Final Report

Project #36
Software development for a wireless electrocardiogram

Date: 25.05.2018

Antti-Juho Nieminen
Miri Piironen
Alexander von Numers
Jan von Steuben

Information page
Students
Anti-Juho Nieminen
Miri Piirainen
Alexander von Numers
Jan von Steuben

Project manager
Jan von Steuben

Official Instructor
Ilkka Laakso

Other advisors
Raimo Sepponen, Lauri Palva

Starting date
4.1.2018

Completion date
24.5.2018

Approval
The Instructor has accepted the final version of this document
Date: 25.5.2018
Abstract
The original target of the project was to develop a PC software for representing an electrocardiogram (ECG) that was measured by existing hardware from Taipei Trading CO LTD. The software delivered with the ECG had usability challenges that we were to overcome and make an intuitive interface. The developed system will then be used for a hands-on practice in the biomedical education program. As that hardware had some serious issues related to measurement performance, signal quality, robustness and safety, it was decided to develop a completely new hardware that would complement the revised PC software. The existing equipment used ZigBee protocol for communication and needed an additional dongle was needed to communicate with PC. By using Bluetooth Low Energy (BLE) as the new communication protocol, the need for a ZigBee dongle is removed, as Bluetooth is widely supported by many computers natively.

The PC software should be operational on all three major operating systems, Linux, Windows and macOS. This is required to enable all of the students to use the equipment from their personal laptops. The system is used for educational purposes and the source code and documentation is openly available, so that it can be developed further.

The final result fell short from the original plan as unforeseen challenges were faced. The team had challenges with the tools, drivers, underestimating the complexity of accessing the resources of different operating systems and the development of a real time embedded operating system for the created hardware. This then led to reduced functionality and in the end only Linux and macOS platforms were supported by the software. Furthermore, the new measurement equipment only has the revision A prototype and it was not properly tested due to time restrictions. As the product is still in its prototype stage the option for a fourth electrode still remains, as we were initially not sure if the signal quality with two or three leads would be sufficient.
# Table of Contents

1. Introduction 5
2. Objective 5
3. Project plan 5
4. PC-Software 6
   4.1. Graphical User Interface (GUI) 6
   4.2. Operating system resource access 8
5. Measurement Module 9
   5.1. Electronics 9
   5.2. Embedded software 10
6. Issues and setbacks 10
7. Reflection of the Project 11
   7.1. Reaching objective 12
   7.2. Timetable 12
   7.3. Risk analysis 12
   7.4. Project Meetings 13
   7.5. Quality management 13
   7.6. Next steps 13
      7.6.1.PC software 13
      7.6.2 Embedded software 13
      7.6.3 Measurement module 13
8. Discussion and Conclusions 14
   8.1 Antti-Juho Nieminen 14
   8.2 Miri Piironen 14
   8.3 Alexander von Numers 14
   8.4 Jan von Steuben 15
9. Links 15
1. Introduction

The Taipei Trading CO LTD True Sense kit, ECG system currently used in the biomedical lectures has significant issues related to the usability, robustness, performance and safety. The original target was to develop a graphical PC interface for the existing hardware. This was necessary as the current system’s user interface has no labeling, and is not very intuitive. After evaluating the feedback, it was a group decision to redesign the whole system, meaning the measurement hardware, embedded software and the PC software. By doing so, we decided to use Bluetooth Low Energy (BLE), which is natively available on all modern laptops, and remove the need for a wireless dongle. In case BLE is not available on the PC itself, a BLE dongle can still be used. Furthermore, this enables the use with mobile devices too.

2. Objective

The target use of this product is for educational purposes in the biomedical field and used to measure, record and visualize electrocardiograms. The expected result is a portable, user friendly product, with an easy to use interface and support of 3 to 5 electrodes to be connected. Furthermore, the measurement data shall be stored to a edf file for further processing. Moreover, the product must be wireless for an easy setup of experiments, comply with regulatory requirements related to patient safety (IEC60601) as well as the European radio directive (RED). European regulations are sufficient, as this is only used at Aalto University for educational purposes. The software needs to be designed so that the patient is not recognizable from the gathered data and that the software is safe for the host computer.

The final product should include an intuitive GUI for connecting the measurement modules to a PC over a wireless connection, display the information in graphical form on the screen for multiple electrodes and record the data. In addition, the software implementation should include different filters to reduce the noise on the measured signals, also removing the 50Hz mains humming that is present in the current system.

The implementation of the new hardware, will be based on the Texas Instrument’s TIDA00096 demo board. Furthermore, safety will be improved compared to the existing system and standard electrodes will be used. The application software for the PC is designed in the project.

There should be at least 20 measurement modules assembled, supporting 3 to 5 electrodes each (support for 2 electrodes would be nice for a simpler test setup, but if the hardware does not support this, 3 electrodes are accepted). The hardware should function for a minimum of 1h, with 48h preferred. Primary batteries (non-rechargeable) will be used. Data logging is not required if all necessary data can be sent over a wireless link in real time.

3. Project plan

The project plan divided the task into several milestones. These included also deadlines for documents and presentations given by the course instructors. Due to this, some phases could not receive the attention they would have required, as for example the pre-study. The deadline for the project plan forced us to decide earlier the path of the project. Business aspects were a nice exercise, but as the project is aiming for openness and educational purposes, a lot of time was used to switch the approach of the project to a business case, only for the presentation.

Milestones were set in accordance with independent tasks and possible deliverables. Learning of tools and tasks was taking into consideration. The tasks included the design of the GUI, access of
the Bluetooth low energy in the PC, hardware design, embedded software as well as testing with potential users, to mention a few.

The project schedule was made with a common understanding and it was believed it could be achieved. On hindsight, the plan was too ambitious for our skill set. Risks were evaluated and two of them came true, which lead at the end to not achieving our set target. One of the challenges is that this is a school environment, and not all the necessary support was available, so that plenty of things had to be learned the hard way. We did try to use our network to get some support, but as there is no real business behind it, the support provided by suppliers and manufacturers is minimal.

For a detailed view of the project plan, refer to the project plan document under the project page website. The link to the page can be found in section 9, Links.

4. PC-Software

4.1. Graphical User Interface (GUI)

The PC Software was designed to achieve ease of use, maximum informativeness, maximal reliability, and cross-platform support. The GUI and UX were designed by using browser-operated graphical design tool Figma. The Interface consists of three voltage measurement plot views and their axis (seconds, mV) readjustment fields, a button to connect the PC with the measurement module, a button to export the measurement data with drop-down menu for filetype selection (.csv or .edf), a button for starting the measurement and data logging, and a bottom bar for displaying additional important information. The connection button and the measurement start button will change their labels to indicate the current operation (NOT CONNECTED/CONNECTED and START/STOP). The bottom bar displayed (from left to right) the name of the device, the battery level, the level of connection from 1 to 5, the duration of the latest or current measurement, and the timestamp of the moment where the last measurement started.

List of the GUI’s specifications:
- Size of the full window (W x H): 920 x 700. pixels
  - background color: #9BC1BC
- voltage measurement plot views (not implemented):
  - size of the plot (W x H): 750 x 140 pixels
  - background color: #EEEEEE
  - Axis and grid color: #000000
  - plot line color: #E41212
  - Location of the three plots by the paddings of Left, Top, Right, and Bottom sides (pixels)
    - TOP: L: 26, T: 40, R: 144, B: 520
    - MIDDLE: L: 26, T195, R: 144, B: 365
    - BOTTOM: L: 26, T: 350, R: 144, B:210
- numeric value input field of plot axes (not implemented):
  - input type: numeric values
  - background color: #EEEEEE
  - text color: #000000
  - grid color: #C7CAD1 (grid not implemented)
  - size of all of them (W x H): 67 x 26 pixels
  - Locations by the paddings of Left, Top, Right, and Bottom sides (pixels)
    - TOP plot
      - Voltage (“mV”): L: 793, T: 84, R: 60, B: 590
      - Time “seconds”: L: 793, T: 130, R: 60, B: 544
The graphic design of the GUI may include also visual effects, for example shadows within buttons (visualized in the Figure X, but not implemented).

The PC software was implemented using Python (9.2) 3.6.5 and following libraries:
- Kivy (9.3) (for GUI design implementation)
- Pygame (9.4)
- PyBluez (0.22)

The version control software used for collaborative code implementation was Git (9.5), and it was hosted in GitLab 9.6) provided by Aalto University.
4.2. Operating system resource access

The basic functions for the user interface to work is documented in the appendix “Functions in easyGee software. The first class includes the functions related to the Bluetooth connection. This includes connection strength, checking if a connection has been established and a function that searches for available devices. The Bluetooth class uses the PyBluez library to access Bluetooth resources with python. The other class includes data handling and exporting the data. Starting and stopping the measurement, which includes the data gathering, and the exporting the file with different file options. With the use of these functions the functionality of the user interface should be able to run basic operations to start with.

By using the function in the Kivy App Life Cycle we would have been able to develop a working interface for the board. By using the functions to display vital information and gather the data, the environment should have been able to run.

Since there was a lot of bumps in the road, that will be presented in chapter 6, the Bluetooth compatibilities haven’t been tested.
5. Measurement Module

5.1. Electronics

The schematic and layout were drawn in Mentor Graphics Pads VX2.2. The complete set of design files can be found on the project webpage, as native Pads files, PDF and as an ascii file format. This should make it possible to continue the design in another tool as well. Gerber files for the printed circuit board, layout drawing, assembly coordinates and bill of material are included.

The design is based heavily on Texas Instruments’ (TI) ADS1293 development board. The ADS1293 is an analog to digital converter (ADC) designed in particular for ECG applications. The ADC includes EMI filtering, 50/60 Hz rejection and a SPI bus for communication with the embedded processor. For communication between the measurement module and the PC, we used the CC2650MODA module from TI, to speed up the design process. That module contained the CC2650 micro controller (mcu) with the radio frequency (RF) circuit build in. The mcu has 2 parts, an Arm Cortex M3 for main processing and an Arm Cortex M0 for the radio processor. Furthermore, the module included a chip antenna and the RF-matching for the said antenna. Because of the high grade of integration, the RF module complies with European RF regulations and in addition has the radio certification for North America.

The device is powered by two AAA alkaline batteries in series, which results in a varying supply voltage from 1.8 to 3.6V, depending on the state of charge (SOC), we used the boost regulator TPS61220, to guarantee a minimum supply voltage of 3V for proper operation of the circuit. The regulator has an adjustable output, which can be set by changing the values of resistors R2 and R3. Ferrite beads were used in the supply for the RF-module and the ADC to filter high frequency. The layout is a simple 2-layer board. Unfortunately, the CC2650MODA as well as the ADS1293 are not easy to solder by hand, so that an assembly service for these two components was utilized. For programming and debugging, a 20 pin JTAG is utilized.

![Figure 2. easyGee schematic](image-url)
Due to unfamiliarity of the tools for the design of the logic diagram and layout, the roles were switched after seven weeks, which caused a delay in getting the first prototype. In addition, the switch of roles then caused delays in the embedded software development as described in the next chapter.

5.2. Embedded software

The following part explains how the embedded software was implemented. In the case of Texas Instruments’ Bluetooth chipsets, the usage of TI-RTOS is needed as it makes the development process much smoother. The TI-RTOS adds software components implemented by the engineers at TI and includes the Bluetooth stack. These components provide services via an API and they can be programmatically configured to fit the device specific needs.

The project was built upon an existing example that was acquired through the Resource Explorer inside the CCS. The example projects function as framework which the software implementation can be developed upon. To create a functional embedded application, it is required to know at least the basics of the Bluetooth protocol and to have a fairly good understanding of C language. Also the developer needs to follow the reference materials to know what sort of functions are provided by the TI-RTOS.

6. Issues and setbacks

6.1. Python libraries and virtualbox

The setup of the programming environment was a hassle to begin with. Since we decided to use Linux operating system for programming that enabled easy download of the libraries, one person had to use the virtualbox for a virtual Linux machine. The problems arose when the Bluetooth
adapter had to be enabled in the virtual machine. The code utilizing the bluetooth tools didn’t work properly, which lead to troubleshooting and trying to figure out why it didn’t work. Since we didn’t find a solution to the problem, we decided to use a school computer. Since you don’t have “sudo” user allowance, the libraries caused a bit of a problem. Fortunately, the service desk helped with the main libraries that needed “sudo” user allowance. As the desk computers don’t have bluetooth as a standard feature, we had to buy a bluetooth dongle for the linux environment. The problem was that the school computer couldn’t find the dongle and therefore we loaned a computer from Ilkka, our instructor.

Unfortunately, all the libraries could be installed on the computer we loaned from Ilkka. We didn’t get far in the troubleshooting, because of time limitations. All in all, we hope that these problems don’t arise for the next group that picks up the project and that the group knows about related problems with the installation of the libraries that we used.

6.2. Windows BLE and Launchpad

This has more with inexperience to do than an issue, but still worth mentioning. We used a CC2650 LaunchPad by Texas Instruments for overall testing and development. The launchpad worked well with the phone app, but when the testing started to move over to the PC side, there was problems with using the Texas Instruments software and overall communication with the device.

Here again I have to stress that this is most likely user inexperience more than troubleshooting, but still worth mentioning.

6.3. Code Composer Studio and TI-RTOS

The team didn’t have anyone who was familiar with embedded programming and we couldn’t find anyone who was experienced with the TI-RTOS or the TI’s tools that we needed. There are vast amounts of material available through the TI’s online resources but still getting a grasp of the bluetooth development was too big of a hurdle to overcome. One problem was that it is very hard to estimate how much studying is needed when the whole Bluetooth and embedded ecosystem is something that most of us had no previous experience with. There are so many components to the bluetooth development that it was hard to understand what is actually needed for our purposes.

Most of the time on the embedded software development was spent reading user’s guides and documentations about the ecosystem. And in the end, we ended up not getting a working application for the custom designed board we had made.

6.4. Design of the custom board in the beginning

The creation of the board layout and schematic files was first given as a learning exercise to a team member that had no previous experience in electronics design. The learning of the new tools took much longer than anticipated. There was also an incident with corruption in the design files and the process had to be started all over. As the design process was not advancing in a fast enough pace to meet the deadlines we decided to change the roles of the embedded software designer and electronics designer. The new roles reflected the members backgrounds and experience much more closely than the previous tasks but some valuable time was lost in the process.

7. Reflection of the Project

When we started the project, we evaluated the feedback on the existing system. The task given and the set of skills were not on par, as most of the group members didn’t have any background in software development. The lack of skills and unfamiliarity with the tools caused significant delays, and still at the very end we struggled with a compatibility of tools, libraries and operating systems. In the beginning, we were able to stick to the plan, but then slowly slipped. The problems were known and we thought we can solve them but it was not long before the next issue came up. Accessing the resources in the PC was causing further troubles. Switching the responsibilities after 7 weeks between electronics design and embedded software was not helping with achieving the
goals. In addition, as this was a school project, without any real business value, the support than can normally be gotten from IC and tool manufacturers was not available at the same level as it normally would be.

7.1. Reaching objective

We did not achieve the full objectives. The target was to have the PC software operating on all three major operating systems (Windows, Linux and iOS). Even though we narrowed the scope for Linux only we did not manage to get it to work. For the GUI Python with Kivy library was chosen for the design of the PC software, as it is portable and can be used easily on the three operating systems. Accessing the hardware components like the BLE caused significant troubles that lead to delays, so that we have not been able to pair the CC2650 Launchpad with the Pc either. The Launchpad was used so the development could be done independent of the state of the measurement module hardware or embedded software.

One prototype revision for the measurement module has been made, but without the embedded software, the functionality and performance could not be verified.

In general, as we are all in the automation and electrical engineering program with major in translational engineering, the skill set for a mainly software project was not given. The effort to learn the tools without much support took significantly longer than our schedule allowed.

7.2. Timetable

The original time table did work for the design, but not for the implementation. There has been a constant trouble with tools, or libraries, which was not anticipated. Accessing resources in a PC, like the BLE, turned out to be far more challenging than expected.

Workload estimates where off as the learning of the tools and even the basic understanding of e.g. software structure took essentially more time then the 40h allocated per person and caused the time table to slip. Due to that, roles were changed between embedded software development and electronics design, to catch up with the delay that occurred at that time.

![Figure 4. Project schedule](image)

7.3. Risk analysis

Two of the identified risks appeared. The knowledge gap of the tools was a much larger problem than anticipated, so that there was a significant amount of time required to achieve the tasks. The impact was underestimated. This unfamiliarity also caused us not to see the challenges with the
software libraries and incompatibilities of the development environments. Due to that, another risk realized, so that a group member was not able to complete his task, which then lead to a change of roles, and caused even more delay for the project as the learning curve started again. This was not recognized early enough, which might have reduced some of the risk.

7.4. Project Meetings
We had scheduled weekly meetings with the instructor. Meeting minutes were kept and stored to our google drive folder. In addition to these regular meetings, one meeting with additional advisors, Raimo Sepponen and Lauri Palva was kept to gain better knowledge of the regulatory requirements and also how to approach the measurement of the ECG signals. Small workshops and other meetings were held as needed, and the team members met independently based on needs. At the end we had a “lessons learnt” session to evaluate the project and what went well and what could have been done in a better way.

7.5. Quality management
For a quality approach of the development, peer reviews were planned and utilized in the beginning, but as the tasks drifted more apart, the review approach was not anymore utilized. Trello was the tool of choice for schedule monitoring, but was not used sufficiently by all four members. No new features were added, and we tried to limit the functionality and supported operating systems to have at least some output from the project but also that was not achieved.

7.6. Next steps
As the project could not be finalized, here are things that can be done to finalize it and provide a tool for the Biomedical engineering lectures.

7.6.1. PC software
The Bluetooth accessibility needs to be solved first, as it is the critical part of the whole system. The full GUI needs to be built also. The functions working with the GUI should be bound to the correct components.

7.6.2 Embedded software
The development process should start with studying the TI-RTOS’ user’s guide and getting familiar with the Bluetooth protocol and stack by trying out the example projects and understanding them. The GATT profiles also need to be implemented to enable the Bluetooth services. For the development, Texas instruments’ Code Composer Studio (CCS) 7.4.0 and the TI-RTOS can be used. The CCS contains also examples for the Bluetooth and peripherals. The tools, RTOS and examples are freely available. For programming and debugging, Segger’s J-Link EDU was used, and can be found at the instructor’s office. To start, the LED’s on the measurement module can be used. To test the Bluetooth, TI’s stack can be used. To test the functionality independent of PC software, TI’s SimpleLink phone app could be utilized to see the if the device will be visible and connect. In order to configure the ADC, the SPI bus must be configured. For the ADC configuration, there is an example available in the ADS1293 manual. Some familiarity with an embedded real-time operating system will be of great advantage for the development.

7.6.3 Measurement module
When embedded software is available, the measurement module should be checked for the performance. Can the number of electrodes be reduced? Currently there is support for four electrodes to provide a good common mode reference. If possible, the measurement module could be taken to an EMC laboratory to see how well the measurement will behave when exposed to
external interference. As the electrode wires are below 3m and the device is fully battery operated, the only tests that are required are radiated immunity and emissions. ESD should be performed once an enclosure has been designed.

One comment was related to the use of an audio-jack as connector for the high impedance electrodes. One reason to go for this connector to reduce the size as the connector is significantly smaller than 3 banana plugs, which affects the form factor. The product is in general rather large due to the AAA batteries.

It should be tested to see if the 50/60Hz interference causes noise in the measurement. This can be filtered also in the ADC but will reduce the sampling rate. Depending on the findings, the number of electrodes might be reduced to 2. This should be taken into account and a new revision can be designed. If there are too many issues with immunity to an RF field (EMC test), using four layers instead of 2 might be beneficial.

Because the measurement module is untested, and therefore the formfactor unknown, an enclosure could be designed and printed to improve the robustness and reliability (reduced risk of ESD).

8. Discussion and Conclusions

8.1 Antti-Juho Nieminen

During this course, new insights of software development were arised. The process of software design was highly unfamiliar with me at first, but it got clearer as the course proceeded. Also, I learnt a lot about project type of work (thanks to our highly experienced Project Manager) and collaborative technological teamwork. The subject was about measuring different aspects of heartbeat with electrocardiogram from which I got high amount of learning experiences.

The course had many issues in general, but our team and our instructor made the course work for us satisfactorily. Our work was organized and we reacted to unexpected surprises effectively. Despite our effort and commitment, we ran out of time, but our vision was clear enough to leave the work to be continued by other group with high quality potential results.

8.2 Miri Piirinen

The topic of the project was very intriguing and I learned a lot about circuit design, embedded software and bluetooth. This was the first hands-on experience with embedded systems I have had even though I am already doing my Master’s degree in the field of embedded systems. The team dynamic and the amount of work everyone was ready to put in was astounding. The team spirit stayed on the positive note even at times when nothing seemed to work.

I feel I have gained more knowledge about electronics and embedded systems during this project than what I have learned through the time I have studied here at Aalto.

8.3 Alexander von Numers

The project course has given me a lot of experience and knowledge. Not only teamwork and communication, but also a time scheduling and problem solving. Since our group felt very warm and inspiring, it is a shame that so much went south. Our project leader did a fantastic job with meetings, problem solving and asking for help outside the university. As this was really a hands-on experience in product development, it served as a good indicator of what to expect in the future.

We may have been too ambitious, not only in development but also in skill required for the work, but you live and learn. Hopefully we have created a base for a great device and user interface, that can be developed by a future project group.
8.4 Jan von Steuben

The course has been giving me some insight into project management, as I am usually in the designer’s role. It helped me to understand the role better and also the way of how to plan and execute a project. This included the breakdown of tasks into more manageable jerks. Part of this was also to get a group of people with different backgrounds to work on a common goal, and keeping the project going.

Originally it was my task to look at the embedded software and I started to work with TI’s code composer studio, in combination with their real-time operating system (TIRTOS). Due to the project manager duties I had a slow start. Because of my background in electronics design, the task for the measurement module design was transferred to me as we hit the deadline. With the change, I gave the embedded software away. During the design of the measurement module I learned about the way how an ECG is measured.

9. Links

9.1. project web page: https://wiki.aalto.fi/display/AEEproject/Software+development+for+a+wireless+electrocardiogram
9.2. Python: https://www.python.org/
9.5. Git: https://git-scm.com/
9.6. GitLab: https://about.gitlab.com/