

# ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG INSTITUT FÜR INFORMATIK

Arbeitsgruppe Autonome Intelligente Systeme

Prof. Dr. Wolfram Burgard

Department of Systems Engineering and Automatics

Dr. Ing. Rafael Sanz Domínguez



The Autonomous Blimp Project

## Diplomarbeit

Pablo González Alvarez

December 2005 - May 2006



## Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Außerdem erkläre ich, dass die Masterarbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

(Pablo González Alvarez)  
Freiburg, den May 24, 2006



## **Acknowledgment**

I would like to thank all the people who have helped me to write this thesis. Especially Prof. Dr. Wolfram Burgard and Dr. Ing. Rafael Sanz Domínguez who gave me the opportunity to work in this research group.

I would like to thank in not particular order: Axel, Oscar, Rudolph, Daniel, Inés, Christian and many others, because they helped me when I needed it.

And finally, and most important, I would like to thank my parents, my girlfriend, my brother, my goddaughter and family, because without their love and support I could not get here.

## **Agradecimientos**

Me gustaría dar las gracias a la gente que me ha ayudado a escribir esta tesis. Especialmente al Prof. Dr. Wolfram Burgard y al Dr. Ing. Rafael Sanz Domínguez quienes me dieron la oportunidad de trabajar en este grupo de investigación.

Me gustaría dar las gracias sin previo orden a: Axel, Oscar, Rudolph, Daniel, Iné, Christian y muchos otros, porque ellos me ayudaron cuando lo necesité.

Y por último, y más importante, me gustaría dar las gracias a mis padres, a mi novia, a mi hermano, a mi ahijada y a mi familia, porque sin su cariño y apoyo no podría haber llegado hasta aquí.



# Contents

<b>1. Introduction</b>	<b>1</b>
1.1. Objective . . . . .	2
1.2. Related Work . . . . .	3
<b>2. Blimp Selection</b>	<b>5</b>
2.1. Selection Criteria . . . . .	5
2.2. The Selected Blimp . . . . .	6
<b>3. Electronic Circuits</b>	<b>9</b>
3.1. Gondola Board . . . . .	10
3.1.1. General Diagram . . . . .	10
3.1.2. Power Source . . . . .	10
3.1.3. Microcontroller . . . . .	11
3.1.4. Motor Drivers . . . . .	13
3.1.5. Wireless . . . . .	13
3.1.6. Battery . . . . .	14
3.2. Personal Computer Board . . . . .	15
3.2.1. General Diagram . . . . .	15
3.2.2. Power Source . . . . .	16
3.2.3. Serial Interface . . . . .	16
3.2.4. Wireless Communication Module . . . . .	17
<b>4. Firmware</b>	<b>19</b>
4.1. Algorithm . . . . .	19
<b>5. Driver</b>	<b>23</b>
5.1. The Interface . . . . .	23
5.2. Serial Port Functions . . . . .	24
5.3. Get Measurements . . . . .	24
5.4. Movement Functions . . . . .	25

<b>6. Altitude Regulator</b>	<b>27</b>
6.1. Ultrasonic Sensor . . . . .	27
6.2. Kalman Filter . . . . .	30
6.3. PID Regulator . . . . .	31
6.4. Fuzzy Regulator . . . . .	33
<b>7. Collision Avoidance</b>	<b>37</b>
7.1. Ultrasonic Sensor . . . . .	37
7.2. Collision Avoidance Function . . . . .	38
<b>8. Experiments and Results</b>	<b>41</b>
8.1. Height Control . . . . .	41
8.1.1. PID Regulator . . . . .	41
8.1.2. Fuzzy Controller . . . . .	42
8.1.3. Comparison PID and Fuzzy . . . . .	43
8.2. Collision Avoidance . . . . .	45
<b>9. Future Work</b>	<b>47</b>
<b>10. Conclusions</b>	<b>49</b>
<b>A. Components</b>	<b>51</b>
<b>B. Electronic Board Plans</b>	<b>55</b>
<b>Bibliography</b>	<b>75</b>

# 1. Introduction

Robotics is the science that studies the design and construction of machines, which are able to do human tasks. The word “Robot” comes from the theater function R.U.R. (Rossum’s Universal Robots) written by Karel Capek (1917) [22] . He uses ‘Robota’ which means servants or forced workers . This word was translated into English as Robot.

Outside the literature, nowadays there is a lot of research in robotics and robots are more and more used in different tasks. There are for instance industrial robots, medical robots or mobile robots as we see in Figure 1.1.

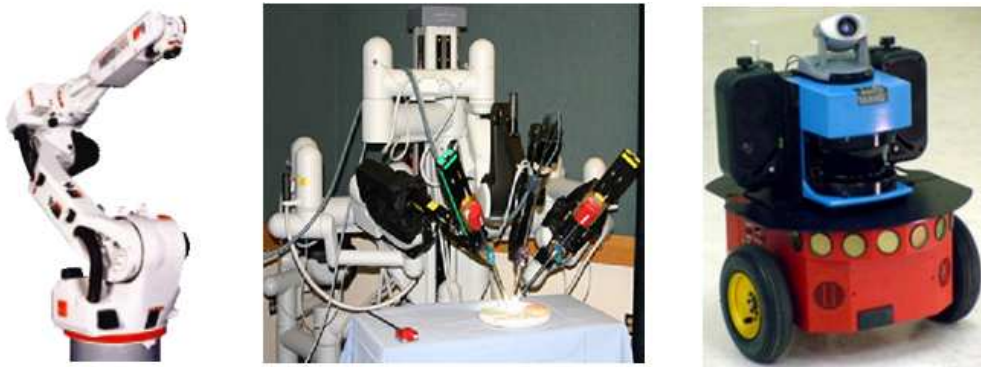


Figure 1.1: Examples of an industrial, a medical and a mobile robot

The field of research of mobile robotics studies how one can create autonomous mobile systems, how they can interact with the environment, and finally how they can make their own decisions.

All those things are possible when the robot has sensors to get information about the environment. In that way it can make decisions based on the measurements. These sensors can measure several characteristics and they can use many physical principles. Examples of sensors are: laser, ultrasonic sensor, thermal sensor.

Normally mobile robotics uses machines like cars or other platforms with similar characteristics. The idea is to obtain autonomous movement. Usually mobile robotics uses wheeled vehicles, because they are easy to control and have easy equations as movement equations. However, they have some limitations because they are fixed to the ground. In our project we want to use a blimp

as platform. This platform is more difficult to control than wheeled vehicles because it is not in contact with any fixed reference. But this characteristic of a blimp gives us lots of advantages in some applications.

If we think of a natural disaster, such as earthquakes, wheeled robots would have lots of difficulties to move in this environment. For lots of reasons such as ground situation, their weight, hidden victims. In this situation aerial robots can analyze the area more fast than wheeled robots. Aerial robots are not dangerous for the victims because they do not have contact with the victims. Another idea is the 3D planes generation of buildings, a wheel robot can not take the complete plans. But a blimp can fly over the building and obtain a complete map.

In this thesis we want to develop an autonomous blimp to implement those and furthermore typical applications of wheeled robots. We think that the blimp is a good platform to use in this applications.

So we want build an autonomous indoor blimp. This blimp will be controlled using a ground computer, therefore we need to develop all hardware and software necessary. When the blimp will be complete, we will develop two basic functions to control the altitude of the blimp and to control the collision avoidance.

### 1.1. Objective

In this work we explain the construction process and program to build an autonomous blimp. The thesis is structured as follows:

- Chapter 2: We describe the reasons to select the blimp. This is a very important point because there are a lot of blimp models, but each of them has different characteristics and is good for specific application.
- Chapter 3: We explain the electronic circuits which control the blimp. This is the interface between the blimp and the computer. We need little weight and long range of signal
- Chapter 4: We describe the firmware for the blimp. In the blimp we have motors, sensors and electronic circuits and we need a program to operate with them and the computer.
- Chapter 5: When the blimp is running, we need to control it with the computer. We are using Linux, therefore we need a driver to control the PC board with Linux. In this chapter we describe the interface to control the blimp with the PC.
- Chapter 6: We have the blimp and we can control it with the PC. At that point we develop some functions to simplify the control. In this chapter we explain the height control function. With this function the blimp controls its height with our reference.

- Chapter 7 : We describe a second function. This function avoids the frontal collision between the blimp and obstacle. We send the distance actuation to the blimp and when an obstacle is found, this function controls the horizontal movement of the blimp .
- Chapter 8: In this chapter the results obtained from the experiments are presented. Furthermore we can see the comportment graphics for the last functions in a real situation.

The final chapters has the conclusions and visions for future work.

## 1.2. Related Work

Recently, several authors have developed robotic systems based on blimps and studied appropriate control paradigms. Kantor and colleagues [23] discuss the use of solar energy as a renewable source of power for airships using an outdoor blimp. Elfes and colleagues [27], based on typical mission requirements, present arguments that favor airships over airplanes and helicopters as the ideal platforms for such missions. Emmanuel and colleagues [31] focus on flight control and terrain mapping issues for the cooperation between ground and aerial robots.

Other authors use blimps in indoor environments. Motoyama and colleagues [29] designed an autonomously controlled indoor blimp and an action-value function for motion planning based on the potential field method [28]. They evaluated its effectiveness in a simulated environment. Geoffrey and colleagues [19] used a commercial indoor blimp. They concluded that the vertical motor was severely underpowered in the teleoperated environment. They used a commercial wireless board to send the measurements of the sensors to the ground computer and a separate board to control the motors. Hydrogen and Helium were used to increase the payload capacity. In this thesis, we use the same type of blimp, with a different vertical motor. We use a more powerful motor and a big propeller to control the altitude. We develop a specific board to control the motors and sensors, which is significantly lighter than their system. Thereby, we have more payload capacity to use for additional equipment. We only use Helium, because Hydrogen is flammable and dangerous in indoor environments.

An important problem for blimps as well as for mobile robots in general is obstacle detection and collision avoidance. Green and colleagues [26] used an infrared sensor to detect obstacles. When the collision avoidance system detected an obstacle, the blimp turned 180 degrees to avoid the collision. Their collision avoidance system did not control the inertia of the blimp. In this thesis, we use an ultrasonic sensor to measure the distance to potential obstacles. We use this distance measurement in a Fuzzy controller to avoid the obstacle.

An important control problem is the automatic adjustment of the altitude and the horizontal movement. If the blimp is in a specific altitude, we can move it in a horizontal plane. Kadota and colleagues [34] used PID controllers to regulate the altitude and the horizontal movement of the blimp. They argue that the trajectory of the blimp was unstable in the vertical direction. In

## *1. Introduction*

---

this thesis, we use PID and Fuzzy controllers as altitude regulators. We evaluate the performance of the developed control systems using those regulators in two different environments.

## 2. Blimp Selection

In this chapter we will describe the kind of blimp that we will use for this project. There are two different types of blimps. We can buy an outdoor or an indoor one. Outdoor blimps are bigger than indoor ones. They can fly at outdoor environments and have a big capacity of payload. Opposite to this solution are the indoor blimps. An indoor blimp is small and can fly in a corridor or room. We want use the blimp in indoor environments, so we select this type.

There are lots of types of indoor blimps. We can select between the smallest blimp, 30 cm of length, and the biggest blimp, 2 meters length. A big blimp has more payload capacity, but we have to check the balloon weight, because balloons are not built with the same materials.

The goal of the project is to build an autonomous blimp. This blimp will be controlled with a ground computer, using a wireless communication. Then we have to put electronic boards (to control the blimp), sensors (to get the measurements), batteries, motors (to move the blimp) and a lot of components more inside the blimp. We have to check this things to take a decision about what blimp select. This is the goal for this chapter.

### 2.1. Selection Criteria

In this section we will describe the important points to select a blimp. Our autonomous blimp is for indoor environments, so we have to check characteristics as size, maximum payload capacity and motors distribution.

Before we start to describe the selection criteria, we have to see the actual situation of the market. In the market we find a lot of types of blimps [11] [2] . Two important items to make the classification are payload capacity and distribution of gondola motors. The first point is the most important because it tells us how many grams we can payload into the balloon. The size has influence on this payload capacity, because if the balloon is big we have more capacity than if the balloon is small. But this is not all, because we need to know the weight of the balloon too. So we need a light and big balloon.

The other point to study is the gondola. We can select a balloon and a gondola independently. Gondola has the motors and space for the hardware. The motors distribution is important to make more easy the control of the blimp. So we need a good distribution to get a good control. We have to check these points to select the blimp:

## 2. Blimp Selection

---

- Size: we have to check this point, because we have to achieve a balance between this one and the other points. The blimp is for indoor use, so it has to be small to fly for corridors and rooms with obstacles. Another question is the capacity, if we use a bigger blimp, we will have more capacity of payload.
- Maximum payload capacity: this point is very important, because it determines how many weight we can put into the gondola. Helium is less dense than the air. The density of helium is 0.126 kg/l. So if we put helium inside the balloon, the balloon flies. Another option could be use hydrogen, because it allows more capacity, but it is very dangerous, because hydrogen is inflammable. This is the reason because we use helium.
- Mechanical control system: we can select between several kinds of gondolas. There are two main groups:
  - Two motors and one servo: This gondola has two motors for propulsion. The servo can spin the motors, so if we turn the motors we can regulate the vertical propulsion and the horizontal movement. The problem of this distribution is that we can not separate the height control from horizontal movement control.
  - Three motors: This gondola has a vertical motor to regulate the altitude of the blimp and other two motors to rotate the blimp. With this solution we can control the altitude of the blimp independently and the altitude control program is more easy.

### 2.2. The Selected Blimp

In this section we will describe the selected blimp and its characteristics. We select only two parts for the blimp, the balloon and the gondola. The other components will be explained in next chapters.

We start with the balloon selection. We searched in several shops to compare balloon sizes. After this process we conclude that to obtain a 144 grams of payload capacity, we need a 52” balloon made of a light material. We think that this capacity for this size is good. This capacity is the balloon capacity, without gondola and motors. So we selected the plantraco blimp, with a 52” balloon as we see in Figure 2.2 [11] [3]. With this size we can use it to fly in a corridor with people and enter in rooms through a door. Definitely, this balloon satisfied our needs.

This blimp has a tri-turbofan gondola. With this distribution we have a vertical motor to regulate the altitude and two motors to control the speed and the rotation of the blimp. This gondola [Figure 2.1] uses tree light dc motors, so it is very appropriate for our application. The vertical motor has few power, but we change it for a DC motor with a Futaba 3003 servo motor. We put a big propeller on this motor and now we have the necessary power to control the altitude. With this changes the gondola weighs only 70 grams. We have 74 grams free for hardware.



Figure 2.1: Tri-turbofan gondola selected. It has right, left and vertical motors. We use the vertical motor to control the altitude and lateral motors to control the horizontal movement



Figure 2.2: Complete blimp selected. The balloon has a size of 52" and at the bottom is the tri-turbofan gondola

## 2. *Blimp Selection*

---

### 3. Electronic Circuits

By now, we selected a blimp for this project. So at this moment we have a balloon with 144 grams of load capacity, and a tri-turbofan gondola. The weight of the gondola is 70 grams. So we have 74 grams free to put in the rest of the components.

In this chapter we will describe which electronics systems we are going to use to control the blimp. We have to select which elements put into the gondola and which ones into the ground PC, in order to control the blimp.

We have two ways to select them: we can buy a commercial communication system or we can build our own communication system. There are no good commercial kits to control a blimp, so we have to design an appropriate communication system. The reasons for this decision are the following:

- we need to control the motors speed, because we can not control the blimp using only maximum speeds. Then we need a speed regulator for the motors. We use a big vertical motor. We can not connect this motor to the original board.
- we need a good range with few weight. We can not implement a wireless communication with a wireless access point, or a wireless router because we have only 74 grams of capacity. Then we have to select a lighter component to make the communications.
- we need a bidirectional communication to receive data from the blimp. The idea is send data from the PC to the blimp and backwards, because we need a closed loop, to control it.

So we had to design an specific board for this application, because the commercial boards do not satisfied our necessities. In the next sections of the chapter we will describe the functions of the circuit parts. We need two boards, one for the gondola, and another one for the PC. With this distribution we complete the hardware part.

## 3.1. Gondola Board

### 3.1.1. General Diagram

In this section we will explain the electronics for the gondola. We have three motors in the gondola and we need to control them. We have sensors in the blimp, that we use to get the measurements and send them to the computer.

In Figure 3.1 we see the system general diagram. We also have the connections between the different parts of the circuit.

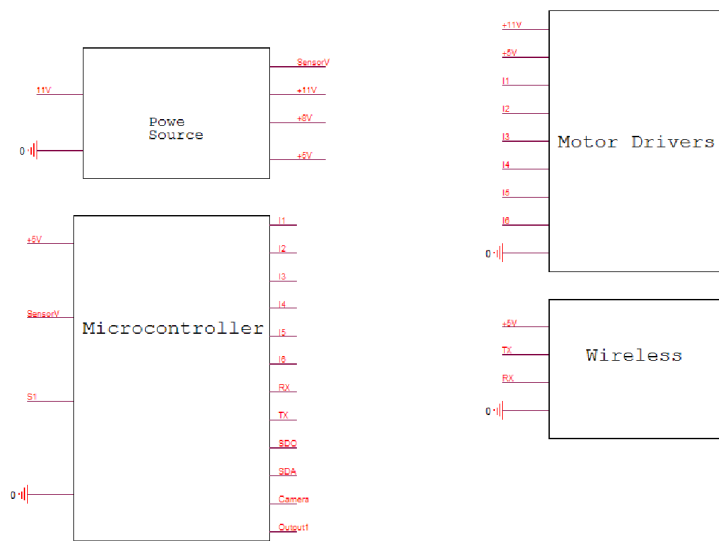


Figure 3.1: Block diagram for the gondola board. We see the main parts for this board. The microcontroller is the brain of this system

All diagrams blocks will be explained afterwards, in this chapter. We can see that the microcontroller is the heart of the system.

### 3.1.2. Power Source

All electronic system needs a power supply to run. Then we use a LiPo battery with 11 volts. This voltage is high for the electronic components, for this reason we put a power regulator into the circuit. We need to obtain two voltage levels: 8V and 5V.

We have three voltage levels in this circuit, each one for the different parts of the circuit. We can see the power source in the Figure 3.2. We connected the battery with the switch in the JH24 and JH28 holes.

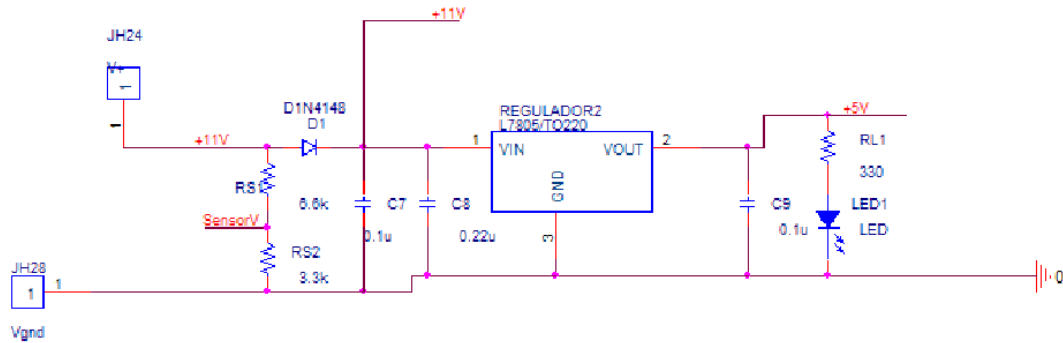


Figure 3.2: Power source diagram. The left side shows the voltage sensor. The central side shows the linear regulator. The right side shows the control LED, if it is ON the blimp is ON

The first part of the circuit is the battery voltage sensor. We made this sensor with two resistors (RS1 and RS2). These resistors reduce the tension in the measurement point and we can read it with the microcontroller analog port. With the voltage sensor we reduce the voltage until 1/3 of the original voltage. This sensor is necessary to obtain a computer control of the battery level. These batteries get damaged if they are under 9V. We use a LiPo saver to show us when the battery is empty.

The diode D1 is to 2A and we use it to prevent the possible polarity inversion. It is to protect the circuit. The 3 condensators and the regulator 7805 provide the 5V source. RL1 and LED1 show if the switch is turn on or off. When the yellow LED is ON the blimp is running.

The other power source provides 8V, for the camera. The real circuit diagrams are in the appendix Plans. The circuit is the same that the 5V source.

### 3.1.3. Microcontroller

The microcontroller is the brain of the circuit. The circuit has the necessary electronic to run the microcontroller. The Figure 3.3 shows the diagram.

R1 is a resistor that we use to connect the reset input from the microcontroller to 5V. This configuration removes the external reset interruption in the microcontroller. C1, C3 and Y1 are the oscillator to the microcontroller. It is fixed to 4MHz.

### 3. Electronic Circuits

---

The microcontroller is a PIC 16F873 [7], the technical characteristics are in the microchip web. We select this microcontroller because it has USART, analog ports, and I2C ports. We need this peripherals for the blimp control. The rest of connections are inputs or outputs to sensors or other parts of the circuit. The SDO/SDA ports can be used as normal input/outputs. These ports are designed to connect an I2C bus.

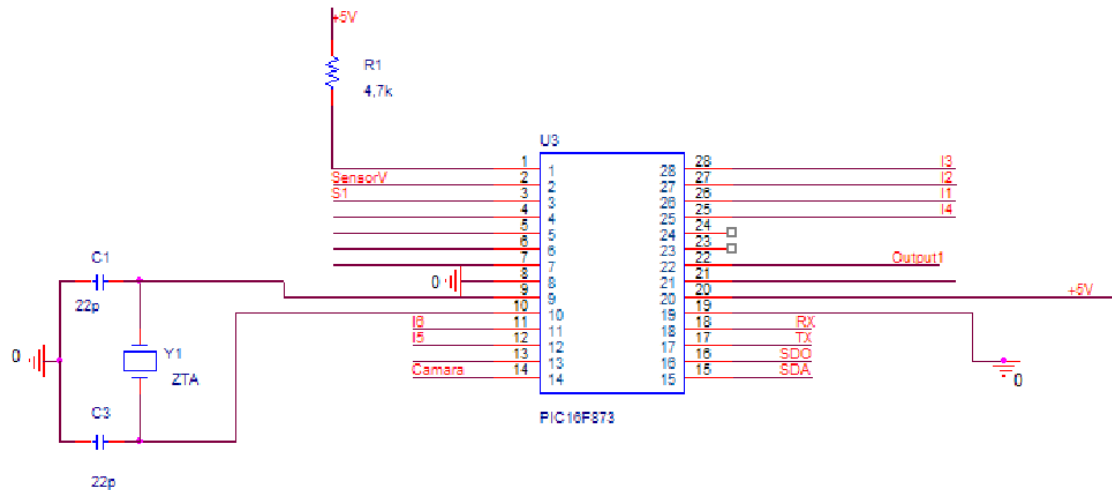


Figure 3.3: The microcontroller is the brain of this board. We see the oscillator and the connections with other devices

The microcontroller contains the code to control the blimp. This program will be commented in the next chapter. We can use the outputs and inputs to communicate with peripherals like a wireless transceiver, or motor drivers. We can see in the plans which is the complete connection between the different parts.

- We have I1, . . . ,I6 outputs to control the motor drivers.
- RX and TX are connected with the wireless module, RX is the receiver input and TX is the transmission output.
- SensorV is the Analogical input and convert the analogical signal from the voltage sensor in a digital data to send it to the computer.
- We use SDO and SDA to read the ultrasonic sensors.
- Output1 is not implemented.

### 3.1.4. Motor Drivers

We told at the beginning of this chapter, that we need a speed regulator to control the blimp motors. We can not use only one speed per sense. For this reason we use the drivers based in a L293B circuits (Figure 3.4). These circuits can work with 1A per canal. With this driver we can modulate the tension and control the motors speed. The modulation is a PWM (Pulse Width Modulation) and it is controlled by the microcontroller.

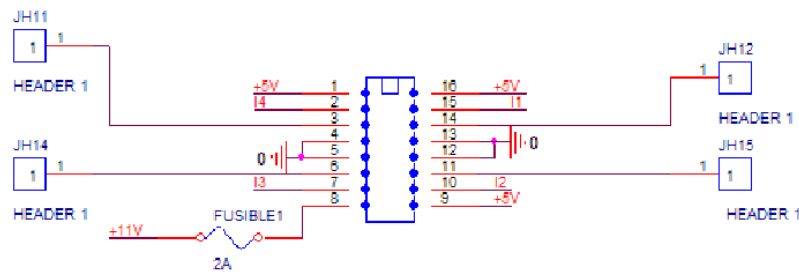


Figure 3.4: Motor drivers circuit. We see the driver (L293B) and the fusible to control the current

We can control 3 motors with 1A per motor. The blimp motors load is 300 mA per motor. We put a 2A fusible in the power input of drivers, to protect it from overload. We set the enabled input of all channels that we use. The enabled inputs are the pins 1 and 9. Every side of the integrated circuit controls one channel. The pin number 8 is the power supply and is used to move the motors. The pin number 16 is the logical power supply, it is used only to compare the voltage levels from the microcontroller and set or clear the power outputs. The Headers are the connections for the motors.

### 3.1.5. Wireless

We want to connect a personal computer with the blimp. For this reason we need a wireless circuit (Figure 3.5). For the blimp we select a ER400TRS transceiver. This is a serial wireless modem that works in the 400 MHz band. This circuit has a speed communication of 19200 Bauds, in the air. The communication with the pc is of 19200 bauds, but we can change it. The other part of the circuit is the 74HC04 that we use as a buffer, to protect the transceiver. The communication with the microcontroller is with TX and RX pins.

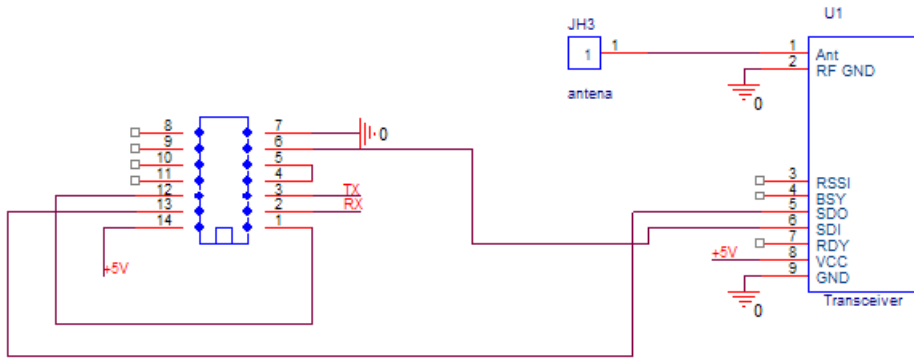


Figure 3.5: Wireless circuit. In the right side we see the transceiver. This transceiver make the connection between the gondola and the computer. In the left side we have a 74HC04 IC to protect the transceiver

### 3.1.6. Battery

The choice of the battery is very important for this project, because this element is very heavy. We need a very light battery and a lot of current. For this project we select a new type of batteries, the LiPo batteries. Lithium Polimer batteries are based in a new technology. They satisfy all our needs. They have a small weight and a lot of current capacity. We select a 350mA LiPo Battery (Figure 3.6), because it has weight of 35 grams and it can supply 3A as max current [5]. This battery provides 40 minutes of autonomy for the blimp. The voltage level of this battery is 11V, because if we put a camera we need 8V.



Figure 3.6: LiPo Battery with 11V, and 350mA

The critical level for this battery is 9V. In this level we have to turn off the load if we do not want to damage the battery. We use a commercial circuit to control the voltage level (Figure 3.7). It is for 3 cells batteries, then it turn on the led when the battery has the critical level (9V).

At this moment we have all hardware to control the blimp (without sensors). The weight of



Figure 3.7: LiPo Saver for LiPo batteries of 11V

gondola hardware is 20 grams and LiPo battery is 35 grams. Then we have 20 grams free to put sensors.

## 3.2. Personal Computer Board

### 3.2.1. General Diagram

We have to communicate the blimp with the computer, so we have to build other wifi board for the computer. You can see in the Figure 3.8 the general diagram for the PC board. In this small board we have three parts. The first part is the power source, that we need to supply current to the board integrated circuits.

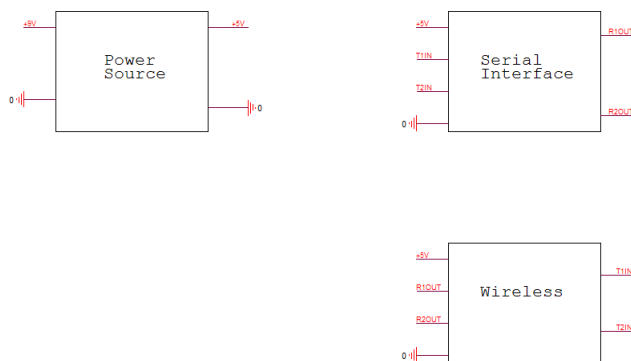


Figure 3.8: Blocks diagram for the PC board

Another block is the serial interface, that we use to connect the board to the PC. This block is only an interface and its mission is to change the voltage levels between the PC and the wireless module. The last part is the wireless module. We use this block to send and receive data between the blimp and the computer.

### 3.2.2. Power Source

The power source has the mission to supply 5V (Figure 3.9). This is the voltage necessary to supply the electronic components of the circuit. We use a 9V battery as power supply for the circuit. This is the reason because we need the regulator. The diagram is the same to the Figure 3.2 without voltage level sensor.

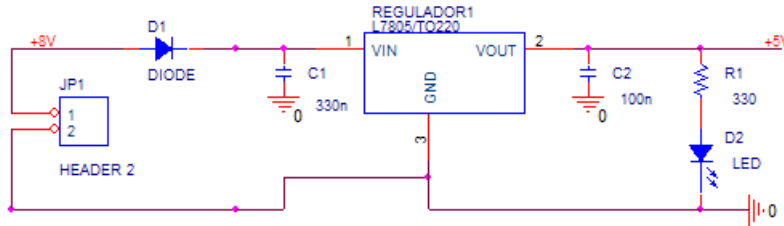


Figure 3.9: Power source for the PC board. Left side connections for the battery and a diode to control the polarity inversion. Middle side, lineal regulator to supply 5V. Right side, LED indicator

### 3.2.3. Serial Interface

This circuit (Figure 3.10) changes the voltages levels between wireless module and the computer serial port. The reason to use this interface is that we have a TTL voltage in the wireless side, and we have +12V, -12V voltage levels in the serial PC port. If we connect directly the wifi board to the serial port, the wireless module could get damage.

We have two parts in this circuit:

- Serial Connector: this is a DB9 connector, for a 9 pins serial port. The most important pins are the numbers 2 and 3. These pins have the serial information from/to the computer. The pins 7 and 8 are used for Handshaking flow control. But we do not use them. To connect this board to the PC we can use a serial cable if we have a serial port in the PC, or a serial-USB converter cable if we only have USB ports free.
- Serial driver: We use a MAX232N driver (Maxim driver). This driver uses 5 Tantalio condensators to run. It provides us 2 serial channels.

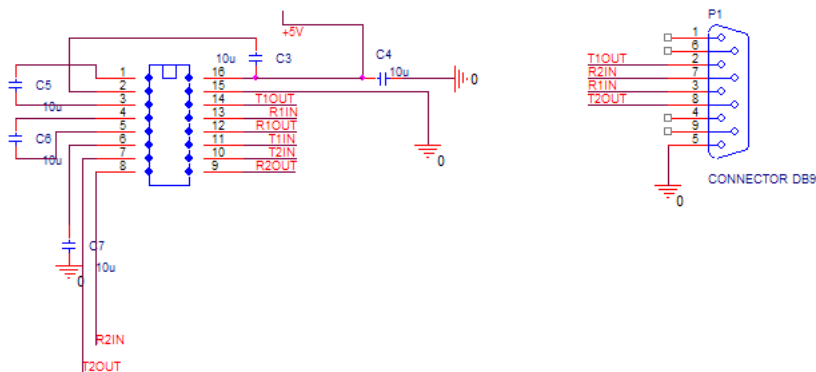


Figure 3.10: Serial interface diagram. We see in the left side the serial port driver. We see in the right side the serial port connector

### 3.2.4. Wireless Communication Module

The main part of this module is the Transceiver (Figure 3.11). This transceiver is the same that we have in the gondola wireless module (Figure 3.5). We need 5V to supply the transceiver. The antenna is connected to pin 1 and we send the data for this pin.

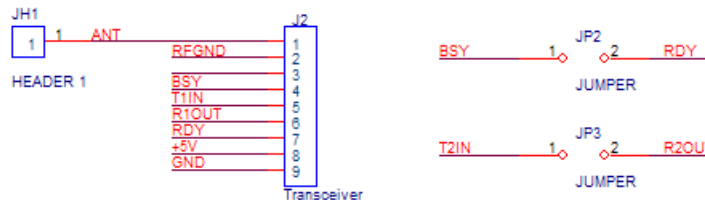


Figure 3.11: Wireless transceiver and jumpers to configure the error protocol

Jumpers are the other important part of this circuit. With these jumpers we can activate the Handshaking control flow. If we connect the pins number 1 with a jumper and the pins number 2 with another jumper, we activate the handshaking control, and we have to configure the transceiver and the computer to use it.



## 4. Firmware

By now, we have selected the blimp components and build all electronic parts. We have made the mechanical and electrical part. But the blimp still can not do anything. So in this chapter we will describe the algorithm developed for the microcontroller. This software controls the motors, takes the measures from the sensors and sends them to the computer.

This chapter and the next one are very important, because they explain how to control the blimp using a software when we will have an appropriate program in the microcontroller. The computer program send control data to the microcontroller. The firmware controls the blimp using these data. We use the computer wireless board to send instructions to the gondola, then the microcontroller reads these instructions and moves the motors if it is necessary. In other hand, the microcontroller gets the sensor measurements and sends them to the computer board. These data are received and processed at the computer. With these data the computer takes new decisions about the gondola motors.

### 4.1. Algorithm

At Figure 4.1 we can see the firmware algorithm. This program is into the microcontroller, inside the gondola. The program controls the motors and gets data from the sensors. It has several parts. All this parts are typical in microcontroller programs. We describe in this section the purpose of everyone important block.

The Serial Port Interruption Configuration block configures the call to Serial Port Interruption Subroutine. This subroutine processes the serial data received in asynchronous mode. We use interruptions to read the serial port, because we can do another task when we do not receive data. When one data is in serial buffer, the interruption flag is set and we can read and process this data and use it in our program.

Motors Control (PWM) block makes the modulation for the motors. We use 11V power supply for the motors. We change the motors speed when we change the voltage level. We use PWM to change the motors voltage and then we can control the speed. We have 3 motors to control the blimp. Those motors can not use more voltage than 5V. This situation is controlled by PWM block.

The next blocks run when the condition is true. We use one variable to synchronize the sensors measurements and data transmission. It is not possible make two sensors measurement at the

#### 4. Firmware

---

same time. We need a delay to avoid the interferences between those sensors when they are making a measurement. This is the reason because we wait between one measurement of one sensor and the next measurement of the another one.

We have two ultrasonic sensors. One uses PWM control signal (vertical sensor). The other one is connected to I2C bus. To use the PWM sensor we have to control the measurement process. We send an activation command to start the measurement. Afterwards we count the time that we are waiting for the end measurement command. This time is proportional to distance. With I2C sensor we send an activation message. Then the I2C sensor starts to measure. We can do other tasks in our microcontroller program while we are waiting for the measurement. When the measurement is ready we call to Read Measure Frontal Sensor function and we get the distance in centimeters.

When it has the information, it is sent to the computer using a wireless connection. With this chapter we finish the job with the blimp. Now we have to develop the software for the computer.

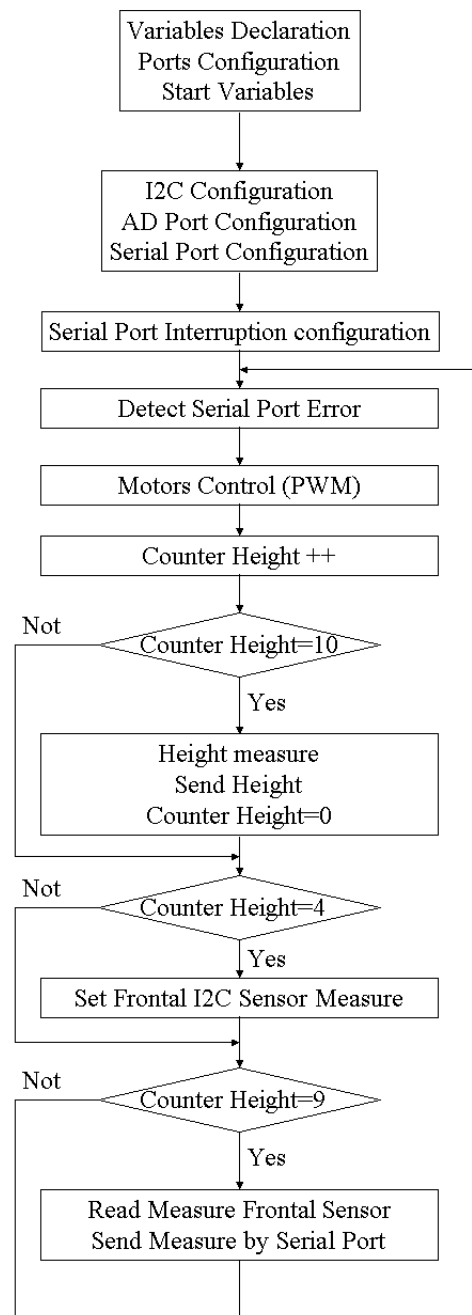


Figure 4.1: Algorithm programmed in the microcontroller. This is the firmware that controls the blimp. When we start the microcontroller the program start in the first block. This algorithm is an infinite loop



## 5. Driver

In the first chapters of this project we have described the hardware construction. So now we have a complete blimp. We have the hardware to control the motors and to get the measurements from sensors. We developed the software for the microcontroller, to use the instructions that the computer sends to it. And the microcontroller sends the sensors measurements to the computer. Then at this moment we need develop a software for the computer, because we have to send the control instructions for the blimp. To develop this software we need a driver to use the transmitter board under Linux. In this chapter we will describe the Linux driver.

To use the computer board we developed a Linux driver. This driver is an interface to control the blimp. We can develop a software using this interface to control the blimp. With this interface we have functions to control the blimp motors and read the sensors measurements. The transmitter board is connected to computer by USB port. We use an USB-RS232 converter to connect them. As technical information we would like to mention that transmitter board uses 8N1 to 19200 baud protocol.

When this chapter had ended, the blimp making will be finished, we will have all necessary tools to move the blimp. Then we will be able to develop software to control the blimp. But if we want to build an autonomous blimp, then we will need to develop a software that allow the blimp to do it. To make this easy we will need a minimum number of functions to do more easy the control. We will see this in next chapters.

### 5.1. The Interface

We want control the blimp using the wireless computer board. This communication is complex and we can not develop software if we have to control the system functions in our program. For this reason we develop a generic interface to give the programmer a package to control the blimp. With this interface the programmer does not need to control the serial port characteristics, or the communication protocol between the blimp and the computer.

Next sections give programmers a guide to know what functions are implemented and how these functions are. The interface has ten functions easy to use. So we think that the programmer can develop a software to move the blimp using this functions in few minutes.

Regard to the Linux system configuration, we need a serial port available and it is very important give the appropriate permission for the serial port. This is a typical error when a new programmer

develop software for the blimp and this one does not run. We solve this problem typing this command as root: `chmod 666 ttyUSB0` if the board is connected to `ttyUSB0` port.

### 5.2. Serial Port Functions

We need to control the serial port to send information to the wireless board [21]. For this reason we have 2 functions to control it. These functions are important because we need the serial port identification to use in our program. With this identification we can write in the serial port system file and send the necessary information to move the blimp. The second function close the serial port system file to finish the communication.

- *int open\_port (const char port[32],double \*time\_open)*

This function is used to open the serial port. We give it the name of the port (serial port or USB port) and the pointer to save the time when we opened the port. This function gives us a reference for the port. We can use this reference while serial port is opened.

- *void close\_port (int fd)*

This function is used to close the serial port. We use the last reference. We send it in this function and close the serial port.

### 5.3. Get Measurements

We want to make an autonomous blimp. So we need information from the blimp. This information comes from the sensors. The firmware uses the sensors and send to computer board the measurements to be process by our program. So we need functions to read this information and use it in our programs. That is why we have three functions to read the blimp information:

- *int get\_voltage(int fd,double \*destination\_time)*

This function give us the battery voltage. The parameters are the serial port reference and the pointer to save the measurement time. The return value is an integer with battery voltage.

- *void get\_height(int fd,double \*value\_destination,int \*flag,double \*destination\_time)*

This function give us the blimp altitude. The parameters are the serial port reference, the pointer where we will save the result, a pointer to save a flag and another for the time. The returned altitude is in meters. The flag tells us if this measure is new or old.

- *void get\_front(int fd, double \*value\_destination, int \*flag, double \*destination\_time)*

This is the last function to read data from the blimp. The parameters are the same than last function. In this case we read the frontal collision distance. We use this measurement in the collision avoidance system.

## 5.4. Movement Functions

At this moment we can use the serial port and read data from the sensors. For this reason we need new functions to control the motors. We can control one, two or three motors at same time. These functions send the desired speed for the motors.

- *void motor\_left(char speed, int fd)*

This function controls the left motor. The first parameter is the motor speed, this parameter is between -12 to +12. With this parameter we can turn the motor in two directions. Positive values moves forward. Negative values moves backwards. The other parameter is the serial port reference.

- *void motor\_right(char speed, int fd)*

This function is like the last function but this one manages the right motor. We can turn the blimp or move to front or back if we combine this two functions.

- *void motor\_up(char speed, int fd)*

This function controls the vertical motor. It is like the other two functions but the speed range is -14 to +14. Positive values move up and negative values move down the blimp.

- *void motor\_left\_right(char speed\_l, char speed\_r, int fd)*

This function is used to control the horizontal motors at the same time. The parameters are the same with the same ranges.

- *void motor\_left\_right\_up(char speed\_l, char speed\_r, char speed\_u, int fd)*

This is the most general function. It can control the three motors at the same time. The parameters are the same than in the other functions.



## 6. Altitude Regulator

With the approach described so far we are able to control the blimp with the computer. Therefore we have built the hardware and implemented the electronic systems for the control. Afterwards we developed the firmware for the microcontroller and so we can send and received data from the PC.

In the previous chapter we wrote a driver under Linux to control the blimp using a serial port. Now, for safe navigation in indoor environments, we want to implement some algorithms which make easier to control the blimp. One important task for an autonomous blimp is an altitude regulator. Therefore we want to specify a goal altitude and the altitude control algorithm drives the blimp automatically to this height. With this control algorithm we can reduce from three degrees to two degrees of freedom, because one degree (vertical) is controlled with this function. This function controls the altitude of the blimp, so we only have to control the horizontal movement. In this chapter we want to explain how we realized this function.

For this function we need the current altitude of the blimp. We use an ultrasonic sensor to get it. Once we have the measurement, we correct the error of the sensor with a Kalman filter, and obtain the estimated height. Kalman filter uses the characteristics of our sensor, our dynamic model for the blimp and the previous measures. Kalman filter gives us the blimp estimated altitude in each moment. As soon as we know the altitude of the blimp, we can apply a regulator to control the altitude. We can control the motor and move the blimp to the specific height. For this control we compare two controllers in order to generate the actions for the motors.

### 6.1. Ultrasonic Sensor

If we want to build an autonomous blimp, we need information about the environment. For this task we use an ultrasonic sensor. We use a SRF05 sensor [15], see Figure 6.1, to measure the distance from the blimp to other object in the bottom. Using this sensor we implement a height regulator for the blimp. This sensor has 1 mm resolution, and a small ratio. We can only detect objects which are directly in front the sensor.

In Figure 6.2 we see the sensor connections. It has two modes. Mode 1 uses 2 pins for the control. Mode 2 uses only one pin for the control. We use mode 2 with a single pin for both, Trigger and Echo, connections. Therefore we can control the sensor using a single microcontroller pin.

Using mode 2 we have the next control signals (Figure 6.3). The Trigger pulse starts the sensor



Figure 6.1: We see an ultrasonic sensor SRF05. We used it to get the measurement of the altitude. This sensor weight 10 grams

measurement (the microcontroller pin is an output), this pulse has  $10\mu s$  of minimum duration. Then we configure the pin as an input. We have a high level until the height measure is over, then we have low level on the microcontroller pin. This sensor uses a PWM connection to give the measurement to the microcontroller. The time that the connection is high is proportional to the distance, so we use this value to obtain the altitude. To calculate this distance we use

$$distance = \frac{t_{measure} * 5}{2900} \quad (6.1)$$

and we obtain the distance in meters. This equation was obtained with experiments and datasheet.

We make some experiments with the sensor to obtain the characteristics. We get measures with the sensor at different distances from a wall. With this data we obtain the statistic parameters which are necessary to characterize the sensor. We get the media of the variance. The sensor is characterized by  $\sigma_{sensor} = 1.15mm$ .

We want to measure the height. Sensors are imperfect and measure with noise. We want to estimate the height of the blimp to reduce the errors in the measurement. For this application we use a Kalman filter.

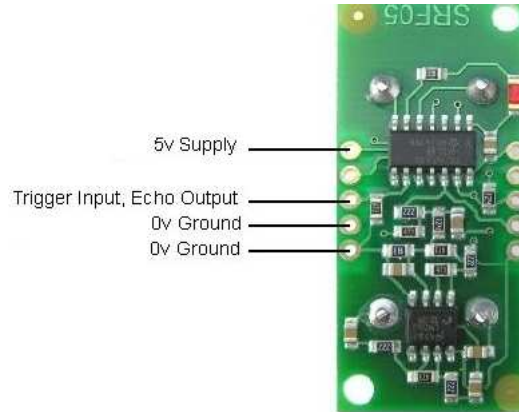


Figure 6.2: SRF05 connections to run the sensor in mode 2. With this mode we use a single pin of the microcontroller to control the sensor

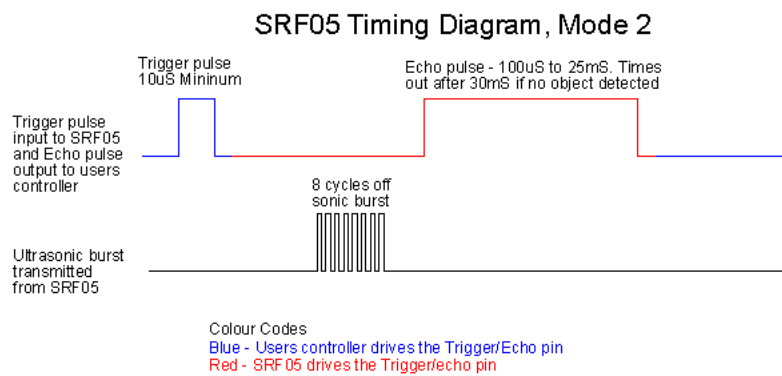


Figure 6.3: Control times for the ultrasonic sensor SRF05

## 6.2. Kalman Filter

As we described at the beginning of this chapter, we want to put the blimp in a specific altitude. To make this task, we need to know the height at each moment. Therefore we put a sensor at the bottom of the blimp. Sensors are not perfect and they give us wrong measurements which we have to correct. For this reason we use Kalman filter [37], because we can get better measurement using it .

The Kalman filter is a set of mathematical equations that provides a recursive solution of the least-squares method. The goal is to find unbiased minimum variance lineal estimator of the state at time  $t$  with base in available information at time  $t - 1$  and update with the additional available information at time  $t$  that estimator. This filter is the principal algorithm to estimate dynamic systems specified in state-space form.

So our objective is to obtain the optimal height measurement of the blimp. We are using noisy measurements from the sensors, and a doubt model of the blimp dynamic. The blimp dynamic is modelled using this equation

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t, \quad (6.2)$$

where  $\bar{\mu}_t$  is the state model for the actual moment,  $A_t$  is the conduct matrix for the system dynamics or state matrix, and  $B_t u_t$  is the actuation vector.

Kalman filter algorithm uses two cycles, propagation cycle and actuation cycle. The propagation cycle obtain the new state using the values that we have in our dynamic system model. In the second cycle we use the measure, and we estimate the new real position for the blimp. We have an interactive algorithm, and it actualizes itself continuously with the new measurements.

The algorithm uses as inputs the last state vector  $\mu_{t-1}$ , the covariance matrix of the system model  $\Sigma_{t-1}$ , the actuation  $u_t$ , and the new measure  $z_t$ . At first we have to assign to this inputs initial values. Next equation calculates the estimator covariance without the measure

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t, \quad (6.3)$$

where  $R_t$  represents the noise of the system model. We are using the variance matrix of the last state  $\Sigma_{t-1}$ . Now we have the new variance of the state equation, but we do not have used yet the new sensor measurement. For this reason we use other symbol, because  $\bar{\Sigma}_t$  is not the definitive variance for the  $t$  time.

Now we calculate the Kalman filter gain with this equation

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}, \quad (6.4)$$

where  $Q_t$  is a doubt vector for the sensor. Gain makes the error covariance of the new state estimation minimum. It specifies the weight of the measurement in the new estimated state. Now we can use the measure to correct the result obtained with the system equations, and for this goal we use the next equation

$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t). \quad (6.5)$$

With this equation we know the estimated altitude for the blimp. This equation give us the new covariance of the dynamic system to use it with the next measurements

$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t. \quad (6.6)$$

So when we apply the filter, we obtain the altitude  $\mu_t$  estimated for this time and the new value of the covariance matrix  $\Sigma_t$ .

In this section we estimate the height using Kalman filter. Now we have to apply a regulator with this measurement to control the vertical motor and put the blimp in the goal altitude.

### 6.3. PID Regulator

Finally, we have all the necessaries parts to make the altitude regulator. We can control the blimp with the computer and know the estimated altitude for the blimp. So we have to select a regulator type. The first idea is to select a PID regulator, because this is a typical regulator for lineal systems [34] [10]. It can erase the error of the system. We see in Figure 6.4 a graph for the closed loop system. In this graph we see the height of the blimp. This height is measured using kalman filter. We compare this measurement with the goal and the difference is the error. We want to erase this error or minimize it, for this reason we use the controller. The controller send the actuation commands to vertical motor. This motor moves the blimp to a new altitude, but this altitude is function of the environment perturbations. So we can calculate a controller for a specific environment with specific perturbations.

The PID (Proportional Integer Derivative) regulator has three components. All these components are function of the system error  $e(t)$ . The proportional component is

$$P = K_p e(t), \quad (6.7)$$

where  $K_p$  is the proportional constant. Using this component we move the blimp fast to the goal, but can not erase the error. The Integer component is

$$I = K_i \int_0^t e(t) dt, \quad (6.8)$$

## 6. Altitude Regulator

---

where  $K_i$  is the integer constant. In this component we store the error. This component erase the error, but insert oscillations in the behavior of the blimp. The derivative component is

$$D = K_d \frac{de(t)}{dt}, \quad (6.9)$$

where  $K_d$  is the derivative constant. This component is function of the error change. When the error changes this component generates the actuation to correct it. We use the error in the last state and the error in the actual state to calculate the variation. Using this variation we create a new actuation proportional to correct the altitude. So the complete regulator is function of the error. We have here the complete equation

$$PID = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt}, \quad (6.10)$$

We see in the Figure 6.5, what is the effect when we add one new component. The proportional component moves quickly the system near to the reference. But this component can not erase the error. We add a Integer component. This component erases the error, but increases the oscillations. The third part is the derivative part, when we apply this component we reduce the overshoot.

We have to calculate  $K_p$ ,  $K_i$  and  $K_d$  components. We need a mathematical model for the system to calculate this parameters. We can not model the system, so we have to estimate those parameters using experiments and *Ziegler Nichols method* [18]. The problem is that we have to recalculate the parameters when the environment or blimp conditions change.

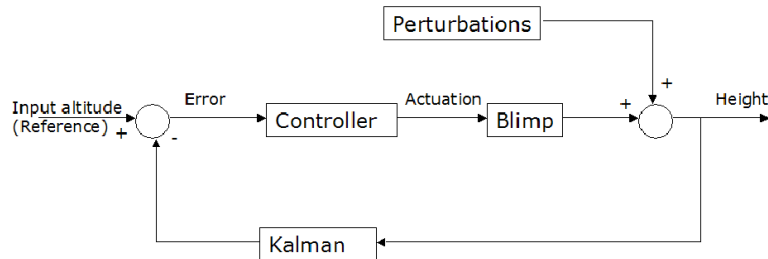


Figure 6.4: Closed loop to control the blimp. Kalman filter get the measurement of the height. The error is the difference between the reference and the Kalman measurement. We use the controller to drive the vertical motor and minimize the error. The perturbations are the problem because we can not control them

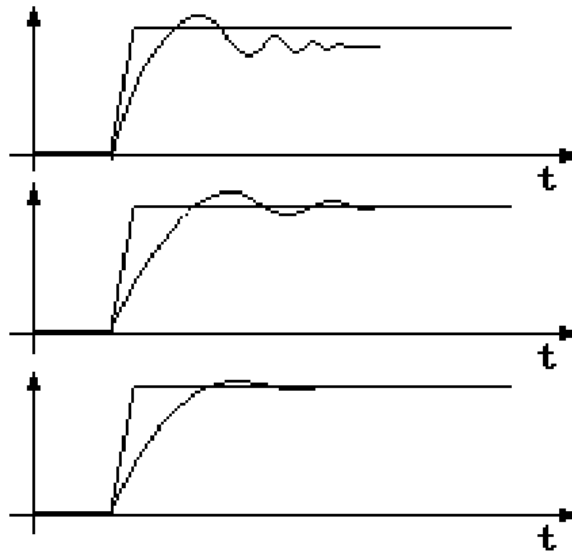


Figure 6.5: PID comportment. The first graphic shows the system with Proportional component in the regulator PID. The second graphic shows the effect of PI components. The third graphic shows the complete PID regulator controlling our system

## 6.4. Fuzzy Regulator

We also can select an other more generic regulator. This regulator is a Fuzzy regulator. Fuzzy uses the knowledge that we have about the system. This regulator is not exact but is more generic, because we do not need to recalculate the parameters when the environment changes [33].

Fuzzy logic uses Fuzzy groups to model the system doubt. The knowledge representation is modeled under rules. So we have to select the variables that we need for control the blimp. We use the height and the estimated speed, because we want to control the inertia. This is the main advantage respect the PID controller, because we use two variables to control the blimp, and PID uses only one. So, we see in the Figure 6.6 the Fuzzy groups. We use the estimated speed, the system error, and the actuation on the motors. This last group is the output for the regulator.

Now we know the variables to use, we give them operation values. Then we build the Fuzzy rules. With this rules we calculate the speed of the motors. We see the Fuzzy rules in Figure 6.7

We will see this problem using two random situations and using the images of this chapter, to

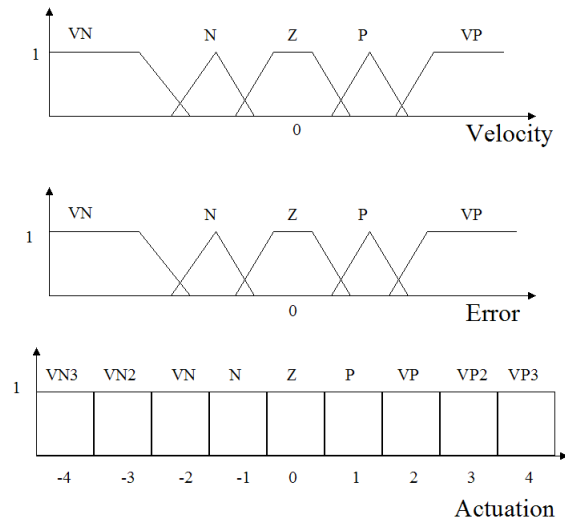


Figure 6.6: Fuzzy Groups

**(E) Error =  $Z_{ref} - Z$**

	VN	N	Z	P	VP
VN	Z	P	VP	VP2	VP3
N	N	Z	P	VP	VP2
Z	VN	N	Z	P	VP
P	VN2	VN	N	Z	P
VP	VN3	VN2	VN	N	Z

VN → Very Negative  
 N → Negative  
 Z → Zero  
 P → Positive  
 VP → Very Positive

Figure 6.7: Fuzzy rules. This table gives us the Actuation (A) in function of the inputs, Error (E) and Velocity (V). We write this table by hand

know how this regulator works. In the Figure 6.8 we see the blimp, the ground and the control variables. In the right side we have  $Z_{ref}$ , this value is the height reference. We want to put the blimp in this height. The  $Z$  variable is the estimated height using Kalman filter. In left side we see the other variables sensed.

The first example is a situation where we have VP Velocity and the Error is VP. This situation means that the blimp is far from the reference and the reference is over the blimp. The velocity is maximum in positive direction. So we do not have to actuate on the system. We see this situation in the fuzzy rules (Figure6.7). We see VP in Velocity and VP in Error, so the value is Z for the Actuation. We can follow this example with Figure 6.8. We do not have to actuate on the system.

Afterwards, we have a different situation. We have VP Velocity, and P Error. In this situation the distance to the reference is smaller than the last situation. Here we have to reduce the speed and reduce the system inertia. So we have to use the vertical motor to generate a down force (negative actuation). We see in Figure 6.7 that in this situation Velocity is VP and Error is P, then the Actuation is N (negative).

With this rules and this groups we can place the blimp in a specific height. We see in Figure 6.6 the error group, this group has a dead band to avoid the continuous motor actuation.

In experiments chapter we will show how this regulator runs. Until this moment we have the blimp, with electronic circuits, and the PC interface. With this chapter we can control the height. Then, now, with this function we can turn on the blimp and it will go to the reference.

In this project we do not use the variance of the Kalman filter, because we have few steps to control the actuation for the motor. So if we have more steps we can make more precise the control using the variance obtained from Kalman filter.

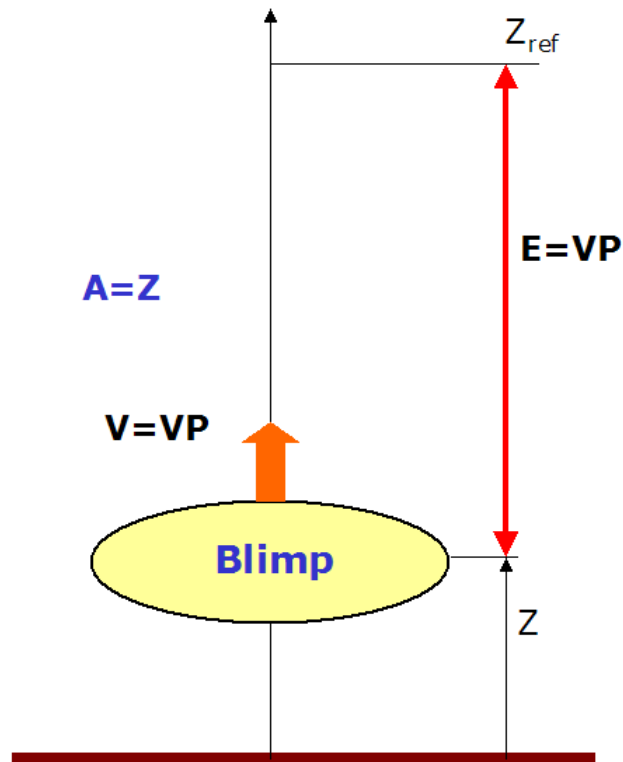


Figure 6.8: This image shows the behavior of the blimp in a specific situation. The Error is VP and the Velocity is VP, so, as we see in Fuzzy rules, the necessary Actuation value is Z

## 7. Collision Avoidance

The approach described at the moment has been the blimp construction and height regulator. We have the height regulator function to put the blimp in a specific height. We have functions in the interface to move the blimp, but, if we move the blimp forwards, the blimp crashes with all the obstacles that it finds. For this reason we have to develop a collision avoidance system. With this system we can move the blimp in an indoor environment where it flies without collision.

To build an autonomous blimp we need sensors to get measurements of the environment. We use a frontal ultrasonic sensor to avoid the frontal collision. This sensor is in the frontal side of the blimp. With this sensor we get the frontal distance measurement. We use this sensor to develop a regulator to avoid the collisions. In this chapter we explain the sensor characteristics and how the regulator runs.

### 7.1. Ultrasonic Sensor

We want to develop a collision avoidance system, so we need a sensor to get the measurement of the frontal distance. We have some types of sensors to get the frontal measurement. We can select an ultrasonic sensor, an infrared sensor or other light sensor.

We use an ultrasonic sensor similar to the altitude sensor. In this case we select a SRF10 ultrasonic sensor [16], because it is lighter and it has an I2C interface. I2C is a serial bus to connect electronic systems between them. For example, we have the microcontroller and we can connect lots of sensors using only two microcontroller pins. In this situation the microcontroller is the master and sensors are the slaves. So with this bus we can connect more sensors using only two pins.

To connect this sensor to the microcontroller board, we need to put one resistor of 3k3 ohms between each signal pin and 5V. The sensor can only work with 5V, so it is very important not to use another work tension. We see in the Figure 7.1 the sensor pins that we have to connect with the microcontroller board. We connect those pins to the correct ones of the microcontroller and to the stabilized power supply.

To control this sensor we use the I2C bus. This bus uses a specific address for the components. SRF10 has a pre-programmed address, but we can change it if its necessary. In our application we use the factory address. We send commands using the bus to configure the measurement in centimeters.

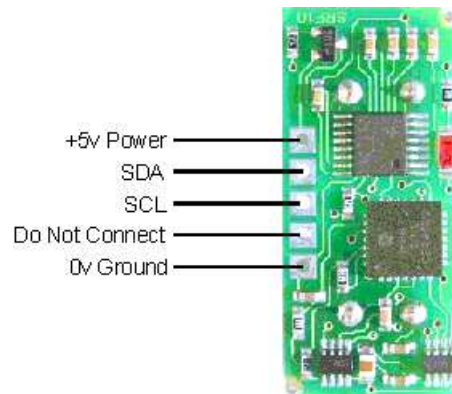


Figure 7.1: SRF10 connections for the I2C bus. We see the power connections and data connections. SDA and SCL are the data connections to control the I2C bus

When we send the instruction to get a measurement, the sensor can not receive other instruction until it finishes the actual measurement. When the sensor is getting a measurement, we can do other tasks in the microcontroller. Afterwards we get that measurement we can send it to the PC board using the wireless connection.

This sensor has bigger ratio than the altitude sensor. We use this characteristic to detect walls when the blimp is close to them, but not directly in front of them.

Now we can develop the collision avoidance function, because we have a sensor to detect the frontal distance. We can close the regulation loop using this measurement. In the next section we will comment the solution adopted to solve this problem.

### 7.2. Collision Avoidance Function

In this chapter we have seen what is the actual situation of the blimp and we have implemented an ultrasonic sensor to measure the frontal distance. We programmed the microcontroller to read those data and to send them to the computer. So, as we have the data in the computer, we can read them using the interface functions.

Using this data we know what is the frontal distance between the blimp and the obstacles. We want to avoid the obstacles because the blimp must not crash with any object. For this reason we develop a function to control the frontal distance. This function gets the control when an obstacle is near to the blimp.

Like in altitude control functions, we can use many types of regulators, but we have to study what regulators are good for our application. The problem is that we do not have a linear system

**(E) Error =  $Z_{ref} - Z$**

Z    P    VP

<b>(V) Velocity</b>	VN	VP	VP2	VP3
	N	P	VP	VP2
	Z	Z	P	VP
	P	N	Z	P
	VP	VN	N	Z

VN → Very Negative  
 N → Negative  
 Z → Zero  
 P → Positive  
 VP → Very Positive

Figure 7.2: Collision Avoidance Rules. This table has the Actuation (A) values to avoid the collision. We write by hand this table

and we can not obtain a model of this system.

So, in this section we use a Fuzzy controller to develop the avoid collision function. The idea of this controller is to put the blimp in a safe distance from the obstacle, but using the control of the inertia. Fuzzy controller gives us an index of speed to use in the horizontal motors. The inputs are the frontal distance and the estimated speed. With this data the fuzzy controller gives us an index of speed for the motors.

We saw in the last chapter how fuzzy controller works, now we will see the parameters that it needs and the rules it uses in this regulator. We have the fuzzy rules in Figure 7.2.

We see in Figure 7.3 how the regulator runs. We will follow some examples to know how to run this regulator. In the first situation, the blimp goes to the wall with speed  $V=VN$  and the blimp has a VP error (near the wall). Here the blimp wants to crash with the wall but the collision avoidance function takes the control and gives the motors a VP3 action. This means that motors give the blimp an opposite force to avoid the collision, and, at that point, the power is the biggest because this is the most critical situation to crash.

If we use this speed a long time the blimp acquire a big inertia and it is impossible to stop it in the reference. Fuzzy controller uses sensor data to control this situation. When the velocity is N the blimp does not need the maximum power, so, we see in rules the new index necessary for the control and this Actuation is VP2.

With those examples we show how the controller works and it is easy to see that the blimp has a small or null inertia when it is in the reference. In experiments chapter we will see the results of this function.

## 7. Collision Avoidance

---

It is important to control the return value and to take decisions over the regulator. Normal configuration is not to run motors if collision avoidance system is running at the same time. When the blimp is out of collision area, we can control the blimp with our control program.

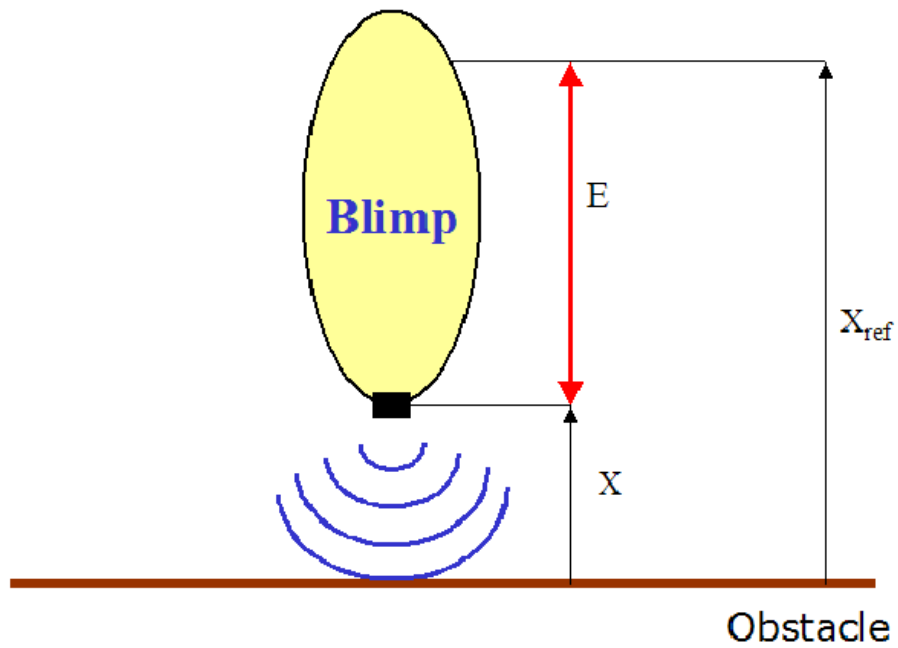


Figure 7.3: Collision Avoidance Diagram. We see the blimp and an obstacle. The ultrasonic sensor gets the measurement and Fuzzy controller avoid the collision

## 8. Experiments and Results

The approach described above has shown how to build a blimp. We developed the necessary software to control the blimp as well as two functions to control the altitude and collision avoidance.

In this chapter we show the experiments for these functions. The main part of these experiments is the comparison between PID and Fuzzy regulators. We want to know which is the most appropriate to control the altitude of the blimp. Therefore we have to design an experiment to compare these regulators. The final section of this chapter is the collision avoidance function, where we show how this function works. Finally, we present an application example which shows the results of this function.

### 8.1. Height Control

The first experiment is designed to compare PID and Fuzzy regulators. We want to know which regulator is the more appropriate to control the altitude of the blimp. We want to conduct these experiments using the same conditions for both regulators. In the following these methods will be described.

In the first experiment both regulators are tested alternating in environment 1. This experiment was repeated eight times. So we got the information for these regulators in the environment 1. This environment is a corridor inside 79 building of Freiburg University.

When we got this data, we changed the environment and put the blimp in an environment 2 with a lower temperature. This environment is the room of photocopier inside 79 building of Freiburg University. This room has a door to go out of the building. We got the information for these regulators in the environment 2. The idea was to reduce the influence of time and the changes of the environment on the blimp. Using this configuration we obtained the same perturbations on the blimp for both regulators. So we can compare this two regulators.

#### 8.1.1. PID Regulator

In this subsection we analyze the behavior of the blimp, while we apply a PID regulator to control the altitude. We conducted 16 experiments to get the data for this controller, eight experiments

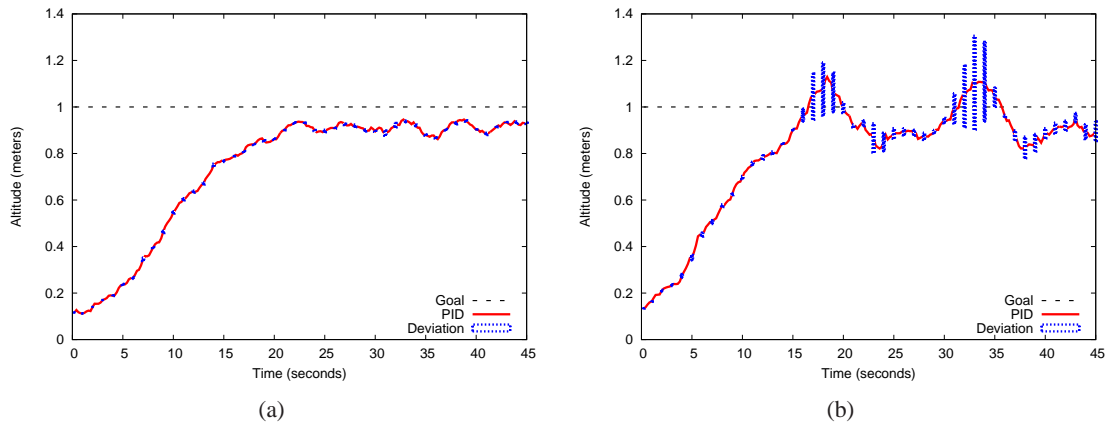


Figure 8.1: Behavior of the blimp when we set the goal position in 1 meter height. We use PID to control the blimp. This experiment takes place in environment 1 (Figure 8.1(a)) and in environment 2 (Figure 8.1(b)). We see a good behavior and low deviation in the data of environment 1, but we see a bad behavior and big deviation in environment 2

per environment. The blimp starts from a quiet state on the ground and afterwards it goes to its final position, which is fixed 1 meter above.

In Figure 8.1(a) we can see the results for this regulator in environment 1. The behavior of the blimp is depicted in red. This graph shows the mean of the experiments. We see a good behavior for the blimp. The deviation from the mean, shown in blue, is small. So the experiments have similar results.

In Figure 8.1(b) we can see the results for the regulator in environment 2. The behavior of the blimp is depicted in red. This graph shows the mean of the experiments. We see a bad behavior for the blimp. The deviation from the mean, shown in blue, is big. So the experiments have different results using the same regulator in the same environment. So if we change the conditions of the environment we have to recalculate the parameters to obtain a good behavior.

### 8.1.2. Fuzzy Controller

In this subsection we analyze the behavior of the blimp, while we apply a Fuzzy controller to control the altitude. We conducted 16 experiments to get the data for this controller, eight experiments per environment. The blimp starts from a quiet state on the ground and afterwards it goes to its final position, which is fixed 1 meter .

In Figure 8.2(a) we can see the results for this controller in environment 1. The behavior of the

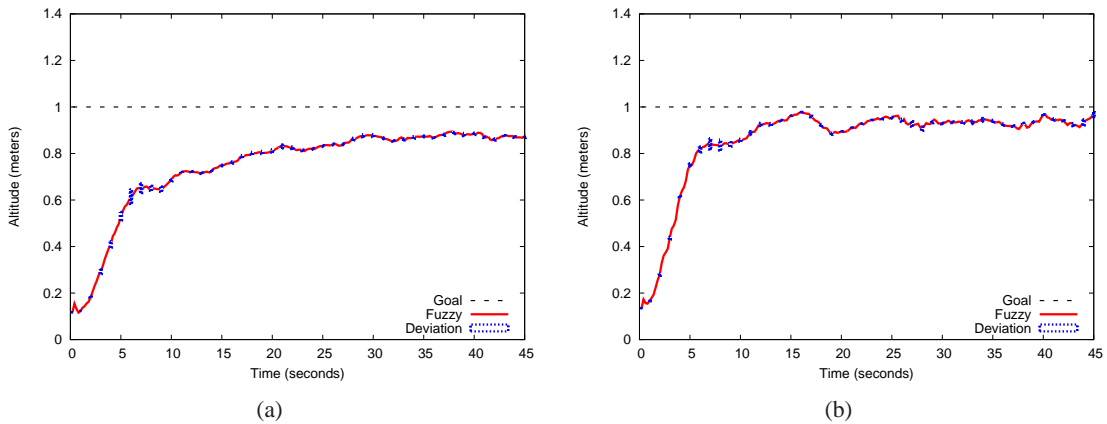


Figure 8.2: Behavior of the blimp when we set the position in 1 meter height. We use Fuzzy to control the blimp. This experiment takes place in environment 1 (Figure 8.2(a)) and in environment 2 (Figure 8.2(b)). We see a good behavior and low deviation in the data of both environments.

blimp is depicted in red. This graph shows the mean of the experiments. We see a small error and good behavior for the blimp. The blimp does not show oscillations. The deviation from the mean, shown in blue, is small. So the experiments have similar results.

In Figure 8.2(b) we can see the results for this Fuzzy controller in environment 2. The behavior of the blimp is depicted in red. This graph shows the mean of the experiments. We see a small error and good behavior for the blimp. The blimp does not show oscillations. The deviation from the mean, shown in blue, is small. The experiments have similar results. So we have good results in both environments.

### 8.1.3. Comparison PID and Fuzzy

Using the data obtained in the two previous experiments we will now compare the PID and Fuzzy controllers. In Figure 8.3 we can see the behavior of the blimp in environment 1. We see that the PID controller shows a better behavior than the Fuzzy. We got the specific parameters for PID in this environment using the Ziegler-Nichols method. In environment 1 we see that PID controller is slightly better than Fuzzy.

In Figure 8.4 we can see the behavior of the blimp in environment 2. In this environment the blimp shows a bad behavior using PID. The blimp shows an oscillating behavior with a large deviation from the mean. The behavior of Fuzzy controller is similar in both environments. So we see that the Fuzzy controller is more general than the PID controller. Fuzzy is not better than

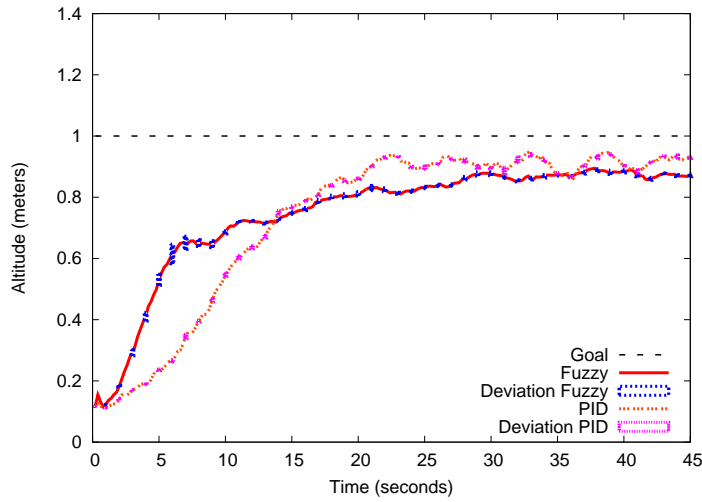


Figure 8.3: Behavior of the blimp when we use Fuzzy and PID controllers. This experiment takes place in environment 1. We see a good behavior and low deviation in the data using both regulators

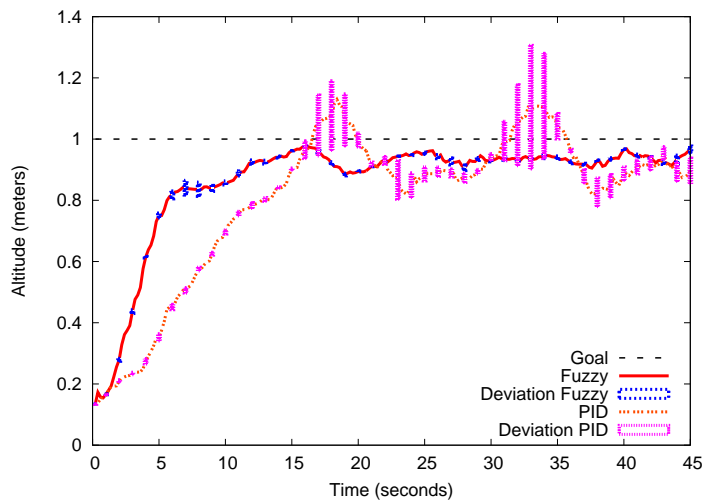


Figure 8.4: Behavior of the blimp when we use Fuzzy and PID controllers. This experiment takes place in environment 2. We see a good behavior and low deviation in the data using Fuzzy controller but using PID we see a big deviation and bad behavior

PID but is more general. Using a Fuzzy controller changes in the environment do not affect the behavior of the blimp too much, and we do not need to change the regulator parameters. For PID we have to recalculate the parameters to see a similar behavior for the blimp in different environments.

## 8.2. Collision Avoidance

The goal of this function is to avoid collision with obstacles. We put the blimp in the corridor and give it a horizontal movement. We see in Figure 8.5 the distance to the obstacles. The blimp has not crashed in any moment.

When the blimp is very close to the obstacles, the collision avoidance system takes over control. At this moment the horizontal movement controller is switched off and the movement of the blimp is controlled by the collision avoidance system. When the sensor does not detect obstacles close to the blimp, the main controller takes over control again.

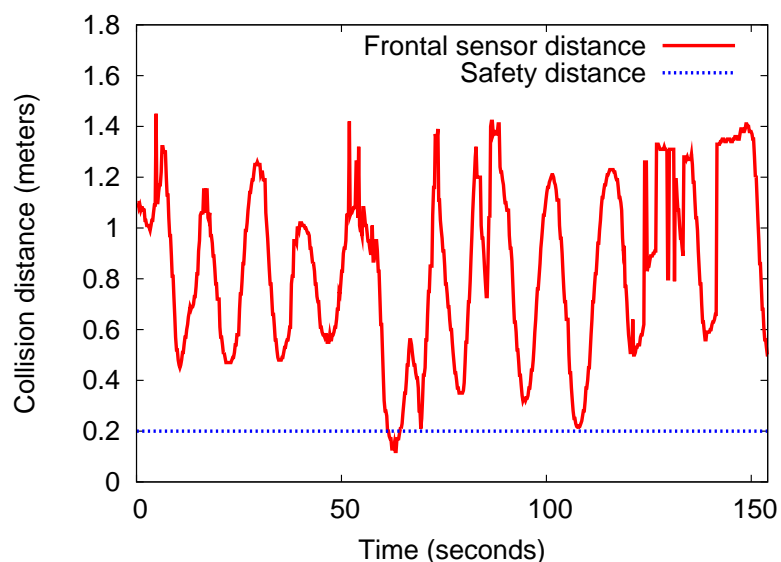


Figure 8.5: Behavior of the blimp when it is in a corridor. We move it with a simple program. The blue line is the safety distance for the blimp. We see the distance to obstacles for the whole experiment.



## 9. Future Work

In this thesis we described the complete process how to build an autonomous blimp. We selected a blimp, built the electronic boards to control it and developed the necessary software for the microcontroller and the computer. Later we put sensors on the blimp and developed two basic functions to control it.

Now we can develop software to move the blimp. We can move it more easy using these functions, but we can do more tasks such as put more sensors on the blimp. In doing so the blimp can get more data from the environment and with the appropriate software it can be more autonomous . This is the reason why we have to incorporate more sensors on the blimp.

The first sensor which we can add to the robot is a compass. We can control the altitude with ultrasonic sensor and we avoid the obstacles using the other ultrasonic sensor. The problem is that we can not move the blimp in an specific direction, because it turns through the contact with air. So, if we add a compass we can develop a controller for the orientation. We can assign a direction and using the compass measurement as well as the horizontal motors we can move the blimp in this direction.

The second sensor we can add to the blimp is a camera. The camera is light and we can use it to get information from the environment. If we use this information we can get the position for the blimp in an indoor environment.

Another possibility is to put two cameras. We can use stereo vision in the blimp to get distances and create maps [31]. With this information we can get the position of the blimp. We can use these cameras to localization and SLAM (Simultaneous Localization and Mapping)

When all these task are ready, we can get another blimp for outdoor environments. We can use all these functions with the other blimp. We can put heavy sensors on an outdoor blimp, then we can use it for a lot of different applications. In this outdoor blimp we can put a GPS to control the position and navigation.

Later on we can implement an intercommunication system between different robots. We can use some robots at the same time sending information to each other. For example send information from the blimp to a wheels robot in a rescue operation. Another example is to send information about the environment to wheeled robots.

## 9. *Future Work*

---

## 10. Conclusions

In this project we developed the complete blimp. In the first chapters, we selected a blimp and built the necessary hardware to control it. Afterwards, we developed the necessary software for the microcontroller and the computer. With this software we could move the blimp with the computer.

Now we needed some basic functions in order to control the blimp more easy. Then we wanted to develop two functions. The first was the altitude control function. The second was the collision avoidance function.

When developing the altitude control function we had to select a regulator. We could choose between a PID and a Fuzzy regulator. Then we made a comparison between these two regulators and found out that the Fuzzy regulator was more general regulator than the PID. We did not change the parameters when we changed the environment and we got similar behavior when using the Fuzzy controller. When using the PID we obtained different behavior when we change the environment. We obtained a big deviation between experiments therefore when we selected the Fuzzy controller as a regulator for our functions.

Now we have the blimp running with these two functions. We can select a altitude desired for the blimp and the blimp goes in direction of its position. We can move the blimp using the collision avoidance function and the blimp has not collision with the obstacles.

## 10. Conclusions

---

## A. Components

In electronic circuits chapter, we saw how we designed the hardware for the blimp. We use electronic components to build this hardware. In this chapter we will see what those components are .

Table A.1: Computer board components

Reference	Component
D1	diode 4148
C1	330nF Capacitor
Regulator1	7805
C2	100nF Capacitor
R1	330 Resistor
D2	Red LED diode
JP2	Jumper
JP3	Jumper
J2	Transceiver ER400TRS
P1	Conector DB9
C7	Tantalo 10 uF capacitor
C6	Tantalo 10 uF capacitor
C5	Tantalo 10 uF capacitor
C3	Tantalo 10 uF capacitor
C4	Tantalo 10 uF capacitor
U1	MAX232

Table A.2: Gondola board components

Reference	Component	Package
RS2	3k3 Resistor	0805
RS1	5k6 Resistor	0805
D1	Diode 3A BY251	
C7	Capacitor 0,1uF	1206
C8	0,22uF Capacitor	1206
C9	0,1uF Capacitor	1206
Regulator2	78M05 Voltage Regulator	D-Pack
RL1	330 Resistor	0805
LED1	Yellow LED Diode	0805
C4	0,1uF Capacitor	1206
C5	0,22uF Capacitor	1206
C6	0,1uF Capacitor	1206
Regulator1	78M08 Voltage Regulator	D-Pack
R2	10K Resistor	0805
Q1	BC807 Transistor	
C1	22pF Capacitor	0805
C2	22pF Capacitor	0805
Y1	Cristal 4MHz	
U3	PIC16F873	Dip 28
R1	4k7 Resistor	0805
U2	74HC04	Dip 14
U1	Transceiver	
U4	L293B	dip 16
Fusible1	Fusible 2A N30	
U5	L293B	dip 16
Fusible2	Fusible 2A N30	

---

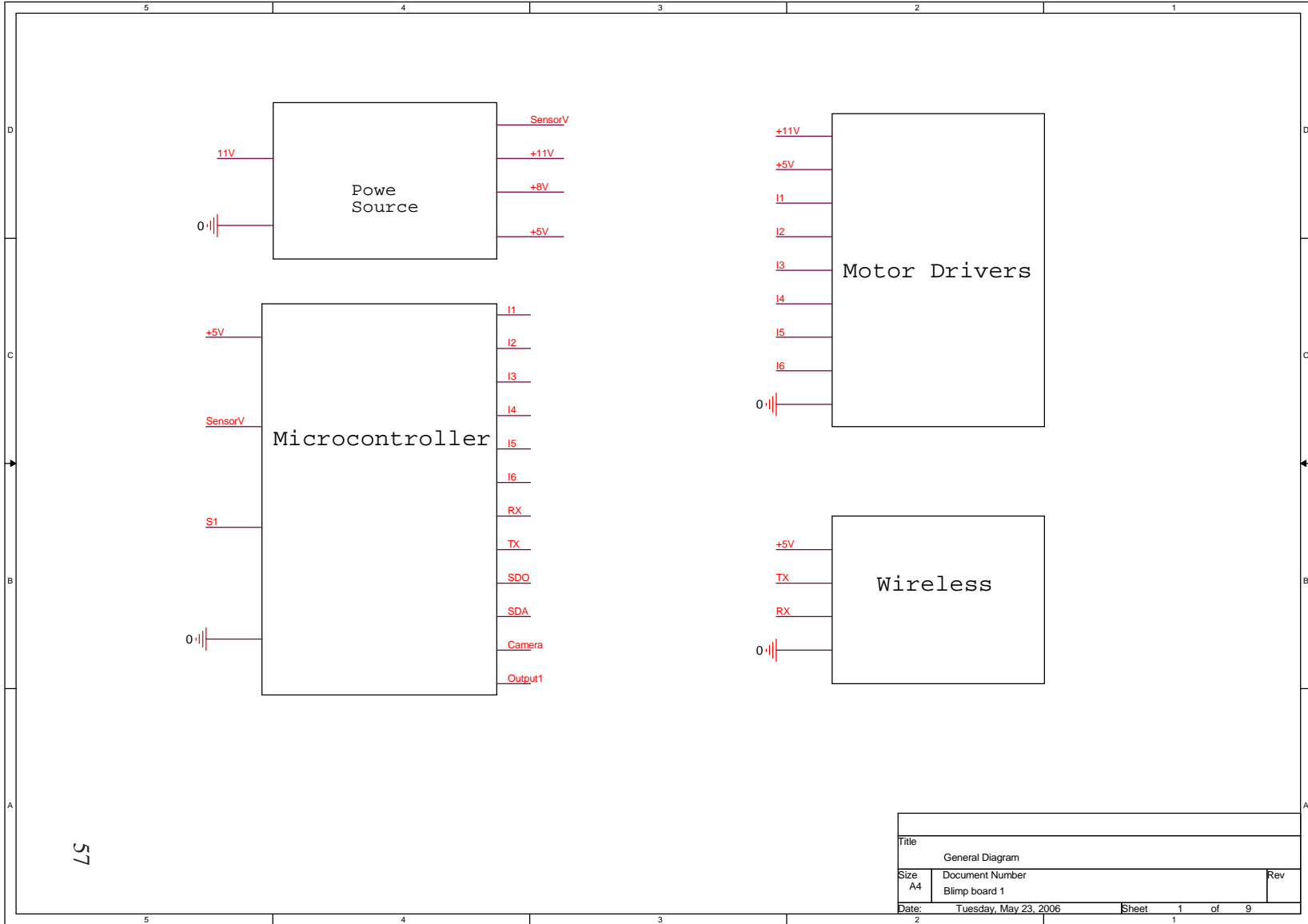
Table A.3: Miscellaneous

<b>Component</b>
LiPo Battery 11V 350mA
2 positions switcher
Servo Futaba Motor
Propeller
Blimp
Ballon Blimp
Ultrasonic sensor SRF05
Ultrasonic sensor SRF10
LiPo saver 9V
LiPo 11V Charger
Cable
Circuit board 0,5mm
Circuit board 1,2mm



## **B. Electronic Board Plans**

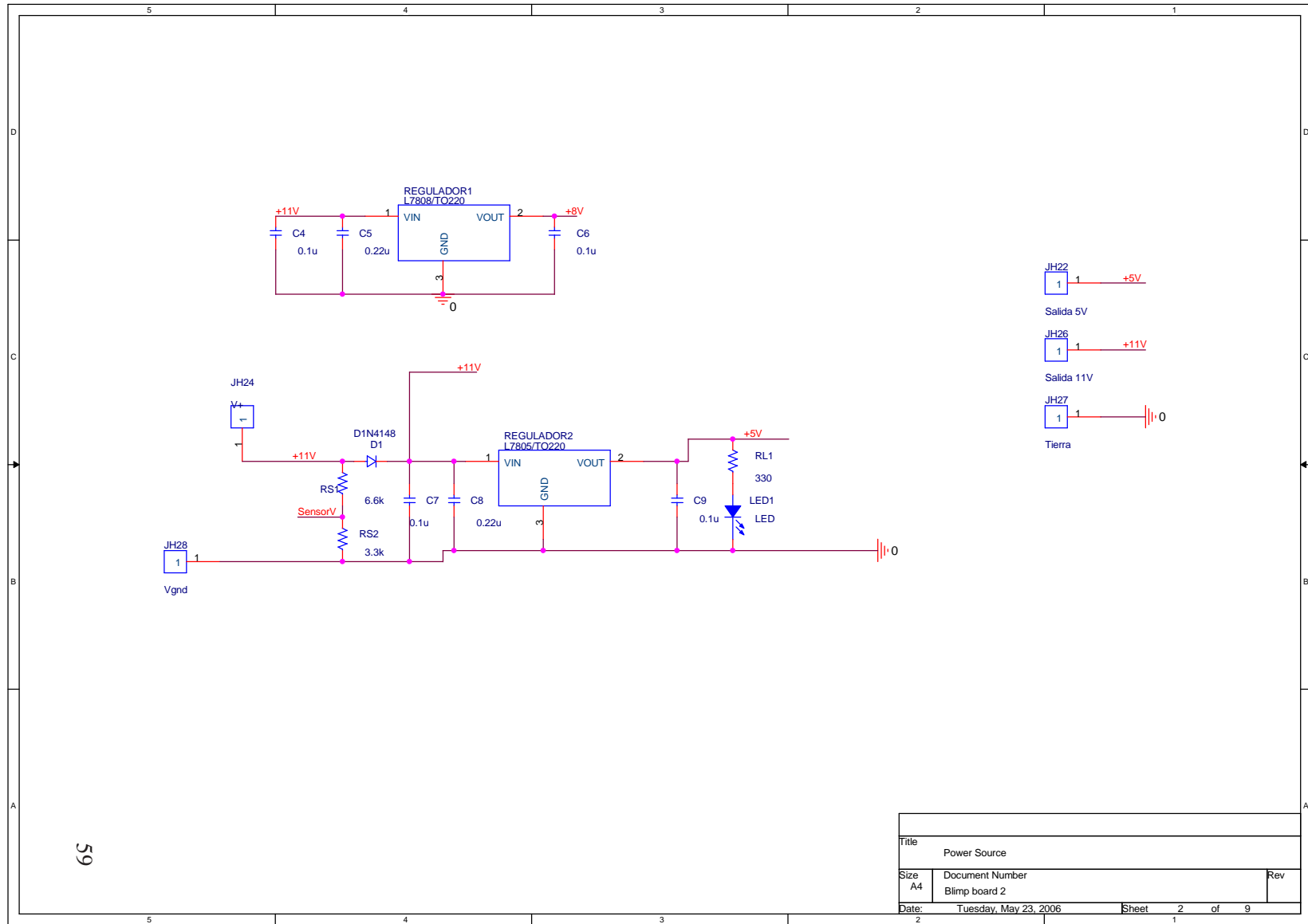




57

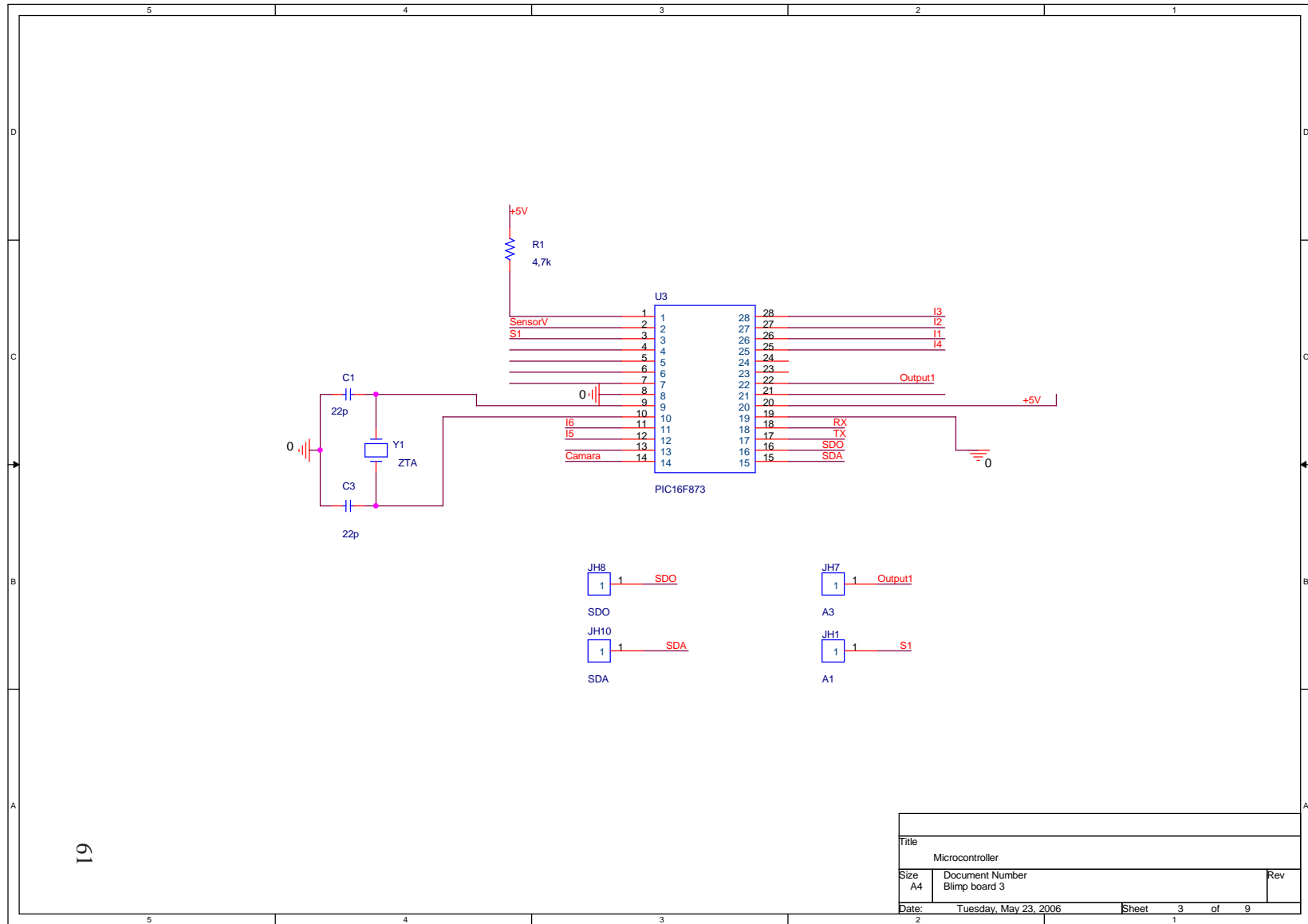
Title		
General Diagram		
Size	Document Number	Rev
A4	Blimp board 1	
Date:	Tuesday, May 23, 2006	Sheet 1 of 9





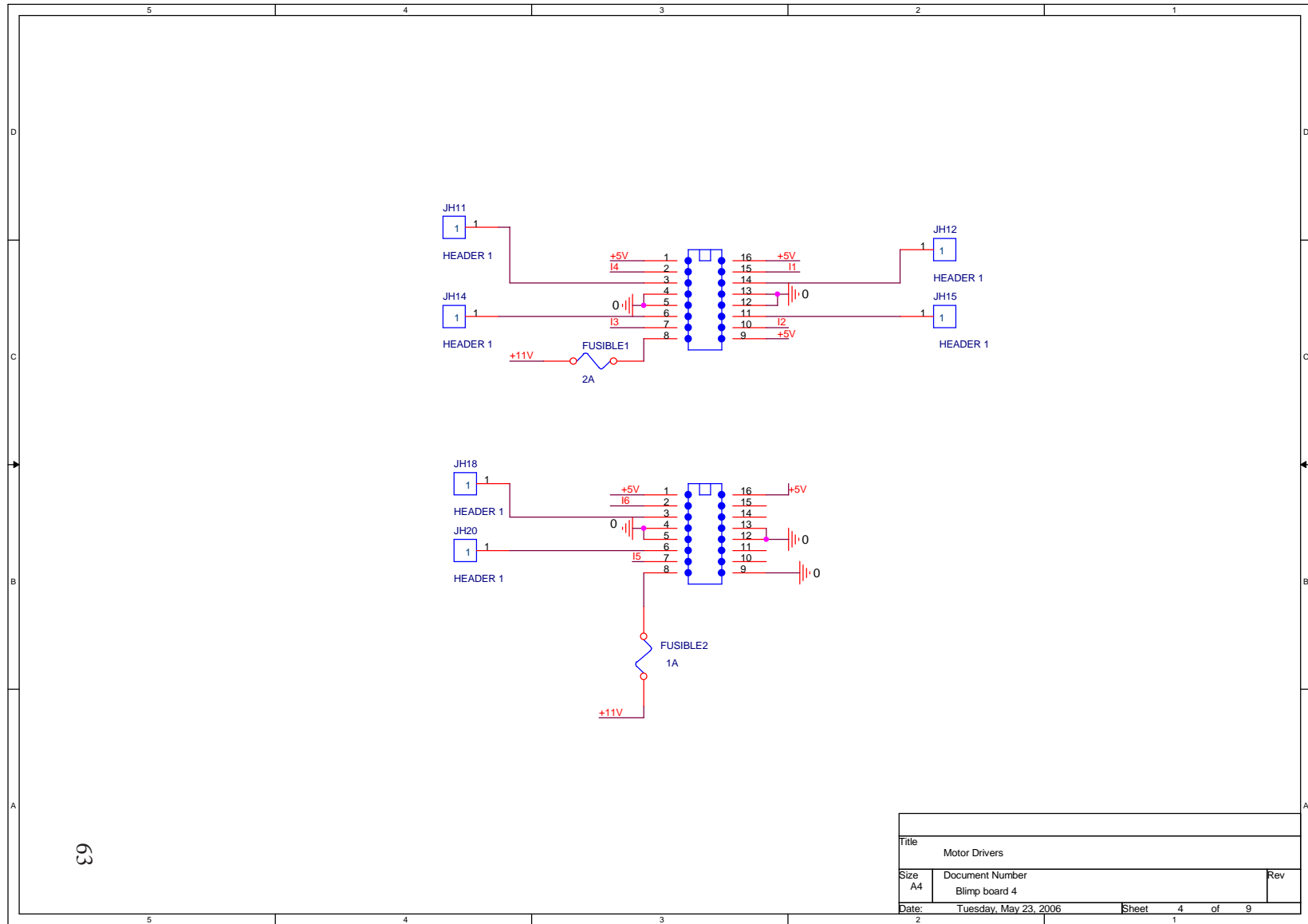
Title		
Power Source		
Size	Document Number	Rev
A4	Blimp board 2	
Date:	Tuesday, May 23, 2006	Sheet 2 of 9





Title		
Microcontroller		
Size	Document Number	Rev
A4	Blimp board 3	
Date:	Tuesday, May 23, 2006	Sheet 3 of 9

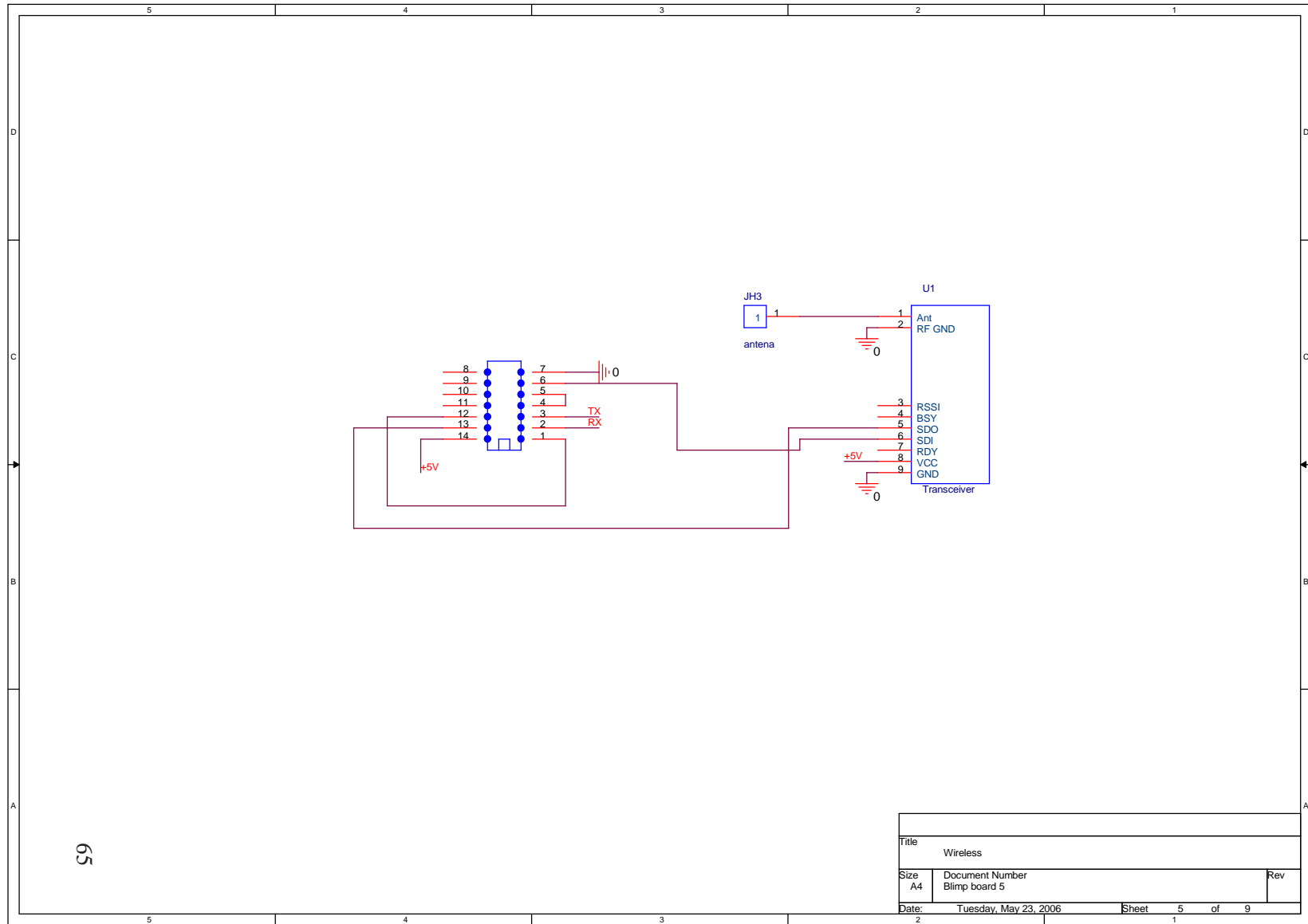




63

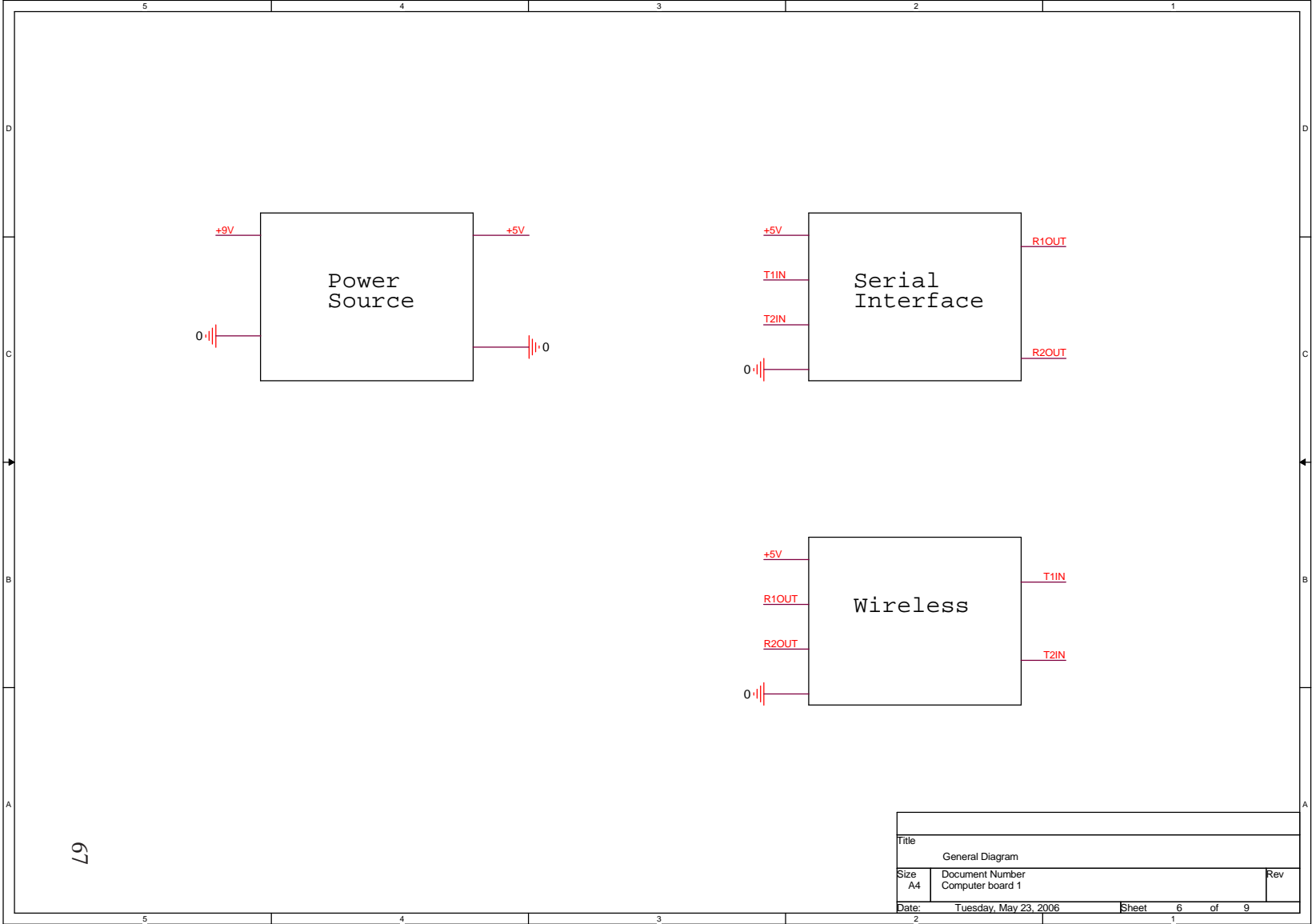
Title		
Motor Drivers		
Size	Document Number	Rev
A4	Blimp board 4	
Date:	Tuesday, May 23, 2006	Sheet 4 of 9





Title		
Wireless		
Size	Document Number	Rev
A4	Blimp board 5	
Date:	Tuesday, May 23, 2006	Sheet 5 of 9

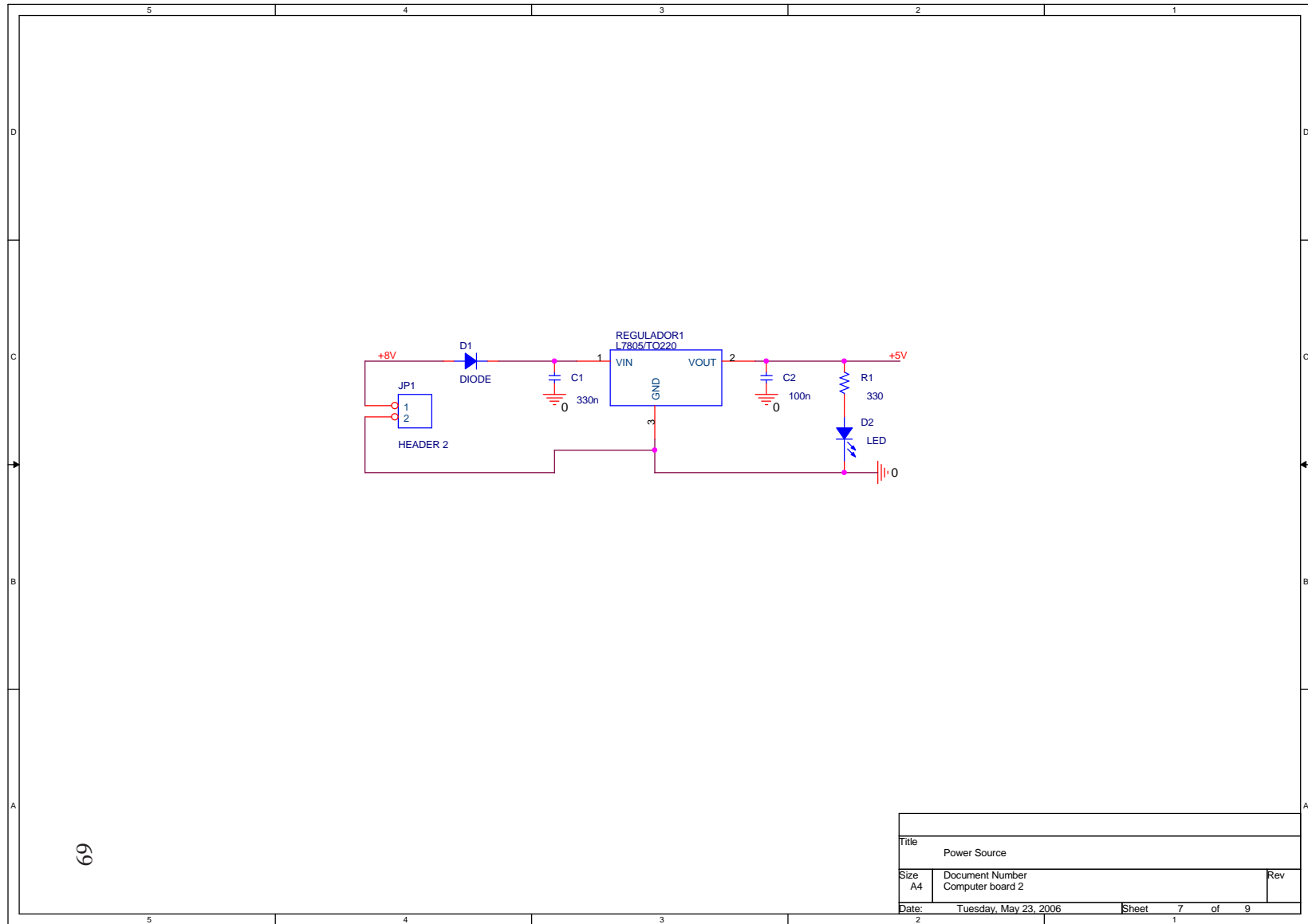




67

Title		
General Diagram		
Size	Document Number	Rev
A4	Computer board 1	
Date:	Tuesday, May 23, 2006	Sheet 6 of 9

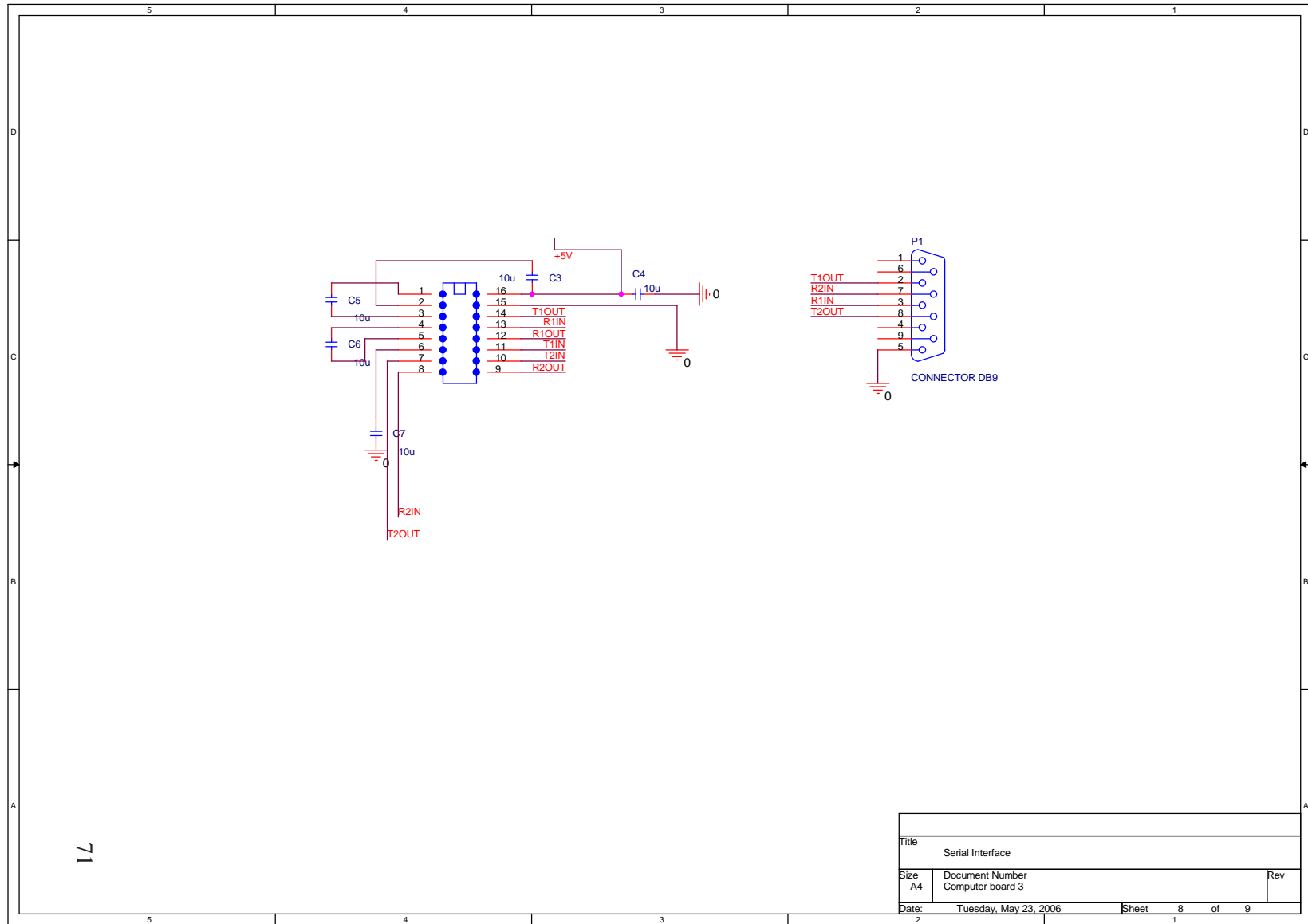




69

Title		
Power Source		
Size	Document Number	Rev
A4	Computer board 2	
Date:	Tuesday, May 23, 2006	Sheet 7 of 9

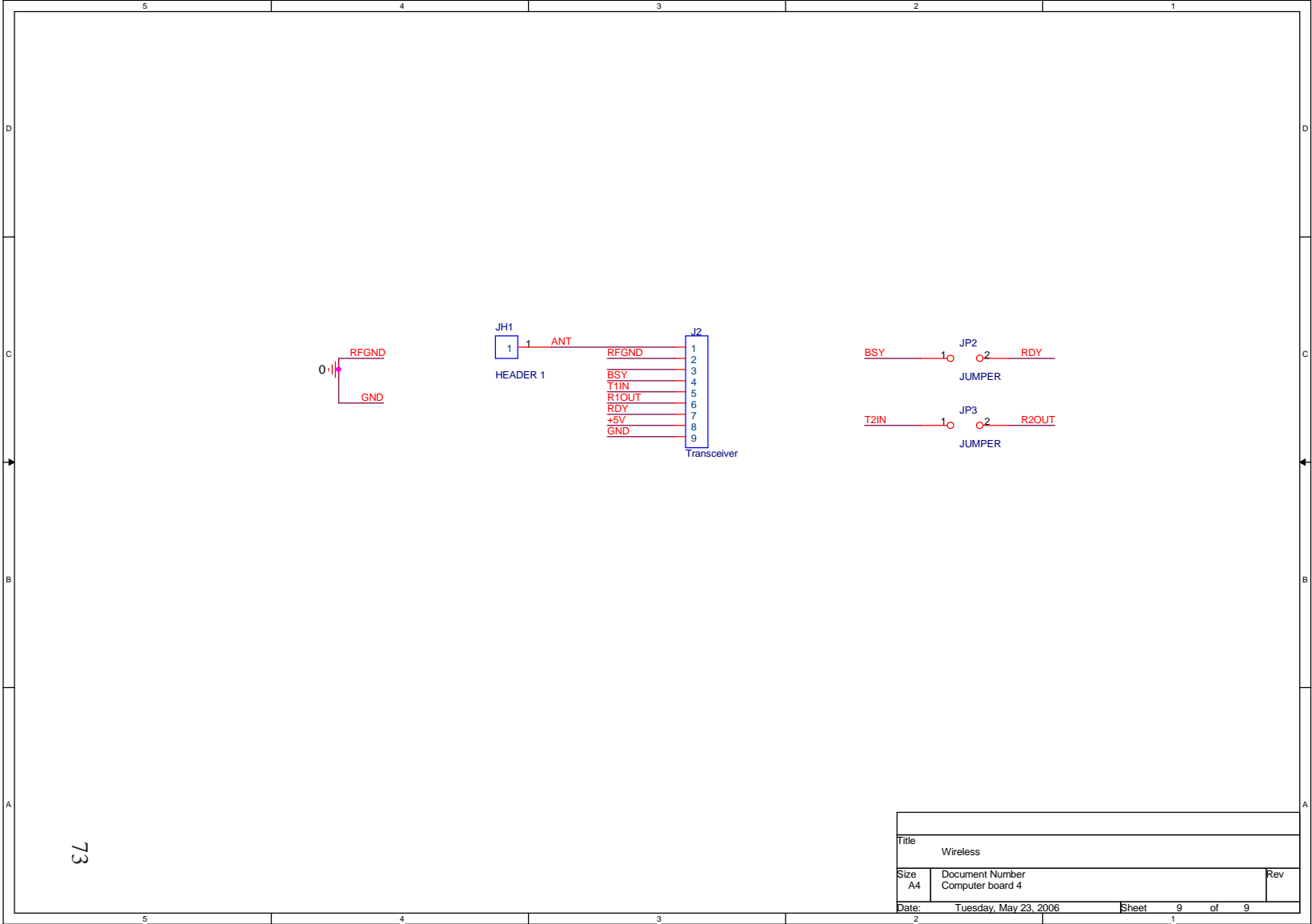




71

Title		
Serial Interface		
Size	Document Number	Rev
A4	Computer board 3	
Date:	Tuesday, May 23, 2006	Sheet 8 of 9





73

Title		
Wireless		
Size	Document Number	Rev
A4	Computer board 4	
Date:	Tuesday, May 23, 2006	Sheet 9 of 9

*B. Electronic Board Plans*

---

# Bibliography

- [1] Aerial robot competition. <http://www.pages.drexel.edu/vn43/main/IARC2006>.
- [2] Balloon shop. <http://www.southernballoonworks.com>.
- [3] Balloon shop. <http://www.heo.com>.
- [4] Blimp shop. <http://j.piri.home.mchsi.com/indoor.htm>.
- [5] Conrad shop. <http://www.conrad.de>.
- [6] Datasheet web page. <http://www.alldatasheet.co.kr/>.
- [7] Microchip web page. <http://www.microchip.com>.
- [8] Micropik shop. <http://www.micropik.com>.
- [9] Picbasic tutorial. <http://www.imagesco.com/catalog/pic/pbpman.pdf>.
- [10] Pid description. <http://www.engin.umich.edu/group/ctm/PID/PID.html>.
- [11] Plantraco shop. <http://www.plantraco.com>.
- [12] Rcguy's shop. <http://www.rcguys.com>.
- [13] Sensors shop. <http://www.roboter-teile.de/Shop>.
- [14] Servos futaba. <http://www.futaba-rc.com>.
- [15] Srf05 datasheet. <http://www.roboter-teile.de/datasheets/srf05.pdf>.
- [16] Srf10 datasheet. <http://www.roboter-teile.de/datasheets/srf10.pdf>.
- [17] Superrobotica shop. <http://www.superrobotica.com>.
- [18] Ziegler-nichols method. <http://www.chem.mtu.edu/tbco/cm416/zn.html>.
- [19] Geoffrey A. Hollinger, Zachary A. Pezzementi, Alexander D. Flurie, and Dr. Bruce A. Maxwell. Design and construction of an indoor robotic blimp for urban search and rescue tasks. 2005.

- [20] Anibal Ollero Baturone. *Control por computador. Descripción interna y diseño optimo.* Marcombo, 1991.
- [21] Peter Baumann. *Serial programming howto.* 2001.
- [22] K. Capek. *R.U.R. (Rossum's Universal Robots).* Dover Publications, 2001.
- [23] George Kantor, David Wettergreen, James P. Ostrowski, and Sanjiv Singh. Collection of environmentat data from an airship platform. *In proceedings SPIE Vol 4571, Sensor Fusion and Decentralized Control in Robotic Systems,* 2001.
- [24] Alberto Elfes, Marcel Bergerman, Jose Reginaldo Hughes Carvalho, Ely Carneiro de Paiva, Josue Jr. Guimaraes Ramos, and Samuel Siqueira Bueno. Air-ground robotic ensembles for cooperative applications: Concepts and preliminary results.
- [25] Juan D. Tardos and Jose Neira. Robust mapping and localization in indoor environments usin sonar data. *MIT,* 2002.
- [26] William E. Green, Keith W. Sevcik and Paul Y. Oh. A competition to identify key challenges for unmanned aerial robots in near-earth environments. 2005.
- [27] Alberto Elfes, Samuel Siquiera Bueno, Marcel Bergerman, and Josue Guimaraes Ramos. A semi-autonomous robotic airship for environmental monitoring missions. *Proceedings of the 1998 IEEE. International Conference on Robotics and Automation,* 1998.
- [28] Keiko Motoyama, Hidenori Kawamura, Masahito Yamamoto, and Azuma Ohuchi. Design of evaluation function in motion planning for autonomous balloon robot. *Proceedings of 2003 Asia Pacific Symposium on Intelligent and Evolucionary Systems,* 2003.
- [29] Keiko Motoyama, Hidenori Kawamura, Masahito Yamamoto, and Azuma Ohuchi. Development of autonomous blimp with intelligent control. *Entertainment Computing/Technologies and Applications,* 2003.
- [30] Peter S. Maybeck. *Stochastic models, estimation, and contro.* Academic Press, 1979.
- [31] Emmanuel Hygounenc, Il-Kyun Jung, Philippe Soueres and Simon Lacroix. The autonomous blimp project of laas-cnrs: Achievements in flight control and terrain mapping. *LAAS/CNRS.*
- [32] Alvaro Solera Ramirez. *El filtro de kalman.* 2003.
- [33] Raul Reyero and Carlos F. Nicolas. *Sistemas de control basados en logica borrosa: "Fuzzy Control".* OMRON, 1995.

- [34] Hisao Kadota, Hidenori Kawamura, Masahito Yamamoto, Toshihiko Takaya, and Azuma Ohuchi. Vision-based positioning system for indoor blimp robot. *Proceedings of The 4th International Conference on Advanced Mechatronics -Toward Evolutionary Fusion of IT and Mechatronics*, 2004.
- [35] Greg Welch and Gary Bishop. An introduction to the kalman filter. *University of North Carolina at Chapel Hill*, 2002.
- [36] Keith W. Sevcik, William E. Green and Paul Y. Oh. Exploring search-and-rescue in near-earth environments for aerial robots. *Drexel University, Philadelphia, PA*.
- [37] Sebastian Thrun, Wolfram Burgard and Dieter Fox. *Probabilistic Robotics*. MIT-Press, 2005.