Bachelor's Thesis (UAS)

Degree Program in Information Technology

2013

David-Alexandre Davenne

# GPS TRACKER

## – Designing and prototyping hardware and software

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR´S THESIS | ABSTRACT
TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Program in Information Technology

30.10.2013| 42

Instructor: Hazem Al-Bermanei

David-Alexandre Davenne

# GPS TRACKER – DESIGNING AND PROTOTYPING HARDWARE AND SOFTWARE

This main goal of this thesis is to define a process starting from an idea and ending with an electronics final product. A GPS tracker is taken as an example to explain each step.

The thesis is divided into 2 main parts: hardware and software. The hardware part consists of dividing the project into modules which are explained in detail. Then, the design part describes the tools required to produce a schematic representation and also give information about basic electronics components. Testing is an important step for that kind of project and different tools are illustrated. Finally, the printed circuit board creation process will be described from design to ordering and soldering.

The software part is based on a web-app. The main goal was to make it available for any platform, simple and easy to use and it was made possible by using bootstrap, jQuery and PHP. The app offers basic functionalities to interpret the data generated from the GPS tracker.

KEYWORDS:

Arduino, GPS, PCB, Eagle, Design, Electronic components, Bootstrap, jQuery, PHP

# CONTENTS

# APPENDICES



# PICTURES



# FIGURES



# TABLES

# LIST OF ABBREVIATIONS (OR) SYMBOLS

| | |
|---|---|
| MIT | Massachusetts Institute of Technology |
| AVR | Alf and Vegard's RISC processor |
| RISC | Reduced Instruction Set Computing |
| PWM | Pulse-Width Modulation |
| I/O | Input/Output |
| USB | Universal Serial Bus |
| ICSP | In-Circuit Serial Programming |
| GPS | Global Positioning System |
| NMEA | National Marine Electronics Association |
| SD | Secure Digital |
| FIFO | First-In First-Out |
| EDA | Electronic Design Automation |
| PCB | Printed Circuit Board |
| LED | Light-Emitting Diode |
| OLED | Organic Light-Emitting Diode |
| GND | Ground |
| Vcc | Voltage at Common Connector |
| IC | Integrated Circuit |
| ERC | Electrical Rule Check |
| DRC | Design Rule Check |
| IDE | Integrated Development Environment |
| EEPROM | Electrically Erasable Programmable Read Only Memory |
| XML | eXtensible Markup Language |
| LiPo | Lithium Polymer |
| DIP | Dual Inline Package |
| SMD | Surface Mounted Device |
| AJAX | Asynchronous JAvascript and Xml |
| PHP | Php Hypertext Preprocessor |
| REGEX | REGular EXpression |
| API | Application Programming Interface |

# 1 INTRODUCTION

In a world where almost everything can be found on the Internet, it becomes possible for anyone to start their own projects. Resistors, capacitors and transistors are used everywhere and by everyone but a few actually know how they work and how they can be utilized. With the apparition of development boards such as Arduino few years ago, people can create simple projects using basic electronics components.

Arduino offers both a hardware and a software platform and has rich documentation available online. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. Thousands of people have chosen Arduino as their main tool to create all kinds of projects, from blinking lights to space exploration.

Creating simple projects is a good start to become familiar with the tools, codes and components but when starting a more advanced project it is necessary to establish ground rules. Without rules and steps a project can become difficult and might be abandoned due to the lack of preparation. Planning is the key and will help during the whole process. Testing will help finding the problems and improve the features of the project. Producing one or more copy is also possible and will affect costs.

This thesis focuses on a simple step-by-step process used to create a GPS tracker from scratch. It is divided in two main parts: hardware and software. The hardware part is the board with the electronic components and the software part is the interface used to display the data from the tracker. The purpose is not to explain in detail the electronics components and software code but to give a general idea of the steps from designing to testing and producing.

# 2 HARDWARE

## 2.1 Modules

A module is a part of a project whose goal is to execute a specific task. By dividing the project into several functional modules, it is easy to try, analyze and improve each functionality separately.

### 2.1.1 Arduino

Arduino is an electronic open-source platform based on a simple microcontroller (hardware) and a development environment (software) used to write, compile, and transfer the program to the microcontroller.

Arduino can be used to develop interactive objects. It can use inputs from a wide range of switches or sensors and can control several items such as lights, motors or any other kind of hardware. Arduino projects can be autonomous or they can communicate with software running on your computer. The platform can be built manually or can be bought pre-assembled. The development software can be downloaded for free. The Arduino programming language is an implementation of *Wiring*, a development platform based on the programming multimedia environment *Processing*.

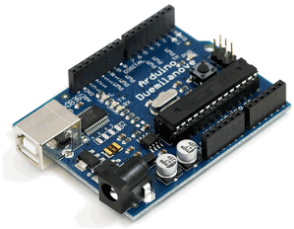
#### 2.1.1.1 Why Arduino?

There are a lot of microcontrollers and platforms based on microcontrollers available for embedded electronics. Parallax Basic Stamp, Phidgets, Netmedia's BX-24, MIT's Handyboard and several others offer the same capabilities. All these tools help microcontrollers programming by proposing an easy way to use them. Arduino simplifies the way to work with microcontrollers and offer several advantages for teachers, students and hobbyists:

- Cheap: Arduino boards are relatively cheap compared to other platforms. The cheapest version of Arduino can be assembled by hand and the pre-assembled versions are usually under 25€.
- Multi-platform: The Arduino software, written in Java, runs on Windows, Macintosh and Linux. Most of the microcontrollers systems are limited to Windows.
- A simple and clear programming environment: Arduino's programming environment is easy to use for beginners but is flexible enough for advanced users. For teachers, it is based on *Processing* environment and students who are used to working with that environment will be familiar with the Arduino software.
- Open-Source software and extensible: Arduino software and Arduino language are published under the open-source license and are available for advanced programmers to work with. The language can also be extended thanks to C++ libraries and people who want to know more about technical details can use C language for AVR microcontroller on which it is based. One can also add the AVR-C language directly in Arduino programs if required.
- Open-source hardware and extensible: Arduino boards are based on Atmel ATMEGA8, ATMEGA168, ATMEGA328, and similar. Schematics are published under the Creative Commons license so advanced electronics designers can create their own version of the Arduino board. Even less advanced users can build the breadboard version in order to understand how it works and save money.

#### 2.1.1.2 Arduino boards

There are several types of Arduino boards. The common basic version, *Duemilanove* (Italian for "2009"), uses the Atmel ATmega328 microcontroller. The former version of *Duemilanove* was using an ATmega168 microcontroller. A more advanced version (i.e., more capabilities and more powerful) named *Mega* is based on the ATmega1280.



**Picture 1.** Arduino Duemilanove

2.1.1.3 Arduino Duemilanove

This thesis will only focus on the Duemilanove board because it is the board used for this project.

The Duemilanove board is a microcontroller board based on the ATmega168 for the first versions or the Atmega328 for the recent versions. Its features:

- 14 digital I/O (Input/Output) whose 6 can be used as PWM (Pulse-Width Modulation) outputs,
- 6 analog inputs which can be used as digital I/O,
- a 16 MHz crystal,
- a USB connector,
- a power supply connector,
- ICSP connector (In-Circuit Serial Programming),
- a reset button.

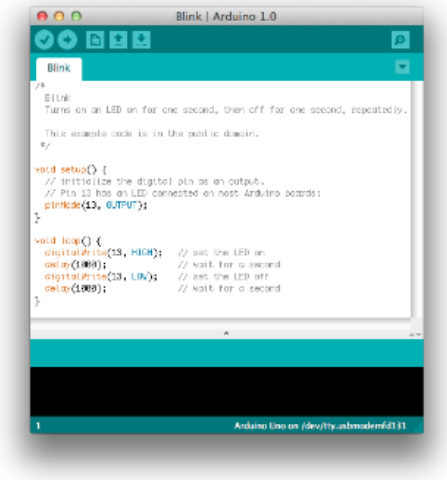
It has everything needed to work with the microcontroller. To use it, it is necessary to connect it to a computer with a USB cable or a separate power supply such as a battery or a transformer.

2.1.1.4 Arduino Software

The Arduino software main functions are:

- write and compile programs to the Arduino board
- connect to the Arduino board to transfer programs
- communicate with the Arduino board

The Arduino software also has a serial terminal that is used to display text messages received from the Arduino and send characters to the Arduino. This functionality is really useful to display the state of variables or results of calculations. It is an essential element to improve, test, and correct programs loaded in the microcontroller.




**Picture 2.** Arduino IDE

### 2.1.2 GPS

Global Positioning System is a space-based satellite navigation system providing position information on Earth. It helps users to determine their position, speed and time of the day on land, sea and in the air 24 hours a day, everywhere in the world. GPS signals are accessible to an unlimited number of people simultaneously.

Each satellite transmits signals to devices on earth. The receptors receive passively the signals from the satellites but they do not emit any signal. They require a clear view of the sky to work properly and are mainly used outside. Their performance can be affected in woods areas or near high buildings. The GPS system works with a network of satellites used to determine accurately the position anywhere on the planet.

#### 2.1.2.1 Skylab SKM53

People always associate GPS with the device used in their vehicle to travel from A to B when they do not know the way. This device contains indeed a GPS receiver but also an interface (hardware + software) to make it easy to use. In order to keep this project small, the only piece of hardware required for this module is the GPS receiver. It can be found online for around 20€ and is relatively small. The Skylab SKM53 GPS receiver was chosen for this project.

This device is tiny (30mm x 20mm x 11.4mm) and fits well in a project where size matters.


**Picture 3.** Skylab SKM53 GPS module

**Table 1.** SKM53 specifications

| | | |
|---|---|---|
| Receiver Type | L1 frequency band, C/A code<br>22 Tracking / 66 Acquisition-Channel | |
| Sensitivity | Tracking<br>Acquisition | -165dBm<br>-148dBm |
| Accuracy | Position<br>Velocity<br>Timing (PPS) | 3m 3D RMS without SA<br>0.1m/s without SA<br>60ns RMS |
| Acquisition Time | Cold Start<br>Warm Start<br>Hot Start<br>Re-Acquisition | 36s<br>33s<br>1s<br><1s |
| Power Consumption | Tracking<br>Acquisition<br>Sleep/Standby | <30mA @ 3V Vcc<br>40mA<br>TBD |
| Navigation Data Update Rate | 1Hz | |
| Operational Limits | Altitude<br>Velocity<br>Acceleration | Max 18,000m<br>Max 515m/s<br>Less than 4g |

The *cold start* is when the GPS memory is empty and the acquisition process has to start from scratch. It will try to find satellites and then will calculate the current position.

The *warm start* is when the GPS remembers the last calculation but still has to find satellites.

The *hot start* is when the GPS device remembers its last calculated position and the position of the satellites.

Another important feature to take in consideration is the power consumption. The project runs on battery so finding a low power device is crucial. The fact that this device will work with 3.3V is good news because all the other modules of the project will work with the same voltage. It means that we can use a common power supply across the board and no regulator will be needed.

The update rate of 1Hz means that the device can update its position every second. It might be a problem depending on how this device will be used and for what purpose. If one needs to track a fast moving object or person then 1Hz might be too slow and one might consider buying a better receiver.

#### 2.1.2.2 NMEA

This receiver uses the NMEA protocol. Each message contained in a packet will start with *$*. The next five characters identify the talker and the type of message. Each data is separated by a comma. There are almost 20 types of interpreted sentences in the NMEA protocol but when it is used with a GPS device, only a couple are used: **$GPGGA** & **$GPRMC**. Here is an example of what a $GPRMC messages look like Figure 1.

```
$GPRMC,220516,A,5133.82,N,00042.24,W,173.8,231.8,130694,004.2,W*70
         1    2    3    4    5     6    7    8      9     10  11 12

      1   220516     Time Stamp
      2   A          validity - A->ok, V->invalid
      3   5133.82    current Latitude
      4   N          North/South
      5   00042.24   current Longitude
      6   W          East/West
      7   173.8      Speed in knots
      8   231.8      True course
      9   130694     Date Stamp
      10  004.2      Variation
      11  W          East/West
      12  *70        checksum
```

**Figure 1.** A GPRMC Message

### 2.1.3 SD card

Secure Digital (SD) is a non-volatile memory card used in portable devices such as cameras and mobile phones. Since 2010, SD cards are a storage standard due to the progressive abandonment of other formats like Sony's memory stick. In 2013, SD cards can have a capacity up to 256GB. There are 3 form factors: original size (32mm x 24mm), the mini SD (21,5mm x 20mm) and the micro SD (11mm x 15mm). This technology offers a way to save data permanently at a cheap price and should be considered in a project where saving information when there is no power supply is a requirement.

In this project, the SD card is used as a data logger. Every time the GPS module gets the coordinates, the latitude, longitude and date/time are saved in a text file. Because the size of the text file can be quite extensive, the use of a SD card is perfect. A 2GB SD card can be found on eBay for less than 3€.

Communicating with an SD card is relatively easy with the right components and libraries. Using a simple voltage divider, an SD socket and the *sdfat* library, it is possible to read and write information to the card.

### 2.1.4 74HC595

The 74HC595 device is an integrated circuit which role is to work as a shift register. A shift register is a bidirectional FIFO (first-in first-out) circuit. To simplify the device, we can imagine 2 input signals: the clock signal and the data signal. The device has also 8 outputs named Q0 to Q7. The clock signal is generally a squared form and is synchronized with the serial data input meaning that every time a high signal (1) or a low signal (0) is sent it needs to be sent on a rising edge of the clock signal. By doing so, the device can identify the serial input easily. When the device has been reset, all the outputs are 0.

Every time a new bit is sent to the data input, the shift register will "shift" the bits to the next output.

Example:
If the output at a given time is **1**0011010 where **Q0** is 1 and **Q7** is 0. When a 1 is sent to the data input, the whole register will shift and will give: **1**1001101 where **Q0** is 1 and **Q7** is 1.

Due to the fact that these devices can run at really high speed (100MHz) it is possible to use them as a serial to parallel converter. The shift register concept has been explained with 1 bit sent to the data input but if 8 bits are sent to the input, these 8 bits will become the output. This is how this device is used in the project.

The 74HC595 is used as an error manager. With the use of 2 RGB (Red, Green and Blue) LED connected to 6 outputs (2x3 outputs) of the device and using binary logic, it is possible to generate error codes and display them by using LED colours. For example, if one LED needs to be red, one LED needs to be blue and they are connected in such a way that the pin for the blue colour of LED1 is on output Q1 and the pin for the blue colour of LED2 on the pin Q5; it means that 0**1**000**1**00 needs to be sent to the data input. By giving a numerical value to each colour for each LED, we can generate the proper output just by giving a decimal value. In the example, 01000100 is 64 (red LED1) + 4 (blue LED2).

## 2.2 Designing

### 2.2.1 Eagle

Eagle is an EDA (Electronic Design Automation) and Printed Circuit Board (PCB) layout editor application by *CadSoft*. It is a great choice for students and hobbyists because they offer a free version (Eagle Light Edition). This version has some limitations though: the usable board are is 100x80 mm which means that we can only design PCB up to that size, only two signals layers (top and bottom), and the schematic editor can only create one sheet. Those limitations might look like a drawback but it will suit perfectly for any kind of simple project that one might need to do. Eagle has also several advantages. Due to its popularity there are numerous libraries available to download. A library is a file

containing one or more electronic component(s) or piece of hardware's specifications translated into Eagle's language. This allows users to use components which aren't in the default libraries. Another advantage is the ease of use. Everything is instinctive and one gets the hang of it after few minutes.

Eagle has additional features but we will focus mainly on the schematic and PCB design part.

#### 2.2.1.1 Schematic

When starting an electronics project, defining each module and looking for the components needed is really important. The next step is to create a "map" with all the components which is called a schematic. An electronic schematic is used to describe the design of the equipment. It is like a subway map where each component connections with others can be seen. Modern tools go beyond the simple drawing of devices and connection and are linked to other EDA tools for verification and simulation. It is worth noting that those tools are only uses to draw the circuit, they will not notice if something is wrong (other than basic checking). For example, if one happens to invert the polarity of a battery in the circuit, it will not notice it.

On a schematic, electronic components are represented by their symbol. There are standards and each component has its own symbol. Each component has also a name and a value. Depending on the category of component, they might as well have the model type. Those standards help people reading the schematic to know directly what they are dealing with.

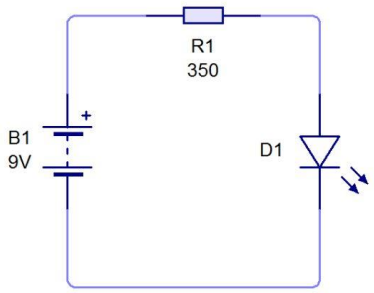



**Figure 2.** Simple schematic

Thanks to electronics standards, the circuit in Figure 2 can be easily identified. We have a 9Volts battery (B1), a 350Ω resistor (R1) and a LED (D1). The values can help us determine unknowns such as the current flowing through the circuit or the voltage across the resistor. With all this information, it is easy to know what is happening.

Schematics help engineers and hobbyists understand the structure of electronic devices. More advanced circuits need more analysis and study to fully comprehend them.

#### 2.2.1.2 Electronic components

The goal of this part is not to explain each component in the details but classify each component used in the GPS tracker and give basic explanation for each one.

Passive components

Passive components cannot introduce net energy to the circuit and they include two-terminal components such as resistors, capacitors, inductors and transformers. They do not require energy in order to work because the current usually flows through them.

Transistor

A transistor is an active electronic component that can be used: as a switch in logical circuits, as a signal amplifier, as a voltage stabilizer and many other

functions. A transistor has usually 3 pins: 1 collector pin, 1 emitter pin and 1 base pin (for bipolar junction transistors).

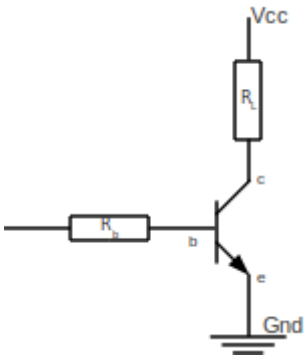



**Figure 3.** Transistor used as a switch

Figure 3 represents a transistor used as a switch. $V_{cc}$ is the supply voltage, $R_L$ is the load resistor, $R_b$ is the base resistor and GND is the ground. The word “load” is used in electronics to define any kind of equipment that could fit in the circuit. For example, a motor or a light could replace $R_L$.
If the circuit is powered up, nothing will happen because the transistor acts like an open switch. It means that if $R_L$ was a light, it would be switched off. If current is sent to the base of the transistor (through $R_b$) the switch will close and the current will flow from $V_{cc}$ to GND, switching on the light.

As stated previously, there are dozens of functions for transistors and it is not the purpose of this thesis to explain them all. The basic switch function was explained because it is the most used function in Integrated Circuits to create logical gates.

Integrated circuit

An integrated circuit (IC) is an electronic component reproducing one or several electronic function(s) more or less complex and is made of several basic electronic components. They are great to use in electronics projects because they are so small. It could be possible to create ICs from scratch but it would require so much space, time and money that it is easier to buy them all-made. There are thousands of integrated circuits and they are split in two categories: analog and digital.

- Analog ICs perform functions like amplification, active filtering, demodulation and mixing.
- Digital ICs usually contain one to several millions of logic gates, flip-flops and other circuits. They work using binary mathematics to process “one” and “zero” signals.

Integrated circuits are really easy to identify because they have from four to several dozens of pins. The models are also easy to determine because they are usually written on the package.

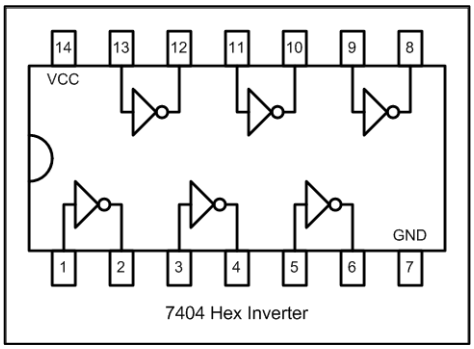



**Figure 4.** 7404 pinout

This IC’s model in Figure 4 is a Hex Inverter. The inverter will invert the input signals meaning that if a logical 1 is sent on pin 13 for example, a logical 0 will come out from the output 12 and vice versa. Hex is Greek for “six” meaning that there are 6 inputs that can invert a signal. In order to function properly, ICs usually need a power supply ($V_{cc}$ and GND) ranging from a couple of volts up to a dozen of volts.

The ATmega328 chip is basically a really complex Integrated Circuit.

### 2.2.2 Datasheet

There is a myriad of transistors, integrated circuits, diodes and all kind of electronic devices. There is also a bunch of companies manufacturing those components which means that they might have different kind of features. When one designs a project, the choice of components is crucial. One has to check

every features such as supply voltage, dropout voltage, maximum temperature, speed, size and so on.

Each component has its own datasheet. A datasheet or spec sheet is a document summarizing the performance and technical characteristics of an electronic component. They contain everything we might need to know about a component: manufacturer's name, product number, list of package formats, device properties, pin connection diagram, absolute minimum/maximum ratings (e.g., voltage, current, and temperatures), recommended operating conditions, timing diagrams, application recommendations etc. Some datasheets even show some circuit examples. The best advantage is that it is completely free to download. These datasheets can be rather small (a couple of pages) but for more advanced devices, they can reach 500 pages.

When we have decided what component will be included in the project, we have to take a look at the datasheet. We also verify that the pin out corresponds to the one we have on Eagle or the designing software, we check the supply voltage to be sure we will not burn it with a too high voltage and the input/output current. If these 3 settings are checked, it should not be possible to destroy the device. Different companies manufacture the same type of device but there might be slight differences in the specs. Always check the datasheet before using any electronic component.

When the schematic design is ready and the datasheet specs have been checked, we run the ERC (Electrical Rule Check) in Eagle will help find the errors. It will NOT warn if a connection is not logic but it will tell you if an integrated circuit has no power supply or warn if a component has no value.

### 2.2.3 Visual Studio

To program an Arduino board, one can use the Arduino integrated development environment (IDE) which is a basic tool but not as good as other software on the market. The best option for this project was to use Visual Studio.

Visual Studio is an IDE created by Microsoft. I have been using it for years with Visual Basic and C# on all kind of project so I know it quite well. By default, Visual Studio cannot program Arduino but thanks to a group of motivated people it is possible to install an Arduino IDE in Visual Studio. It is called Visual Micro and offers all the features of Visual Studio such as Intellisense making it easy and fast to program Arduino.

Programming Arduino is like creating a code for the core of the project. This code will handle different kinds of input signals, sending output signals, read data, save data and anything possible related to the project. It is the only part of the project that can be modified relatively easily. All the other components are designed to do one or several tasks and that is all they will do but one can add, modify or delete code in the Arduino to shape it the desired way.

In order to create an efficient code, one needs to split it in different parts, each representing a module. For this project, one part will deal with the GPS, one part will save data on the SD card and one part will handle errors. Before writing the whole code at once, creating the code for each module separately and then trying them out is the best approach. When each module has been tested separately, codes can be added together and tested as a whole.

Creating an efficient code from scratch is possible but it might take a long time which is why there are libraries with Arduino. Some libraries are official and some are made by Arduino users. The latter ones have been tested and improved through time and are usually working quite well. For this project, two libraries are being used: tinyGPS and sdFat.

- TinyGPS is designed to provide most of the NMEA GPS functionality an Arduino user would want – position, date, time, altitude, speed and course without the large size that seems to accompany similar bodies of code. To keep resource consumption low, the library avoids any mandatory floating point dependency and ignores all but a few key GPS fields. Different kind of examples are available and make it easy to implement the functionalities

required.
More information can be found at: www.arduiniana.org/libraries/tinygps

- SdFat is an Arduino library that supports FAT16 and FAT32 file systems on standard and high capacity SD cards but only supports short 8.3 file names. It supports file creation, deletion, read, write, and truncation. SdFat supports access to subdirectories, creation, and deletion of subdirectories and is designed for 328 or larger Arduino's but smaller applications will run on 168 Arduino's.
  More information can be found: https://code.google.com/p/sdfatlib/

As we create an Arduino project in Visual studio, the IDE will create an .ino file. This file will have two functions: *setup()* and *loop()* written. The *setup()* function is called when a sketch starts and is used to initialize pin modes, variable and libraries. This function will run only once when the Arduino has started. The *loop()* function, as the name suggests, loops consecutively and allows the program to change and respond as it runs. The code inside the loop is used to control the Arduino.

The goal is not to teach how to create code for the Arduino but it might be difficult to understand the whole concept without an example. The following example is really basic and should help to understand how it works.

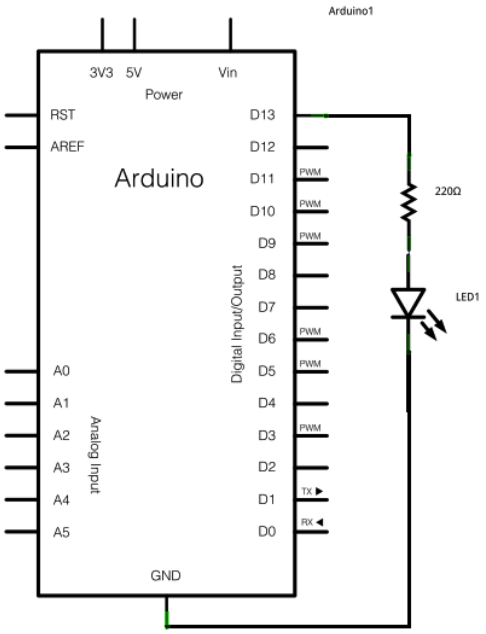



Figure 5 represents an Arduino and a simple LED-resistor circuit. An LED (LED1) is connected in series with a resistor (220Ω) in order to limit the current and avoid burning the LED. One side of the circuit is connected to the digital input/output #13 (D13) and the other end is attached to the ground (GND) making it a closed loop.

**Figure 5.** Simple Arduino Circuit

The code in Table 2 is used to turn on and off the LED. Because it is in the loop function, it will result in a blinking effect. D13 is used as an *output* because a signal is sent to the LED but it is not trying to read anything. Because the signal is sent, it uses *digitalWrite* to write a state (HIGH or LOW) on the pin D13. A HIGH state will set a positive voltage between D13 and the ground thus lighting up the LED, a LOW state will light it off.

**Table 2.** Arduino basic code example

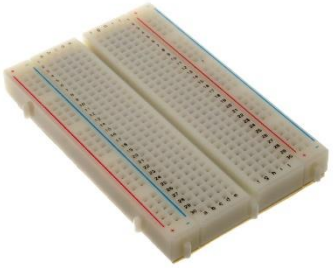
```
// Pin 13 has an LED connected on most Arduino boards.
int led = 13; // the setup routine runs once the Arduino starts:
void setup() {

  pinMode(led, OUTPUT);      // initialize the digital pin as an output.
}
// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(1000);                // wait for a second
}
```

## 2.3 Testing

### 2.3.1 Breadboard

Since all the theory part has been completed, it is time to deal with the practical part. The best tool to practise electronic circuits is the breadboard. A breadboard is a tool used to create prototypes or testing electronic circuits. The main advantage of this system is that it is totally reusable because it does not require any soldering. It comes in different sizes and shapes but the main design is always the same.



**Picture 4.** Breadboard

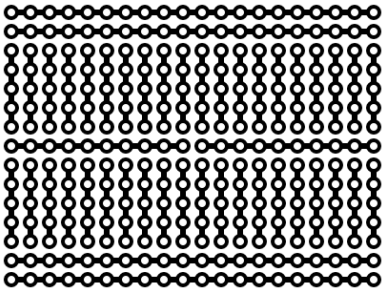


**Figure 6.** Breadboard's hole pattern

These boards are really convenient and are sold around 10€. They are perfect for experimenting with electronics circuits because we just need to plug in the components and wires according to the pattern on the Figure 6. This means that it will be impossible to plug in a component with two pins on the same line because this would cause a short circuit. Therefore, the component has to be plugged in two different parallel lines. It might look quite difficult but once the breadboard has been used for a while, it is easy to use.

Arduino is a development platform based on the ATmega 328 microcontroller by Atmel as previously mentioned but it is possible to run a minimalistic version of Arduino with only the microcontroller and a few extra components: “Arduino to Breadboard”. The easiest way to proceed is to upload the program to the microcontroller then create a circuit similar to the one in Figure 7.

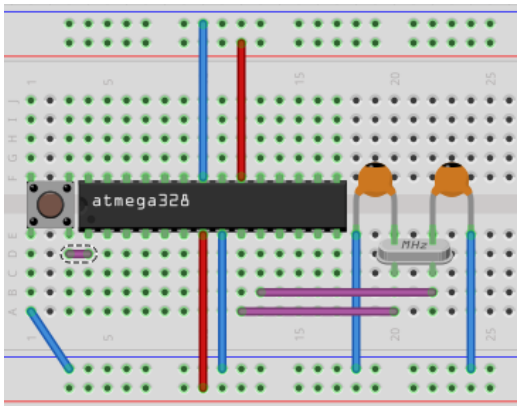



**Figure 7.** Arduino on breadboard

Using designing tools such as Fritzing can help beginners to design their circuit on a breadboard. Figure 6 represents the minimum setup to run a sketch: a button to reset the Arduino, an external clock (called a crystal) running at 16MHz with 2 ceramic capacitors to reduce noise. The whole circuit also requires a power supply (red and blue wires) between 1.8V and 5.5V. We suppose here that the buses ($V_{cc}$ and GND) are connected to each other.

In order to test the circuit, a LED and resistor connected to the ground can be added to the pin 19 (=digital I/O #13) on the chip and the LED blinking sketch can be loaded. When it is working, the main code is loaded and the different modules plugged in the breadboard.

### 2.3.2 GPS tracker testing

For the GPS tracker, each module has been tested separately:

- The GPS module had been connected according to the datasheet and coordinates were received without too many problems thanks to the tinyGPS library. The module needs to be in a direct sight with the sky in order to detect satellites and determine the position. When the satellites have been found, it works inside a building.
- The SD card module was slightly trickier because there were several different pins and the required circuit to some extend more complex and thus confusing. But after a while, it works.

- The error module using the 74hc595 IC requires basic binary encoding with the RGB LED's but no major problems were encountered.

When each module has been tested, the next step is to combine modules together to check everything is working. When all the modules are working together, the code looks good and it is running the desired way, it is ready for the next step.

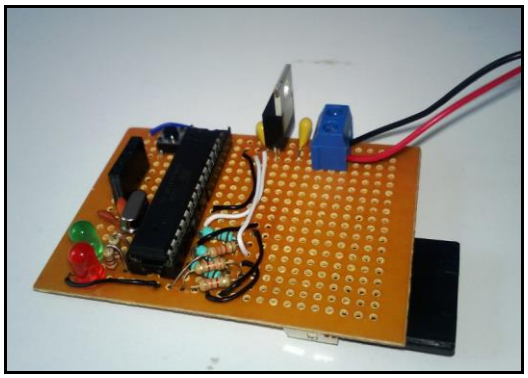
### 2.3.3 Strip board

Having a breadboard to test connections is good but not very practical for a lot of different projects. It is big, there is a bunch of wires everywhere, the board is not really steady and so are the components. It will do to test and improve but we will not be able to use it in a real situation, especially for this project which should be small and transportable easily.

A stripboard is a prototyping board characterized by a 2.54mm regular grid of holes. There are different types of stripboards but the most used have wide parallel strips of copper cladding running in one direction all the way across one side of the board. Another type of board has independent holes which are not connected to each other whatsoever. The former is similar to the breadboard discussed earlier because of the parallel strips.

The 2.54mm might seem odd but is in fact the metric value of 0.1 inch which is the commonly used standard for pin spacing of electronic through hole components. For some components such as resistors the spacing is not a problem because we can modulate the spacing size manually but it is not possible for certain types of devices such as Integrated Circuits.

The stripboard has a main advantage compared to the breadboard: components are soldered on the board making them sturdy and they won't move easily thus making the circuit much more reliable. The downside of the soldering is that it is really difficult to change something when everything is in place. It is not impossible but it requires a de-soldering technique which is not always working properly.



**Picture 5.** GPS Tracker on a stripboard

Picture 5 shows the battery connector with the voltage regulator providing a constant 3,3V to the circuit. Resistors are used for the SD card module and the LEDs to display the state of the code (e.g., running, recording, errors). The GPS module is plugged in the headers on the left. The SD card socket is soldering on the bottom part of the board and can be seen on the right side. The size of the board is 7x5cm which is a good size and can be used to test it outside and it can be installed inside a small plastic box.

This type of prototyping board is more suitable for a more reliable and long-term project. It is good enough if only 1 copy is required but totally inappropriate if several dozen are needed because it is time-consuming to solder all the components and wires.

### 2.3.4 Results

Once the stripboard is ready and basic checking has been done, it is time to test it out. The first tests were good and the data received can be used to display the path on a map. The text file containing the coordinates has the following format:

1. The name of the file is generated every time the device reset and is composed of the date of the day followed by _ and a number/letter such as *300913_1.TXT*. The number is incremented after each reset and saved in the EEPROM of the micro controller. It goes from XXXXXX_1.TXT to XXXXXX_Z.TXT meaning that 35 (9+26) files can be created every day. The easiest way to create a new file is to push the reset button on the device. This will result in a hot start for the GPS module and the position will be found within seconds.

2. Each line in the file represents one data record and is defined as follow:
   **60.123456#22.123456#300920130900**
   Each line is divided like this: the **latitude**, the **longitude**, the **date**, the **clock time** and the **separator character**.

When the file is created and filled in with data, it is not human readable and without a proper tool it is impossible to obtain a proper vision of the path. The best way to visualize something on a map is to use a free tool such as Google Earth. Google Earth uses a .kml format file based on XML that can be generated from a text file containing lines with latitude and longitude. After tweaking the code to generate that kind of data, it was possible to generate and download a .kml file from a website like http://kmltools.nobletech.com/gps2kml .

The trace on the map was usually accurate but sometimes it was a slightly off by a few meters on the side. This can be explained by the fact that Google Earth satellite pictures might not be perfectly aligned with the correct coordinates. The problem can be verified by changing the picture timestamp in Google Earth. For example, with the satellite pictures of 2010, the trace is slightly off but perfectly aligned with the road on the 2012 pictures.

Google Earth provides a lot of useful information for that kind of trace:

- The total distance of the trace
- The elevation profile
- Minimum, maximum & average elevation

### 2.3.5 Battery life

One of the most important feature of a portable device such as a GPS tracker is the battery life. The main purpose of this tracker is to be used for hiking meaning that a power source might not always be available to reload it. The goal was to have a battery life of at least 8 hours corresponding to 1 day of intensive hiking.

The first tests were made using a 9V LR61 Alkaline type battery. It was working well and thanks to the voltage regulator of the device, the circuit could get a steady 3,3V. There were three major drawbacks to this setup though:

- The LR61 battery is not rechargeable.
- The battery is not cheap (~3€).
- That kind of battery has a low capacity (~500mAh).

By using an ammeter in the circuit, it was possible to determine the average current consumption which was around ~60mAh. Those results meant that a LR61 battery would last around 8 hours when fully charged. It was good but not really convenient because it would cost a fortune to buy a new battery every time the tracker is used for 8 hours. After some researches into battery types and technologies, the lithium-polymer battery was the answer.

Lithium-ion polymer battery (abbreviated LiPo) is rechargeable battery and is usually composed of several identical secondary cells. They can be built in parallel to increase current capabilities or in series to increase total voltage. That kind of battery is widely used in the radio-controlled world (such as aircraft and cars). These batteries have the following advantages:

- They are relatively small.
- They can be recharged really quickly depending on your charger and the type of cell.
- They have large capacity ( some are over 10000mAh)

All these features make LiPo the perfect choice for that kind of application. The battery can be found on eBay at around 7€ for a 3,7V 1000mAh (Picture 6) and a few euros for a USB charger. A few precautions have to be taken when working with LiPo batteries though. For example, overcharging can cause an explosion or fire and the load has to be removed as soon as the voltage drops below 3V. This is why a battery with built-in protection circuit (avoiding overcharging and over discharging) is preferred.

**Picture 6.** A 3,7V 1000mAh battery with built-in protection

The results with this battery were astonishing. It takes 1 hour to charge it to its full capacity with the USB charger and it runs non-stop for 16 hours with the GPS tracker. This means that it can be used for a 2-day-trek without recharging it. Higher capacity batteries with in a slightly bigger can also be found on eBay.

## 2.4 Producing

### 2.4.1 PCB

A printed circuit board (PCB) mechanically supports and electrically connects electronic components using conductive tracks, pads and other features etched from copper sheets laminated onto a non-conductive substrate. Advanced PCB's may contain components (such as capacitors, resistors or active devices) embedded in the substrate. (Wikipedia)

A PCB is the ending part of any kind of electronics project because it gathers all the components together in a definitive way. Once the PCB has been manufactured, there is no turning back and modifications are really hard to make due to the fact that we don’t have access to the traces because they are usually covered with a protective layer. Double checks and triple checks are required before ordering a custom PCB.

PCBs are created by a complex set of machines which have specific sets of rules. For example, due to accuracy limits of these machines, two traces cannot be too close to each other. Having both traces next to each other could result in a short circuit on the board. PCB manufacturers often provide those sets of rules to customers for them to use with their favourite electronic design software. Eagle has a DRC (Design Rule checking) feature that can be used to check the design. When the set of rules file has been downloaded, it can be loaded into Eagle and a summary with the errors and warning is displayed.

When the design is ready and has been carefully checked, it is time to order the boards. With the Internet, it is easy to find a PCB manufacturer but not always at a cheap price. There are tons of options regarding PCB manufacturing. The main

concept to keep in mind is that the more we order, the cheaper it will be but as a hobbyist we might not need 1000 copies of a project. If only a few copies are needed, the cheapest way is to order a certain amount of pre-sized PCB. It means that we cannot choose the size and the design has to fit in a defined size. For example, *Seeedstudio* offers this service with sizes from 5cm*5cm up to 20cm*20cm (by 5cm range). It will cost around 10€ for 10 boards. *Seeedstudio* is a great choice if the board size does not need to be a specific size but the waiting time can be quite long (around 1month from ordering to receiving). For this project, the PCB needed to have a specific size to fit in a plastic project box therefore another manufacturer had to be found.

OSH Park is a community printed circuit board order. They take designs from different people, put them together on a panel and order the panel from the manufacturer. The boards are made in the USA and can be easily recognizable because of their purple silk screen. The process from ordering to receiving is really easy and the website is intuitive. Another great feature is the Eagle compatibility. Usually the gerber files (file format used by the PCB industry) are required and have to be generated separately from the design with specific tools. With OSH Park, it is possible to send the .brd (board) file from Eagle.

To sum up :

1. Download the set of rules and check your design
2. Upload the .brd file of your project (board file) to the website
3. Top and bottom layers pictures are generated so they can be checked for errors
4. The design is sent to the manufacturer (every couple of days)
5. The boards are sent home

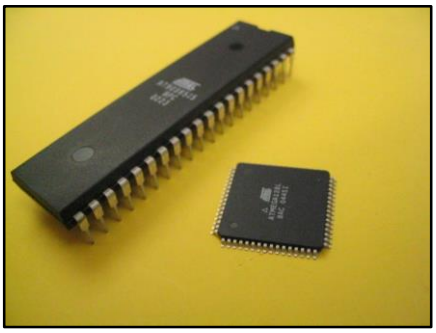
The total process takes a little more than 2 weeks and the price is calculated according to the size of the board: 5$ per square inch for a 2 layers board with a minimum order of 3 boards. The total cost for this project was around 30€ for 3 boards.

### 2.4.2 List of components

With the boards at home, the only part missing is the components. To be efficient in the creation process we should have already tested the components with the breadboard. If the plan is to produce several boards then ordering online in large quantities is the best bet to ensure cheaper prices. Every component bought should have its datasheet inspected carefully and each feature analysed in order to be sure it will fit the purpose intended.

Depending on the type of board designed, two types of components are available: through hole and SMD.

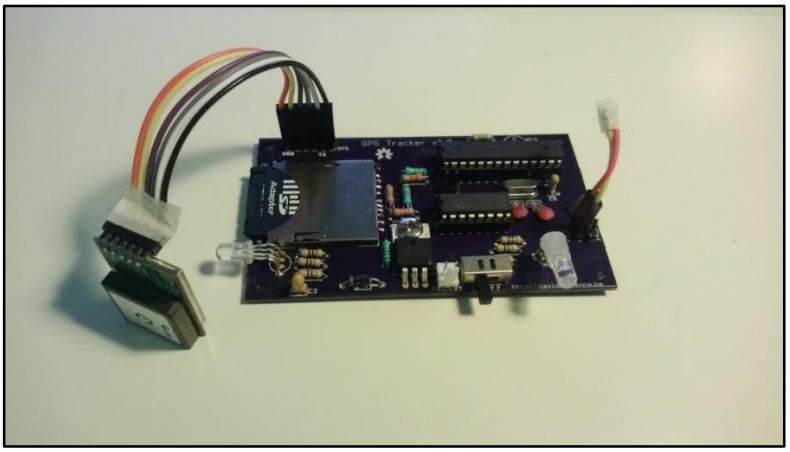
- Through-hole technology refers to the mounting scheme involving leads on the component being inserted into drilled holes. They are then soldered on the opposite side. These components are usually easy to solder and a perfect choice for a beginner's projects.
- SMD stands for Surface-Mount Technology and is a technology in which the component are mounted directly on the surface of the PCB. The main advantage is the size of the components as they are really small compared to Through-hole components but require much more dexterity to solder. They are also much cheaper but need to be ordered in large quantities.



**Picture 7.** Comparison between a DIP and a SMD micro controller

The total cost of one device is under 45€ as shown in the Table 3. A SD card must be added to the total price as well as a project box to protect the whole assembled board. The main part of the budget is absorbed by the PCB custom

design and the GPS module. The total price could be reduced by using SMD type components and by decreasing the size of the board.



**Picture 8.** Final product with custom PCB from OSHpark

**Table 3.** List of components for the GPS tracker

| Type | TH/SMD | Value | unit price | quantity | total price | |
|---|---|---|---|---|---|---|
| Resistor | TH | 1,8K | 0.01 | 3 | 0.03 | € |
| Resistor | TH | 3,3K | 0.01 | 3 | 0.03 | € |
| Resistor | TH | 470 | 0.01 | 7 | 0.07 | € |
| Potentiometer | TH | 50K | 0.19 | 1 | 0.19 | € |
| LED | TH | RGB | 0.27 | 2 | 0.54 | € |
| Capacitor | TH | 1uF | 0.12 | 3 | 0.36 | € |
| Capacitor | TH | 22pF | 0.01 | 2 | 0.02 | € |
| Crystal | TH | 16 MHz | 0.1 | 1 | 0.1 | € |
| Diode | TH | 1N4001 | 0.02 | 1 | 0.02 | € |
| Male Pin header | TH | 40 pins | 0.15 | 1 | 0.15 | € |
| DIP socket | TH | 28 pins | 0.11 | 1 | 0.11 | € |
| DIP socket | TH | 16 pins | 0.06 | 1 | 0.06 | € |
| Wafer socket | TH | 2 pins | 0.02 | 1 | 0.02 | € |
| Switch | TH | 1P2T | 0.05 | 1 | 0.05 | € |
| Switch | TH | Push button | 0.22 | 1 | 0.22 | € |
| Socket | SMD | SD card | 0.1 | 1 | 0.1 | € |
| Micro Controller | DIP | Atmega328 | 3 | 1 | 3 | € |
| IC | DIP | 74HC595 | 0.2 | 1 | 0.2 | € |
| Regulator | TH | LM1117 3.3V | 0.4 | 1 | 0.4 | € |
| GPS Module | X | Skylab SKM53 | 20 | 1 | 20 | € |
| Battery | X | LiPo 1000mAh | 8 | 1 | 8 | € |
| PCB | X | 2 layers 95mmx55mm | 10 | 1 | 10 | € |
| Miscellaneous | X | wires, solder, … | 1 | 1 | 1 | € |

| Total | 44.67 | € |
|---|---|---|

## 2.5 Improvements

The project could be improved because it has been made in a way that it is easier to debug and fix than with a more compact version. First, the main part of improvement will focus on the choice of components and their sizes. As previously written, SMD technology would be a better choice and the size could be decreased by half. Using a microSD socket instead of a SD socket would gain a lot of space as well. Then, the LED error system could be replaced by an OLED display. It is relatively cheap, easy to interface with Arduino and don't consume much power. It could

display much more information such as real time coordinates and distance calculations.

Another useful improvement would be to add a micro USB connector to: load the battery directly, program the micro controller and retrieve the data. This would require much more designing which can be tricky to do though. Finally to reduce the costs, the best way would be mass production which would drastically decrease the prices of the components and the PCBs.

# 3 SOFTWARE

Different projects require different needs. For this project, working only with the hardware would be fine but being a programming enthusiast it was desirable to create both hardware and software parts. The main advantage of creating a custom software is that it is possible to create and include features that might not exist in other available solutions or use these features in a different way. This helps share the data between the hardware and the software. For example, not having a custom software interpreting the data might mean having to create fully compatible output from the hardware and that might increase the amount of data generated and thus decrease the performance of the hardware. For the GPS tracker, the only data provided was latitude, longitude and date/time. There might not be any software able to create something explicit from those data but if it is used with an application designed properly, it can work with them and create custom features. This helps to focus the hardware part on providing the bare minimum (raw) data and focus on performance where the software will analyse the data and work with them using high power calculations.

When designing an application, one has to list the top priorities. The main requirement for the application was to make it available on any platform. This already scopes down the possibilities and Java comes in mind when talking about cross platform compatibility. Java was indeed the first solution but, in my opinion, has a really bad user interface so further researches were necessary. I decided to go for a web application, using Bootstrap with jQuery as a front end and PHP as a back end. I had little knowledge of bootstrap and jQuery and that could be a way to improve my skills in this area.

## 3.1 Modules

### 3.1.1 Bootstrap

Twitter Bootstrap is a collection of tools for creating websites and web apps. It is free and contains HTML/CSS-based design templates for any kind of forms, buttons and other components. Bootstrap is compatible with all major browser

making it a great choice for a multi-platform app. Bootstrap is open source and available on GitHub.

The main feature of Bootstrap is its responsiveness. A responsive web site adapts the layout to the viewing environment using fluid, proportion-based grids, flexible images, and CSS3 media queries. This makes it possible for users to have a nice experience while navigating the website with a mobile device such as a phone or a tablet.

Bootstrap is a great tool for people who do not want to spend hours to design their app and focus on the functionality. It offers a really nice visual appeal and is really easy to modify to your needs. It was used in this project to create a clear and easy to use front page where users would upload their data files from the GPS tracker to the website. They can then load the data to the server which would generate basic stats and show up the track on a Google Maps.

### 3.1.2 jQuery

jQuery is a fast, small, and feature-rich JavaScript library. It is an easy-to-use API that works across a multitude of browsers, jQuery simplifies HTML document traversal and manipulation, event handling, animation, and Ajax.

jQuery and Bootstrap are usually combined because they offer the possibility to create amazing web app. When using jQuery or JavaScript on a web page, it is possible to update the DOM (Document Object Model) of the page without reloading it and making the web app look like a real app.

For this project, the basic File Upload widget for jQuery has been used. It allows users to upload their files to the server in a very smooth way. They can upload one or multiple files at the same time and a progress bar is displayed while the file(s) is/are uploaded. When configured properly, it is possible to restrict upload to specific kind of files and set a file size limit which is very useful for this project.

jQuery was also used for Ajax. Ajax is a group of web development techniques used on the client-side to create asynchronous web apps. Without using Ajax, a

web page would need to reload in order to display information retrieved from a server. Ajax creates an asynchronous request to a web server, the server sends back the results and the data is shown on the page without reloading the page. In this project, Ajax sends a request to a PHP script that checks the file, reads the file, and generates stats.

### 3.1.3 PHP

PHP is a server-side scripting language designed for web development. This means that all the code executed by PHP is executed on the server and not on the client.

For the GPS tracker web app, PHP is used to do the following tasks:

- Upload the text files from the GPS tracker to the server
- Check the files structure using a regex
- Parse the files and return them in a JSON format
- Generate stats for each file (total # of records, total time, estimation of total distance and average speed)

The estimation of total distance is made by using the *haversine* formula in Table4.

**Table 4.** haversine formula

| |
|---|
| $a = \sin^2(\Delta\varphi/2) + \cos(\varphi1).\cos(\varphi2).\sin^2(\Delta\lambda/2)$ |
| $c = 2.\text{atan2}(\sqrt{a}, \sqrt{(1-a)})$ |
| $d = R.c$ |

Where φ is latitude, λ is longitude, R is earth's radius (6371km)

### 3.1.4 Google

The Google Maps API is used in this project to display the track on a map. The track is represented by an array of coordinates which creates a polyline. A starting point and ending point are also created.
The web app also allows the users to download a google earth file. The kml file

used by google earth has a similar format to XML. Google earth offers much more information such as the elevation profile and the satellite view.

### 3.2 Process

The web app is accessible at the address http://david.davenne.be/gps . It is really easy to use and does not require any advanced skills.

- The first step is to click on the "*Select files …"* button.
- When the window opens, chose one of several files. Each file should have the format **000000_X.TXT** as its name and extension.
- During the upload, a progress bar is displayed, several cases are possible
    - a green box means that the file has been uploaded correctly
    - a yellow box means that there is a problem with the file : size too big (maximum size is 2MB) or extension incorrect (must be .TXT)
- If the box is green, there will be a "*Load*" link on the right that needs to be clicked in order to generate the stats and load the data.
- When the "*Load*" has been clicked:
    - The data loads properly: stats, table with coordinates, google map and google earth link are available
    - There was a problem when parsing the file and the line number where the problem has been detected is shown.

# 4 CONCLUSION

The main goal of this thesis was to explain the different steps possible to create an electronics related end-product from a basic idea. The main idea to keep in mind is that there is no perfect way to do it and mistakes will happen during the process. By doing new projects and ameliorating methods, it is possible to improve the whole process which will reduce their general costs. No matter what the steps are, planning before acting is always the best rule.

First dividing the project into modules will help identify the needs and requirements of each part of the project. Trying and testing them separately will

decrease debugging time and increase the expertise. When each module has been successfully tested, they can be assembled together and tested.

Then creating a schematic using standard symbols will help you and any other people that might need to help you. Schematics can be created using several tools available on the market and Eagle was the tool discussed in this thesis. Eagle can create both schematics and PCB design and connect both part of the project together meaning that when a modification is made on the schematics, the PCB is modified according to that. It is worth noting that both schematics and PCB design have their own rules and checking these rules carefully is essential to success.

In order to give a “professional” look to your project, it is possible to order PCBs from specialized manufacturers. The price mainly depends on the quantity ordered. Manufacturers have different rules and different kind of options and checking them carefully can save money and time. Ordering components for a board can be tricky but will become easier with time. Checking datasheet specifications for each element will avoid problems. SMD and through hole components are both viable options and both have their pros and cons.

Finally, designing a web app follows the same rule as the hardware part: planning and testing. There are many advantages in creating a custom app, including the possibility to optimize the hardware by reducing the amount of resources used and using the power of the web app to do the calculations, implement own features and improve general knowledge by using technologies which might not be familiar.

This thesis, schematics and codes are available and free to download from: http://david.davenne.be/gps

# SOURCE MATERIAL

Arduino hardware related information:

http://arduino.cc/fr/Main/DebuterIntroduction
http://arduino.cc/fr/Main/MaterielI
http://arduino.cc/en/Main/ArduinoBoardUno
http://arduino.cc/en/Main/Products

Arduino software related information:

http://arduino.cc/fr/Main/DebuterPresentationLogiciel

GPS related information:

http://en.wikipedia.org/wiki/Global_Positioning_System
http://www.fraskito.net/?q=system/files/u1/SkyNav_SKM53_DS.pdf (SKM53 Datasheet)
http://aprs.gids.nl/nmea/ ( NMEA protocol )

SD card related information:

http://en.wikipedia.org/wiki/Secure_Digital

74HC595 related information:

http://en.wikipedia.org/wiki/Shift_register

Eagle related information:

http://www.cadsoftusa.com/eagle-pcb-design-software/product-overview/

Components related information:

http://en.wikipedia.org/wiki/Transistor
http://en.wikipedia.org/wiki/Integrated_circuit

Visual Studio related information:

http://www.visualmicro.com/
www.arduiniana.org/libraries/tinygps (Tinygps library)
https://code.google.com/p/sdfatlib/ (SdFat library)

Breadboard related information:

http://en.wikipedia.org/wiki/Breadboard

LiPo related information:

http://en.wikipedia.org/wiki/Lithium_polymer_battery

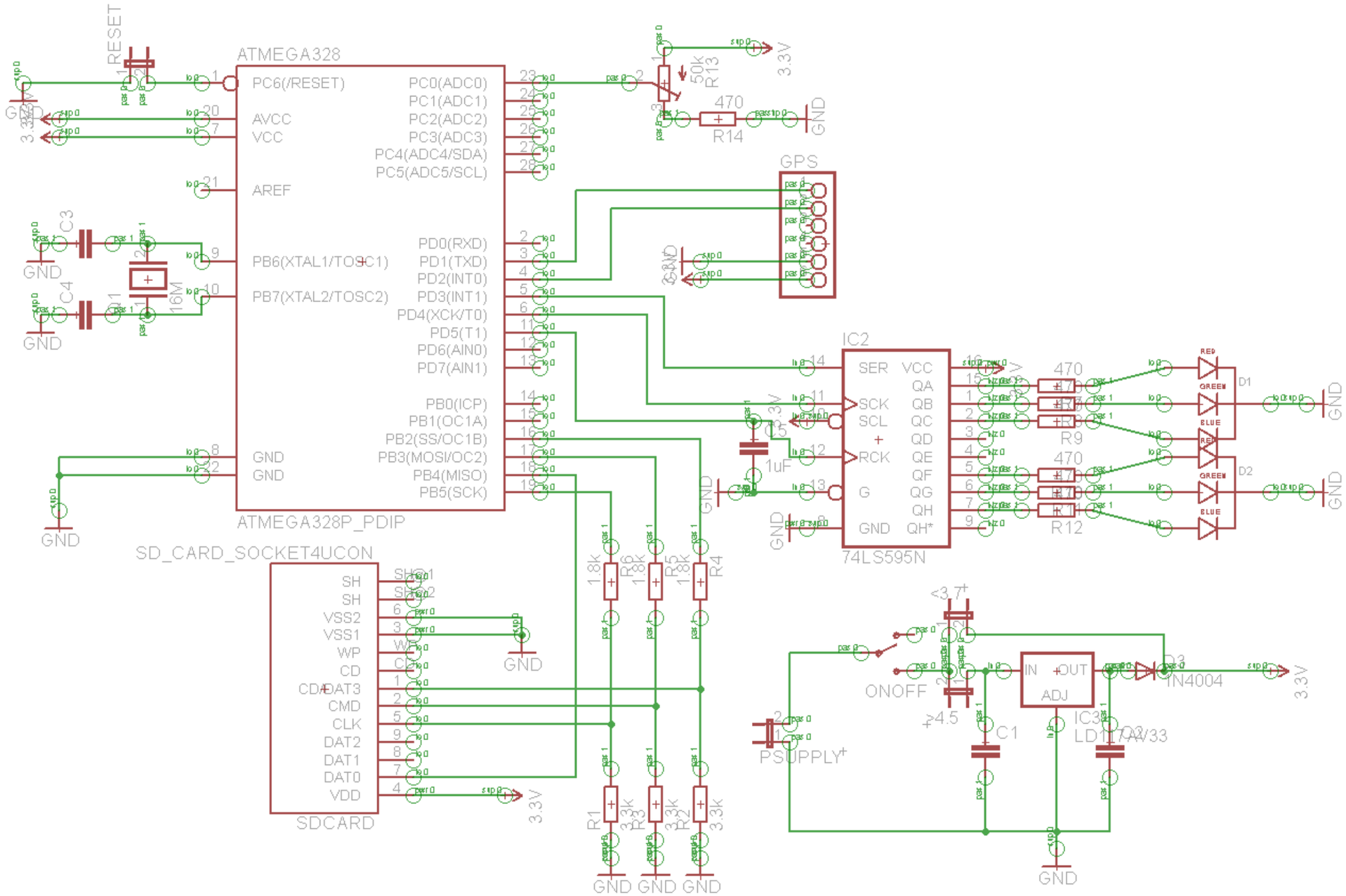
PCB related information:

http://en.wikipedia.org/wiki/Printed_circuit_board

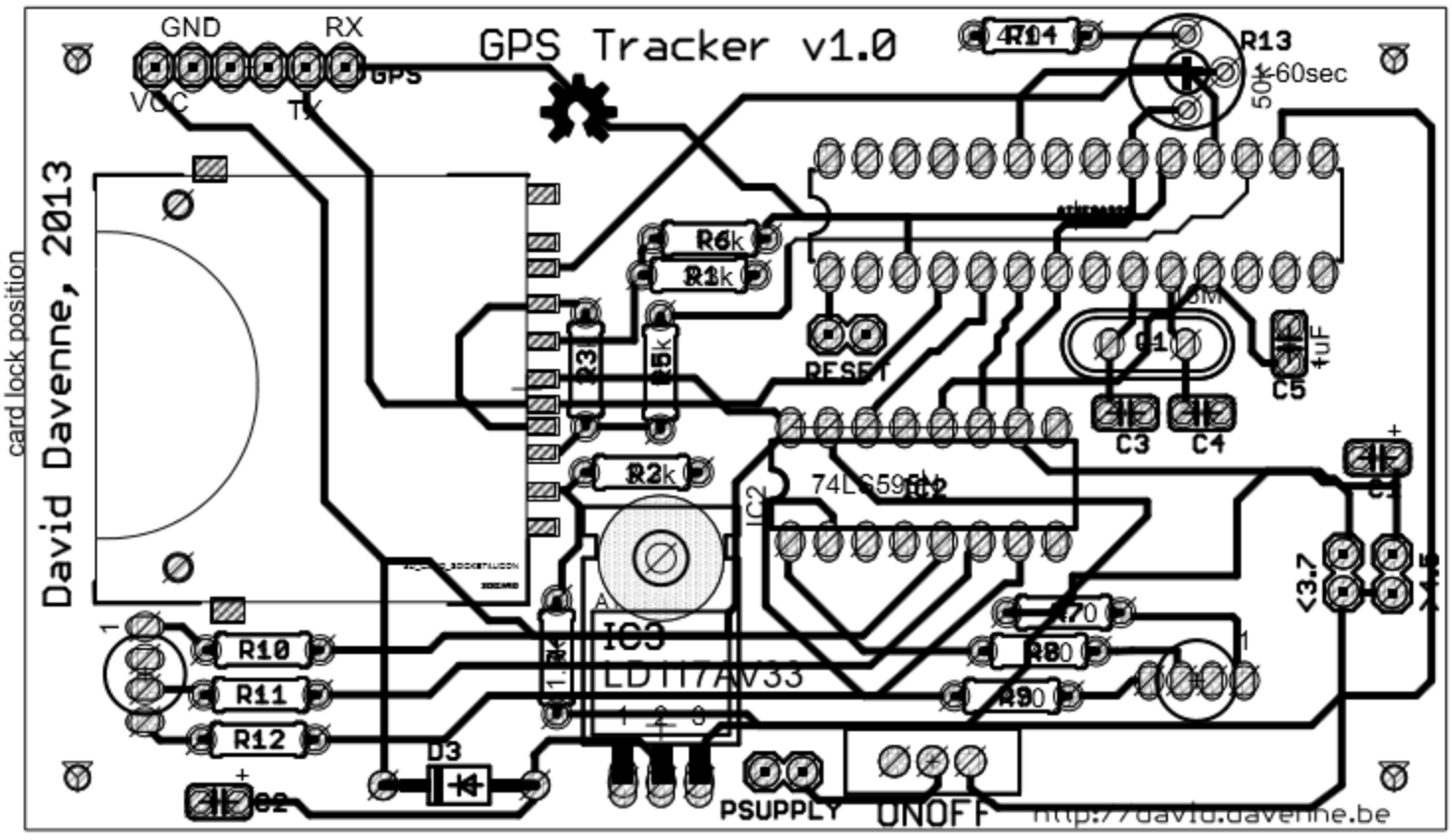
WebApp related information:

http://en.wikipedia.org/wiki/Twitter_Bootstrap
http://en.wikipedia.org/wiki/JQuery
http://php.net/

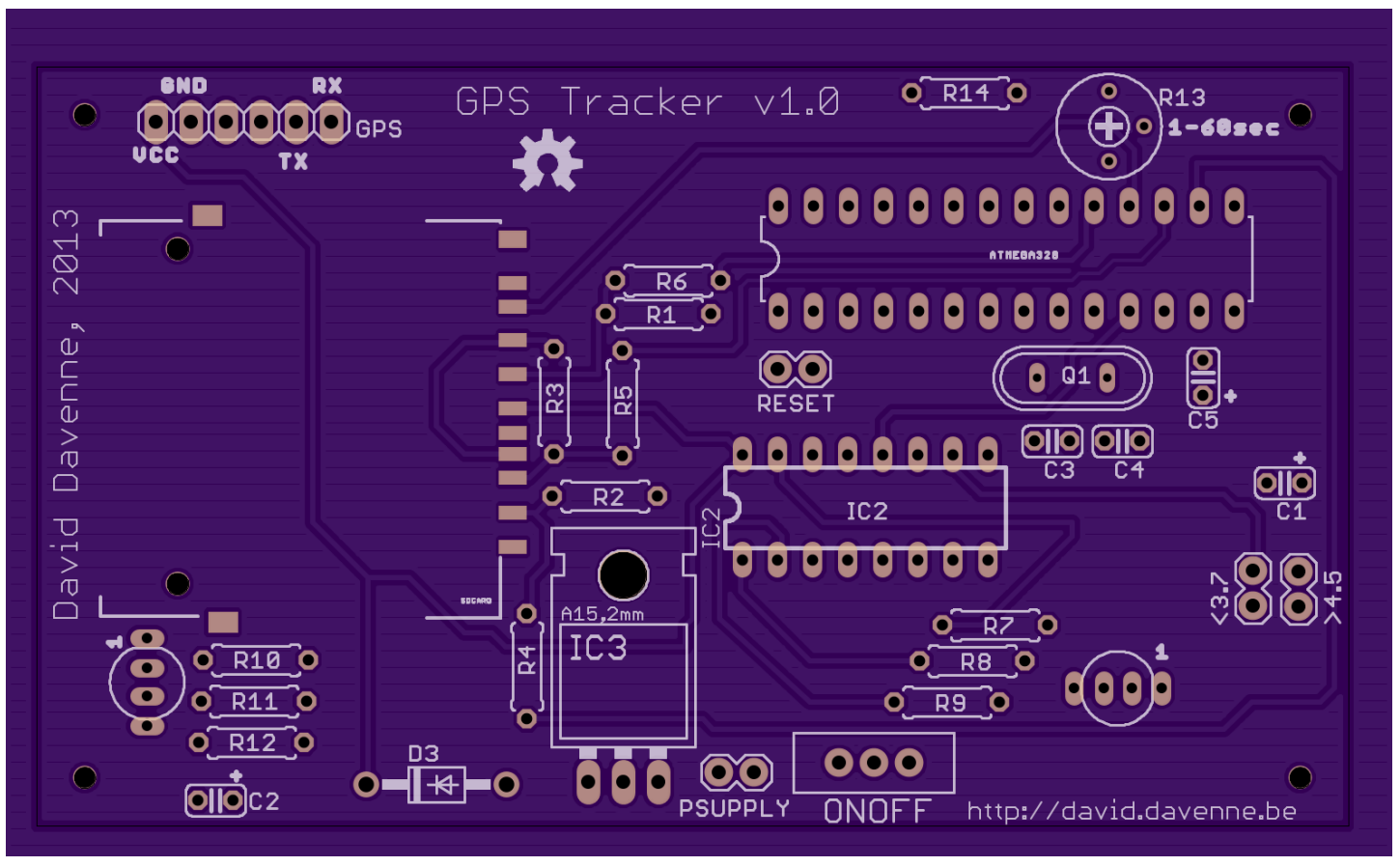
## Eagle Schematic



# Eagle PCB & OSH Park PCB

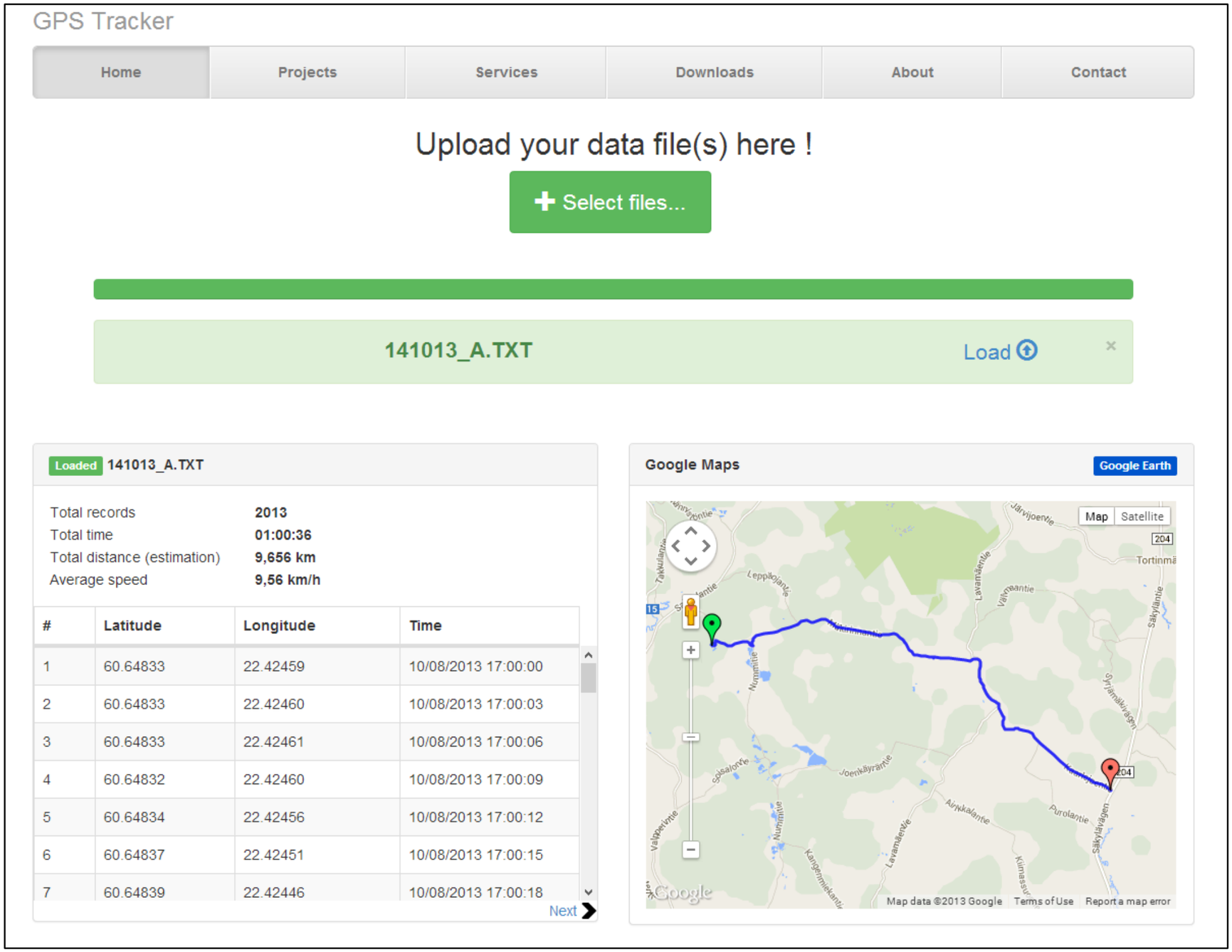



Eagle PCB




OSH Park PCB render

**Website's front page with tracker data loaded**




http://david.davenne.be/gps