Institutionen för systemteknik
Department of Electrical Engineering

Examensarbete

Design of the Hardware Platform for the Flight Control System in an Unmanned Aerial Vehicle

Examensarbete utfört i Elektroniksystem vid Tekniska högskolan i Linköping
av

Mårten Svanfeldt

LiTH-ISY-EX--10/4366--SE

Linköping 2010
Design of the Hardware Platform for the Flight Control System in an Unmanned Aerial Vehicle

Examensarbete utfört i Elektroniksystem
vid Tekniska högskolan i Linköping
av

Mårten Svanfeldt

LiTH-ISY-EX--10/4366--SE

Handledare: Jonas Lindqvist
Inopia AB

Kent Palmkvist
ISY, Linköpings universitet

Examiner: Kent Palmkvist
ISY, Linköpings universitet

Linköping, 3 September, 2010
Design of the Hardware Platform for the Flight Control System in an Unmanned Aerial Vehicle

Mårten Svanfeldt

Abstract

This thesis will present work done to develop the hardware of a flight control system (FCS) for an unmanned aerial vehicle (UAV). While as important as mechanical construction and control algorithms, the electronics hardware have received far less attention in published works.

In this work we first provide an overview of existing academic and commercial UAV projects and based on this overview three different design approaches has been developed: network of independent microcontrollers, a central powerful CPU with helper logic and an field programmable gate array (FPGA) based approach. After evaluation the powerful CPU alternative with an ARM9 CPU is found to be most suitable.

As a final step this design approach is developed into a full design for the FCS which is evaluated and finally implemented. Initially a system incorporating an OMAP-L138 CPU, 256MByte DRAM, sensors and GPS is developed, however due to supply issues and cost limitations the final design instead incorporates a SOM-module with an OMAP35x processor, 128MByte DRAM as well as a sensor module and GPS. This design has been built and tested in the lab but not yet integrated into the UAV.

Keywords UAV, electronics, flight control system
Abstract

This thesis will present work done to develop the hardware of a flight control system (FCS) for an unmanned aerial vehicle (UAV). While as important as mechanical construction and control algorithms, the electronics hardware have received far less attention in published works.

In this work we first provide an overview of existing academic and commercial UAV projects and based on this overview three different design approaches have been developed: network of independent microcontrollers, a central powerful CPU with helper logic and an field programmable gate array (FPGA) based approach. After evaluation the powerful CPU alternative with an ARM9 CPU is found to be most suitable.

As a final step this design approach is developed into a full design for the FCS which is evaluated and finally implemented. Initially a system incorporating an OMAP-L138 CPU, 256MByte DRAM, sensors and GPS is developed, however due to supply issues and cost limitations the final design instead incorporates a SOM-module with an OMAP35x processor, 128MByte DRAM as well as a sensor module and GPS. This design has been built and tested in the lab but not yet integrated into the UAV.
Acknowledgments

I would first of all like to provide my sincere thanks to Jonas Lindqvist and Inopia AB for inviting me to take part in and contribute to this project. Even though we have worked together before, I appriciate the trust you put in me by giving me this opportunity.

I would also like to target a big thanks to Field Embedded Communications AB with CEO Henrik Bergström and CTO Jonas Persson for giving me an office to work in, a computer to work on, someone to call co-worker and invaluable technical guidance in areas such as high speed digital design and PCB layout, areas that are new to me. It has been a pleasure to work together.

Let me also take this opportunity to extend my gratitude to Kristoffer for reviewing the report and giving comments and to my examiner Kent Palmkvist for his input on the project and the report.

Last I would like to take the opportunity to extend a greeting to my family and friends. Without their encouragement and support I would have forgotten there is a life outside of work and writing a thesis report. My final thanks goes to Ms Lin without whos support this thesis and report would never have been completed on time.
# Contents

1 Introduction
   1.1 Background ............................................. 1
   1.2 Target vehicle ......................................... 2
   1.3 Goal ...................................................... 2
   1.4 Contribution ........................................... 3

2 Previous work
   2.1 Commercial and hobbyist UAV systems .................. 5
   2.2 Academic UAV systems ................................... 7
   2.3 Summary ................................................ 9

3 Design alternatives
   3.1 Design 1 - Network of microcontrollers ............... 11
   3.2 Design 2 - One powerful central processor ............ 14
   3.3 Design 3 - FPGA with softcore ........................ 17
   3.4 Summary ................................................ 19
   3.5 Common elements ....................................... 20

4 Flight control system design
   4.1 Initial design ........................................... 23
   4.2 Revised design ........................................... 33

5 Implementation
   5.1 Manufacturing .......................................... 41
   5.2 Bootstrap and verification ................................ 42
   5.3 Summary ................................................. 47

6 Summary
   6.1 Summary ................................................ 49
   6.2 Future work ............................................. 50

Bibliography .................................................. 51

A Requirement specification
   A.1 Requirements .......................................... 55
B FPGA Estimation

C Initial design details
  C.1 Analog filters .......................... 61
  C.2 PCB Layer stackup .................................. 62
  C.3 RFQ Responses .............................................. 62

D Revised design details
  D.1 PCB Layer stackup .............................. 64

List of Tables

  2.1 Summary of presented UAV projects .................. 9
  3.1 Design alternatives summary .............................................. 20
  3.2 Sensors - Three axis accelerometers ................. 21
  3.3 Sensors - Gyroscopes .............................................. 21
  3.4 Sensors - Small satellite navigation units ............. 22
  4.1 TI OMAP-L138 Specification .............................................. 26
  4.2 DDR2 bus simulation results .......................... 31
  4.3 SOM Modules .............................................. 35
  5.1 Power supply test results .......................... 44
  5.2 SOM module verification .............................................. 45
  5.3 Sensor module verification .............................................. 47
  B.1 FPGA Estimation - IP cores used for estimation ............. 60
  B.2 FPGA Estimation - Estimated implementation size ............. 60
  C.1 Initial design - Gyro filters .......................... 61
  C.2 Initial design - PCB layer stack .............................................. 62
  C.3 Initial design - PCB manufacturing quote ............. 62
  C.4 Initial design - Assembly quote .............................................. 63
  D.1 Revised design - PCB layer stack .............................................. 64

List of Figures

  3.1 Design alternative 1 - Block Diagram ............. 12
  3.2 Design alternative 2 - Block Diagram ............. 14
  3.3 Design alternative 3 - Block Diagram ............. 18
  4.1 Initial design - system block diagram ............. 25
  4.2 Initial design - PCB floorplan .................. 29
  4.3 Initial design - PCB layout .......................... 30
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4 Revised design - System block diagram</td>
<td>34</td>
</tr>
<tr>
<td>4.5 Revised design - Base board PCB layout</td>
<td>37</td>
</tr>
<tr>
<td>4.6 Revised design - Modules PCB layout</td>
<td>38</td>
</tr>
<tr>
<td>4.7 SOM software - Overview</td>
<td>38</td>
</tr>
<tr>
<td>5.1 Assembled base board</td>
<td>42</td>
</tr>
<tr>
<td>5.2 Assembled module boards</td>
<td>43</td>
</tr>
<tr>
<td>5.3 Verification setup</td>
<td>43</td>
</tr>
</tbody>
</table>
Acronyms

BSP  board support package.
CAN  controller area network.
FCS  flight control system.
FPGA field programmable gate array.
GPIO general purpose IO.
GPS  global positioning system.
I2C  inter-integrated circuit.
IMU  inertial measurement unit.
INU  inertial navigation unit.
MCU  microcontroller unit.
MEMS micro electro-mechanical systems.
PCB  printed circuit board.
RTOS real-time operating system.
SPI  serial peripheral interface.
UART universal asynchronous reciver and transmitter.
UAV  unmanned aerial vehicle.
Chapter 1

Introduction

1.1 Background

Man has been interested in being able to fly since the beginning of time, and flying machines have always attracted great attention. The earliest written account of a helicopter, or rotary wing aircraft as the technical term is, comes from China[22] in the 4th century and since then many big scientists and thinkers have dreamed about and designed different kinds of flying machines, some of them with fixed wings, some with moving. In 1907 the first practical design of a helicopter was built and since then they have become a natural part of the set of vehicles utilized by our society.

From the first manned flight it took another 60 years before the first unmanned rotary wing aircraft made its successful maiden flight in the early 1970s, notably 50 years after the creation of the first remote controlled fixed wing aircraft. It was a very complex toy that was incredibly hard to fly and in much that still is the case; radio controlled helicopters are available at most hobbyist stores but require a competent pilot with long training. Utilizing a remote controlled helicopter for professional flying such as for photo- and videography puts even higher requirements on the pilot to the degree that it is still very uncommon.

Thanks to the extremely quick development of electronic- and electromechanical systems in the last few years this is however about to change. In the last ten years a number of projects and products aiming to reduce the difficulty of flying helicopters and helicopter-like vehicles have been introduced, and today it is both technically and economically feasible to build a vehicle that is totally autonomous or provide a high degree of pilot support. Initially this technology was only available for military uses but now it is unregulated and also so cheap that commercial applications are possible.

It is in the light of this that the project this work is a part of was initiated. The overall goal is to design an unmanned aerial vehicle (UAV) for purely civilian use in professional photo- and videography, focusing on consumer market issues such as cost, reliability, ease of use and availability.
1.2 Target vehicle

Ideally there are two ways a flight control system (FCS) could be designed, just as any embedded system. Either it can be designed to work with any kind of UAV, or have a very specific and well specified target vehicle and be optimized for that target. This work however will take a middle stance as the physical vehicle design was finalized in parallell to this work, with design decisions influencing in both directions. It is however imperative that the FCS design is not too locked to this specific vehicle but rather provide an acceptable base for a number of UAVs.

This means that there is no physical vehicle setting the exact requirements for the FCS designed in this work. Rather this work use an earlier prototype in combination with the general description given below and the requirement specification in appendix A to define the external environment.

- The target UAV is a rotary wing aircraft capable of hovering and forward flight in any direction.
- The UAV is controlled either by direct control of the motor speed controllers via a dedicated bus or by control of aerodynamic surfaces via servo motors.
- The UAV is symmetric and provide a volume of approximately 20x20x5cm to house the electronics package.

1.3 Goal

This section defines the problem that this work is trying to solve. It also gives an outline to the steps taken to solve it and the organization of this report.

The overall goal for this work is to design the electronics hardware platform of the flight control system of the UAV described above such that it fullfills the requirement specification in appendix A. This should be done in a way so that the final design is manufacturable, cost efficient and provide some degree of flexibility for future upgrades and improvements. This will be accomplished by a three step process, which maps to the three major parts of this report.

The initial step is trying to analyse existing projects from commercial suppliers as well as previously academically published UAV designs from a hardware perspective. Especially for the comercial projects this comes down to qualified guesses and extrapolations from public information as manufacturers tend to be secretive with specific and accurate information, and we have not had physical access to any other vehicles. In general there might also be a problem with the amount of published information concerning the area of this work.

As a second step the results from the first will be used to develop three initial design alternatives and evaluate these on a theoretical level. The goal is to get three from a hardware perspective vastly different designs to explore as much as possible of the design space and in that way find an optimal mix of different design elements. During this phase focus will not be on specific circuits or board designs but rather on an architectural level, finding the right combination of processing elements and helper circuits.
1.4 Contribution

The third step will be to design the final systems based on the evaluation in the second step. This we consider to be the main part of the work covered in this report and the main goal will be to arrive at a final design fulfilling the requirements. This step will comprise system breakdown, component selection, design finalization, schematic capture and printed circuit board (PCB) design. At the end of the third step the design will also be implemented and an initial verification done to ensure the requirements are met. It is also likely that some software has to be implemented in order to verify proper operation of some components.

1.4 Contribution

This work and report makes contributions in topics including but not limited to:

- Making an overview of commercially and freely available UAV FCS systems from hardware perspective.
- Specify a number of criterias for evaluating the suitability of a FCS design.
- Presenting a number of different possible architectures and evaluating the suitability of these for an FCS.
- Designing a flight control system hardware including:
  - Overall architecture and design.
  - Component selection for all parts of the system.
  - Schematic capture and design.
  - Design of a flight worthy PCB.
- Setting up development environments for all the programmable components in the system.
- Modifying and writing the system level code needed to bootstrap the FCS.
- Verifying the basic functionality of the designed FCS by verifying correct execution of any processing elements, proper communication and functioning sensors.
Chapter 2

Previous work

This chapter will describe some of the previous work that have been done in the area of UAV platforms, focusing on UAVs for civilian applications. We will try to give a general overview of the construction of the systems from an electronics point of view based on available information. The intention is not to give an in-depth recollection or review of the constructions, neither to list all systems available on the open market. Focus for the review is the hardware of the flight control system but it will also include a general description of the full system.

In addition to the commercial/hobbyist projects this chapter will also list a few of the academic research projects and thesis projects related to UAVs. In the review of these we will also focus on the hardware aspect, even if the project itself has a different main research objective.

2.1 Commercial and hobbyist UAV systems

During the last few years a number of commercial UAV systems have been introduced on the market such as [21, 12, 29]. They all target the same market segment for small, semi-automatic UAV systems for recreational use or security and aerial photography purposes using small cameras. At the same time several people have developed similar systems on an hobby basis [28, 34] releasing both hardware designs and firmware source code under more or less permissive licenses. Common for most these projects, as far as information is available, is that they build their systems upon commercial off-the-shelf components.

2.1.1 MikroKopter

MikroKopter[21] provides an exception among the commercially available UAV platforms as the project have published not only exact specifications of the components used but also hardware schematics and the firmware source code under a license that allows it to be used for non-commercial purposes. The basic MikroKopter platform is made up of an aluminum structure with four propellers, four motor speed controllers and a flight control system. The standard platform

5
can carry a smaller camera as payload however the same set of electronics can also be utilized in larger constructions.

The flight control system of MikroKopter provides a combination of manual remote control via a standard RC radio and automatic stabilization and hover capabilities. The central processing element is an ATMega644 microcontroller unit (MCU) by Atmel Corporation with custom firmware. In addition the flight control system includes a LIS3L02AS4 three axis linear accelerometer by STMicroelectronics, three Gyrostar ENC-03 piezoelectric vibrating gyroscopes and a Freescale MPX4115A pressure sensor that together makes up the inertial navigation unit and absolute altitude computations.

Further there exists an addon board for navigation and automatic control. The navigation board is equipped with a STR911FAM44X6 ARMv9 based microcontroller from STMicroelectronics, an u-blox LEA-4H global positioning system (GPS) module and three Philips KMZ51 hall sensors that in combination with an Atmel ATMega168 functions as a compass. The navigation control system adds the functionality of logging current position, holding the current position by means of GPS and navigation along a pre-made flight path.

One thing that can be noted regarding the MikroKopter platform is that it is made up of multiple independent microcontrollers each handling one function. A four rotor system including the navigation board addon, with motor controllers, contains seven microcontrollers of four different types. This makes firmware uploads and updates a bit complicated both during the build phase and for updating as multiple programming sequences using different software is needed.

The distributed nature could result in a more reliable and failsafe system, however experience from the MikroKopter system we have access to have showed that the firmware does not always handle errors gracefully. For example if one motor controller indicates an error, real or only perceived, the main flight control system will shut down all other motors and naturally the UAV will crash.

2.1.2 DraganFlyer X6

The DraganFlyer X6 UAV by DraganFly Innovations Inc[12] is targeted at simpler aerial photography and surveillance applications such as within police and fire brigades. By utilizing six rotors configured as three pairs it can lift a payload of up to 500g. In addition to manual remote control using a custom 2.4GHz data link the DraganFlyer X6 can also automatically hold a given position using the built in GPS. During both flight modes the on-board software provides control augmenting for stabilization.

DraganFly Innovations Inc have not published any specifics regarding the components utilized in the DraganFlyer X6 however publically available material still gives some hints. For sensor input it utilizes three micro electro-mechanical systems (MEMS) linear accelerometers, three MEMS gyros, three magnetometers, one absolute pressure sensor and one GPS unit. From publically available image material we make the conclusion that DraganFlyer X6 utilize a monolithic electronics architecture with one larger processor element accompanied by sensors and auxiliary elements for communications. The exact architecture of the system
2.2 Academic UAV systems

is however not publically available and therefore we can not make any definitive conclusions regarding the X6.

2.1.3 Microdrones md4-1000

The md4-1000 model by Microdrones Gmbh[29] is another commercial UAV platform focusing on aerial video applications. Most of the technical specifications are still unknown, but it is based on the smaller md4-200 model with upgrades that increase the payload capability to 1.2kg. As the md4-200 it provides autonomous navigation using a combined approach of inertial navigation and GPS navigation. Microdrones also utilize a custom 2.4GHz data link for command uplink and status downlink.

According to [44] the earlier Microdrones UAV project used a ATM18 board which contains an ATMega88 microcontroller from Atmel Inc. We think it is probable that the md4-1000 also will be based on a microcontroller solution.

One interesting feature of the Microdrones system is their ground station, which however is outside the scope of this thesis.

2.1.4 Other

Apart from the three projects presented above there exists a number of other UAV projects doing single- or multirotor platforms. The Universal Aerial Video Platform project[28] is in many ways similar to MikroKopter presented above, utilize the same type of decentralized microcontroller based architecture but under a full open source license. They leverage the motor controllers from MikroKopter and use the same Atmel ATMega644 microcontroller for the flight control system.

2.2 Academic UAV systems

The academic sector also put effort into research related to unmanned aerial vehicles. A main focus for many researchers is control systems and sensor algorithms such as vision based navigation[20, 19, 10, 37], sensor fusion[17, 45] and simultaneous mapping and localization[25, 45, 32] referred to as SLAM. However in this work we focus not on the algorithms or methods but rather the hardware platform and unfortunately there does not seem to be many previously published articles or works within this area.

2.2.1 WITAS

One notable academic UAV project is the WITAS project[32] at Linköping University which was active during the years 1997 to 2005 with the resulting Autonomous Unmanned Aircraft Technologies Lab formed at Linköping University in 2004 continuing the work. The Witas project was a research project focusing on intelligent UAVs, autonomous flight algorithms, independent knowledge acquisition and processing.
Previous work

The WITAS project mainly utilize a hardware platform based on the Yamaha RMAX unmanned helicopter\[47, 19\]. The Yamaha RMAX is a traditional single-rotor helicopter driven by a two stroke engine. Included in the RMAX package is two proprietary modules called Yamaha Attitude Sensor, which is a six axis inertial measurement unit, and Yamaha Attitude Control System that provides automatic attitude stabilization and control. In addition to this the WITAS project adds an extensive flight control system and mapping system.

The WITAS FCS is made up of three main modules, each one hosted on a separate PC104 board running Intel X86 processors\[19\]. Two of the boards contains a Pentium3 processor with 256MB RAM and runs the primary flight control system and the image processing system respectivly while the third hosts a Pentium-M processor at 1.4GHz and 1GB RAM and contains the functionality for higher level control and planning. The three processor communicate over both ethernet, for non-critical information, and RS232C serial links for hard realtime communication. RS232C is also used for communication between the control processors and the sensor subsystem, and with the systems built into the Yamaha RMAX.

2.2.2 LinkMAV

The LinkMAV\[14\] UAV was also developed at Autonomous Unmanned Aircraft Technologies Lab at Linköping University but with different goals than the WITAS project. LinkMAV is a small (below 1kg take-off weight) single axis coaxial two-rotor helicopter. The basic platform incorporates all the mechanics, motor controller electronics and a commercial of-the-shelf avionics package from Micropilot.

At least one project for an improved flight control system based on the LinkMAV platform has been presented \[27\]. In his thesis Majewski evaluates a flight control system based on a DIOPSIS 940 multiprocessor-on-chip CPU by Atmel Inc. The DIOPSIS 940 incorporates both an ARM9 general purpose CPU core and an Atmel mAgicV DSP core.

Majewski made a couple of discoveries which might be of importance to our choice of architecture. The first is that the combination of an ARM general purpose core with a specialized DSP core does give an advantage for the algorithms used in a flight control system. For example using the mAgicV DSP core for computations needed in a Kalman filter for sensor fusion provides a 7 to 9 time speedup\[27, p. 51\]. The second discovery pertains to the suitability of a Linux operating system with patches for realtime operations and the ARM architecture.

The final important discovery made by Majewski is that the DIOPSIS 940 is not suitable as the single processing element. The results presented\[27, p. 58\] show that using a multitasking CPU with Linux operating system does not provide accurate enough timing measurement of external inputs to handle input and output of pulse-position modulated signals to a normal RC remote control and servos. They solve it by combining the DIOPSIS 940 processor with an field programmable gate array (FPGA) device handling these timing sensitive inputs and output totally independent from the main processor.
2.3 Summary

<table>
<thead>
<tr>
<th>System</th>
<th>Characteristic</th>
<th>Electronics architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>MikroKopter</td>
<td>4-8 axis multirotor</td>
<td>Distributed, microcontrollers</td>
</tr>
<tr>
<td>DragonFlyer X6</td>
<td>3 axis coaxial multirotor</td>
<td>Centralized</td>
</tr>
<tr>
<td>md4-1000</td>
<td>4 axis multirotor</td>
<td>Probably microcontrollers</td>
</tr>
<tr>
<td>WITAS</td>
<td>Single axis rotor</td>
<td>Distributed, x86</td>
</tr>
<tr>
<td>LinkMAV</td>
<td>Single axis coaxial</td>
<td>ARM+FPGA</td>
</tr>
<tr>
<td>MonashUni</td>
<td>Fixed wing</td>
<td>Distributed, microc.+FPGA</td>
</tr>
</tbody>
</table>

Table 2.1: Summary of presented UAV projects

2.2.3 Monash University

The Aerobotics Group at Monash University[15], Australia, performed a wide range of research related to UAVs between 1999 and 2005. Focusing on fixed-wing UAVs they developed a number of UAV platforms including Duigan and P15035 high wing aircrafts. Both of them are electrically powered and have replaceable autopilot module.

For some research projects the Monash University group utilized a commercial of-the-shelf autopilot from MicroPilot, focusing on other aspects of UAV navigation such as vision based navigation[6] and landing. They however also performed at least one project for an FPGA based primary flight control system[23]. In the paper Young and Price presents that a flight controller based on an FPGA with a soft-core for general processing and dedicated modules for different kinds of IO and acceleration provides a feasible alternative. Unfortunately they do not provide any details to what FPGA was used or the exact results obtained.

2.3 Summary

In table 2.1 we provide a summary of the UAV projects discussed in this chapter, their physical characteristic and a characterization of the electronics architecture utilized.

What can be noted is that some of these projects, especially WITAS and the projects at Monash University, have very different goals and design requirements compared to this work. For WITAS the much higher take-off weight and power capabilities reflects in the choice of control system components, while Monash University focus on fixed wing systems and thus the control focus on forward flight, not stable hovering or slow stable flying.

One important note to make is that none of the presented projects have provided any rationale for their choice of hardware, even in the cases when the exact architecture is known. This leads us to believe that in at least some cases the choice has been more or less arbitrary, utilizing components known to the designer of the projects and possibly not the best available components.
Chapter 3

Design alternatives

This chapter will introduce three different possible designs of the flight control system. Based on the information gathered from other projects in the previous chapter combined with general embedded system knowledge three basic designs, or architectures, for the electronics subsystem was developed and will be presented. For each design a discussion of advantages and disadvantages in relation to the requirement specification is also provided. All information regarding specific specifications or requirements of circuits are, unless otherwise noted, extracted from the datasheets provided by respective manufacturer.

Focus of this section is not to have three detailed designs with all components selected and matched to each other but rather to provide an overview of three vastly different design approaches and provide a good foundation for a more detailed design later on.

3.1 Design 1 - Network of microcontrollers

3.1.1 Overview

The first design is based on the MikroKopter project presented in section 2.1.1 and could be characterized as “network of microcontrollers”. The flight controller is divided into three different tasks: sensor data input and handling, remote communication and the main stabilization and navigation. Each of these three main tasks is handled by a separate MCU. A microcontroller is an integrated device that in the same chip includes a central processing unit, RAM memory, some kind of program memory (usually flash-memory) and drivers for different IO protocols. Usually they also expose many general purpose IO (GPIO) pins that can be utilized for driving custom logic or to provide interface to other circuits.

An overview of the architecture with the MCUs, the sensors and the communication buses and protocols can be seen in figure 3.1

The first MCU handles all sensor input acquisition, both analog and digital, digital filtering of the signals and format conversion. By selecting a microcontroller of sufficient size one could also do some other processing that depends heavily on
sensor inputs such as integration of velocities, positions and attitudes. Having all the sensor inputs in the same microcontroller can also become an advantage when one wants to implement sensor fusion algorithms as those need access to all inputs at the same time, preferably acquired with low jitter and good relative timing.

The second MCU, referred to in figure 3.1 as IO MCU or IO-controller, acts as a communication gateway between the internal and external buses such as the radio link to the base station, USB and RC-servo control of the payload and general purpose IO for future extensions. It also provides an serial peripheral interface (SPI) interface to a small memory card such as an microSD for reading configuration and writing a flight log. An alternative approach could be to connect the the pulse-position modulated (PPM) input from the RC radio to the IO-controller MCU, however this introduce extra latency and failure points in the manual control loop and thus we decided not to do this.

The third MCU is the main flight controller that contains the main flight control loops. Based on the input from the sensor and communication MCUs it computes the corrections needed to maintain stable flight, to hold altitude and to follow any preloaded flight plan and sends these to the motor controllers. The flight control MCU also takes direct control input in pulse-position modulated format from an of-the-shelf RC radio system, either as an override of the autonomous flight or as the main steering input augmented by automatic stabilization.

### 3.1.2 Possible devices

In order to be able to evaluate the advantages and disadvantages we must first define what is referred to by microcontroller and also some of the specific devices to consider. It is somewhat hard to find a single definition for microcontroller and what distinguish it from a microprocessor system, but in this work we have adopted the view that a microcontroller is a single chip integrating a microprocessor (CPU),
a ROM or flash-memory containing the software, RAM memory and some kind of I/O devices. It might also contain interfaces for specific bus protocols such as inter-integrated circuit (I2C) or controller area network (CAN).

Utilizing only this restriction there are too many specific devices available to be able to draw any conclusions, so to make the comparison with the other alternatives easier the selection have further been restricted to 8-bit microcontrollers, as these at least until recently was the most common kind, manufactured by Atmel Inc (the ATmega series)[8] and ST Microelectronics (ST8 series)[38]. These families provide a wide series of MCUs operating in the range of 8 to 40 MIPS, having 4kB to 128kB of flash and 1kB to 6kB of RAM. Many of the devices also contains AD-converters for analog input and controllers for the I2C bus.

3.1.3 Advantages and disadvantages

The first obvious advantages from utilizing smaller microcontrollers is related to the fact that they comprise one small compact unit integrating both central processing unit and memory subsystems. This results in a simpler PCB design and manufacturing with fewer components, as well as a simpler layout with less constraints as high speed memory buses require precise layout[26] for good performance. Microcontroller units also usually have lower clock rate than stand-alone microprocessors which also makes PCB design somewhat simpler. In addition to this, in the event of a partial redesign of the hardware, for example replacing an analog sensor with a digital counterpart, having several smaller and more separated parts means that one can be replaced without a system wide impact.

The division into three separate units on hardware level also gives a natural modularization of the software. For initial software development this can be an advantage as it forces the same modularization however in the long term it can become a problem. If the initial estimation of the required memory or flash size in the MCU is too small one can end up in a situation where one microcontroller is fully utilized while there still is capacity in one of the other MCUs. To solve this you would have to either redesign the hardware or break the modularization, neither of which is desirable.

A third possible advantage of the independent modules relates to fault isolation and fail-over. Given enough care in hardware and software design the system could handle that one microcontroller dies or reboots in a graceful manner. A total loss of the communications MCU should leave normal operation of the flight controller and sensor MCUs unaffected and thus not impact normal flight operations however total loss of any other microcontroller unit will make operation impossible. In that case the only advantage provided is an ability to reduce the impact of the crash and to log data for post-mortem analysis.

Most major drawbacks of the network of microcontrollers architecture comes directly from the fact that it contains multiple processing elements. By utilizing three microcontrollers there are three sources of faults, both during construction and runtime. In construction and setup phase having three separate units increase the risk of an error in PCB design, manufacturing and assembly as it is three circuits with supporting logic. It also requires three different firmwares to
be developed, debugged and uploaded into the flash memories. The last parts, debugging and uploading, can be mitigated a bit by utilizing a single JTAG debugging chain but it does not remove the problem totally especially as some MCUs and toolchains does not support chained JTAG devices.

A final problem with this design alternative is related to pure processing power. Most microcontrollers give approximately one instruction per cycle and frequencies up to 40MHz with a few devices running at somewhat higher frequency, resulting in a total system computational capacity of 120 million instructions per second (MIPS) which can be compared to design alternative 2 and 3 which can get above 1000 MIPS.

### 3.2 Design 2 - One powerful central processor

#### 3.2.1 Overview

The second design alternative, for which an overview drawing is seen in figure 3.2, is designed to be an opposite to design 1. Whereas design 1 builds on a modular design with each logical function implemented by its own physical processor, design 2 is based on a single monolithic CPU running all the functions in the system.

By integrating all the functionality of the flight control system: sensor data input, flight stabilization and sending motor control commands to the motor controllers the big CPU replaces the three MCUs of design 1. The central CPU also handles secondary tasks such as communicating with the ground station over a wireless link, interface with permanent storage such as microSD and if required interface with the payload.

Utilizing a larger CPU allows the usage of a COTS/standardized operating system such as Linux which can provide a number of advantages. It does however limit the available devices somewhat and introduces additional steps in the software bootstrap procedure, but we believe the advantages are bigger and the design should allow for this.

Based on the experiences in [27, p. 19], where a designed based an ARM9 CPU running Linux is presented but found to have a too high interrupt latency for
directly interfacing timing critical parts such as the pulse-position modulated radio and RC servos, the design also include a secondary unit for example a FPGA or microcontroller to handle these timing critical external interfaces. The secondary unit can also provide an efficient way to add GPIO pins to the design.

The design probably also needs to integrate separate memory ICs, bus interfaces and other auxiliary circuits that are not integrated within the CPU but needed to complete the system. These have for simplicity not been included in this overview.

### 3.2.2 Possible devices

There exists a large number of devices representing many different architectures to select from when it comes to stand-alone processors, ranging from IA32-architecture from Intel for PC systems to very small processors such as the Atmel AVR32 CPUs. Based on the goal to utilize a commercial or free off-the-shelf operating system as well as an advantage by having a general purpose CPU architecture that is well proven and long lived, it was decided to focus on ARM based processor. ARM is a family of architectures designed by ARM Holdings. ARM Holdings provide the CPU as basic IP block that then is implemented together with memory controllers, IO devices and other units by a number of companies including ST Microelectronics, Freescale, Marvell and Texas Instruments (TI). In 2009 it was the most commonly sold processor architecture.

Most of the ARM based processors focus on either telecommunication devices, such as the TI OMAP1-3 product lines[42], or multimedia devices, such as Freescale i.MX[18] and TI DaVinci[41] product lines. In this project it is no advantage to have hardware acceleration for 3G mobile communication or high resolution video output, so such auxiliary units only results in a more expensive and power consuming design. As a result focus have been put on general purpose ARM9-based CPUs from ST Microelectronics and Texas Instrument.

As noted above this design also requires a processing unit for real time signals and for this purpose we suggest utilizing either a small FPGA such as the Xilinx Spartan3AN (XC350AN), a CPLD or a microcontroller unit. If utilizing a MCU it is an advantage if it is based on ARM7 or ARM9 as a common toolchain can be shared between the main processor and the IO processor.

### 3.2.3 Advantages and disadvantages

Just as design of alternative two is in many ways the opposite to alternative one, its advantages and disadvantages in some ways are a mirror those of design alternative one.

One of the main advantages of the centralized processor design is the high processing power and low intrinsic overhead. The ARM9 core studied provides about 1 MIPS per MHz[7] for general code and as most devices support clocking from 200MHz up to, or exceeding, 600MHz the total performance is on the order of 200-600 MIPS which is several multiples more compared to the microcontroller solution in design alternative 1. Several of the ARM-based devices also contains
hardware accelerators for different functionality areas that provides even more performance. Apart from the earlier mentioned communication or video accelerators several of the studied processors integrate general purpose digital signal processors (DSP) that can greatly accelerate numerically heavy computations such as filtering of sensor signals, inertial navigation mechanization and the innermost control system loops and even though these types of accelerated computations are not needed initially, future development might benefit from having access to them. In some cases the peak performance of the DSP co-processor is equal to or exceeds that of the CPU, especially for purely numerical work loads.

Having all the functionality centralized also improves performance by reducing the communication overhead. In this design the different components can share data and commands via high performance inter-process communication methods such as shared memory instead of utilizing external busses such as I2C or SPI that has both higher latency and lower bandwidth.

Another big advantage of using a larger CPU with MMU is the possibility to utilize one of several full scale operating systems, including Linux, QNX or Windows. While this does not in itself provide any technical advantage, it can greatly reduce the time and complexity of software development. Given that the processor and accessories used have adequate support, which especially for the combination of ARM9 and Linux is good, the operating system will directly handle the main peripheral units such as communication busses, SD card interface and serial communications. Having an operating system also provides a pre-structured framework for integrating custom drivers for sensors and other devices that are not supported directly. Together these two parts allows for a reduction of the total software development time.

The third big advantage is also related to software development. Having a larger processor and a full scale operating system means that there are a big number of pre-made software components available under different licenses, both commercial and free licenses. Licensing already written code both results in a shorter total development time and possibly higher quality as focus can be on testing and integration instead of implementing basic functionality.

The increased complexity of the processor system, while providing the above mentioned advantages, also introduce a number of disadvantages into the design. The first one is that it puts much higher requirements on the design and manufacturing process of the printed circuit board. Modern CPUs, such as the TI DaVinci line of processors, utilize external memory busses operating at, or above, 200MHz effective frequency\(^1\). This puts very tight constraints on the on-board matching between different signal traces both in terms of impedance matching and matching between the trace lengths[13].

The circuit board design is also complicated by the fact that it must utilize many layers, both to provide enough stable reference planes for the impedance controlled busses and to be able to fan-out all signals. As a comparison many ARM9 processor implementations specify that a PCB should utilize 6 to 8 layers at least compared to the 2-4 layers needed for a MCU solution. A more complex circuit board does not only mean it is harder to design, but also more complicated.

\(^1\)Either directly or by utilizing dual-data rate (DDR) technology
and costly to manufacture. While most PCB manufacturing companies provide standard processes for 1, 2 and 4 layer boards, only some provide more layers. Initial contact with one manufacturer indicates that each additional pair of layers increase the cost by about 10%.

Another factor which increase complexity of board design and manufacturing is the type of package utilized for many ARM9 circuits and memories, high density BGA packages. While most circuit packages have connections or leads along the edge of the package, a BGA package consists of a small piece of circuit board with a high number of connection points on the bottom side. Connection between the package and the main circuit board is achieved by small balls of solder that are attached to the package on manufacturing. While this results in a package with very high connection density with low pitch, it also means that hand soldering of BGA packages virtually impossible and X-ray techniques are required for reliable inspection[39].

Compared to design 1 with separate processors for different functions this design might also offer a lower degree of redundancy. As all the main functionality is concentrated into a single processor, the design also contains a single point of failure which might result in a lower overall reliability of the system compared to a system with distributed points of failures. We believe it can me mitigated or solved in a later design by either using external watch and security circuits, either separate or to combined with the IO-processing unit.

3.3 Design 3 - FPGA with softcore

3.3.1 Overview

Design alternative 3 takes a different approach compared to alternative 1 and 2. While alternative 1 and 2 focus on designing the best possible flight control hardware in the first iteration, this alternative focus on having a very extensible design which can be reconfigured and adapted in the future by utilizing a FPGA as the main processing element. An FPGA is an integrated device that contains a large number of small but fixed processing elements as well as runtime configurable interconnect that determines the final functionality.

In most cases FPGA devices are programmed in a hardware description language such as VHDL or Verilog, which compared to the high level languages or assembly languages used for general CPUs and MCUs provides a very low level of abstraction, which results in longer development time with more issues and means extensive verification needs to be performed. As this is a problem for all FPGA designs there exists a number of standard solutions to increase productivity and decrease the number of bugs.

The first is to utilize premade building blocks, so called “IP-cores”. IP-cores are libraries of functionality such as processors, control elements for busses or custom accelerators that communicate on some standardized interface. By combining pre-made IP-cores together with custom made functionality and glue logic development time can be decreased and number of bugs be reduced at the same time, although the use of IP-cores are not universally considered positive[46]. IP-core
blocks can either be obtained under a number of free or costless licenses from for example OpenCores[31] or be purchased from a number of different companies.

The second technique to reduce development time is to include a CPU programmable in a high level language into the FPGA. Some FPGA devices includes pre-made CPU cores in the silicon, however this is pretty rare. A more common way is to utilize an IP-core that implements a general purpose CPU in the programmable logic of the FPGA, which is called a "soft-core"[33]. While this approach use more space, power and requires a bigger device than directly implementing all functionality directly in Verilog or equivalent, it can provide a good trade-off in terms of development time and complexity. Thus any FPGA based solution should utilize some form of soft-core.

3.3.2 Possible devices

The market today contains a huge number of FPGA devices ranging from very small ones containing a few thousand gate-equivalents to extremely large ones that can replace modern high-end computers. Apart from the two biggest manufacturers of FPGA devices, Xilinx[24] and Altera, who each carries several product lines of multiple devices there are a big number of smaller companies providing both general and special FPGA devices targeted at specific application domains.

To be able to evaluate possible devices we have in this work limited ourselves to mid-range devices from the Spartan3AN and Virtex4 product families from Xilinx, due to previous experience with these devices and access to appropriate toolchains. Utilizing the Xilinx Platform EDK a small mock-up FPGA design was created including a CPU, memory controller, SPI and I2C controllers and
universal asynchronous receiver and transmitters (UARTs) to estimate the size of
the FPGA required. The IP-blocks used and the result from synthesis can be
found in appendix B. From this it can be concluded that to fit the required
design, including some possibility of future expansion, a device of at least the size
of an XC3S1400AN is required or in other words about 1.4M gate equivalents and
500kbits of on-chip RAM.

The choice of device also affects which soft-core can be utilized as most com-
mercial soft-cores are tailored towards devices from one specific manufacturer and
the freely available cores are only tested on some devices so using another device
might require extensive customization and testing.

3.3.3 Advantages and disadvantages

The major advantage of, and the sole reason to, utilize an FPGA is the extreme
programmability and extensibility of the hardware. As an FPGA can be pro-
grammed to perform any logical or sequential function or combination thereof and
also map input/output to more or less any pin of the device, basically any com-
putational operation or protocol processing can be accelerated and customized as
required.

In general the development time for an FPGA based design is much higher com-
pared to using an MCU or general purpose CPU to solve the same problem[35, 24],
as the FPGA is programmed using lower level of abstraction and more hardware-
like techniques. The use of a pre-made soft-core that can be programmed in a
similar way to a normal CPU, in either assembly or C, mitigate this a bit but to
utilize the full generality of an FPGA it needs to be programmed using lower level
techniques.

Another issue related to the development on FPGA platforms is the required
toolchain for compiling Verilog or VHDL into a vendor specific bit-format. For
some devices there exist free tools, either limited editions from the vendors or
open source tools, but in most cases development on FPGAs require very expen-
sive tools. In the same way there exist some free soft-cores and other IP-blocks,
however most high performance and highly functional ones must be licensed at a
considerable cost.

From a board construction perspective an FPGA based design offers similar
challenges to high end ARM processors described in an earlier section. Just as the
ARM processors FPGA devices of the required size are packaged in fine pitch BGA
packages with several hundred or up to a thousand pins and combined with high
speed external memory devices this results in a PCB design with a high number
of layers and very small feature sizes and spacings. All the disadvantages of BGA
packages discussed in section 3.2.3 applies in this case as well.

3.4 Summary

Table 3.1 presents a summary of the three design alternatives and the defining
characteristics of each of them.
### 3.5 Common elements

In addition to the earlier described central processing elements, a fully functional flight control system requires a number of additional components. Evaluating all aspects of these auxiliary systems in detail would amount to a full thesis in itself, so in order to finish a design we have decided to rely on evaluations made by others and allow for non optimal selections as long as these can be improved or changed later in the lifetime of the product.

#### 3.5.1 INU units

One of the most important components to ensure good navigation performance is the inertial unit, usually referred to as inertial measurement unit (IMU) or inertial navigation unit (INU). The main constituents of an inertial navigation unit is a set of accelerometers and gyros measuring the linear and angular movement around the three principal axes, logic to process the output and calibration circuits.

Designing and evaluating the performance of different INUs is well beyond the scope of this work and utilizing tactical or aviation grade INU units is infeasible due to cost, weight and power consumption. This leaves only consumer grade inertial measurement circuit. Several articles\[17, 36\] have shown that it is possible to get good accuracy and repeatability using cheap off-the-shelf MEMS accelerometers and gyros by incorporating cleaver calibration techniques and a number of error correcting calculations. These types of devices are available both in fully integrated versions and in simpler one-function ICs from a number of suppliers.

As can be seen from table 3.2, which lists some common three axis MEMS accelerometers, and table 3.3 which lists some MEMS gyroscopes, the performance of accelerometers and gyroscopes from different manufacturers are essentially the same and the only real difference is the interface and cost/availability. This also means that selecting “the best” accelerometer or gyro will not be as big issue when it comes to making a well functioning IMU but rather focus needs to be on proper interfacing and software.
### 3.5 Common elements

<table>
<thead>
<tr>
<th>Parameter</th>
<th>AD ADXL327</th>
<th>AD ADXL345</th>
<th>ST LIS331AL</th>
<th>ST LIS331DLH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Analog</td>
<td>I2C/SPI</td>
<td>Analog</td>
<td>I2C/SPI</td>
</tr>
<tr>
<td>Range</td>
<td>±2 g</td>
<td>±2 – 16 g</td>
<td>±2 g</td>
<td>±2 – 8 g</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>2.38 g/V</td>
<td>3.9 mg/LSB</td>
<td>2.08 g/V</td>
<td>1 – 4 mg/LSB</td>
</tr>
<tr>
<td>Offset</td>
<td>470 mg</td>
<td>40 mg</td>
<td>180 mg</td>
<td>20 mg</td>
</tr>
<tr>
<td>Noise</td>
<td>250µg/√Hz</td>
<td>290µg/√Hz</td>
<td>300µg/√Hz</td>
<td>218µg/√Hz</td>
</tr>
<tr>
<td>Cost (aprx)</td>
<td>4.50 USD</td>
<td>6.10 USD</td>
<td>11.80 USD</td>
<td>8.70 USD</td>
</tr>
</tbody>
</table>

Table 3.2: Sensors - Three axis accelerometers

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ST LPR430AL</th>
<th>ST LY330ALH</th>
<th>AD ADXRS610</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface</td>
<td>Analog</td>
<td>Analog</td>
<td>Analog</td>
</tr>
<tr>
<td>Type</td>
<td>Pitch/roll</td>
<td>Yaw</td>
<td>Yaw</td>
</tr>
<tr>
<td>Range</td>
<td>±300/1200deg/s</td>
<td>±300deg/s</td>
<td>±300deg/s</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>300/1200deg/s/V</td>
<td>266deg/s/V</td>
<td>166deg/s/V</td>
</tr>
<tr>
<td>Noise</td>
<td>18mdeg/s/√Hz</td>
<td>14mdeg/s/√Hz</td>
<td>50mdeg/s/√Hz</td>
</tr>
<tr>
<td>Cost (aprx)</td>
<td>10.00 USD</td>
<td>10.00 USD</td>
<td>31.50 USD</td>
</tr>
</tbody>
</table>

Table 3.3: Sensors - Gyroscopes

#### 3.5.2 Satellite navigation units

The second most important sensor in the navigation system is one or more receivers for satellite navigation systems such as the American GPS or European Galileo. Satellite navigation works by receiving very precise timing-information from a number of satellites which together with exact information about the satellite orbits can be used to calculate the position of the receiver. Using the commercially available encodings and chips results in absolute position reports with an accuracy of a few meters with very low and slow drift, i.e. even if the position is two meters off target it will have the same offset during a number of consecutive updates.

The type of satellite navigation unit that is interesting for this application is fully integrated units that contains signal amplifier, decorrelator, position calculation logic and communication interface for one or several standardized serial protocols. There exists a number of circuits like this, a selection of the considered units is listed in table 3.4 together with the most important parameters considered during the selection of unit.

In addition to the normal satellite navigation systems that utilize encoded timing signals there exists another way of obtaining very exact relative positions in relation to a base station called real-time kinematic GPS (RTK-GPS)[16] that works by correlating the phase information in the carrier wave from multiple satellites at both the base station and on the mobile unit. This gives an offset precision on the range of centimeters at the expense of vastly increased startup and fix-times. There might also be options to utilize a combination of traditional satellite
navigation and RTK-GPS to gain the advantages of both.

For this application the satellite navigation unit has two purposes. The first is the obvious function of providing absolute position to the navigation and waypoint subsystems and then the absolute precision and accuracy is important to properly navigate the environment. The second function of the satellite navigation system is to provide calibration and bias elimination in the inertial navigation unit. For this it is more important of having a stable position report rather than absolute precision so that when the vehicle is stable (not moving) the position reports can be used for zeroing the bias in the inertial navigation unit. For stable operation the second function is more important so a traditional system should work sufficiently well for most cases.

<table>
<thead>
<tr>
<th>Unit</th>
<th>Interface</th>
<th>Time-to-fix</th>
<th>Update rate</th>
<th>Sensitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>u-Blox LEA5S</td>
<td>TTL UART</td>
<td>29 s / &lt;1 s</td>
<td>4 Hz</td>
<td>-160 dBm</td>
</tr>
<tr>
<td>Trimble Copernicus</td>
<td>TTL UART</td>
<td>39 s / 9 s</td>
<td>1 Hz</td>
<td>-152 dBm</td>
</tr>
<tr>
<td>GlobalSat EM408</td>
<td>TTL UART</td>
<td>42 s / 8 s</td>
<td>1 Hz</td>
<td>-159 dBm</td>
</tr>
</tbody>
</table>

Table 3.4: Sensors - Small satellite navigation units
Chapter 4

Flight control system design

Based on the initial development and analysis of the three alternatives presented in chapter 3 a thorough evaluation was made by the steering group behind the project that includes the author, the supervisor and other technical experts as well as other members of the project team. Based on the advantages and disadvantages of the three alternatives, a decision was made with regards to the overall architecture of the final system.

This chapter is divided into two parts; the first part presents the initial design developed based on the overall architecture alternative decided upon and includes component choices and detailed design of all circuits. It also includes the final circuit schematics and board layouts as well as a theoretical evaluation of the design including manufacturing constraints and a cost analysis.

The second part of the chapter will present a refined design based on an evaluation of the initial design. It does not redo the design from the beginning but rather build upon the circuits developed during the first step while modifying them to solve problems encountered in the evaluation.

4.1 Initial design

Based on the advantages and disadvantages presented in chapter 3 the decision was made to base the design of alternative 2 from section 3.2, a single high-performance central processing element. The final decision was made based on a combined evaluation of all identified aspects of each design, but a number of factors are considered as more important than others in favour of the second alternative.

- High level programmable and standardized programming environment
- Ability to use general purpose/standardized operating system and programs
- Processing power

Especially the programmability and reducing the barrier for software development is important, even if it is at the expense of a more complicated hardware
design. As noted in [11] most embedded systems development projects spend a majority of time and resources on software development, and we expect the same pattern to emerge in this project even though the design cycle is much shorter than the one Debardelaben et.al. present. It is also expected that the hardware will have a much longer life time than the software, as the software will go through several iterations of refinement while still executing on the same hardware platform.

The usage of a general operating system also simplifies software development as it gives access both to a premade set of hardware device drivers and framework for building new ones as well as a large set of open source and commercial software development libraries. In the long term it can also simplify software development as a known environment reduces the entry barrier for new developers and might allow the use of rapid prototyping tools.

Another big advantage of the second design alternative is the pure processing power. While this might not be required initially it is almost universally true that projects, especially those involving software, tends to expand beyond the initial requirements. In light of this having more processing power than initially required will give a longer life time as future expansion won’t be limited by hardware constraints.

4.1.1 Hardware design

Based of the experience presented in chapter 2 and section 3.2 a decision to utilize a fully integrated system with a single circuit board containing all parts of the finished flight control system was made. The main advantage of integrating everything into a single board comes down to reliability. A connector is an inherently unreliable component as it is designed for deattaching parts of the system and although with correctly chosen board-to-board connector the risk of unintentional deattachment is very low it is still greater than zero. A single board solution should also lower manufacturing cost as many steps in the manufacturing process incures per-board setup costs.

To simplify the board design and reduce possibility of injecting noise into the systems the decision was made to interface all component, as far as it is possible, using standard serial interfaces such as I2C and SPI and to keep analog sections of the circuits as small and confined as possible. Utilizing high speed SPI busses also has the advantage of reducing the risk of bus collisions with following measurement jitter and latency, which is an advantage when doing more advanced types of processing such as sensor fusion[30]. Further, care has be taken to sufficiently decouple high speed signals and to separate analog and digital signals.

The final system-level diagram can be seen in figure 4.1.

CPU and memory

As noted in section 3.2.2 there exist a vast number of processor devices that could work well for our intended application. To sufficiently limit the search a decision was made to focus on devices fullfilling
4.1 Initial design

- ARMv5 or later architecture with MMU\(^1\)
- High number of available serial communication interfaces such as I2C, SPI and UART
- Integrated DSP an advantage, or at least if other chips in same family exists with integrated DSP
- Packaged in a package that most contract manufacturing firms can handle during assembly

The fourth requirement disqualified the latest chips from Texas Instrument including the OMAP35xx series as they utilize a so called package-on-package (POP) configuration for the memory interface, which is very new and as far as we have been able to determine not standard practice for most contract manufacturing firms.

After evaluating the specifications and datasheets of a large number of devices the decision was made to utilize a combined ARM9 and DSP circuit, OMAP-L138 from Texas Instrument. The TI OMAP-L138 integrates an ARM926EJ-S CPU clocked at 300MHz, a C674x series floating point DSP also running at 300MHz and a number of peripheral interfaces. A summary of the specifications can be found in table 4.1.

\(^1\)MMU required for Linux, Windows and other operating systems
To maximize performance, and also to maximize utilization of the heavily multiplexed IO pins on the OMAP-L138 CPU, the system use DDR2 memory connected via dedicated DDR2 bus. At the time of design the highest memory density that could be integrated with the OMAP-L138 was 2 Gbit (256 MByte), for example by utilizing two Micron MT47H128M8 memory modules. Each module provides 8 banks with 16 M words of 8-bit length and are connected in parallel to the 16-bit databus of the OMAP-L138 DDR2 interface in accordance with the manual[43]. To reduce the risk of problematic RF emissions and crosstalk series termination have been used on the data bus, but only the data bus as it operates at a higher frequency than the address bus.

For system and program storage we made the decision not to utilize a normal parallely attached NOR or NAND flash chip to save pins and reduce the fan-out. Instead a relatively small serial (SPI) flash of 8MByte will be used to store boot routines while the operating system and the flight control system is stored on one of the two SD cards. This gives a combination of a large storage capacity, easy upgrades in the field and good reliability while maintaining low cost and easy circuit design.

The OMAP-L138 processor and the DDR2 memory circuits are all very high speed digital devices and as such require good power supply. Significant care has been taken to properly decouple and filter both supply and ground networks. Decoupling is done by a mix of large number of 10 nF and 100 nF capacitors to handle high frequency noise combined with larger 10 uF bank capacitors for longer transients.

As IO processor it was decided to use an ARM7 based microcontroller from Atmel, as a development environment for this family of processors was accessible during the project and previous experience of these MCUs is good. The specific

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>ARM926 EJ-S</td>
</tr>
<tr>
<td>DSP</td>
<td>C674x</td>
</tr>
<tr>
<td>Clock frequency</td>
<td>300 MHz</td>
</tr>
<tr>
<td>On-chip L1</td>
<td>64 kByte</td>
</tr>
<tr>
<td>On-chip L2</td>
<td>256 kByte</td>
</tr>
<tr>
<td>Memory interface</td>
<td>DDR2 + SDRAM/NAND/NOR</td>
</tr>
<tr>
<td>UART</td>
<td>3</td>
</tr>
<tr>
<td>SPI</td>
<td>2</td>
</tr>
<tr>
<td>I2C</td>
<td>2</td>
</tr>
<tr>
<td>USB</td>
<td>USB2 (OTG) + USB1.1 (host)</td>
</tr>
<tr>
<td>Power supply</td>
<td>1.2V, 1.8V, 3.3V</td>
</tr>
<tr>
<td>2IO voltage</td>
<td>3.3V</td>
</tr>
<tr>
<td>Package</td>
<td>361 ball BGA</td>
</tr>
<tr>
<td>Cost (aprox)</td>
<td>38 USD</td>
</tr>
</tbody>
</table>

Table 4.1: TI OMAP-L138 Specification
device SAM7A3 was choosen for its extensive set of communications protocols including two CAN controllers, one USB2 port, two SPI interfaces and one I2C interface as well as reasonable tradeoff between size and complexity.

Communication between the IO-processor and the OMAP-L138 CPU is done using the second SPI bus of the OMAP. This bus is also used for the SPI flash, but not for the sensors so it does not interfer with communications between the OMAP and the sensor subsystem. In the transactions the IO processor acts as slave device but in addition to the bus there is a dedicated interrupt line from the SAM7A3 to the OMAP processor.

Sensor system

Together with the CPU system, the sensor system is central for the overall performance of the flight control system. To achieve best possible navigation and control the design combines a number of different measurement techniques.

The main sensor input comes from the inertial navigation unit (INU). The INU measures linear and angular acceleration around each of the three principal axes, and with proper processing this gives the instant orientation and movement of the vehicle. As noted earlier in section 3.5.1, different single-chip accelerometers and gyros within the low price bracket provide essentially identical performance, so selection of component is not critical from a performance standpoint. Instead the selection of accelerometers where made based on the principle of preferring digitally interfaced components and to prefer those with higher availability, which happens to be the ST LIS331DLH. As the accelerometer is small, cheap and don’t require many external support components it was decided to include two. This both increase reliability and manufacturing yield, as a card with one functioning accelerometer still functions satisfactory, and in the case of two functional accelerometers the raw data can be combined and thus reducing the noise variance.

Unfortunately at the time of designing the sensor system there did not exist any gyros with digital interface and acceptable price. The second best solution was opted for which is to use analog gyros, a combination of ST LPR430AL and ST LY330ALH, with a dedicated analog to digital converter in close proximity to the sensors. As recommended by the datasheets of the gyros an analog first-order bandpass filter with upper and lower frequencies of 0.03 Hz and 48 Hz is included, the design of these can be seen in section C.1.

In addition to the INU the sensor system incorporates a number of auxiliary sensors to augment the navigation and to function as calibration references. First of these is a three axis magnetometer, HMC5843 from Honeywell, which provide absolute value measurement of the earth-magnetic field. Together with knowledge of the absolute positon, which is provided by the GPS, this can be used to calibrate the absolute orientation of the vehicle. Secondly there is an absolute pressure sensor, VTI SCP1100-D10, to provide altitude and vertical speed measurements. Reliable vertical speed measurement is important both for stable forward flight and during automatic take-off and landing procedures. Both these sensors have digital serial interfaces.
The whole sensor subsystem has a separate linear power regulator and extensive per-chip decoupling to ensure that power supply noise and ripple does not limit the performance of the sensors.

**GPS system**

The GPS unit is just as the sensor system crucial to the navigation performance. In this project two factors have been deemed most important, having a short time to first fix and having a high update rate. A shot time to first fix is important to reduce the time before take-off and ensure a smooth user experience of the full system while a high update rate gives better navigation and improves the error correction calculations for the INU.

Of the GPS units listed in section 3.5.2 the u-Blox LEA-5 module best fulfill our requirements, even though it is not the cheapest available solution, as it has the lowest time to first fix and a high update rate.

**Other components**

In addition to the above described major subsystems the FCS board also incorporates a number of auxiliary systems.

The power supply is provided by a TPS 650250 circuit by Texas Instrument, which is an fully integrated power management IC (PMIC) with three DC/DC converters and three linear power regulators especially designed for processors such as the OMAP-L138. This way a single input voltage is transformed into all the required power supplies for all subsystems. Components and schematics supporting the PMIC has been designed based of the application notes provided by TI. A backup power device based on devices from Texas Instrument has also been included.

The flight controller board also incorporates an USB hub controller. The OMAP-L138 CPU expose both an USB2 on-the-go (OTG) port and a USB1.1 host port. To increase the effective number of ports available for controlling payload the controller board connects the USB1.1 port to a 4-port USB hub from Texas Instrument, TUSB2046B. This allows for an increased flexibility with very few extra components.

**PCB layout**

After completion of the design and schematic capture described in last section, a PCB was designed to hold the flight control system. Putting all components on a single board does make it more complex, but also simplifies manufacturing and assembly as there is just a single board to handle. It also increase reliability, board-to-board connections while reliable are always a possible point of failure.

During the layout process a few general guidelines were followed in order to ensure both a speedy design process and high quality layout:

- Prefer any guidelines regarding layout published by the manufacturer of each component.
4.1 Initial design

<table>
<thead>
<tr>
<th>GPS</th>
<th>Reset logic</th>
<th>DDR Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash memory</td>
<td>Sensors</td>
<td>OMAP L138</td>
</tr>
<tr>
<td>SAM7A3</td>
<td>Power supply</td>
<td>USB</td>
</tr>
</tbody>
</table>

Figure 4.2: Initial design - PCB floorplan

- Layout signals in order of importance, in general this means power decoupling, high speed signals, low speed signals and last power supply
- Use dedicated planes for all power and ground supplies
- As a last resort, follow sound general guidelines such as those published in [13, 26]

Layout was performed in a mostly bottom-up fashion where each subsystem was independently layed out, then combined according to the floorplan in figure 4.2 and finally interconnected. A bottom-up approach reduce the complexity of a big layout such as this which includes over 400 individual components and reduce the total time spent doing layout at the cost of a less dense result.

The initial and main focus was spent on layout of the most critical component, the DDR2 memory interface. Due to its very high speed the DDR2 memory interface requires very careful layout according to the rules in [13, 40], most importantly the different traces within each group were length-matched to within a maximum skew of 100 mils (2.54 mm). The DDR2 bus also requires each trace to be impedance matched to within a 20 Ohm range, which puts a constraint on the trace widths used. The tools built into the PCB CAD tool was used to automatically constrain the trace widths to an acceptable range.

Special care was also taken regarding layout of the analog part of the sensor subsystem. The analog outputs from the gyros and filters for the gyro signals have been put very close together separated from any digital signals on the same layer, and as there is a ground plane between the top signal layer and lower signal layers the interference from other layers should be minimal. In a similar way the routing of analog, especially the antenna, signals in the GPS system is separated from any interfering source in accordance with the datasheet published by the manufacturer.

The final layout of the PCB can be seen in figure 4.3 and the used layer stack in table C.2. The final board is 120x93x1.6 mm, with in total 8 copper layers. Smallest trace width utilized is 4 mil (0.1 mm) and smallest separation is 4 mil (0.1 mm). The card need both through-hole vias and blind vias from both sides, but no burried vias.
Figure 4.3: Initial design - PCB layout
4.1 Initial design

<table>
<thead>
<tr>
<th>Net class</th>
<th>Address</th>
<th>Data</th>
<th>Clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impedance (Ohm)</td>
<td>59.6 / 1.5</td>
<td>59.4 / 0.86</td>
<td>60.5 / 0.44</td>
</tr>
<tr>
<td></td>
<td>56.8 / 62.6</td>
<td>57.9 / 60.1</td>
<td>59.8 / 61.2</td>
</tr>
<tr>
<td>Length (mil)</td>
<td>2040 / 58</td>
<td>922 / 83</td>
<td>1901 / 37</td>
</tr>
<tr>
<td></td>
<td>1969 / 2186</td>
<td>817 / 1034</td>
<td>1868 / 1980</td>
</tr>
<tr>
<td>Overshoot (mV/V)</td>
<td>5.77 / 2.5</td>
<td>0.76 / 0.28</td>
<td>4.62 / 1.13</td>
</tr>
<tr>
<td></td>
<td>1.26 / 10.9</td>
<td>0.46 / 1.22</td>
<td>2.67 / 6.35</td>
</tr>
<tr>
<td>Data given as</td>
<td>Mean / Stdev</td>
<td>Min / Max</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: DDR2 bus simulation results

4.1.2 Preliminary evaluation

DDR2-bus simulations

As noted earlier the DDR2 memory bus is the most critical component in the system both for functionality and performance, and as such it was given special attention in the post-layout analysis. In addition to thorough visual inspection of the layout, the DDR2 bus was also subjected to the signal integrity and transmission line simulation included in the PCB design package. A summary of the simulated performance metrics can be found in table 4.2. The simulation also included a crosstalk simulation between the different lines in the bus, however due to a bug in the software we were not able to determine any numerical values for the crosstalk, just that it was all below the threshold of 20 mV/V for the signal characteristics of the DDR2 bus.

Component sourcing and manufacturing

The final step for obtaining a final flight control board after completing the design is the physical manufacturing. As soon as the final design draft was finalized and reviewed a request for quote (RFQ) was sent to a number of manufacturers, both pure circuit board manufacturers and contract manufacturing firms that does assembly. Due to the inclusion of high density BGA components the board needs to be machine assembled and as a result of this cannot be done in-house.

An anonymized summary of the responses received can be seen in C.3. Together with the approximate component cost of 3 000 SEK it sums up to an initial prototype cost close to 25 000 SEK, which is above the budget allotted for this work, especially if the first board contains critical defects and need a second design and manufacturing iteration. We believe that the relative complexity of the board together with a low potential for high volume manufacturing later increase the price above what would be the case for a similar design but with higher potential for volume production.

Simultaneous to requesting quotes for manufacturing component sourcing was performed, and in doing so a major problem was encountered: component shortage and very long lead-times. During spring 2010 there is a general compo-
ment supply problem[5], however in this case a more specific shortage of memory circuits proved problematic. On design the selected DRAM circuits were Micron MT47H128M8HQ-3, which existed in great supply although marked as "pre-production" series. During the period from component selection to component sourcing, a period of approximately two months, the HQ-3 component was replaced by the pin compatible CF-3 with identical specifications however the model replacement incurred a great reduction in availability. Initial availability for MT47H128M8CF-3 was very limited and during the component sourcing time the estimated lead times were in excess of 20 weeks and with Micron listing the component as having a normal lead time of 14 weeks this would incur unacceptably long delays in the prototyping process.

The two options to delaying production for at least 15 weeks would be to either use a smaller memory with pin compatible packages, such as the MT47H64M8, or to redesign the board to accept a different DDR2 memory circuit with better availability. The first option has similar problems as the issues in the initial design it is trying to solve, the 64MBit memory circuits also has a lead time of at least 10 weeks, and it only provides half the amount of memory. Redesigning utilizing a different incompatible memory circuit might also results in a long delay as the DDR2 bus is the most complicated part of the layout and would take considerable time to properly relayout. Redesigning with another memory module does not guarantee that the general shortage of components does not incur similar delays and lead times either, so it is not a real solution. High density DDR DRAM is among the components with biggest supply problems at the time of this work.
4.2 Revised design

Given the problems with the initial design that was identified and presented in section 4.1.2, especially issues related to component availability and manufacturing cost, the decision was made to try to redesign the system to solve the underlying problems and not only the consequences of them. The goals set for the redesign was to, without losing any major functionality of the initial design,

- Reduce the complexity of the design so that it can be assembled by hand in-house
- Reduce cost of manufacturing to allow for at least two iteration within the budget
- Try to protect against short-term component shortage problems by careful component selection

The revision is intended to be incremental and reuse both components and solutions/layout from the initial design as far as possible, while meeting the new goals.

The first step toward both a simplification of the layout and a reduction of cost, both cost of the initial manufacturing and for new spins with fixed as well as subsequent revisions, is modularising the design into several independent modules. In the initial design it was decided against that for reliability and handling reasons, however the increased focus on cost and manufacturability shifts the balance. To be able to still use a high performance ARM9 CPU we have chosen to utilize a so called System-On-Module (SOM) module, which is a pre-made module that contains CPU, RAM and flash memory, and some companion chip that does power management and physical interfaces for USB, ethernet etc.

4.2.1 Hardware design

As described above, and as can be seen in the system overview in fig 4.4 the revised design does not constitute a full change but rather a refinement of the earlier design, which for example means that all sensors and external IO systems are identical to the ones presented earlier. The same set of accelerometers, gyros and GPS as well as the same IO processor are included in the system.

SOM modules

The SOM module is the only major new hardware component and thus the majority of the design time was spent in investigating and selecting the appropriate module. At design time (spring 2010) there existed a wide range of SOM modules from a number of manufacturers on the market, so the evaluation was limited by a number of constraints and requirements

- ARM9 or similar processor. As identified earlier in section 3.2.2 and 4.1.1 this provides a good tradeoff between complexity and performance
Figure 4.4: Revised design - System block diagram
4.2 Revised design

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>LogicPD</th>
<th>LogicPD</th>
<th>CompuLab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module</td>
<td>TI OMAP-L138 SOM</td>
<td>TI OMAP35X Torpedo</td>
<td>CM-T3530</td>
</tr>
<tr>
<td>CPU</td>
<td>TI OMAP-L138</td>
<td>TI OMAP35</td>
<td>TI OMAP3503</td>
</tr>
<tr>
<td>Memory</td>
<td>128MB</td>
<td>128MB</td>
<td>128MB</td>
</tr>
<tr>
<td>Flash</td>
<td>8MB NOR</td>
<td>256MB NAND</td>
<td>128MB NAND</td>
</tr>
<tr>
<td>UART</td>
<td>3</td>
<td>3</td>
<td>3 (1 at RS232)</td>
</tr>
<tr>
<td>I2C</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>SPI</td>
<td>1 (4 CS)</td>
<td>3 (4+2+0CS)</td>
<td>3 (2+2+1CS)</td>
</tr>
<tr>
<td>USB</td>
<td>1+1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>SD-card</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Size</td>
<td>30x40 mm</td>
<td>15x27 mm</td>
<td>66x44 mm</td>
</tr>
<tr>
<td>Connector</td>
<td>3x100-pin Hirose</td>
<td>2x100-pin Hirose</td>
<td>2x140-pin</td>
</tr>
<tr>
<td>Cost</td>
<td>204 (USD)</td>
<td>210 (USD)</td>
<td>128 (USD)</td>
</tr>
<tr>
<td>Lead-time</td>
<td>1-4 weeks</td>
<td>1-4 weeks</td>
<td>10 weeks</td>
</tr>
</tbody>
</table>

Table 4.3: SOM Modules

- At least 128MByte RAM
- Two SD-card interfaces or one interface and 128MByte flash memory for system and program storage
- Multiple external serial busses including UART, I2C and SPI

In the initial design we excluded processors provided in too complex packages due to manufacturing constraints, however in a design with a SOM module this is not an issue and therefore such processors can be included. Instead, the important constraint is the type of connector used between the SOM module and the base board.

After a wide market overview the modules in table 4.3 were chosen for the final evaluation. Most modules can be bought in a great number of configurations with different peripheral interfaces such as WLAN, BlueTooth etc, but in many cases such customizations are only available if ordering in volumes of 1000 units; for smaller orders a default config with all extra options must be used. In this application there is no benefit to include such extra interfaces so priority is small size, low price, availability and local connectivity.

For this application it was decided that the Torpedo-SOM from LogicPD would best fit the requirements providing plenty of processing power in a very small package. Also it does not include any extra options and interfaces that are not required and has good availability. In addition to the specifications listen in table 4.3 the Torpedo SOM module also includes a power management solution with integrated battery handling, which allows the module to have a separate one-cell LiP0 battery in addition to the main power supply. This means an external backup battery can provide power to the SOM module during a limited time for example to reduce the startup time after switching the main batteries in the system.
One issue with the selected SOM and indeed all circuits based of the TI OMAP35x line of processors is that all external IO busses, with the exception of the SD/MMC interface, utilize 1.8 volt signaling while the rest of the system including both sensors and the IO processor operate at 3.3 volt. To handle the conversion, especially for two-way signals such as those used in I2C-bus a number of dedicated voltage level shifter circuits from Texas Instrument have been incorporated into the base board design.

**GPS and sensor module**

At the same time as the CPU was moved to a separate module board the GPS and sensor subsystems where also separated from the base board into two separate module boards. This allows for a simpler routing of both the modules and the base board, using fewer layers compared to the initial design while retaining good signal integrity. Just as for the SOM module, separating sensors and GPS into separate modules also reduce the manufacturing risk and allows for easier bug fixes, changes and future development of the system.

Both the GPS and the sensor module boards have their ground nets decoupled to contain high frequency noise. In addition both boards are provided with the raw 5V supply and use local voltage regulators and decoupling to ensure maximal performance in terms of noise localization and stable power supply. All external IO signals for the GPS and sensor module are 3.3V.

**Power supply**

To provide regulated power to all components the base board incorporate a power supply circuit. The SOM module, the GPS and sensor module as well as the CAN bus transceivers requires 5V external supply while the rest of the components require 3.3V. To get a compact solution providing both voltages while being efficient and supporting a wide range of input voltages the power supply circuit was based around a dual DC/DC-converter, specifically a TPS54386 from Texas Instrument. The support circuit surrounding the IC were designed using TIs web-based design tool with a target specification of 3A/3.3V and 3A/5V output with 8V to 26V input.

**PCB layout**

For most parts the PCB layout of the revised design follow the same guidelines as outlined for the initial design in section 4.1.1, however to ensure a reduced manufacturing cost and that the board can be hand assembled a few additional restrictions were imposed on the layout:

- Minimum feature size and separation 6 mil (0.15mm)
- No blind or buried vias
- All PCBs should be 4 layers (2 planes) and put on a single panel
To not waste time the layout of the GPS module, the sensor system and the IO processor was reused from the initial design, with some modifications as the number of layers was reduced. For the power supply circuit a reference layout is provided by the Texas Instrument design software and to ensure high performance and low RF emissions this layout was followed.

The floorplan for the base board is similar to the original design, with a few small modifications for practical reasons. The only real constraint applied to the floorplan is to place the GPS, that is sensitive to RF emissions, as far as possible from possibly emitting sources such as the switching power supply.

Figure 4.5 contains the finished layout of the base board, figure 4.6(a) and figure 4.6(b) contains the GPS module and sensor module. The prints might not be in right scale in this report, for reference the GPS board is 31x34 mm and the sensor board 34x39.5 mm.

4.2.2 Software design

This work focus on the design of the hardware platform, designing and implementing the software platform for the flight control system is in itself as time and resource consuming as the hardware platform and is not included in this work. However to verify the functionality and suitability of the hardware an initial software stack is needed, and the design of this will be described below. The system designed in this work contains two programmable elements that needs software for correct function, the SOM module and the SAM7A3 IO processor.
SOM module software

The processor on the SOM module is the most powerful processing element in the design, and it will contain a majority of the software once the system is fully implemented. Figure 4.7 contains an early overview of the total software stack but for this work only the components with a solid outline have been considered. These software modules are enough to boot the board and verify that all hardware components are functional and have proper communication.

The foundation for the software in the OMAP is the Linux kernel. The Linux kernel already contains most of the support code needed for running on the OMAP35x line of processors, and other ARM systems, but in order to configure it for differences between different setups the kernel needs to be augmented by a board support package (BSP) that defines exactly which hardware the board contains and on what interfaces it is connected.

In order to boot the Linux kernel we utilize modified versions of X-loader by
4.2 Revised design

Texas Instrument, as first stage boot loader, and then U-Boot as second stage boot loader. This setup allows for booting the system both from SD-card, for example in a bootstrap or firmware upgrade situation, and from the built in NAND-flash during normal usage. X-loader also handles setting up the processor configuration for the DDR memory interface, clock generation and gating and proper pin multiplexing config. Both X-loader and U-Boot have been configured to utilize the first UART interface for debugging and manual configuration during the boot process.

At this point the drivers in the system have been designed to expose a simple API in the form of files with human readable information. This is not the most efficient interface performance wise and will probably have to be redesigned in the future but during bootstrap it allows sensors and other periphelia to be interfaced using normal shell commands.

**SAM7A3 IO processor software**

The SAM7A3 IO processors main purpose is to handle timing critical IO and communication protocols and thus one of the most important aspects of the software design is ensuring low interrupt latency and high IO throughput. In order to facilitate this we propose to use a dedicated real-time operating system (RTOS) especially designed for critical applications in the final implementation however for initial bootstrap this is too complicated and takes too long time to setup, as most RTOSs requires processor and board specific customizations. For the initial iteration we instead utilize custom written code that will test all the functionality of the processor and all its communication interfaces.

Communication between the IO processor and the main SOM module physically utilize a dedicated SPI bus with the SOM module acting as master and the SAM7A3 acting slave. Software-wise the IO processor implements the communication by exposing a virtual register set, as this is the most common way for other devices to communicate over the SPI bus; and on the SOM side the each part of the IO processor maps to a separate driver with communication being handled by a so called multifunctional device (MFD) driver.
Chapter 5

Implementation

The main goal and contribution of this work is to devise a good design for the hardware part of the flight control system of an UAV and this design has been presented in the previous chapter. It should be noted that all aspects of a design cannot be verified or judged suitable for the intended purpose without an actual implementation that is built and tested.

In this chapter we will briefly present the first implementation of the revised design from section 4.2 including issues encountered and how they were, or could be, solved. We first describe the manufacturing process and after that the initial bootup, or bootstrapping, and basic functional verification of the system.

5.1 Manufacturing

At the same time the revised design was finalized, a request for quotation was sent out to a number of PCB manufacturers, providing them with a final draft version of the circuit board layout for quotation purposes. Compared to the initial design presented in section 4.1, the revised design is considerably simpler which resulted in both lower manufacturing cost and very little difference in cost between different manufacturers. In the end the boards where panelized onto a combined larger board and manufactured in east Asia.

One design goal with the revised design was to decrease the difficulty and cost of assembly and to utilize components with good availability, which we believe to have reached. All components where bought off-the-shelf from a number of big international suppliers and the final circuit boards were assembled by the author utilizing manual pick-and-place. Soldering and fixups were performed by hand by using soldering paste and a hot air soldering station. In total the assembly amounted to about three workdays of labour.

For a larger manufacturing batch we estimate a few hours of manual work in-house or by a manual assembly service. The design should also be suitable for assembly by pick and place machine followed by reflow soldering however the setup cost of such an operation is quite high which only makes it a viable alternative for manufacturing of larger batches.
The final assembled base board can be seen in figure 5.1 and the module boards in figure 5.2. All accessible soldered joins have been visually inspected by 15x visual light microscopy. Ideally hidden solder joints should be X-rayed, but during this work an X-ray machine was not accessible.

5.2 Bootstrap and verification

Following the assembly the boards were bootstrapped, that is powered up and loaded with initial software, and basic functionality tested. At this point verification mainly amounts to checking that the on-board signals are correct by visual inspection using oscilloscope and manual interaction with the system using the built in serial ports. During the verification the boards were connected as seen in figure 5.3. Apart from a test PC running Linux with terminal software two FTDI UM232R USB-to-UART converters with TTL levels and one Olimex ARM JTAG adapter were used to build the required test-rig.
5.2 Bootstrap and verification

(a) GPS module  (b) Sensor module

Figure 5.2: Assembled module boards

Figure 5.3: Verification setup
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input voltage</td>
<td>7 - 20V</td>
</tr>
<tr>
<td>3.3V rail voltage</td>
<td>3.31 ± 0.05V</td>
</tr>
<tr>
<td>5V rail voltage</td>
<td>4.98 ± 0.05V</td>
</tr>
<tr>
<td>Ripple unloaded</td>
<td>≤ 1mV-pp</td>
</tr>
<tr>
<td>Ripple 1A load</td>
<td>≤ 70mV-pp</td>
</tr>
</tbody>
</table>

Table 5.1: Power supply test results

5.2.1 Power supply

The power supply was the first sub-module to be assembled and tested. As the card is the first version of an untested design the power supply was tested before any other component was assembled to make sure it provided the correct rail voltages and acceptable ripple under load. The test results from the power supply verification can be seen in table 5.1 and they are within the limits provided by post-design simulation and thus also meets the target performance.

5.2.2 SOM module

Following the power supply module, the SOM module was the second component to be bootstrapped. Initial power up and booting of the SOM module proved to be more complicated than anticipated. In large part this is due to not having any access to the official development platform and all associated documentation, and a few errors and omissions in the documentation that was available.

First problem encountered during the bootup procedure was an inconsistency between documentation and reality related to the power supply for the SOM. The SOM module main power supply is by the 5V rail but in addition to this it has a connection for a single cell LiPO battery for backup and an coin-cell battery to keep the RTC powered. According to the documentation the 5V rail can power the entire module, however our testing showed that both the RTC battery and the one cell LiPO battery rails need to be connected to 3.3V for the module to boot normally.

The second problem and the single largest hardware problem encountered during the entire bootstrap of the designed system is related to the external communication paths from the SOM module. The OMAP35x processor on the SOM module contains three UART ports, numbered 1 to 3, but on the SOM module the UARTs are named A through C. During our testing we realized, which is also documented in a revised version of the SOM module documentation, that there is not a straight mapping between OMAP35x UART number and the SOM module names, but rather the mapping is 1-A, 2-C, 3-B. Furthermore it was discovered that UART 3/B was non-functional on the base board, probably due to either a soldering problem or a small dent in the contact between the base board and the SOM module. For the purpose of this work two UARTs are enough and the error in mapping could be resolved with a small cable patch on the board.
The missing UART does have one implication for the bootstrap process. The OMAP35x processors allow booting from UART3 and SD/MMC-card in addition to the built in NAND-flash and for the initial bootstrap when the flash is empty using the serial port provides a very convenient way that can be automated in a simple way. In this case however we were forced to load the initial software from an SD-card.

After patching the UART problem and resoldering two badly soldered level converters we were able to boot the software stack. X-loader and U-Boot were modified for the Torpedo-SOM based on but not identical to work done by Bihari [9]; then signed and loaded onto the SD card. The Linux kernel supports the OMAP35x processor unmodified so the only modification necessary were a build configuration file (.config in Linux nomenclature) and a board file that lists the SPI and I2C devices connected. The userspace, which includes tools, shell and configuration files, where built using Arago and OpenEmbedded which is a framework to speed up configuration and building of embedded Linux distributions.

Table 5.2 contains the verification results for different parts of the SOM module once the software stack and Linux was successfully booted. The SPI and I2C busses where verified functional by utilizing Linux-internal testing drivers, spidev and i2c-dev respectively, and visually inspecting the output waveforms after the level shifters. Due to time constraints not all parts of the SOM modules could be tested, priority was given to those that are essential for the operation of the full system.

### 5.2.3 SAM7A3 IO processor

The Atmel SAM7A3 MCU was verified in two steps. Initially a small test program was developed and loaded into the SAM7A3 over JTAG. The program just initialized the external main oscillator and sent some text on the external debug UART to verify basic booting capability and allow for measuring correct oscillator frequencies. As a second step more extensive test program was developed that in addition to the required setup also tested all wired GPIO pins and tested receiving data from the SOM module via the SPI bus interface. Communication from the
SAM7A3 to the SOM module over SPI and the interrupt signal was not tested, due to missing driver support on SOM side.

During verification no problems were discovered with the SAM7A3 subsystem with one exception; the external reset circuit driving the reset pin on the circuit is inadequate and had to be disconnected from the circuit for proper operation. In this first implementation the power-on-reset (POR) internal in the MCU is sufficient, for next iteration a redesign of the reset circuit is required.

5.2.4 System modules

Once the programmable elements and the associated external busses where verified to function as specified, the external modules, that is the GPS and sensor modules, where attached to the base board and the basic functionality of these modules verified. As with the earlier parts verification was made piece wise by first electrically verifying correct operation and performance followed by digital “in-system” verification.

GPS module

The GPS module was verified by using an external antenna, GlobalSat AT-65 which is a patch antenna combined with a 27dB LNA. To simulate the environment that the GPS will be in once the system is active the antenna was placed outside a window during the testing.

The GPS module was verified to function correctly by manual inspection of the generated NMEA stream. It was checked that the GPS managed to acquire a fix of high quality (low HDOP and PDOP) and that the reported time was correct, however no external verification of the reported position has been performed as the position of the lab is not known with sufficient accuracy.

Sensor module

Given that the sensor board is physically the most dense board and contains most of the fine pitch components in the design, including several TQFN/LCC package components, it was expected that it would contain most soldering issues, which also was the case. A number of components had to be resoldered to remove shorts circuits and unconnected pads.

Initial analog testing of the analog circuits, which includes the gyros, the filters connected on the output of the gyros and the analog to digital converter, uncovered one design problem. The design incorporates high pass filters with very low cutoff frequency to compensate for drift and bias in the output and design these filters are based on the application note published by ST. The problem is not these filters on their own, they work as expected, but during design we failed to take into account the interaction between the very high output impedance of the filter and the relatively low input impedance of the AD converter. The input stage of the AD converter is a sample-and-hold circuit which basically works by charging a capacitor and then measuring the voltage over the charged capacitor. This results in a hard to diagnose problem when used to sample the high impedance high pass
5.3 Summary

In this chapter we have reported the experiences gained and problems encountered during the implementation phase for the designed system. The first thing to note is that we fulfilled the goals set for the system redesign related to manufacturing cost and complexity. The four layer panel was manufactured by one out of many possible cheap manufacturers at a competitive price, and the low complexity in assembly is proven by the fact that the whole system could be hand assembled by the author and required just a few fixes post assembly.

In the second part we have showed that the design is working so far as it is performing the functionality it was designed for, with the exception of the non-working accelerometers. This does however not mean that system is fully fit for flight. Apart from the few hardware fixes needed on the board to make all sub-modules work a flight control system needs a big amount of software to actually fly and as this has not been developed the design cannot yet be flight tested.

<table>
<thead>
<tr>
<th>Component</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pitch/roll gyro</td>
<td>Fully working</td>
</tr>
<tr>
<td>Yaw gyro</td>
<td>Fully working</td>
</tr>
<tr>
<td>AD converter</td>
<td>Fully working</td>
</tr>
<tr>
<td>Linear accelerometers</td>
<td>Non-working</td>
</tr>
<tr>
<td>Magnetometer HMC5843</td>
<td>Com. working</td>
</tr>
<tr>
<td>Pressure meter SCP1000</td>
<td>Com and pressure working</td>
</tr>
</tbody>
</table>

Table 5.3: Sensor module verification

filter. The charging period is very short and during this short period the high pass filter cannot deliver enough current to properly charge the input capacitance, and thus the measured voltage will be too low. The solution for this first version of the board was to remove the high pass filter, which is optional anyhow, and directly sample the signal after the low pass filter. In a later updated design it can be solved either by permanently removing the high pass filters, or by adding a set of buffers between the filter output and the AD input to boost the driving capability.

Following verification of the analog parts the digital circuits and communication were tested utilizing alpha-versions of the Linux drivers running on the SOM module. As the software is in initial stages not all parts of all sensors could be tested, but rather focus was to make sure communication between the SOM and sensors were working and the sensors reported some status. Verification results for the sensor module can be seen in table 5.3. As can be noted the linear accelerometers were found non-working, and after careful debugging it does seem as if both circuits where damaged in the assembly process but due to the delivery time for new circuits it was not possible to get replacement components and integrate these during the duration of this work.
Chapter 6

Summary

In this chapter we will summarize the work presented in the earlier chapters of this report and highlight some of the main conclusions that can be made based upon our results.

We will also present a number of different areas where we think the hardware platform can be improved or more thoroughly investigated at a later time, but the scope of this work did not allow for us to include it.

6.1 Summary

In this work we have presented the development of the hardware platform for the flight control system in an UAV. In the first part of the work we present an overview of other systems both commercial and academic, focusing on the hardware platform they are based on. During this review it was found that while there are many published articles concerning the software and algorithms used in UAVs but very few publications deals with the hardware and even less is publically available regarding the commercial systems.

Based on this combined with general embedded systems knowledge three vastly different designs were proposed and evaluated theoretically in a pretty general way. The three proposed designs were based on three different basic architectures

- A network of independent microcontrollers
- One central CPU with minimal glue components
- FPGA with soft-core and dedicated accelerators for specific functionality

Following an evaluation of the alternatives, the second option was chosen to be most suitable and a full design was developed based on this architecture. In the initial iteration the system incorporated:

- TI OMAP-L138 Application processor with an ARM9 core and a floating point DSP
Summary

- 2x128Mbyte DDR2 RAM, 8MB boot flash and micro-SD for firmware
- INU including two 3d accelerometers, 3 axis gyros and magnetometer
- Top of the line GPS receiver
- IO system with an Atmel AT91SAM7A3 microcontroller handling timing critical IO protocols

In the final stages of the system design while preparing it for manufacturing a common problem for embedded systems showed in this project as well; there is often a difference between the technical side of the system and the economic reality. In this case the designed system was considered to be technically well suited for the application however it proved to be more expensive to manufacture than anticipated and together with a big shortage of some components it was decided the economic risk was too great for the overall project at this point.

The solution was to redesign the system to build on a SOM module which meant that the most complex part of the system could be bought pre-assembled and connected via standard connectors to the rest of the system. Apart from the CPU, which was upgraded to a TI OMAP 3530 processor, and a smaller RAM at only 128MByte the system contains the same components as presented above. At the same time as the CPU was moved to a separate board the rest of the system was redesigned by separating it into a number of smaller modules attached to a larger base board. This reduces the prototype manufacturing risk and cost for additional iterations.

As a last step of this work the revised system was built and verified to function according to the intended specification.

6.2 Future work

Based on the work presented in this report we believe that there are a number of areas which are interesting for further investigation and where more needs to be done before the system is optimal

- The problems identified during manufacturing and bootstrap in chapter 5 should be permanently solved.
- System software should be improved for better performance and stability.
- The flight control software, sensor fusion and control loops needs to be implemented.
- Further investigate how an external security system can increase the reliability of the system.
Bibliography


[22] Ge Hong. Baopuzi. 318 AD.


Appendix A

Requirement specification

This appendix contains the requirement specification setup before the start of the project and outlines the external requirements imposed on the flight control system designed in this work.

In this section, requirements are specified on two levels:

**Must** The requirement must be met if at all technically possible

**Should** The requirement is desirable but not highest priority

A.1 Requirements

A.1.1 Functional requirements

This section details the functional system-level requirements that exists for the FCS, that is what other parts of the system "expects" the FCS to handle

**Must** The FCS receives control commands from the remote units such as the RC radio and serial link and translates into control commands to the motor controllers

**Must** The FCS must be able to assist the pilot by providing augmented control and stabilization

**Should** The FCS should provide capabilities for autonomous navigation by GPS and/or IMU to predetermined locations

**Should** The FCS should be able to combine information from multiple sources to form an unified computation of position and orientation

**Must** The FCS must accept standardized roll, nick, yaw and throttle comamnds from a remote controller, either RC radio or serial link

**Must** The FCS must be able to control at least four (4) external motor controllers via one of its communication interfaces
**Should** The FCS should be prepared for controlling up to twelve (12) external motor controllers

**Should** The FCS should be prepared for translating control commands into another format such as PPM servo control, instead of or in addition to sending them to motor controllers

### A.1.2 General requirements

This section details general requirements on the overall FCS system.

**Must** The FCS is made up of one or more central processing units (CPUs) and auxiliary logic

**Must** The CPUs must be programmable in a high level language

**Must** The CPUs and other programmable elements must contain the software to fulfill all functional requirements

**Should** The CPUs and other programmable elements should provide sufficient capacity for continued development, within reason

**Should** The CPUs should be able to run a freely or commercially available operating system handling low-level interaction and operations

**Must** The FCS must have a physical construction suitable for operating in an outdoor environment under normal weather conditions without requiring extensive or extreme protection

**Must** The FCS system must be able to work on a single 9 volt DC power supply. Any other required voltage must be internally derived from this single supply

**Must** The FCS must be power efficient to the degree that the energy consumption of the FCS is small comparable to the motor driving during time of flight

**Must** Any programmable component of the FCS system must be programmable and debuggable via a single JTAG [3] chain

**Should** The JTAG chain should be available to tie together the FCS with other modules in the system to a single JTAG domain

**Should** The FCS should be constructed in a way so that manufacturing does not become prohibitively complicated or expensive

**Should** The FCS should if possible use standardized or already existing components and construction blocks

**Should** All state and command information should be available to all software modules within the FCS

---

1 Currently rated for max 2.6kW
A.1 Requirements

A.1.3 Reliability

**Should** The FCS system should have backup power supply to handle loss of main power supply for no more than thirty (30) seconds without noticeable interruption of service once power is restored

**Must** The FCS system must provide monitoring of all component and detect any lockups or failures. Failures should be handled as gracefully as possible, priorities are safety, protect payload, protect vehicle

**Must** The FCS electronics and components must be constructed so that they are able to withstand normal flight operations as well as frequently occurring incidents including hard landings and smaller crashes without permanent damage

A.1.4 Components

This section outlines specific component requirements for sub-components in the FCS system

**Must** The FCS must contain a component for providing audio feedback and error reporting such as a piezo-summer

**Should** The FCS should be able to provide audio feedback using a wave decoder and speaker

**Must** The FCS must contain enough sensors to adequately fulfill the functional requirements

**Should** The FCS should contain sensors to measure acceleration in six axes and altitude

**Should** The FCS should contain sensors to measure magnetic field in three axes

**Should** The FCS should contain a GPS receiver

**Should** The FCS should contain a secondary altitude sensor for altitude detection close to ground level

A.1.5 Communication

The FCS does not act in isolation but rather is part of a larger system and therefore it needs communication interfaces to the rest of the system and also the outside of the system. This section details requirements for communication interfaces and protocols which the system must be able to communicate over.

**Must** The FCS must be able to communicate over at least one I2C [1] bus

**Should** The FCS should be able to communicate over at least one CAN [4] bus

**Must** The FCS must be able to communicate over at least one USART standard serial bus
**Should** The FCS should be able to communicate over two or more independent USART standard serial busses

**Should** One or more of the USART busses should be connected to a ground station by means of a radio modem link

**Must** The FCS must be able to receive direct control over a standard RC radio in pulse-position-modulated (PPM) format where up to 12 channels are sent on a single input

**Should** The FCS should be able to transparently combine control over the radio modem link and from the RC radio

**Must** The FCS must be able to output auxiliary servo control signals in pulse-position-modulated (PPM) format for controlling at least three (3) servos in the payload

**Must** The FCS must be able to communicate with the payload via I2C [1] as well as USB [2] busses
Appendix B

FPGA Estimation

To get an rough estimate of the size of an FPGA needed for design alternative 3, Xilinx Platform Studio was used to generate a design with a MicroBlaze softcore and auxiliaries. Table B.1 shows the IP-cores included, however it does not include any interfaces for AD converters, USB bus or CAN bus as these were not available for this quick estimate.

The estimation design was implemented on one Spartan3AN device and one Virtex4 device from Xilinx. The resulting resource usage is shown in table B.2 from which we can see that a fairly large FPGA will be needed. Synthesis was done targeting 100MHz with high optimization effort. For the Virtex4 design power analysis was performed using Xilinx XPower Analyzer which for the design resulted in an estimated power consumption of 600mW quiescent power and 600mW dynamic power.
<table>
<thead>
<tr>
<th>IP</th>
<th>Version</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MicroBlaze</td>
<td>7.10a</td>
<td></td>
</tr>
<tr>
<td>lmb_bram_if_cntrl</td>
<td>2.10a</td>
<td>2 copies</td>
</tr>
<tr>
<td>mpmc</td>
<td>4.00a</td>
<td>Configured for DDR</td>
</tr>
<tr>
<td>bram_block</td>
<td>1.00a</td>
<td></td>
</tr>
<tr>
<td>mdm</td>
<td>1.00b</td>
<td>Debug unit</td>
</tr>
<tr>
<td>xps_gpio</td>
<td>1.00a</td>
<td>32 pin GPIO</td>
</tr>
<tr>
<td>xps_iic</td>
<td>2.00a</td>
<td>2 copies</td>
</tr>
<tr>
<td>xps_spi</td>
<td>1.00a</td>
<td></td>
</tr>
<tr>
<td>xps_sysace</td>
<td>1.00a</td>
<td>CompactFlash IF</td>
</tr>
<tr>
<td>xps_timer</td>
<td>1.00a</td>
<td>2 32-bit timers</td>
</tr>
<tr>
<td>xps_uartlite</td>
<td>1.00a</td>
<td>2 copies</td>
</tr>
</tbody>
</table>

Table B.1: FPGA Estimation - IP cores used for estimation

<table>
<thead>
<tr>
<th>Resource</th>
<th>Spartan3AN (% of xc3s1400an)</th>
<th>Virtex4 (% of xc4vlx100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slices</td>
<td>5436 (48)</td>
<td>5672 (30)</td>
</tr>
<tr>
<td>BRAM</td>
<td>19 (59)</td>
<td>19 (19)</td>
</tr>
<tr>
<td>External IOB</td>
<td>199 (39)</td>
<td>197+2 (43+2)</td>
</tr>
</tbody>
</table>

Table B.2: FPGA Estimation - Estimated implementation size
Appendix C

Initial design details

This appendix contains details of and overview circuit schematics for the initial design presented in section 4.1. It only contains the most vital ones and does not fully document the system.

C.1 Analog filters

The analog gyros contain a discrete component bandpass filter on the 1X outputs to remove bias as well as high frequency noise. The filters have been realized as a first-order highpass RC filter followed by a first-order lowpass RC filter.

For both highpass and lowpass RC filters of first order the cutoff frequency is $f_c = \frac{1}{2\pi RC}$. The component sizes were selected based of information in the datasheet of the gyros which results in the filter configuration in table C.1.

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highpass R</td>
<td>1 MOhm</td>
</tr>
<tr>
<td>Highpass C</td>
<td>4.7 μF</td>
</tr>
<tr>
<td>Highpass freq</td>
<td>0.03 Hz</td>
</tr>
<tr>
<td>Lowpass R</td>
<td>33. kOhm</td>
</tr>
<tr>
<td>Lowpass C</td>
<td>100 nF</td>
</tr>
<tr>
<td>Lowpass freq</td>
<td>48 Hz</td>
</tr>
</tbody>
</table>

Table C.1: Initial design - Gyro filters
C.2 PCB Layer stackup

<table>
<thead>
<tr>
<th>Layer</th>
<th>Material</th>
<th>Thickness (mil)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top</td>
<td>Copper</td>
<td>1.4</td>
</tr>
<tr>
<td>-</td>
<td>Core</td>
<td>4</td>
</tr>
<tr>
<td>Ground 1</td>
<td>Copper</td>
<td>0.7</td>
</tr>
<tr>
<td>-</td>
<td>Prepreg</td>
<td>4</td>
</tr>
<tr>
<td>Inner 1</td>
<td>Copper</td>
<td>0.7</td>
</tr>
<tr>
<td>-</td>
<td>Core</td>
<td>15</td>
</tr>
<tr>
<td>Power 2</td>
<td>Copper</td>
<td>1.4</td>
</tr>
<tr>
<td>-</td>
<td>Prepreg</td>
<td>8</td>
</tr>
<tr>
<td>Power</td>
<td>Copper</td>
<td>1.4</td>
</tr>
<tr>
<td>-</td>
<td>Core</td>
<td>15</td>
</tr>
<tr>
<td>Inner 2</td>
<td>Copper</td>
<td>0.7</td>
</tr>
<tr>
<td>-</td>
<td>Prepreg</td>
<td>4</td>
</tr>
<tr>
<td>Ground 2</td>
<td>Copper</td>
<td>0.7</td>
</tr>
<tr>
<td>-</td>
<td>Core</td>
<td>4</td>
</tr>
<tr>
<td>Bottom</td>
<td>Copper</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Table C.2: Initial design - PCB layer stack

C.3 RFQ Responses

In table C.3 and C.4 we present the quotes received from a number of manufacturers for manufacturing of PCB and assembly respectively.

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Price</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company A</td>
<td>5900 SEK (1 piece), 6750 SEK (5 pieces)</td>
<td></td>
</tr>
<tr>
<td>Company B</td>
<td>9500 SEK (1 piece), 11500 SEK (5 pieces)</td>
<td></td>
</tr>
<tr>
<td>Company C</td>
<td>Max 4 layers</td>
<td></td>
</tr>
</tbody>
</table>

Table C.3: Initial design - PCB manufacturing quote
### Table C.4: Initial design - Assembly quote

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Price</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company A</td>
<td>16100 SEK (1 piece), 26400 SEK (5 pieces)</td>
<td></td>
</tr>
<tr>
<td>Company B</td>
<td>3000 SEK + 850SEK/hour machine time</td>
<td></td>
</tr>
<tr>
<td>Company C</td>
<td>Won’t do, no X-ray</td>
<td></td>
</tr>
<tr>
<td>Company D</td>
<td>Won’t do, too low volume</td>
<td></td>
</tr>
<tr>
<td>Company E</td>
<td>28800 SEK (1 piece), 62500 SEK (5 pieces)</td>
<td>Price includes parts except GPS</td>
</tr>
</tbody>
</table>
Appendix D

Revised design details

This appendix contains details of and overview circuit schematics for the revised design presented in section 4.2. It only contains the most vital ones and does not fully document the system. In some cases the design has remained unchanged from the initial design and thus is not reported again.

D.1 PCB Layer stackup

<table>
<thead>
<tr>
<th>Layer</th>
<th>Material</th>
<th>Thickness (mil)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top</td>
<td>Copper</td>
<td>1.4</td>
</tr>
<tr>
<td>-</td>
<td>Core</td>
<td>12.6</td>
</tr>
<tr>
<td>Ground</td>
<td>Copper</td>
<td>1.4</td>
</tr>
<tr>
<td>-</td>
<td>Prepreg</td>
<td>12.6</td>
</tr>
<tr>
<td>Power</td>
<td>Copper</td>
<td>1.4</td>
</tr>
<tr>
<td>-</td>
<td>Core</td>
<td>12.6</td>
</tr>
<tr>
<td>Bottom</td>
<td>Copper</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Table D.1: Revised design - PCB layer stack
Upphovsrätt

Detta dokument hålls tillgängligt på Internet — eller dess framtidiga ersättare — under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för icke-kommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera åktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnt som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

Copyright

The publishers will keep this document online on the Internet — or its possible replacement — for a period of 25 years from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for his/her own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/

© Mårten Svanfeldt