19
MAX30003 ECG	!Chip Select ECG	from ESP32	**15**	**15**	**15**
MAX30003 ECG	Interrupt B	to ESP32	**33**	**33**	**33**
MAX30003 ECG	Interrupt 2B	to ESP32	(27)	**27**	**34**
Battery Voltage Sense	3.7 V/2 Analogue	to ESP32	**35**	n/a	n/a
Tied to GND	(by ECG board)	to ESP32	**21**	n/a	n/a
NeoPixel LED Array	LED Data	from ESP32	**16**	n/a	n/a
Extension Header	Ana./Dig. Input 1	to ESP32	(any)	**34**	n/a
Extension Header	Ana./Dig. Input 2	to ESP32	(any)	**35**	n/a
Extension Header	Ana./Dig. Input 3	to ESP32	(any)	**39**	n/a
RGB LED 1	!Red LED 1	from ESP32	(**!13**)	**21**	**21**
RGB LED 1	!Green LED 1	from ESP32	n/a	22	22
RGB LED 1	!Blue LED 1	from ESP32	n/a	23	23
RGB LED 2	!Red LED 2	from ESP32	n/a	**21 (36)**	**27**
RGB LED 2	!Green LED 2	from ESP32	n/a	**13**	**13**
RGB LED 2	!Blue LED 2	from ESP32	n/a	14	14
Basic Audio Output	Square Wave	from ESP32	n/a	**16**	**16**
BMX160 9-Axis Sensor	SPI Clock	from ESP32	n/a	n/a	**5**
BMX160 9-Axis Sensor	SPI MOSI	from ESP32	n/a	n/a	**18**
BMX160 9-Axis Sensor	SPI MISO BM...	to ESP32	n/a	n/a	**35**
BMX160 9-Axis Sensor	!Chip Select BMX	from ESP32	n/a	n/a	**2**
BMX160 9-Axis Sensor	Interrupt 1	to ESP32	n/a	n/a	**33**
BMX160 9-Axis Sensor	Interrupt 2	to ESP32	n/a	n/a	**34**
BME280 Environment	SPI Clock	from ESP32	n/a	n/a	**5**
BMX280 Environment	SPI MOSI	from ESP32	n/a	n/a	**18**
BMX280 Environment	SPI MISO BM...	to ESP32	n/a	n/a	**35**
BMX160 Environment	!Chip Select BME	from ESP32	n/a	n/a	**0**
Battery Voltage Sense	3.0 V/2 Analogue	to ESP32	n/a	n/a	**36**
Battery Voltage Sense	1.5 V/2 Analogue	to ESP32	n/a	n/a	**39**

Table 3.4: ESP32 GPIO pin assignments for all generations of the ecghelper device. GPIO 21 and 36 connected in prototype, see text. Boot control GPIO can be re-used after boot. Prefixed ! marks active low signals, P.U. / P.D. use 45 kΩ Pull Up / Down during boot

4 Software design

The software version written for this thesis builds upon the proof of concept project version described in [Auer 2020]. It replaces the slow, simple periodogram of derivatives from the proof of concept system with proper spectral analysis, using a third party Fast Hartley Transform (similar to Fast Fourier Transform, but using real-valued input and output data) library function – see section 2.2.

Spectral properties are processed not only for the overall recording, but also for R-wave aligned beat snippets. This also allows analysis of the phase stability across beats, as a function of frequency.

While the project software only had minimal R/R interval display, the current software also generates some Heart Rate Variability (HRV) style metrics from the R/R distribution. All ECG spectra are subjected to a set of metrics generation steps, such as finding, counting and listing the strongest peaks, or computing statistical properties of the spectra. The R/R metrics are inspired by [Nabian 2017]. Thanks go to Prof. Dr.-Ing. Dara Feili for the recommendation.

Using R/R metrics, spectral metrics and ECG morphology metrics, `ecghelper2` creates a feature vector of in total 256 properties from any given ECG recording (see section 4.2. Based on feature vectors of thousands of 10 second ECG recordings from the PTB-XL corpus ([Wagner 2020a, Wagner 2020b], see section 4.6), the new software has been trained (sections 4.8, 4.10) to reliably distinguish recordings of Normal Sinus Rhythm (NSR) ECG, including bradycardia and tachycardia examples with otherwise normal rhythm, from recordings of Atrial Fibrillation (A-Fib) or Atrial Flutter (A-Fl).

In line with the project software, the current software essentially treats A-Fib and A-Fl as symptoms of the same pathology, so in the rest of this section, A-Fib will refer to either of the two cases, unless explicitly noted otherwise. The classifier output will be either A-Fib ('bad') or NSR ('good' or in less obvious cases 'okay') for most recordings.

The software is also able to classify recordings as too noisy (or too unusual) to classify into either NSR or A-Fib. This result is called 'noisy' (or n/a). This decision does not use training data, as there is no category for unusable recordings in the PTB-XL corpus. Instead, it uses a combination of manual checks and testing whether the current ECG recording has all features inside the feature space volume spanned by most, or even all NSR and A-Fib training data.

However, the Telehealth Corpus [Khamis 2016], which can be a good stress-test for QRS detection algorithms, deliberately has a large share of low-quality recordings. So testing `ecghelper`'s algorithm, here using the `offlineecg2` Linux implementation of it, with this corpus can give some indication of whether the software is able to 'know (or estimate) whether it cannot know'. Note that the ESP32 embedded version of the software is now called `ecghelper2` to distinguish it from the project version.

4.1 Embedded system usage and training workflow

The embedded firmware aims to be very easy to handle for the user. As soon as the device is held with both hands, a circuit between left and right electrode is closed, which is detected by the ECG AFE chip. This generates an interrupt, waking up the ESP32 controller from a low power standby state. The firmware is booted and shows a test pattern and version information. Other user interface hardware (RGB LED and audio output) are tested at this stage, too. Then, a self-calibration cycle is entered for a few seconds, to express raw measurement data as absolute voltages later.

Once self-calibration has finished, the device starts displaying the user's ECG while it is being recorded. Flashing LED indicate detection of QRS complexes (R/R events) and progress bars show how much of the recording buffer and how many R/R events have been collected yet. If the user breaks the circuit by releasing the electrodes, the measurement is aborted and the device either

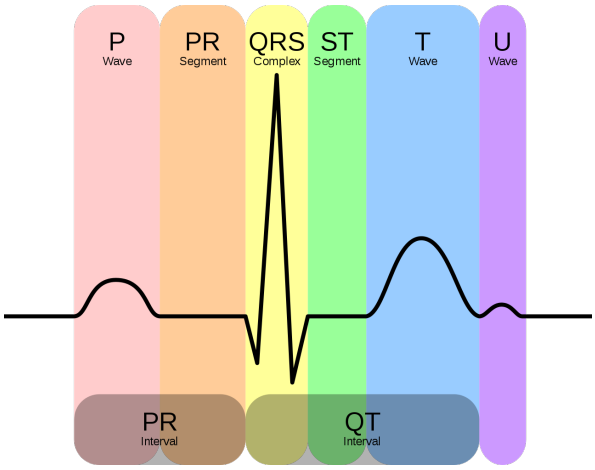


Figure 4.1: Schema of normal ECG with main wave and segment labels. Source: Hank van Helvete on Wikimedia.org, CC-BY-SA license

returns to standby or, if enough data has already been collected, proceeds as if it had completed the measurement.

If the user keeps providing ECG to the device by holding it, the device will record the ECG until a pre-set number of heartbeats have occurred, or the recording buffer is full, whichever happens first. Then, the device quickly analyses the ECG and displays the results – a graphical summary of the recording and of the averaged ECG shape, a Poincaré Plot and a plot of the R/R intervals, several bar graphs revealing values of processed features and of intermediate classification results (for advanced users) and, importantly, icons which indicate the final classification result.

There are icons for A-Fib (red cross), NSR (green tick) and for failure to classify (purple star, e.g. low ECG quality or unknown pathology). The RGB LED of the device light up in colours matching the icons and each result has a specific audio signal associated to it, so the device can easily be used without seeing the display.

For training and corpus analysis purposes, the algorithm also exists as a Linux application. This version will take one or more raw ECG data files (signed 16-bit, 500 samples per second, Einthoven lead I) as input. For each file, features are generated and the classification is performed. Various intermediate results such as key features, as well as the classification results, can be logged as part of the process – either to standard output (Linux) or through an USB serial connection (ESP32).

The user can optionally select several file creation options. The tool can save plots (same style as the result display of the embedded device) in TGA file format or in the older PCX file format, creating one graphics file for each ECG input file. Names of the output files are derived from the names of the input files. Another option will append the binary (256 unsigned bytes per ECG file) feature vectors to a feature log data file.

Both TGA and PCX support a fast, simple compression algorithm to reduce disk usage significantly for typical plots: RLE (Run-Length Encoding), which [Smith 1998] mentions in chapter 27. The exact algorithms used in the `offlineecg2` implementation follows the specifications of and documentation about the respective graphics file formats[55].

By collecting the feature vectors of all ECG in different corpora, scripts written for this thesis can then process the differences between the cases (here: A-Fib versus NSR) to generate rules and transformations in form of C source code. This source code is used as part of the classification code in both the Linux (`offlineecg2`) software and the embedded system `ecghelper2` firmware.

[55] See in particular `http://www.martinreddy.net/gfx/2d/PCX.txt` and `http://www.gamers.org/dEngine/quake3/TGA.txt` and the imagefile.cpp source code

ECG wave	Features
P-Wave	Height, onset time, offset time, duration
PR Interval 1	Interval from P-onset to R-peak: PR time
PR Interval 2	Interval from **P-onset** to Q-onset: PQ time
PR Segment	Interval from **P-offset** to Q-onset: **Not** used
QRS Complex 1	Q-onset time, Q-peak time and amplitude, R amplitude,
QRS Complex 2	S-peak time and amplitude, S-offset time, QRS duration
ST Segment	Average amplitude (relative to baseline)
T-Wave	Height, onset time, offset time, duration
QT Interval	Here from Q-peak time to T-wave offset
QTc Interval	Derived from QT interval, using Sagie correction
U-Wave	**Not** detected
Extra Waves	Detected before / after P-wave and T-wave

Table 4.1: Shape metrics extracted from the averaged ECG: All times are expressed relative to the R-wave peak, heights are amplitudes relative to the onset and offset amplitudes of the same wave

For bootstrapping, any earlier version of the generated source code can be used. The classification results will be wrong, but the software still generates normal feature vectors, which can be used to create the optimised source code in the next iteration. Two scripts have to be run to fully update the source code. The output of `featurerules.m` is the `rules.h` file and the output of `pca_lda.m` is the `ldachoice.h` file. Details of the system are described in the following sections.

4.2 ECG metrics and classification workflow

Each recording is segmented into individual heart beats, anchored around the QRS complexes. If possible, the R-wave is used for this. If it is much weaker than the S-wave, the latter is used to detect the QRS complexes initially and the R-wave is pinpointed later. See [Kramme 2011] chapter 10 and [Menche 2007] chapters 14 / 15 for introductions to ECG and the cardiovascular system.

By selecting a subset of beats which are most similar to each other and which have the most coherent RMS (root mean square) energy, the software then generates a prototypical ECG of a single beat, averaging the best recorded beats. This significantly improves the signal to noise ratio, as noise will not be phase-aligned to the beats, averaging out, while the actual beat ECG will be very similar across the beats, so the average will preserve that. Figure 4.1 shows a schema of a typical single beat, single channel ECG.

As discussed in [Auer 2020], recordings with A-Fib tend to have reduced stability of the P-wave when looking at R-aligned averages across beats. The software extracts the metrics presented in table 4.1 from the averaged ECG shape.

The Sagie corrected QTc is $\mathbf{QT} + (0.154 \times (1000\,\mathrm{ms} - \mathbf{RR}))$ with RR being the average R/R interval in milliseconds. Another, somewhat outdated, correction would be Bazett ($\frac{\mathbf{QT}}{\sqrt{\mathbf{RR}}}$). This and the Friderica correction ($\frac{\mathbf{QT}}{\sqrt[3]{\mathbf{RR}}}$) take the average R/R interval in seconds as input, not in milliseconds. It is more accurate to not use Bazett and easier to avoid roots, so Sagie is a good choice here.

Compared to the algorithm in [Auer 2020], the new algorithm uses more advanced wave detection. There is an improved function to scan for 'half-peaks', which means finding the next onset or offset of a positive or negative peak within a selectable time period and in a selectable direction. For the P-waves, a new 'twin peak' detection tries to extend waves where onset and offset have big amplitude differences. It assumes that the end with the higher amplitude is only a local minimum and scans for an additional wave beyond that point. If the combination of both waves has better symmetry, it is selected instead of the original range. An example can be seen in figure 4.2 (right), taken from

the ECG-ID corpus, used to demonstrate the use of ECG as biometrical feature [Lugovaya 2014]. Other new functions measure wave height relative to a straight line between onset and offset and wave area as sum of heights over all included samples.

The half-peak search works by looking at a point 20 samples (at most 1/3 of the scan period) from the initial extreme point to measure the initial slope. Then, it proceeds in the selected time direction until a counter-slope of more than half of the current noise estimate is encountered. This makes it able to skip over local extrema caused by noise. If the slope drops below 1/8 of the initial slope or 5 times the noise estimate, the function also assumes having reached the onset or offset next to the initial peak.

To get a full set of ECG metrics, a pulse shape analysis function is called with the averaged ECG and corresponding slope data as input. The processing steps are:

1. If R is negative, expect curve to be S-anchored. Scan for maximum before S to find R. If R is positive, scan 120 ms for next minimum to find S. If none is found, instead walk time axis starting 24 ms after R, until slope drops below a quarter of the value at the 24 ms point.
2. Go backwards from 10 ms before R to find Q, with a limit of 50 ms before R, until the end of the down-slope or a counter-slope is encountered.
3. Find P between 20 ms before Q and half of the R/R interval before R. If the maximum amplitude in that area is at the start or end of the search interval, retry after excluding zones rising towards the start or end. Find onset and offset of P, if any P could be found.
4. Improve the Q location estimate by scanning between P and R, then search for Q onset.
5. Find T as the maximum after S offset (found by scanning from S) and scan for onset and offset of T. The Q to T offset time is later used to compute QT and QTc.
6. Search the period before P onset and between P offset and Q onset for additional waves taller than a configured fraction of the P height. Also search between S offset and T onset and after T offset, now with a search threshold which is a fraction of the weighted average of P and T heights. The maximum extra wave height is stored as ‘P2’ amplitude.
7. The mean amplitude between S offset and T onset is used as ST segment amplitude.

All key points in the ECG are marked with coloured lines on the analysis screen and all metrics are made available as a labelled list of integers, stored in units of milliseconds or microvolts.

To classify the shape, the P-wave height is compared to a configurable fraction of the R/S peak-peak amplitude or the absolute R amplitude relative to baseline, whichever is more. The score linearly depends on the distance between actual height and the reference height computed as described from the QRS height. Taller P-waves initially classify as ‘good’ or ‘okay’ (NSR) depending on height, lower ones as ‘bad’ (A-Fib). Note that the current version of the P-wave height score depends mostly on height. In [Auer 2020], more constraints on overall ECG morphology were used, lowering the score on violations. This is now implemented by adding morphology to the feature vector and processing that separately.

If the points of maximum slope are too asymmetrical with respect to the P-wave timing, a good judgement is downgraded to ‘okay’. The same happens when the PR interval is too short (below 120 or 100 ms depending on whether R/R is above or below 320 ms) or too long (above 205 ms).

If the P-wave onset is almost at the start of the search window, the P-wave is classified as ‘bad’ – this is the case for ECG with no stable P-waves, having just a shallow probability distribution hill in the average instead of a normal P-wave.

If extra waves are found (ignoring shallow ones, as described), the ECG also is classified as ‘bad’. A P-wave duration below 45 ms or below 1/5 of the search window is logged and marked, but does

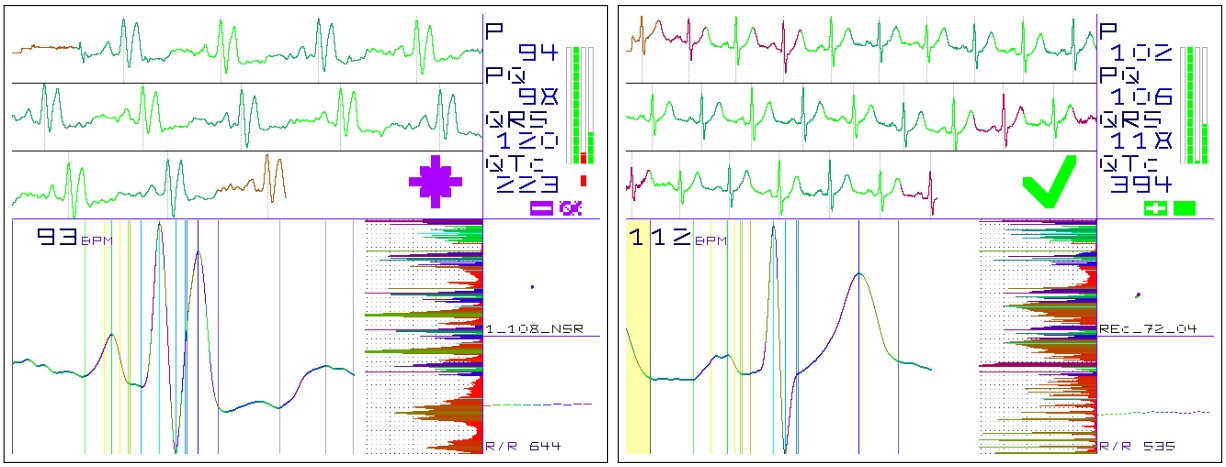


Figure 4.2: Left: Heterogeneous Dataset of Arrhythmia corpus example 1/108 with mixed pathologies, which the algorithm classifies as neither NSR nor A-Fib. Right: ECG-ID corpus item 72/04, NSR with two-peaked P-wave

not influence the classification. This is left to later analysis of the features. The same is the case for P-wave durations outside the range of 38 to 120 ms or 148 ms, although the latter also downgrades 'good' to 'okay'. If the peakedness (height to area ratio) is high, the judgement also is downgraded.

Other empirical rules used in the [Auer 2020] algorithm no longer change the judgement, e.g. absolute heights of P, Q and T, the interval from R to T, the intervals from P peak to R and from R to T peak and the value of QTc. Those have all been removed in favour of letting other modules of the algorithm combine all features for classification.

To create the averaged ECG required as input for the shape analysis, a noise estimate is computed first. For each second, the 0.02 s part with the lowest amplitude range is found. The average of all amplitude ranges selected in this way is the noise estimate, which is later adjusted for the expected signal to noise ratio improvement in the ECG averaging step.

Using the R timestamps detected earlier, R/R intervals below 390 ms and below 70% of the average, as well as R/R intervals shorter than 250 ms, are merged with the next, assuming that a T-wave has accidentally triggered the R detector in such cases. Around each remaining R, excerpts of 250 ms (PQRWINDOW samples) before to 600 ms absolute or 500 ms QTc Sagie after, whichever is later, are selected.

For each of those snippets, the RMS is calculated, as well as the mean of all RMS. Snippets with more than 150% mean RMS are considered to have too much noise or artefacts. For the remaining snippets, the average RMS of pairwise differences to all other snippets is computed. Snippets with this value below 125% of the minimum of the mean RMS, the average RMS of differences and 150% of the lowest RMS are used for averaging. Those are most similar to each other. Assuming that noise is not phase-locked to the ECG signal, it is assumed that averaging N snippets created a prototypical ECG beat shape with $1/\sqrt{N}$ of the original noise. If this noise is above a configurable threshold, the ECG recording will not be classified, being of too low quality.

Next, the averaged ECG shape is processed by measuring and, optionally, removing trends from the time window between start and 50 ms before R, but only if trend was below 0.5 mV/ s rising, or a falling trend. The latter may be seen at high pulse rates, from overlapping with preceding T-waves. Then, the baseline estimate is re-centred, considering all samples where slope is below a quarter of any slope after 60 ms after R, excluding any samples within 60 ms of R.

If R/R is not between 200 and 1750 ms, or if fewer than 1 beat per 3 seconds of recorded ECG or below 4 beats in total contributed to the average, the algorithm does not attempt to classify the

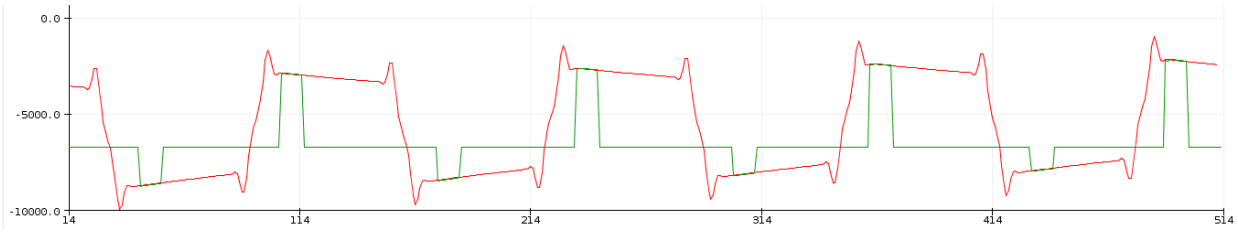


Figure 4.3: Self-calibration of ecghelper2, using the MAX30003 built-in source

ECG. Instead, it returns 'noisy'. Otherwise, smoothing is applied, excluding the QRS complex, and a filter is used to create a separate slope (derivative) curve.

Spectra are computed for the last 2^n samples (at most 4 096 in the embedded system, the Linux corpus analysis tool allows 16 384) and for the set of beat snippets, as listed in figure 4.2. The beat spectra are computed for 512 sample windows centred around the R peak, using a Blackman window. Having the spectra aligned with the QRS complexes, they also have a meaningful phase interpretation.

By comparing spectra with and without phase information, an estimate of the phase stability for each frequency is computed, creating yet another spectrum. For each spectrum, a number of higher level features such as the dispersion or number of peaks are calculated as well, also described in figure 4.2. As already mentioned, HRV style statistics about the R/R intervals also serve as the basis of additional features.

Note that features 102, 142 and 229 (related to 2 Hz spectral properties and R/R interval) have been excluded from LDA and threshold rule generation described later. This is to avoid classification bias at higher heart rates around 120 bpm.

4.3 Global classification

Using all spectral features, R/R features and metrics extracted from the ECG shape, the actual classification module is invoked. Because P-wave duration plus P-wave onset equals P-wave offset in a linear way, that duration feature is replaced by $15 \times 255/x$, with x being the original value saturated between 15 and 200, to avoid the linear correspondence trapping later (e.g. LDA) processing. Also, some high level features like phase instability are used in classification.

If either the P-wave scoring or the feature processing (threshold rules and LDA) classify the recording as 'noisy', the overall classification will be noisy. If feature processing is neutral, P-wave classification takes precedence. Both P-wave score and feature processing have to agree on a classification as NSR to make the overall result NSR, so if either of both votes for A-Fib, it will be A-Fib.

4.4 Calibration and recording

To be able to express the heights and amplitude features of the ECG in absolute voltages, the embedded software implements self-calibration. The MAX30003 has a built-in calibration voltage generator which can output positive, negative or bipolar rectangular waves of well-defined amplitude, frequency and duty cycle. This signal is measured for a few seconds after booting the software. The system boots when a lead-on event is detected, in other words, when the user touches both main electrodes with both hands.

As shown in figure 4.3, a section of the settled signal is thresholded to find edges. After each edge, samples in a small time window are averaged. The difference between those averages are the estimated peak-peak amplitudes of the calibration signal. As the calibration signal is measured

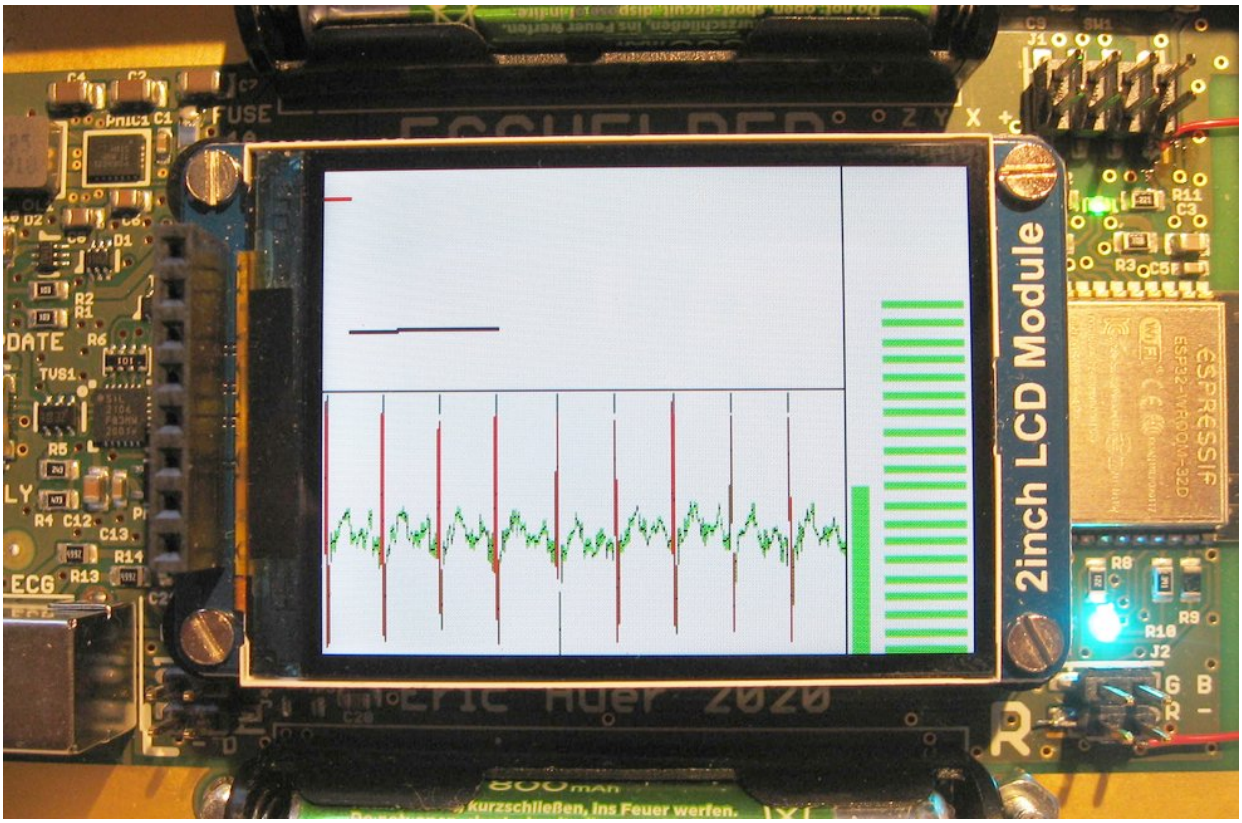


Figure 4.4: While recording the ECG, a simple scope view and progress indicators are shown

after applying all filters, both the analogue high-pass filter and digital filters of the MAX30003 and digital filters of the `ecghelper2` software itself, this peak-peak amplitude is very suitable for comparison to the ECG data recorded after calibration.

Note that while the chosen time windows avoid the ringing artefacts of the digital `ecghelper2` filters, it fails to capture the maximum calibration signal amplitudes. The signal decays after each edge, due to the high-pass filtering of the MAX30003. No compensation for this is present in the software at the moment, but sampling amplitudes shortly before the next edge would be a way to get an estimate of the decay rate and extrapolate more exact amplitudes.

In corpus processing, the `offlineecg2` software expects the conversion factor between raw data and actual values through an LSB per millivolt command line argument. The PTB-XL corpus is specified as using 1 LSB per microvolt: 1000 LSB per millivolt. The corpora used in the original `ecghelper` project often have fewer, but constant LSB per millivolt. Some even use different factors for each recording, so the factors have to be extracted from the provided metadata. For uncalibrated recordings, a rough manual estimate is used in lieu of an actual known conversion factor.

After calibration, the embedded device starts recording the ECG of the user. While doing so, a simple scope view of the ECG, shown in figure 4.4 is shown. After a short warm-up period, the software can detect QRS complexes in the recording. For each detected complex (R/R event), the device displays the R/R interval to the previous event in a graph in the upper left of the display and marks the event in the scope graph at the lower left.

At the right, two bar graphs indicate how much of the recording buffer has already been filled and how many R/R events have been collected. When either the (35 s) recording buffer or the (26 QRS, 25 R/R intervals) R/R event buffer are full, the recording ends. If lead contact is lost earlier than that, the recording is either discarded (if too little ECG has been gathered yet) and the device returns to standby mode, or the recording is processed as if it had been completed normally.

Unlike `ecghelper2`, the `offlineecg2` software will revisit the beginning of the recording after using it to warm-up (self-calibrate detection thresholds) the QRS detection mechanism, which still uses essentially the same simplified Pan-Tompkins algorithm as described in [Auer 2020]. See [Pan 1985] for the complete original version of this classic algorithm. While the original algorithm uses a very simple band-pass filter to work with extremely low CPU loads, the old `ecghelper` (not `ecghelper2`) version of the software uses a 63-tap FIR band-pass filter instead. The `ecghelper2` software replaces this by a shorter 47-tap FIR correlation filter generated by forming the average of the actual detected QRS complexes and weighting it by the inverse of the standard deviation for each sample. This gives the filter a bias towards detecting QRS complexes in NSR (or A-Fib) while sacrifying some sensitivity for arrhythmia related beats such as premature contractions. Both filters are shown in the appendix, as figures B.7 and B.8, illustrating the differences. For digital filters and signal processing topics, [Smith 1998] has been a useful source.

Being able to use the same data for warm-up and analysis makes the `offlineecg2` software more successful in processing shorter recordings such as the 10 second excerpts in the PTB-XL corpus. The second reason why the `ecghelper2` software attempts to gather longer recordings is that the `ecghelper` prototype hardware is expected to produce lower quality ECG, since it uses simple sheet metal electrodes instead of disposable sticky electrodes and the users are unlikely to apply electrode gel to improve contact. So by using longer recordings, a larger improvement in signal to noise ratio can be achieved in the averaging step.

4.5 Custom digital filtering

In addition to the filters built into the MAX30003 and the filtering already applied by ECG corpus creators, the `ecghelper` algorithm also applies fast digital filters (see section 2.3) to the signal. In the previous version of the software, a 10-tap moving average convoluted with a short high-pass filter was used. This compensates the low-pass effect of the moving average a bit, while preserving the original property of the latter to suppress all multiples of 50 Hz in a signal recorded at 500 sps. For those frequencies, 10 samples correspond to an integer multiple of the period length. This offers an efficient way to remove mains power harmonics in countries with 50 Hz mains frequency.

The `ecghelper2` software improves this step by convoluting the moving average filter with two different high-pass filters and then truncating the resulting filter to keep latencies low. This produces a more flat frequency response with less ringing and less damping of frequencies below 40 Hz, while still preserving the notch property for all multiples of 50 Hz. Some good example frequency responses for different combinations of filters are shown in figure 4.5. By doing a search through various parameter settings and measuring key properties of the resulting frequency responses, it was possible to generate a short list of candidates. The filter actually used in `ecghelper2` has been manually selected from this list, based on the trade-off between damping and ringing. Details of this filter and of the filter used in the original `ecghelper` software are shown in the appendix in figures B.3 and B.4.

To optimise handling of corpus recordings with 60 Hz mains frequency artefacts, an IIR biquad Pei Tseng 60 Hz notch filter is available in the software. The width of the notch has been chosen to give sufficient damping even when frequencies vary a bit, while not causing too much damping or phase distortion for ECG signals below 40 Hz. Likewise, a 187.25 Hz Pei Tseng notch filter is available. For unknown reasons, some of the recordings in the corpora used in [Auer 2020] show artefacts at that frequency, according to spectral analysis. No filters for harmonics such as 120 Hz are used. The Pei Tseng filter properties can be found in figures B.6 and B.5 respectively, in the appendix.

The `offlineecg2` software checks how much effect applying the filter would have for the current recording. If it is above a configurable threshold, the Pei Tseng notch is activated. This prevents unnecessary distortion of recordings which do not contain 60 Hz or 187.25 Hz artefacts anyway. At most one of the two Pei Tseng filters will be activated, depending on which has more effect on the

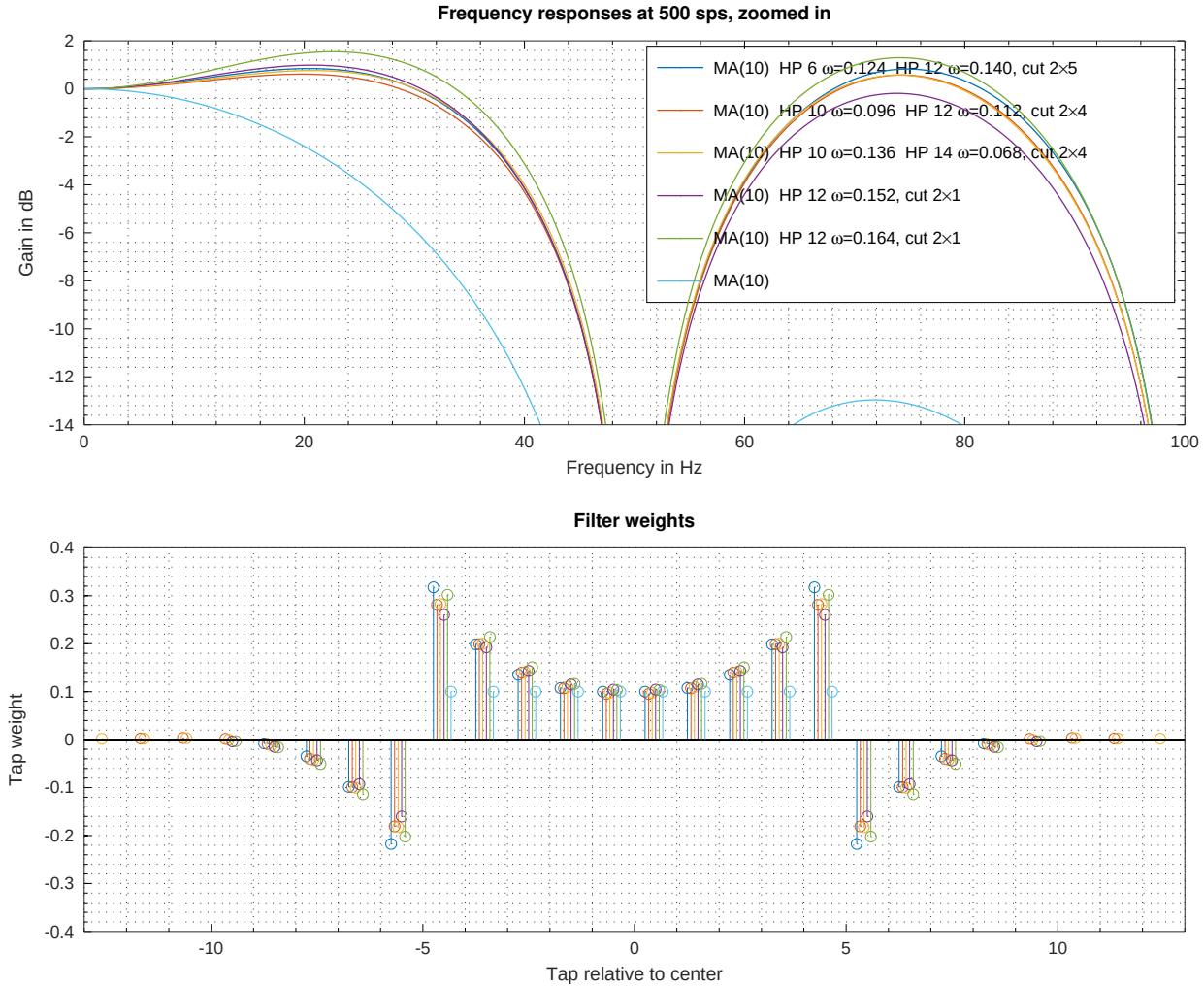


Figure 4.5: Comparison of old (purple = compensated), new (orange = double compensation) and naive (blue = plain moving average) 50 Hz comb filters, plus 3 runner-up choices

signal. The embedded `ecghelper2` software does **not** use either of the two filters. To make a version of the device for markets with 60 Hz mains frequency, a more optimised filter should be considered first, possibly combined with different sampling rates for the raw ECG as well.

The last two filters used by either version of the software are for smoothing and analysis of the averaged ECG shape. First, a short smoothing filter is applied in several passes, each time excluding a time window for the QRS complex. The properties of the filter are shown in the appendix in figure B.9. As said, the QRS complex is spared from this filter, so the strong low-pass effects of the filter still only cause limited distortion, as the P-wave and T-wave reside mostly in lower frequency spectral space. Finally, a short binomial derivative filter, shown in the appendix in figure B.10 is used to get a smoothed estimate of the slope of the averaged ECG at specific points in time. The resulting smoothed average ECG and smoothed slope of the smoothed average ECG are both used for the extraction of metrics about the ECG shape, as listed above.

4.6 PTB-XL corpus excerpts

The machine learning part of this thesis makes heavy use of the PTB-XL corpus, a large, publicly available ECG corpus based on data acquired from Schiller AG by the Physikalisch-Technische Bundesanstalt, the National Metrology Institute of Germany. It contains almost 22 000 ECG recordings of 10 seconds length each. The majority of the recordings are of high quality and all come

with diagnosis metadata, which again are human-reviewed in most of the cases [Wagner 2020a, Wagner 2020b].

While the corpus has full 12-lead data, upsampled from 400 to 500 sps, only the Einthoven Lead I channel is used in this thesis, in two sets of data, used for both training and testing. The A-Fib set (atrial fibrillation, some atrial flutter) and the NSR set, with Normal Sinus Rhythm ECG.

To gather entries for the A-Fib corpus, PTB-XL metadata have been searched, via case-insensitive matching, to **select all items containing 1. 'atrial' followed by 'fib', 2. 'atrial' followed by 'flu' or 3. 'vorhofl', then 4. remove all items containing 'sinus'.**

This matches 1 511 recordings: 1 473 from the AFIB and 46 from the AFLT (atrial flutter) categories of PTB-XL – 16 ECG recordings are in both categories at the same time. The whole categories are 1 514 and 73 items respectively, with 17 overlap, so a total of 1 570 AFIB and/or AFLT examples exists in PTB-XL. Nine of those mention 'sinus': 2 AFIB and 7 AFLT items.

This means that 63 items (including 5 which mention 'sinus') in AFIB or AFLT are not captured by the above query. Those can be used as additional test set of recordings which are not found in the training data. 36 of those items use the Swedish metadata text 'fÖrmaksflimmer/-fladder' (sic!) and others simply contain spelling errors or unusual wordings in their metadata.

Of this additional test set, 60 get classified as A-Fib, 1 as NSR (04874: 142 bpm, metadata text mentions only supraventricular tachycardia and RBBB (right bundle branch block), but the category is AFLT) and 2 are classified as neither NSR nor A-Fib (06205: 141 bpm, metadata mentions AV tachycardia, peripheral voltage, possible subacute inferior infarction, category AFLT / 10564: 76 bpm (?), metadata text mentions supraventricular tachycardia and LBBB (left bundle branch block), but the category is AFLT).

Next, a selection of NSR examples has been made, as follows: **1. Select all items from the NORM category (excluding all AFIB) 2. Remove all items containing 'arryth' 3. Keep only items containing any of: ';sinusrhythmus normales ekg;', ';sinusrhythmus linkstyp sonst normales ekg;', ';sinustachykardie sonst normales ekg;', ';sinusbradykardie sonst normales ekg;', ';sinusrhythmus lagetyp normal ekg normal;', ';sinusrhythmus lagetyp normal normales ekg;', ';sinus rhythm. normal ecg.;', ';sinus rhythm. no definite pathology.;', ';sinus bradycardia. otherwise normal ecg.', ';sinusrhythmus lagetyp normal normales ekg', ';sinusrhythmus linkstyp sonst normales ekg', ';sinusbradykardie lagetyp normal sonst normales ekg' ';sinus rhythm normal ecg;', ';sinus rhythm. normal ecg;', ';sinus rhythm. normal ecg.', ';sinusrytm normalt ekg;', ';sinusrhythmus lagetyp normal sonst normales ekg;'.**

This includes NSR at variable pulse rates, including otherwise physiological bradycardia and tachycardia recordings. To extend the selection, all records matching the following query were added, unless already part of the previous set: **1. Select all items matching 'sinusbrady', 'sinus brady' or 'tachy' 2. Keep only those containing at least one of 'sonst normal', 'normales ekg', 'otherwise normal' or 'eljest normalt ekg'**

This matches 6 125 recordings (without tachycardia and bradycardia, it would be 5 832). Checksumming the data revealed that PTB-XL contains a small number of duplicate recordings. Those have not been removed from the corpus, but only the following 18 NSR record numbers are affected:

00456 = 00457, 00459 = 00460, 02506 = 02507, 18150 = 18151, 00145 = 00463, 00462 = 13802, 02511 = 07784, 09825 = 11813 and 11809 = 15742.

Few additional duplicates exist in areas of the PTB-XL corpus not used in this thesis: 03800 = 03801 = 03802, 11814 = 11815 = 11816, 00137 = 00138, 00143 = 00144, 03795 = 03796, 03798 = 03799, 07777 = 07778, 07779 = 07780, 07782 = 07783, 09821 = 09822, 09888 = 09889, 13791 = 13792, 13793 = 13794, 13797 = 13798, 13799 = 13800, 00139 = 07781, 00140 = 01370, 00141 =

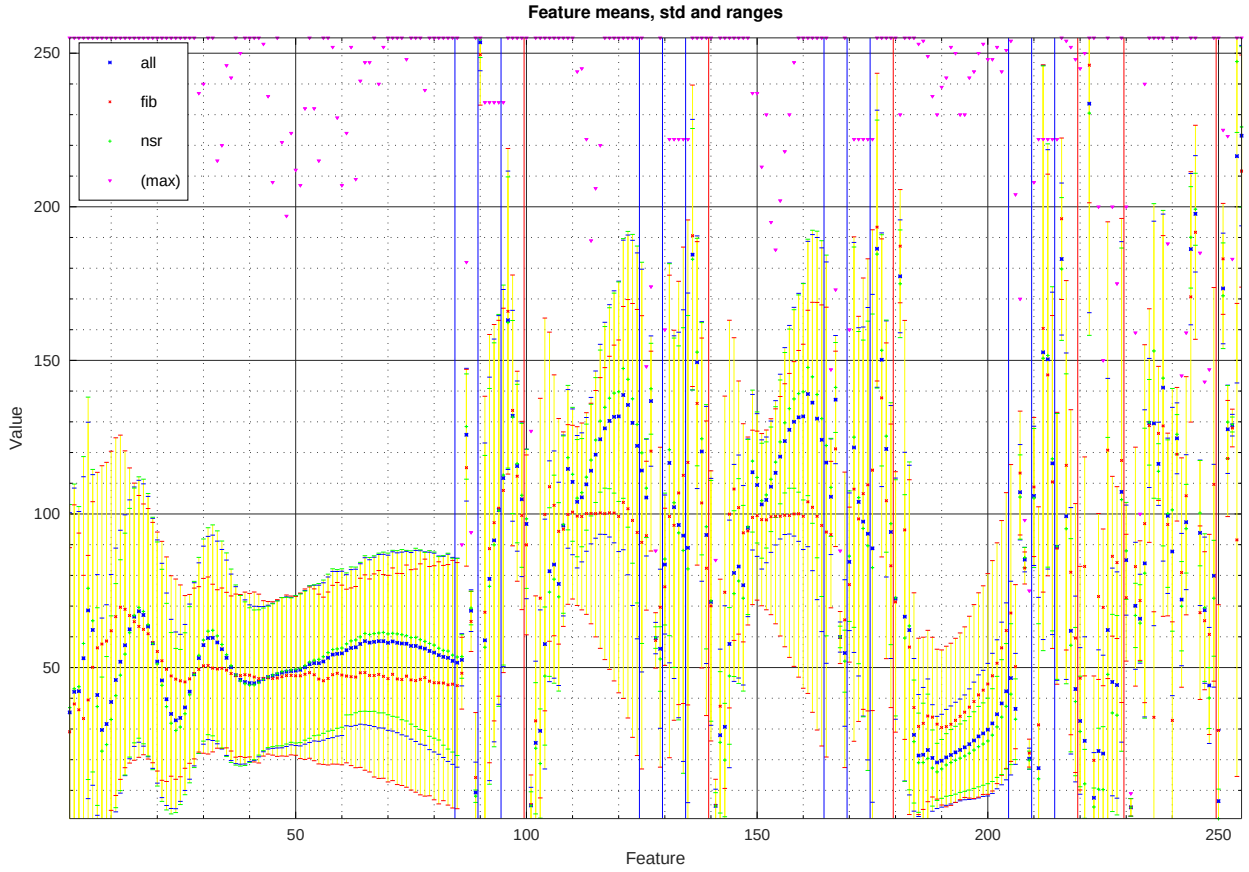


Figure 4.6: Some statistics about the used feature vectors

02509, 00142 = 13803, 00146 = 11817, 00458 = 03797, 00461 = 09823, 01371 = 13796, 05817 = 09824 and 11810 = 15741.

By storing the last digit of the record number as a feature, the machine learning experiments can select constant, pseudo-random subsets of the data, for example leave-one-out cross-validation.

However, training results have not differed much with that approach. On one hand, this is a good sign – training results do not depend on the exact choice of training examples and there is no serious overfitting. On the other hand, it neglects a key problem of PTB-XL based training: PTB-XL has been recorded with professional high-end medical devices and not with an affordable home-use device like the `ecghelper2` prototype, so there are systematic differences in quality between training data and ECG as seen by the prototype.

To still get a realistic impression on how well the trained classification can perform on the actual `ecghelper2` hardware, without having to perform a high-effort test series with dozens of subjects, both healthy ones and subjects who are affected by atrial fibrillation of flutter, requiring explicit permissions etc., the corpora used in [Auer 2020] have been used as additional evaluation corpora.

Those smaller corpora include lower quality ones and the sensitivity of the `ecghelper2` (and of course `offlineecg2`, always implied when mentioning either!) algorithm has been adjusted manually, also based on informal leave-one-out experiments, where trained rule thresholds were found to rarely differ by more than 5% of their value, usually by far less than that, depending on which tenth of the training corpus had been left out.

For example, there is a configurable margin in feature range rules. Rule generation based on PTB-XL data could contain ‘if feature 103 is above 183, vote for A-Fib’, while a margin of 5 will turn that into ‘if feature 103 is at least above 183 by at least 5, vote for A-Fib’. However, none of the automated training steps has access to any other data than PTB-XL and the margin is applied

globally to all rules. This means the manual adjustments are not too specific to the additional corpora, but rather tune the algorithm to work better with lower quality recordings in general.

Also, the algorithm computes a noise level estimate from the input ECG and automatically adapts some margins based on that. For example, if more noise is found in a recording, the algorithm will more readily skip over small irregularities in the averaged ECG shape when scanning for key points like peaks, onsets and offsets of individual waves.

4.7 Algorithm feature landscape

The algorithm uses an up to 256-dimensional space to describe each ECG, quantified in byte-valued feature vector coordinates. This is meant to keep the features coarse enough against overfitting to exact training data details and detailed enough to represent key ECG properties.

Features	**Description** (all features scaled to unsigned byte ranges)
0 to 84	global spectrum of last 2^n samples: 1/4 Hz per bin. Starting at feature 60: 1/2 Hz per bin. The spectrum is normalised in the range above current pulse frequency, with some margin. Then, higher frequencies are boosted by a de-emphasis factor using a function growing with frequency, parameters adjusted manually to flatten the spectrum in figure 4.6
85 to 99	global spectrum metrics: Mean value, central (centre of gravity) frequency, dispersion, skew, count of peaks, frequencies of the 5 highest peaks in the spectrum (highest first), values of those 5 peaks
100 to 124	phased beat spectrum, magnitude of average of complex spectral values: 1 Hz per bin. Starting at feature 120: 2 Hz per bin. This spectrum also applies normalisation and de-emphasis, with the same de-emphasis parameters but different normalisation constant
125 to 139	phased beat spectrum metrics: Same method as for global spectrum
140 to 164	beat spectrum, average of magnitudes of spectral values: 1 Hz per bin. Starting at feature 160: 2 Hz per bin. Again, normalisation and de-emphasis are applied, with the same de-emphasis parameters
165 to 179	beat spectrum metrics: Same method as for global spectrum
180 to 204	phase instability spectrum: (1− arcus sinus of ratio of phased beat spectrum values to beat spectrum values), 1 Hz per bin. Starting at feature 200: 2 Hz per bin. As this spectrum is based on the ratio between two already normalised spectra, it is not normalised again
205 to 219	phase instability spectrum metrics: Same method as for global spectrum
220 to 229	R/R Heart Rate Variability metrics: Standard deviation of R/R intervals and of their differences, magnitude of mean difference between the intervals. Fractions of cases where next R/R is at least 50 ms above, below or different from current. The same 3 metrics for 20 ms threshold. Mean R/R interval when saturating outlier values
230 to 249	Averaged ECG shape metrics: See table 4.1, or `ecgLabels` array in `ecgshape.cpp` for numbering: Pseudo-feature 230 + 0 is the last digit found from the input file name, for leave-one-out use.
250 to 254	High level features: Phase instability, baseline offset, averaged ECG pre-QRS slope, P-wave height & shape score, fraction of beats good (consistent) enough to be used to compute the averaged ECG shape
255	Classification result (**not** used for training)

Table 4.2: Overview of the 256 byte ecghelper2 algorithm feature vector

Feature	Feature description, rule and rule performance
220	Standard deviation of R/R intervals (unit 0.25% of mean R/R interval) is below 8 → NSR, match: 2 251 actual NSR, 20 A-Fib recordings
235	Height of P-wave relative to a line from onset to offset (units of 0.5 μV) is above 167 → NSR, match: 2 243 actual NSR, 17 A-Fib recordings
239	Height of T-wave relative to a line from onset to offset (units of 2 μV) is above 130 → NSR, match: 1 312 actual NSR, 15 A-Fib recordings
235	P-wave height, see above, is below 36 → A-Fib, match: 960 real A-Fib (almost two thirds of all A-Fib instances), 14 NSR recordings
231	Period between P- and R-peak (unit sampling periods of 2 ms) is below 38 → A-Fib, match: only 306 A-Fib, only 5 NSR
183	Phase instability at 3 Hz (from 0, stable, to 255, random) is above 183 → A-Fib, match: 181 A-Fib, no NSR
156	Beat spectrum at 16 Hz (custom scaling, after normalisation and de-emphasis) is below 42 → A-Fib, match: 226 A-Fib, 1 NSR
119	Phased beat spectrum at 19 Hz (custom scaling, after normalisation and de-emphasis) is below 20 → A-Fib, match: 180 A-Fib, no NSR
126	Phased beat spectrum central frequency (custom scaling) is below 77 → A-Fib, match: 178 A-Fib, no NSR
115	Phased beat spectrum at 15 Hz (custom scaling, after normalisation and de-emphasis) is below 46 → A-Fib, match: 219 A-Fib, 1 NSR
166	Beat spectrum central frequency (custom scaling) is below 78 → A-Fib, match: 175 A-Fib, no NSR

Table 4.3: All rules for NSR detection and eight best rules for A-Fib detection, based on automated analysis of feature data from the PTB-XL NSR and A-Fib subsets described in section 4.6

At the same time, there are enough features to give the algorithm a lot of flexibility to find signs of A-Fib or NSR, without relying too much on human expectations, as was the case in [Auer 2020] which included multiple knowledge-based rules. Table 4.2 gives a full list of the used features.

Figure 4.6 gives an overview of the mean values of all features, as well as the ±1 standard deviations range around them and the maximum encountered values. Minimum encountered value for almost all features is zero, not shown in the plot.

4.8 Rule-based feature processing

An effective ad-hoc way to capture some more obvious A-Fib cases is based on generating feature threshold rules. For this, the `offlineecg2` software is first used to extract feature vectors for all A-Fib corpus and, separately, for all NSR corpus data from PTB-XL. Then, an Octave[56] script is used to compute distributions for each feature and searching potential rules.

An effective rule, for the purposes of this thesis, is one which matches many recordings from one of the two categories (A-Fib or NSR) while matching as few as possible of the other. To avoid too high ratings for rules which make zero mistakes, the ratio of good to bad decisions is biased by adding a constant (4) to the denominator. Also, rules which make more than 20 errors at all, or rules involving thresholds too close to 0 or 255, are rejected.

On the other hand, rules with less favourable ratio are considered nevertheless, if the difference between means is at least 1.4 standard deviations – after separately normalising the distributions

[56] A free and open source alternative to Matlab, `https://www.gnu.org/software/octave/` – see also the portal `https://octave.sourceforge.io/packages.php` for signal processing and other packages

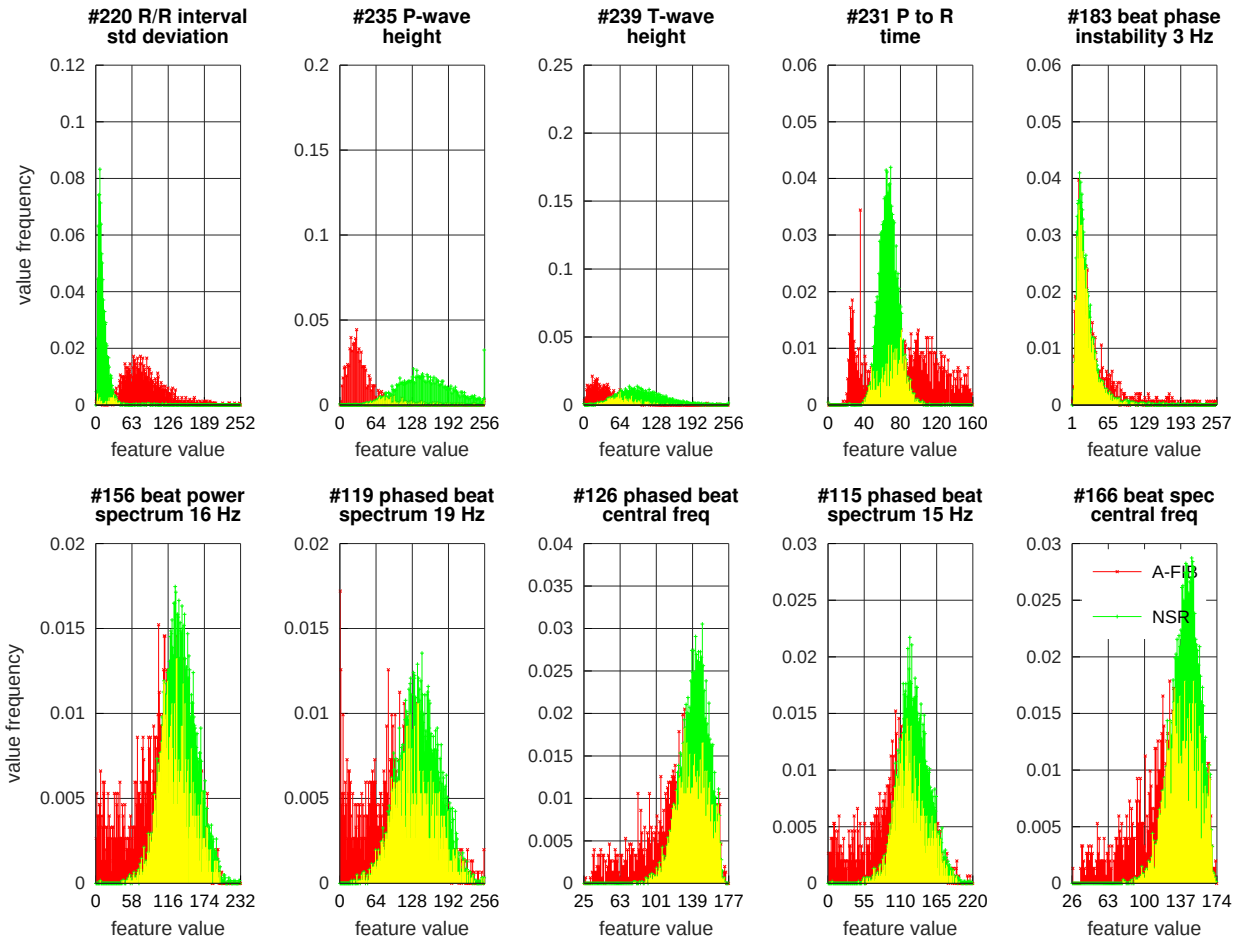


Figure 4.7: Best features when using threshold rules for classification

for both classes to standard normal distribution, by removing means and dividing by individual standard deviation. If either the mean differences are large enough or the biased ratio is at least 40, a rule is generated. Rules can be of any of 4 types: 1. vote for NSR when feature is above a constant, 2. vote for NSR when feature is below a constant, or 3. & 4. similar, but for A-Fib.

Based on those settings, only 3 NSR rules are generated – but each of them covers up to a third of all NSR recordings. They are listed in table 4.3. Far more rules are generated to trigger on A-Fib recordings, which means that those can be more readily identified by having feature values outside ranges normal for NSR recordings. The 8 most effective rules in this set are, sorted by effectiveness, best first, also listed in table 4.3.

Figure 4.7 presents the value distributions of the features mentioned in the 3 NSR rules and the most effective A-Fib rules listed above.

In addition, feature value distributions were checked for absolute range limits found in the NSR and A-Fib corpora. Features which are outside that range trigger votes for the classification that no decision between NSR or A-Fib is possible ('noisy') one example is recording 1/108 from the Heterogeneous Dataset of Arrhythmia [Medhi 2019], shown in figure 4.2 (left).

Excerpts from metadata for this recording: 'Resection and grafting of abdominal aortic aneurysm / Coronary disease / Post-aortic valve replacement / Hypothermic @ 95F / Normal sinus rhythm with right bundle branch block @ 65 bpm / ?Sinus block / Atrial ectopy / Multifocal ventricular ectopy, occasional bigeminy / Left axis deviation / Left anterior hemiblock / Nonspecific ST segment and T wave changes'.

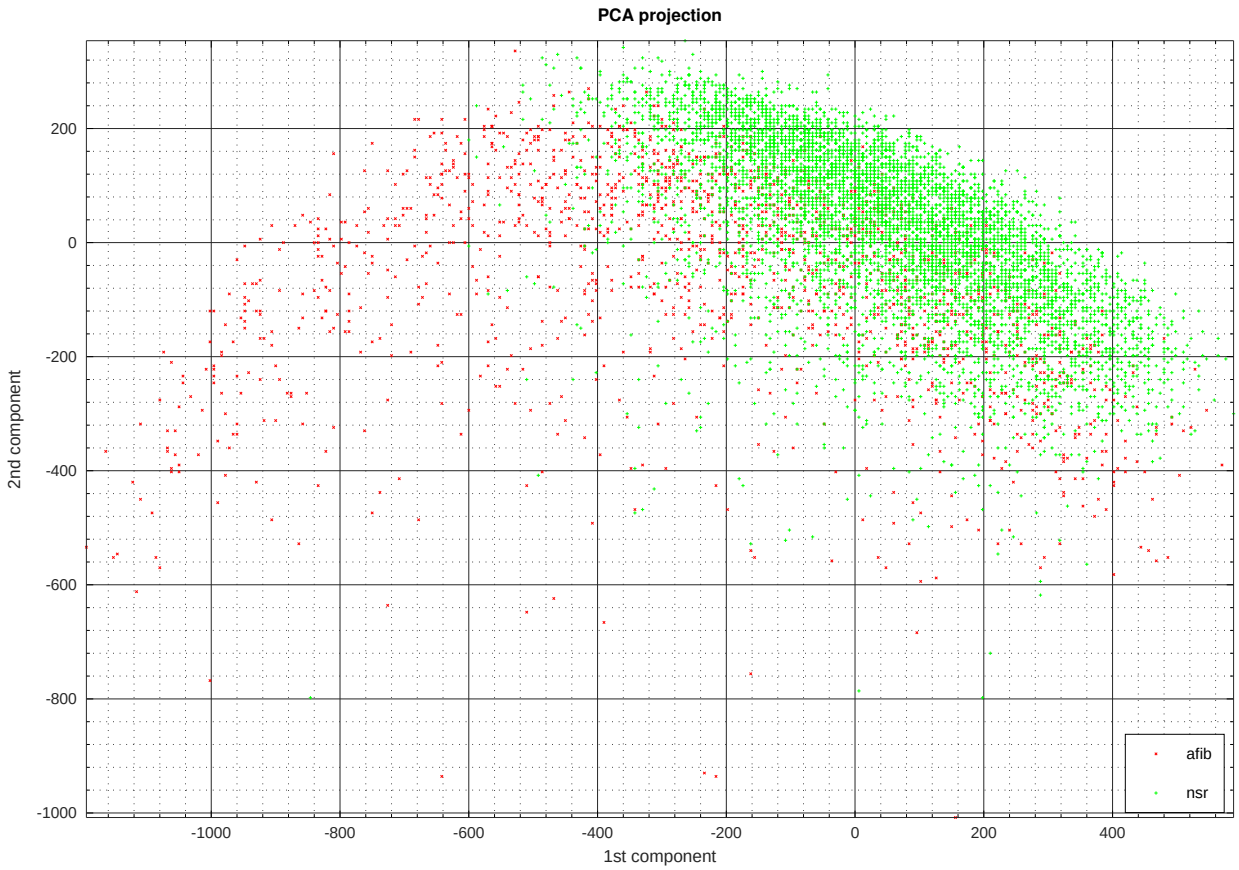


Figure 4.8: Projection of the test sets using the most relevant PCA dimensions

4.9 Principal component analysis

A generic method to reduce the dimensionality of data such as our feature vectors is Principal Component Analysis (PCA) – see section 2.4. PCA can be used to project unlabelled data onto a new feature space in a way which makes the first few coordinates capture most of the variance in the dataset.

However, PCA turned out to be not very effective in capturing much of the data variance in few dimensions, as shown in table 4.4. Figure 4.8 shows the projection of the features of all recordings (colours indicate NSR, green, or A-Fib, red) onto the two most significant PCA dimensions. It would have been necessary to retain most dimensions even after PCA to get a reliable classification.

The original feature vector already is small enough for further processing and has the advantage of having a clear mapping to properties of the ECG. So the algorithm chooses to not apply PCA feature space reduction, but use the full feature vector for further processing.

By not using PCA, other classification steps will still use all original, well-defined features. This makes it more feasible to interpret training results for those other steps in terms of actual ECG properties.

Dimensions	1	2	4	8	16	32	64	128	192
Captured variance in %	21.4	32.1	41.2	51.9	62.6	75.8	88.6	98.0	99.9

Table 4.4: Variance in the used PTB-XL corpora captured by the first N PCA dimensions

Feature set variation	Missed NSR	Missed A-Fib
Full feature set (more overfitting risk)	30	16
Omit raw global spectrum	50	40
Omit metrics for global spectrum	38	20
Omit global spectrum and metrics	55	50
Omit raw phased beat spectrum	38	23
Omit metrics for phased beat spectrum	37	20
Omit phased beat spectrum and metrics	58	29
Omit raw beat spectrum	27	30
Omit metrics for beat spectrum	31	25
Omit beat spectrum and metrics	42	28
Omit raw phase instability spectrum	43	20
Omit metrics for phase instability spectrum	32	20
Omit phased instability spectrum and metrics	35	31
Omit R/R HRV metrics	66	25
Omit ECG shape metrics	36	32
Omit high level features	38	18
Use **only** raw global spectrum	1 014	311 (!)
Use **only** global spectrum metrics	947	194 (!)
Use **only** global spectrum with metrics	634	176
Use **only** raw phased beat spectrum	561	325
Use **only** phased beat spectrum metrics	2 052	432
Use **only** raw beat spectrum	655	292
Use **only** beat spectrum metrics	1 345	554
Use **only** phase stability spectrum	1 481	333
Use **only** phase stability metrics	1 569	425
Use **only** R/R metrics	208	151
Use **only** ECG shape metrics	133	160
Use **only** high level features	401	124
Omit all spectra and metrics	95	91
Use **only** beat spectra and metrics	409	180
Use **only** R/R and ECG shape metrics	104	93

Table 4.5: Effects of different reductions in feature space on LDA. Note that a stability spectrum was used at that time (September) instead of an instability spectrum. More intense shades of green mark combinations with low error counts while using small feature sets

4.10 Linear discriminant analysis

Because the PTB-XL corpus already annotates all recordings with reliable metadata concerning the diagnosis, Linear Discriminant Analysis – see section 2.4. LDA can be a very effective method for generating a classifier for the given ECG data.

This algorithm transforms the original high dimensionality feature vectors into a new space which optimises linear separability of the data. It is so effective, that a simple threshold operation on the first dimension of the transformed coordinates only missed 16 A-Fib and 30 NSR out of the thousands of selected PTB-XL recordings in an early test.

The resulting transform also is trivial to implement for the software. The feature vector is multiplied with weights and the result is compared to a constant threshold. A proper LDA projection of the vector onto the first coordinate would have removed feature means first. Not doing that here simply

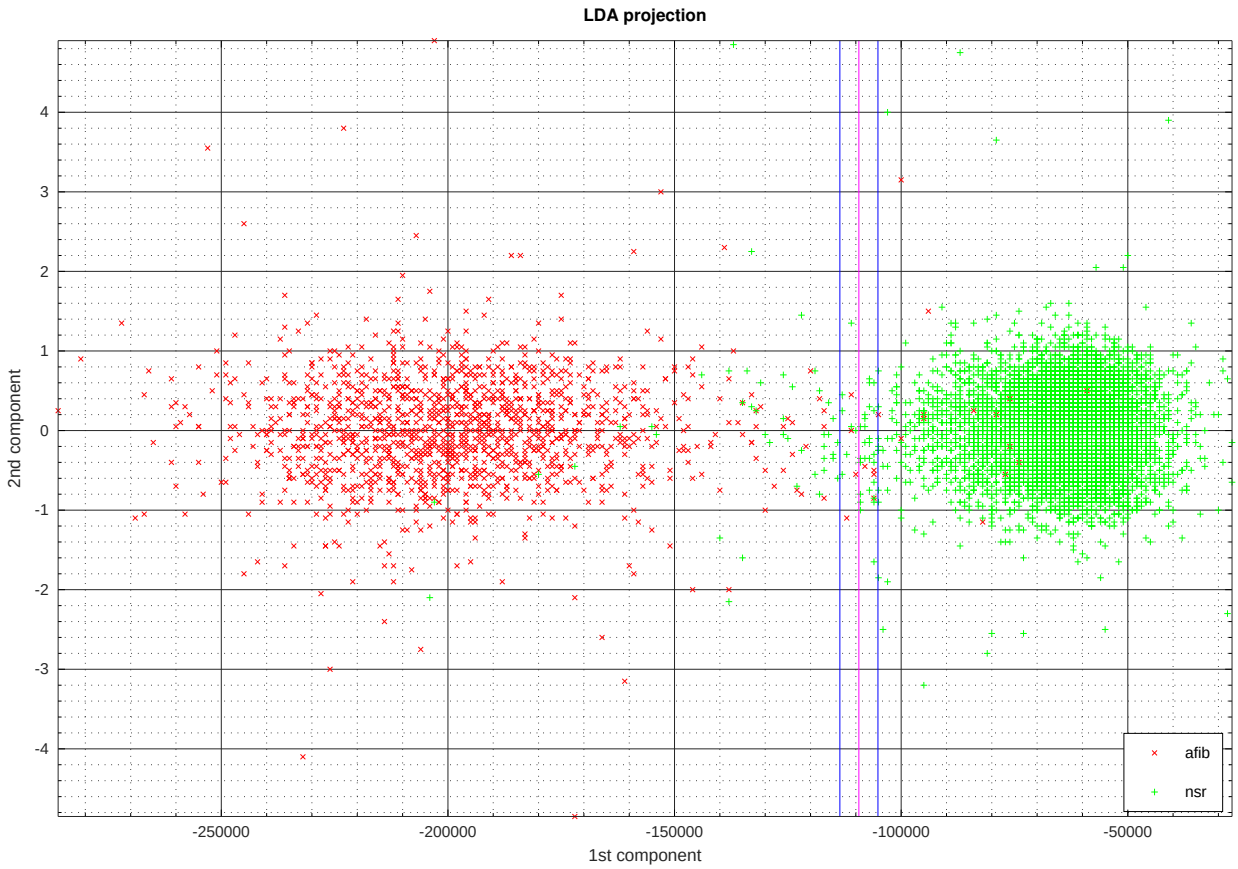


Figure 4.9: Projection of the test sets using the best trained LDA dimensions

introduces a constant offset. Likewise, weights have been scaled and quantified to integers, without losing the classification ability of the transform.

The ability of LDA to efficiently transform a classification problem for training data with known labels into a geometric feature space transform, however, also is a weakness for the `ecghelper2` algorithm. Applying the trained classifier to the **old** corpora known from [Auer 2020], 12 A-Fib cases were missed and falsely classified as NSR. As the old corpora only have less than 70 A-Fib explicitly known examples in the first place, this represents a rather low performance.

The problem is that having 256 features gives the LDA algorithm too much freedom for overfitting. It may capture properties of prototypical A-Fib or NSR using a part of the features, but then use the remaining features to add 'distortions' to the transform which pull a smaller number of cases into the requested classes, overriding the more useful prototype matching.

To investigate and reduce this effect, some LDA runs with reduced feature vectors have been computed. It turns out that LDA is still effective at the classification task, but of course less effective than with the full feature set. The results are shown in table 4.5.

Interestingly, using only the R/R metrics, only the ECG shape metrics or only all spectral data already gives relatively good results. Combining R/R and ECG shape metrics, 104 NSR and 93 A-Fib are missed: 428% of the amount of errors compared to the full feature set, but achievable without using any frequency transforms. On the other hand, using **only** spectral data, a similar amount of errors can be achieved (95 NSR, 91 A-Fib) without having to do any shape analysis and without having to compute the average ECG shape at all.

Using only R/R metrics performs worse than using only ECG shape metrics, but it has to be taken into account that R/R metrics are extremely easy to acquire. The MAX30003 actually has built-in R/R detection (a variant of the Pan-Tompkins algorithm, see [Auer 2020] and [Pan 1985])

in hardware. So an embedded system could gather R/R timing data without even fetching the raw ECG samples from the chip at all, requiring very little CPU power or memory. This can still perform better than using any one of the spectra alone, optionally with corresponding metrics. But instead of trimming the feature vector, `ecghelper2` uses another approach to reduce overfitting:

LDA projection parameters are modified before use in the software, zeroing all weights below 1/40 of the highest encountered weight. Quantification uses a factor of 4 096, which is twice the lowest power of two above which no additional improvements in classification occur. A factor of 64 is already reasonable, 512 is quite good. In addition, weights which have an importance of less than 0.0002 when measured in eigenvalue to standard deviation ratio have been zeroed as well. As with the feature rules described earlier, features 102, 142, 229 and 255 have been forced to zero weights, too.

Together, this results in non-zero weights being used for 171 out of the available 256 features, missing 52 NSR and 18 A-Fib recordings, finding 99.15% of all NSR and 98.81% of all A-Fib in the PTB-XL selections. To reduce overfitting even further, the software uses a manually selected threshold of ±4200 units around the decision boundary. Results which are too close to the boundary are classified as 'can not classify' ('noisy') and not as either A-Fib or NSR. This reduces the error counts to 44 NSR and 13 A-Fib being explictly misclassified and introduces 37 NSR and 9 A-Fib for which the LDA component of the algorithm claims to be unable to make a decision. Conversely, for 6 044 NSR and 1 489 A-Fib recordings, the LDA produces a high confidence classification result.

A plot of the projection and of the thresholds is shown in figure 4.9. Note that the second LDA projection is not used in the software. It just captures a good part of the remaining variability in the input data and makes the plot easier to understand intuitively.

Interestingly, the LDA weights sometimes suggest pairs of features being used together. Examples are QT and QTc weights, as well as phased beat spectrum 9 Hz and beat spectrum 9 Hz weights all being high, but with different signs between QT and phased beat spectrum on one hand and QTc and beat spectrum on the other hand. Similarly, the frequency of the highest peak and the frequency of the fifth highest peak in the beat spectrum are weighted at similar strength, but with different polarity.

4.11 Graphical user interface and result presentation

The `ecghelper` systems, both old and new, have a two stage user interface. During the interactive ECG recording, LED signals are used to provide status information. In the `ecghelper2` device, this is augmented by beep signals to support visually impaired users and to make the device more friendly to use in general. While recording, the incoming ECG is displayed on a screen, which is part of the embedded device.

Once recording is complete, the embedded screen is used to visualise the ECG analysis results. Thanks to having a much larger, higher resolution screen, the `ecghelper2` device can present more information in a more friendly way compared to the old breadboard system. For consistency, `ecghelper2` shows a scaled down, 320 × 240 pixel version of the type of diagrams which can be generated by the `offlineecg` corpus processing tool for Linux. So in the following, diagrams created from corpus ECG recordings will be used in place of embedded device screenshots.

There are some minor differences. For example, real-time recordings of course have no file name, so none is displayed on the embedded device. Also, the low resolution of the embedded display makes the icons for 'hum filter' (a drawing of a waveform) and noise (a block filled with a semi-irregular pattern) less elegant on the hardware version.

One improvement compared to the [Auer 2020] version is that the offline implementation now outputs the more modern TGA file format for images, instead of the older PCX file format. Both use

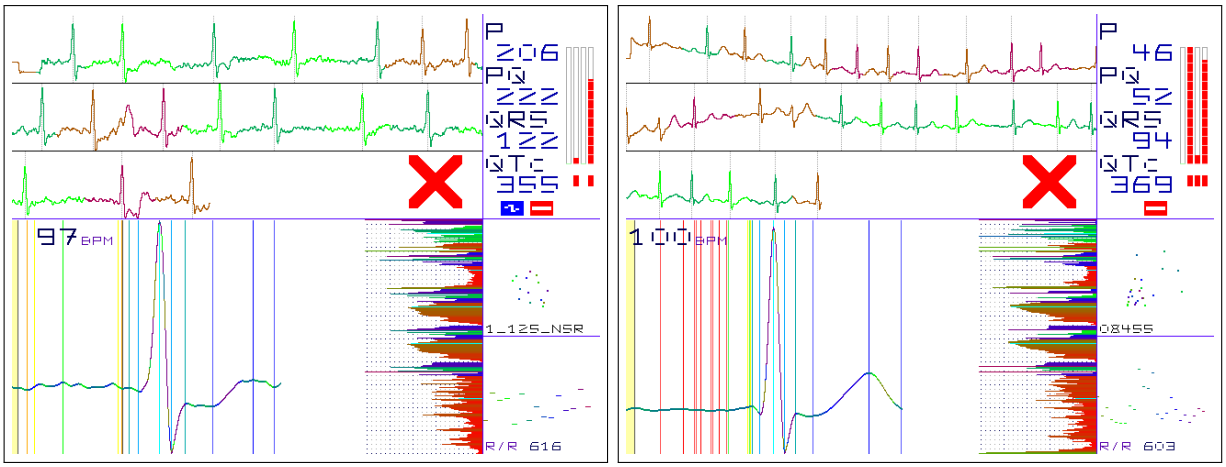


Figure 4.10: Left: Heterogeneous Dataset of Arrhythmia corpus example 1/125. Right: Atrial Fibrillation Database example 08455 (both classified as A-Fib)

run-length encoding to compress pixel data, but TGA takes a bit more CPU time to generate and the files have somewhat worse compression ratios. The latter is because PCX allows optimised encoding of individual pixels if their colour index has a small enough value. However, TGA images can be used directly in more situations, often permitting to dispense with additional conversion steps into other file formats. Note that TGA today also supports far more efficient compression algorithms, which is why TGA is still a relatively popular format. Luckily, modern software supporting TGA files implicitly still supports the classic RLE compressed variant of TGA.

Figure 4.10 shows two examples of ECG which are classified as A-Fib by the algorithm. The left recording, from the Heterogeneous Dataset of Arrhythmia, have metadata including: 'Normal sinus rhythm with right bundle branch block @ 90 bpm / Ventricular ectopy / Changing P-R ? A-V Dissociation / Junctional rhythm / Left anterior hemiblock / Left axis deviation'. Nevertheless, it is classified as A-Fib. The plot (figure 4.10 features different types of data:

At the top left, the entire ECG is shown, wrapped over three rows. Alternating tones of green mark beats used for averaging, while reddish tones indicate beats not used, for example because they differ too much from the others. The large red cross indicates the A-Fib classification.

At the bottom left, the averaged ECG shape is displayed, along with the average pulse rate. More to the right, a visualisation of the feature vector is plotted. Different colour gradients are used for different categories of features. For example the global spectrum features near the bottom use a red to green gradient.

The bottom right is used for R/R related plots. The average R/R period (in milliseconds) and, of course only in the offline corpus processing version and not in the embedded device, the file name are displayed in this area, too. The plot at the bottom shows the R/R intervals from first to last. It can be seen that they change in irregular ways for this recording. A Poincaré Plot above it, using current and previous R/R value for each R/R interval as the coordinates for a pixel cloud, illustrates the irregularity in another way.

At the top right, some ECG metrics are shown as text and numbers: P-wave duration, PQ interval, QRS complex duration and QTc interval. Below, icons indicate the classification status. The blue 'wave' icon means that additional hum filters were triggered and the red 'minus' icon stands for the A-Fib classification result.

Three bar graphs show the partial results on the `ecghelper2` display or the plots generated by `offlineecg2`, from left to right: P-wave score, feature threshold rules and LDA. Green bars are for NSR, red bars for A-Fib, taller bars mean clearer decisions. The global judgement is indicated with icons. A fourth, unused bar graph is visible between the bar graphs and the P, PQ, QRS and

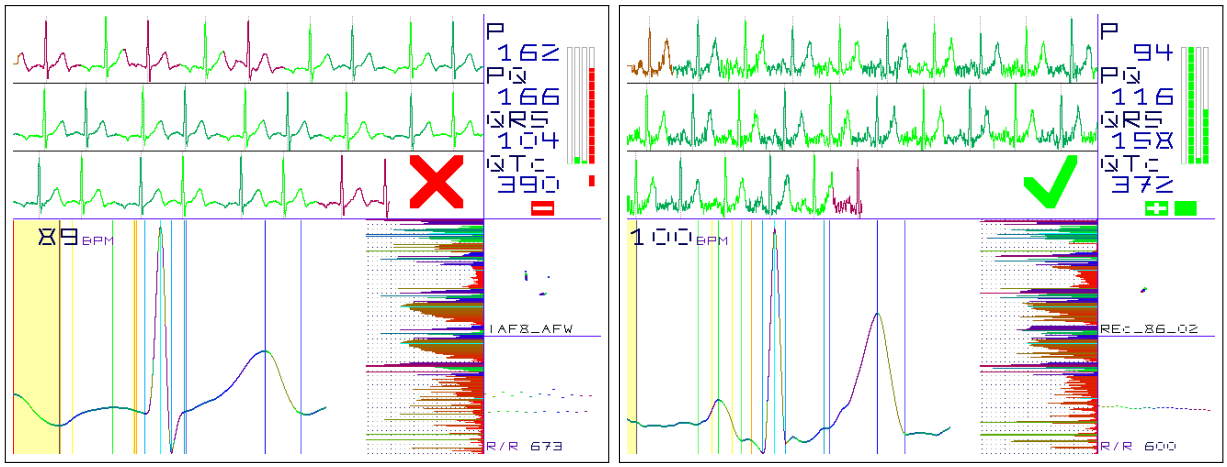


Figure 4.11: Left: Sample IAF8 AFW (Intracardiac Atrial Fibrillation Database) bigeminal A-Fib rhythm. Right: ECG-ID item 86/2, classified as NSR in spite of low ECG quality

QTc values. Red blocks underline each bar graph with a red (A-Fib) outcome, to make the plots readable even in black and white prints or for colour-blind users. For the same reason, all icons also have clearly distinguishable shapes, not just relying on colours.

Comparing left and right examples in figure 4.10, the left case has hum filtering on (blue icon) and selects one long, wavy segment as P-wave candidate. The right case finds very little P-wave (longer left red bar graph) and red time markers in the averaged ECG show that irregularities now are significant compared to the P-wave. As the P-wave is either malformed (left) or too small (right), the LDA outcome (right bar graph) is a clear A-Fib (red) in both cases. Overall classification (red icons) is A-Fib in both cases. The right example is from the Atrial Fibrillation Database [Moody 1983], which belongs to a study using R/R metrics to detect A-Fib.

Figure 4.11 shows another A-Fib example at the left, taken from the Intracardiac Atrial Fibrillation Database [Peterson 2003], and an example of Normal Sinus Rhythm, from the ECG-ID biometry corpus. As metadata confirm, the A-Fib example shows a case of atrial flutter. In a bigeminy pattern, R/R intervals alternate between two durations, multiples of the underlying atrial flutter period. The R/R plots show characteristic clustering patterns. The averaged ECG fails to present a normal P-wave. Instead, a wide, shallow hill is found, barely passing the P-wave score test. The LDA score clearly indicates A-Fib (here A-Flutter), leading to the overall classification as A-Fib.

In the right half of figure 4.11, a case from the ECG-ID corpus demonstrates how the algorithm copes with artefact-rich ECG. After averaging, the ECG shape is measured as having plausible properties for NSR. Some small extra waves are present in the averaged ECG, but stay below configured thresholds when compared to the P- and T-waves. The R/R plots look relatively normal and both LDA and P-wave score strongly suggest this recording to be in the NSR class. Note that the global spectrum, shown as part of the features, has some suspicious peaks at middle frequencies. As the ECG-ID corpus does not have suitable metadata, there is no gold standard diagnosis given for the recording. The NSR classification by the `ecghelper2` algorithm does seem plausible, though.

4.12 Additional interactive user interface modalities

When using the embedded `ecghelper2` device, in addition to the graphical result presentation and real-time ECG scope and progress display mentioned in sections 4.11 and 4.4, the device also employs RGB LED and audio output for real-time status signals.

Table 4.6 lists the audio signals and signals presented using the RGB LED of the `ecghelper2` hardware, for the current version of the firmware. Signal frequencies have been kept low and at octave

Event	Audio signal
System boot	440 Hz, 1/4 s
Signal lost, return to standby	220 Hz, 2 × 1/4 s
Measurement complete	880 Hz, 2 × 1/8 s
Classification: NSR 'good'	440 Hz, 1/2 s
Classification: NSR 'okay'	440 Hz, 1/4 s
Classification: A-Fib 'bad'	220 Hz, 3 × 1/2 s
Classification: not possible 'noisy'	220 Hz, 880 Hz, low/high/low/high
Bluetooth activated	110 Hz, 1/2 s
Bluetooth done, return to standby	880 Hz, 2 × 1/8 s
Bluetooth failed, return to standby	880 Hz, 4 × 1/8 s

Event	RGB LED signal
Standby state	off, display off: Hold electrodes to start
Boot process, calibration	bottom (status) LED orange
Calibration done	status LED green, top LED dark green
Recording data	top LED dark green, status LED green
Recording data	status cyan flashes for QRS complexes
Lead-off, signal loss	status dark blue, dark yellow if retrying
Classification: NSR 'good'	both green
Classification: NSR 'okay'	both green/yellow
Classification: A-Fib 'bad'	both red
Classification: not possible 'noisy'	both magenta
Bluetooth activated	both blue
Bluetooth done, return to standby	cyan
Bluetooth failed, return to standby	red/magenta

Table 4.6: Interactive audio and RGB LED signals used by the embedded ecghelper2 device firmware

distances from each other to be easy to recognise even by users with hearing impairments. Some RGB LED colours have been chosen to reduce energy consumption, while those for classification results match those of the corresponding display icons. Bluetooth related events correspond to blueish LED colours.

4.13 Bluetooth access

Bluetooth access is enabled by the user keeping contact to the electrodes at the moment when the result display timeout expires. This gives the users the freedom to decide whether or not they want to share their ECG, for example for telemetry or to archive the data locally. If the user stops touching the electrodes earlier, the device returns to standby immediately, without allowing Bluetooth access.

At the moment, there are no additional security features, but it would be possible to display for example a pairing code on the display of the `ecghelper2` display and require the user to enter that on their peer device. This would block eavesdroppers in radio range of the device to connect to it. They still have to be nearby at the right moment, of course. It is expected that most measurement cycles will take place without enabling Bluetooth, because the device already computes and shows the full ECG analysis without the user having to make any data accessible.

Note that Bluetooth communications draw significantly more power than normal measurement cycles, affecting battery life. For testing purposes, the software also copies all Bluetooth data to (and from) the USB serial interface, which can be used to log data and to send commands which would otherwise be received wirelessly. Note that for safety reasons, the USB connection should

not be used while recording actual ECG. If it is used, the USB peer should be battery powered. A separate lid prevents normal users from using the USB socket while the housing is properly closed and even while the battery lid is open.

Once the user has activated Bluetooth and a peer has connected, the `ecghelper2` device will send a prompt message to the peer:

```
'+M+S+R+F+E+?+X>';0;0
```

The user can now send a command, which is either `+M`, `+S`, `+R`, `+F`, `+E`, `+?` or `+X`. Requiring a plus sign as prefix protects the device from interpreting text from unintentional pairings as commands. For example, a smartphone or computer might detect that the device became available for connections and assume that it is a modem, sending some modem commands to it to identify the device further. The question mark command makes the `ecghelper2` firmware display a help message, the X command is used to end the connection. Any timeout event also aborts the connection and lets the device return to standby. On initial connection, the firmware also sends the most important values in a CSV style format:

```
'JUDGEMENT';0;???
'SCORES';0;0x????????
'BITMAP';0;0x????????
'NUMMETRICS';0;???
'NUMSHAPE';0;???
'NUMR/R';0;???
'NUMFEATURES';0;???
'NUMECGSAMPLES';0;???
'LSBPERMV';0;???
'+M metrics +S shape +R r/r';0;0
'+F features +E ecg +? help +X exit';0;0
```

As can be seen, the first column always is a string, the next column is zero and the last is a numerical, possibly hexadecimal, value, shown as ??? in the template above.

Judgement can be any of the constants in `ecganalysis.h`: 1 for 'good' NSR, 2 for 'okay' NSR, 3 for 'bad' A-Fib, or 4 for 'noisy' (unable to classify as either NSR or A-Fib). Scores are a 32-bit value created by concatenating up to 4 byte values used in the bar graph displays. At the moment, those indicate the P-wave score, threshold rule vote score and LDA score, leaving one slot unused. The bitmap indicates which beats have been selected for averaging and the LSB/ mV value is used to convert raw ECG amplitudes into millivolt amplitudes.

All 'num' values indicate the lengths of vectors available on request by other commands – metrics (ECG shape metrics), shape (samples from the averaged ECG), R/R (R timestamps and R/R intervals), features (the feature vector) or ECG samples (the raw ECG recording).

The `+M` command fetches the ECG metrics. Each metric has a number (second column), value (third) and label (fourth), for example:

```
'METRIC';1;148;'t:P'
'METRIC';2;28;'t:Q'
...
'METRIC';14;398;'QTc'
...
```

With the `+S` command, the shape is transferred. Columns here are sample index (second), raw value (third), slope (fourth) and marker colour (fifth, 0 for none), for example:

```
'SHAPE';0;-25;0;21
'SHAPE';1;-36;0;0
```

```
...
'SHAPE';55;-58;-8;29
...
'SHAPE';80;151;455;0
'SHAPE';81;166;460;60
'SHAPE';82;181;452;0
...
```

The `+R` command reads the R/R data, with the numerical columns index (second), timestamp (third, in milliseconds), R/R (fourth, also in milliseconds). Note that early R/R events are not detected by the embedded software, as it first needs to adjust sensitivity thresholds:

```
'R/R';0;2376;814
'R/R';1;3190;832
'R/R';2;4022;842
'R/R';3;4864;838
...
```

As expected, the `+F` command reads the feature vector. There are 256 unsigned byte-valued features in the current implementation:

```
'FEATURE';0;51
'FEATURE';1;57
'FEATURE';2;30
'FEATURE';3;31
...
```

Finally, the `+E` command can be used to fetch the raw ECG data. As this will be thousands of individual samples, a more compact way to encode them is used. This keeps the transfer time and, importantly, the energy consumption limited:

```
'ECG';0;-1;0;0;1;0;0;0;0;-1;0
'ECG';10;-1;0;0;0;0;0;0;0;0;-1
'ECG';20;-3;-3;-5;-12;-23;-43;-75;-130;-234;-452
'ECG';30;-1966;-2136;74;454;463;114;-154;-139;-4;78
'ECG';40;-3087;220;279;231;156;121;92;37;32;142
'ECG';50;-1480;333;172;-40;-71;141;435;491;168;-247
```

In this format, only one in ten samples is transferred as raw value. The nine next samples are instead transferred as differences to their predecessor, using nine additional columns.

To make use of the data after transferring it wirelessly, a dedicated `csvreader` tool has been created. The help screen explains the features:

```
Command line options:
--help prints this help (must be the only argument)
--tga=filename create TGA image of ECG diagram
--wav=filename store ECG recording as PCM WAVE file
--raw=filename store ECG recording as raw 16-bit data
--eric=filename same as --raw=filename
--bin=filename store features as raw unsigned 8-bit data
--binaryfeatures=filename same as --bin=filename
--features output feature list as text
--rrdata output R/R data (times and periods) as text

To re-process ecghelper 160 V/V recordings:
csvreader --eric=test.eric < bluetoothlog.txt
```

```
offlineecg --lsbpermv=5700 test.eric
```

As the tool can be used both to plot the analysis results and to create ECG data files again, it is possible to use the latter to re-process an ECG recorded earlier with a newer version of the algorithm, using `offlineecg2` (the binary is called offlineecg for all versions). Exporting features as binary data is useful for using them as training data for classifiers and exporting the raw ECG as WAVE audio file helps users without dedicated ECG software to visualise the data, for example using the free, open source audio editor Audacity[57].

As should be expected, diagrams generated by `csvreader` use the same style as those created by `offlineecg`, which in turn is just a higher resolution variant of the result display on the embedded `ecghelper2` device.

[57] `https://www.audacityteam.org/`

5 System evaluation

5.1 Hardware evaluation

As already described in section 3.14, the prototype design matches the requirements envisioned for the hardware aspect of this thesis. As with the breadboard system, it is able to measure single-channel ECG at a quality suitable for the required analysis and classification tasks. At the same time, the new hardware presents a range of new features, mainly in the areas of user interface and power supply.

For the user interface, the `ecghelper2` hardware introduces audio feedback, a larger, higher resolution display and two low-power full-colour RGB LED instead of the high-brightness NeoPixel array used in [Auer 2020] which draws between $\approx 1\,\text{mA}$ (standby) and $> 50\,\text{mA}$ (full intensity) per LED.

Instead of external, wired electrodes or disposable stick-on electrodes, the new device uses stainless steel electrodes integrated into the housing, which also holds two AAA size batteries as energy source, for a compact, fully integrated design. It is still possible to connect external electrodes, for full flexibility in a variety of use cases. In addition, the device has been designed to be free of specific safety risks for the users, as explained in section 3.2.

On the power supply side, the new hardware has been designed to maximise battery life both in standby and during measurement cycles. While the current design promises months of battery life and many ECG measurements with a single pair of AAA batteries, it has a power supply which fails to deliver sufficient current on the 3.3 V rail as soon as the input voltage drops to circa 1.8 to 2.0 V. The supply voltage then experiences brown-outs depending on load, measurements are no longer possible. This affects in particular the ability to use `ecghelper2` with rechargeable NiMH batteries, as those have lower cell voltages and the combined output voltage of two NiMH batteries will drop below 2.0 V much earlier than the combined voltage from a pair of alkaline batteries.

To address this issue, a proposed third generation version of the device, described in section 3.15 will feature both even lower standby power – introducing power supply gating for the LCD display – and an even more efficient power supply, which can work with lower input voltages. According to the datasheet, 0.5 A at 1.5 V, which should be sufficient for all ECG measurements and possibly even for Bluetooth transfers.

Table 5.1 presents some early current measurements for the prototype. By limiting the usage of CPU speed and peripheral devices, the current software reduces energy consumption. One trivial example is the use of a dimmed green LED signal with cyan flashes during the measurements. The green LED channels use only 1.5 mA each at full brightness, while red or blue channels use 6 mA each to achieve similar brightness. Also, power consumption varies significantly with CPU clock speed, according to the ESP32 datasheet and practical experience.

Additional oscilloscope screenshots regarding SPI signal quality on one hand and (qualitative) tests of the power consumption on the other hand can be found in appendix A.1.

As the measurements of power consumption show, standby operation is marked by narrow single cycle current spikes of the voltage converter, at frequencies below 30 Hz. While the device is active,

Battery type	Event	Voltage	Current	Power
Alkaline	Standby	3.13 V	87 to 102 µA	0.27 to 0.32 mW
NiMH	Recording	2.2 to 2.3 V	165 to 170 mA	0.38 to 0.40 W
Alkaline	Recording	≈ 2.8 V	124 to 133 mA	0.35 to 0.38 W
Alkaline	Bluetooth	2.48 to 2.54 V	276 to 308 mA	0.71 to 0.76 W

Table 5.1: Power consumption measurements for the ecghelper2 prototype design

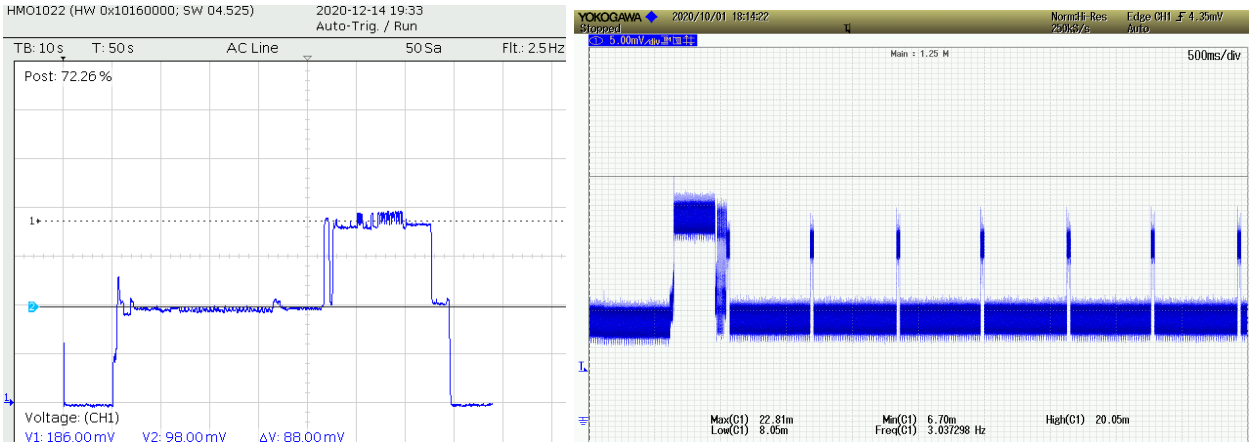


Figure 5.1: Left: Current over time for full measurement cycle with Bluetooth (scaled to 1 mV per 1 mA, 2.5 Hz low-pass). Right: Bluetooth connection setup current, qualitative

a sawtooth current pattern can be observed, with currents depending on CPU load, Bluetooth activity and usage of peripheral devices such as RGB LED, LCD display or audio output (see appendix). The power consumption of the MAX30003 can be neglected compared to other loads during measurement sessions, so it is not expected to be observable on the screenshots.

Using extreme (2.5 Hz) low-pass filtering, it is possible to plot the current consumption of the device over time, in the left plot of figure 5.1. When booting, the `ecghelper2` firmware first tests the different peripheral devices and computes a splash screen (initial spikes). Then, self-calibration takes place and the recording starts, visible as a few seconds of constant current (a bit over 10 seconds after booting). Next, the flashing heart beat indicator LED current is visible as ripples in the plot for roughly 20 seconds. A short rise in power consumption marks the data processing step, followed by a period of constant current while the results are shown on screen.

As the user has enabled Bluetooth by keeping in touch with the electrodes (actually, the plot uses a simulated 80 bpm ECG from a signal generator) the wireless system is activated next. Power consumption almost doubles. A short drop occurs while waiting for the peer to connect, then currents return to even higher levels, depending on bandwidth usage over time. There are more than 20 seconds of Bluetooth activity in this session, enough to transfer not only all analysis results, but also the complete ECG recording during the 10 seconds with peak power consumption. At the end of the measurement cycle, a sign-off screen is displayed for a few seconds and the device returns to standby state with minimal energy consumption.

A zoomed in plot on the right of 5.1 shows the period of waiting for a Bluetooth connection. With less filtering, currents can be seen to be a lot more irregular than in the overview on the left. This means that it is important that the power supply of the device has a good load step response.

The large difference between both views also posed the question whether the standby current measurements have been accurate. Measurements with an Agilent U1252B True RMS multimeter cast some doubts here at first. With a lab power supply set to 3.1 V, the `ecghelper2` prototype draws 85 to 90 μA DC, but also 258 μA AC (25 Hz) and up to 300 μA RMS. Using a 0.5 Ω **shunt** and measuring half of the voltage drop at the shunt through a **voltage divider** ($2 \times 10\,\text{k}\Omega$) even shows 40 μV DC, 1.17 mV AC and 1.17 mV RMS. Luckily, the AC component turns out to be only a risk of EMI, not a risk of excessive battery drain:

With a 1 μF capacitor in parallel to the 10 kΩ resistor where voltages are measured, creating an ≈ 30 Hz low-pass filter, shunt voltages are far lower: 15 μV DC and $< 40\,\mu$V AC and RMS. The latter two even drop to 15 and 20 μV when using a 10 μF capacitor instead (≈ 3 Hz low-pass filter). The active currents correspond to ≈ 25 mV (DC and RMS) while recording, or up to 46 mV for Bluetooth transfers with both cut-off frequencies, with little AC (≈ 1 mV) at the measurement

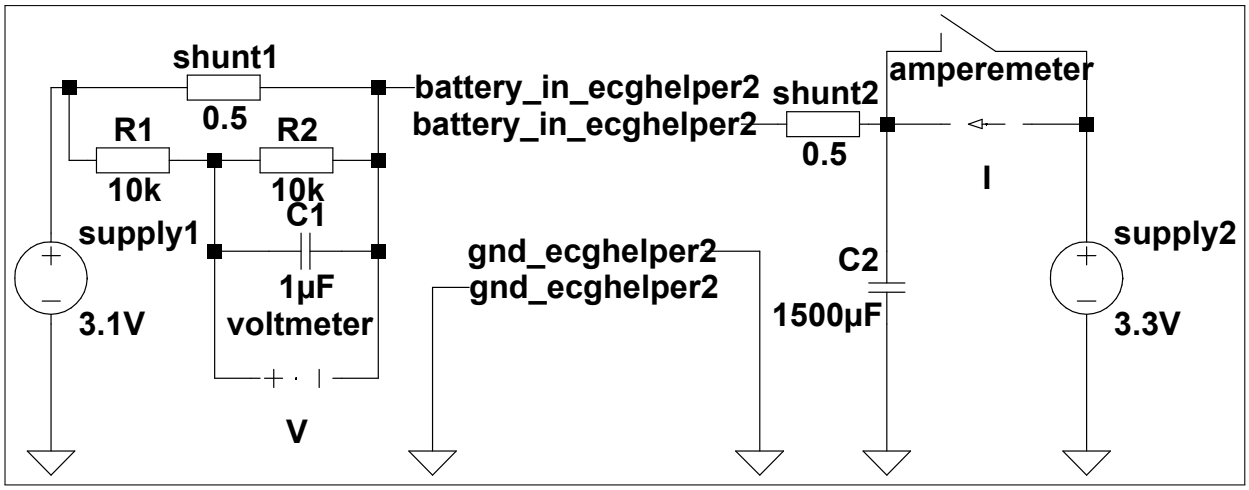


Figure 5.2: Left: Current measurement using a shunt, voltage divider and low-pass filter. Right: Current measurement using a low-pass supply filter (switch to measure only standby in μA range)

points. The left half of figure 5.2 shows this **low-pass filtered shunt voltage** based approach, which may underestimate the actual currents.

Yet, **direct current** measurements with the U1252B show 90 μA DC standby, with as much as 1.45 mA AC and RMS! While recording, 100 to 105 mA DC and 110 to 115 mA RMS. Even up to 290 mA (in peaks) RMS can be observed for Bluetooth.

To confirm that the relatively high AC currents are at most an EMI issue, but not related to higher than expected battery drain, the current measurements have been repeated with a **RC filter directly in the power supply rail**, as shown in the right half of figure 5.2. The large capacitor in the supply filter holds enough energy to keep the device in standby for more than 20 seconds with disconnected supply. Longer interruptions will result in a cold boot when power returns.

Now, the standby current can be found to be on average 104 μA RMS (with 5.6 μA AC, 'battery' voltage > 3.2 V). While recording, 112 mA RMS and 1.6 mA (peak 3.9 mA) AC have been measured as averages with the Agilent U1252B True RMS multimeter. Input voltage drops to 2.5 V at the `ecghelper2` battery terminals in some situations (depending on LED, sound and CPU activity) during the recording, so it was not possible to check the much higher Bluetooth-related currents without changing the setup again.

The < 4 mA, probably higher while using Bluetooth, AC component remains to be considered as possible EMI source in the battery circuit itself, although that has relatively short traces. Yet, the third generation design reduces EMI by moving the batteries close to each other at the bottom of the circuit board. This updated design also improves bus signal and power distribution layout.

5.2 Cost considerations

The net component price of the prototype design is between 60 and 65 Euros including display, housing, printed circuit boards (PCB) and stainless steel electrodes. The Bill of Materials can be found in appendix A.2. However, actual device costs have been between 90 and 100 Euros, including all taxes, postage and additional materials:

The displays had to be imported directly from Waveshare and a solder paste stencil has been ordered along with the PCB to prepare for the SMD manufacturing process. Also, building the housings only required part of the materials bought, as no smaller quantities were available. Of course, this price does not include the use of assembly and other lab facilities and the significant amounts of time spent in development of the device.

Among the SMD costs, the core power supply components including passives and LDO, but excluding ideal diodes and similar, are roughly 10.50 Euros. The smaller, more efficient power supply of the third generation device would reduce costs by 2 Euros, or 3 when also using a Richtek RT9073N-18 LDO instead of the current Maxim MAX1726EUK18. At the same time, reducing board area by one third, the proposed new device generation can also be built at significantly lower PCB costs.

On the other hand, the proposed new design features additional sensors. A BME280 environmental sensor (3.20 Euros) and a BMX160 inertial and compass sensor (5.70 Euros) and several SN74AUP3G34 triple buffer gates for power gating etc. ($\approx$ 1 Euro per device). By saving around 2 Euros worth of debug and extension headers, the overall costs for the device will be very similar to those for the prototype, given the additional cost reduction from shrinking the PCB area.

The audio output feature contributes around 4 Euros to the costs of either design, second generation prototype or third generation, while the displays including headers, spacers, screws etc. clearly drive costs, at 12 to 23 Euros. With suitable order planning, the lower value can be achieved. So future third generation prototypes, in spite of having more features and improved power management, can be built at significantly lower prices.

Excluding the USB interface from the board could reduce costs by an additional 6 Euros or more, also simplifying the power infrastructure, but of course it will make firmware installation and debugging harder since it would require a dedicated external programming device.

5.3 Matching embedded hardware and efficient software

The improved user interface peripheral hardware makes it possible for the `ecghelper2` system to offer a friendly user experience for both untrained users and users with more experience with ECG. The former can rely on audio signals, coloured status LED and easy to understand icons (tick for NSR, cross for A-Fib etc.), while the latter can also use the displayed average ECG shape, some selected timings (as text) and an overview of the recorded ECG for further information or simply to see whether the recording has been low in noise and drift.

Although the ESP32 WROOM32 module is both small and very affordable ($\approx$ 4 Euros at small quantities), it has proven to bring far more CPU power, RAM and storage than required for `ecghelper2`. There would be enough RAM to store a few minutes of ECG recordings with dynamic memory allocation. As less than one minute are needed for the algorithm, all temporary data, for example for floating point spectral computations, use static allocation for simplicity.

The ESP32 has hardware-accelerated multiplications for both integer and single precision floating point values (according to the datasheet and to a `xtensa-esp32-elf-objdump -dS` listing of the FHT object file) and other operations which reduce CPU load for basic signal processing such as FIR filters. Based on profiling, even at 80 MHz, the ESP32 spends more than 90% of the time waiting for new ECG samples to arrive in the MAX30003 FIFO and less than a second to analyse a full recording before displaying the results.

The algorithm and training also are efficient on the Linux side. It takes less than half a minute to process more than 7 500 recordings from PTB-XL (10 s each) and store their feature vectors. Computing optimised threshold rules from them takes less than 20 s and LDA based generation of an efficient transform of feature vectors into a single A-Fib versus NSR rating value takes only a few seconds. This allows to manually tune other high level parameters of the feature extraction and classification algorithms across multiple training runs to find the best balance for practical use of the `ecghelper2` system.

The `offlineecg2` Linux implementation of the algorithm has been profiled by letting it process a 10% subset of the entire PTB-XL corpus (not limited to NSR and A-Fib) based on selecting all files with file names matching `*5_hr.*`, in other words all files with 5 as the last digit of the record

number. Command line options for `offlineecg2` were `--tga --gain=5 --lsbpermv=1000`, as the PTB-XL corpus nominally uses 1 μV amplitude units and a gain of 5 has been empirically found as a good choice for processing.

Note that the software also has an automatic gain feature, but this would cause variable additional computational load, making it less interesting for profiling. Also note that TGA image writing takes a significant fraction of the overall runtime of the software. An updated version of the software skips all graphical processing if no image files are to be generated.

When using the `--features` option instead of the `--tga` option, the runtime drops to 45% of the time with TGA output – less than 2 ms for processing each 10 seconds of ECG data on a single desktop CPU core ($\approx$ 2 GHz CPU clock selected for this by automatic operating system power management). Neither GPU nor multithreading were used.

More than half of the runtime is spent for graphics related calls if TGA output **is** enabled: 39% `tgaOutput`, 11% `canvasLine`, 3% `showAllSamples`, 0.6% `showPulseShape` etc.

The embedded version does not have to compress graphics as TGA files, but it has to transfer pixel data (and coordinate ranges of the screen areas receiving updates) through a SPI connection with limited bandwidth. Still, the ESP32 easily handles the task at hand with 10 MHz display bus clock and 80 MHz CPU clock (up to 240 MHz selectable on ESP32 WROOM32 modules) including all signal processing and analysis and responsive display updates.

Some simple profiling by accumulating the amounts of time spent during versus between calls of the main worker function suggests that recording could work smoothly at 10 to 20 MHz with some optimisation of the ECG status flag polling and FIFO reads, as far more time is spent in waiting than in processing. One possible approach would be to use more interrupt signals from the MAX30003. Note that while the ESP32 supports even lower clock rates, some interfaces may pose lower bounds for possible CPU (or internal bus) clock frequencies.

If TGA output is **not** active, most CPU cycles of the Linux version are used for filtering of the ECG: 15.1% + 3.5% for `qrsFilter` and `rrDetector` (QRS detection), 13.7% for `firFilter` (50 Hz comb notch etc.), 4.2% + 3.8% for `biquadExtraFilter` and `biquadFilter` (187.25 Hz + 60 Hz notch). Only the Linux version uses the biquad filters. They are not needed for the embedded firmware, to be used in areas with 50 Hz mains frequency. Also, ECG averaging already reduces 60 Hz artefacts as part of the general signal to noise ratio improvement effect, even if no biquad filters are used.

In the Linux version, the 187.25 Hz biquad filter is always used, the other biquad notch is first applied in a simulation and only enabled for the actual signal if found to be necessary. This avoids unnecessary distortion when no significant 60 Hz hum is present in the actual ECG recording data.

Considerable CPU time is also spent in spectral analysis (9.7%+2.4%+1.4%+0.9% `fht_dif_iter`, `applyBlackmanWindow`, `bitrev_permute_real` and `fht_dif`) and related functions (4.5% + 2.8% `analyzeSpectrum` and `analyzePhases`).

The third major user of CPU time are the beat selection and averaging: 7.1% + 4.5% for `dataRMSD` and `dataRMS`, 1.2% + 3.3% for `ecgAveraging` and `ecgSmoothing`.

Remaining significant contributions are: 3.1% + 2.4% `findMinIndex` and `findMaxIndex` (ranges, shape analysis), 2.6%+1.2%+0.9% (only in the Linux version) `processOneFile`, `ecgPreprocessing` and `ecgReadFile`, 1.7% + 1.7% `writeInt` and `itoa`, and 0.9% `intsqrt`, with less than 6% of the total runtime spent in other functions not mentioned here.

Constant	Value	Description
DIGITALHPF	0	Use MAX30003 0.5 Hz IIR high-pass filter next to analogue HPF (November 20 version: 1)
DOUBLEPFACTOR	5/9	Min. height of additional waves relative to P-waves (or average of P- and T-waves) to count as problematic
ECGBUFSIZE	17500	Max. ECG recording duration in samples (35 s)
FEATURERELAX	5	Feature rule threshold trigger margin (fewer hits)
FHTBUFSIZE	4096	Max. Fast Hartley Transform size (Linux version: 16384)
GLITCHLIMIT	5	Max. glitch count before rejecting **offline** recording
GLITCHTHRESHOLD	500	Max. jump of signal in μV before rejecting it as QRS candidate
HUMTHRESHOLD	40	Min. spectral value near notch frequencies to activate additional 50 Hz or 60 Hz smoothing of average ECG
LOWBINFACTOR	1.5	Frequency factor relative to pulse frequency for lower limit of energy-normalised range of beat spectra
LOWBINFACTORFINE	1.2	As LOWBINFACTOR, but for more fine-grained global spectrum
MAXAUTORANGE	16000	Linux auto-gain will not amplify peak-peak range above this
MAXBOOSTFREQ	45	Infinite amplification frequency of de-emphasis
MAXBOOSTRANGE	40	Upper frequency limit for applying de-emphasis
MAXNOISELEVEL	25	Max. noise estimate in μV before rejecting ECG (can be 15 for 40 Hz low-pass filtered ECG, some old corpora need 25)
MINAUTORANGE	7000	Linux auto-gain will amplify peak-peak range to at least this
MINDURATION	7	Min. usable recording duration (Linux version skips over initial artefacts if necessary and ECG long enough)
MINNOISEVOTES	3	Min. rule votes for 'noisy' category to neither classify a recording as NSR nor A-Fib (2 too picky for old corpora?)
NORMENERGY	1.9	Normalisation goal for average signal energy in beat spectra
NORMENERGYFINE	1.0	As NORMENERGY, but for global spectrum: Saturate less
PQRPLOTWINDOW	125	Number of samples before R-peak to show in average ECG
PQRWINDOW	160	Max. number of samples before R-peak to scan for P-waves and artefacts, automatically reduced at high pulse rates
PT60HUMTHRESHOLD	30	Linux only: Min. mean effect of filter, in 1/1000 of range, to warrant application of the 60 Hz biquad notch filter
PTHFACTOR	31	Min. height of NSR P-wave in averaged ECG as fraction of R to S peak-peak amplitude (22 to 31 recommended)
QRSLEARN	750	QRS detector learning period in samples (1.5 s)
QRSLOCKOUT	100*	QRS lockout period in samples: Ignore QRS detector triggers for 200 ms after each R/R event. *Actually 100 + QRSLATENCY
QRSMAXTHRESHOLD	22500	Upper limit for QRS detection threshold tracking (depends on automatic or manual gain and signal range)
QRSMINTHRESHOLD	256	Lower limit for QRS detection threshold tracking
RRBUFSIZE	26	Max. number of R/R events to collect, if enough ECG buffer space (embedded) or offline data (Linux) available
RSEARCHWIDTH	75	1/2 Search window size in samples for moving detected R-peaks to a nearby sample with highest amplitude
SLOPELEN	9	Half of FIR filter length for binomial derivative (9 or 17)
SMOOTHEXCLUDE	30	1/2 QRS exclusion size in samples when smoothing average
SMOOTHPASSES	13	Number of 3-tap moving avg. passes used to smooth average ECG, minus 2 or 3 if 8- or 10-tap moving avg. hum reduction on
SPECSMOOTH	4	[2 7 14 7 2] / 32 FIR smoothing passes for global spectrum (3-4)
TUNEPREQRSSLOPE	1	Attempt to make pre-QRS average ECG flatter (November 20: 0)
UNFILTERHPF	216	Fraction of area integral to add to the avg. ECG per sample to compensate high-pass filter (embedded only, 96 if DIGITALHPF is on, **0 for Linux**, fine-tune with 2 Hz square wave)
VOTEBEATSLDA	3	Min. number of rule votes to outvote LDA classification: Note that only 3 NSR rules exist at the moment, so a value of 4 would make rules unable to outvote LDA results saying A-Fib (2-4)

Table 5.3: Alphabetical list of key tunable parameters of the ecghelper2 classification algorithm

5.4 Algorithm parameter tuning

As the algorithm and training infrastructure for it can process entire corpora rapidly, it was possible to experiment with various settings and implementation details.

One tuning example has been ten-fold leave-one-out cross-validation of the feature threshold rules, using an earlier version of the generator which only stores the best rule for each feature. Each run generated between 144 and 152 rules. Processing the full data set generated 151 rules. Of all 1625 produced rules, including the full-set case, only 366 were distinct at all.

The 46 most stable rules were even identical for all ten validation folds and 45, 18 and 13 more were identical for 9, 8 and 7 out of ten folds, respectively. So more than 80% all rules were unchanged for the large majority of the training data subsets, and effects on classification quality regarding the left-out fold can be expected to be minimal. To give some examples of the opposite case, the trained threshold for feature 100 varied between 25 and 35 depending on fold, the one for feature 105 between 133 and 139 in nine of the folds, with the outlier being 154 for the remaining tenth way of using 90% of the data for training and leaving 10% unused.

The threshold for feature 142 varied between 118 and 153, which is one of the reasons for banning that feature in later versions. It relates to the 2 Hz bin of a spectrum, which can vary significantly based on whether the actual heart rate is near that frequency. Yet heart rate distribution within the training corpus should not be the basis for the distinction between A-Fib and NSR.

Whether or not rules are generated at all depends on whether they reach sufficient performance with any threshold. Rules which varied in that respect across folds were typically rules with near-extreme thresholds. For example 5 subsets have a rule for feature 165 being below 1, while the other 5 do not. Assuming that this feature often had low values for both A-Fib and NSR, the ability to distinguish specific cases is limited by the active value range. Therefore, the current rule generator only produces rules with thresholds which are **not** near 0 or 255.

Similarly, a number of parameters directly in the implementation of the algorithm has been optimised by hand, such as the required P-wave height in terms of a fraction of QRS complex height, or the number of smoothing passes applied to the averaged ECG shape before the morphology analysis is launched to extract shape metrics. Current settings for all tunable parameters can be found in the header (`*.h`) files of the software. Table 5.3 shows the central tunable parameters of the classification algorithm, as well as their values used in the November 20 and December 14 versions.

5.5 Classification performance

To evaluate the overall performance of the classification system, a number of statistical measures are used to describe the relative ratios of true and false positives and negatives.

For the purposes of this section, **positives** are recordings classified as A-Fib, while **negatives** are NSR classifications. A system which can exactly reproduce the gold standard classifications of the PTB-XL corpus would return 1 511 true positives and 6 125 true negatives. In the following, false negatives are called missed A-Fib, while false positives are called rejected NSR.

The system actually does not produce a binary result, though. The classification result can also be that the system finds itself unable to make a decision between the two classes. This ‘noisy’ case can either be treated as true (system refused to make the wrong decision) or as false (system failed to make the right decision). Depending on which interpretation is used, there is an optimistic or a pessimistic set of statistical scores for the system performance.

At an earlier point, the first generation `ecghelper` and `offlineecg` algorithm has been applied to both old and new corpora, old being those used in [Auer 2020] and new being PTB-XL. Based on the pessimistic assumption that undecided classes are wrong, this produces the scores shown in

Corpus	Precision	Recall	Accuracy	MCC	F-Score	DOR
Old, see text	69.4%	95.8%	91.3%	76.7%	80.5%	212
PTB-XL	46.3%	89.7%	76.5%	51.8%	61.1%	23.7

Table 5.4: Performance statistics for the old proof of concept ecghelper system (rounded)

table 5.4. Note that the old algorithm lacks optimisations to cope with the very short 10 s ECG fragment size of PTB-XL. The old corpora often lack gold standard annotations.

For simplicity, it has been assumed that all ECG-ID recordings are NSR and all recordings from special A-Fib corpora are A-Fib, while ignoring all other old corpora. For a more exact measurement, the old recordings would have to be classified by hand to have a reference. The ECG-ID corpus has a biometry background, so it can be expected that most recordings are from healthy subjects. The A-Fib corpora, on the other hand, contain only recordings from subjects with A-Fib, but the recordings are often much longer than the excerpts processed here. So it is possible that the excerpts used here do not show signs of A-Fib in some cases.

The table also introduces the set of performance metrics used in this section. Precision (fraction of positive detection outcomes which are correct: $TP/(TP + FP)$) and recall (fraction of actual positives which got detected: $TP/(TP + FN)$) are classics, as is accuracy (sum of true positives and true negatives as percentage of the entire set of cases).

Precision would be perfect for algorithms which only return A-Fib when they are sure, even if that means missing many less obvious A-Fib, as long as no NSR accidentally gets classified as A-Fib. Recall, on the other hand, would already be perfect when the algorithm simply declares all input to be A-Fib, because that way, it will never miss any actual A-Fib.

As class sizes differ, just maximising the total number of correct classifications might not be the best choice either. In the PTB-XL case, four times more NSR than A-Fib examples are provided. In the general population, far more NSR ECG than A-Fib ECG will be found. On the other hand, A-Fib ECG will be more common in known A-Fib patients. So it is good to list three more measures which attempt to be more robust.

The first metric is the Matthews Correlation Coefficient[58] (MCC) which is a value between 0 (random) and 1 (perfect classification) for useful systems. For systems which are more often wrong than right, it is between -1 (always wrong) and 0. It can be computed from the confusion matrix as:

$$\mathrm{MCC} = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP+FP) \times (TP+FN) \times (TN+FP) \times (TN+FN)}}$$

As can be seen, the MCC uses **all** fields of the confusion matrix in a balanced way. Another useful score is the F_1-Score or F-Score[59], which is the harmonic mean of precision and recall, **assuming** both high precision and high recall are both equally desirable. Note that the number of true negatives has no influence on it, as neither precision nor recall depend on TN:

$$F_1 = \frac{2 \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

The final metric used in the table is the Diagnostic Odds Ratio[60] (DOR) which is the ratio between positive and negative likelihood ratios and can be between 0 and infinity. If the DOR is more than 1, the prediction of the described system is more often right than wrong:

$$\mathrm{DOR} = \frac{\mathrm{TPR}/\mathrm{FPR}}{\mathrm{FNR}/\mathrm{TNR}}$$

[58] https://en.wikipedia.org/wiki/Matthews_correlation_coefficient
[59] https://en.wikipedia.org/wiki/F-score
[60] https://en.wikipedia.org/wiki/Diagnostic_odds_ratio

Comment	Prec.	Recall	Acc.	MCC	F-Score	DOR
Threshold rules & old P-wave score	93.7%	97.0%	98.1%	94.2%	95.3%	1999
Pure (raw) LDA: PTB-XL overfitting, causes 12 missed A-Fib in old corpora	98.4%	98.8%	99.4%	98.2%	98.6%	20706
20 September version: better rules & P-wave score, LDA, phase **in**stability, tachy/brady trained, < 10 unclassified?	96.5%	99.0%	99.1%	97.2%	97.7%	11285
20 November version: 10 FN, 33 of 1511 A-Fib unclassified, 48 FP, 40 of 6125 NSR unclassified, pessimistic case	94.3%	97.2%	98.3%	94.7%	95.7%	2360
20 November version: optimistic case, 7 missed A-Fib in old corpora, see text	97.8%	99.3%	99.4%	98.2%	98.6%	27814
14 December version: TUNEPREQRS-SLOPE off, pessimistic case, 14 FN, 35 FP, 28 A-Fib & 22 NSR unclass.	96.3%	97.1%	98.7%	95.8%	96.7%	3557
14 December version: TUNEPREQRS-SLOPE off, optimistic case	98.2%	98.9%	99.4%	98.2%	98.5%	20386

Table 5.5: Performance statistics for different versions of the new ecghelper2 algorithm applied to PTB-XL, pessimistic case of treating unclassified as wrong, unless noted otherwise (rounded)

To compute the DOR, the true and false positive and negative rates are used: FNR (miss rate) is $FN/(FN+TP)$, FPR (fall-out, probability of false alarms) is $FP/(FP+TN)$, TNR (also selectivity or specificity) is $TN/(FP+TN)$ and TPR equals recall (sensitivity), it is $TP/(TP+FN)$.

To test the system performance on other data than the training set, it has been applied to the old corpora used in [Auer 2020]. Extra metadata processing of the Heterogeneous Dataset of Arrhythmia [Medhi 2019] extended the set to 69 known A-Fib examples used for the newest measurements. Appendix C gives a verbose review of all individual misclassifications for both old and new (PTB-XL) corpora regarding missed A-Fib, as well as a quick overview of all PTB-XL NSR files classified as A-Fib by the algorithm.

While leave-one-out cross-validation may have given additional insights regarding the PTB-XL classification, it would still have been a within-corpus way of keeping training data and testing data separate. The problem with this is that the PTB-XL corpus contains shorter recordings made with more professional hardware than either the old corpora or recordings made by the embedded `ecghelper2` hardware. So, without having access to a population of test subjects with known A-Fib and a population of known healthy control subjects, using the old, often lower quality, corpora a test set seemed like the next best alternative. It tests the algorithm on data which is different from the PTB-XL data in various aspects and represents lower quality, but often longer recordings in comparison to the PTB-XL ones.

The 20 September version of the software misses 4 A-Fib cases in the old corpora, the 20 November version misses 6 A-Fib in the original old test corpora and 1 additional A-Fib in the Heterogeneous Dataset of Arrhythmia. Testing in [Auer 2020] treats the latter as only likely to include A-Fib, but does not use per-recording metadata to decide which recordings should be classified as A-Fib and which should be classified as NSR. The A-Fib cases missed by the 20 November version are: 04015, 04048, 04126, 05091, 05261, IAF5 TVA and 1/135.

Core algorithm and settings have not been changed after 20 November, with one exception: TUNEPREQRSSLOPE changed to 0 (do not attempt to make pre-QRS average ECG more flat). This changes the results a bit, shown in italics in table 5.6. **In general, it seems better to use this (14 December) setting in the future, as it manipulates the natural ECG shape less.**

Corpus (source)	**NSR 'good'**	**NSR 'okay'**	**A-Fib** (or A-Fl)	**neither 'noisy'**	**Comments** (also see text)
AFDB [Moody 1983]	4	1	**15**	3	expect **mostly A-Fib**,
	3	*0*	***17***	*3*	long recordings
IAFDB [Peterson 2003]	0	1	**21**	2	expect **A-Fib**, intra-
	0	*1*	***20***	*3*	cardial ECG available
ECG-ID [Lugovaya 2014]	**252**	**17**	24	17	expect **mostly NSR**,
	262	***24***	*16*	*8*	medium quality
Hetero... [Medhi 2019]	43	4	77	26	**see text!**
	41	*5*	*79*	*25*	
Telehealth [Khamis 2016]	61	29	112	48	low quality, various
	56	*31*	*122*	*41*	

Table 5.6: Classification results for current algorithm and old corpora: Normal text with TUNEPREQRSSLOPE setting 1 (20 November), *italics text* with setting 0 (14 December)

Seven missed A-Fib in the old corpora may seem a lot, but one old 'known A-Fib' corpus includes 23 very long recordings of > 9 hours of ECG each: [Moody 1983]. The analysis in [Auer 2020] and in this thesis uses only short excerpts of those, starting 4 hours after the beginning of each recording. That time span may not always contain ongoing A-Fib for all subjects. Apart from IAF5 TVA and 1/135, all cases of NSR detection outcomes in alleged A-Fib recordings were in [Moody 1983] excerpts. See appendix C for a discussion of all 7 possible false negatives.

At least one obvious case of atrial flutter (IAF5 TVA) has been missed by the algorithm, as pointed out in the appendix. The simple explanation for this is that no explicit training for recognition of atrial flutter has been conducted with the algorithms – neither the one developed in this thesis nor the old one used in the proof of concept system.

The entire PTB-XL database contains only 73 cases of atrial flutter (PTB-XL category: AFLT), of which 17 are in both the A-Fib (PTB-XL: AFIB) and atrial flutter category at the same time. Only 56 cases of pure atrial flutter remain in the PTB-XL database. Even fewer cases exist in the corpora used for the proof of concept project. So enhanced abilities to detect atrial flutter would be a nice to have feature, but there would be a lack of training data.

Features of the 20 November version include improved input flexibility by using voltage based metrics where possible (instead of A/D unit based computations), automatic adjustments of shape processing to estimated noise levels in the signal and improved beat selection for the averaging process. While this exhibits a good balance for processing the old corpora, it has reduced the performance on PTB-XL somewhat if the $33 + 40$ unclassified cases are counted as wrong. When counting them as correct (because they avoid false classifications), performance scores are better than in the 20 September version, which still tried to avoid unclassified cases, like the 19 August version (5 A-Fib and 0 NSR classified as unclassified then).

Applied to the selected PTB-XL subsets, the algorithm now reports 1 468 A-Fib, 1 'good' NSR, 9 'okay' NSR and 33 unclassified cases for the 1 511 A-Fib recordings and 3 129 'good' NSR, 2908 'okay' NSR, 48 A-Fib and 40 unclassified cases for the 6 125 NSR recordings.

For practical purposes, the current version of the algorithm, at the current settings for the tunable parameters, combines a good ECG classification performance at MCC $> 94\%$. With robust generalisation for practical applications, where input data differs significantly from data seen in the training set, for example regarding recording quality and duration.

To demonstrate cross-corpus performance, table 5.6 shows the classification outcomes when the current algorithm and settings (20 November version) are applied to the old corpora from the

proof of concept project. For the Heterogeneous Dataset of Arrhythmia, metadata have now been processed to extract two values for each file – LSB / mV conversion factor and the diagnosis.

According to metadata, this corpus contains 69 NSR, including 1 manually tagged by the author as inverse polarity. Of those, 53 are sinus tachycardia, 6 sinus bradycardia, 18 A-Fib, and 4 atrial flutter recordings. This metadata-based classification uses keywords found in the 'underlying rhythm' field. Metadata are actually more detailed than those 5 categories. Based on the numbers, the algorithm tends to classify files from the NSR group as A-Fib, probably based on **other** anomalies. All files in the corpus are expected to contain **some** type of arrhythmia. Looking at individual files, this indeed happened for 30 files, including the inverse polarity one. An additional group of 14 files from the NSR category of the Heterogeneous Dataset of Arrhythmia had the outcome 'unclassified'.

Of the tachycardia subset of the Heterogeneous Dataset of Arrhythmia, 27 files got classified as A-Fib, as well as 2 of the bradycardia files. From those subsets, $8+1$ were classified as 'unclassified' (neither NSR nor A-Fib). All 4 files from the atrial flutter subset have been classified as A-Fib, as well as 14 of the A-Fib subset files. One of the A-Fib files (1/135) has been classified as NSR, as discussed in appendix C. The remaining 3 files (1/97, 1/126 and 1/140) have been classified as neither A-Fib nor NSR. Those 3 files indeed are marked by unusually low recording quality.

Separately testing the algorithm on **all** AFLT items in PTB-XL, 66 are classified as A-Fib, 3 as 'okay' NSR (04874, 08953, 16401) and 4 as neither A-Fib nor NSR (06205, 10065, 10160, 10564). In 04874, the extra wave check fails to trigger because at 142 bpm, additional waves just get interpreted as twin peaked P-wave. The analysis of recording 10160 is similar, but the twin peak interpretation is near decision threshold, leading to an overall neither/nor classification. Recording 10564 has contradictory partial results, which gives it a neither/nor result when combining the parts. Fragment 08953 (55 bpm) shows a 428 ms QTc, but otherwise passes NSR tests.

Recording 06205 (141 bpm) does not narrow the P-wave search window enough. Parts of the previous T-wave are interpreted as P-wave, rules and LDA are undecided, overall classification is neither A-Fib nor NSR here. The same happens for recording 10065. Item 16401 passes NSR tests in spite of having only a lucky probability distribution instead of an actual averaged P-wave. A common feature of atrial flutter ECG in the feature vector view is a comb shaped global spectrum.

5.6 Leave-1-out performance with PTB-XL

To check the generalisation performance within the PTB-XL corpus, a leave-one-out cross-validation test has been conducted. For each value of the last digit of the PTB-XL recording numbers, the `ecghelper2` system has been trained using all recordings **not** matching that digit, then tested on the remaining ten percent which **do** match the digit. In addition, for each of the folds, the performance on the 'quick A-Fib' set of 63 recordings and the A-Flutter set of 73 recordings (see section 4.6) are reported as well. All results of this test can be found in table 5.7.

When the classification result is NSR, the sum of 'good' and 'okay' outcomes is specified separately as good + okay count. Note that the folds vary slightly in size, as the distribution of last digits in the recording numbers is not exactly flat in all used subcorpora.

For each fold, there are between 573 and 657 NSR recordings and between 142 and 166 A-Fib recordings in the matching test subsets. Of those, at most 4 A-Fib got classified as NSR (never more than 1 as 'good' NSR) and at most 7 NSR recordings got classified as A-Fib in a single fold. Between 2 and 6 NSR and between 1 and 5 A-Fib recordings got classified as undecided between NSR and A-Fib.

The worst performance in A-Fib detection itself gave NSR results for 4 out of 144 A-Fib recordings (fold 6), while in 3 folds (3, 8 and 9) not one single A-Fib recording falsely got classified as NSR.

Fold	63 Quick A-Fib			All 73 A-Flutter			A-Fib fold			NSR fold		
(digit)	NSR	A-Fib	??	NSR	A-Fib	??	NSR	A-Fib	??	NSR	A-Fib	??
0	0+1	61	1	0+2	69	2	0+3	137	2	299+305	4	2
1	0+1	61	1	0+3	68	2	1+3	158	4	304+290	5	2
2	0+1	62	0	0+2	70	1	0+3	145	3	327+284	5	4
3	0+1	62	0	0+2	70	1	0	153	2	340+273	7	6
4	0+1	62	0	0+2	71	0	0+4	144	5	329+328	6	6
5	0+2	61	0	0+6	66	1	0+2	142	3	313+277	2	4
6	0+2	60	1	0+4	68	1	1+3	139	1	306+267	4	3
7	0+1	62	0	0+2	71	0	0+1	144	1	312+292	5	4
8	0+1	61	1	0+2	70	1	0	147	4	328+290	5	3
9	0+2	61	0	0+5	67	1	0	155	1	279+299	4	2

Table 5.7: Performance in leave-1-out cross-validation with PTB-XL corpora. ?? stands for the classification result 'could be classified as neither NSR nor A-Fib' / noisy. For NSR, the table specifies 'good' + 'okay' results separately.

Performance for the special A-Flutter test set varied between 2 and 6 errors out of 73, plus undecided results for at most 2 recordings, in spite of A-Flutter examples being quite rare in the training data.

5.7 Performance with synthetic ECG

As in the proof of concept project, Prof. Dr. Michael Möller has supported the work of the author by making a Fluke PS420 patient simulator available at the lab – thanks for the support! Among other signals, the device can generate single- and multi-channel ECG for a variety of physiological and pathological cases, which allows testing the `ecghelper2` system not only with different NSR and A-Fib (and -flutter) signals, but also with a number of other (simulated) cardiovascular conditions.

The device can either be powered by a 9 Volt battery or by an AC power supply specifically designed for the purpose. Nevertheless, ECG signal quality is considerably worse when using the AC power supply, leading to interesting classification differences in some cases.

As expected, the `ecghelper2` system detects all three A-Fib ECG available from the PS420 as A-Fib – atrial fibrillation, atrial fibrillation half value and atrial flutter. The former and latter case are shown in figure 5.3. In addition, it correctly classifies all **adult NSR between 30 and 140 beats per minute** (bpm) as NSR.

The PS420 can also be configured to create pediatric ECG. In that case, NSR at 80 bpm is recognised correctly, while the algorithm fails to classify the 40 bpm variant in the AC-powered case, but not in the battery-powered case. Too few beats meet the averaging requirements, because the very high signal quality and regularity lets the algorithm put the bar too high for beat selection and because few beats per minute are available at 40 bpm in the first place.

Interestingly, the device classifies **pediatric NSR of 100 to 220 bpm** as A-Fib. The T-R peak-peak interval, QT and QTc are all shorter than expected by the training data based rules. The same happens for **adult NSR at 200 bpm** and, only when using a battery, 180 bpm.

NSR at higher heart rates (and at 160 bpm for the adult setting) are classified as neither A-Fib nor NSR, because those heart rates are uncommon in either of the training data sets. The PS420 can generate signals at 30 bpm and all multiples of 20 bpm between 40 and 300 bpm.

Recognition of 80 bpm NSR by the device is not disturbed by any of the simulated technical artefacts available from the PS420 (50 or 60 Hz hum, muscle and respiratory artefacts, baseline wandering).

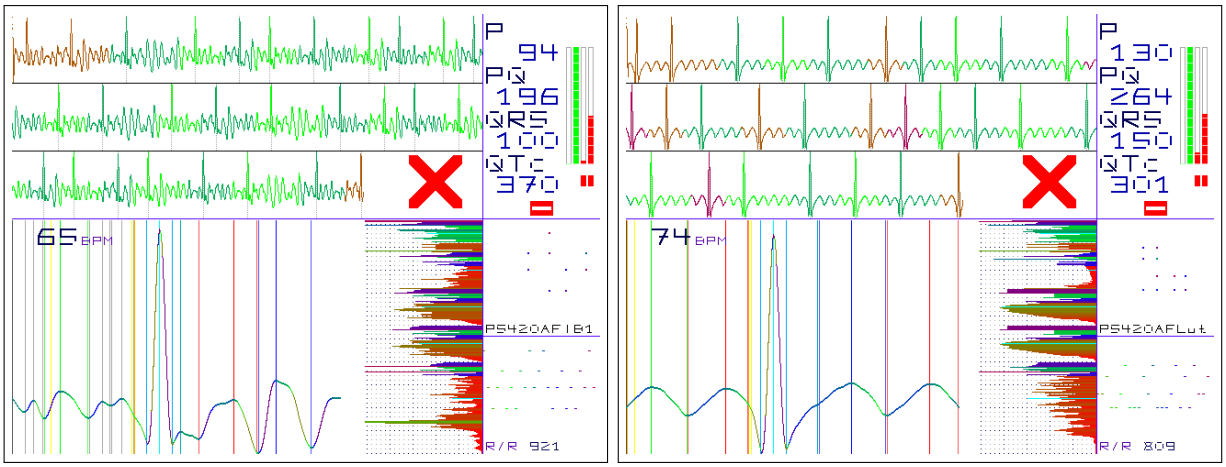


Figure 5.3: Left: Atrial fibrillation simulation by the PS420 device. Right: Atrial flutter simulation by the PS420 device

Also, various types of premature ventricular contractions (PVC) do not alter the NSR classification. Early and normal left and right PVC, pairs or clusters of 5 PVC, PVC at a frequency of 6 per minute. However, runs of 11 PVC or frequent PVC (12 or 24 per minute), as well as R-on-T right PVC often trigger A-Fib classifications, because too few NSR beats are seen. In some measurement cycles, there are enough NSR beats for a NSR classification even for the 24 PVC per minute setting of the PS420.

Figure 5.4 shows an easy case with 6 PVC per minute. Note how the new QRS detector, with a new FIR filter trained to specifically detect normal QRS complexes, has just skipped both PVC in the recording, while normal QRS complexes are marked by dotted vertical lines. Beats anchored at green QRS complexes were used for averaging, those anchored at red QRS complexes were not. The PVC are treated as extra data after the end of the T-wave of the previous QRS.

The algorithm is tolerant to limited pathological deviations. A first degree heart block, nodal rhythm or ST changes from -0.4 to $+0.2\,\mathrm{mV}$ are still classified as NSR, while larger ST changes are all classified as A-Fib, as are bigeminy and trigeminy (of NSR and PVC), second degree heart block, sinus arrhythmia, ventricular tachycardia and left bundle branch block.

More deviant pathological ECG are classified as neither NSR nor A-Fib. ST elevation of $0.8\,\mathrm{mV}$, ventricular fibrillation, asystole, right bundle branch block, asynchronous pacemakers and supraventricular or atrial tachycardia. This also applies to nodal PNC (premature nodal contracture), atrial PAC (premature atrial contraction) and R-on-T left PVC. Unsurprisingly, the same holds for a number of technical test signals (sine waves from 0.5 to 100 Hz, triangle waves from 2 to 2.5 Hz and 30 bpm 60 ms pulses, 2 Hz square wave), with the exception of a 60 bpm 60 ms pulse, which got classified as A-Fib by the algorithm.

Finally, there are two classes of simulated ECG where the analysis depends on whether the PS420 is used with AC power supply or rather with a battery. In the latter case, AV sequential pacemaker, non-capture or non-functional pacemaker rhythms are classified as A-Fib, while they are classified as neither NSR nor A-Fib in the former, lower signal quality case.

This is due to not finding enough beats suitable for averaging when on AC power. When on battery power, enough beats are found, but the averaged pacemaker ECG fails to meet NSR requirements. As the algorithm rates beats by similarity relative to their RMS, the low RMS of failed pacemaker stimulation corresponds to high requirements regarding the repetition accuracy to use them for averaging. In a non-synthetic ECG with more ambient noise, it is more likely that the algorithm will decide A-Fib for (high demand or failing) pacemaker rhythms.

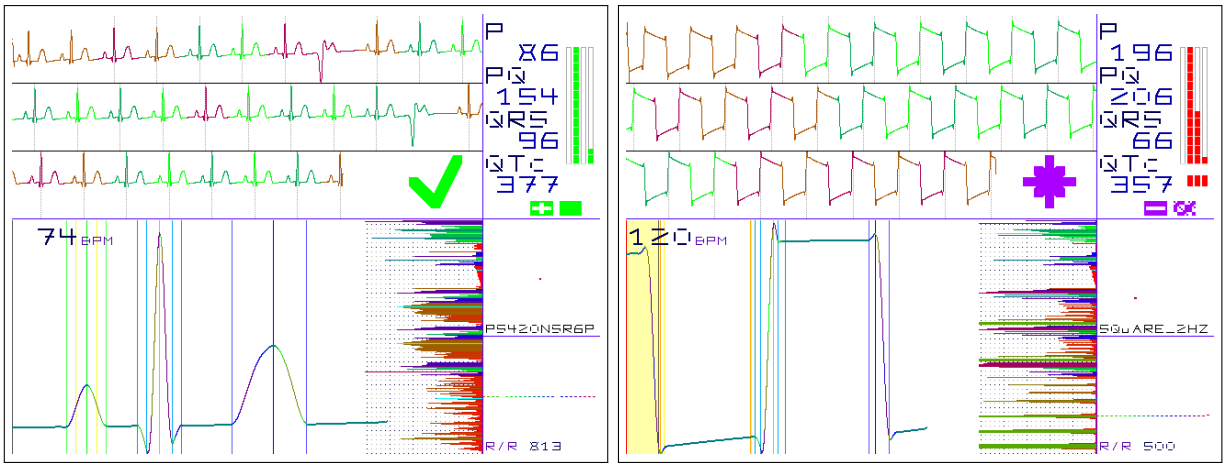


Figure 5.4: Left: PS420 simulation of NSR with on average 6 PVC per minute. Right: 2 Hz square wave from PS420, best compensation settings for averaged ECG

The opposite case is the class of simulated pathologies where the `ecghelper2` system produces an A-Fib classification only when the PS420 uses AC power. Those are occasional demand pacing, third degree heart block, several missed beats or frequent demand pacing.

In most cases, this may be related to signals containing longer stretches of NSR which, in particular when combined with the almost unrealistically high signal quality of the PS420 on battery power, can push the overall classification to NSR, or at least to neither NSR nor A-Fib.

In the case of third degree heart block, the RMS per beat drops (the P-waves end before the start of the QRS-anchored processing scope) which lets the algorithm refuse classification because the beats vary too much relative to their average RMS. The extra noise in the non-battery case lets the algorithm **accept** the averaging task, which then results in **not** finding a suitable P-wave within the processing scope, which in turn is one of the standard reasons for A-Fib classifications for `ecghelper2`.

Figure 5.4 shows the best filter compensation settings for averaging a 2 Hz square wave: TUNEPREQRSSLOPE and DIGITALHPF are both off and UNFILTERHPF value 216 gives best results for the device under test to compensate the analogue high-pass filter for the particular MAX30003 and capacitor tolerances of the device. The constant voltage periods of the input signal are almost flat again in the un-filtered average, while some ringing (in particular from the $N \times 50$ Hz FIR comb notch filter) remains at the edges. Note how the rising edge shows more ringing, as the algorithm treats it as QRS complex and excludes it from smoothing.

6 Conclusions

As intended, during the course of this thesis, an integrated embedded system with custom hardware has been developed, built and tested which can be used, among other ECG related tasks, to capture and classify the ECG of an untrained user to give an immediate indication of whether or not the user is likely to be experiencing atrial fibrillation at the moment of the measurement, using a quick (less than one minute) measurement paradigm with instant presentation of analysis results. The device also features a friendly user interface, including audio feedback, colour coded status lights and a large display. The battery-powered device is safe to use and has integrated stainless steel electrodes which can be used for an arbitrary number of measurement cycles.

Also as planned, the device can perform many measurement cycles and stay in standby for a long time with a single set of batteries. However, when using **rechargeable** NiMH batteries, those have to be charged more often than desired, because the power supply of the device is less efficient for the lower voltages provided by rechargeable batteries. From a verification point of view, the requirements from the thesis proposal have been achieved, regarding both hardware and software.

A proposed third generation device upgrades the power supply to use an even more modern integrated circuit with again better performance. Also, it uses power gating not only for the USB interface (already in the prototype built for this thesis) but also for the display, further reducing standby power consumption, while at the same time including additional sensors. The latter makes the device more versatile and can serve as an incentive to use the device more often, which will make the user more aware of when it is necessary to replace or recharge the batteries.

In addition to the stand-alone operation, the device also features Bluetooth connectivity, letting the users share their ECG recordings or ECG analysis outcomes with others or store them on their local devices such as computers or smartphones, while still keeping control over when to share data.

Using a subset of the recently published PTB-XL corpus [Wagner 2020a, Wagner 2020b], (offering more than 20 000 short ECG fragments annotated with their individual diagnoses), it was possible to let the `ecghelper2` system and algorithm classify short ECG as either normal sinus rhythm or atrial fibrillation at much higher accuracy than with the system designed in [Auer 2020].

The classification parameters generated with both ad-hoc and statistical methods for feature vector based supervised machine learning, form an embodiment of the concept of the difference between NSR and A-Fib which can then be applied as static knowledge embedded into the firmware of the `ecghelper2` device. The device does not have to process or access the training corpus to perform the classification task and both the amount of required user ECG data and computational resources and memory are low. The device can present the analysis and classification immediately after a short recording session of circa 30 seconds duration.

This makes the new system appropriate as a source of easily accessible information for users who want to keep track of the development of current health status regarding atrial fibrillation, provided that they use advice from medical professionals about the scope of their condition in parallel.

The device has yet to be tested with groups of actual patients and healthy controls (other than the author himself), pending a suitable environment and permission to do so.

As the development has taken place in an open source, open hardware context – to be published after completing the master's course – either the author or other scientific, maker community or commercial parties will be able to pursue validation tests, build the proposed third generation device, conceive new versions of the hardware or software, work further towards turning the device into a product, use the underlying algorithms for other purposes or port the created software to new or existing hardware platforms.

List of tables

3.1 MAX30003 signal quality data, excerpted from the datasheet 16
3.2 Comparison of low power TI buck/boost converters by min. input voltage, quiescent & max. current and efficiency at typical battery voltages & example currents 22
3.3 Expected power consumption in standby mode, while recording, processing or displaying ECG and analysis results, or in Bluetooth mode 27
3.4 ESP32 GPIO pin assignments for all generations of the ecghelper device . . . 38
4.1 Shape metrics extracted from the averaged ECG: All times are expressed relative to the R-wave peak, heights are amplitudes relative to wave baseline 41
4.2 Overview of the 256 byte ecghelper2 algorithm feature vector 50
4.3 All rules for NSR detection and eight best rules for A-Fib detection, based on automated analysis of feature data from the PTB-XL NSR and A-Fib subsets . . 51
4.4 Variance in the used PTB-XL corpora captured by the first N PCA dimensions 53
4.5 Effects of different reductions in feature space on LDA. Note that a stability spectrum was used at that time (September) instead of an instability spectrum 54
4.6 Interactive audio and RGB LED signals used by the ecghelper2 device 59
5.1 Power consumption measurements for the ecghelper2 prototype design 63
5.3 List of key tunable parameters of the ecghelper2 classification algorithm . . . 68
5.4 Performance statistics for the old proof of concept ecghelper system (rounded) 70
5.5 Performance statistics for different ecghelper2 versions applied to PTB-XL . . 71
5.6 Classification results for current algorithm and old corpora: Normal text: TUNEPREQRSSLOPE on (20 Nov.), *italics text:* off (14 Dec.) 72
5.7 Performance in leave-1-out cross-validation with PTB-XL corpora 74
A.1 Bill of Materials and price comparisons for the ecghelper2 device prototype . XXV
C.2 List of PTB-XL recordings classified as A-Fib in spite of NSR metadata . . . XLII

List of figures

1.1 Left: Original ecghelper hardware: Stacked off-the-shelf modules, OLED, NeoPixels, for USB power bank & wristband electrodes. Right: Test with patient simulator 1
3.1 Left: ECG electrode circuit with safety resistors, external electrode connector & test header. Right: ECG AFE circuit, 1.8 V supply & low-power 32768 Hz oscillator 17
3.2 Left: Power supply (without T1) with fuse & ideal diodes (battery/USB selection). Right: USB (without R1, R2) with boot control, ESD protection & power gating 21
3.3 Experiments with the new, larger display in September 2020: A Mandelbrot fractal and an earlier version of the ecghelper user interface. 24
3.4 Left: Audio circuit with piezo transducer. Right: Core ESP32-WROOM-32D module with built-in antenna, tiny RGB LED on either side, debug & extension headers 26
3.5 Layout of the ecghelper2 device, original, actual size 28
3.6 Layout of the ecghelper2 device, improved, actual size 29
3.7 Early test of ecghelper2 with external wristband electrodes 30
3.8 Plywood and acrylic glass test housing for the ecghelper2 31
3.9 Stainless steel sheet metal electrodes at the bottom side of the housing, dimensions and drill locations relative to the housing, photo of the actual prototype . . . 32
3.10 Bugfix: Connect RED1 and RED2 RGB LED signals, red wire routed below the board for aesthetic reasons and away from the Bluetooth antenna to avoid interference 33
3.11 TI Webench comparison of TPS63021 to TPS63802 at very low loads 35
3.12 Layout of a proposed ecghelper3 device, actual size 36
4.1 Schema of normal ECG with main wave and segment labels 40
4.2 Left: Heterogeneous Dataset of Arrhythmia corpus example 1/108 with mixed pathologies, which the algorithm classifies as neither NSR nor A-Fib. Right: ECG-ID corpus item 72/04, NSR with two-peaked P-wave 43
4.3 Self-calibration of ecghelper2, using the MAX30003 built-in source 44
4.4 While recording the ECG, a simple scope view and progress indicators are shown 45
4.5 Comparison of old (purple = compensated), new (orange = double compensation) and naive (blue = plain moving average) 50 Hz comb filters, plus 3 runner-up choices 47
4.6 Some statistics about the used feature vectors 49
4.7 Best features when using threshold rules for classification 52
4.8 Projection of the test sets using the most relevant PCA dimensions 53
4.9 Projection of the test sets using the best trained LDA dimensions 55
4.10 Left: Heterogeneous Dataset of Arrhythmia corpus example 1/125. Right: Atrial Fibrillation Database example 08455 (both classified as A-Fib) 57
4.11 Left: Sample IAF8 AFW (Intracardiac Atrial Fibrillation Database) bigeminal A-Fib rhythm. Right: ECG-ID item 86/2, classified as NSR in spite of low ECG quality 58
5.1 Left: Current over time for full measurement cycle with Bluetooth (scaled to 1 mV per 1 mA, 2.5 Hz low-pass). Right: Bluetooth connection setup current, qualitative 64
5.2 Left: Current measurement using a shunt, voltage divider and low-pass filter. Right: Current measurement using a low-pass supply filter 65
5.3 Left: Atrial fibrillation simulation by the PS420 device. Right: Atrial flutter simulation by the PS420 device 75
5.4 Left: PS420 simulation of NSR with on average 6 PVC per minute. Right: 2 Hz square wave from PS420, best compensation settings for averaged ECG 76
A.1 Display SPI clock and data at 10 MHz (left) and 40 MHz (right) XXIII
A.2 Left: Idle current measurement with 0.5 Ω shunt. Right: Sawtooth current pattern of active device XXIII
A.3 Active current measurement with uncalibrated shunt. Left: Test cycle with display, LED and sound. Right: Bluetooth connection setup XXIV
A.4 Circuit diagram of the ecghelper2 device, page 1 of 2 XXVI
A.5 Circuit diagram of the ecghelper2 device, page 2 of 2 XXVII

A.6 Circuit diagram of a proposed ecghelper3 device, page 1 of 2 XXVIII
A.7 Circuit diagram of a proposed ecghelper3 device, page 2 of 2 XXIX
B.1 Best features according to LDA classification XXXI
B.2 Relative feature weights for classification: Main LDA dimension XXXII
B.3 Optimised FIR comb filter multiples of 50 Hz, used in ecghelper2 XXXIII
B.4 Old FIR comb filter multiples of 50 Hz, used in ecghelper project XXXIII
B.5 Pei Tseng biquad notch filter 187.25 Hz XXXIV
B.6 Pei Tseng biquad notch filter 60 Hz XXXIV
B.7 FIR QRS detection filter based on corpus data, used in ecghelper2 XXXV
B.8 Old FIR QRS detection filter, used in ecghelper project XXXV
B.9 ECG shape smoothing: Effect of repeated short filters, QRS excluded XXXVI
B.10 Short FIR slope detection filter for shape analysis XXXVI
C.1 A-Fib in old corpus classified as NSR. Left: 04015, inverse T, rules 2:2, 1 noise rule, LDA: NSR. Right: 04048, intermittent A-fib, rules 1:0 NSR, LDA: NSR . . . XXXVII
C.2 A-Fib from old corpus classified as NSR. Left: 04126, LDA: NSR, overrules 2 A-Fib rules and A-Fib shape. Right: 05091, shallow T, long PR and P duration, 1 NSR rule, LDA: NSR XXXVIII
C.3 A-Fib from old corpus classified as NSR. Left: 05261, only intermittent A-Fib, P and T amplitude rules NSR, ST amplitude rule A-Fib, LDA NSR, good P shape. Right: iaf5 tva, flutter, LDA decides NSR, overrules A-Fib shape XXXVIII
C.4 A-Fib in the PTB-XL corpus classified as NSR 1/5 XXXIX
C.5 A-Fib in the PTB-XL corpus classified as NSR 2/5 XL
C.6 A-Fib in the PTB-XL corpus classified as NSR 3/5 XL
C.7 A-Fib in the PTB-XL corpus classified as NSR 4/5 XLI
C.8 A-Fib in the PTB-XL corpus classified as NSR 5/5 XLI
C.9 NSR in the PTB-XL corpus classified as A-Fib 1/6 XLIII
C.10 NSR in the PTB-XL corpus classified as A-Fib 2/6 XLIV
C.11 NSR in the PTB-XL corpus classified as A-Fib 3/6 XLV
C.12 NSR in the PTB-XL corpus classified as A-Fib 4/6 XLVI
C.13 NSR in the PTB-XL corpus classified as A-Fib 5/6 XLVII
C.14 NSR in the PTB-XL corpus classified as A-Fib 6/6 XLVIII

References

[AlMusallam 2019] M AlMusallam, A Soudani, *Embedded Solution for Atrial Fibrillation Detection Using Smart Wireless Body Sensors*, IEEE Sensors Journal, Vol. 19, No. 14 (2019) https://doi.org/10.1109/JSEN.2019.2906238

[Auer 2020] E Auer, *Embedded detection of atrial fibrillation and atrial flutter in single-channel ECG*, project report (2020) https://doi.org/10.5281/zenodo.4311642

[Duarte 2020] R Duarte, A Stainthorpe, J Greenhalgh et al, *Lead-I ECG for detecting atrial fibrillation in patients with an irregular pulse using single time point testing: a systematic review and economic evaluation*, Health technology assessment 24(3):1-164 (2020) https://doi.org/10.3310/hta24030

[Goldberger 2000] A Goldberger et al, *PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals.*, Circulation Online. 101 (23), pp. e215 – e220 (2000)

[Hartley 1942] RVL Hartley, *A More Symmetrical Fourier Analysis Applied to Transmission Problems*, Proceedings of the IRE. 30 (3), pp. 144 – 150 (1942) https://doi.org/10.1109/JRPROC.1942.234333

[Hau 2020] YW Hau, HW Lim, CW Lim, S Kasim, *P204 Automated detection of atrial fibrillation based on stationary wavelet transform and artificial neural network targeted for embedded system-on-chip technology*, European Heart Journal, Volume 41, Issue Supplement 1 (2020) https://doi.org/10.1093/ehjci/ehz872.075

[Jolliffe 2016] IT Jolliffe, J Cadima, *Principal component analysis: A review and recent developments* Philosophical Transactions of the Royal Society A, 374: 20150202 (2016) https://doi.org/10.1098/rsta.2015.0202

[Khamis 2016] H Khamis, R Weiss, Y Xie, C-W Chang, NH Lovell, SJ Redmond, *QRS detection algorithm for telehealth electrocardiogram recordings*, IEEE Transaction in Biomedical Engineering, vol. 63(7), p. 1377-1388 (2016) https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/QTG0EP

[Kramme 2011] R Kramme, KP Hoffmann, RS Pozos (eds.), *Springer Handbook of Medical Technology* (2011)

[Lahdenoja 2018] O Lahdenoja, T Humanen, Z Iftikhar, et al, *Atrial Fibrillation Detection via Accelerometer and Gyroscope of a Smartphone*, IEEE Journal of Biomedical and Health Informatics 22(1):108-118 (2018) https://doi.org/10.1109/jbhi.2017.2688473

[Lugovaya 2014] TS Lugovaya, *Biometric human identification based on electrocardiogram*, Master's thesis, Faculty of Computing Technologies and Informatics, Electrotechnical University LETI, Saint-Petersburg (2005) ECG data published 2014: https://physionet.org/content/ecgiddb/1.0.0/ (see also [Goldberger 2000])

[Medhi 2019] K Medhi, *Heterogeneous Dataset of Arrhythmia*, Mendeley Data, v1 (2019) https://dx.doi.org/10.17632/mmhw3vhf6w.1 (excerpted from MIT-BIH Arrhythmia PhysioNet database)

[Menche 2007] N Menche (ed.), *Biologie, Anatomie, Physiologie* (2007) (This book is a just slightly abridged variant of *Mensch, Körper, Krankheit*, but in a very convenient form factor, a good choice for medical engineering teaching)

[Moody 1983] GB Moody, RG Mark, *A new method for detecting atrial fibrillation using R-R intervals.* Computers in Cardiology. 10:227-230 (1983). http://ecg.mit.edu/george/publications/afib-cinc-1983.pdf

[Nabian 2017] M Nabian, A Nouhi, Y Yin, S Ostadabbas, *A Biosignal-Specific Processing Tool for Machine Learning and Pattern Recognition.* IEEE-NIH 2017 Special Topics Conference on Healthcare Innovations and Point-of-Care Technologies (HI-POCT 2017). https://web.northeastern.edu/ostadabbas/code/

[Paine 2019] S Paine, *The am atmospheric model* (2019) https://doi.org/10.5281/zenodo.640645

[Pan 1985] J Pan, WJ Tompkins, *A Real-Time QRS Detection Algorithm.* IEEE Transactions on Biomedical Engineering, Vol BME-32, No. 3 (1985) https://courses.cs.washington.edu/courses/cse474/18wi/labs/l8/QRSdetection.pdf (note errata pages)

[Pei 1996] SC Pei, CC Tseng *IIR Multiple Notch Filter Design Based on Allpass Filter* IEEE Tencon (1996) https://doi.org/10.1109/TENCON.1996.608814

[Perales 2020] CRL Perales, HGC van Spall, S Maeda et al., *Mobile health applications for the detection of atrial fibrillation: a systematic review*, EP Europace, euaa139 (2020) https://doi.org/10.1093/europace/euaa139

[Peterson 2003] J Peterson, D Kynor, *Intracardiac Atrial Fibrillation Database* Data: https://physionet.org/content/iafdb/1.0.0/ (2003) (see also [Goldberger 2000])

[Pizzuti 1985] G P Pizzuti, S Cifaldi, G Nolfe, *Digital Sampling Rate and ECG Analysis*, J Biomed. Eng., 1985 ; 7(3) : 247 – 250. doi:10.1016/0141-5425(85)90027-5 https://pubmed.ncbi.nlm.nih.gov/4033100/

[Reeves 1990] AA Reeves, *Optimized Fast Hartley Transform for the MC68000 with Applications in Image Processing*, Master's thesis (1990) https://imagej.nih.gov/nih-image/download/nih-image_spin-offs/imageFFT/docs/thesis.pdf

[Rincón 2012] F Rincón, PR Grassi, N Khaled, D Atienza, D Sciuto, *Automated real-time atrial fibrillation detection on a wearable wireless sensor platform*, 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (2012) https://doi.org/10.1109/EMBC.2012.6346465

[Sedghamiz 2018] Sedghamiz, *BioSigKit: A Matlab Toolbox and Interface for Analysis of BioSignals.* Journal of Open Source Software, 3(30), 671, (2018) https://doi.org/10.21105/joss.00671 Sources: https://github.com/hooman650/BioSigKit

[Smith 1998] S W Smith, *The Scientist and Engineer's Guide to Digital Signal Processing* (1998) http://www.DSPguide.com/

[Wagner 2020a] P Wagner, N Strodthoff, R Bousseljot, W Samek, T Schaeffter, *PTB-XL, a large publicly available electrocardiography dataset* https://doi.org/10.13026/x4td-x982 (see also [Goldberger 2000])

[Wagner 2020b] P Wagner, N Strodthoff, R Bousseljot, D Kreiseler, FI Lunze, W Samek, T Schaeffter, *PTB-XL: A Large Publicly Available ECG Dataset. Scientific Data* https://doi.org/10.1038/s41597-020-0495-6

Websites mentioned in the text have been checked for validity in December 2020.

List of websites mentioned

This list is alphabetically sorted by URL (Uniform Resource Locator, web address), excluding the protocol (for example 'https://') and 'www.' for the purpose of sorting. All links have been checked for validity December 23, 2020. Websites already mentioned in the references themselves are not listed again here. Additional comments (such as date or version information) provided in parentheses. All trademarks belong to their respective owners. They have not been marked explicitly in this list – refer to the listed websites instead.

NeoPixels: Adafruit Industries, Unique & fun DIY electronics and kits
`https://www.adafruit.com/category/168`

AISLER - Powerful Prototyping made in Germany (PCB & stencils)
`https://www.aisler.net/`

AliveCor (Tiny, wireless ECG for use with Smartphones etc.)
`https://www.alivecor.com/`

Arduino - Home
`https://www.arduino.cc/`

Audacity – Free, open source, cross-platform audio software for multi-track recording and editing
`https://www.audacityteam.org/`

Humidity Sensor BME280 – Bosch Sensortec
`https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/`

Absolute Orientation Sensor BMX160 – Bosch Sensortec
`https://www.bosch-sensortec.com/products/motion-sensors/absolute-orientation-sensors/absolute-orientation-sensors-bmx160.html`

Press release distribution, EDGAR filing, XBRL, refulatory filings – Business Wire
`https://www.BusinessWire.com/`

Fitbit Receives Regulatory Clearance in Both the United States and Europe for ECG App to Identify Atrial Fibrillation (AFib) – Business Wire
`https://www.businesswire.com/news/home/20200914005215/en/Fitbit-Receives-Regulatory-Clearance-in-Both-the-United-States-and-Europe-for-ECG-App-to Identify-Atrial-Fibrillation-AFib`

home (Bytefish – Philipp Wagner)
`https://www.bytefish.de/`

Principal Component Analysis and Linear Discriminant Analysis with GNU Octave
`https://www.bytefish.de/blog/pca_lda_with_gnu_octave.html`

Creative Commons – Attribution-NonCommercial-NoDerivatives 4.0 International – CC BY-NC-ND 4.0
`http://creativecommons.org/licenses/by-nc-nd/4.0/`

Creative Commons – Attribution-ShareAlike 4.0 International – CC BY-SA 4.0
`https://creativecommons.org/licenses/by-sa/4.0/`

HeartyPatch – Crowd Supply
`https://www.crowdsupply.com/protocentral/heartypatch`

MAX30003 – Ultra-Low Power, Single-Channel Integrated Biopotential (ECG, R-to-R Detection) AFE – Maxim Integrated
`https://datasheets.maximintegrated.com/en/ds/MAX30003.pdf`

Diodes Incorporated DDC(XXXX)U Dual NPN pre-biased transistors in SOT363 (DDC114TU)
https://www.diodes.com/assets/Datasheets/ds30345.pdf

PAM8904E (18VPP Output Piezo Sounder Driver, Diode Inc.)
https://www.diodes.com/assets/Datasheets/PAM8904E.pdf

PAM8904 (not recommended for new design – use PAM8904E)
https://www.diodes.com/assets/Datasheets/products_inactive_data/PAM8904.pdf

Diagnostic odds ratio – Wikipedia (2020-12-05)
https://en.wikipedia.org/wiki/Diagnostic_odds_ratio

EAGLE (program) – Wikipedia (2020-11-16, section License model)
https://en.wikipedia.org/wiki/EAGLE_(program)#License_model

F-score – Wikipedia (2020-11-06)
https://en.wikipedia.org/wiki/F-score

Linear discriminant analysis – Wikipedia (2020-11-29)
https://en.wikipedia.org/wiki/Linear_discriminant_analysis

Matthews correlation coefficient – Wikipedia (2020-12-13)
https://en.wikipedia.org/wiki/Matthews_correlation_coefficient

Principal component analysis – Wikipedia (2020-12-16)
https://en.wikipedia.org/wiki/Principal_component_analysis

ESP32 Wi-Fi & Bluetooth Modules – Espressif
https://www.espressif.com/en/products/modules/esp32

ESP32 Series Datasheet (Version 3.4 2020)
https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

ESP32 Technical Reference Manual (Version 4.3 2020)
https://www.espressif.com/sites/default/files/documentation/
esp32_technical_reference_manual_en.pdf

Onlineshop für Embedded Systems, SBC und Mikrocontroller – EXP Tech
https://www.exp-tech.de/

Electrical Safety Standards and Testing (Fluke Biomedical)
https://www.flukebiomedical.com/blog/electrical-safety-standards-basic-testing

TGA specs, from the 2D graphics format web collection
http://www.gamers.org/dEngine/quake3/TGA.txt

GitHub: adafruit/Adafruit-HUZZAH32-ESP32-Feather-PCB: PCB files for the Adafruit HUZZAH32 ESP32 Feather
https://github.com/adafruit/Adafruit-HUZZAH32-ESP32-Feather-PCB

GitHub: MartyMacGyver/ESP32-Digital-RGB-LED-Drivers: ESP32 Digital RGB(W) LED Drivers
https://github.com/MartyMacGyver/ESP32-Digital-RGB-LED-Drivers

GNU Octave (Scientific Programming Language, free software, drop-in compatible with many Matlab scripts)
https://www.gnu.org/software/octave/

Welcome to HeartyPatch (ProtoCentral)
https://heartypatch.protocentral.com/

Kingbright APHF1608**L**SEEQBDZGKC 1.6 × 0.8 mm Full-Color Surface Mount LED
http://www.kingbrightusa.com/images/catalog/SPEC/APHF1608LSEEQBDZGKC.pdf

Kingbright APHF1608SEEQBDZGKC 1.6 × 0.8 mm Full-Color Surface Mount LED
http://www.kingbrightusa.com/images/catalog/SPEC/APHF1608SEEQBDZGKC.pdf

Technical Reference Manual (ZSoft Corporation 1988, section Image File (.PCX) Format)
http://www.martinreddy.net/gfx/2d/PCX.txt

MAX40203 Ultra-Tiny Nanopower, 1A Ideal Diodes with Ultra-Low-Voltage Drop – Maxim Integrated
https://www.maximintegrated.com/en/products/analog/analog-switches-multiplexers/MAX40203.html

MAX1726 12V, Ultra-Low-IQ, Low-Dropout Linear Regulators – Maxim Integrated
https://www.maximintegrated.com/en/products/power/linear-regulators/MAX1726.html

Distributor für elektronische Bauelemente – Mouser Electronics Deutschland
https://www.mouser.de/

Octave Forge – Packages (Community packages)
https://octave.sourceforge.io/packages.php

List of Functions for the 'signal' package (Octave Forge)
https://octave.sourceforge.io/signal/overview.html

L10: Linear discriminant analysis (Ricardo Gutierrez-Osuna)
https://people.engr.tamu.edu/rgutier/lectures/pr/pr_l10.pdf

L9: Principal component analysis (Ricardo Gutierrez-Osuna)
https://people.engr.tamu.edu/rgutier/lectures/pr/pr_l9.pdf

Declaration of Authorship (Universität Düsseldorf: Philosophische Fakultät)
https://www.phil-fak.uni-duesseldorf.de/uploads/media/Declaration_of_Authorship.pdf

How Much Current Do WS2812 / NeoPixel LEDs Really Use?
https://www.pjrc.com/how-much-current-do-ws2812-neopixel-leds-really-use/

Teaching PSI Lab (perception sensing instrumentation lab – Texas A&M University)
https://psi.engr.tamu.edu/courses/

SMT-1141-T-5-R – Buy Components for Transducers & Order SMT-1141-T-5-R at PUI Audio
https://www.puiaudio.com/products/SMT-1141-T-5-R

(JSON for Arduino IDE to install Espressif Systems ESP32 support)
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

Elektronik und Technik bei reichelt elektronik günstig bestellen
https://www.reichelt.de/

Richtek RT9073 – 1 μA I_Q, 250 mA Low-Dropout Linear Regulator
https://www.richtek.com/assets/product_file/RT9073/DS9073-06.pdf

A smartphone application can help in screening for atrial fibrillation – ScienceDaily
https://www.sciencedaily.com/releases/2018/08/180825081735.htm

Novel necklace detects abnormal heart rhythm – ScienceDaily
https://www.sciencedaily.com/releases/2020/05/200505072158.htm

CP2102N-GQFN20 USBXpress USB Bridge – Silicon Labs
https://www.silabs.com/interface/usb-bridges/usbxpress/device.cp2102n-gqfn20

CP2104 Classic USB to UART Bridge – Silicon Labs (Not Recommended for New Designs)
https://www.silabs.com/interface/usb-bridges/classic/device.cp2104

Leakage Current in Medical Devices – The Talema Group
https://talema.com/de/leakage-current-in-medical-devices/

Texas Instruments TLV522 Dual Nanopower, 500 nA, RRIO CMOS Operational Amplifier
https://www.ti.com/lit/ds/symlink/tlv522.pdf

24-bit, 1-ch, Low-Power Analog Front END (AFE) for ECG Applications
https://www.ti.com/product/ADS1291?qgpn=ads1291

1.5 V to 5.5 V, 1.5 A, 0.5 μA IQ ideal diode with Integrated FET (Texas Instruments LM66100)
https://www.ti.com/product/LM66100

Low-Power Triple Buffer Gate (Texas Instruments SN74AUP3G34)
http://www.ti.com/product/sn74aup3g34?qgpn=sn74aup3g34

TPS63021 High Efficiency Single Inductor Buck-Boost Converter with 4A Switch (Texas Instruments)
https://www.ti.com/product/TPS63021

TPS63802 2 A, high-efficient, 11 μA quiescent current buck-boost converter in QFN/DFN package (Texas Instruments)
https://www.ti.com/product/TPS63802

Waveshare Electronics
https://www.waveshare.com/

2inch LCD Display Module, IPS Screen, 240 × 320 Resolution, SPI Interface (Waveshare)
https://www.waveshare.com/2inch-lcd-module.htm

Sitronix ST7789VW Datasheet Version 1.0 2017/09
https://www.waveshare.com/w/upload/a/ae/ST7789_Datasheet.pdf

Waveshare 2inch LCD Module (Schematic and PCB Prints)
https://www.waveshare.com/w/upload/e/ee/2inch_LCD_Module_SchDoc.pdf

webench Power Designer (Texas Instruments)
https://webench.ti.com/power-designer/

Appendix

A Appendix: hardware measurements, bill of materials, schematics

A.1 Signal quality and current measurements

Using a Hameg HMO1022 digital oscilloscope, LCD bus signal quality has been checked on the prototype. As figure A.1 shows, quality is limited at 40 MHz, but sufficient. The display controller would allow up to 60 MHz. The software now uses 10 MHz, which improves signal quality and reduces energy consumption. EMI and cross-talk affecting the ECG are also expected to be better at 10 MHz. Note that the LCD controller, unlike the OLED one, has no built-in line or rectangle drawing: The bus needs enough bandwidth for all updated pixels, which is the case at 10 MHz.

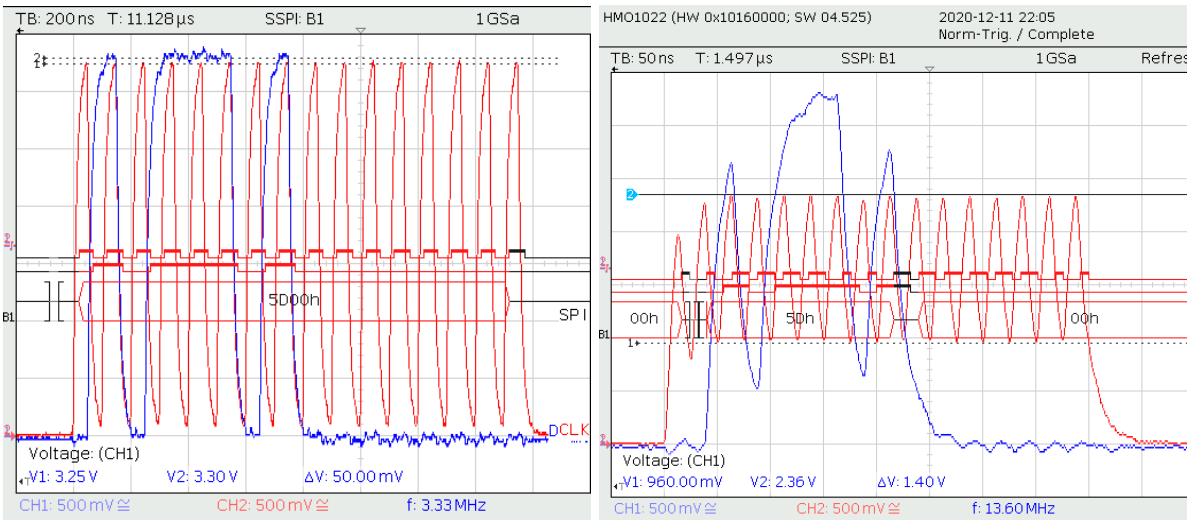


Figure A.1: Display SPI clock and data at 10 MHz (left) and 40 MHz (right)

For the MAX30003, a SPI clock speed of 1 MHz is more than enough to transfer 500 tagged samples per second and poll status registers, although up to 12 MHz are supported and 8 MHz have been tested. The lower clock frequency keeps interference and power consumption low.

Checking with a shunt and a Yokogawa oscilloscope, the power supply does single switching cycles ($\approx$ 25 to 30 Hz) while the device is in standby, to recharge the 3.3 V output capacitors as needed. A multimeter shows $90 - 100\,\mu$A average current at 3.1 V combined battery voltage in standby. The average standby current at somewhat lower battery voltages was still below $120\,\mu$A.

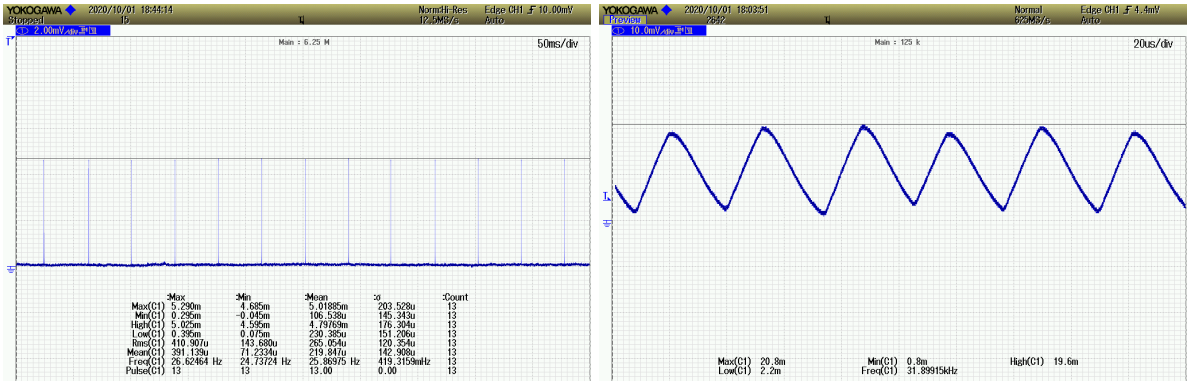


Figure A.2: Left: Idle current measurement with 0.5 Ω shunt, scope bandwidth set to 8 kHz. Right: Sawtooth current pattern of active device, scope bandwidth set to 5 MHz

Figure A.2 shows those single cycles, compared to a sawtooth current pattern when the `ecghelper2` device is in a measurement cycle. Note the low-pass filter (8 kHz scope bandwidth) applied to make the standby current signal easier to see above the noise floor.

Using the oscilloscope at low bandwidth and for example 500 ms per div is a good method to get an impression of how current draw changes over time in a typical measurement cycle. In a test cycle, the device boots from standby state, then starts the display and tests the RGB LED and later the audio feature, returning to standby at the end. The ECG activity itself only is a small part of the power budget, while choice of LED signals does make a significant difference.

The final figure figure A.3 shows power consumption during a test cycle compared to Bluetooth connection setup: Starting the wireless interface triggers a current surge of less than 0.5 s followed by some short ping style peaks. While streaming ECG data to a peer, such transmit peaks occur in rapid succession.

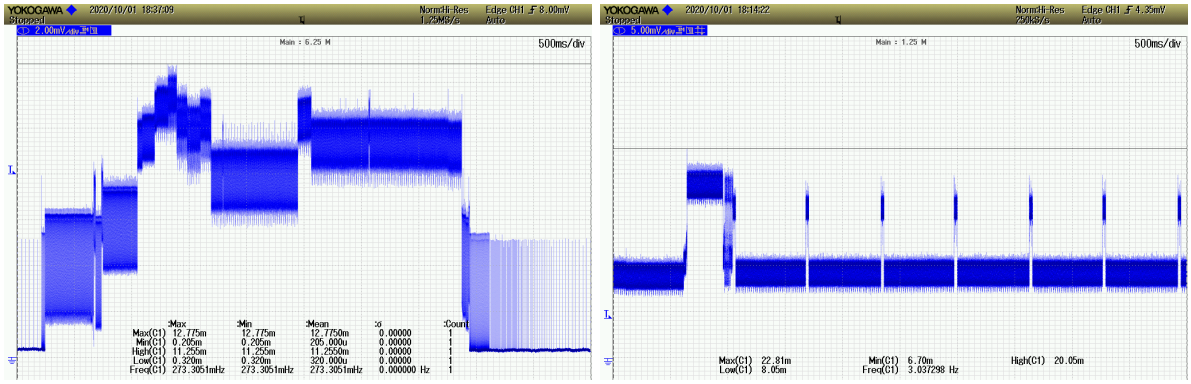


Figure A.3: Active current measurement with uncalibrated shunt. Left: Test cycle with display, LED and sound. Right: Bluetooth connection setup (scope bandwidth 8 kHz for both measurements)

A.2 Prototype bill of materials

Table A.1 presents an extended bill of materials (BoM) for the second generation prototype, including prices and availability at Mouser, Digikey and Farnell (August 2020). For many parts, Farnell had too high minimum order quantities, or was not able to provide special components. The displays had to be ordered directly from Waveshare, as explained in 3.8. Mechanical parts were either available at the university lab or were acquired from local stores at low cost, most of them not explicitly listed in this BOM.

A.3 Prototype circuit diagram

The schematics of the `ecghelper2` prototype described and tested in this thesis are shown in figures A.4 and A.5. See section 3 for a discussion of the individual modules: Power supply, USB interface, ESP32 dual-core wireless controller, piezo sound, batteries and power-ORing, RGB LED, MAX30003 ECG AFE, test and extension headers, LCD display and ECG electrode interface.

A.4 Third generation circuit diagram

Section 3.15 proposes an improved third generation device, which has more features, uses only two thirds of the board area and significantly reduces standby power power consumption by more than one third according to estimates. The schematics are shown in figures A.6 and A.7. The proposed device adds environmental and 9-axis sensors, as well as power gating for the LCD display. In addition, it upgrades to an even more efficient, even lower quiescent current, smaller power supply.

Qty	Value	Brand and Type (Mouser, Digikey, both)	Digikey (incl. VAT)	Mouser	Farnell	Device	Package	Parts	Description BOM item for ECGHELPER - 24.08.2020
6	100nF	Yageo CC0603KRX7R9BB104	0,09 (10 a 0,06)	0,09 (10 a 0,04)	10 a 0,08	C0603-1608	C0603K	C1, C5, C13, C22, C23, C31	Capacitor 50V X7R
3	10k	Panasonic ERJ-P06J103V	0,10 (10 a 0,10)	0,11 (10 a 0,10)	10 a 0,14	R0805-2012	R0805	R1, R2, R3	Resistor
2	10pF	TDK CGA3E2C0G1H100D080AA	0,17 (10 a 0,12)	0,16 (10 a 0,11)	10 a 0,18	C0603-1608	C0603K	C18, C19	Capacitor 50V C0G
10	10uF	Murata GCM21BR71A106KE22K	0,46 (10 a 0,32)	0,42 (10 a 0,29)	5 a 0,37 alternative	C0805-2012	C0805K	C3, C6, C8, C10, C20, C24, C27, C28, C29, C32	Capacitor 10V X7R
2	1nF	Murata GRM1885C1H102JA01D	0,09 (10 a 0,06)	0,09 (10 a 0,05)	10 a 0,08	C0603-1608	C0603K	C21, C26	Capacitor 50V C0G
5	1uF	Kemet C0603C105K4RACAUTO	0,15 (10 a 0,11)	0,13 (10 a 0,05)	10 a 0,14	C0603-1608	C0603K	C9, C14, C15, C25, C30	Capacitor 16V X7R
2	2.2uF	Kemet C0805C225K4RACAUTO	0,50 (10 a 0,35)	0,41 (10 a 0,21)	5 a 0,40	C0805-2012	C0805K	C11, C16	Capacitor 16V X7R
2	220R	Panasonic ERJ-P06J221V	0,10 (10 a 0,10)	0,11 (10 a 0,10)	reel	R0805-2012	R0805	R8, R11	Resistor
3	22uF	Kemet C1206C226M8PACTU	0,84 (10 a 0,61)	0,72 (10 a 0,42)	5 a 0,75	C1206-3216	C1206K	C2, C4, C7	Capacitor 10V X5R
1	24k	Panasonic ERJ-P06J243V	0,10 (10 a 0,10)	0,09 (10 a 0,09)	n/a	R0805-2012	R0805	R5	Resistor, surge proof
2	390R	Panasonic ERJ-P06J391V	0,10 (10 a 0,10)	0,11 (10 a 0,10)	reel	R0805-2012	R0805	R8, R14	Resistor
2	4.7uF	Murata GCM21BR71C475KA73L	0,52 (10 a 0,36) 0,45+V	0,28 (10 a 0,20)	5 a 0,35	C0805-2012	C0805K	C12, C17	Capacitor 16V X7R
1	47k	Panasonic ERJ-P06J473V	0,10 (10 a 0,10)	0,11 (10 a 0,10)	reel	R0805-2012	R0805	R4	Resistor, surge proof
2	49.9k	Vishay CRCW080549K9FKEAHP	0,23 (10 a 0,20)	0,22 (10 a 0,19)	10 a 0,09 alternative	R0805-2012	R0805	R11, R12	Resistor, surge proof
1	499k	Vishay RCS0805499KFKEA	0,24 (10 a 0,20)	0,21 (10 a 0,19)	n/a	R0805-2012	R0805	R13	Resistor, surge proof
2	68R	Vishay RCS080568R0FKEA	0,23 (10 a 0,20)	0,21 (10 a 0,19)	10 a 0,18	R0805-2012	R0805	R9, R12	Resistor
2	71308-5404	**Molex 15-91-0040**	0,42 (10 a 0,38) 0,36+V	0,35 (10 a 0,33)	0,37 (10 a 0,35)	71308-5404	71308-5404	J2, J4	Pin strip 2x2 71308-5404 basic RM2.54 SMD **(15-91-0040 or 71308-5404)**
2	71308-5408	**Molex 15-91-2080**	0,77 (10 a 0,72) 0,66+V	0,64 (10 a 0,60)	n/a	71308-5408	71308-5408	J1, J3	Pin strip 2x4 71308-5408 basic RM2.54 SMD **(15-91-2080 or 71308-5408)**
2	APHF1608	**Kingbright APHF1608SEEQBDZGKC or, preferred, APHF1608LSEEQBDZGKC**	0,75 (10 a 0,58) LSEEQ	0,68 (10 a 0,53) SEEQ now, 0,75 (10 a 0,58) LSEEQ	n/a	APHF1608	RGB1608	LED1, LED2	**Kingbright RGB LED APHF1608LSEEQBDZGKC 4-15/20-70/4-10 mcd at 2 mA, 1.8-2.1/2.65-3.1/2.65-3.1 Volt at 2 mA, max 30/20/20 mA: ...1608SEE... binned for 20 mA, similar to ...1608LSEE… at 2 mA**
1	BLM18KG601SN1D	Murata BLM18KG601SN1D	0,09 (10 a 0,08)	0,09 (10 a 0,04)	10 a 0,49	BLM18KG601SN1D	C0603K	FB1	BLM18KG601SN1D Ferrite Bead 1.3 A 600 Ohms 0.15 Ohms 0603 (1608)
1	C1Q1	Bel Fuse C1Q 1	0,30 (10 a 0,30) 0,26+V	0,25 (10 a 0,24)	n/a	C1Q1	R1206	F1	Belfuse SMD Fuse 1A 1206 (3216 metric)
1	CAY16-101J4LF	Bourns CAY16-101J4LF	0,09 (10 a 0,07)	0,09 (10 a 0,02)	10 a 0,02	CAY16-101J4LF	R-ISOLATED-4	R6	4 x 100 Ohm 62.5 mW 50 V
2	CAY16-220J4LF	**Bourns CAY16-22R0F4LF (Mouser) / CAY16-220J4LF (Digikey) any available**	0,09 (10 a 0,07) (other type 7.9.)	0,12 (10 a 0,06) (other type 2021…)	10 a 0,05 (type 220J)	CAY16-220J4LF	R-ISOLATED-4	R7, R17	**4 x 22 Ohm 62.5 mW 50 V (CAY16-22R0... 1% / CAY16-220... 5%)**
1	**CP2104**	Silicon Labs CP2104-F03-GMR	1,62 (1,40 + VAT)	1,40 (suggests CP2102 for N.D.)	1,92 (10 a 1,72)	CP2104	QFN24-4X4-050	IC2	CP2104 USB UART Serial Bridge QFN24 4x4 0.5 (GMR reel, GM tube) (newer CP2102N-QFN24 would be less optimal)
1	DDC114TU	Diodes DDC114TU-7-F	0,38 (10 a 0,29) 0,33+V	0,33 (10 a 0,21)	n/a	DDC114TU	SOT363	T1	Dual pre-biased NPN transistors 10k Rbase, 100mA, 50V, hFE 250
1	IP4234CZ6	Nexperia IP4234CZ6,125	0,42 (10 a 0,31) 0,36+V	0,36 (10 a 0,27)	5 a 0,37	IP4234CZ6	SOT457	TVS1	IP4234CZ6 Dual ESD Protection Network
2	KEYSTONE2466	Keystone 2466	1,15 (10 a 1,06) 0,99+V	1,10 (10 a 0,89)	2,27 (50 a …)	KEYSTONE2466	KEYSTONE2466	BAT1, BAT2	Keystone 2466 AAA battery holder (2466RB extra ribbon not necessary)
1	**KX3211A0032.768000**	Diodes KX3211A0032.768000	2,47 (10 a 2,41) 2,13+V	2,13 (10 a 1,98)	n/a	KX3211A0032.768000	OSC3225B	OSC1	KX321 series 3.3V low power 32768 Hz crystal oscillator
1	**LCD2INCH**	**n/a – Waveshare.com 2inch IPS LCD**	**n/a**	**n/a**	**n/a**	**LCD2INCH**	**LCD2INCH**	**LCD1**	**Waveshare.com 240×320, General 2inch IPS LCD Display Module – only available from Waveshare!**
1	LL3301NF065QG	LL3301NF065QG	0,68 (10 a 0,65) 0,59+V	0,44 (10 a 0,41)	n/a (US only)	LL3301NF065QG	LL3301NF065QG	SW1	SMD Pushbutton switch LL3301NF065QG 6 x 6 mm, 50 mA, 12 V DC
1	**MAX1726EUK18+T**	Maxim MAX1726EUK18+T	2,09 (10 a 1,88) 1,80+V	1,71 (10 a 1,61)	n/a (maybe Richtek?)	MAX1725/MAX1726…	21-0057F_MXM-M	PMIC3	MAX1726 LDO, fixed 1.8V version
1	**MAX30003CTI**	Maxim MAX30003CTI+ (TQFN28)	8,08 (10 a 7,30) 6,97+V	6,94 (10 a 6,26)	7,57 (10 a 6,83)	MAX30003CTI…	21-0140L_T2855-8_MXM-M	IC3	MAX30003 ECG AFE, TQFN28 5x5 version
1	MAX40203AUK	Maxim MAX40203AUK+T	0,81 (10 a 0,73) 0,70+V	0,70 (10 a 0,63)	0,66 (10 a 0,64)	MAX40203AUK+T…	21-0057F_MXM-M	PMIC2	MAX40230 ideal diode 1 Ampere (nanopower, over temp protection)
1	MOLEX473460001	Molex 47346-0001	1,03 (10 a 0,91) 0,89+V	0,81 (10 a 0,64)	0,69 (50 a 0,60)	MOLEX473460001	MOLEX473460001	X2	MOLEX 47346-0001 Micro USB B receptable (with flange, -1001 without)
1	PAM8904E	Diodes PAM8904EJPR	0,81 (10 a 0,72) 0,70+V	0,70 (10 a 0,58)	0,97 (JPR type)	PAM8904E	QFN12	IC1	PAM8904E Piezo buzzer driver, larger 3x3 QFN12 type A 0.5 package
1	RB168MM-40TF	Rohm RB168MM-40TFTR	0,39 (10 a 0,32) 0,34+V	0,34 (10 a 0,25)	5 a 0,31	RB168MM-40TF	SOD123	D3	RB168MM-40TF 40 V 1 A 0.6 Vf Schottky Barrier Rectifier Diode
1	**SJ1-3535NS**	CUI SJ1-3535NS	1,40 (10 a 1,23) 1,21+V	1,20 (10 a 0,95)	n/a	SJ1-3535NS	SJ1-3535NS	X3	MIC/HEADPHONE AUDIO JACK (shielded and 2x switched version)
1	**SMT1141T5R**	PUI SMT-1141-T-5-R	2,18 (10 a 1,80) 1,88+V	1,79 (10 a 1,28)	n/a	SMT1141T5R	SMT1141T5R	SPK1	PUI Audio SMT1141T5R Piezo buzzer transducer 1-25Vpp SMD
1	SRP5030CA-1R5M	Bourns SRP5030CA-1R5M	1,72 (10 a 1,60) 1,48+V	1,35 (10 a 1,25)	5 a 1,39	SRP5030CA-1R5M	SRP5030CA	L1	Bourns Shielded Power Inductor 1.5µH 10 mOhm I rms 8A I sat 14A SMD 5.5x5.3x2.9mm
2	LM66100	TI LM66100DCK (-R or -T)	0,36 (10 a 0,29) 0,31+V	0,35 (10 a 0,26)	n/a	LM66100	SC70-6	D1, D2	LM66100 ideal diode 1.5A, 1.5-5.5V, 100mOhm, 150nA, polarity protection
1	**TPS63021**	TI TPS63021DSJ (-R or -T)	2,62 (10 a 2,35) 2,26+V	2,42 (10 a 2,05)	3,04 (10 a 2,30)	TPS63021	TPS63021	PMIC1	TPS63021 High Efficiency Single Inductor Buck Boost converter with 4A Switches 1.8-5.5 to 3.3V
1	**WROOM32**	Espressif M113DH2800PH3Q0	4,43 (3,82+VAT)	3,81 (no discount)	n/a	WROOM32	WROOM32	X1	ESP32-WROOM-32D (16MB) module
1	**SOCKET STRIP 1x8**	Samtec SSQ-108-01-G-S	1,47 (10 a 1,34) 1,27+V	1,31 (10 a 1,14)	1,20 (-G-S type)	n/a	n/a	n/a	1x8 2.54mm socket strip (for LCD stacking 8.5mm)
1	**TERMINAL STRIP 1x8**	**Samtec SSQ-108-03-T-S (Mouser) or -F-S (Digikey) any available SSQ-108-03-?-S**	1,28 (-F-S type) 1,10+V	0,84 (-T-S type) 1,09 (-F-S type)	1,09 (-F-S type)	n/a	n/a	n/a	**1x8 2.54mm socket and pin terminal strip (for LCD stacking) (TSW-108-07-G-S needs 2.5mm extra below board)**
4	**SPACER**	**n/a**	**n/a**	**n/a**	**n/a**	**n/a**	**n/a**	**n/a**	**Spacer / standoff / screw for LCD stacking, len. 10-15mm, max dia. 6mm, with insulators 6.35mm**

Table A.1: Bill of Materials and price comparisons for the ecghelper2 device prototype

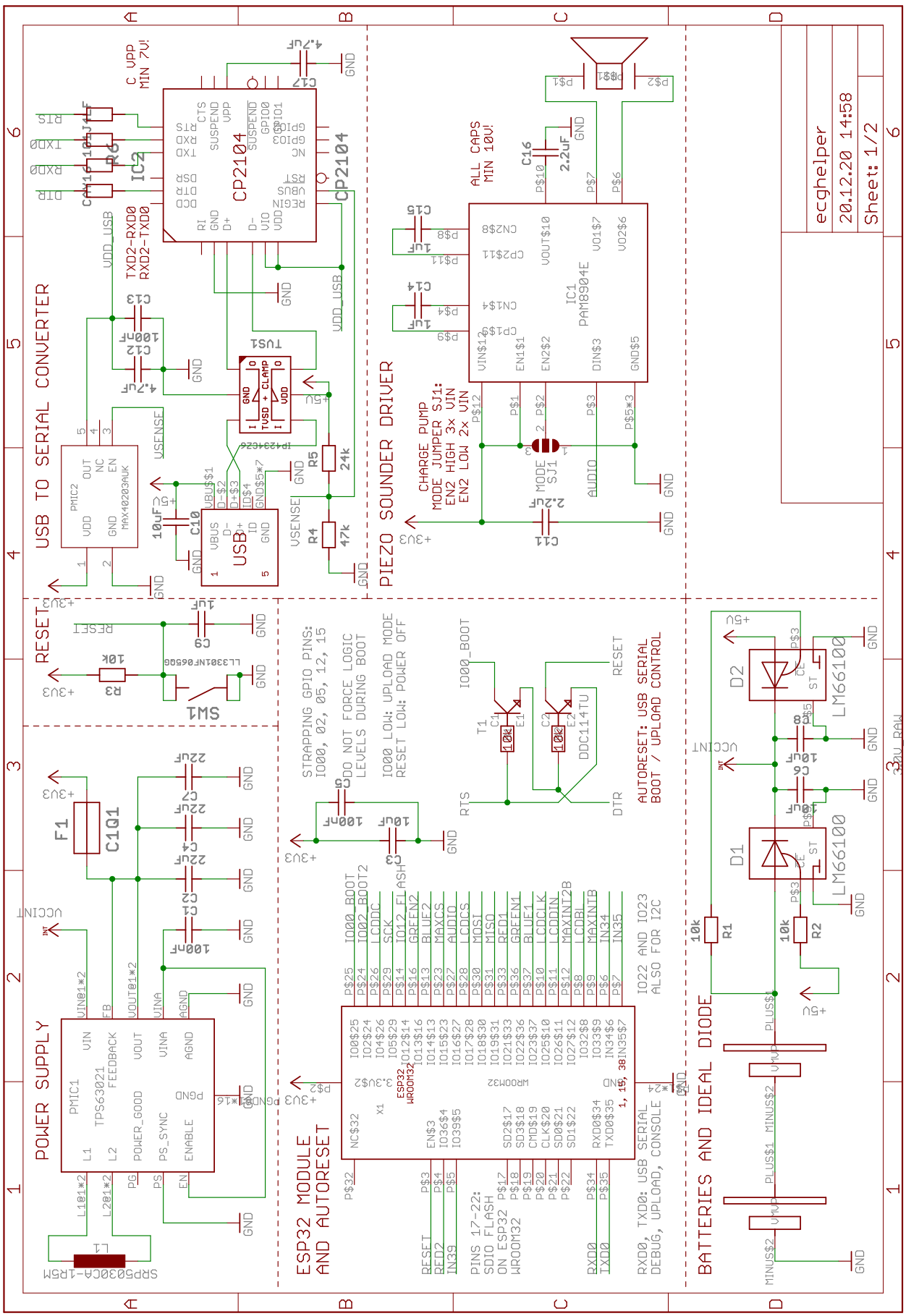


Figure A.4: Circuit diagram of the ecghelper2 device, page 1 of 2

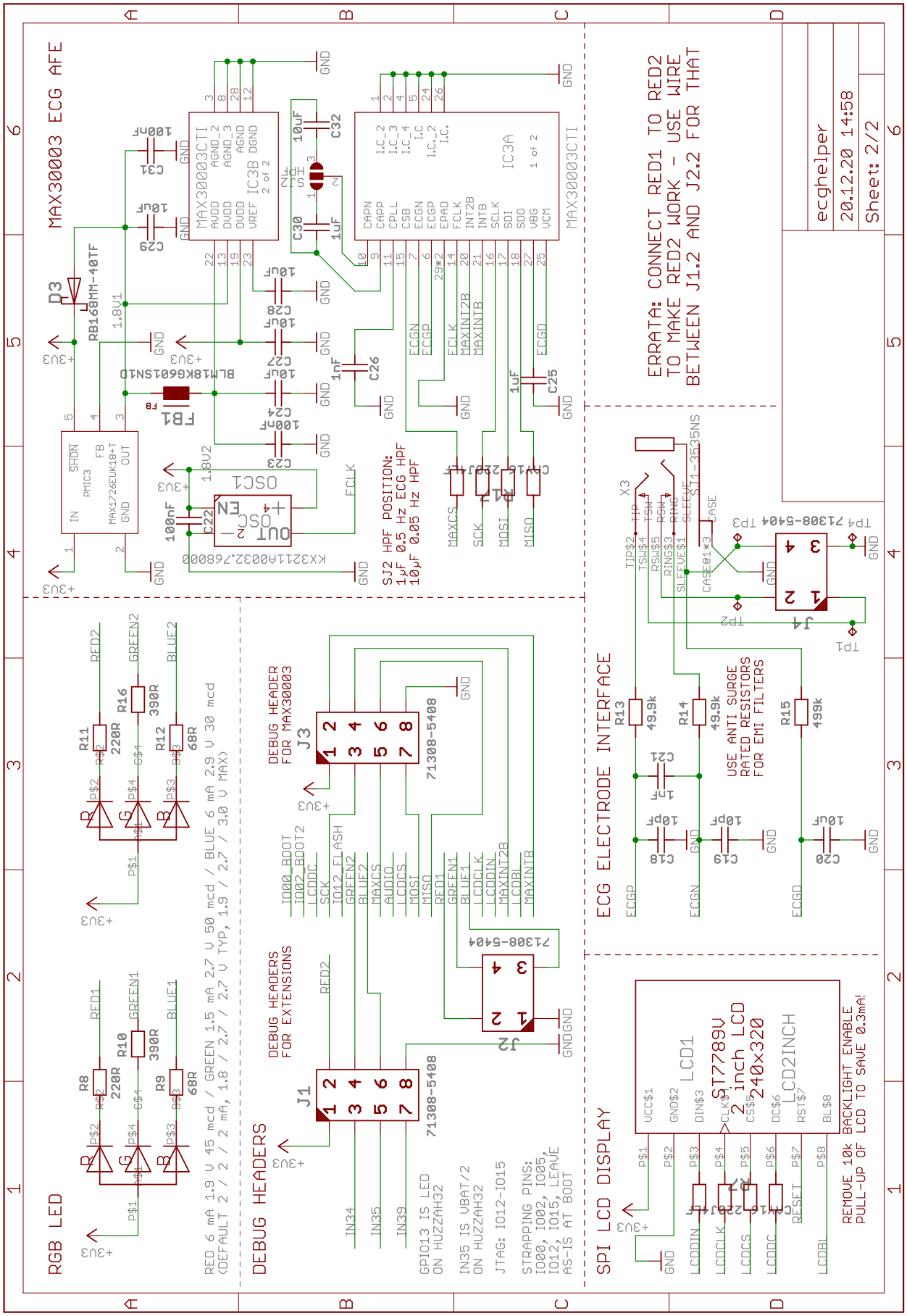


Figure A.5: Circuit diagram of the ecghelper2 device, page 2 of 2

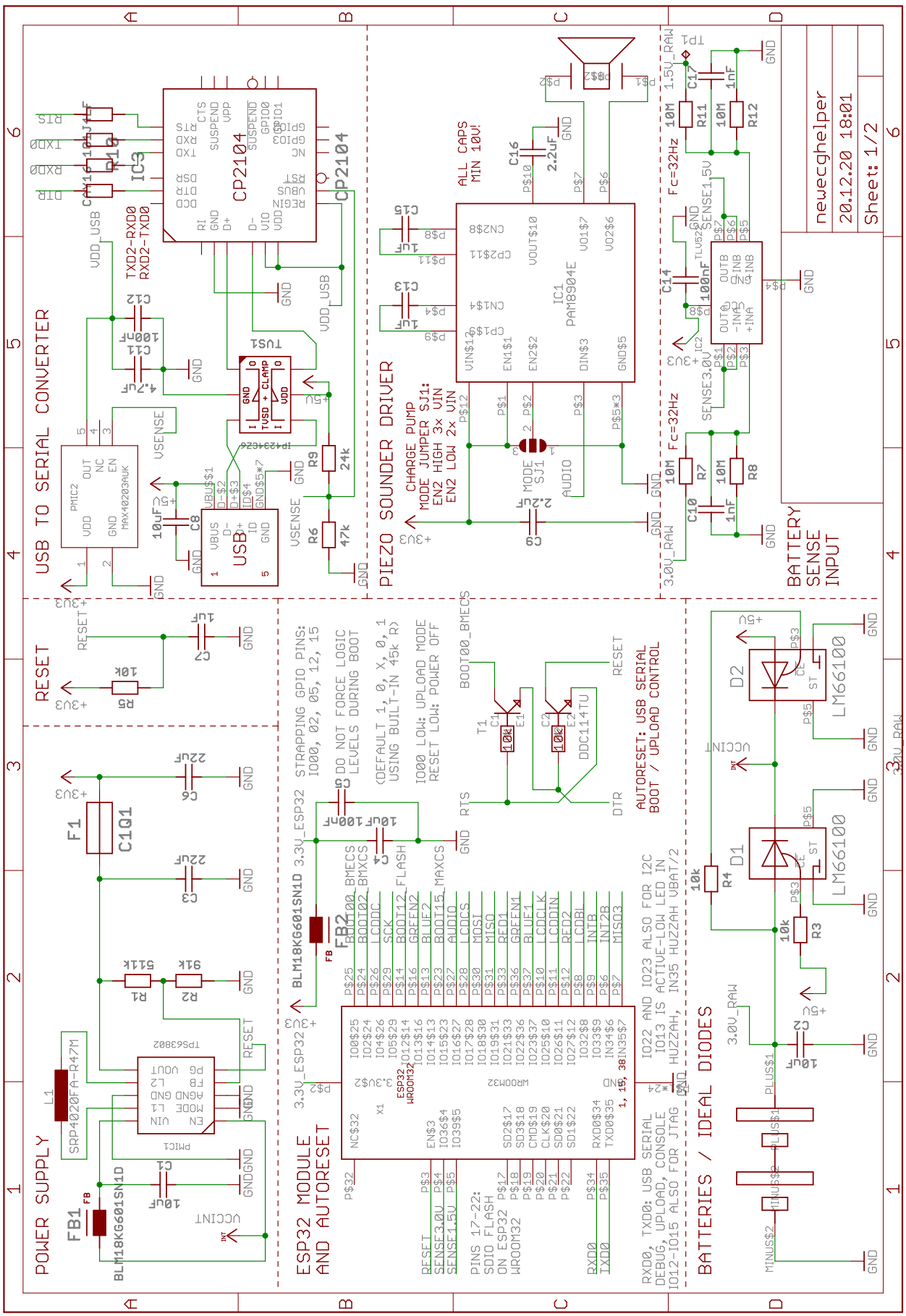


Figure A.6: Circuit diagram of a proposed ecghelper3 device, page 1 of 2

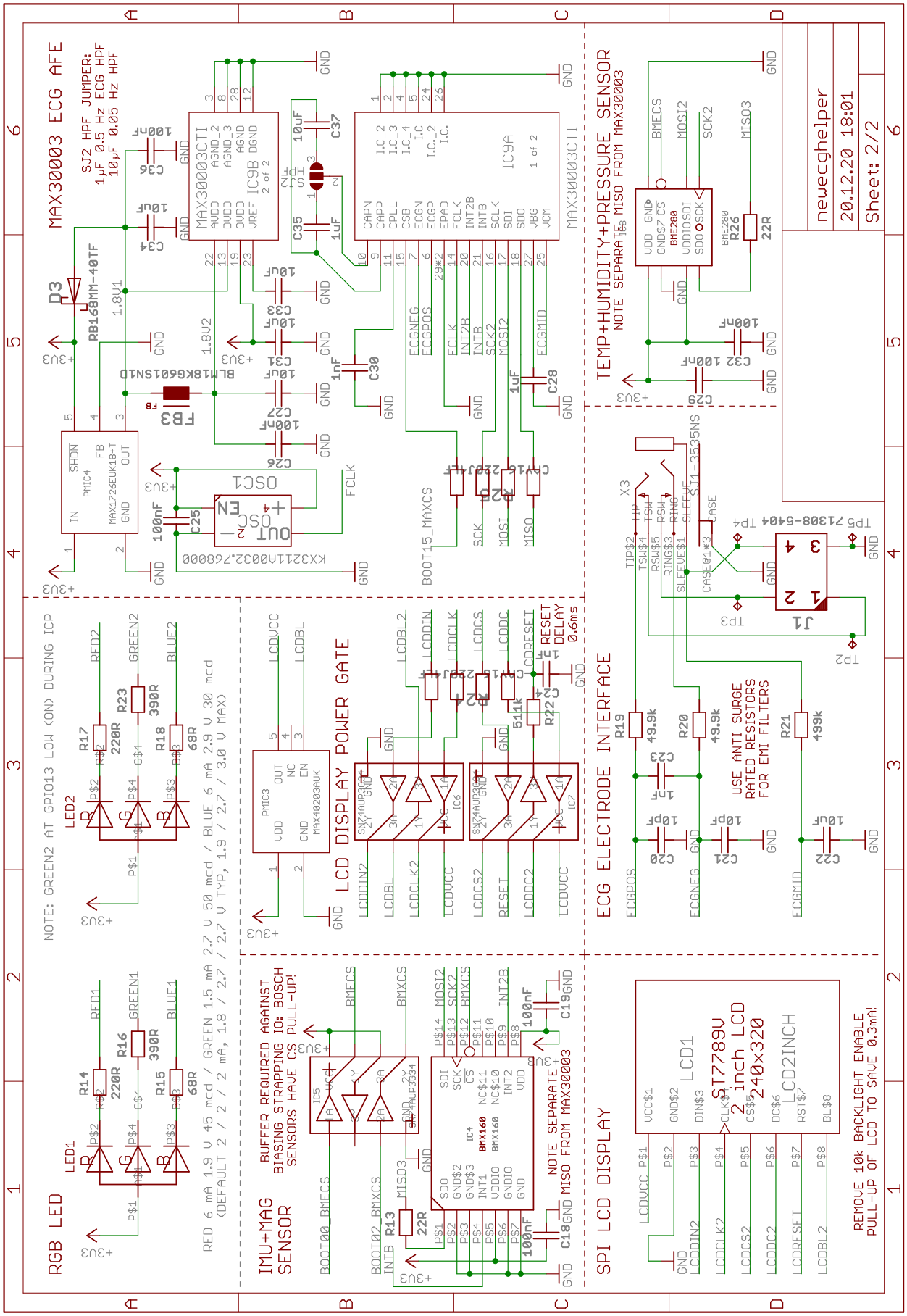


Figure A.7: Circuit diagram of a proposed ecghelper3 device, page 2 of 2

B Appendix: software details

B.1 Additional classification plots

Figure B.1 presents the histograms of ten particularly characteristic features from the `ecghelper2` and `offlineecg2` algorithm feature vector: Those include high level features derived from beat spectra and beat spectrum phase instability, such as the number of peaks in those. More peaks will be seen in a spectrum which exhibits a comb structure or irregular noise.

Spectral energy relates to spectral skew: After the energy above R/R frequency (with some margin factor) is normalised, weighting is applied to boost higher frequencies. So the energy feature reflects how overall spectral energy is balanced between higher and lower frequencies, similar to the separate spectral skew features.

Spectral dispersion describes how focused a specific spectrum is around a focus (centre of gravity) frequency. The difference between beat and phased beat spectra is whether an average of magnitudes is used (beat) or the magnitude of the average of complex values (phased). It can be seen that A-Fib data has a tendency to have lower global spectrum energy and dispersion and lower phase instability spectrum dispersion, while having a more variable phased beat spectrum dispersion.

Figure B.2 plots the weights for all items in the `ecghelper2` and `offlineecg2` feature vector used for the LDA classification, expressed in two different ways: The left axis (blue) shows the absolute eigenvector values, filtered by omitting near-zero values. The right axis (orange) plots the values in units of the standard deviation of the feature in question. This gives a better impression of which features have most effect on the decisions for the given data.

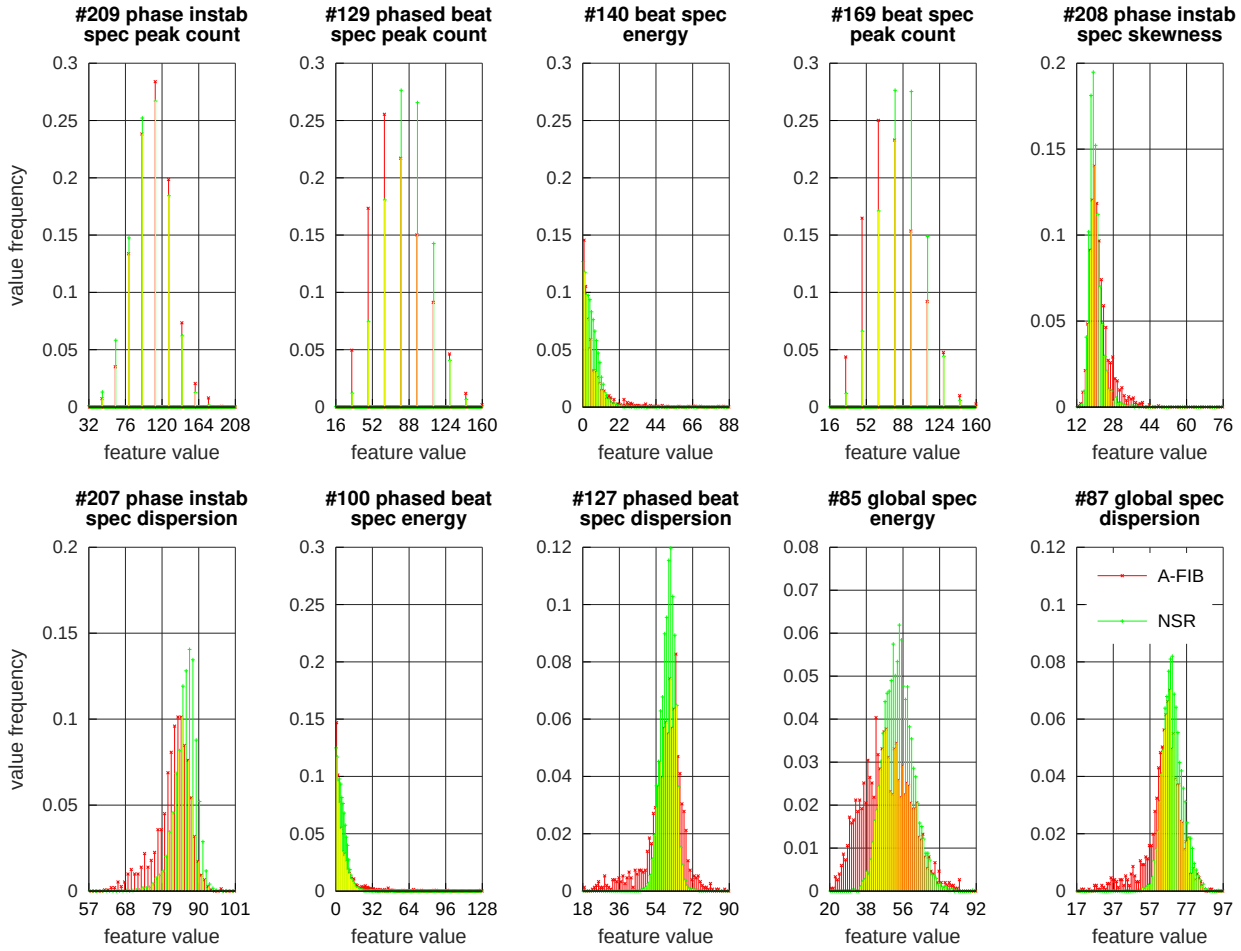


Figure B.1: Best features according to LDA classification

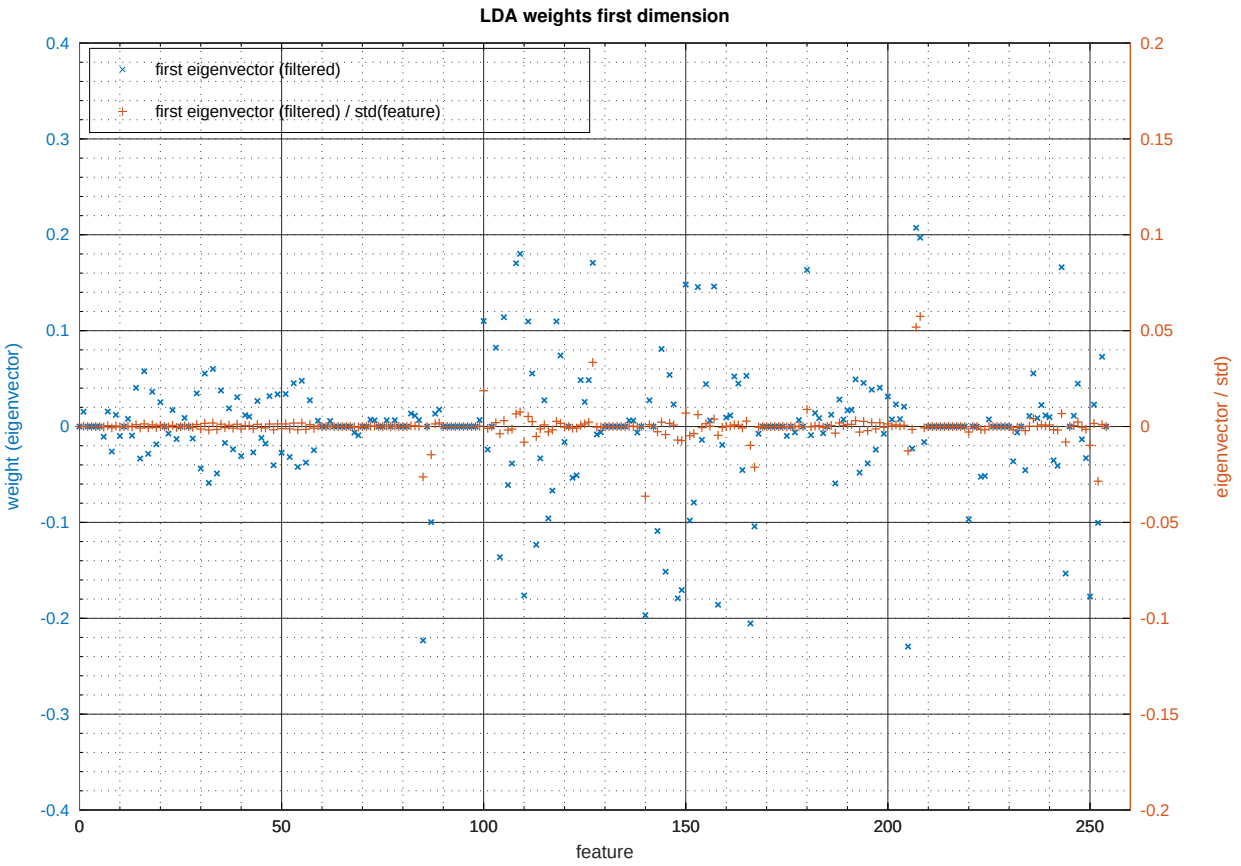


Figure B.2: Relative feature weights for classification: Main LDA dimension

B.2 Digital filter details

In the following, frequency responses, tap weights (impulse responses) and step responses for all digital filters used by the `ecghelper2` and `offlineecg2` software are shown. Filters have been chosen to give a good balance between computational effort

The section starts by presenting the new version of the 50 Hz and harmonics thereof comb filter, derived from a 10 sample moving average (sampling rate is 500 sps for all `ecghelper` generations) and comparing it to the old version from [Auer 2020], in figures B.3 and B.4

Next, the two optional Pei Tseng notch filters which are only used by `offlineecg2` are shown in figure B.5 for the 187.25 Hz filter and figure B.6 for the 60 Hz filter. Some recordings from the old corpora contain artefacts at 187.25 Hz for unknown reasons.

The basis for the QRS detection in `ecghelper2` and `offlineecg2` is a FIR correlation / matching filter generated from ECG corpus data, shown in figure B.7. It replaces the FIR band-pass filter in figure B.8 used in the previous generation, here shown with a matched gain factor.

As part of the analysis step of the averaged ECG, a smoothing filter and a slope filter, shown in figures B.9 and B.10 are applied. The smoothing filter effect is the result of repeated application of a 3 sample moving average filter, not one longer filter as the diagram would suggest.

The binomial smooth derivative slope filter, however, is applied as shown in the plot, as a single pass FIR filter. The actual analysis step uses a different gain factor, though: This fits better with the integer data type used for the slope vector.

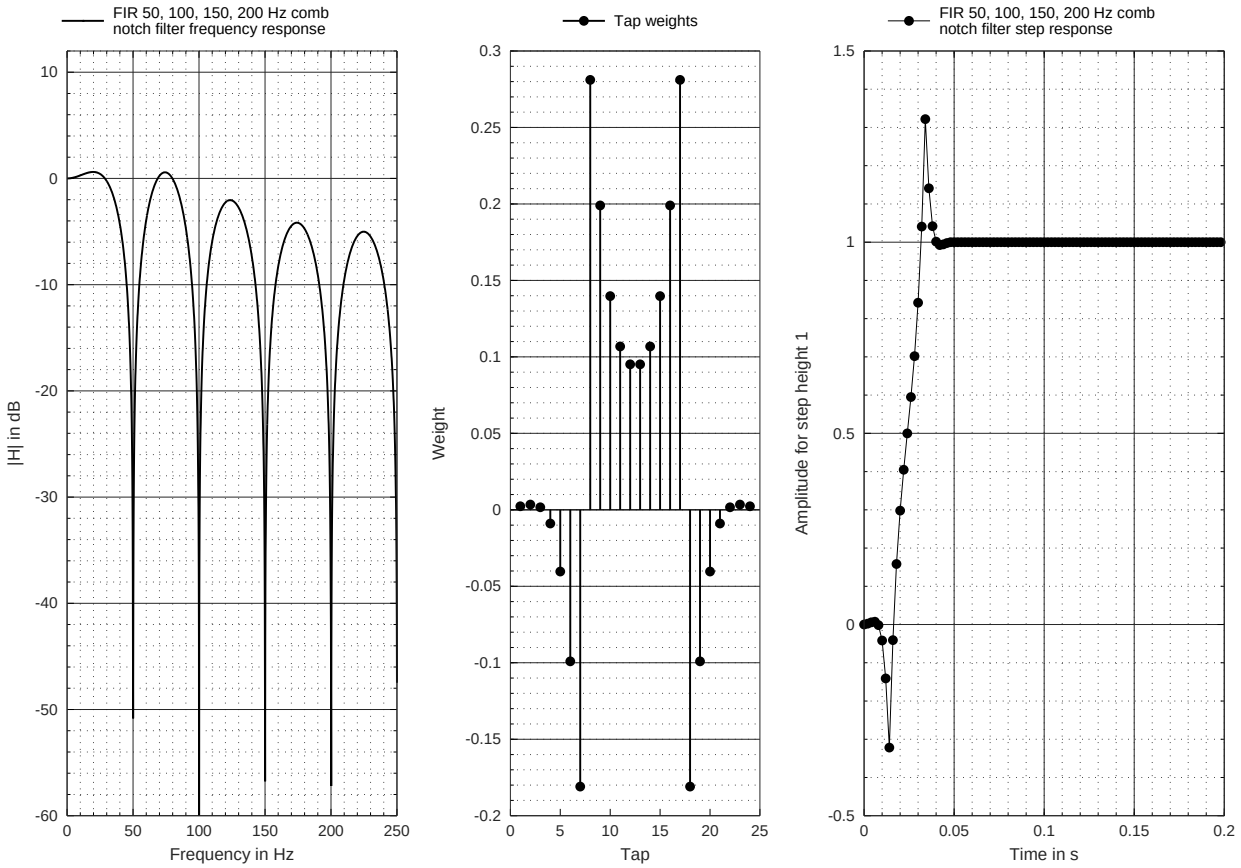


Figure B.3: Optimised FIR comb filter multiples of 50 Hz, used in ecghelper2

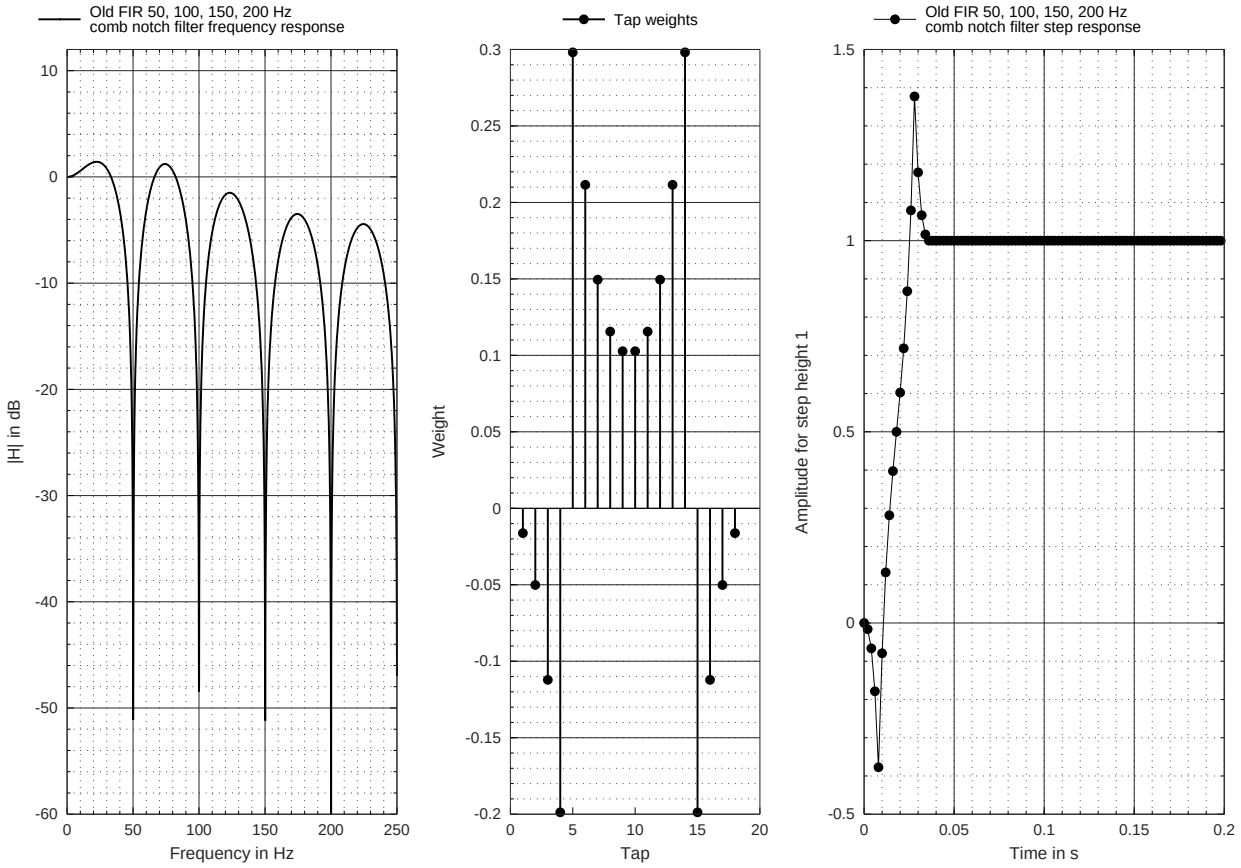


Figure B.4: Old FIR comb filter multiples of 50 Hz, used in ecghelper project

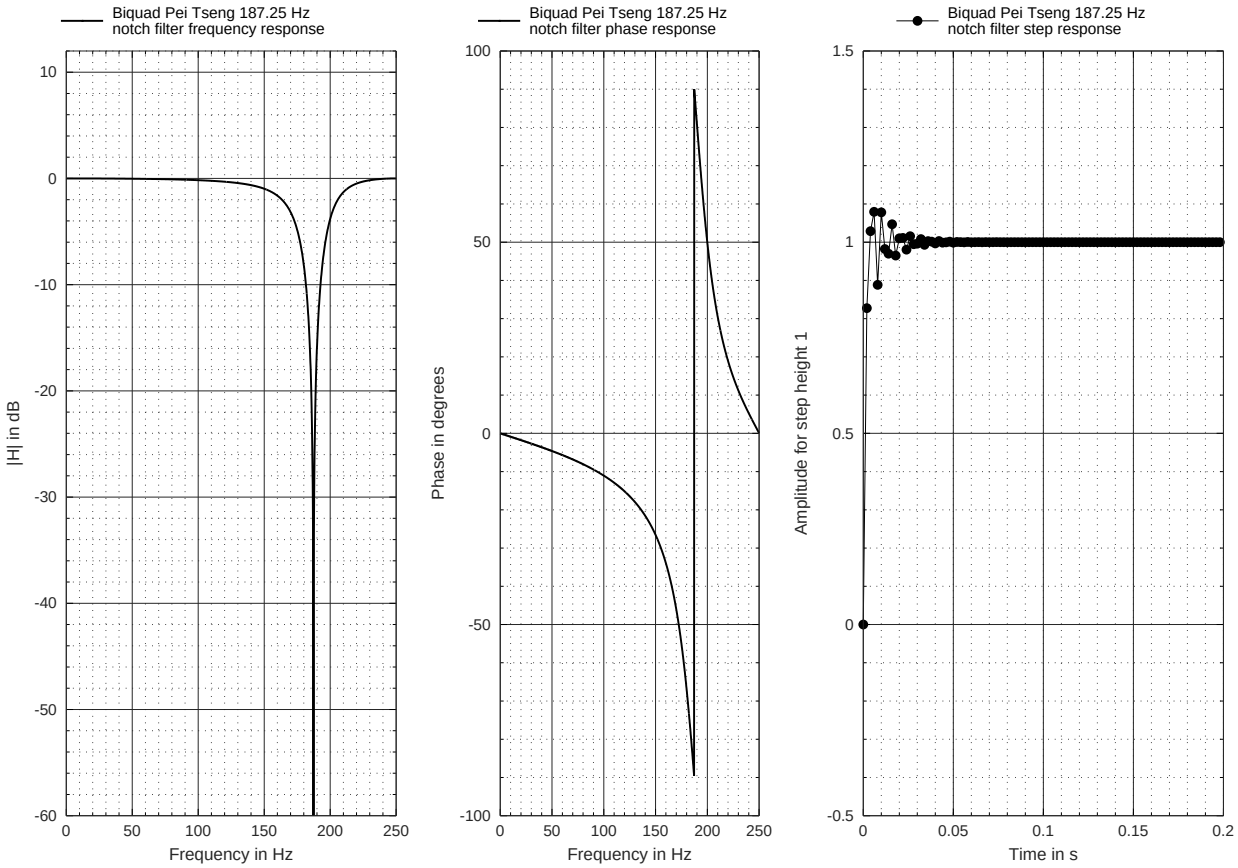


Figure B.5: Pei Tseng biquad notch filter 187.25 Hz

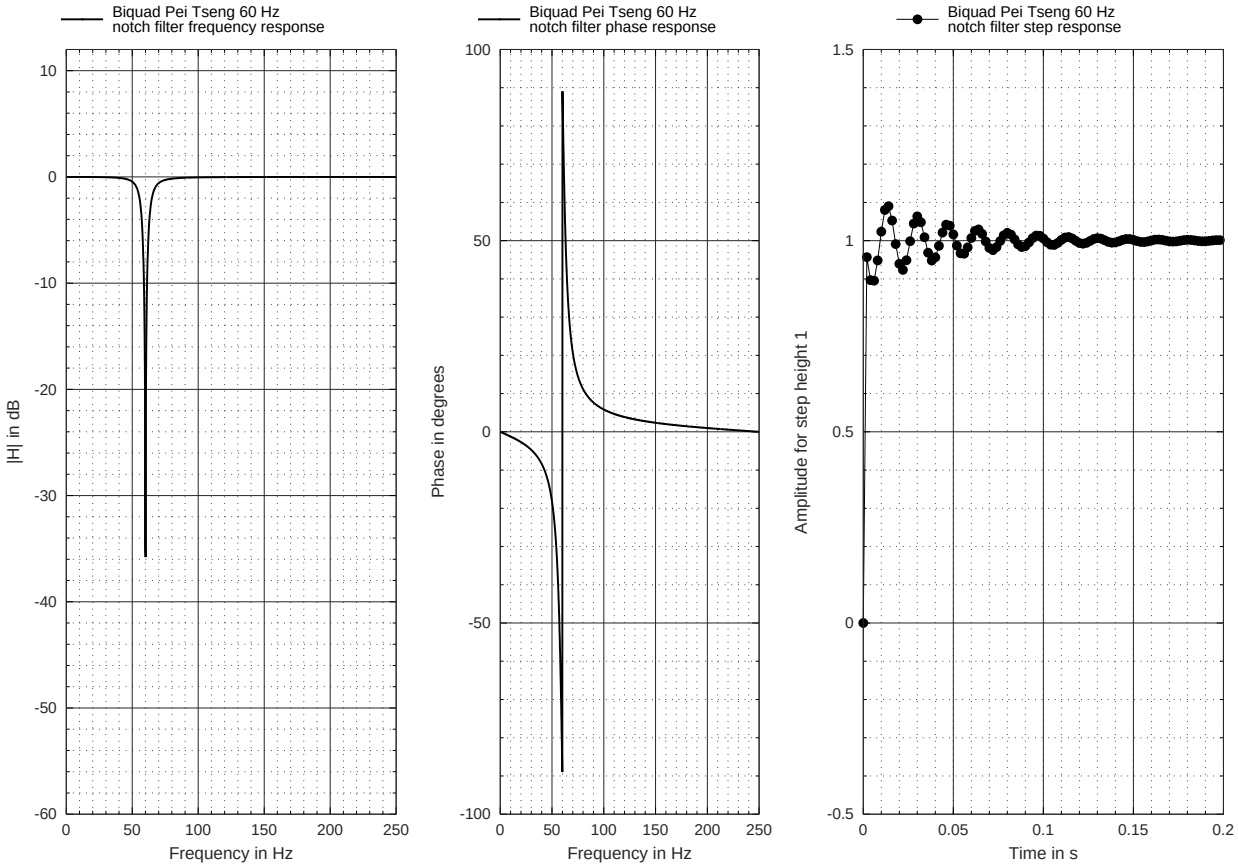


Figure B.6: Pei Tseng biquad notch filter 60 Hz

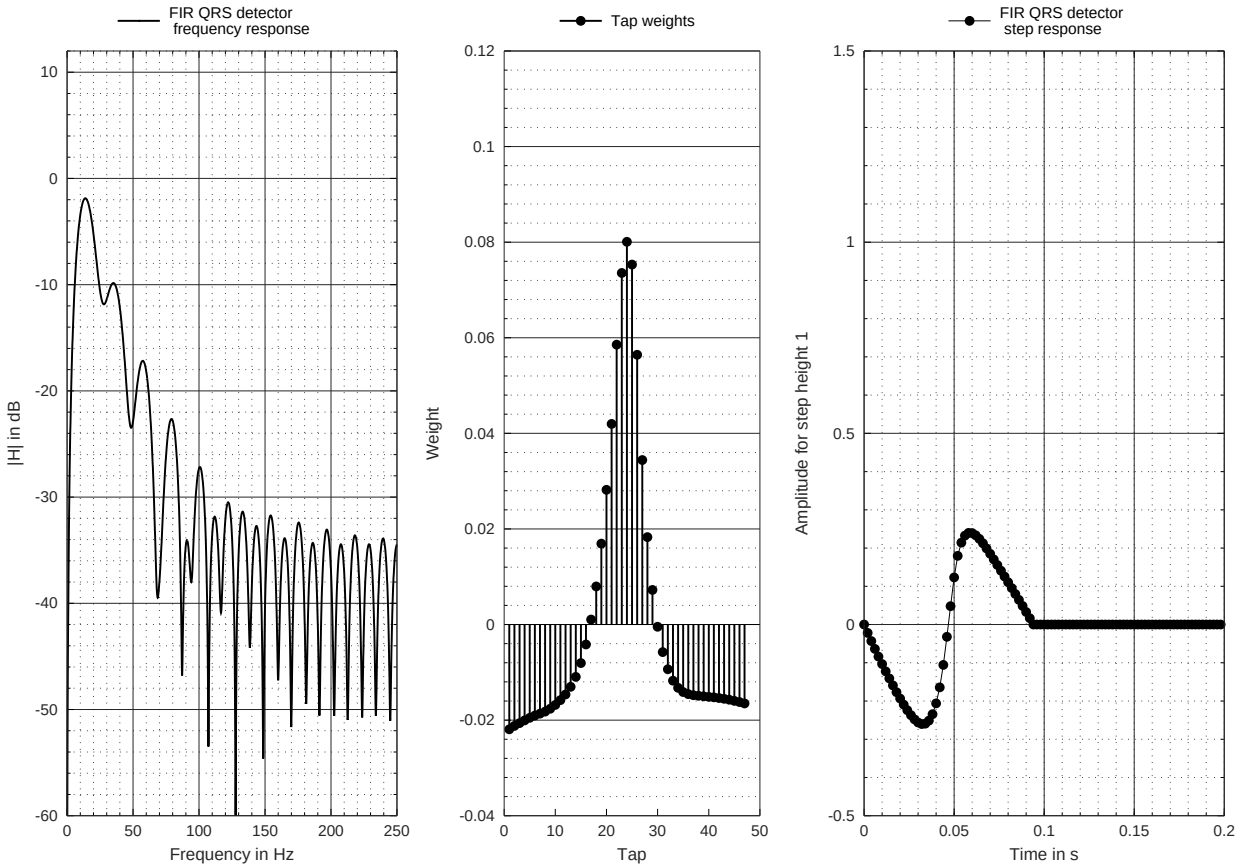


Figure B.7: FIR QRS detection filter based on corpus data, used in ecghelper2

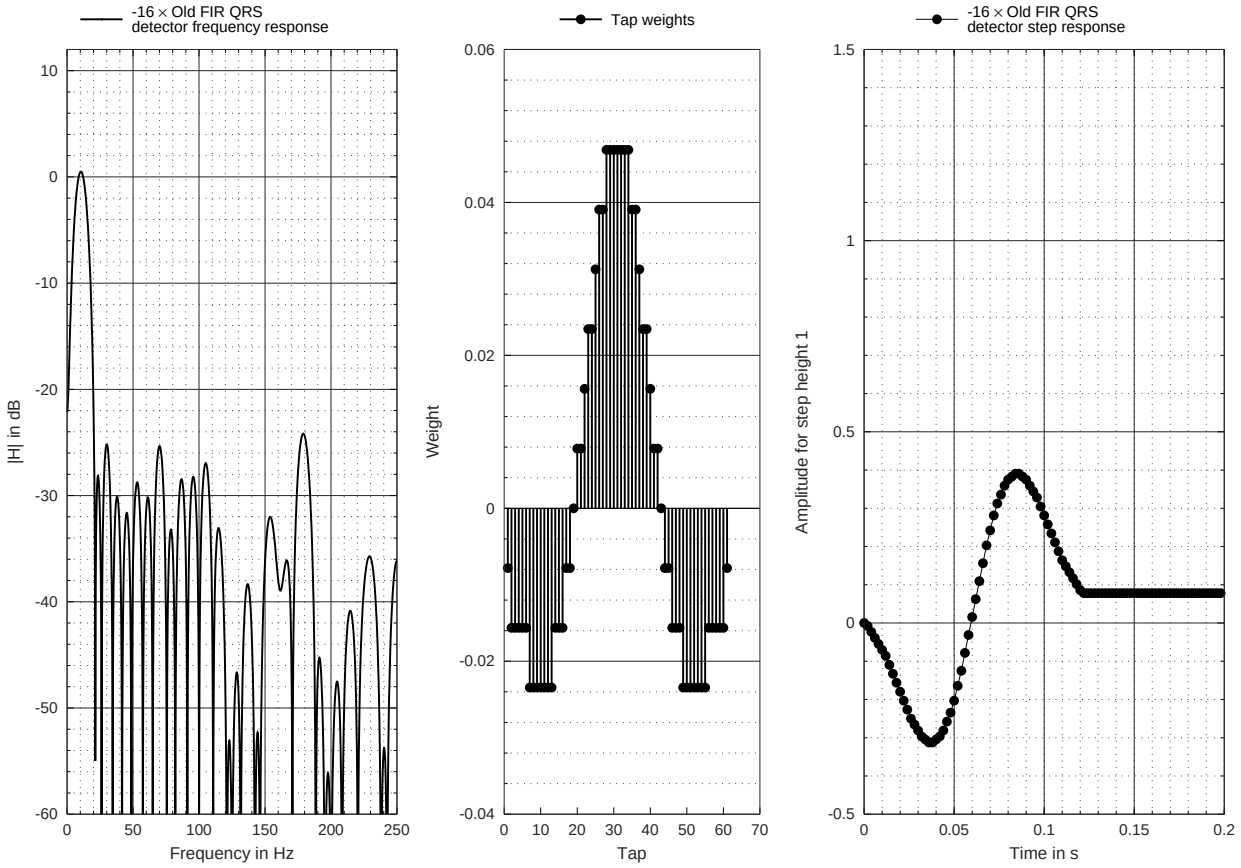


Figure B.8: Old FIR QRS detection filter, used in ecghelper project

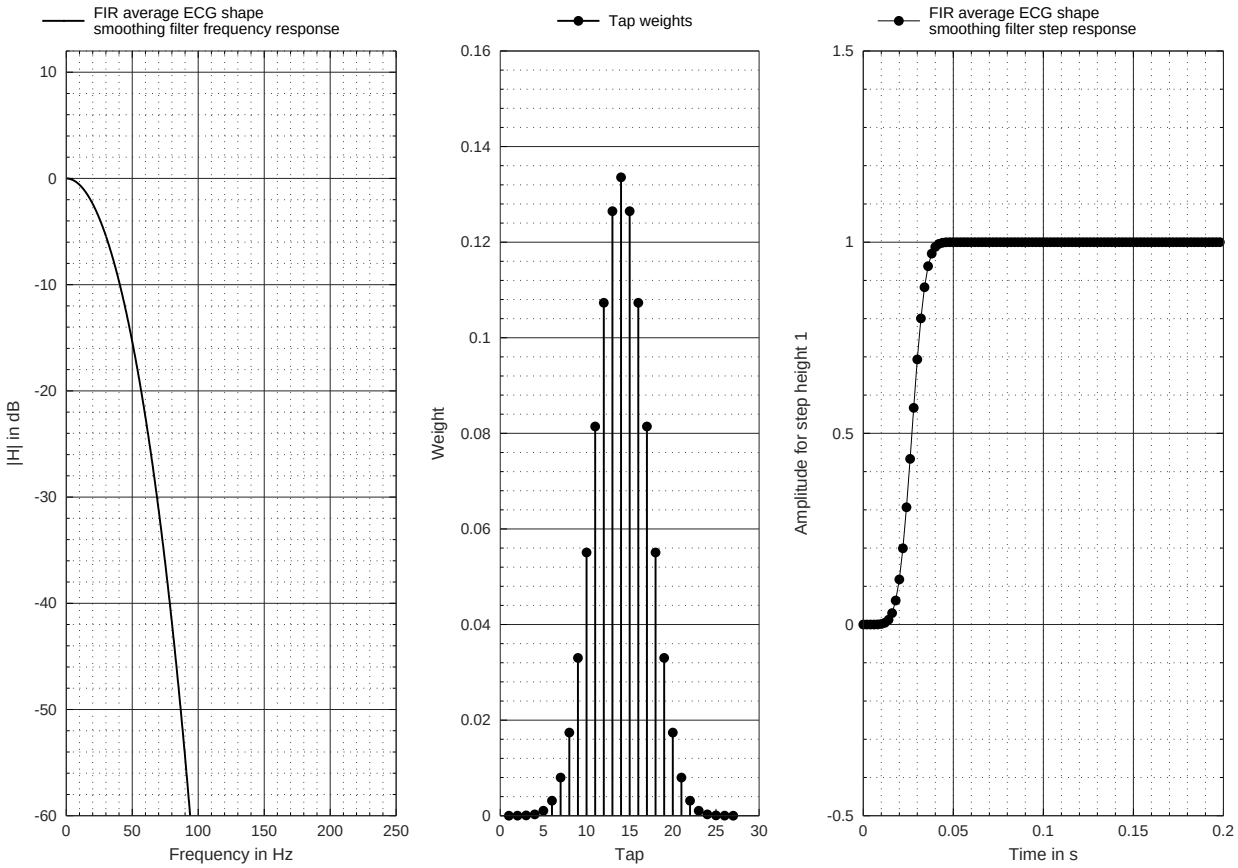


Figure B.9: ECG shape smoothing: Effect of repeated short filters, QRS excluded

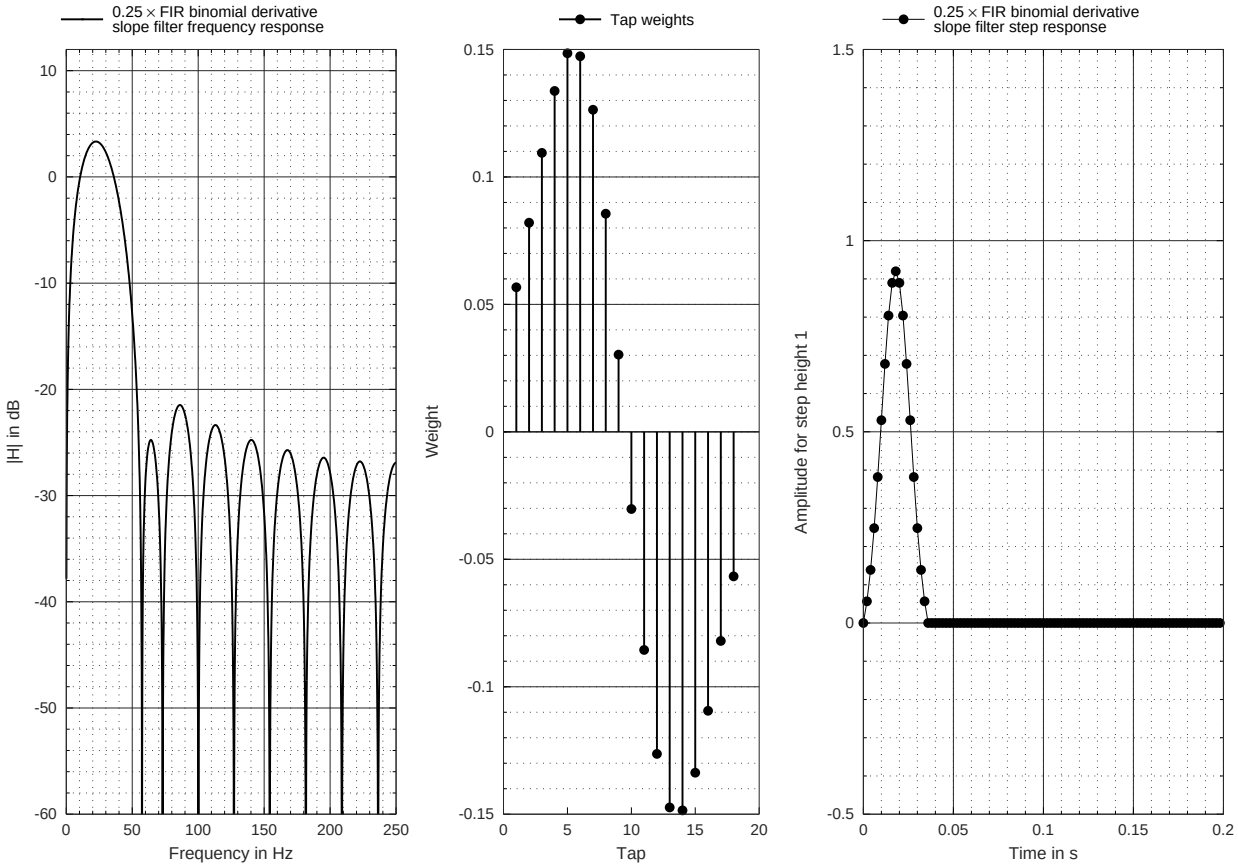


Figure B.10: Short FIR slope detection filter for shape analysis

C Appendix: software evaluation details

This appendix presents the classification results for a November 20 run of the `offlineecg2` software: It discusses the 6 cases in the project corpus, for which the new software decided that a Normal Sinus Rhythm would be present, while the corpus metadata suggest A-Fib. Note that A-Fib can stand for either atrial fibrillation or atrial flutter for the purposes of the classification.

A seventh recording, number 1-135, from the Heterogeneous Dataset of Arrhythmia [Medhi 2019], also should have been classified as A-Fib according to metadata, but got classified as NSR by `offlineecg2`. However, as that corpus is not one of the pure A-Fib corpora used to evaluate performance of the software in [Auer 2020], this recording is not discussed further in this appendix.

The appendix also shows the 10 cases in PTB-XL, where the software decides for NSR, while metadata explicitly list A-Fib, but not the 33 additional cases, where the PTB-XL metadata say A-Fib, while `offlineecg2` decides that no decision between NSR or A-Fib can be made.

Also, this appendix shows all 48 cases of NSR (according to PTB-XL metadata) where `offlineecg2` decides that a case of A-Fib could be deduced from the data. But it does not show 40 additional cases where `offlineecg2` decides that it cannot decide between NSR or A-Fib.

Note that PTB-XL is a full 12-lead corpus and diagnoses in metadata are based on all channels, while `ecghelper2` and `offlineecg2` are limited to process Einthoven Lead I ECG. The bar graphs in the plots are, from left to right: One unused slot, P-wave shape and height rating, feature threshold rule votes and LDA rating. Green stands for NSR, red for A-Fib.

C.1 Problematic analysis examples from old corpora

Recording 04015 does not exhibit obvious signs of A-Fib in manual inspection, apart from having negative T-wave polarity. The recording is from the MIT-BIH Atrial Fibrillation Database [Moody 1983] which contains 25 long-term recordings from patients with A-Fib: Analysis here uses short excerpts starting 4 hours after the beginning of each of the 23 recordings which feature Einthoven Lead I data, so it is easily possible that no A-Fib was ongoing in the analysed time frame. Note that the corpus originally is about R/R interval based A-Fib detection and that the analysis here does show suspiciously constant R/R intervals.

Recording 04048, from the same corpus, does exhibit some irregularities in the averaged ECG shape. The R/R intervals are quite constant, apart from a single exception caused by a mistake of the R-

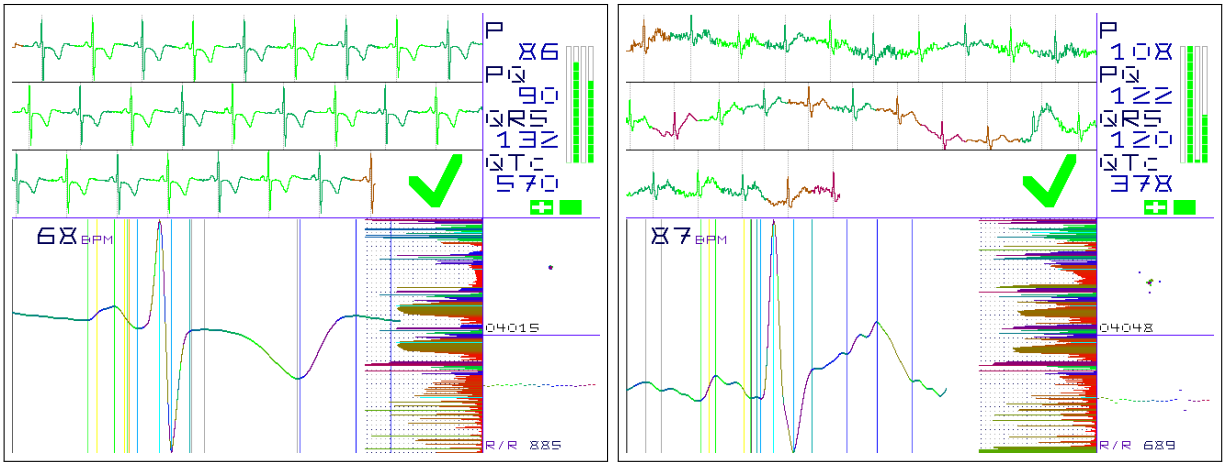


Figure C.1: A-Fib in old corpus classified as NSR. Left: 04015, inverse T, rules 2:2, 1 noise rule, LDA: NSR. Right: 04048, intermittent A-fib, rules 1:0 NSR, LDA: NSR

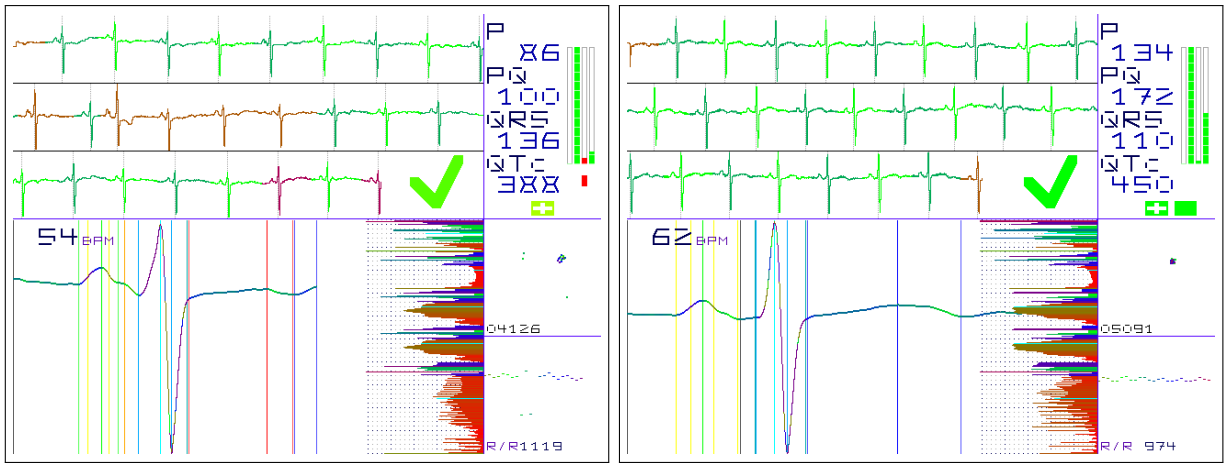


Figure C.2: A-Fib from old corpus classified as NSR. Left: 04126, LDA: NSR, overrules 2 A-Fib rules and A-Fib shape. Right: 05091, shallow T, long PR and P duration, 1 NSR rule, LDA: NSR

peak pinpointing algorithm which moved the found R timestamp to a nearby artefact from signal drift combined with a T-wave. As the recording generally has some drift, the beat around the misaligned R still got selected as one of the beats for averaging. The amount of artefacts in the recording varies over time, so there could be a tendency towards developing A-Fib.

Recording 04126 is notable for showing very weak T-waves. Some feature threshold rules voted for A-Fib, but not enough to make the overall classification switch to A-Fib. The recording also contains a single premature beat, while otherwise having normal periodic R/R variations. The LDA result is close to undecided, but tends towards NSR. As clear P-waves are present, the final `offlineecg2` classification is NSR.

Similarly, recording 05091 only has weak T-waves, which also exhibit a suspiciously high QTc. While feature rules are undecided, LDA decides that the class is NSR, supported by the well-shaped P-waves. As with recording 04126, R/R variations are small, but plausibly periodical.

Recording 05261 has somewhat suspiciously shaped and timed T-waves, but clear P-waves. The R/R intervals show little variation. Feature rules are neutral and LDA leans well towards NSR.

From the Intracardiac Atrial Fibrillation Database [Peterson 2003] corpus, recording IAF5 TVA is a clear case of atrial flutter when inspected manually.

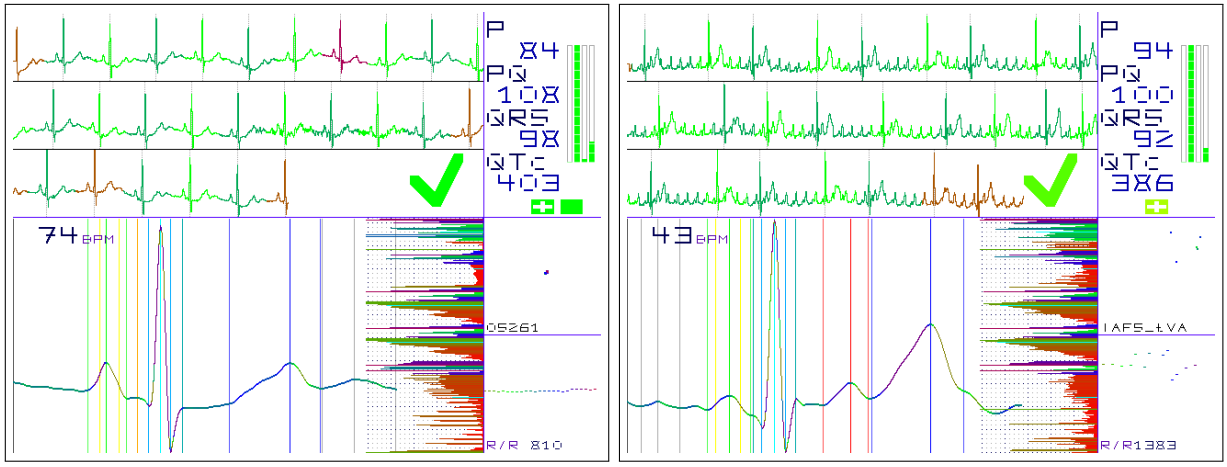


Figure C.3: A-Fib from old corpus classified as NSR. Left: 05261, only intermittent A-Fib, P and T amplitude rules NSR, ST amplitude rule A-Fib, LDA NSR, good P shape. Right: iaf5 tva, flutter, LDA decides NSR, overrules A-Fib shape

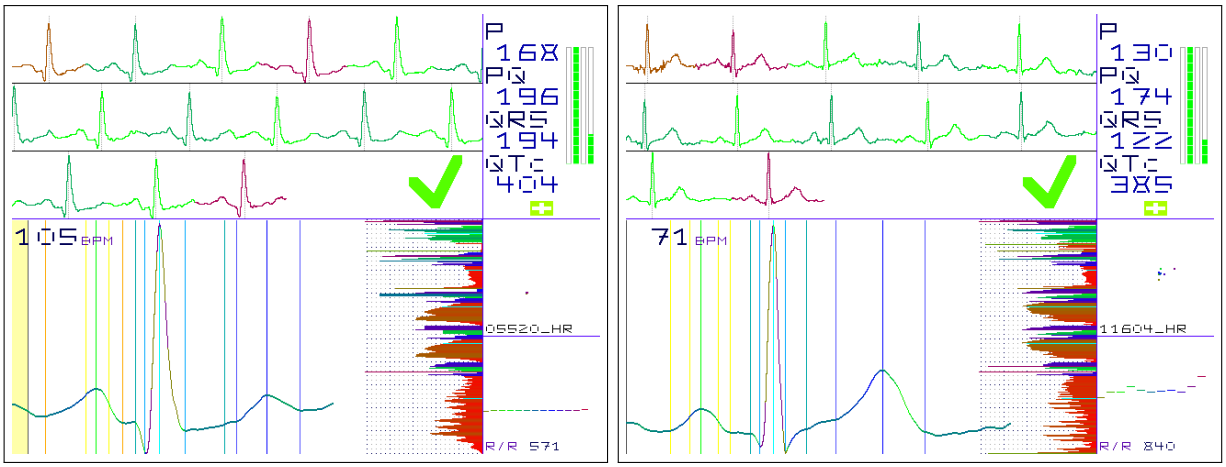


Figure C.4: A-Fib in the PTB-XL corpus classified as NSR 1/5

However, the flutter waves just failed to raise enough A-Fib triggers to get the recording as a whole classified as A-Fib: The R/R metrics show irregularities and the flutter waves do get noticed after the averaged QRS complex, but not before it. The P-wave shape (emphasised by the Q-wave) and other features contribute to a LDA result which is far enough into NSR range to override the A-Fib detection which would normally have been triggered by spotting the flutter wave between QRS and T-wave. So the ultimate classification is NSR here, which clearly is an error made by the software.

C.2 PTB-XL A-Fib recordings classified as NSR

In the following, all missed cases of A-Fib when applying the `offlineecg2` analysis to all A-Fib recordings in PTB-XL are discussed, based on PTB-XL metadata. Recordings for which the software decided that it would not be able to make any decision have been excluded here: In those 33 instances, the embedded device would have indicated that the user has to repeat the measurement to get better data, which of course is no option while processing PTB-XL recordings.

Apart from having somewhat degraded S- and T-waves and an elevated, almost constant heart rate, PTB-XL recording 5520 shows a plausible P-wave morphology and LDA supports the decision towards NSR. Using longer recordings could have clarified the diagnosis.

Recording 11604 does seem to be the recording of a NSR, based on manual inspection by the author. The spectrum does have a bit of a skew towards higher frequencies and LDA is not extremely clear, but in general, the evidence does make the decision of the software to classify this as NSR rather plausible, contradicting PTB-XL metadata.

The T-waves of recording 11921, note the low 53 bpm pulse rate, are peaked and have even higher amplitudes than the QRS complexes. The QTc of 412 ms also is a bit high and there are few jumps in R/R interval length. Still, LDA leans towards NSR and P-waves are well visible in the averaged ECG, although a bit long. The global spectrum suggests that atrial flutter might be more likely than fibrillation for this recording.

For recording 15180, a segment where 3 small waves can be seen between 2 QRS complexes could be a clue that we are indeed seeing atrial flutter in this recording. The oddly shaped averaged T-wave supports this. For the automated classification, however, too few feature rules trigger for A-Fib, P-waves are clear and LDA supports the NSR classification. Note the relatively high QTc of 416 ms and the suspiciously constant R/R intervals.

For recording 15632, the general impression is that of an NSR ECG. Spectral properties and the long P-wave cast some doubt on this. Still, P-wave amplitude and LDA support a NSR result.

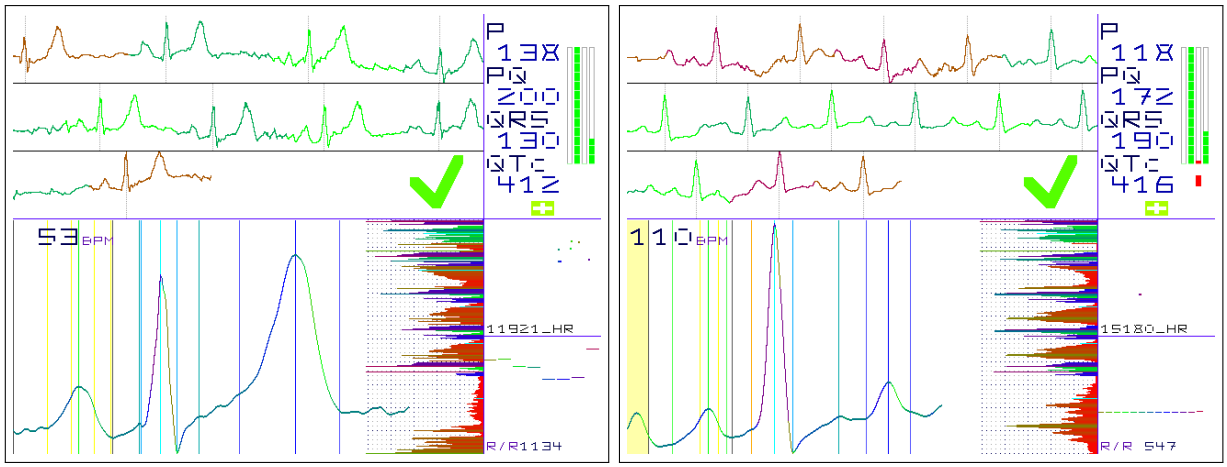


Figure C.5: A-Fib in the PTB-XL corpus classified as NSR 2/5

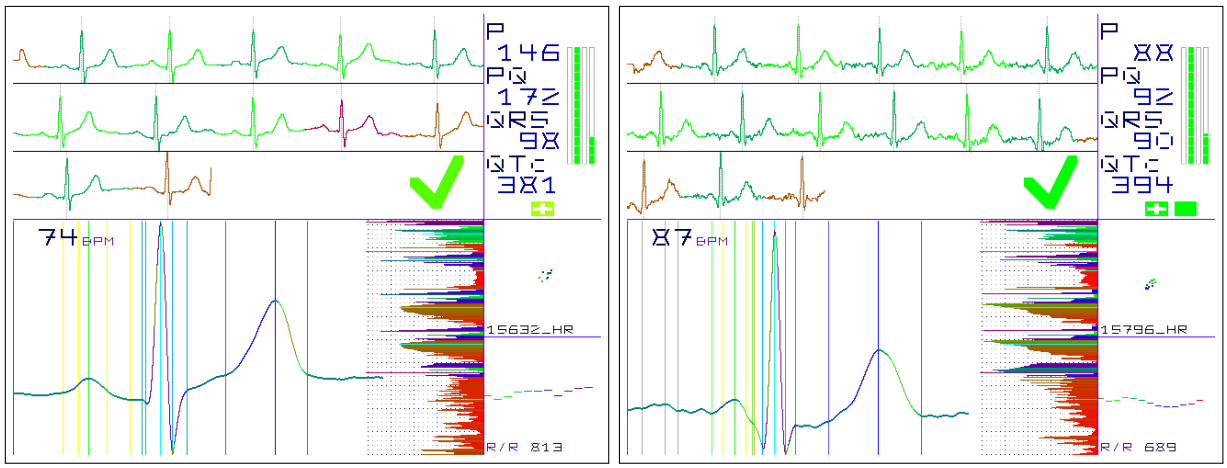


Figure C.6: A-Fib in the PTB-XL corpus classified as NSR 3/5

In recording 15796, the R/R intervals show smooth variations, which would support a NSR diagnosis. However, all spectra – global and beat related ones – are skewed towards higher frequencies. While P-waves are strong with some support of their Q-wave context, there are some weak irregularities before them and the P morphology and timing raise doubts. LDA is not too sure, but taking all aspects of the classification into account, this is the only PTB-XL A-Fib ECG for which `offlineecg2` is absolutely sure (good rating, not just okay rating) to see NSR. It should not be. According to the author, the software better had classified this case as undecidable.

Recording 16401 features barely enough P-waves to pass the NSR test and a barely NSR outcome of the LDA. Tuning parameters towards more strict NSR requirements would have let `offlineecg2` classify this as A-Fib, which seems more appropriate.

While spectra are skewed towards higher frequencies in recording 19250, most are phase-consistent across beats, which may support NSR. Both P- and T-waves are somewhat shallow and long, but R/R distribution seems normal. The classification as NSR is relatively plausible. It would be interesting to check a longer recording here.

Recording 20184 marks the minimum standards of `offlineecg2` with the current settings: LDA is almost undecided, but leans a bit towards NSR, while too few feature rules vote for A-Fib. The P-wave has sufficient amplitude, but is suspiciously long before the QRS complex, which also is not much more than a R-peak. The T-wave timing is off and there are some irregularities after it.

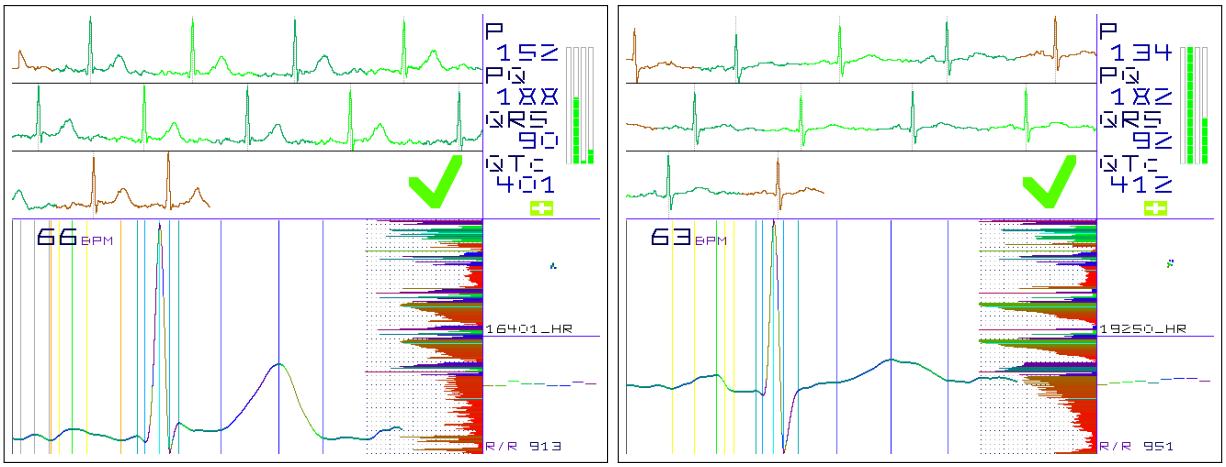


Figure C.7: A-Fib in the PTB-XL corpus classified as NSR 4/5

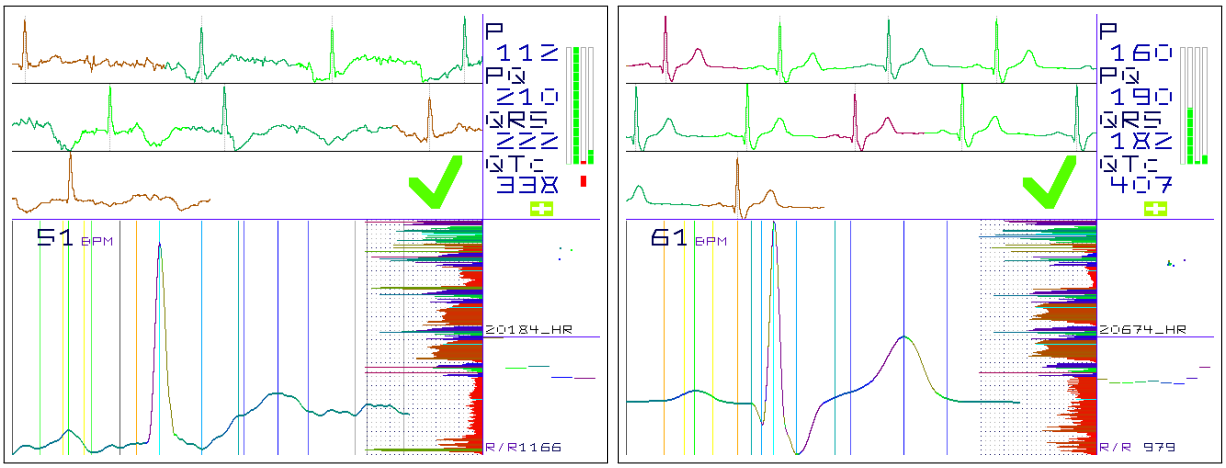


Figure C.8: A-Fib in the PTB-XL corpus classified as NSR 5/5

Low heart rate and only 5 beats usable for averaging also are near the minimum requirements to pass the classification. Based on manual classification, this would be an undecided, suspicious recording, with artefacts near the start hinting at A-Fib. Also, the longer pause before the two beats near the end of the recording (both not used in averaging) supports the suspicion that something pathological is going on in this ECG, but not necessarily A-Fib.

Metadata for this recording contain, translated from German: A-Fib / A-Fl, left axis deviation, peripheral voltage, abnormal QRS(t) and anteroseptal infarction, probably old, inferior infarction, age unconfirmed, unconfirmed diagnosis.

Finally, recording 20674 seems to be another case of near-NSR ECG, but the averaged P-waves are long and shallow and the LDA is close to being undecided. According to PTB-XL metadata, both A-Fib and a right bundle branch block are present in the recording, probably based on the prolonged S-wave visible even in the Einthoven Lead I data and averaged ECG.

C.3 PTB-XL NSR recordings classified as A-Fib

As the PTB-XL corpus contains 4 times more NSR examples than A-Fib examples, it is not surprising that dozens of NSR are falsely classified as A-Fib by the same algorithm and settings which only missed the 10 out of more than 1500 A-Fib examples discussed above. Comments for this section are deliberately kept short, but the readers can use their own judgement regarding the examples and classifications.

Record	Comments
00080	A-Fib LDA, may contain intermittent A-Fib? Good P-waves
00645	Negligible P-wave, large R/R jumps, might really be A-Fib
00862	P-wave, LDA, R/R and features **all** weakly suggest A-Fib
01331	Weak T- and too weak P-waves, LDA almost undecided, spectrum?
02374	Frequent R/R detection errors, muscle or A-Fib artefacts? Weak T
02393	Weak R (logs say 210 μV), artefacts, weak LDA A-Fib decision
02760	Irregular R/R and minimal P-waves suggest actual A-Fib case
03103	Strong artefacts (4 Hz peak?) suggest A-Fib, 60 Hz hum? QTc 435 ms
03301	Irregular R/R, spectrum and LDA support classification as A-Fib, metadata list PAC / possible brief supraventricular tachycardia
03343	P-waves absent, other ECG shape metrics support A-Fib classification
03395	R/R bigeminy, LDA: A-Fib, QTc 425 ms, just NSR in metadata
04210	Slightly too shallow averaged P-waves, LDA slightly A-Fib
04678	No P-wave in averaged ECG, hum (?), LDA slightly A-Fib, QTc 418 ms
05618	Strong artefacts, R only 315 μV, LDA: A-Fib, no plausible T-shape
05704	Averaging fails to find consistent P-wave, LDA says A-Fib
06860	Artefacts or fibrillation, no T-waves in averaging, metadata: sinus bradycardia and left axis deviation, LDA clearly A-Fib, extra beats
07937	R only 146 μV, strong artefacts make R detection hard, LDA: A-Fib
08771	Software finds 206 μV R in spite of artefacts, irregular ECG, LDA: A-Fib
08876	Weak R, feature rule based classification as A-Fib
09773	Too weak averaged P-wave, LDA supports A-Fib classification
10183	Very weak QRS complexes, clear A-Fib decision from feature rules
10369	Strong artefacts, extra waves in average after T-wave, LDA. A-Flutter?
10373	No clear P-wave in average, LDA supports decision
10627	Again no clear P-wave in average, LDA clearly A-Fib, QTc 421 ms
11112	Corner case, overall decision of algorithm is A-Fib
11149	Reduced R, two-humped P, weak LDA decision for A-Fib
11209	Artefacts / extra waves in average: A-Fib? Slow, irregular R/R
11407	60 Hz filter triggered, spectrum skewed towards higher frequencies, multiple extra waves in average, no clear T-wave
11452	Weak LDA decision towards A-Fib, looks like NSR with weak artefacts
11576	LDA: A-Fib, averaged ECG shape good, global spectrum may support A-Fib
11721	Weak averaged P-wave, weak LDA decision for A-Fib, QTc 420 ms
11790	Corner case for the classification, degraded averaged P-wave
11853	Artefacts and spectrum suggest A-Fib by LDA and flutter waves
11884	Algorithm fails to extract P-wave, almost okay in average?
11984	Weak LDA decision towards A-Fib, R/R steps are a bit large
11989	R only 192 μV, some suspicious waves in raw ECG, LDA: A-Fib
12283	Weak averaged P, missing S, metadata list sinus bradycardia with sinus arrhythmia, LDA strongly suggests A-Fib
12489	Weak A-Fib decision, suspicious P-wave shape Metadata: supraventricular tachycardia. p wave axis is abnormal and pr interval is short: rhythm is probably junctional
13341	ECG with significant drift, extra waves in average
14078	No clear P-wave in average, LDA supports A-Fib, fixed R/R
16183	Weak P and weak LDA towards A-Fib, long QTc of 435 ms
17820	Weak LDA towards A-Fib, two extrasystoles
17962	No clear P-wave in average, LDA clearly A-Fib, extra QRS detection candidates
18741	Some R/R instabilities, average lacks P-wave, shows signs of probability distribution instead, LDA strongly supports A-Fib
18999	One extra QRS detector trigger, LDA says A-Fib, looks like NSR
19184	A-Fib based on LDA only, but R/R bigeminy might support LDA
21607	Supraventricular extrasystoles and sinus bradycardia according to metadata, LDA suggests A-Fib, average ECG okay, QTc 423 ms
21756	R/R bigeminy, reasonable average ECG shape, weak LDA A-Fib decision

Table C.2: List of PTB-XL recordings classified as A-Fib in spite of NSR metadata

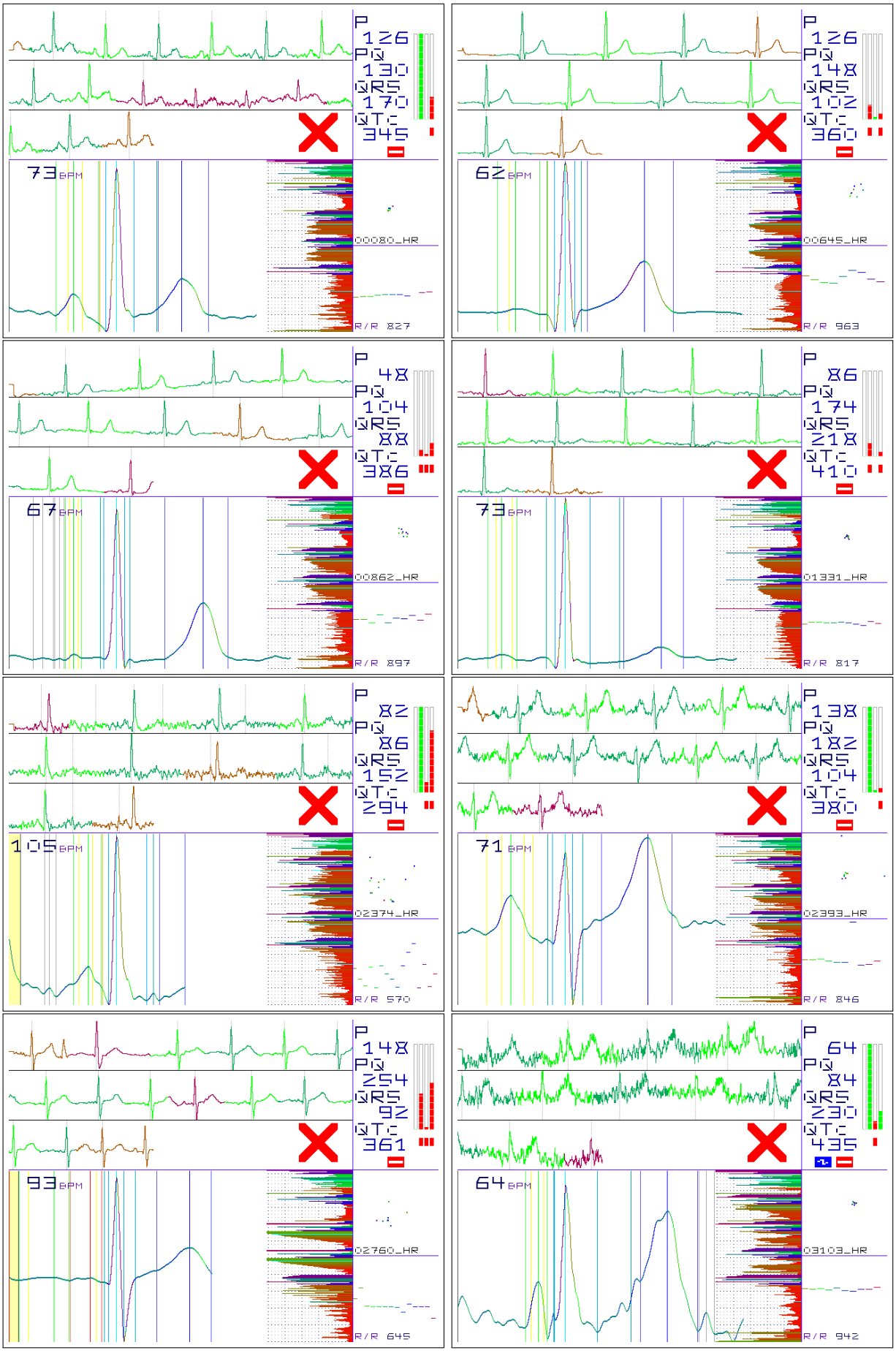


Figure C.9: NSR in the PTB-XL corpus classified as A-Fib 1/6

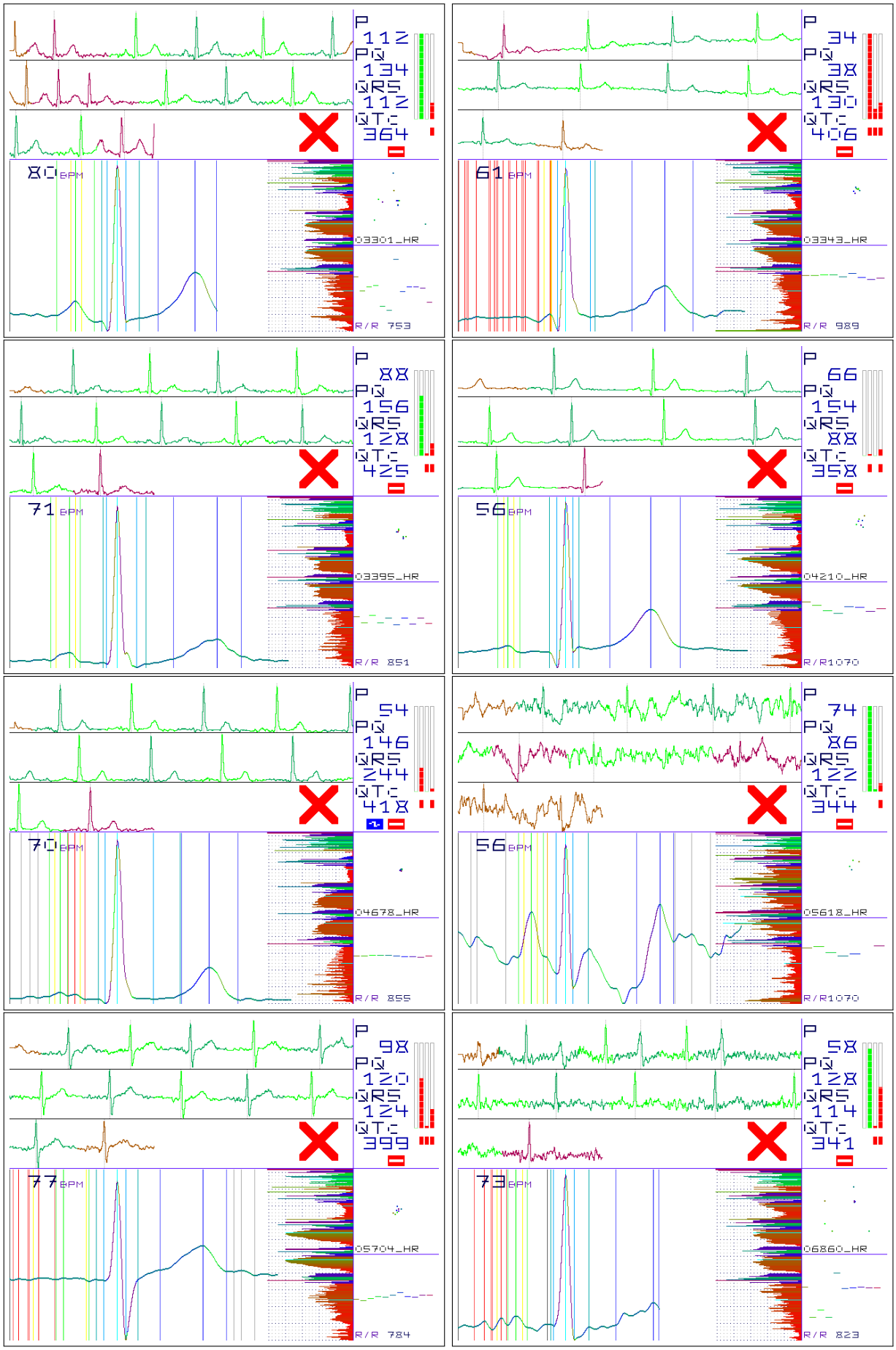


Figure C.10: NSR in the PTB-XL corpus classified as A-Fib 2/6

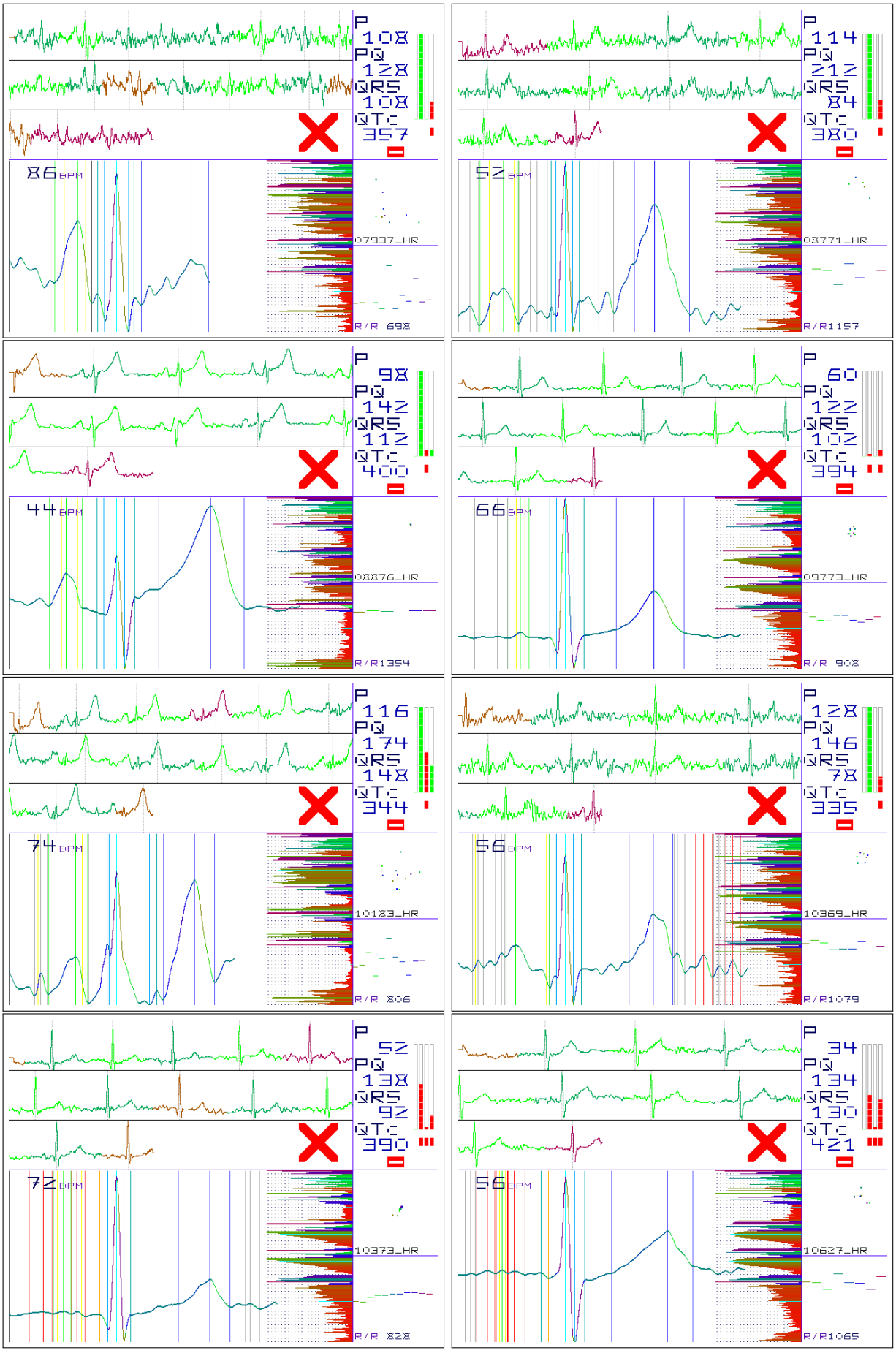


Figure C.11: NSR in the PTB-XL corpus classified as A-Fib 3/6

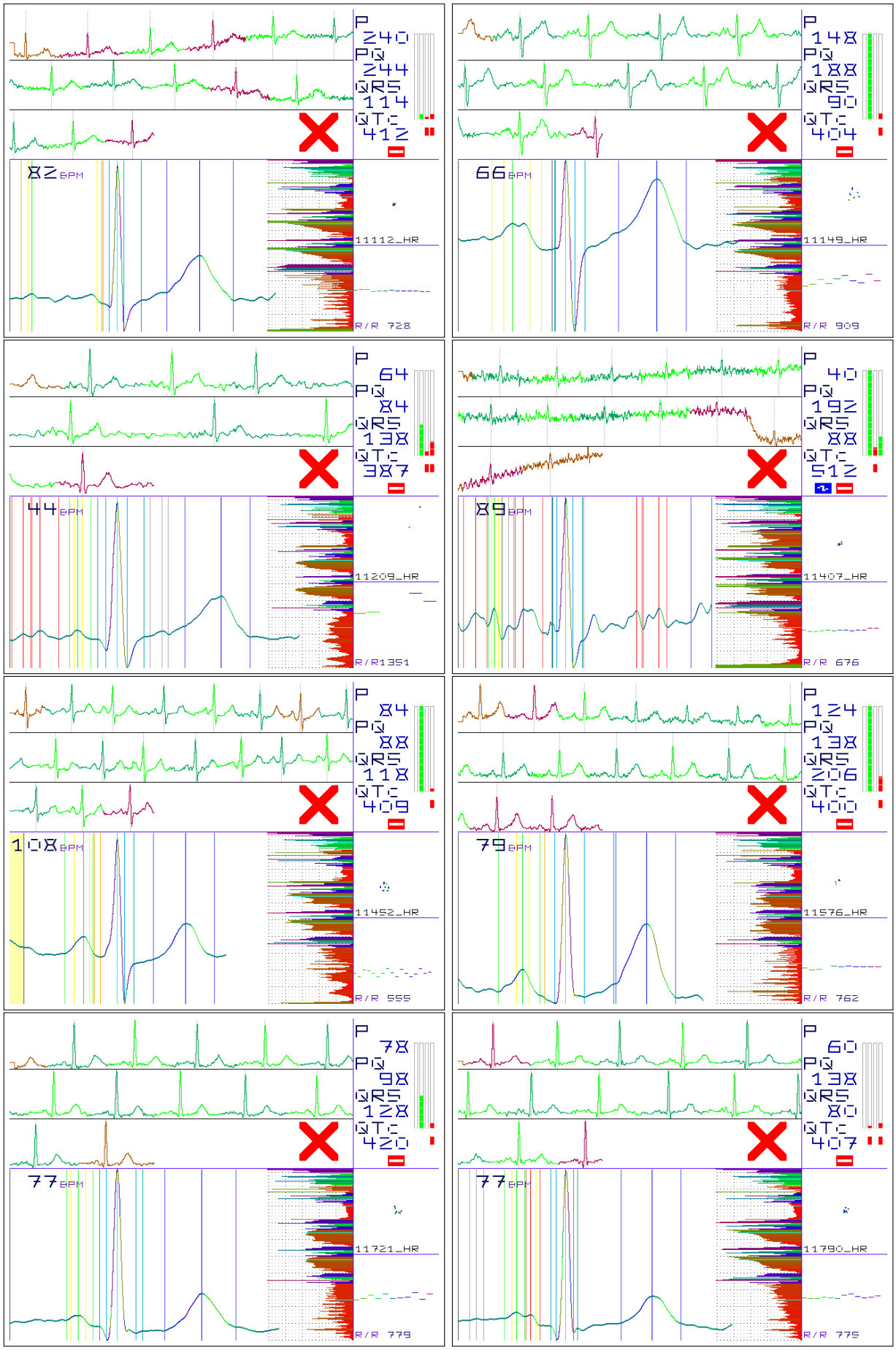


Figure C.12: NSR in the PTB-XL corpus classified as A-Fib 4/6

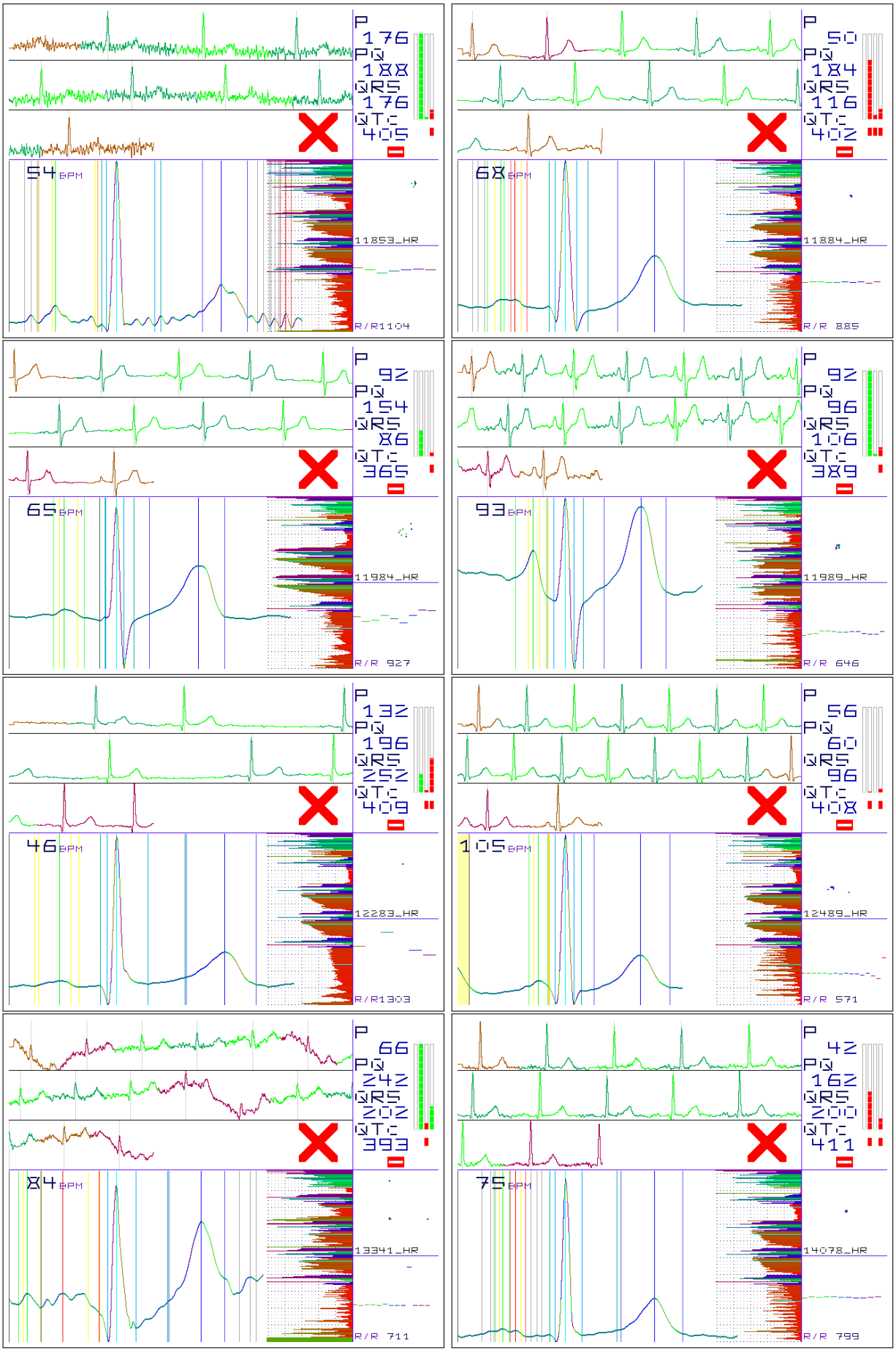


Figure C.13: NSR in the PTB-XL corpus classified as A-Fib 5/6

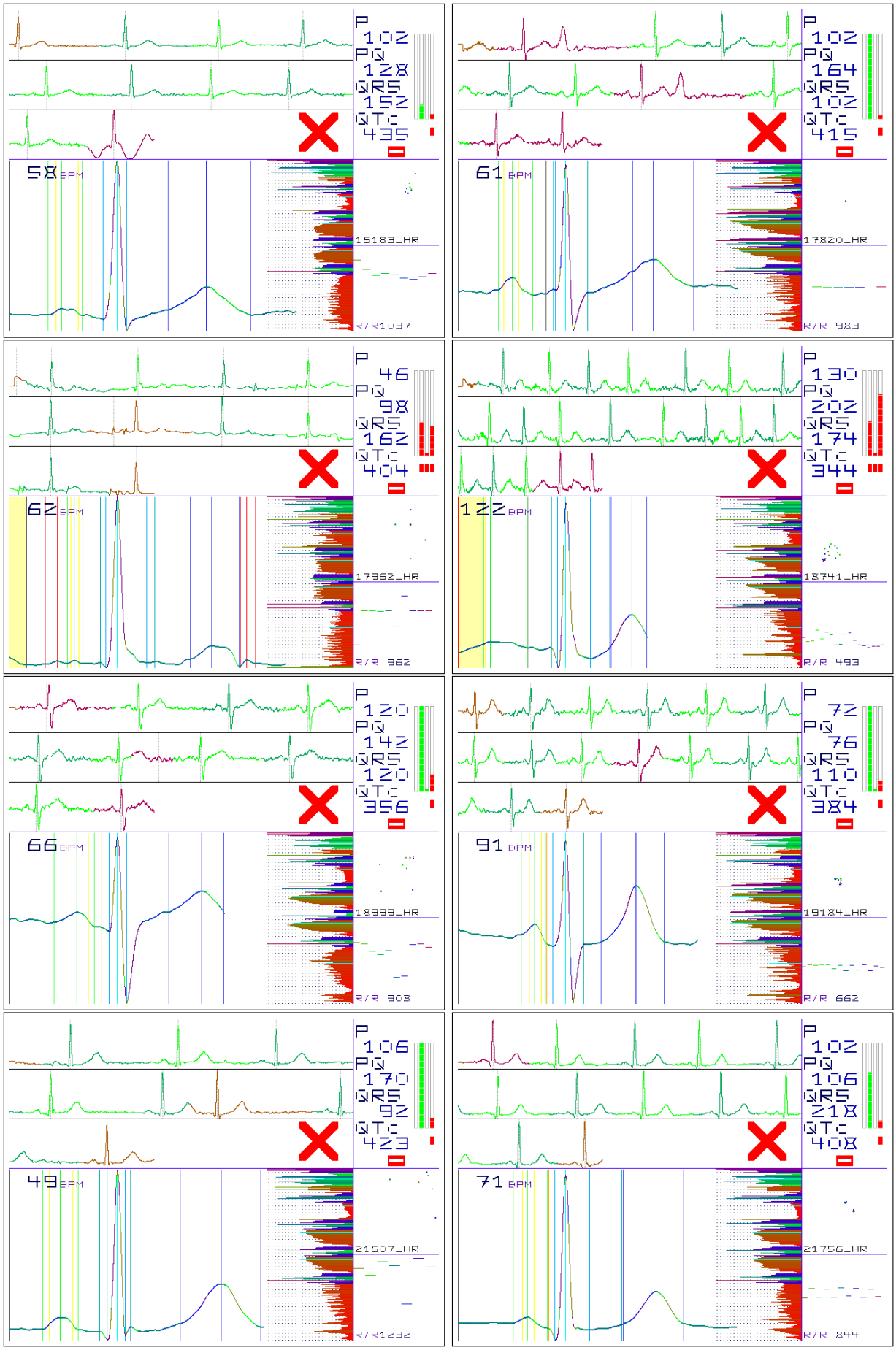


Figure C.14: NSR in the PTB-XL corpus classified as A-Fib 6/6

D Appendix: Licensing of hardware designs and software

D.1 Software licenses

The license of the `ecghelper2` and `offlineecg2` software itself, and of the additional `csvreader` tool for processing data recorded via Bluetooth, is the open source FreeBSD License. This allows the use of the software in a variety of contexts, provided the original copyright is clearly stated:

```
Permission is hereby granted to use this work under the 2-clause BSD License,
also known as Simplified BSD License or FreeBSD License. Contact the author
for other licensing options, such as the MIT License or GNU Public License.

Copyright (c) 2020, Eric Auer <e.auer@jpberlin.de>

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this
   list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
   this list of conditions and the following disclaimer in the documentation
   and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

The spectra are generated using a third-party Fast Hartley Transform implementation. The global spectrum uses at most 4 096 samples in the embedded device and at most 16 384 in the `offlineecg2` implementation, as the latter has more RAM available. For PTB-XL, with 5 000 samples per recording, this does not make a difference, but for some older corpora, it does. The FHT library is adapted from the source code of 'am' version 11.0, which is 'a program for radiative transfer computations at microwave to submillimeter wavelengths.'[61]

This implementation is particularly well-suited for use in this thesis. It lacks most hardware-specific optimisations of many larger alternatives aimed at spectral processing of much longer data vectors. The code provides a good balance between efficiency and compactness, both on Linux (e.g. i386, amd64 or ARM) and on ESP32. The license of 'am' (including its FHT implementation) is:

```
This computer program containing an atmospheric propagation model for the
submillimeter band is a work of the United States and may be used freely,
with attribution and credit to the Smithsonian Astrophysical Observatory.

The program is intended for educational, scholarly or research purposes. In
connection with any commercial use of the program, the user should disclose
clearly and conspicuously all of the information contained in the first
sentence of this notice.
```

[61] `https://doi.org/10.5281/zenodo.640645`

As the new ecghelper2 software can still be used on the old breadboard system, it also still contains the NeoPixel drivers by Marc Falatic, under the open source MIT License (MIT: Massachusetts Institute of Technology): https://github.com/MartyMacGyver/ESP32-Digital-RGB-LED-Drivers

```
MIT License

Copyright (c) 2017 Martin Falatic

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

This library dependency can easily be removed by setting USEREALNEOPIXEL to 0 in neopixel.h, which forces the software to use only discrete RGB LED. Note that to use NeoPixel, the patch esp32_digital_led_lib.cpp.diff provided with the ecghelper2 software has to be applied to the library. This is required to be able to enter a sleep state.

D.2 Hardware licenses

The ecghelper2 and ecghelper3 hardware designs by Eric Auer <e.auer@jpberlin.de> are licensed under the Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) License. To view a copy of the license, visit https://creativecommons.org/licenses/by-sa/4.0/

Part of this hardware design is based on the Adafruit Feather ESP32 HUZZAH32 design by Limor Fried / Ladyada for Adafruit Industries, which is licensed under the CC BY-SA 3.0 License: https://github.com/adafruit/Adafruit-HUZZAH32-ESP32-Feather-PCB

You are free to:

- **Share**: copy and redistribute the material in any medium or format
- **Adapt**: remix, transform, and build upon the material for any purpose, even commercially.

Under the following terms:

- **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **ShareAlike**: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
- **No additional restrictions**: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.