NTNU – Trondheim
Norwegian University of
Science and Technology

Wireless Positioning and Collision Avoidance System

Karsten Fernholt Rennæs

Master of Science in Engineering Cybernetics
Submission date: July 2012
Supervisor: Amund Skavhaug, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Thesis Description

Assignment title: Wireless Positioning and Collision Avoidance System

Assignment text: The primary objective is to evaluate a possible solution for a stand-alone wireless positioning and collision avoidance system. The wireless module is intended for the use in combination to the NTNU project CyberBike (CB), but should also be capable of functioning as a individual module. The task will include looking into previous work on the area, acquiring necessary theoretical material, proposing and designing a solution and if time is sufficient producing and testing the solution. The goal is to keep the cost and complexity at a minimum, yet making it capable for future expansions and continued development.

Assignment targets:

- Differential Global Positioning System (DGPS)
- Assisted Global Positioning System (AGPS)
- Range measurement
- Wireless communication
- Expandable solutions

Assignment given: 23. January 2012

Supervisor: Associate professor Amund Skavhaug, Department of Engineering Cybernetics, NTNU

Abstract

Early in the 1980's Jens G. Balchen wanted to create an autonomous bike, capable of driving without any help from supporting wheels or human interaction. The intriguing idea included a variety of complicated concepts and was at that time almost an impossible task to accomplish. As time progressed and both technology and equipment developed, the possibility of a driverless bike becomes more than just an idea. The Norwegian University of Science and Technology (NTNU) has during the later years dedicated resources, time and effort in making a reality of the concept through the project named CyberBike. Every year, clever solutions are brought to the table, adding more functionality and better designs, bringing the project closer to a complete solution of an autonomous bike.

The main focus of this thesis has been to develop a system for the bikes positioning system, as well as collision avoidance. It also includes the communication made from the bike to a potential operator via wireless data transfer. The goal is to make a solution for the bike so that it could travel a given route, while communicating important data back to the observers. The task encompasses gathering the information made available by previous work, defining key areas of improvement and designing and testing a proposed solution.

First, the overall design is presented, showing how the two circuit boards made as the solution are connected with possible peripherals. The technical communication challenges pertaining the wireless communication is touched upon and relevant concepts are introduced. Furthermore, the selected microcontroller for the system is presented, giving key pointers in specific areas which might be confusing. Different possible devices are then discussed for the positioning system, the wireless communication and different setups of range sensors.

Testing done using the equipment explained in the thesis is presented, showing the results of the system. Improvements to the solution are introduced based on the experience obtained through the work, giving a solid basis for further work relating to the subject.

Preface and acknowledgement

This report is the result of the master's thesis *Wireless Positioning and Collision Avoidance System*. The thesis concludes my master's degree at the Norwegian University of Science and Technology (NTNU), under the Department of Engineering Cybernetics. The work was performed during the spring semester in the 5th year of the Master of Science study.

Aimed at being a helping guide for future students which participate in the CyberBike project or as a source of inspiration on other projects, the report is written to help simplify the understanding of designing prototypes.

I would like to thank my supervisor, Associate professor Amund Skavhaug for continuous guidance and motivations, not to mention interesting conversations and helpful insights. I would also like to thank John Olav Horrigmo for the enormous help I got during the endless days at the ITK workshop.

Trondheim, July 03, 2012

Karsten Rennæs

Contents

List of Figures

List of Tables

Glossary

ADC Analog To Digital Converter. 36

AGPS Assisted Global Positioning System. 1, 10

CB Cyberbike. 1, 3

DGPS Differential Global Positioning System. 1, 9

ERP Effective Radiated Power. 12

FTDI Future Technology Devices International. 20

GPS Global Positioning System. 4

I/O Input / Output. 18

IR Infrared. 35

JTAG Joint Test Action Group. 19

LED Light-emitting Diode. 20

MCU Microcontroller. 18

MHz Mega Hertz. 18

RC Combination of resistors and capacitors. 20

SPI Serial Peripheral Interface. 18

TTFF Time To First Fix. 10

TWI Two-wire Interface. 18

UART Universal Asynchronous Receiver/Transmitter. 18

UAV Unmanned Arial Vehicle. 3

V Voltage. 18

WiFi Wireless data exchange. 26

WSN Wireless Sensor Network. 3

Chapter 1

Introduction

1.1 Motivation

Today, in both the industrial and commercial market the focus is gradually shifting towards wireless communication. Not only is the reduced cost based on less cabling an important motivator, but it also offers a more tidy and practical solution, especially in home appliances. Everyday, new areas of interest towards wireless sensor networks (WSN) are discovered as cost and size is reduced and functionality increased. Combining WSN with autonomous functionality creates a powerful tool in which the area of use is almost only limited by the imagination of the designer. Unmanned vehicles, fully integrated home solutions (power, lights, alarm, music, etc.) and portable control stations for industrial systems are but some examples on what is possible to achieve.

The NTNU project CyberBike (CB) was initially the basis for this thesis. In order not to be limited by the dependency of other modules of the project, this task was formed as a stand-alone solution which also could be used in other projects. The main goal was to design a positioning and collision avoidance system for the bike, but the approach has been on a more general matter to keep the options for expanding even more freely and not as a specific solution to the CB.

1.2 Previous Work

Table 1.1 list previous relevant work done on subjects relating to the scope of this report.

Title	**Author**	**Year**
General Platform for Unmanned Autonomous Systems	K. R. Skøien and H. Vermeer	2010
Remote Presence on Offshore Wind Turbines	Viktor Fidje	2010
Ground Station and Hardware Peripherals for Fixed-wing UAV: CyberSwan	Mikael Kristian Eriksen	2007

Table 1.1: Previous work relating to this thesis

1.3 Areas of Improvements and Goals

Using the material from previous works, in combination with my ambitions on the subject, the following list states where improvements can be achieved. In addition some new fields of interest are added to complement earlier work and widen the application area:

- **In-depth configuration and usage guide:** During the start of a new PCB-project (printed circuit board project) a considerable amount of time is spent on getting to know component layout, their datasheet and how they work. As the matter of fact, many of the components used in this thesis have been used in several of the previous work posted in table 1.1. Unfortunately they are in lack of a useful guide and the risk of doing the same mistakes as they did is great. To spare future students of needlessly spending time on areas which have been done before, I wanted to create a thorough and detailed presentation of the components.
- **Create a working prototype:** As a motivating factor it was desirable to create something that actually worked and could be tested. Even though production is a time consuming phase, this is where you truly learn how the components work, giving the process value. A design may seem perfect when it is still just a design, but will show its true potential and flaws when produced.
- **Create a module for expansion:** Many earlier projects are specifically manufactured for a given task and got little or no room to be modified in order to perform different tasks than the original one. In contradiction, the module made through this thesis is designed to handle further expansions. Future students will then be able to continue using the design, adding the functionality they desire, without the need to start all over.
- **Implementing DGPS:** The use of a single global positioning system (GPS) unit has been suggested in several of the previous works done on the CB. Mostly it has come down to the drawback of inaccurate readings and concluded that it has not been sufficient enough to control a vehicle. By adding another GPS unit to the system, taking advantage of the added accuracy in using DGPS (explained in section 3.1), one should be able to rely on the position well enough to control a vehicle.
- **Implementing AGPS:** Assisted GPS is a term referring to the use of triangulation of the cellular towers to obtain a faster fix on the position. Using only AGPS would not result in a higher accuracy of the position, but a faster fix. Combining DGPS and AGPS would create a more rigiours system in terms of positioning. AGPS is explained in section 3.2.

1.4 Overview of Proposed Solution of the CB

The product created through this thesis is a module planned to be used with the rest of the CyberBike system (See figure 1.4). Functioning as an individual unit, the solution made in this task communicates via wireless transmission to a computer. It is also connected to the main system of the CB in order to communicate data from both the collision avoidance system and the positioning system. Using these data the Bike Main System is able to calculate the necessary response in case of obstacles and/or changes in the scheduled route.

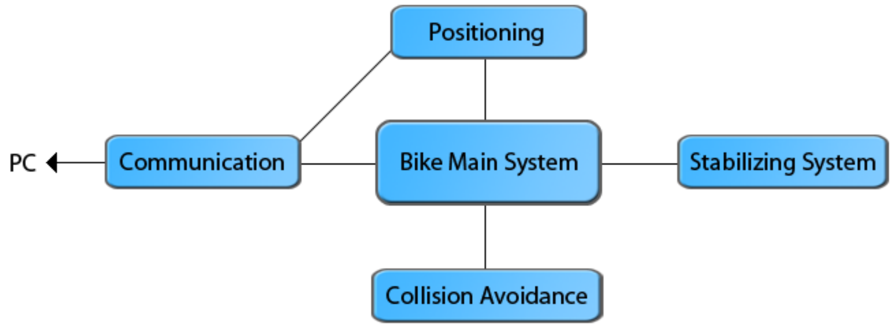


Figure 1.1: Consisting of the 5 main modules, the CyberBike would be a fully autonomous vehicle capable of travelling a given route, while avoiding obstacles and reporting data wirelessly back to an operator.

1.5 Report Outline

Below is a summary of what each chapter contains and the primary focus. Even though the selections made throughout this thesis have been thoroughly considered, there are always some problems which does not surfaces until the modules are being tested. It should therefore be noted that several of the chapters include a section named *Alternatives and Improvements*. Here, reflections on what could have been done different in the given solution is presented, in addition to alternatives to the chosen component.

Webpages that are referred to in the text are mostly given as a footnote on the corresponding page. The most important code segments and procedures can be found in the Appendices, whilst other relevant information is placed on the attached DVD.

Chapter 2: Background

This chapter describes the reason why the task was chosen in the first place, and how the idea of the concept behind CyberBike came to life. It also includes my intentions with the task, and what I want to achieve and accomplish during the thesis.

Chapter 3: Related Theory

In this chapter relevant theory behind the concepts used is explained. Challenges concerning wireless transmission is also presented.

Chapter 4: Overall System

In this chapter the overall system is presented. How each segment is connected to each other and the different types of interfaces are displayed, as well as the tools and equipment used during the thesis.

Chapter 5: Components

The different components which are suggested for the solution are explained and described. Alternatives and improvements to the different units are also discussed. For the components which have been implemented and tested in the solution there are some key pointers added on what to look out for.

Chapter 6: Electrical Design and Production

The structure of the solution is explained in this chapter. It also gives a brief introduction on how to think and plan ahead when starting on a new PCB project. Different types of production is also mentioned in this chapter.

Chapter 7: Graphical User Interface

The program used to control the vehicle used in combination with the boards is explained. How to operate the program, and how it is structured is presented in this chapter.

Chapter 8: Testing

How the system was tested and the most significant results are presented in this chapter.

Chapter 9: Discussion

This chapter presents discussions and evaluations based on the obtained results and experiences.

Chapter 10: Conclusion

The thesis most important achievements are summarized here.

Chapter 11: Further Work

Suggestions and recommendations towards future work based on the result of this thesis are presented in this chapter.

Chapter 2

Background

2.1 The Bike

In the 1980's the idea for the CyberBike came to life when founder of the Department of Engineering Cybernetics, Jens G. Balchen, wanted to create an unmanned autonomous bike. The idea was to create a vehicle capable of driving by itself while balancing without any form of support made by extra wheels. The project has been undertaken in various forms by several students since it first came to life. Mostly the focus has been on creating a system to regulate the balance and computing correct propulsion and turning.

Amund Skavhaug has been a part of this development and expressing the lack of a positioning system and a way to 'see' the surroundings led to the creation of the idea behind this thesis. In order to navigate the bike safely, it was necessary to prevent crashing into obstacles (i.e. people, curbs, trees, etc.) and following a path, preferably a road, while keeping its balance. The task of this thesis then became two-folded; firstly, creating a system to calculate an accurate enough position of the bike. Secondly, observe the surroundings of the bike in order not to crash into obstacles.

2.2 Assumptions and Simplifications

The turning radius and maneuverability of the bike is somewhat limited. It should not be assumed that the system could avoid any kind of obstacles at any given speed. Creating a system which would handle e.g. a running person would be overambitious and is left for future work. The focus is rather to observe static obstacles, calculate various options and taking the appropriate measure to avoid it. As it is discussed in [Sølvberg(2007), p. 3] there is also made simplifications to the model of the bike, giving slight deviations from the real world, which can create offsets on the bikes position.

The range of the wireless system planned for the bike is not intended for out-of-sight operation. Especially in areas with dense buildings, interference is a well know fact, creating irregularities for the transmission. The system suggested for wireless communication should therefore be based on line-of-sight operation.

2.3 Intentions

My intension with this thesis is to create a prototype of my own. By doing so I do not mean to discard previous work done on the subject, but rather minimizing the dependency of finished products and complete solutions. In my opinion the learning curve is steepest when you get the hands-on experience and really have to indulge yourself in the components. By creating a prototype all from my own design I wanted to get to know the process from when the first idea is created to the working, physical product is tested.

Chapter 3

Related Theory

3.1 DGPS

Just using a single GPS receiver is common in most cases where the normal accuracy[1] (20 meters/66 feet [Press(2004)]) of the system is sufficient. This is what is used today in the typical commercial GPS one might buy for the car or the one placed in mobile phones. Using this setup, an average of the most recent readings is calculated and displayed as an area where the person or object is located. Trying to find a building or a street, the accuracy will be good enough. For industrial use, like road or building construction, the accuracy has to be far better, often as precise as <1cm. To achieve such high accuracy the method called differential GPS (DGPS)is used. DGPS requires one roving receiver and one receiver fixed at a known location relatively close to the roving. Observations made by the fixed receiver are then subtracted from the correct position and transmitted to the roving. Corrections are then made to the coordinates the roving unit records and a high accuracy position is acquired (depicted in figure 3.1).

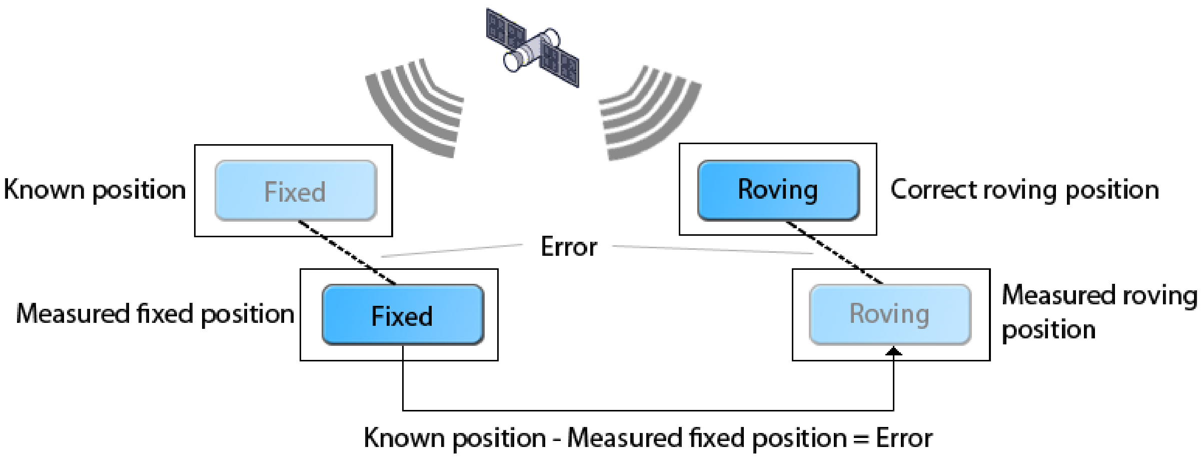


Figure 3.1: Error of measured fixed position is sent to the roving unit. Measured roving position is corrected and correct position acquired.

Because the inaccuracy is cause by interference along the path from the satellite to the ground it is imperative to maintain as short as possible distance between the two units. If kept close

[1]Bill Clinton ordered May 1, 2000 the Selective Availability to be turned off, improving the precision of civilian GPS from 100 meters to 20 meters

enough the idea is that both receivers will get the same error. The further away the two units are placed, the greater the error will become.

3.2 AGPS

Assisted GPS (AGPS) is used to enhance the startup performance of a GPS stallite-based positioning system. This is also known as the time-to-first-fix (TTFF) and refers to the time it takes from the system is powered on to when all necessary data has been acquired and a position can be calculated. To understand why the use of AGPS is useful, an explanation on how crucial information is retrieved from the satellites is necessary, starting with the navigation message.

Navigation Message

The navigation message consists of the three following parts:

- **GPS status:** Containing each satellites date and time, as well as the current status of the satellites and an indication of its health.
- **Ephemeris:** Is referred to the position of any astronomical object in the sky at a given time. This is used by the receiver to calculate the exact position of the satellite and is crucial to transform the data received to a position on the ground. The ephemeris is valid for a maximum of 4 hours from first received.
- **Almanac:** The almanac is a rather large amount of data which contains the information of the satellite network in total. When used, the almanac gives the approximate orbit for each satellite and is considered valid for a maximum of 180 days.

The almanac, which is a more general information package than the ephemeris, is downloaded from any satellite in range of the receiver. From this data the receiver can extract which of the 24 satellites in the sky that are in potential reach and determine which to search for. The receiver then downloads each of the potential satellites ephemeris in turn directly from the satellite. If a download has been initiated and the satellite orbits out of reach, all data is discarded and a new satellite must be chosen. Unless the complete and total package of the ephemeris has been downloaded the receiver can not calculate the position.

The navigation message is made up of a 1,500 bit frame and can be downloaded individually from different satellites. Each of the navigation frames is divided into 5 subframes, each containing 300 bits and is transmitted at 50bits/s. The subframes are arranged as in table 3.1. A complete almanac message consists of 15,000 bits and all must be downloaded in order to start the search for potential satellites. As the download has a bit rate of 50bits/s a total of 12,5 minutes is necessary in order to download the complete navigation message from a single satellite.

As each satellite transmits its own ephemeris, but the almanac for all satellites in the network the receiver can potentially download the almanac from different satellites. If this is the case the timing of the almanac frames could result in a longer download time than the 12,5 minutes, which is a minimum. Even though the almanac is considered to be valid for 180 days, cheaper GPS receivers might not be able to store the almanac when powering down for longer periods. Therefore a new download might be necessary, creating a longer TTFF when starting up again. In order to minimize the time needed to acquire the latest almanac the use of AGPS comes in

Subframes	Words	Description
1	1-2	Telemetry and handover words
	3-10	Satellite clock, GPS time relationship
2/3	1-2	Telemetry and handover words
	3-10	Ephemeris
4/5	1-2	Telemetry and handover words
	3-10	Almanac component

Table 3.1: Navigation message

handy. Instead of relying on the rather slow bit rate delivered from the satellites the almanac can be downloaded using the cellular network. At a potentially much higher speed the GPS receiver can be handled the almanac through the use of a GSM module.

3.3 Wireless Challenges

3.3.1 Multipath Propagation

When using wireless communication the signal can reach the receiver through numerous paths after being transmitted from the source. In contradiction to cabled system where the signal only had one distinct path to follow, the wireless signal will reflect of various objects like houses, trees and walls and reach the receiver at different times, because they are travelling paths of different lengths (see figure 3.2). Any system based on wireless communication is prone to experience this phenomenon, called multipath propagation. Even though the targeted receiver is in line of sight of the transmitter, each of the signal paths will have individual amplitude, angle and delay and can cause complications at the receiving end when trying to interpret the signal. If there are no specific control functions implemented to distinguish between the signals, a simple receiver would not be able to separate the incoming data, but rather just add it up. As different signal flight-times could lead to phase shifting, the accumulated signal can either be constructive or destructive. A constructive signal would result in amplitude variance, whereas a destructive signal would, in the worst case, nullify the signal.

The effect of multipath propagation is not only relevant when communicating on the ground, e.g. between two radio transceivers, but also when communicating with satellites in the sky. As the GPS system heavily depends on the correct timing of the signal, multipath propagation could lead to deviations in the measured position. When using a GPS within tall buildings, trees or mountains, or in confined spaces in general, the risk of experiencing multipath propagation is large.

3.3.2 Shadowing

When working with mobile units, wireless communication problems might occur when objects like houses and high-rises block the line of sight. Even if the signal finds a way around or through the object blocking the path, the signal would attenuate and the amplitude would significantly decrease. It is said that the intended receiver is in the radio shadow of the blocking

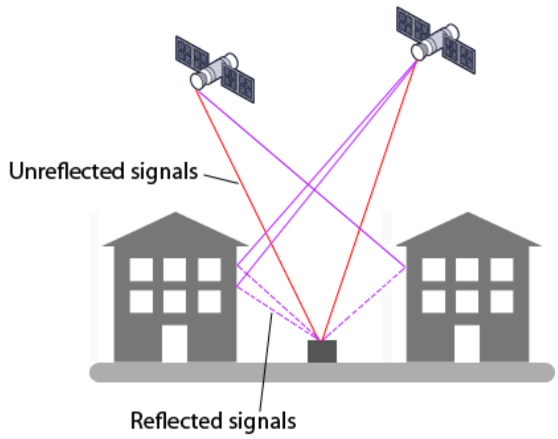


Figure 3.2: Multipath propagation: Red lines represent the unreflected signals, whilst the purple are reflected of the buildings, creating longer propagation time.

object, and the phenomenon is known as shadowing. How much the received signal will deviate from its original form is of course dependent on several factors like signal flight-time and distance from the target.

3.3.3 Spectrum Limitations

As the utilization of wireless communication has been around for a little more than 150 years, regulations on how to divide the frequency spectrum is a necessity to avoid chaos. In Norway this is carried out by the Norwegian Post and Telecommunications Authority (NPTA[2]) which divides the available spectrum into sectors depending on demand and priority. Manufacturers who want to introduce a new wireless product to the marked must apply for the appropriate frequency range and is accepted if the specification is in accordance with regulations. In addition, geographical location and availability is considered. Transmitting on a restricted frequency is illegal and could result in decreased throughput, or in worst case, a total malfunction of the given frequency range. NPTA therefore continuously publishes an overview of allocated frequencies.

When producing prototypes like the one made in this thesis it is important to choose wireless products which transmits on a legal frequency. In Norway there are sections of the radio spectrum reserved for amateur use and the components must operate within these sectors when tested. In figure 3.3 the 2,4 GHz (ZigBee, WiFi) and the 433 MHz (RC1240) spectrum is displayed, respectively from top to bottom.

3.3.4 Energy Limitations

In addition to having limitations on the frequency spectrum there is also a restriction on the maximum effective radiated power (ERP) allowed to transmit with. As mentioned in the previous section, NPTA divides and distributes the spectrum based on geographical location. To prevent units from exceeding their delegated area and to keep the issue under control, ERP is tested when a new product is approved and also on a regular basis. In a wireless system, the

[2]Post- og teletilsynet: http://www.npt.no

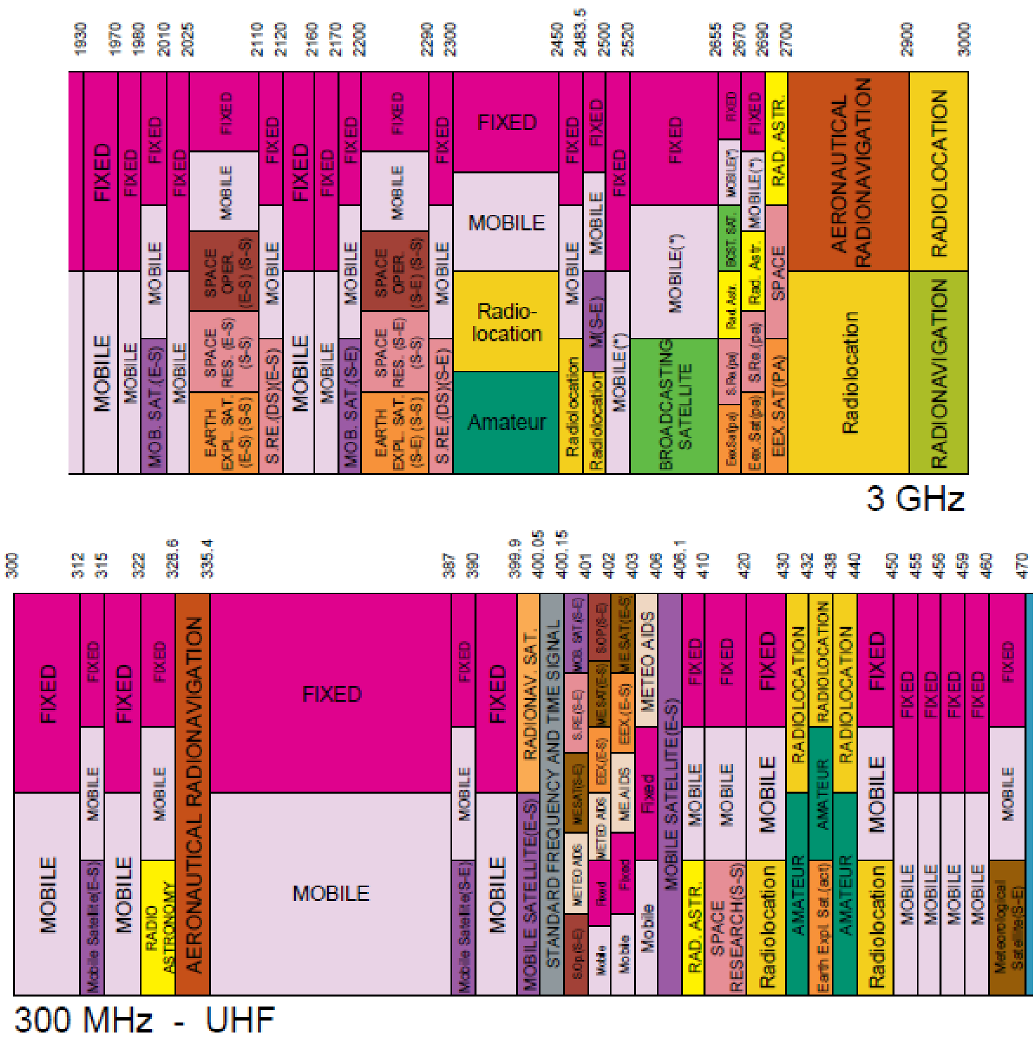


Figure 3.3: Ranging from 2,3 to 2,4 GHz the amateur sector contains the ZigBee products. Partly covering sectors from 430 to 440 MHZ is where the Radiocrafts RC1240 is found.

design might rely on the use of a mobile station (MS) or a portable unit, without the possibility to be tied to a power net. The requirement for low energy consumption then plays a crucial role in the design of the circuit and/or system. The unit should be able to maintain the required sending distance as well as the given life endurance whilst considering the limitations on both size and weight.

In Norway the maximum ERP is set to 10mW. The Radiocrafts RC1240 is by default factory settings limited to 6mW, whereas the ZigBee Xbee and the ZigBee Xbee-Pro protocol is set to 2mW and 10mW, respectively.

Chapter 4

Overall System

4.1 Overview

The idea behind the whole design is to make a versatile system which can handle expansion both in functionality and size. The system as it is today, consists only of two modules - a master and a slave. Both of these are designed from the same basis to create simplicity in production and programming. But, if desirable, it can easily be expanded to work with several other units, and the components were chosen to also allow more functionality being added at a later stage. Given that each module is made from the same basis design they can all work as a stand-alone unit, working as either a master or slave, or it necessary, a relay for other units. In figure 4.1 the two modules are shown with connected peripherals. The connection named *Ext. system* is planned to function as either a second slave of the same design or a totally different system.

It should be noted that the GPS and RF components were implemented and tested, while the range sensors only were tested. The IMU is yet to be tested and implemented, but [Sølvberg(2007)] has done exstensive testing on one type compatible to the board.

In figure 4.2 one single modules is depicted, showing the interfaces to each component.

4.1.1 Power Supply

The reason the standard USB connection was chosen as interface to the computer was to eliminate the need for separate power supply. Standard USB 2.0 can supply connected units with power through the same cable as the data signal, but is limited to 500mA due to safety reasons for the connected peripherals. When connected and using the GPS and the RF at full capacity the total power consumption is at 270mA, well below the limit of the USB port, and still with potential for additional units. If more components are connected to the host board, exceeding the maximum power limit of the USB port it is possible to attach a secondary power supply to the board. This is done using the power connector pins (Shown in figure 6.1 in section 6.1). The design is made to handle 5V on these pins. If the board operates as a slave and using battery, this is the same power supply connection.

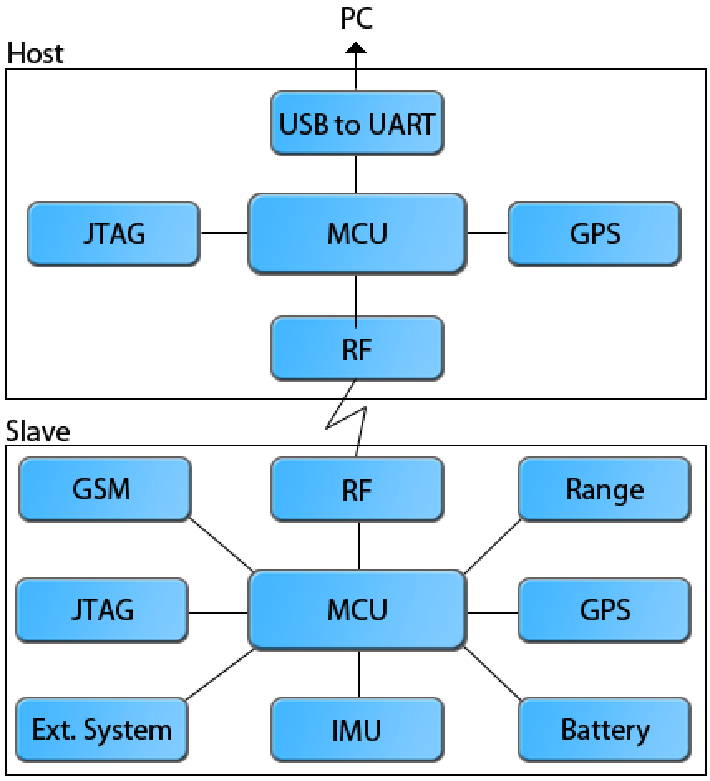


Figure 4.1: Overall System.

4.2 Tools

4.2.1 STK600

The STK600 is a development system supporting all 8- and 32-bit AVR devices. The STK600 kit includes multiple expansion boards which makes it possible to attach DIP packages to the board for programming. In the case of this thesis the JTAG connection found on the board was used to program the AtXMEGA256A3 externally. If needed, the STK600 can supply the attached component with power using a single jumper to engage this. The board also includes several standard 10-pins male headers, as well as buttons and LEDs for easy debugging and testing. In addition, the STK600 provides a RS-232 level converter made available through the UART pins using jump wires. This enables the device to communicate with a computer by using a serial port terminal application.

4.2.2 Boards

The two boards depicted in figure 4.3 were created for this thesis. They are made specifically for this thesis and produced at the ITK workshop. Design, setup and production is explained in chapter 6. The board on the left of figure 4.3 has some added functionality for when it is attached to the car (described in section 4.2.3). Extra LEDs are included to indicate the wireless transmission, in addition to extra pin headers for connection of the motor and servo.

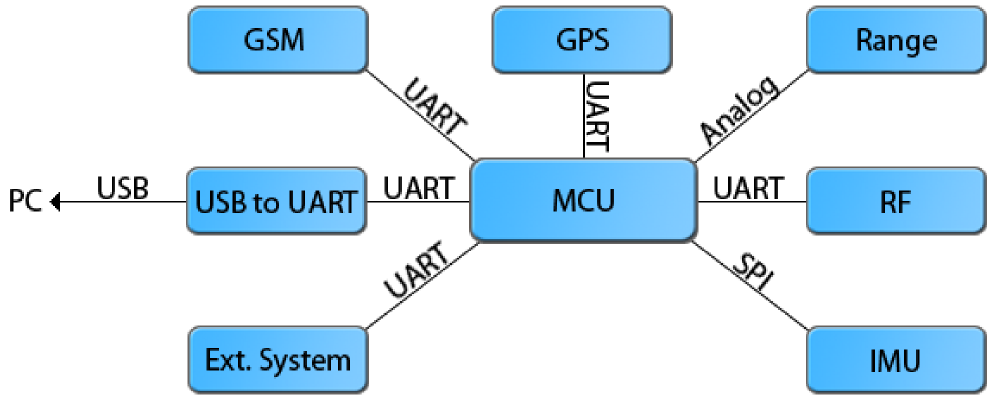


Figure 4.2: Overview board with connected peripherals.

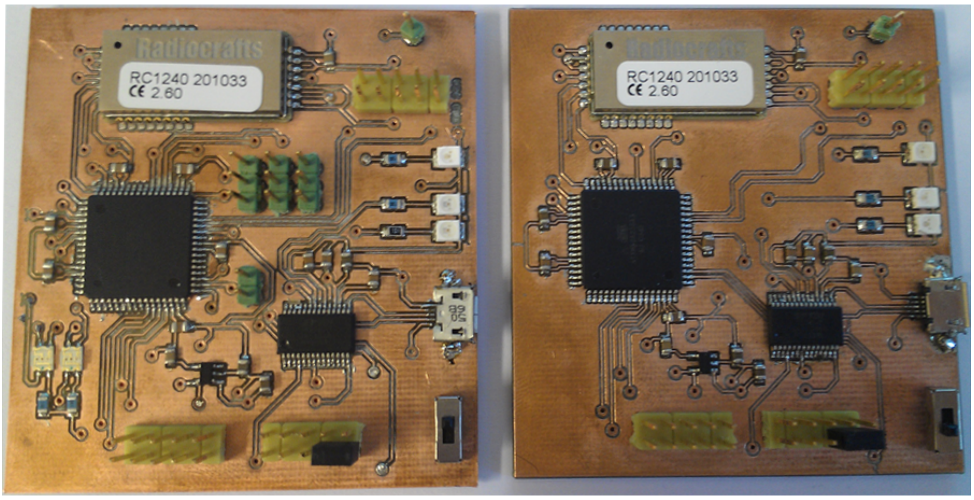


Figure 4.3: The two boards made at the ITK workshop

4.2.3 Car

Instead of testing the solution on the actual CyberBike a replacement vehicle was used. The remote car eCrasher displayed in figure 4.4 consists of a 5V high torque brushless motor for propulsion and a single servo for steering. The motor is controlled by a Dragster Sport Motor Controller which inputs a PWM signal. A battery of 4500 mAh supplies the car with power and also the board if attached, outputting 5V from the motor controller. Unfortunately the eCrasher crashed (!) during test runs and one of the supporting ledges was broken, rendering the car useless until spare parts could be found. The crash happened relatively late in the thesis period and further testing was therefore suspended.

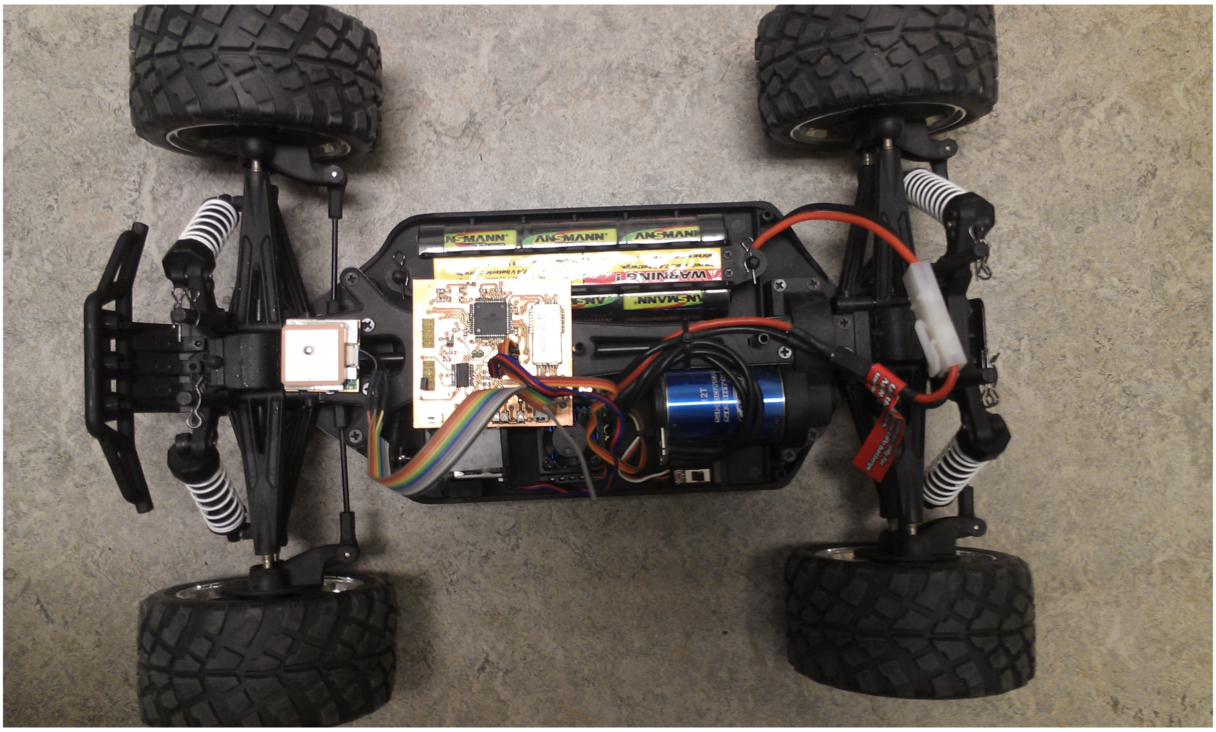


Figure 4.4: Test car eCrasher connected to the board

Chapter 5

Components and Concepts

5.1 MCU

AtXMEGA256A3 Key Features

- Flash: 256 kB
- Pin count: 64
- Max Operating Frequency: 32 Mhz
- Max I/O pins: 50
- External interrupts: 50
- SPI: 10
- TWI (I2C): 2
- ADC channels: 16
- UART: 7
- EEPROM: 4096 bytes
- Operating voltage (Vcc): 1,6 to 3,6

5.1.1 AtXMEGA256A3

The ATxmega256A3 is a high performance, low power 8/16-bit microcontroller from Atmel which has a wide range of functions. Equipped with several I/O interfaces like SPI, UART and TWI the AtXMEGA256A3 is capable of handling a variety of peripherals. 50 out of the total 64 pins functions as I/O pins, all with external interrupts implemented. Operating at 1,6 - 3,6V the AtXMEGA256A3 can power down applications in order to minimize power consumption, running on only 5μA in power-down mode. Integrated clock generators, ranging from 32kHz to 32MHz, reduces the need for external clock generators and simplifies the design as a total.

One of the main reasons the AtXMEGA256A3 was chosen as the MCU for this thesis was the high performance of running at 32MHz in combination with the wide selection of on-board functions. As of today the board design is far from exceeding the possible performance of the MCU, keeping the doors open for future students to continue on the same design whilst adding more functionality to the system.

Given the intended board design depicted in figure 4.2 there is a need for 5 UART connections to the MCU. With the lack of chip-select signals in UART there is a need to either add a hardware

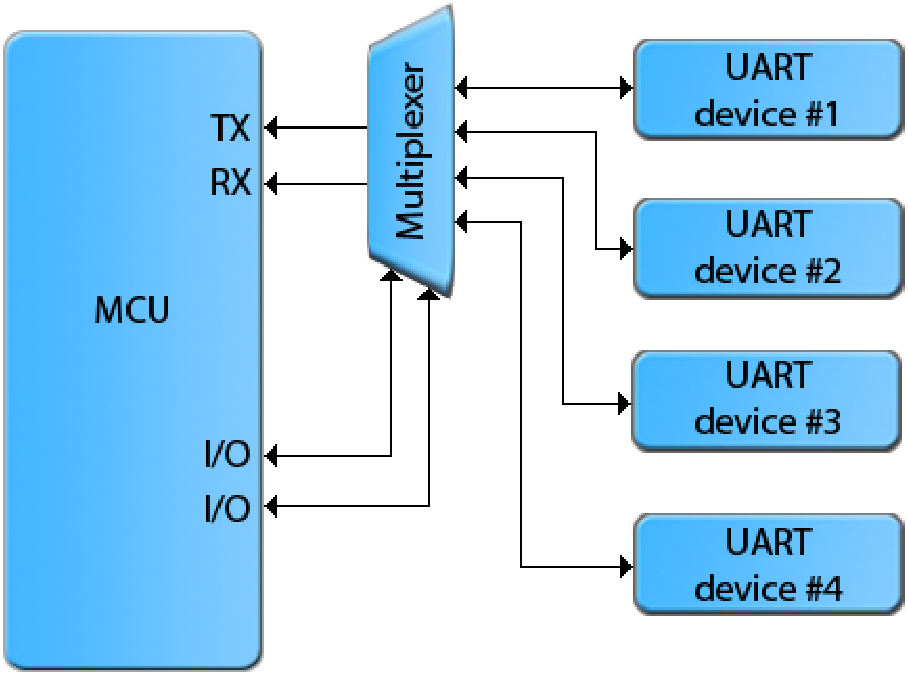


Figure 5.1: Multiplexer setup to handle more UART connections

multiplexer or choosing a MCU with enough UART connections to support the design. A setup as depicted in figure 5.1 was considered to handle several UART peripherals on a MCU with only one UART connection. Due to the added expenses to the design cost, as well as the added complexity of the circuit it was found a better choice using the AtXMEGA256A3 which has 7 UART connections implemented.

Being familiarized with the Atmel products through various projects during the years at NTNU it was an obvious choice to stay within their product portfolio. Atmel also offers the freeware AVRStudio 5[1] which simplifies the programming of the MCU via JTAG. Given the relation between Atmel and Omega Workshop spare parts are easy accessible if something should go wrong and the MCU has to be replaced. Several of the members of Omega Workshop also has extensive experience with the Atmel products and gladly offers comments, tips and help if needed.

5.1.2 Usage

As with most of Atmels products, the AtXMEGA256A3 is thoroughly described in both datasheets (simplified[2] and extended[3]) and application notes[4], and can be found on their homepage[5].

Even so, there is one important and rather crucial point of interest which they have not described very well. That is the concern on how to configure the clock rate on the chip. As a standard

[1] http://www.atmel.com/microsite/avr_studio_5/
[2] http://www.atmel.com/Images/doc8068.pdf
[3] http://www.atmel.com/Images/doc8077.pdf
[4] http://www.atmel.com/PFResults.aspx#(data:(category:'34864[33180[33083[32208]]]',type:!(13)),sc:3)
[5] http://atmel.com

the AtXMEGA256A3 runs on 2 MHz internal RC oscillator, and does have the ability to also use the internal 32 MHz RC oscillator. When communicating with peripherals over UART and the desired baud rate is exceeding 9600 b/sec it is necessary to use a faster oscillator than the default one. To change the oscillator the system critical I/O register settings has to be engaged. To prevent accidental modifications to these registers there is a 2 step procedure:

1. The application code writes the signature for change enable of protected I/O registers to the CCP register.
2. Within 4 instruction cycles, the application code must write the appropriate data to the protected registers.

Because step 2 in the procedure has to be completed in 4 cycles and the optimization in AVR studio 5 often makes this difficult to achieve, the safest way is to program this step using assembly code. A working code segment for configuring the use of 32 MHz internal clock oscillator is included in Appendix B.

5.1.3 Alternatives and Improvements

The ATxmega256A3 was compared to the older ATMEGA644P which would also be sufficient in many of the areas required, but had limitations on the amount of possible expansions. The ATMEGA644P can only handle two UART connection to external peripherals and was therefore discarded as it would be desirable to include more components than just the two.

For some, the ATxmega256AU3 might seem as a better choice as this MCU also has a direct USB connection implemented. Because production of prototypes often result in a great deal of time searching for errors it was seen as a better choice to rather include the FT232RL (reviewed in section 5.2.1) than to go for the AtXMEGA256AU3. This way, an extra layer of security was added in terms of making sure the circuit was functioning correctly. Using the FT232RL in combination with LEDs produces a distinct blinking pattern when the USB/UART communication to the computer is working as it should. This would save a great deal of time wondering if the circuit was flawed, or if something was wrong with the UART connection.

As a last note, if the AtXMEGA256AU3 is chosen to be used, one must pay attention to the design rules regarding high-speed signals. If the differential USB signal travels over larger areas on the board there are special requirements with regards to spacing between the positive and negative signal, as well as the total length of the path. This is explained and discussed more thoroughly in [Koteich(2011), p. 70].

5.2 Computer Interface

5.2.1 FT232RL and UART protocol

The FT232RL from Future Technology Devices International (FTDI) is a USB to serial UART interface with internal clock circuit. Using the FT232RL one can easily create a bridge between standard USB and UART without the need for level converting or external clock generators. It is easy to solder by hand and has the ability to choose either 5V or 3.3V output voltages using only a single jumper.

In cases of prototyping the FT232RL has a clever attribute to eliminate potential errors. By connecting the pins *CBUS0* and *CBUS1* to LEDs it is possible to check whether or not the connection to the computer has been accomplished. If the virtual com port drivers (explained in section 5.2.2) has been installed correctly the LEDs will flash 4 times, two in sequence followed by a pause and the last two. Getting this confirmation also replies whether or not the FT232RL chip is functioning as planned and is a great tool to consider when designing circuits.

5.2.2 Installation and Setup

When using the FT232RL with Windows 7 it is necessary to install the drivers for virtual COM ports, but these are made easily accessible by Microsoft. When the drivers are successfully installed the FT232RL can be connected to the computer using a standard USB port and will be displayed as a COM port. Any terminal window with access to the COM ports can then be used to communicate with the FT232RL. In order to install the drivers first visit FTDIs webpage[6] and acquire the driver files. Then use the install guide[7] to complete the setup. Doing this will give the correct flashing pattern on the LEDs and the port is visible through terminals.

5.2.3 Buffer timing on computers

Sending data over UART and displaying it using a terminal window is usually an easy task. The problem with UART transmission first arises when there is a need to handle the data, and replying in accordance to what is received. The reason for this is the buffer handling on computers. As the processing speed of most computers today far exceeds the transmission speed of UART problems is create in timing when to read the computer buffer to get the correct data. Because this is a more software related issue the subject is handled in section 7.2.3.

5.2.4 Alternatives and Improvements

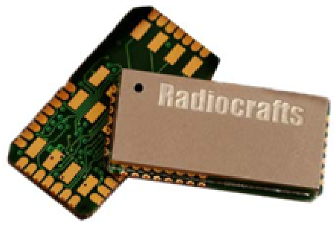
There are two fully plug-and-play components on the marked as of today, the FT232RL and the MCP2200 by Microhip. The latter is somewhat cheaper and offers most of the same attributes as the FT232RL. The main difference is that the FT232RL has the capability to deliver 3,3V or 5V to the connected MCU. This is not an option with the MCP2200 as this will only produce 5V UART signals. Given the AtXMEGA256A3 used in this thesis there would be necessary to implement another component, regulating the 5V delivered from the MCP2200 down to 3,3V. As the FT232RL would to this by only adding some more circuit and a jumper it came out as the superior choice.

[6]http://www.ftdichip.com/Drivers/VCP.htm
[7]http://www.ftdichip.com/Support/Documents/InstallGuides.htm

5.3 Radio Communication

RC1240 Key Features

- Small size (12,7 x 25,4 x 3,5 mm)
- Low power consumption, 26 mA in transmit mode
- Embedded RC232-protocol
- 128 byte buffer
- Simple UART interface
- Wide supply voltage range, 2,8 - 5,5 V
- Ready-to-go for single-channel use



5.3.1 Radiocraft RC1240

The RC1240 is currently the smallest narrowband transceiver module available, operating at 433,05 - 434,79 MHz. The compact shielded module is applicable for hand soldering and is ready to use out of the box. It is by default factory settings limited to a total of 6mW transmission power, a value that cannot be changed. Regulated transmission power for amateur unlicensed use is 10mW and the RC1240 therefore stays well within range of this. The RC1240 is embedded with the RC232 protocol and configuring the component is made easy by writing to the RX pin. Both the RC232 protocol and the component are comprehensively described and supported on Radiocrafts homepage[8]. Earlier projects like [Karlsen(2010)] and [Fidje(2010)] has used the RC1240 which gives a solid basis to work with.

5.3.2 RC232 Protocol

The RC232 protocol is an easy-to-use protocol embedded in several of Radiocrafts products, amongst others, the RC1240. The protocol offers a wide range of features for bidirectional wireless transmission and is easy to configure. The RC232 protocol can handle both point-to-point and peer-to-peer communication with addressing, or point-to-multipoint utilizing broadcasting. In addition, several useful properties are included like host communication, data buffering and error check. Combined these functions makes the RC232 protocol ideal for small and time crucial projects where generalized message sending is utilized.

5.3.3 Configuration of RC1240 using RC232-protocol

Firstly, there is a need to explain an important element which might create a great deal of confusion and frustration when trying to configure the RC1240. The RC232 protocol has a rather simple response format which deserve explanation. If a command is accepted and the correct mode is activated, the chip will respond with a ’>’. Unfortunately the chip also responds with a ’>’ if the command is not accepted/not recognized. To overcome this ambiguous respond one has to match the responsive pattern of each individual command sequence to determine if

[8] http://www.radiocrafts.com/index.php?sideID=241&ledd1=32

the command was stored as planned. An example in table 5.3 is added to help understand this procedure.

All communication to the RC1240 component is transmitted through the RX/TX pins, even the configuration. Before configuration the CONFIG pin must be asserted (pulled low). Once the pin is pulled low the chip enters COMMAND mode and parameters are accepted. In some cases the chip acknowledges by sending a ’>’ back to host MCU, this is extensively described in the datasheet [Radiocrafts(2012)]. It is important to notice that the CONFIG pin must be de-asserted before any more commands can be submitted, but after the initial response has been made. When the chip is in COMMAND mode there are two main possible forms of configuration. If the letter ’M’ is sent, the chip will enter MEMORY mode. All the following parameters in this mode will be store in non-volatile memory and kept even if the power is turned off. The second possibility is all other commands. These will make configurations to the volatile memory and lost if power is switched off. To exit the CONFIG mode and return to NORMAL mode an ’X’ needs to be transmitted to the chip. Table 5.1 shows an example on how to select RF channel 3.

Command	Hex	Response	Comment
CONFIG asserted		’>’	De-assert CONFIG after ’>’
’C’	0x43	’>’	
3	0x03	’>’	Wait for ’>’ prompt
(Alternative command)			
’X’	0x58	(none)	Module returns to IDLE state

Table 5.1: Configuring the RC1240 using the RC232 protocol

Memory mode

As mentioned, if ’M’ is submitted, the non-volatile memory is accessed. In table 5.2 the available commands one can enter during MEMORY mode is presented. After the memory has been configured, the command ’0xFF’ has to be transmitted in order to exit the MEMORY mode. To completely exit the CONFIG mode and return to NORMAL mode an ’X’ needs to be sent trailing the ’>’ response from exiting the MEMORY mode. An example is displayed in table 5.3.

Setup Example Code

To successfully write and update the non-volatile memory on chip there is specific timing and command rules to follow. Below is an example on the procedure on how to change the PACKET_END_CHARACTER (explained in section 5.3.4), in addition a code segment written in C is added in appendix A to clarify the command usage.

5.3.4 Usage

As the RC232 protocol is embedded in RC1240, sending and receiving data over the wireless link is made very easy. If the appropriate setup is initiated, the RC1240 chip will be seen as

Paramter	Description	Address hex	Default settings
RF_CHANNEL	Default RF channel	0x00	
RF_POWER	Default RF output power	0x01	0x05 (5)
RF_DATA_RATE	Default data rate	0x02	0x03 (3) = 4,8 kbit/s
PREAMBLE_LENGTH		0x0A	0x08 (8)
PACKET_LENGTH	Size of buffer	0x0F	0x80 (128)
PACKET_TIMEOUT	Buffer timeout	0x10	0x7C (124) = 2s
PACKET_END_CHARACTER		0x11	0x00 (0) = None
MAC_MODE		0x12	0x02 (2)
ADDRESS_MODE		0x14	0x02 (2)
CRC_MODE		0x15	0x02 (2)
UNIQUE_ID	Unique ID	0x19	0x01 (1)
SYSTEM_ID	System ID	0x1A	0x01 (1)
DESTINATION_ID		0x21	0x01 (1)
BROADCAST_ADDRESS		0x28	0xFF (255)
UART_BAUD_RATE		0x30	0x06 (6) = 19200
Exit CONFIG mode		0xFF	

Table 5.2: Parameters used in MEMORY mode

Command	Hex	Response	Comment
CONFIG asserted		'>'	De-assert CONFIG after '>'
'M'	0x4D	'>'	Module ready to receive paramter
17	0x11	(none)	
36	0x24	(none)	Sets end character to '$'
(Alternative command)			
255	0xFF	'>'	Wait for '>' prompt
'X'	0x58	(none)	Module returns to IDLE state

Table 5.3: Configuring the RC1240 using the RC232 protocol

a regular UART link to the host and receiver. Sending bytes on the RX pin of the chip will be transmitted to the receiving module and extracted via the TX pin. In order to makes this work, the setup must be chosen in accordance to the data flow of the given circuit. There are 3 different triggers which may be set for the module to transmit:

- Buffer is full. When the buffer reaches the value stored in PACKET_LENGTH the data in the buffer will be transmitted.
- The predefined time in PACKET_TIMEOUT is reached after the last byte is received on the RX pin.
- The predefined end character in PACKET_END_CHARACTER is received on the RX pin.

Note that all the of the mentioned triggers might be active at the same time, giving different transmission criteria depending on the data packet. On the other hand, only one of them has to be triggered in order to initiate the transmission.

Message sending using RC1240

Choosing the first option, transmission when the buffer is full, will often result in the need for bit stuffing. That is, if the data packets being sent are of variable sizes, one has to fill the rest of the buffer with garbage until the buffer size is reached and the transmission initiated. This creates a lot of unnecessary overhead and does not constitute as a viable option when the data packets are of different sizes.

The second option is in most cases a more suitable option, but could in worse case give undesirable delays or end up sending prematurely. If the PACKET_TIMEOUT is kept as in the default settings, to 2 seconds, an extra delay is added to the system. Unless the data is sent in frequency larger than this, the system suffers in regards to the extra time added for each packet sent. On the contrary, if the PACKET_TIMEOUT is set at a too low value the risk of cutting the packet short is a fact. The trigger might send the first part of the data stream, before all bytes have reach the buffer.

The most reliable and controllable option is the third and last one. Choosing a specific end character gives total control and timing is no longer of the same importance. When sending data using the RC1240 I implemented the use of 2 end characters. This is because the RC1240 does not include the end character in the data stream, but send all up to this character. To make sure the complete packet is received at the opposing MCU an extra end character is added. Choosing '\$' as end character for the RC1240 (as shown in table 5.3) and '&' for the MCU gives <data>\$& as the complete data packet. In figure 5.2 the complete data stream from host MCU to receiving MCU is displayed.

5.3.5 Alternatives and Improvements

When choosing what type of communication devices to use for the wireless transmission there are several factors to examine:

- Data transfer Rate
- Range from base to target
- Power consumption
- Size, weight and form
- Cost per node

The demand for stable and high speed wireless transmission is increasing due to demanding users and increased complexity in applications. There are several wireless standards on the market today, all with their own advantages and disadvantages. During this thesis the three following options were focused on when deciding which direction to take:

- Radiocraft
- ZigBee
- WiFi
- GSM

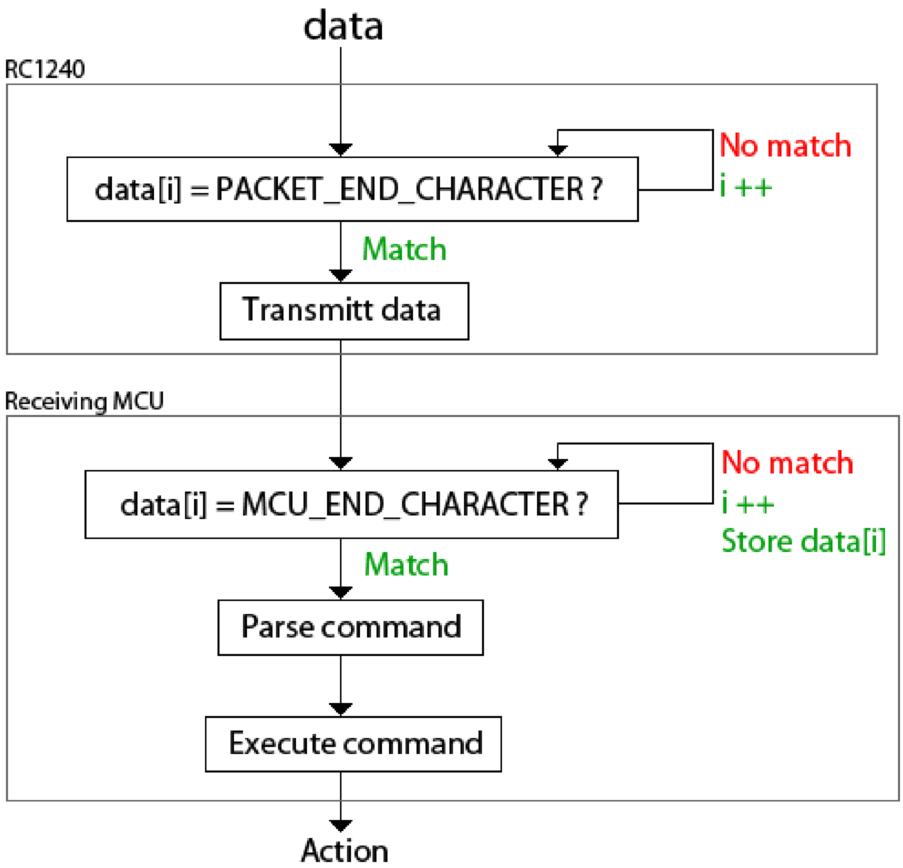


Figure 5.2: Message sending between MCU and RC1240

ZigBee

ZigBee offers a wide range of products for WSN and present a viable solution for the intended usage in this thesis. The protocol is aimed at short range wireless transmission at a relative low data rate. Addressing makes it possible to handle up to 65,535 devices in a single mesh network utilizing 2,4GHz as a standard.

Comparing the products from Radiocrafts to the ZigBee family there is a vast difference in forum related support. Radiocrafts does not have the same international attention as the ZigBee products got. Finding answers to problems one might encounter with the wireless transceivers from Radiocrafts is not an easy task. No official forum for consumers have been created, and contacting the development section at Radiocrafts directly often comes as the best solution. When that is said, they are usually very helpful in solving problems and questions one might enquire.

As the ZigBee products are intentionally meant for home appliances the transmission range is somewhat limited. The XBee platform has a maximum capability of sending at 2mW with a theoretical range of 120 meters. With a transmission power of 4mW lower than the RC1240 and 8mW below the maximum allowed limit the XBee platform renders as a less suitable solution for the intended usage in this thesis. An other alternative is the XBee-Pro with a maximum transmission power of 63mW and an optional setting at 10mW. Unfortunately the transmit current is at a much higher level than the RC1240 (87mA compared to 26mA) and is therefore also considered unsuitable in the current setup. The higher power consumption is much due to the heighten data rate, peaking at 250 Kbps. As long as there is no video module planned for the design in this thesis, the extra amount of data rate is not needed.

WiFi

WiMAX is a wireless communication standard for larger metropolitan areas and even whole countries. WiMAX operates from 10-66 GHz and was primarily defined for point-to-multipoint use. The range is typically around 2-5km and has a data throughput of 70Mbits. Based on the 802.16e specification WiMAX is specifically designed for mobility and roaming considerations, functioning very well as an alternative for the current task. The system architecture is Internet Protocol (IP) making it capable of communicating with standard off-the-shelf Wi-Fi consumer products. Some areas, such as the center of Trondheim, Norway do have a large area covered by Wi-Fi. The problem is of course the range, and the area of operation is still limited to the city center. Matrix[9] offers break-out boards ready to use at a reasonable, yet not cheap, price. The E-block wireless LAN board (figure 5.3) is a possible solution if the WiMAX/Wi-Fi solution is pursued. The component is roughly 2 x 5cm with an on-board antenna and optional 3.3/5V.

Figure 5.3: E-block wireless LAN board

GSM

As an alternative to WiMAX/Wi-Fi the cellular network has had a rapid expansion in both data rate and users, following the popularity of smartphones. HSDPA (High-Speed Downlink Packet Access) builds on the UMTS (Universal Mobile Telecommunications System) and allows high data transfer and is of today well established. The speed differs a lot, but continuously expansions increases the area of operation and capacity almost daily. Operators in Norway claim they have coverage of about 95% of all houses, and with the newly opened Tele2 net the coverage should be just about every house in the country. The range is of course limited here as well and is a result of the radiation effect mobile phones can emit. The E-block GSM board displayed in figure 5.4 is fully equipped with on-board antenna, quad band capabilities and SIM card slot. Dimensions are also 2 x 5cm.

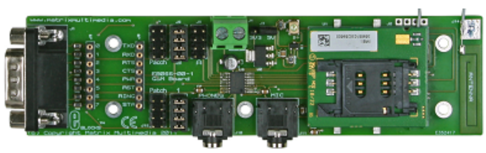


Figure 5.4: E-block GSM board

[9]http://www.matrixmultimedia.com

5.4 GPS

EM-406A Key Features

- Size: 30 x 30 X 10,5 mm
- Extremely high sensitivity: -159 dBm
- Hot start: 1s
- Warm start: 38s
- Cold start: 42s
- Current consumption: 70 mA
- Supply voltage: 4,5 - 6,5 V
- NMEA 0183 and SiRF binary protocol

The hot start refers to when the GPS unit remembers the last calculated position and the satellites in view. In addition the almanac and time is stored on-board, available and updated. The GPS device then attempts to lock onto the same satellites and calculates a new position based on the already stored information. Hot start is the fastest GPS lock to obtain TTFF, but does only work correctly if the GPS unit is in the same location as it were when turned off.

Warm start is when the GPS unit remembers its last calculated position, almanac used, and time, but not which of the satellites were in view. It then performs a reset and attempts to obtain the satellite signals and calculates a new position.

Last, the cold start is when the GPS device have discarded all the information, attempts to locate satellites and then calculates a GPS position. This takes the longest because there is no known information.

5.4.1 USGlobalSat EM-406A

The EM-406A is a 20 channel GPS receiver from USGlobalSat based on both the NMEA 0183 and SiRF binary protocol. The module is a sturdy compact construction with built-in antenna, LED status indicator and battery backed RAM for configuration. At a price of around 60$ the EM-406 is fairly cheap for most users, but has a drawback in varying accuracy (discussed in section 8.2), despite what is stated in the datasheet. The EM-406 has been used in several projects at NTNU in the past, amongst others [Eriksen(2007)]. Based on the feedback from the mentioned projects and master thesis, the EM-406 was chosen for this thesis.

5.4.2 NMEA 0183 protocol

NMEA 0183 is a standard developed, defined and controlled by the National Marine Elevtronics Association [10]. The NMEA 0183 is a combined electrical and data specification intended to support one-way serial data transmission from one unit (talker) to one or more receivers (listeners). The electrical standard which is used with the NMEA 0183 protocol is mostly the EIA-422 (RS422), although most NMEA 0183 based hardware also supports a single EIA-232 (RS232) port. Furthermore, the standard is based on the use of ASCII serial communication with the following configuration:

[10] http://www.nmea.org/content/nmea_standards/nmea_083_v_400.asp

- Baud rate: 4800 bps
- Data bits: 8
- Parity: None
- Stop bits: 1
- Handshake: None

The NMEA 0183 protocol is based on sentence passing between the listeners and the talkers. Queries can be sent to the listeners to request a particular data packet from the talkers, or the talkers can be configured to continuously send data streams. When using the NMEA 0183 protocol in relations to GPS receivers a possible format is the following (GGA - Global Positioning System Fix Data. Time, Position and fix related data for a GPS receiver):

$–GGA,hhmmss.sss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh

A random value numbered example is displayed below:

$–GGA,161229.487,3728.2475,N,12158.3416,W,1,07,1.0,9.0,M,,,,0000*18

Each value, seperated by a comma, is explained in table 5.4.

Name	Example	Units	Description
Message ID	$GPGGA		GGA protocol header
UTC Time	161229.487		hhmmss.sss
Latitude	3723.2475		ddmm.mmmm
N/S Indicator	N		N=north or S=south
Longitude	12158.3416		dddmm.mmmm
E/W Indicator	W		E=east or W=west
Position Fix Indicator	1		See table 5.5
Satelites Used	07		Range 0 to 12
HDOP	1.0		Horizontal Dilution of Precision
MSL Altitude	9.0	meters	
Units	M	meters	
Geoid Seperation		meters	
Units	M	meters	
Age of Diff. Corr.		seconds	Null fields when DGPS is not used
Diff. Ref. Station ID	0000		
Checksum	*18		
<CR><LF>			End of message termination

Table 5.4: GGA Data Format

NMEA 0183 Application Layer Protocol Rules

The NMEA 0183 protocol has the following application layer protocol rules:

- Each message starts with a dollar ($) sign.
- The next two characters identify the talker, followed by three characters identifying the type of message.

- All data fields are comma-delimited.
- If data is unavailable, the corresponding field contains a NULL (i.e. AA,,AA the double comma represent a NULL element).
- Trailing the last data field there is an asterisk (*) to represent the end of data.
- If checksum is supplied it will follow the asterisk . The checksum is the exclusive OR of all characters between the $ and *, not including the commas[11].

Value	Description
0	Fix not available or invalid
1	GPS SPS Mode, fix valid
2	Differential GPS, SPS Mode, fix valid
3-5	Not supported
6	Data Reckoning Mode, fix valid

Table 5.5: Position Fix Indicator

5.4.3 Setup and Usage

When connected to power the EM-406 will automatically send out the following messages (factory default): GLL, GSA, GSV, RMC and VTG. It is possible to use command messages to query the NMEA protocol inside the EM-406 to limit which data it should produce. It can either be polled once, or setup for periodic output at constant frequency. Checksums may also be enabled or disabled depending on the needs of the receiving program. All the NMEA message settings which are applied to the EM-406 are stored in battery-backed memory for each entry the message is accpeted. In order to change the output of the GPS receiver the query command $PSRF is used. In table 5.6 the possible options are displayed, with the possible message type in table 8.1.

Name	Example	Units	Description
Message ID	$PSRF103		PSRF103 protocol header
Msg	00		See table 8.1
Mode	01		0=SetRate, 1=Query
Rate	00	seconds	Output: off=0, max=255
CksumEnable	01		0=Disable Checksum, 1=Enable Checksum
Checksum	*25		
<CR><LF>			End of message termination

Table 5.6: Query Control Data Format

As an example the following message must be sent to the EM-406 to disable the output of GLL:

$PSRF103,1,0,0,1*25

[11] NMEA checksum calculator: http://www.hhhh.org/wiml/proj/nmeaxor.html

Value	Description
0	GGA
1	GLL
2	GSA
3	GSV
4	RMC
5	VTG

Table 5.7: Messages

5.4.4 Alternatives and Improvements

After extensive use of the EM-406 some elements has raised to the surface on how to improve on the selection of GPS. One of the problems with the EM-406 when using it with a fast moving vehicle is the update frequency limit. It is possible to adjust, but with a maximum output at 1Hz. With a vehicle of any kind moving at 10 km/h or faster this is somewhat on the boundary on what is acceptable. If the propagation time and computational overhead is included, an update of the current position every 1 seconds is just not enough. For other intended purposes it might be sufficient, but when it comes to controlling a vehicle within limited maneuvering space (e.g. a road) it is quickly to drive of the road before a new measurement is produced. As improvements to this both the SUP500F, LS23060 and the DS2523T has higher update rates (10Hz, 5Hz and 4Hz, respectively) and poses as better options on this matter. They are all approximately the same form factor as the EM-406.

As explained in section 3.2 the TTFF is dependent on how and if the almanac is stored on the GPS unit. The EM-406 uses a supercapacitor to store the almanac. This means it will acquire a rather fast TTFF for only a limited period of time (based on usage: 10-15 minutes) after the power of the module is turned of. If the EM-406 stays off for too long the almanac will have to be downloaded again, spending at least 12,5 minute before capable of producing a position fix. The SUP500F has the advantage of permanently storing the almanac in flash memory and is therefore capable of a very fast TTFF with the period of validity of the almanac. On the other hand, the SUP500F does spend significantly more time (20 minutes) on the initial almanac download, party because the need to store it on-board.

Though not a problem in the design made in this thesis, the EM-406 only operates at 5V. As it has become more common to rely on 3,3V when designing circuits, it is necessary with an extra component in order to regulate the voltage. Both the SUP500F and the LS23060 does have the ability to run on either 3,3V or 5V, and might be suited as a better alternative.

The DS2523T is the only GPS unit found so far which offers the use of an external antenna. It is not certain, but the accuracy of the EM-406 might not be the best due to the smart-antenna mounted directly on the component as a ceramic patch. In projects in need of greater accuracy the DS2523T should be considered as an option.

5.5 GSM Module

The SM5100B (depicted in figure 5.5) is a miniature quad-band GSM module which was purchased for this thesis, but never implemented. It was meant to function both as the unit to

Figure 5.5: From left: LS23060, DS2523T and SUP500F.

provide the almanac (see section 3.2) and as a potential wireless solution (see section 5.3.5). The SM5100B includes a standard SIM slot and communicates through standard UART and SPI. It is powered by 3,3V, but with an on-board voltage regulator it can also support 5V input.

The SM5100B is a very powerful tool which comes with a variety of possibilities. It offers the functions as a standard phone, with making and receiving calls, sending text messages and acquiring information about the unit position. Unfortunately it was not enough time to actually test the module, but the board made in this thesis does have the required expansion slots in order to make it work.

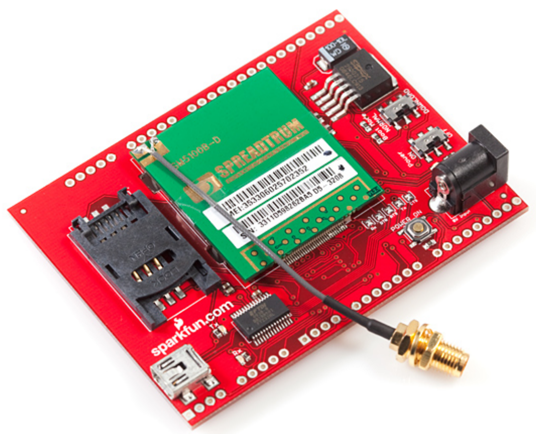


Figure 5.6: The quad-band module SM5100B with integrated SIM-slot

5.6 IMU

Even though not implemented in the current design, it is almost imperative to include an IMU to complete the maneuvering system. As the GPS does not have the same capability to measure the movement direction, an IMU could be useful when calculating the appropriate path. When choosing an inertial measurement unit (IMU) for a project there are some characteristics one have to consider:

- **Range of measurement:** Determine the desired range of the measurement, and whether or not it is necessary to measure both negative and positive acceleration. Choosing a full-scaled IMU which can measure up to ±200g, when using the data to control a RC car, would be foolish. The range of the IMU would be far from fully exploited and the resolution of the measurements would most likely be too coarse to get any use of it. In most smaller-ranged accelerometers the precision increases if the range is decreased, thus giving more accurate data for the intended use.
- **Interface:** Choosing the IMU must often be done in accordance with the choice of MCU. There are mainly 3 different types of interfaces with the IMUs and choosing the correct type depends on the types of I/O functions the MCU inhabits:
  - **Analog:** These produces a voltage value that is directly proportional to the sensed acceleration. Most of today's microcontrollers have multiple analog-to-digital converters (ADCs).
  - **Digital:** Often digital IMUs include more features than analog, but could be harder to implement, depending on the level of experience of the designer. The transmission of values are done either via SPI or I2C, where the data format is dependent on each model and explained in the associated datasheet.
  - **PWM:** Accelerometers using PWM signals to indicate acceleration poduces a square signal with varying duty cycle in correspondence to the sensed acceleration.
- **Number of axis:** Depending on the type of vehicle being measured the accelerometer can have up to 3 axis, measuring x,y,z.
- **Bandwidth:** The bandwidth determines how often the module is capable of outputting data to the receiver. This depends heavily on the specific usage of the project and what functions it is desirable to achieve.
- **Power consumption:** An important factor which much be taken into account if the module is driven on battery. Most units has the ability to power down when not in use, limiting the power consumption to a minimum.

5.6.1 Alternatives

Even though the IMU is a crucial component, there has not been done testing on the matter. The fact that the IMU first is important when looking at waypoints and path planning it is left for future work. One model were examined to pose as possible options if further work was done with the board. The 3-axis IMU ADXL345 (figure 5.6.1) is small in size (1 cm x 2 cm) and with a extremely low power consumption at 40 - 145 μA (depending on the functions applied) is could easily be implemented using the current board design without worrying about power shortage. Both SPI and I2C is supported and a maximum bandwidth of 3200Hz is sufficient enough for the CyberBike. The ADXL345 also has the ability to set different scale ranges: ± 2, 4, 8 and 16g is possible.

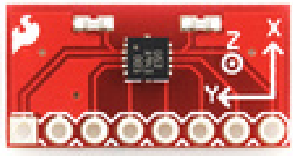


Figure 5.7: The ADXL345

5.7 Collision Avoidance

Mainly two models were planned for the collision avoidance system. The first and most general purpose is based on observing the distance to the obstacles occurring in the planned path, and then taking measure to avoid it. This would be achieved by placing 4 range sensors at the front of the vehicle of operation as seen in figure 5.8. The lower sensors would then be placed as close to the ground as possible, detecting small objects and obstacles like curbs and animals. The other set of sensors would be place at a height suitable to detect people and cars. Two different range sensors were tested for this system and is explained in section 5.8.

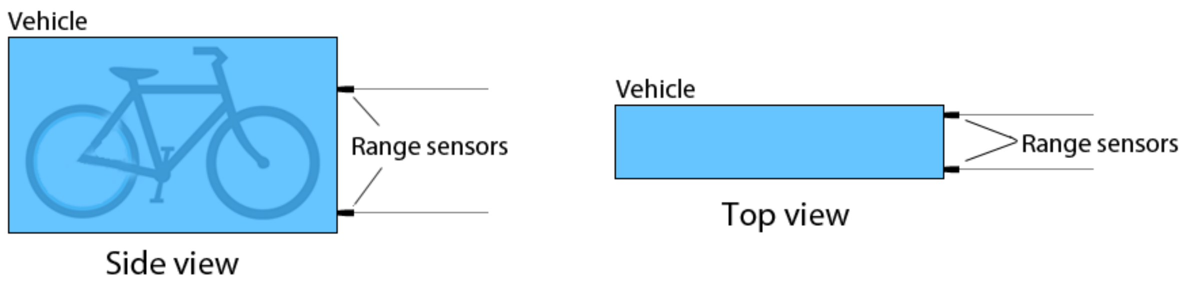


Figure 5.8: Possible solution for collision avoidance.

A problem which quickly surfaces with the type of setup displayed in 5.8 is when the vehicle is turning. As the range sensors proposed for the setup has a limited area of detection, mostly straight forward, it is difficult to see obstacles in areas to the side of the vehicle (shown in figure 5.9).

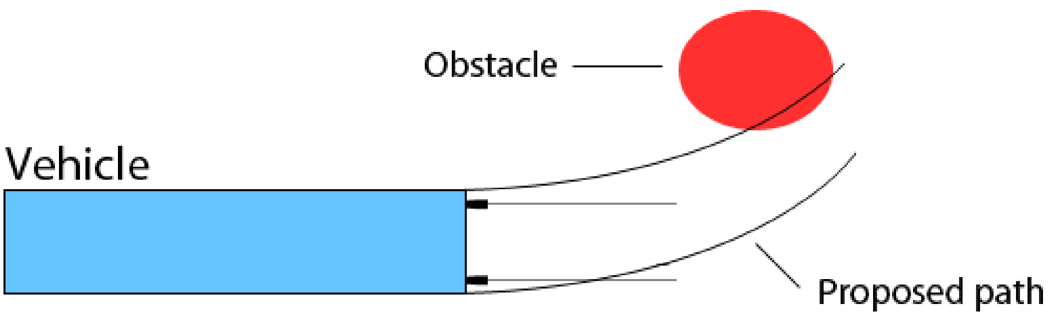


Figure 5.9: Obstacles might go unnoticed if the turning is too sharp, making it possible to crash.

In order to avoid the problem when turning there are two possible solutions. One is to attach the range sensor on a servo (figure 5.10), making it turn as the bike turns. A similar technique is being used on the headlights of modern cars. As the driver turns the steering wheel the lights illuminates the road in accordance to the angle of rotation of the steering wheel. Utilizing the same principle, replacing the head lights with the range sensor would increase the angle of observation and eliminate the problem depicted in figure 5.9.

Figure 5.10: Range sensor mounted on a servo.

The second solution to the problem, and the second main system, is using a laser with an area of detection much wider than the simple range sensor. A proposed laser is presented in section 5.8. The laser is larger, more complex and much more expensive, but offers a area of detection of totally 240 degrees. As the laser still only detects obstacles in a single plane it is necessary with two units, pushing the total cost of the system to a new level. The best overall solution might be to combined the two systems, using the laser to detect people and cars at the highest elevation and simple range sensor at ground level.

5.8 Range Sensors

5.8.1 Sharp GP2Y0A700K0F

The GP2Y0A700K0F from Sharp was tested and propose as the best solution for simple infrared (IR) range sensors. According to the datasheet the range is 1 - 5,5 meters, therefore lacking the ability to measure objects too close to the sensor. This is due to how the sensor measure the distance. The infrared light is emitted from a source on one side of the sensor. Continuously, the sensor measure the angle of the reflecting light on the opposite side. If the object measured is too close, the reflecting light will not be received at the sensors input. The result is that it seems there are no objects in front of the sensor (displayed in figure 5.12).

At a price of 30 USD it is affordable, but to make a complete working system 4 is needed. This drives the total cost up some. The power consumption is stated to be 30 mA, also adding a significant power need to the system as a whole.

Testing the sensor was done by connecting it to the AtXMEGA256A3 and selecting one of the pins with ADC (Analog to digital converter). The sensor will then report the distance measured as an analog voltage, ranging from 2,2 - 5 V. Problems occurred when both the measured object and sensor were static. With this scenario, the sensor does not seem to be able to produce useful readings, but as soon as either of them moves the readings are again produced. The accuracy was surprisingly accurate, involving only a minor disturbance at the maximum distance.

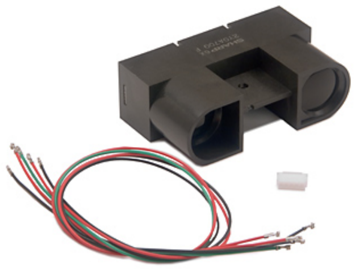


Figure 5.11: Sharp GP2Y0A700K0F infrared range sensor.

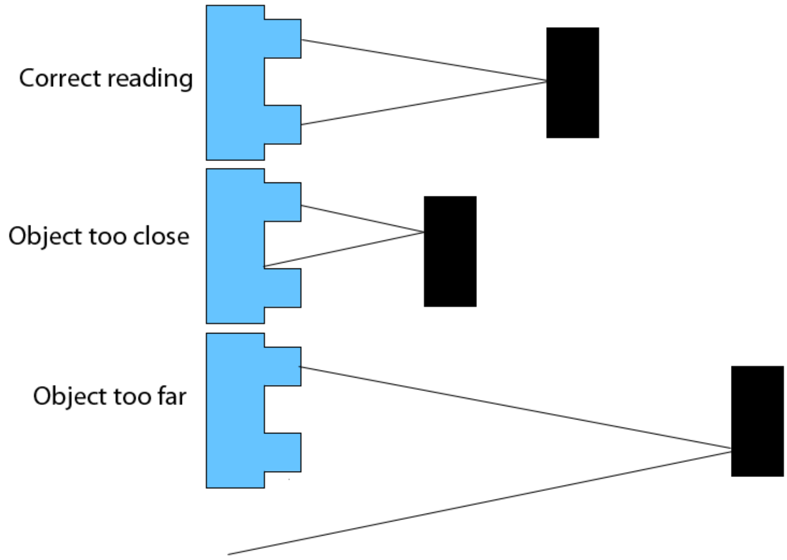


Figure 5.12: Correct reading at the top. In the middle the object is too close, making the angle of incoming light too small and thus not hitting the sensor. At the bottom the object is too far away and produces too high angle, resulting in not hitting the sensor.

5.8.2 SRF08

The second option for range measurement is based on sound instead of light. The SRF08 is a ultrasonic range sensor operating at 5V and outputs a frequency of 40KHz, using 20 mA of current. Capable of measuring from 3 cm to 6 meters the SRF08 is a very reasonable choice. With the use of sound the sensor does not suffer with bad readings in close scenarios, but has rather a wider span of area of detection as shown in figure 5.14. The downside is the price at 60 USD per unit. With twice the price for each unit the cost of 4 of these will drastically increase the total cost of the system.

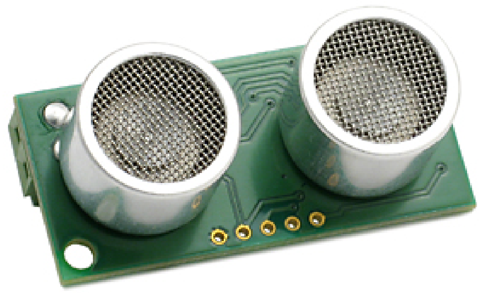


Figure 5.13: SRF08 Ultrasonic range sensor.

5.8.3 Hokuyo Laser

As the last example, the Hokuyo Laser is presented. In comparison with the Sharp infrared and SRF08, it is considerable larger, more power hungry and last, but not least, much more expensive. At a price of 3,300 USD it totally changes the budget of the system. Even so, the area of detection is a staggering 240 degrees with an angular resolution of .36 degrees. Using the Hokuyo Laser would discard any need for servos to obtain wider field-of-view, and only a single unit would cover the entire section of interest. Current consumption is 500 mA at 5V which is the equivalent to 25 SRF08s or 16 Sharp infrared sensors. The update frequency is set to 10 Hz and the range is stated to be from 20 mm to 4 meters. With the laser a diagnostic software is included, as well as how to manipulate the data delivered by either RS-232 or USB interface. The size is 5 x 5 x 9 cm which is not too bad considering the functionality.

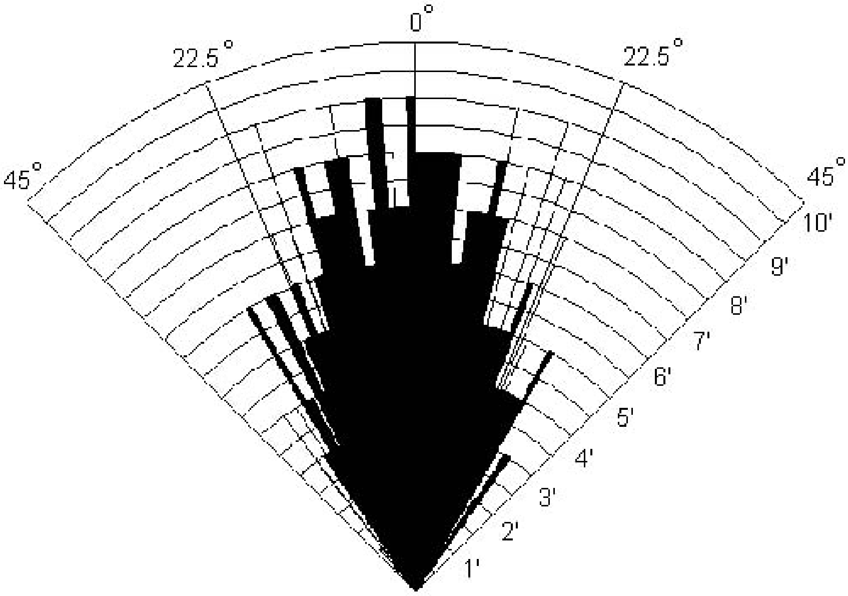


Figure 5.14: SRF08 detection area

Figure 5.15:

Chapter 6

Electrical Design and Production

This chapter works a guide to those who are interested in either continue working with the boards made during this thesis, or as helping tool for improving or making a board of their own. The current board design is explained in detail in section 6.1, and a simplified guide on making PCB cards is presented in the following section. All that is needed to produce PCB boards similar to the ones used in this thesis can be found in [Koteich(2011)], ranging from the very beginning with schematic and board layout, choosing components and producing the board using ProtoMat S62.

6.1 Schematics, Board Layout and component placement

The design was made using EAGLE (Easily Applicable Graphical Layout Editor) which is a freeware product made by CadSoft and is possible to download their homepage[1] in a Light Edition. Even though EAGLE present a rather simple user interface with some strange default hotkeys it includes most of the needed functions to create a 2 layer circuit board from scratch. EAGLE also includes the possibility to create customized components based on the soldering footprints found in the datasheet.

The board schematics can be found both in Appendix C and the board layout can be found both in Appendix D.

The actual placement of the components are displayed in figure 6.1 with description in table 6.1.

6.1.1 How To Think When Designing Prototypes

When creating a circuit board from scratch there is a lot of planning involved as the process is tedious and should not be undertaken without the proper information and correct planning. To begin with, define the purpose of the prototype. What should it be able to do, and what are realistic goals for you. It is easy to get carried away and follow your imagination, but ideas are often easier to perform in your head than in the real world - so keep it simple to begin with. If

[1] http://www.cadsoftusa.com/

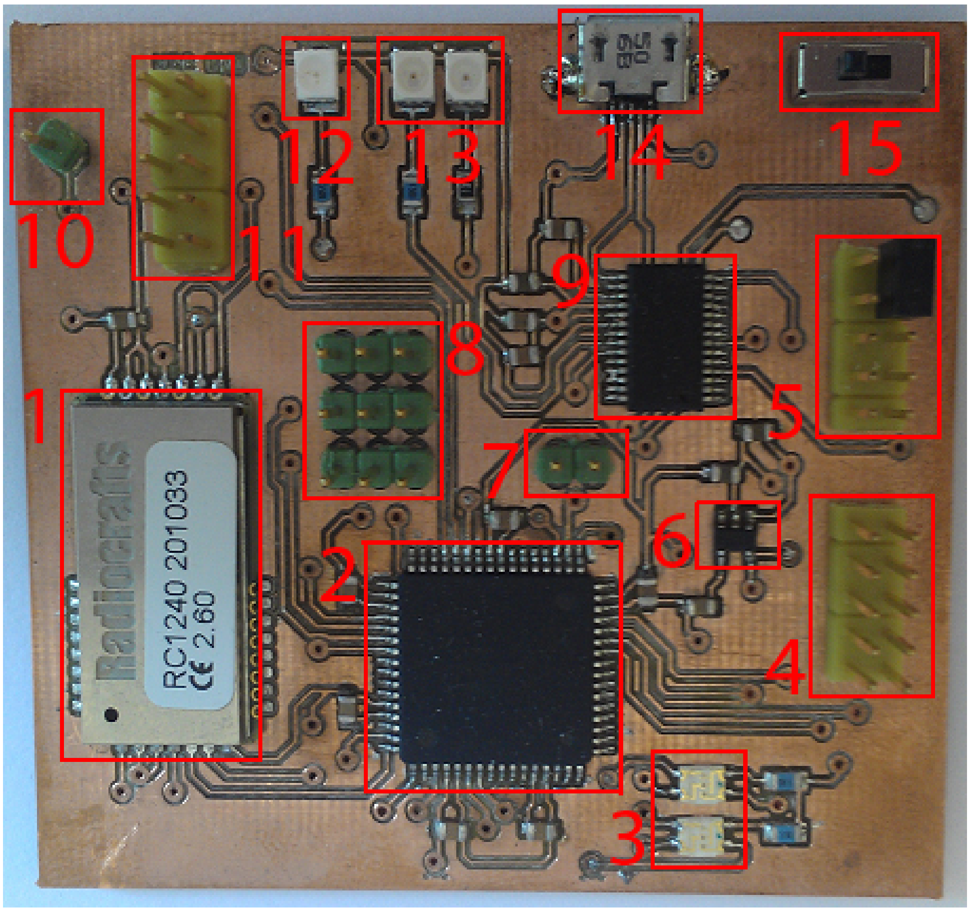


Figure 6.1: Component Placement on the board

you feel you lack competence in certain areas, make sure you read up on the topic. Prototyping does not cope well with doing first and learning after.

6.1.2 Where To Produce Prototypes

The complexity of the system defines which main path to follow when choosing where to produce the finished design. A simple 2-layer board with a limited number of components and wires automatically implies that an external PCB fabrication house (e.g. PCB-Cart[2]) would be both overly expensive as well as a waste of time considering the outcome. One should of course take into consideration what the finished product is going to be used for, as an external manufacturer would in most cases end up with a better looking result than if you do it yourself (figure 6.1.2). In addition, there are often limitations to what one can achieve with commercial equipment.

After producing numerous circuit boards I have come to the conclusion that if the design includes the same number (or more) of components and functions used on this thesis, it should be considered to use an external manufacturer. Even though making the board yourself will give some better knowledge and understanding of the components, it is too much room for error and the process takes too much time. Instead, by ordering the production elsewhere it could save a lot of time and frustration trying to find flaws comming from the production phase.

[2] http://www.pcbcart.com

Number	Component	Description/Function
1	RC1240	Wireless tranceiver
2	AtXMEGA256A3	MCU
3	Triple-color LEDs	Indicates data over RC1240
4	10-pin header	JTAG
5	10-pin header	3,3V selection jumper
6	MC7805ACTG	Voltage regulator, 5V to 3,3V
7	2-pin header	UART test pins
8	3-pin header	Motor/Servo pins
9	FT232RL	UART to USB
10	1-pin header	Antenna
11	10-pin header	Perhiperals (GPS,IMU,Range sensors)
12	AP131-33WG-7 (Green LED)	MCU Power
13	DJS-BRGBC-TR8 (Blue LED)	Indicates USB data
14	PRT-08533	Micro USB connector (Interface PC)
15	COM-09609	ON/OFF switch

Table 6.1: Component Description

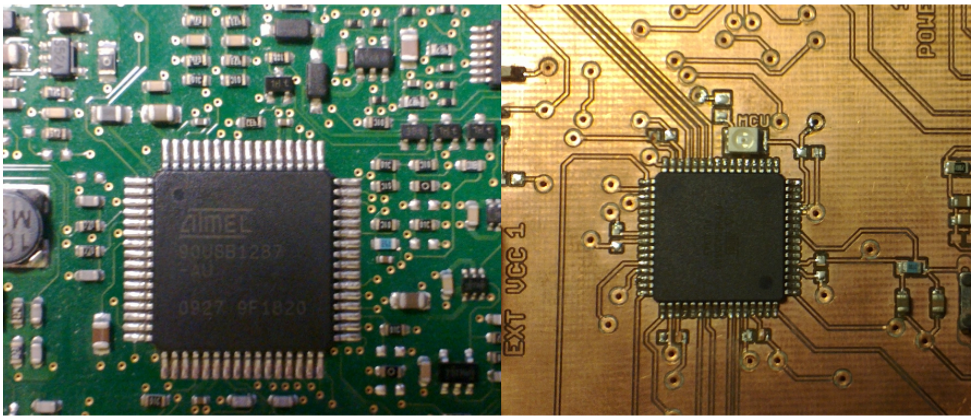


Figure 6.2: External manufacturer on the left, in-house on the right

6.1.3 Design for testing

As mentioned, it is easy to get carried away when you create your first design. Unfortunately one has to do all the work once first to understand how much work it actually is. The process from having an idea to the place where the board is actually working is an extensive workload. Make sure all the correct information has been gathered before starting to minimize the probabilty of making mistakes. When that is said - people still mess up. A list of things to consider during the design phase have been made to help with the prototyping:

- Place the components relatively far apart to ease the soldering and further testing.
- If possible, divide the circuit using connectors and jumpers. This makes it a lot easier to find possible shorts and errors later.
- Do not add more functionality than necessary, this will only lead to more places errors can occur. Stick to what you need.
- Use LEDs where they can be used. Even though it might be just to indicate the power of

a small component, it saves you for a lot of extra time not having to use the multimeter.

- Read the datasheet for each component - and then read it again. You would be amazed on how often significant information is put in a footnote.
- Examine the components you are using before placing them in the board layout. Some components can only be soldered from the bottom side.

6.1.4 Gerber Files

There are lots of different layout packages used as of today (Protel, Orcad, Eagle, PCB, etc). Luckily there is one common standard called Gerber files. In general these files gives the designated machine the correct coordinates to where wires should go, where to drill holes, and what dimensions to use. The most standard Gerber setup named gerb274x.cam will result in the following 7 files:

- Top Copper - Position for top wires (GTL)
- Top Soldermask - Masking on top where solder will not stick(GTS)
- Top Silkscreen - Writings for instructional use on top (GTO)
- Bottom Copper - Position for bottom wires (GBL)
- Bottom Soldermask - Masking on bottom where solder will not stick (GBS)
- Bottom Silkscreen - Writings for instructional use on bottom (GBO)
- Drill File - Position and dimension for holes

In addition to the mentioned files above it is recommended to include a board layout file named Excellon.

Chapter 7

Graphical User Interface

7.1 Overview

In order to present the data received from the board a program has been made in Microsoft Visual Studio (MVS) 2010 using C#. The program creates an interface for the user of the system to read important data and, if available, control vehicles using the keyboard. Figure 7.1 shows the control panel of the program with each section described in table 7.1. The program is included with the rest of the files on the DVD. It is easy to set up and requires only a working edition of MVS.

Section	Description	Comment
1	Port selection and connection	All available ports will be shown
2	Main administration	Close program: Terminates program and close active port Clear chart: Clears the map Set MASTER: Set connected board to master Set SLAVE: Set connected board to slave
3	User output	Manually send commands
4	Master and Slave GPS	Connecting: Red: No fix, Green: Fix
5	RC1240 panel	Test config mode: Engages config mode Send 'TEST' RC: Sends the string TEST Set end character to $: Applied if reset has been activated
6	Car panel	Throttle: Displays the throttle in numeric value Sync car: Synchronizes the throttle and turning
7	GPS Config	Update freq: Changes how often the GPS outputs data Output only GGA: Removes all other output than GGA
8	GPS offset	
9	Program output	Shows what the program transmits to the board
10	Input	Displays the input of the program
11	Status	Current program status

Table 7.1: GUI interface

7.2 Usage

7.2.1 Connection

As the program does not actively search for new connections to the computer after initialized, it is important to connect and power on the boards before starting the program. Trailing this, the appropriate port can be selected using section 1 depicted in figure 7.1 and then click the button named 'Connect'. This will engage the connection and data is now ready to be sent and received. Initially the boards are configured as slaves and in order to choose the connected board as a master the button with 'Set MASTER' in section 2 must be clicked. Immediately after the connection is accepted the GPS will transmit data to the program. As long as the GPS does not have a fix on satelites the 'connection' field in section 4 will remain red and unusable data will be displayed in the input field (section 10).

7.2.2 Normal Mode

When the GPS receiver have connected to the satellites and obtained a fix the 'Connecting' field in section 4 will turn green and display 'Connected'. Data extracted from the GGA data stream will now be displayed in the same section as seen in figure 7.2. The NMEA 0183 protocol produces coordinates with format *DDmmss*. Because it is also common with the use of decimal degrees the last two fields in section 4 displays the coordinates in decimal, using equation 7.1.

$$Decimal\ Degrees = Degrees + minutes/60 + seconds/3600 \tag{7.1}$$

When the program is running and the board is attached and connected to the car it is possible to use the keyboard to ouput commands to the car. The letters 'W' and 'S' controls the throttle and the letters 'A' and 'D' controls the steering.

7.2.3 Data reception using UART

As mentioned before, there is an important concept one must be aware of before handling data transmitted over UART. Because most modern computers far exceeds the UART speed in regards of handling the incoming data, often the buffer is read before the complete intended command arrives in the computer buffer. As shown in figure 7.3 the buffer read event is triggered at random times when the computer buffer receives data. If no end character is used and the command can not be sufficiently executed before the complete command, it will only end up as errors in the program.

To prevent this to happen it is possible to implement the use of end characters. By adding a special character at the end of the transmitted command it is possible to know when the complete command have been read from the buffer. As shown in figure 7.4 the buffer read event is triggered at the same times as before, only now the program does not execute the command before the end character is received and read.

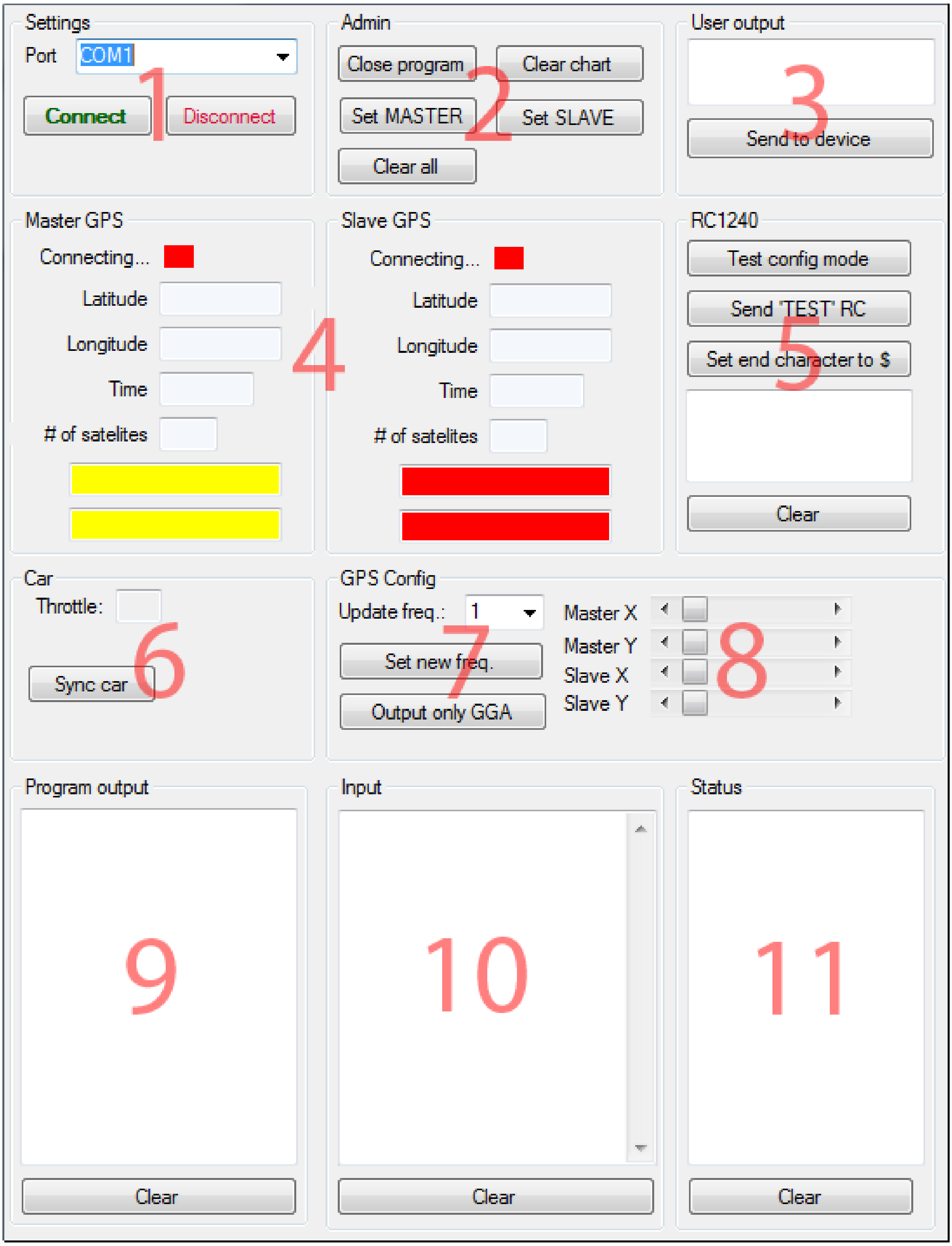


Figure 7.1: User GUI

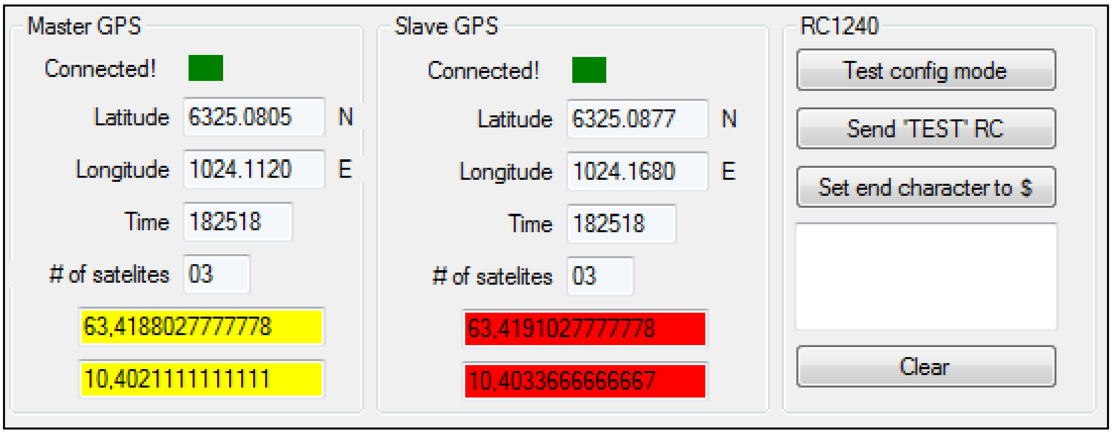


Figure 7.2: User GUI

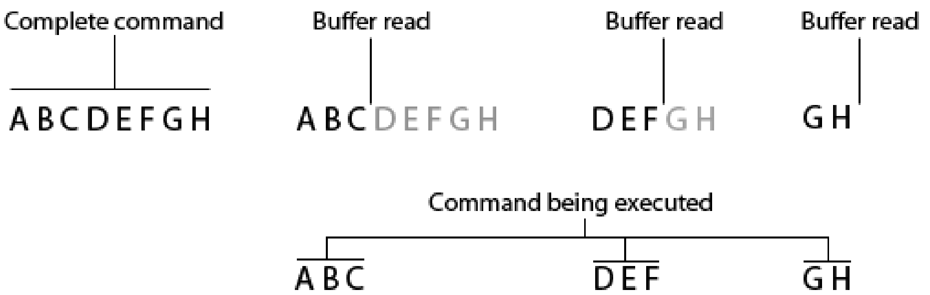


Figure 7.3: The computer buffer reads the incomming data too early, resulting in an incomplete command.

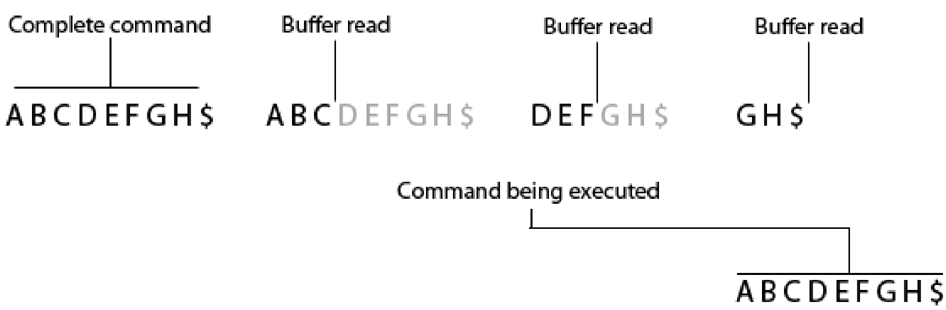


Figure 7.4: By using end character the command is not executed until the special character is read.

Chapter 8

Testing

Different test scenarios were performed to get an understanding on whether or not the design and components would be sufficient to accomplish the task. As the eCrasher (4.2.3) was damaged before a fully working navigation system was implemented, most of the tests were done by foot while carrying the car. Even so, the test results are representabel for the outcome of the system, as the focus was not to test the car, but rather the wireless solution and positioning.

8.1 RC1240

The RC1240 wireless transceiver was tested using a quarter length wave antenna as recommended in the datasheet. On low data rates (100 bytes / second) and short distances the modules works flawlessly, resulting in only minor loss of data. Surprisingly, the errors did not come with longer distances, but when the data rate was converging the maximum baud rate. At this point the loss was so great, resulting in multiple broken commands leading to a crash of the program.

Data rate (bytes / second)	Distance (meters)	Packets sent	Loss (percentage)
100	50	1000	0,12%
	500		0,27%
	1000		1,2%
300	50	1000	0,7%
	500		1,3%
	1000		3,3%
600	50	1000	4,7%
	500		6,7%
	1000		7,1%

Table 8.1: Messages

8.2 GPS Accuracy

The GPS module EM-406 defintely gave some interesting test results worth noticing. First of all, the GPS module does not cope well with staying at the same place. That is, when the module is not moving the position produced seems to be floating. Even though it is not significantly the deviations vary from around 0,5 meters to about 2 meters. As soon as the EM-406 is moving the accuracy improves and a stable, more likely correct position is produced. This is shown in figure 8.2 where the yellow colored plotting represent the master module at a static position.

Figure 8.1: GPS device floating in static position.

The fact that the static and moving GPS units behaves differently creates a problem when trying to utilize the concept of DGPS. In order to use the calculated error from the static GPS, the roving unit should produce the same. When the behavior changes as much as it does when going from static to moving, the use of DGPS does no longer apply.

For this thesis it was purchased only one additional unit of the EM-406. The other was already in place at NTNU and therefore the need to buy another seemed unnecessary. As figure 8.2 shows the two GPS modules, even though exactly the same, behaves differently. The same setup is used for both units, but still one of them seems to float more. Tests were done where the GPS units worked both with the master and slave board, without any improvements. One reasonable explanation is that the GPS receivers are relatively cheap and therefore resulting in less accurate positioning due to production inaccuracy. Another reason could be that either of the two GPS modules could have been damaged during transportation or testing, giving slight deviations in the readings.

Also, as mentioned in section 5.4.4 the EM-406 does not have the ability to store the almanac for long and therefore needs to download it when powered off over longer periods of time. The EM-406 sometimes spend a lot of time (15 - 20 minutes) trying to get the first fix position after being powered down. Being a relatively cheap receiver it does not come as a big surprise, but it is still worth mentioning for future work.

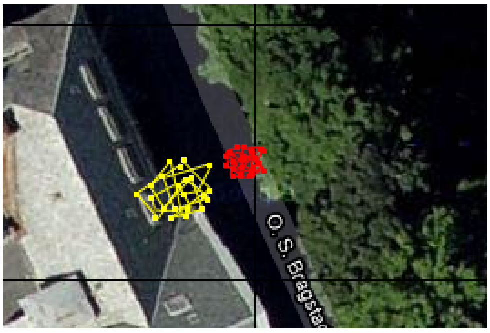


Figure 8.2: The two GPS devices producing different position deviations.

Chapter 9

Discussion

Workload, method and structuring

There is always a big risk regarding the workload when producing prototypes. Everything that can go wrong during the process, usually do. This is especially true when starting with a new design from scratch, like I did in this thesis. Even so, I am convinced that in order to really learn how the different components work and how the system will function when finished, the complete process has to be undertaken. The same basic principle also applies to the implementation and testing. Planning how modules are intended to work will only get you so far, as the true pitfalls and flaws are discovered when the actual system is constructed and powered on. The GPS EM-406 is a great example of this. Multiple previous thesis have written about it, acknowledging its prominent functions and statistics based on the datasheet. But when it was actually tested during this thesis several drawbacks were discovered.

When I created my first circuit board I was filled with joy. Something of my own design was actually produced and worked, all made by my own hands. It was truly worth all the time and effort put into making the card. In my opinion, any engineer in the same field of education should undertake the very same process. But, when the first and second card had been made, I got to understand why there are numerous companies offering the production of circuits boards. It is a tedious and time consuming process doing it all by yourself. Therefore, after creating a couple of boards, learning how the job is done, future projects should be produced by external manufacturers. This way, crucial time could be spent on other important matters, as well as the chances of production errors would be minimized, saving time on testing.

As stated in the introductory part of this thesis, the solution created is intended to be part of the much larger scaled project CyberBike. Even though the product of this thesis is part of a cooperation between several other students at NTNU, the decision to make a stand-alone module turned out to be wise. Being dependent on other people to complete their work before you can continue yours is both annoying and inefficient. Especially when the amount of time at disposal is limited, as it usually is in most cases, being able to freely structure the progress of the work is an advantage.

Positioning

Looking back at when the GPS device first was chosen it is difficult to see if it could have been done different. On the paper the EM-406 seems like a viable solution and does not impose shortcomings in the matter of DGPS. Unfortunately, it proved to be too unstable and lacked the accuracy needed to create a system based on DGPS. The problem when the GPS device was not moving, resulting in inaccurate readings, was not stated in the datasheet and therefore difficult to anticipate.

A matter which should have been considered more thoroughly, is the update frequency of the EM-406. Using the GPS on a moving vehicle, continuously producing position data, whilst avoiding obstacles, should have a higher update rate than 1 Hz. As the CyberBike got limited maneuverability, in addition to the relative high speed, the chances of driving off the road or hitting objects is definitely present. The 3 alternatives in section 5.4.4 all got higher update frequency than the EM-406 and should be considered as better options if similar tasks are chosen.

As a last observation to the choice of GPS devices, it should be noted that the EM-406 uses a supercapacitor to store the almanac. Given the long download time of a new almanac when the device has been turned off too long, the EM-406 is not suited for rapid initiation. If the setup is to be used as a demonstration, a preparation period of 20 minutes will probably not impress the audience. Choosing a GPS unit with flash memory or similar to store the almanac would then be preferred.

Wireless transmission

When choosing the solution on how to handle the wireless transmission, WiFi, ZigBee and cellular communication were quickly discarded. Instead the choice fell on the RC1240 from Radiocrafts. This was mainly done because it was a known component, but also because the implemented RC232 protocol which comes standard with the RC1240. As it is a simple, yet sufficient protocol capable of handling the desired functionalities of the design, the RC232 makes working with the component an easier task.

Even though there is not yet a substantial base of customers of Radiocrafts products, lacking the support in forum related help, Radiocrafts offers well documented application notes and datasheets for their products. In addition, they are more than willing to answer questions and inquiries if contacted.

The range of the RC1240 has proven to be reliable in accordance with the documented specifications. As it were to be used within line-of-sight, but still relatively far away, the RC1240 offers high enough data rate at the desired distance, which made it to be the best choice.

Collision avoidance

The collision avoidance system was not implemented during this thesis due to limited time. Different viable solutions were instead discussed and various types of sensors were proposed. In retrospective, the collision system should be handled as a separate task. The magnitude of the system in order to make it work well is too great to be combined with both a positioning and wireless design. Even so, it is of great importance as a safety for the system in case of malfunctioning. Solely based on the positioning system, the bike would be able to navigate through a given path. But if deviations to the planned path happened as a result of failure, the range sensors would be greatly appreciated to avoid possible collisions.

Chapter 10

Conclusion

The production of two prototype circuit boards have been made in order to test the proposed system. The process trying to make them work flawlessly have been tedious and hard. The result of using the ProtoMat S62 at the ITK workshop has lead to a large amount of production errors, some almost impossible to discover. Therefore a great amount of time have been spent on trying to find the errors and repair them. The conclusion concerning prototype production is that if the complexity is in the range of the design used in this thesis, i.e. 4-5 integrated circuits or more, an external manufacturer should be considered. The same goes for the software selection. EAGLE which was used during the process does not cope well with larger designs. A more suitable design software is Altium[1], even though more complicated, it offers a range of functions which comes in handy when designing more advanced circuits.

The process seen as a whole on the other hand, is still recommendable. It does take a lot more time to create prototypes from scratch, but it is definitely the best way to learn. Getting the hands-on experience with the components gives a lot of useful insight to bring into further work, and is not obtained if depending on finished products.

A program to present the data to a user of the system was also made. Unfortunately, the time spent on this program did not pay off. It was a lot of work put into understanding how the data reach the computer buffer, and how it was handled from here. The event system on a computer is quite different from a microcontroller and therefore presenting a whole range of new challenges for me. Also, as the time was limited, the full scaled testing of the program was not conducted, as it included a lot more applications than the ones used. Even with the disadvantages of spending time on the program, it is almost a necessity. Using a simple terminal window to handle the data of such a complicated system would not work well, and a program to present the data must be made.

The choice of GPS was intentionally good, but discoveries made during testing resulted in discarding the use of DGPS. Different behavior on the two devices gave inaccurate position readings, making it impossible to use the errors obtained at one unit to correct the other. Also, it would be advisable to choose a GPS with higher update rate. This is at least the case if the GPS is to be used with vehicles moving at a speed exceeding 10 km/h.

The wireless transceiver RC1240 from Radiocrafts worked very well during the whole project. It is a versatile, easy to operate multi-functional unit which I recommend strongly for further

[1] www.altium.com

projects. Well documented and easy to configure, which resulted in a wireless system with the possibility to expand if desirable.

In retrospective, I see that my ambitions for the system were too great. Too many subjects were included, resulting in a deficit of time to really implement them all. As a recommendation to future students I would advise to limit the functionality of the design, focusing only on single parts of the system used in this thesis, for example only the use of GPS.

Chapter 11

Further Work

The following list presents recommendations for further work:

- Additional work should be done finding a GPS device more suitable for use with the concept of DGPS. The enhanced accuracy is definitely needed for controlling the bike and DGPS is a good solution to achieve this.
- Design and implement a complete collision avoidance system. As a safety feature to the bike this would defiantly improve the system as a whole.
- Creating a link between the proposed communication system made through this thesis and the bike main system. In order to make the bike work as a complete solution, the task of merging all the peripheral systems together with the main must be done.
- Based on the design presented in this report, implement the use of an IMU unit. The need to get additional measurements is imperative when planning waypoints and paths.

Appendix A

Configuring RC1240 - Changing PACKET_END_CHARACTER (C)

```
// Set end-character
  activate_config_mode();
  respons = UsartReadChar_rc1240();
  if (respons == '>')
{
deactivate_config_mode();
UsartWriteString_usb("&SYSb_on$");
if (Params[0] == 'M')
{
UsartWriteChar_rc1240('M');
_delay_ms(10000);
respons = UsartReadChar_rc1240();
if (respons == '>')
{
UsartWriteString_usb("&SYSm_mode$");
UsartWriteChar_rc1240(17);
_delay_ms(10000);
UsartWriteInt_rc1240(36);
_delay_ms(10000);
UsartWriteChar_rc1240(255);
respons = UsartReadChar_rc1240();
if (respons == '>')
{
UsartWriteString_usb("&SYSm_writ$");
_delay_ms(5000);
}
}
}
}
UsartWriteChar_rc1240('X');
UsartWriteString_usb("&RCUb_off$");
  break;
```

Appendix B

Configuring AtXMEGA256A3 - Setting Clock Rate (assembly)

```
#include <avr/interrupt.h>

#define AVR_ENTER_CRITICAL_REGION( ) uint8_t volatile saved_sreg = SREG; \
                                     cli();

#define AVR_LEAVE_CRITICAL_REGION( ) SREG = saved_sreg;

static void CCPWrite( volatile uint8_t * address, uint8_t value )
{
#ifdef __ICCAVR__

// Store global interrupt setting in scratch register and disable interrupts.
        asm("in  R1, 0x3F \n"
    "cli"
    );

// Move destination address pointer to Z pointer registers.
asm("movw r30, r16");
#ifdef RAMPZ
asm("ldi  R16, 0 \n"
            "out  0x3B, R16"
    );

#endif
asm("ldi  r16,  0xD8 \n"
    "out  0x34, r16  \n"
#if (__MEMORY_MODEL__ == 1)
    "st     Z,  r17  \n");
#elif (__MEMORY_MODEL__ == 2)
    "st     Z,  r18  \n");
#else /* (__MEMORY_MODEL__ == 3) || (__MEMORY_MODEL__ == 5) */
    "st     Z,  r19  \n");
```

```
#endif /* __MEMORY_MODEL__ */

// Restore global interrupt setting from scratch register.
        asm("out  0x3F, R1");

#elif defined __GNUC__
AVR_ENTER_CRITICAL_REGION( );
volatile uint8_t * tmpAddr = address;
#ifdef RAMPZ
RAMPZ = 0;
#endif
asm volatile(
"movw r30,  %0"       "\n\t"
"ldi  r16,  %2"       "\n\t"
"out   %3, r16"       "\n\t"
"st     Z,  %1"       "\n\t"
:
: "r" (tmpAddr), "r" (value), "M" (CCP_IOREG_gc), "i" (&CCP)
: "r16", "r30", "r31"
);

AVR_LEAVE_CRITICAL_REGION( );
#endif
}



void cpu_set_freq() {

uint8_t clkCtrl;

// Start up 32MHz internal oscillator
OSC.CTRL = OSC_RC32MEN_bm;

// Wait for 32MHz internal oscillator to start
while(!(OSC.STATUS & OSC_RC32MRDY_bm));
clkCtrl = (CLK.CTRL & ~CLK_SCLKSEL_gm) | CLK_SCLKSEL_RC32M_gc;
CCPWrite(&CLK.CTRL, clkCtrl);

OSC.CTRL &= ~(OSC_RC2MEN_bm);
}
```

Appendix C

Board Schematic

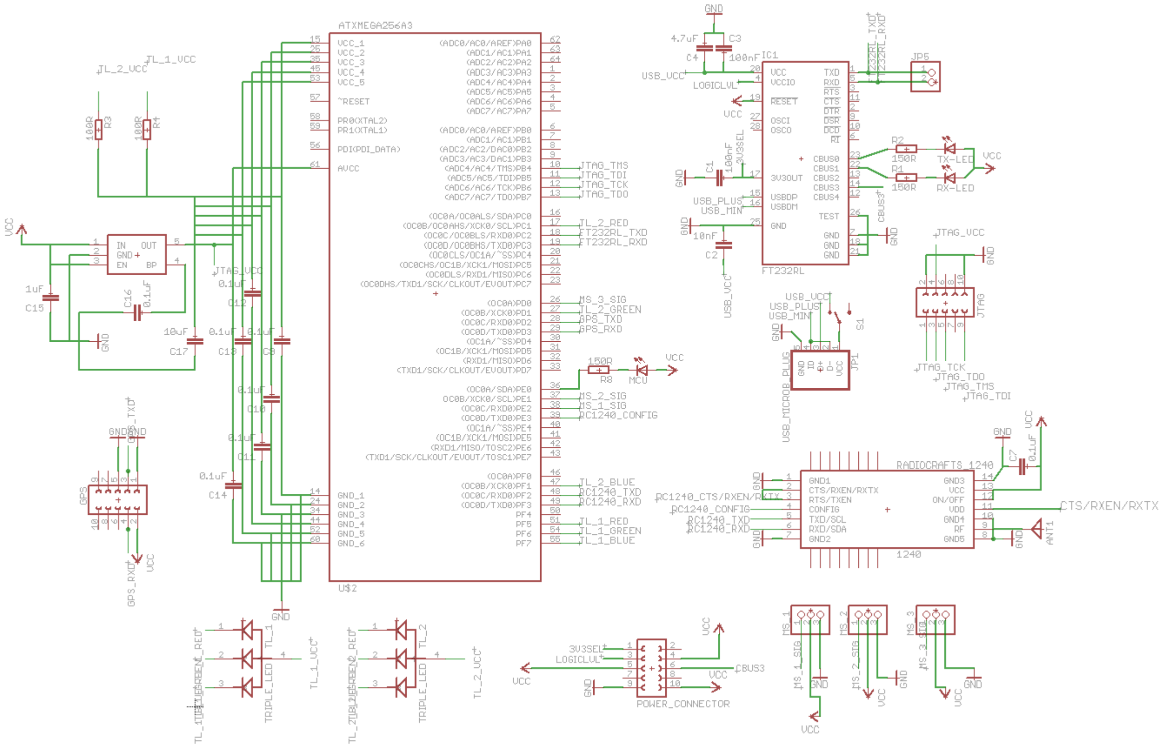


Figure C.1: Board Schematic

Appendix D

Board Layout

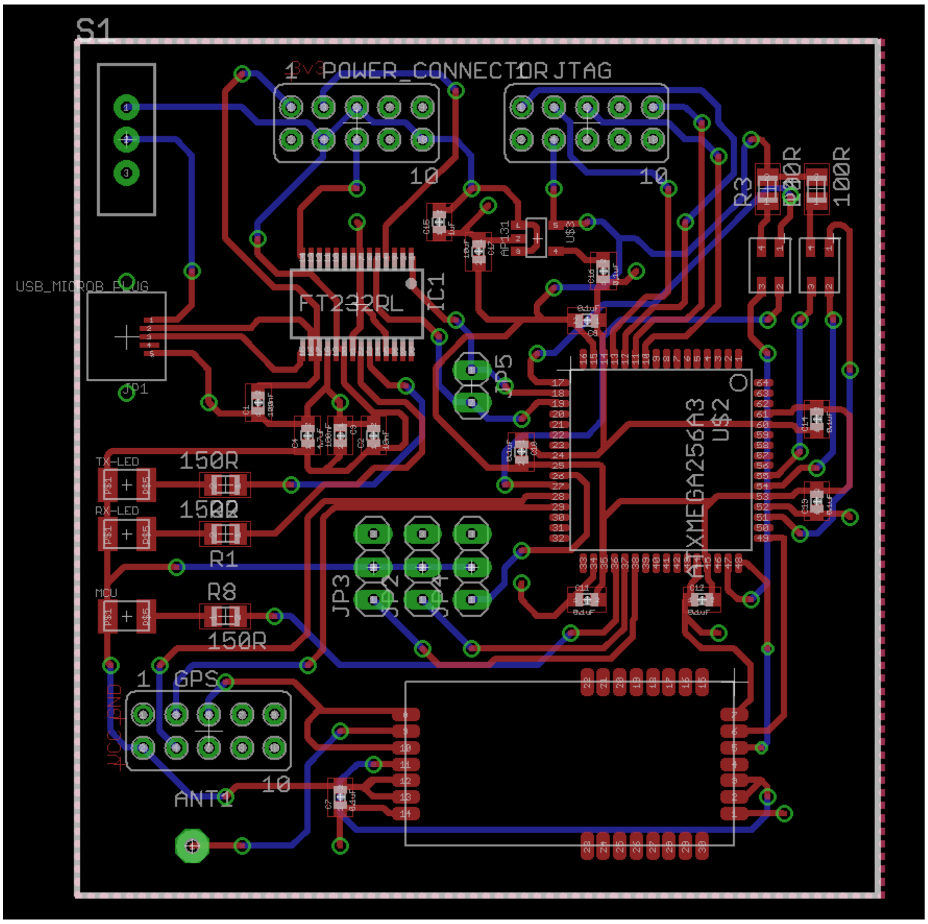


Figure D.1: Board Layout

Bibliography

[Eriksen(2007)] Mikael Kristian Eriksen. Ground station and hardware peripherals for fixed-wing uav: Cyberswan, 2007.

[Fidje(2010)] Viktor Fidje. Remote presence on offshore wind turbines, 2010.

[Karlsen(2010)] Tor Mæhlum Karlsen. Remote presence on offshore wind turbines, 2010.

[Koteich(2011)] Karsten Koteich, Mohammad & Rennæs. Control and communication for unmanned aerial vehicle, 2011.

[Press(2004)] National Academy Press. The global positioning system: A shared national asset, 2004.

[Radiocrafts(2012)] Radiocrafts. Rc232 embedded rf protocol user manual, 2012.

[Sølvberg(2007)] Audun Sølvberg. Cyberbike, 2007.