UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics and Computer Science

Module 1

Introduction to Electrical Engineering and Electronics

Project Manual

Electrical Engineering

University of Twente

2013/2014

M. Odijk

F. van der Heijden

C. Salm

J.C.T. Eijkel

B. Molenkamp

Enschede

September 2013

Contents

1. Introduction

1.1. Objectives

1.2. Organization

The project consist of 8 **weekly** sessions (chapter 2-9) that prepare for the final project. The **final project** starts with a special session on project design (chapter 10) in week 9. The goal for the final project is listed in chapter 10.

1.3. Reporting

An overview of how a report should be constructed is shown below. Note that it is a guideline and that you are free to make other choices or add some information, if that gives a better report.

- Title page - containing at least: a suitable title, the name and student numbers of all group members, and the hand-in date.
- Introduction – describing the what and why aspects, i.e. the goal (what is required and why)
- Analysis – analyze the relation between the goal specifications and ways or components to satisfy the goal. Also theory about e.g. a sensor or network should be given here.
- Method / Design – Based on the analysis choose the most suitable option and design a concrete implementation
- Evaluation – describing how the performance of each design/method will be tested according to the specifications
- Results – the results of the evaluation of the design
- Discussion – discussing the results and mention which parts worked well and which ones did not (and give an explanation for this).
- Conclusion and recommendations – did you reach the main aims? How good was performance? What are the main limitations and recommendations for improvement?
- References - list of literature sources
- Appendix – additional information, such as for instance the task distribution of the group members, planning of the events, the scripts made for Matlab or the C code used. Also the log book can be given here.

Note 1: There is a clear difference between a report and the lab notebook. The lab notebook allows to track the order of events. For a report that is not so relevant. Instead it focuses on the main considerations and results (looking back at the lab notebook, what are in retrospect the key issues to report?). Still, the structure during experiments in your lab notebook can also be used in a design report with some extensions.

Note 2: Due to the limited time available, weekly reports are limited in size between 2-4 pages (excluding the title page, references and the appendix).

Weekly reports need to be handed in at **Mondays before 10:45** via blackboard.

The mark for the **weekly reports** will be determined on the following criteria (unless stated otherwise):

- Creativity (10%).

- Quality of analysis and solution (30%).
- Quality of design (and/or method) (30%).
- Quality (readability, clarity and structure) of report (30%).

The **final project** will be assessed on the design and readout of the sensors (25%), the report (25%), the presentation (25%) and the process and teamwork (25%). As group you will get the end mark (EM), which will be multiplied by the number of group members (N). The group itself has to divide this total score (N×EM) among the group members via a so-called peer review, deciding on the final mark each group member will get. For this purpose, your group together with a tutor will discuss the participation and contribution of each group member and decide which part of the total score (N×EM) will be for this group member ranging between 1-10.

The final mark for the whole project will be based on 40% of the weighted average of the weekly reports and 60% of the mark for the final project. The project mark accounts for 20% of the whole module. This final mark needs to be 5.5 or higher to pass the module. Two weekly reports are allowed to be 3 or higher.

2. Kick-off project

M. Odijk / T.H. de Kluijver

Teambuilding

This weekly assignment is about teamwork.

Assessment

It is not required to hand in a report. However, you do need to fill in the spaghetti questionnaire before the end of this weeks project session, as the results are required for the first session of Study Skills.

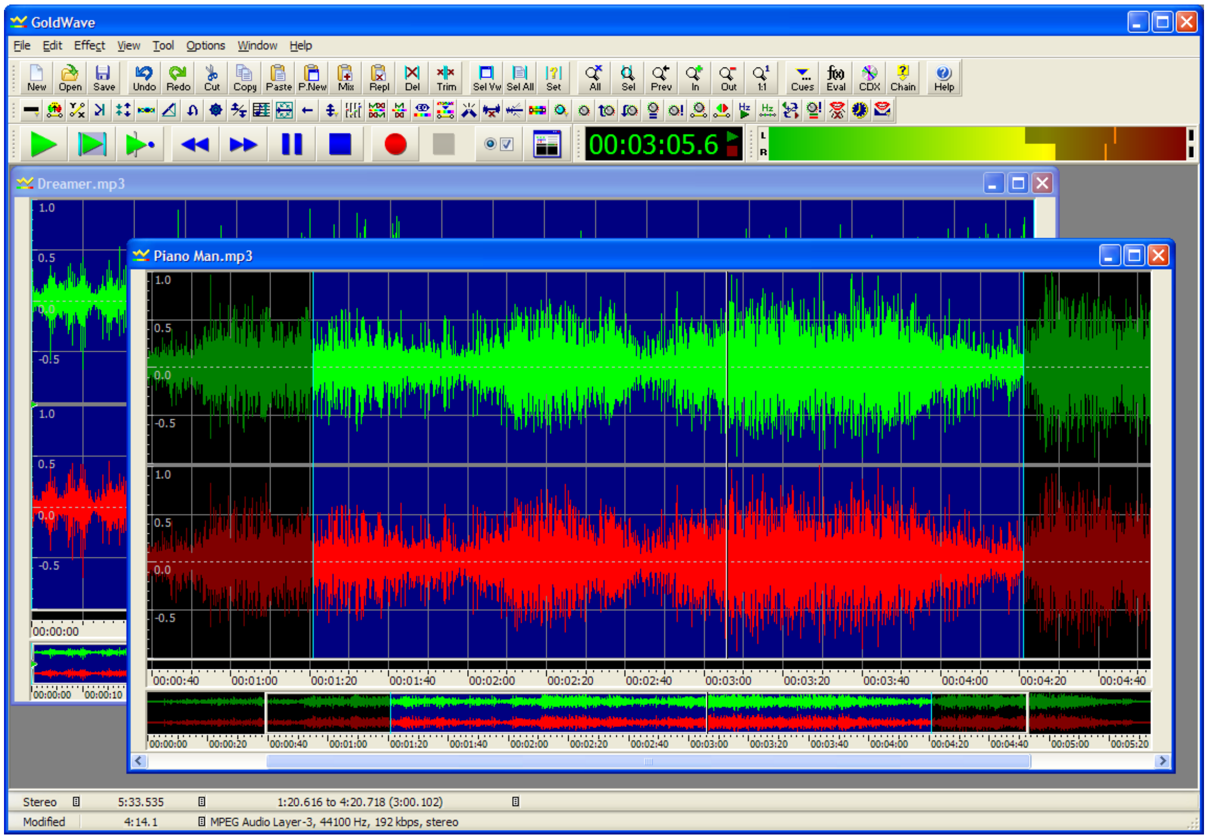
3. Signal editor

3.1. Introduction

A sound editor is a software tool for recording, visualizing, editing, and conversions of digital sound waves. Editing involves various forms of manipulation (*Figure 1*), e.g.:

- volume control
- fade in and fade out
- noise suppression
- selecting, copying, cutting, and pasting sound fragments
- sound effects, e.g. reverberation and echo
- extraction of features (parameters, properties) of the sound

A sound editor is an example of a signal editor. The application area of a signal editor is not limited to sound. It can handle any other sequence of numeric values representing samples taken at a regular interval. Nevertheless, a signal editor is often designed with some application area in mind. Examples are software for recording, visualization and manipulation of ECG signals. Other editors are focussed on EMG signals (Electromyograms; electrical activities at the surface of the skin due to activation of underlying muscles), EEG signals (Electroencephalography electrical activity along the scalp), seismological data, and so on.


GoldWave
File Edit Effect View Tool Options Window Help
New Open Save Undo Redo Cut Copy Paste P.New Mix Repl Del Trim Sel Vw Sel All Set All Sel Prev In Out 1:1 Cues Eval CDX Chain Help
00:03:05.6
Dreamer.mp3
Piano Man.mp3
Stereo
5:33.535
1:20.616 to 4:20.718 (3:00.102)
Modified
4:14.1
MPEG Audio Layer-3, 44100 Hz, 192 kbps, stereo
Figure 1 A sound editor   (source: http://www.goldwave.com)

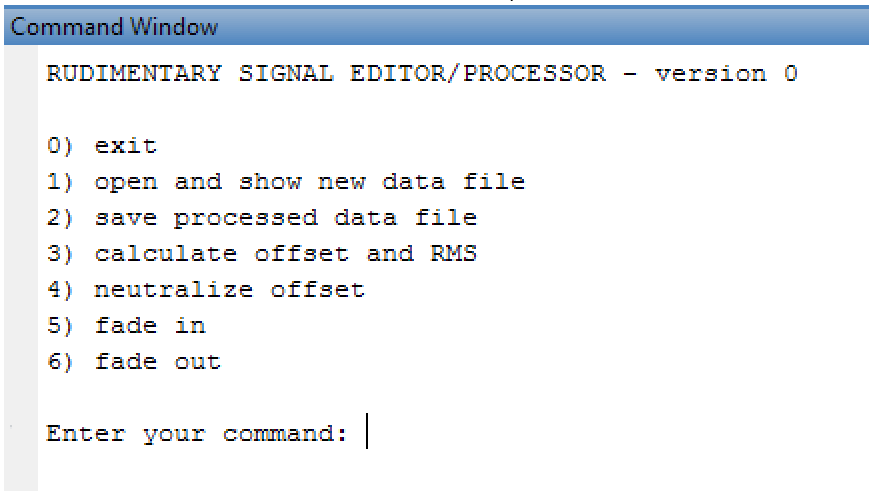
3.2. Problem statement

The aim of this weekly project is to develop a basic framework for a signal editor. The framework will be as simple as possible, but still embedding the essential functionality needed for basic signal editing. The word "simple", here, means:

1. That the user interface will not be graphical (as in the example of *Figure 1*). Instead the user interface will be text-based. See *Figure 2*.
2. That the offered functionality is rather limited. The minimal functionality consists of the following items:
    - A facility for reading signals that are recorded and stored in files, i.e. opening files
    - A facility for writing signals that are manipulated to files, i.e. saving files.
    - A facility for visualizing signals
    - A facility for calculating the offset of the signal (offset = mean), and neutralize this offset. That is, adding a constant to the signals such that the offset becomes zero.
    - A facility for calculating the RMS of the signal, and to normalize the RMS to unity. That is, multiplying the signal with a constant such that the RMS becomes one.

    These are the minimal set of functions. However, you are invited to extend the functionalities with facilities of your own choice. Examples are:

    - Applying a fade-in and a fade-out of the signal, so that the start and the end of the signal decay smoothly to zero.
    - Calculating the minimum and/or maximum of the signal.
    - Calculating the point in time at which the minimum or maximum occur.
    - Selecting a part of the signal which then can be deleted (so that the duration of the signal becomes smaller).
    - Copying and pasting a selected part of the signal.
    - Etc.
3. Data formats are limited to sound, i.e. wav-files.



```
RUDIMENTARY SIGNAL EDITOR/PROCESSOR - version 0

0) exit
1) open and show new data file
2) save processed data file
3) calculate offset and RMS
4) neutralize offset
5) fade in
6) fade out

Enter your command:
```

Figure 2 A text-based user interface

In a follow-up project, the tool will be developed further. For instance, the functionality will be expanded. Other data formats will be introduced. If possible, the user interface will be replaced by a graphical user interface. (Matlab has tools for creating graphical user interfaces; so-called GUI's)

Questions with respect to the design in the current project are:

- How to implement these different functions in Matlab?
- How to implement the user interface in Matlab?
- How to integrate user interface and the functions into a working tool?
- Which additional function would be eligible for implementation (if any?)
- How to test the tool?

In order to streamline the development, some example matlab code is already provided. This can be used as a template for further development. The software is available in the following m-files:

• `textbased_ui_template.m`	the main script that implements the UI
• `opendat_file.m`	a function for opening files
• `savedat_file.m`	a function for saving data
• `empty_func.m`	a function that does not nothing

Install these files in a folder and run the m-file `textbased_ui_template.m`. You then invoke the user interface in the command window, and you can start entering commands.

3.3. Assessment

You have to hand a report, your code of the signal editor, and the test data, in a single zip archive. This should be done via Blackboard before Monday 10:45 AM.

The report (pdf format) should mention:

- The functionality that has been implemented. Describe the functions that implement signal processing in mathematical terms. (introduce variables and describe the processing as a mathematical equation)
- The method that you have used to validate you code (test), the type of test data that you used, and the results of that validation.
- As an appendix: the division of tasks (which function is implemented by who; who did the system integration; who did the testing, the report writing, and so on).

The code consists of a script, and a few functions. They should be placed in one folder together with the test data, such that running the script is sufficient to start up the signal editor.

Assessment will be based on the following criteria:

- Creativity, e.g. what additional functionality did you implement. (33%)
- Quality of design (is the Matlab code well and efficient written? And is the functionality correctly described in the report?) (33%)
- Quality (readability, clarity and structure) of report (33%).

In other to pass, your final code should immediately work without any modification or correction.

We will continue with this code in week 4.

4. Energy for portable sensor systems

J.C.T. Eijkel and Cora Salm – duration 4 hours

4.1. Introduction

In our portable sensor systems, for example for sports applications, we will need a portable supply of energy. For this purpose we will use a battery pack. The pack must enable stand-alone and in-the-field operation.

Batteries convert energy that is stored in the chemical domain to electrical energy. They come in different types, whereby the chemistry that goes on inside the battery gives it its name (e.g. NiCd batteries). Depending on this chemistry, a single cell of the battery gives a certain typical voltage. Some types of chemistry furthermore allow recharging. Batteries also come in different capacities. The discharge curve can be used to characterize a battery, whereby every battery type has a different discharge curve.

4.2. Problem statement

A common problem with batteries is, that we don't know their State Of Charge (SOC): are they still full or almost empty? A further problem that is specific for rechargeable batteries is, that their capacity decreases on prolonged use. In this assignment you will design a battery tester to assess battery SOC.

4.3. Assessment

Write a report following the guidelines handed in section 1.3.

Herein:

1) From literature or the web give an overview of the most common battery types and their main characteristics.

2) Give the design for a battery pack that can power sensors and microcontroller in a portable system for four hours under typical operating conditions. Give details on the batteries and the circuit via which they are connected. Motivate your choice. For the battery type(s) chosen provide a typical discharge curve(s).

3) Describe all the relevant factors that influence the discharge curve of the battery pack you chose.

4) Design a battery tester to assess the SOC of the batteries/battery pack that you chose under 1). Give the electrical circuit of the tester. Motivate your choice of circuit and components and be aware that if you produce a complicated circuit you will have to be able to explain it well to get your marks! Describe in detail how you will test the pack with this tester.

5. EMG processing

F. van der Heijden – duration 2x4 hours

5.1. Introduction

Often, sensory input is not directly interpretable without processing. As an example, we will have a look at sEMG signals in this weeks project.

"Surface Electromyography" (sEMG) is the measurement of electrical activities at the skin. These activities, so called myoelectric signals, are due to the activation of the underlying muscles. An example of an EMG signal is shown in Figure 3. Here, 16 electrodes are attached to the face measuring activities of the facial musculature. Shown in the graph is one of the measured EMGs. In this set-up, the motion of the face is synchronously recorded by a two video cameras. Markers (small black ink spots) are placed around the lip. These markers can be tracked in the video. By combining the two videos, the 3D positions of the markers can be reconstructed. Figure 4 shows the measured y-position (vertical position) of the center of the lower lip.

5.2. Problem statement

The statement of the problem in this project:

To what extent is the given EMG signal a predictor for the lower lip position?

In other words: suppose that we know the EMG signal. Can we use that signal to find the lip position? An elaboration of this problem leads to the following subproblems:

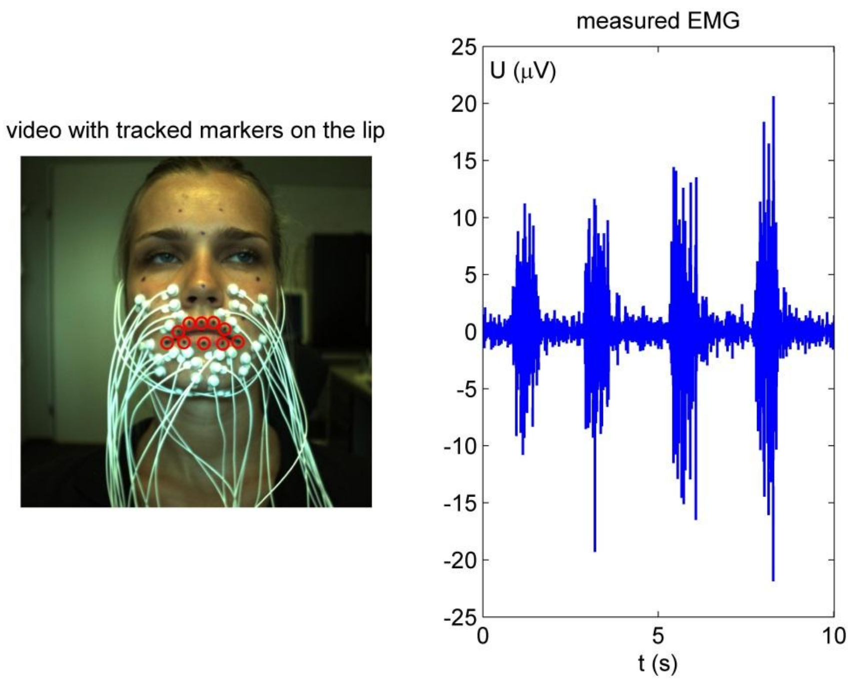


Figure 3 EMG recording

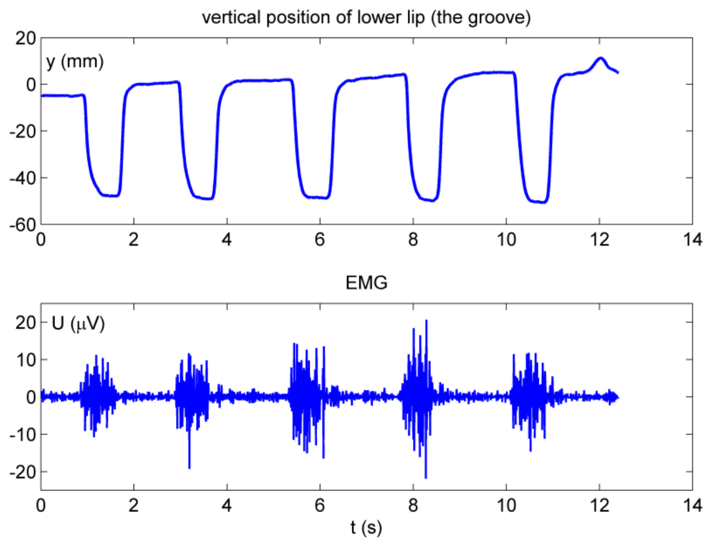


Figure 4 A recorded EMG together with the measured position of the lower lip

- Is the EMG signal directly a predictor for the position without much signal processing?
- If not, what kind of manipulation on the EMG signal is useful to predict the position?
- What kind of relationship exists between the EMG signal (or the manipulated version) and the lip position, e.g. linear, nonlinear, a deterministic one or a more random one? Or does such a relationship not exist? How do we assess such a relationship?

The manipulated EMG signal is called an “EMG feature”. The manipulation itself is called “feature extraction”. In literature, many different EMG features are described. So, if feature extraction is needed, choices need to be made which one. These feature extractors **must be** embedded in the signal editor that was designed in project 2. The emg signal is available in the file “emg.wav”. The vertical lip position can be loaded from the file “lippos.mat”.

The work **must** be structured in two stages:

Stage 1 (morning session):
- Orientation of the problem
- Interpretation of the problem
- Exploration of the possible solutions
- Project plan

Stage 2 (afternoon session):
- Implementation of the different parts in Matlab
- Integration of the different parts
- Analysis and interpretation of the results
- Final report

The project plan is a short intermediate report which includes:
- The feature extractors that are going to be implemented.
- A mathematical definition of these feature extractors

- How the relationship between an EMG feature and the lip position is going to be evaluated
- A division of tasks

Mandatory: the project plan must be finished and checked by the assistants before the actual matlab implementation starts. It is allowed to copy (parts of) the project plan into your report.

5.3. Assessment

You have to hand a report, your code of the signal editor, and the test data, in a single zip archive. This should be done via Blackboard before Monday 10:45 AM. The report (pdf format) should mention:

- The functionality that has been implemented. Describe the functions that implement signal processing in mathematical terms. (introduce variables and describe the processing as a mathematical equation, possibly complemented by pseudocode)
- The method that you have used to validate you code (test), the type of test data that you used, and the results of that validation.
- As an appendix: the division of tasks (which function is implemented by who; who did the system integration; who did the testing, the report writing, and so on).

The code consists of a script, and a few functions. They should be placed in one folder together with the test data, such that running the script is sufficient to start up the signal editor.

Assessment will be based on the following criteria:

- Creativity, e.g. what additional functionality did you implement. (33%)
- Quality of design (is the Matlab code well and efficient written? And is the functionality correctly described in the report?) (33%)
- Quality (readability, clarity and structure) of report (33%).

In other to pass, your final code should immediately work without any modification or correction.

Files available from blackboard:

emg.wav:	emg signal	Sampling period $\Delta = 0.01$ s	Units: mV
lippos.mat	y position	Sampling period $\Delta = 0.01$ s	Units: mm

ABC of EMG.pdf	tutorial on EMG technology

6. Introduction to the Arduino

E. Molenkamp, B.H.J. Dekens and M. Odijk (duration 4 hours)

6.1. Introduction

Knight Rider is an American television series that originally ran from September 26, 1982, to August 8, 1986. The series was broadcast on NBC and stars David Hasselhoff as Michael Knight, a high-tech modern crime fighter assisted by an advanced, artificially intelligent and nearly indestructible car (source Wikipedia).

At the front of that car a display is mounted with a moving and fading light, see: http://www.youtube.com/watch?v=1Jmy-a7cqNE

6.2. Problem statement

You have to realize the effect of the moving and fading effect with 8 LED's and the Arduino Uno.

The Arduino Uno (www.arduino.cc) is programmed with a C variant. A tutorial that includes the installation of the Arduino IDE on your laptop and a demonstration of a blinking LED is at http://arduino.cc/en/Guide/Windows. Begin with this tutorial before you are using your own program.

6.3. Assessment

You will have to write a report on the design of your knight rider, following the guidelines stated in chapter 1.3 of this manual. In the report concentrate on this Knight Rider problem, so the installation of the Arduino IDE and the blinking LED demo, is not included in the report.

Focus in the report on:

1. Methods to vary the brightness of a LED and a motivation of your design choice (15%).
2. Schematic of the hardware (15%).
3. Nassi Schneidermann diagram of your software (15%).
4. The "Arduino C" code (a zip file with the code must be included) (25%).
5. A discussion of the results (10%).
6. Quality (readability, clarity and structure) of report (20%).

The percentages indicate how much these aspect are taken into account for the grade of the report. If your solution does not (entirely) have the "Knight Rider effect" then this has consequences for the grade.