Intelligent Sensing Anywhere

Faculdade de Ciências e Tecnologia da Universidade de Coimbra

EasyBreathing@Home

Remote Vital Signs Monitoring

Specific Application to Respiratory Failure

Neuza Pimentel Aguiar

Master of Biomedical Engineering

Coimbra, Septembre 2009

Intelligent Sensing Anywhere

Faculdade de Ciências e
Tecnologia da Universidade
de Coimbra

EasyBreathing@Home

Remote Vital Signs Monitoring
Specific Application to Respiratory Failure

Acquisition and Processing Module

Neuza Pimentel Aguiar

Student Number: 2004107364

A thesis submitted for the degree of
Master of Biomedical Engineering

Coimbra, September 2009

ABSTRACT

Nowadays is of great interest for medical care to reduce costs and improve the health status of patients. Thus, several researches and developments have been made in the area of technology in the context of allowing an improvement in medical systems, portable systems capable of monitoring patients continuously in order to provide greater mobility and flexibility not only to patients but also less work to the health professional.

The Chronic Obstructive Pulmonary Disease (COPD), chronic respiratory failure, is considered by the World Health Organization a public health's problem. For many patients suffering from hypoxemia the Long-Term Oxygen Therapy (LTOT) at home is a common treatment.

Currently, it is not available in the market any device which could respond to a patient's real needs because the oxygen's delivery is regulated manually. So, in order to overcome this, studies have shown that the automatic control of oxygen delivery is more effective in maintaining levels of SpO_2 at an optimal range.

The portable device proposed includes two principal modules: a module for the acquisition and processing of physiological signs (oximetry and accelerometry) and an actuation module (actuation of the electronic proportional valve). The system adjusts the oxygen flow automatically to a patient based on his SpO_2 and accelerometry readings, regulating the opening of a proportional valve according to the patient needs. This allows better match of the treatment to the patient's real needs and greater oxygen's saving, avoiding its waste.

RESUMO

Hoje em dia é de grande interesse, relativamente aos cuidados médicos, reduzir os custos associados e melhorar o estado de saúde dos pacientes. Deste modo, diversas pesquisas e evoluções têm sido efectuadas na área da tecnologia no âmbito de permitir um melhoramento nos sistemas médicos, surgindo sistemas portáteis capazes de monitorizar pacientes continuamente de forma a conferir uma maior mobilidade e flexibilidade não só aos pacientes mas reduzir também o trabalho dos profissionais de saúde.

A DPOC (doença pulmonar obstructiva crónica) é considerada pela Organização Mundial de Saúde um problema de saúde pública. Para muitos pacientes que vivem com hipoxemia a ODP (oxigenoterapia domiciliar prolongada) em casa é um tratamento muito comum.

Actualmente, não está disponível no mercado qualquer dispositivo que possa corresponder às necessidades reais do paciente, uma vez que o débito de oxigénio é regulado manualmente e modo contínuo. De modo a ultrapassar este obstáculo, foram feitos estudos que mostram que o controlo automático de fornecimento de oxigénio é mais eficaz em manter os níveis de SpO_2 numa gama óptima.

O dispositivo portátil proposto é constituído por dois módulos principais: um módulo de aquisição dos sinais fisiológicos (oximetria e acelerometria) e processamento dos mesmos e um módulo de actuação de hardware (actuação da válvula proporcional electrónica). Deste modo, o sistema regula automaticamente o fluxo de oxigénio para um paciente com base nos dados de SpO_2 e acelerometria, regulando a abertura da válvula proporcional de acordo com a necessidade do paciente. Isto permite uma melhor adequação do tratamento às necessidades bem como possibilitar uma maior poupança de oxigénio, evitando assim o seu desperdício.

ACKNOWLEDGMENTS

The development of this master's thesis involved the collaboration of several people who I would like to highlight and thank.

First, I have to thank the Engº José Malaquias, Engª Catarina Pereira and Engº Eduardo Faria who helped me integrate the team of ISA Health Care and allowed to start the development of this project.

I have a special thanks to Engª Soraia Rocha and Engº Nuno Varelas that always helped me in developing this project, for their ideas, effort to guide and their valuable advices. For their help in critical times, for believe the future of this project and for their contribution to my growth professionally.

I’m thankful to my project colleague who directly participated with me in this project, for the good work environment and companionship, the support and help that she gave me with some ideas discussion and suggestions.

I thank all my friends that in a direct or indirect way always supported me and helped me to be optimistic and achieve my goals. A special thanks to André, João and Rita.

To my dear parents a huge thanks for helping me to be who I am today and believed that I would be able to overcome all obstacles. Thanks for the motivation, understanding and strength that always accompanied me.

A very special thanks to my boyfriend Hugo, for being special and for always encouraging me, giving me strength in moments of discouragement and making me believe that everything is possible.

To my family, boyfriend and friends

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS AND DEFINITIONS

Acronyms	Definition
A/D	Analog/Digital
bps	Bits por segundo
BRG	Baud-Rate Generator
CEI	Centro de Electrónica e Instrumentação (Electronics and Instrumentation Centre)
CHC	Centro Hospitalar de Coimbra (Hospitalar Centre of Coimbra)
COPD	Chronic obstructive pulmonary disease
CPU	Central Processing Unit
DCE	Data Communication Equipment
DODS	Oxygen Demand Delivery System
DPG	2, 3 Diphosphoglycerate
DPOC	Doença Pulmonar Obstructiva Crónica
DTE	Data Terminal Equipment
EEPROM	Electrically Erasable Programmable Read-Only
EIA	Electronic Industries Association
FLASH	Non-volatile memory
Hb	Hemoglobin
HbO_2	Oxyhemoglobin
I/O	Input/output
ICD	In-Circuit Debugger
IDE	Integrated Development Environment
ISA	Intelligent Sensing Anywhere
LED	Light Emitting Diode

lpm	Liters per minute
LTOT	Long-Term Oxygen Therapy
LSB	Least Significant Bit
MSB	Most Significant Bit
H^+	Acid
NRZ	Non-Return-to-Zero
O_2	Oxygen
ODP	Oxigenoterapia Domiciliar Prolongada
OEM	Original Equipment Manufactured
$PaCO_2$	Arterial pressure of carbon dioxide
PaO_2	Arterial pressure of oxygen
PIC	Peripheral Interface Controller
PLL	Phase-Locked Loop
psig	Pound-force per square inch
PWM	Pulse Width Modulation
RAM	Random Access Memory
REM	Rapid Eye Movements
RS-232	Recommended Standard 232
RS-485	Recommended Standard 485
RX	Reception
SpO_2	Saturation of peripheral Oxygen
TTL	Transistor-to-Transistor Logic
TX	Transmission
UART	Universal Asynchronous Receiver Transmitter
USB	Universal Serial Bus
W/R	Write/Read

Chapter 1: Introduction

1.1. Motivation

The respiratory failure is defined by reducing the partial pressure of O_2 in arterial blood (PaO_2) with or without an increase in $PaCO_2$. The PaO_2 was correlated with the turn in O_2 saturation of hemoglobin measured by pulse oximetry (SpO_2). The chronic respiratory failure is the final phase of a wide range of chronic pulmonary diseases, with emphasis on chronic obstructive pulmonary disease (COPD).

The World Health Organization considers the COPD a public health's problem. COPD is a chronic, progressive and irreversible disease that affects the respiratory system (lungs, bronchioles, alveolar tissue). Disease like chronic bronchitis, asthma and emphysema are considered chronic obstructive pulmonary disease. The main symptoms of the patients with COPD are the airflow limitation, muscle fatigue, dyspnea, among others. Dyspnea is typically the first symptom, it is a subjective feeling of "shortness of breath". Over the years, dyspnea has worsen gradually, it has developed slowly and progressively, leading, sometimes, to major diseases [1].

In Portugal, the National Center for Respiratory Diseases estimates that more than 500 thousand people suffer COPD, a number that is still underestimated, due to the high number of cases misdiagnosed and others for diagnosing. The COPD in Portugal is the 7th cause of death, representing 2.5% of all deaths annually, while in other European countries is 3.6%. At worldwide level this disease is the 6th leading cause of death and tended to increase at least until 2020 to become the 3rd cause of death [2].

The Long-Term Oxygen Therapy (LTOT) at home is a common treatment for many patients suffering from hypoxemia, in which the oxygen is used for periods exceeding 15 hours a day. It has been shown to be the only treatment which prolongs the life of patients suffering from chronic lung disease [3].

In fact, thousands of patients around the world benefit from this type of treatment, either in hospital or at home; this treatment aims to minimize the negative consequences of hypoxia, by reducing the pressure in the pulmonary artery and cardiac arrhythmias and also by improving the quality of sleep, promoting quality and increasing life expectancy of patients [4].

Nowadays, the oxygen flow rate in these treatments is adjusted manually, it is not available any device that can meet the needs of the patient. Moreover, the delivery of oxygen is continuous and constant, so, during the day and for low levels of activity the debit is 2 lpm, and at night it suffers an increase of 1 lpm, so it becomes 3 lpm. This is due to stage REM sleep, characterized by intense brain activity, with respiratory, heart rate and blood pressure similar to a state of alert, causing a decrease of oxygen saturation requiring an increase of the oxygen delivery [4].

Moreover, physical exercise is highly recommended for patients suffering respiratory diseases. This is, it allows to develop the physiology of the respiratory system, it improves the mechanism of gas exchange and consequently increases the oxygenation of the blood, avoiding a rigorous and long-term oxygen therapy.

1.2. Objectives

According to what has been said above, it is necessary to develop a system that allows the patients to keep more time at optimal oxygen saturation level. Studies have shown that the automatic control of oxygen delivery is more effective than treatments adjusted manually by either the patient or caregiver. Today, no device is available to automatically regulate oxygen flow to a patient.

So, the project EasyBreathing@Home aim is to develop a portable device that monitors continuously the patient’s oxygen saturation levels and movement during his daily activities. Signals acquisition must be made by a signal acquisition module using wearable sensors as a pulse oximetry and accelerometry. After the data’s acquisition, the system should process them and, according to a clinical protocol previously defined, the device automatically adjusts a valve’s diameter which regulates the flow of oxygen to the patient needs in each moment.

Besides all this functions, this device must include several features. It has to be functional in a home setting; this is, the device should be safe, easy to use, reliable, portable and it must be a low cost product.

In addition to the system development, the knowledge acquisition is required to perform all the tasks proposed to achieve the goal of this project. During its development, the study resulted in a positive experience for best results.

1.3. Audience

This thesis has for main audience jury members and supervisors of the project in course. It is a result of a knowledge acquisition, research, process and decisions that were made during the project development. So it is essential for advertise the project's scope and its objectives.

1.4. Document Structure

The present chapter intents to advise the reader about the scope of the project, the objectives proposed, the document structure and the target audience.

The second chapter – Project Management – presents the way the project was organized such as the project members, tasks division and the project supervising.

The following chapter reports the problem analysis, the state of the art and the advantages that the pretended solution may offer when compared with the already existing.

The fourth chapter contains the most of the knowledge acquired during project's execution. So, in this chapter the reader is provided with theoretical background that allows understanding most of the thesis' technical concepts.

In the Chapter 5 will be discussed the project architecture, more specifically, the project requirements such as the scenery application of this system and the medical monitoring equipment indispensable for vital signs acquisition; the acquisition module and the hardware's actuation module.

On the 6th chapter it is presented in short the configuration of the microcontroller's module firmware acquisition and filtering of the vital signs obtained by pulse oximetry and the firmware for processing these data.

Chapter 7 consists on the results obtained while making several tests *in loco-the hospital,* after the technical conclusion.

The last chapter concludes the present project report and refers to both the project's status and an analysis of possible future developments. A student's personal appreciation regarding the project is also presented with the intent of final conclusion.

Chapter 2. Project Management

2.1. Project Members

The Remote Vital Signs Monitoring – Specific Application to Respiratory Failure project was developed by various elements, including two Biomedical Engineering students from the Faculty of Sciences and Technology - University of Coimbra who were responsible for the projection and development of the system and their supervisors who were responsible for the coordination of the project and orientation of the student's work.

The following table presents the project members of this project.

Name	Role	Contact
Neuza Aguiar	Student	naguiar@isa.pt
Patrícia Borges	Student	pborges@isa.pt
Professor Carlos Correia	Coordinator	correia@lei.fis.uc.pt
Professor José Basílio Simões	Coordinator	jbasilio@lei.fis.uc.pt
Engineer Soraia Rocha	Supervisor	srocha@isa.pt
Engineer Nuno Varelas	Supervisor	nvarelas@isa.pt
Doctor Joaquim Moita	Supervisor	joaquimmoita@chc.min-saude.pt

Table 2.1. Project Members.

2.2. Project Supervising

There were three entities involved in the development of this project: ISA (Intelligent Sensing Anywhere), CEI (*Centro de Electrónica e Instrumentação*) and CHC (*Centro Hospitalar de Coimbra*).

2.2.1. Supervising at ISA

ISA was founded in 1990 as a spinoff company of the Department of Physics – University of Coimbra. This company is global leader in different market segments exporting products, applications and solutions in the fields of electronics, software development, telemetry, industrial automation, environment and healthcare.

The supervising at ISA was made by Engª Soraia Rocha and Engº Nuno Varelas. Initially, several meetings were made in order to discuss the project evolution as well as the system's requirements and architecture.

2.2.2. Supervising at CEI

The CEI has been created in the Physics Department of the University of Coimbra. The members of the CEI work on a wide range of research areas such as Atomic and Nuclear Instrumentation, Biomedical Instrumentation, Plasma Physics Instrumentation, Microelectronics, Optical Signal Processing and Telemetry and Industrial Control.

The supervision at CEI was made by Professor Carlos Correia. Almost half of the project's execution time was spent in CEI's installations, where the students could attain the knowledge for the development of the project.

2.2.3. Supervising at CHC

The CHC's Department of Pulmonology provides assistance in the internment area and external consultation, it is also important the sectors for the implementation of additional tests, particularly in the field of broncology, functional respiratory exploration, study of the pathology of sleep and the activity of research, especially clinical.

The supervision at CHC was made by Doctor Joaquim Moita. He helped the students to understand better the difficulties of the patients with COPD, the treatment that they are subject to and what to do in Pulmonology area in order to improve the patient's life quality.

2.3. Project Planning

This project was realized over one curriculum year; it required a planning of the main stages and tasks that the students had to carry on. So, it was necessary to identify the methods to achieve the objectives proposed.

At the first and second phases the two students have acquired knowledge necessary for the project's development and they made some research to purchase the proportional valve and the oximeter adequate. During the third phase the tasks were divided into two modules. Neuza was responsible for the pulse oximeter integration and its signal acquisition and processing, while Patricia was responsible for the proportional valve and the driver board's integration and actuation. The following table shows how the division was made.

Name	Project's Module	Contact
Neuza Aguiar	Vital Signs Acquisition Module	Firmware design of the Acquisition and Processing Module
Patrícia Borges	Hardware Actuation Module	Firmware design of the Hardware Actuation Module

Table 2.2. Tasks Assignment.

The tasks performed during each phase are presented in the following Gantt-charts. This planning was done during the year by the project members and their supervisors.

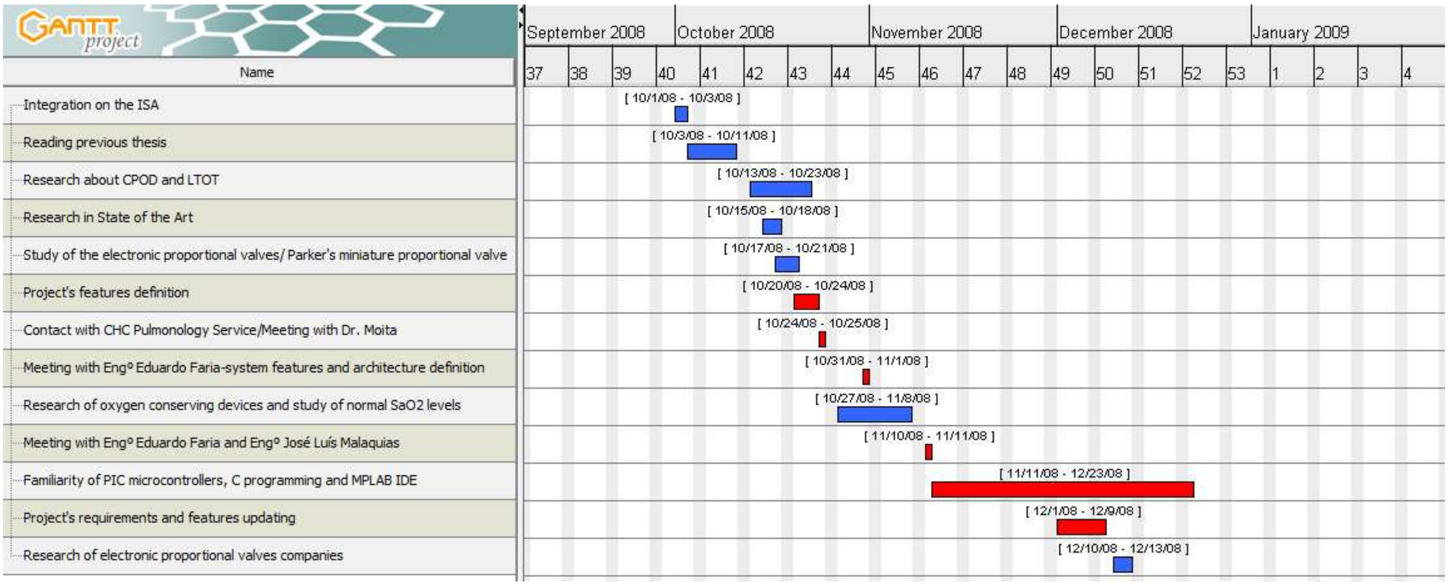


Table 2.3 Tasks performed at the first phase: Gantt-chart.

Tasks carried out by Neuza

Tasks carried out by Neuza and Patrícia

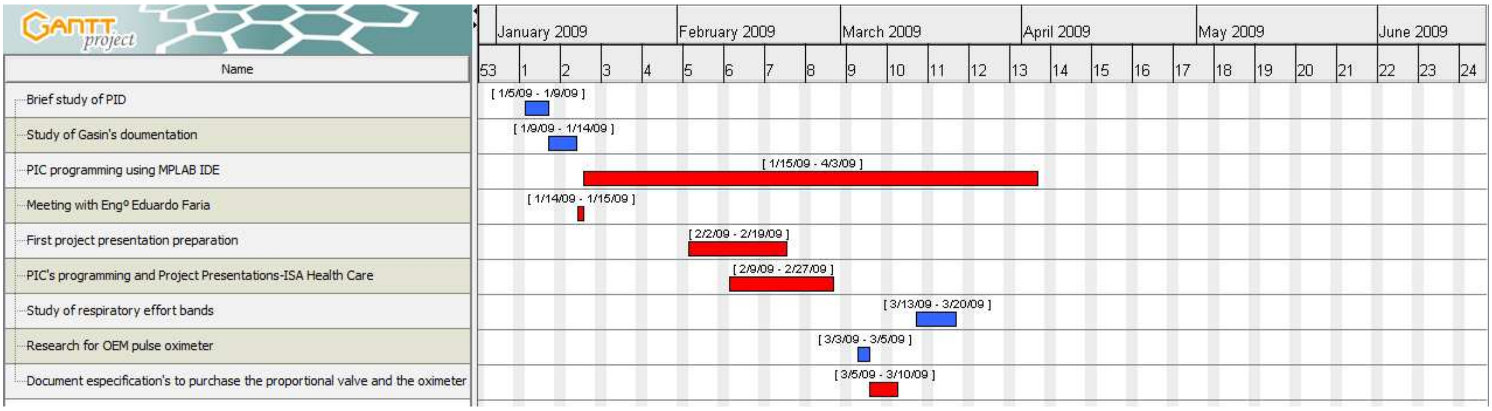


Table 2.4. Tasks performed at the second phase: Gantt-chart

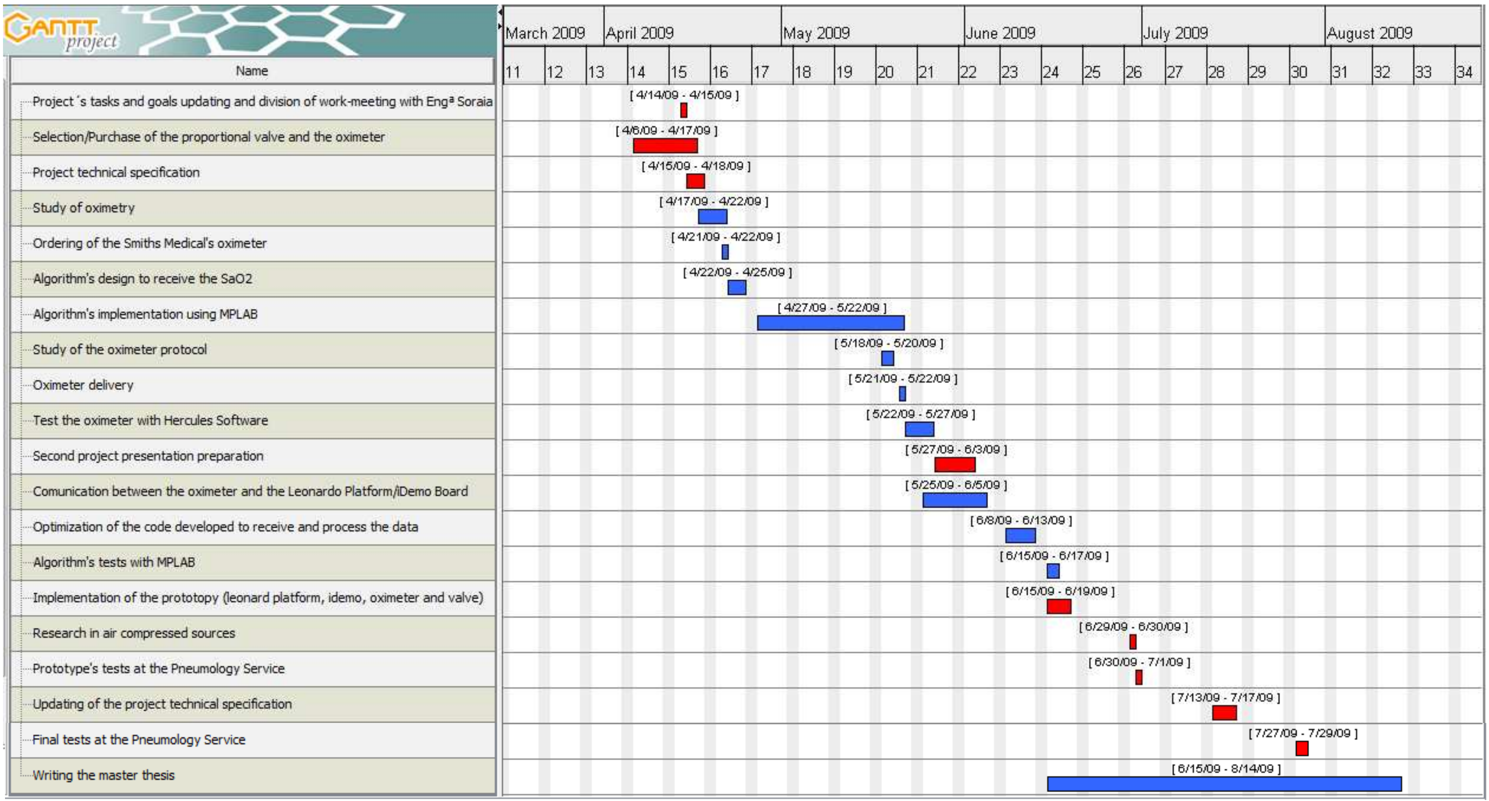


Table 2.5. Tasks performed at the third phase: Gantt-chart.

Chapter 3. Related Works

3.1. Problem Analysis

As previously said, the currently treatment options (LTOT) for diseases like CPOD include a permanent supply of oxygen that, besides being very expensive, is very limited in terms of the autonomy of the patient. This limitation in the patient's mobility is concerned mainly with the currently available sources of O_2. Associated with the problem of autonomy it is the control of rates of oxygen. The rates vary inter and intra-individually, i.e. in a particular patient, every time, depending on the state of health, level of activity, sleep or particular situations.

Nowadays, the treatment of patients, under home oxygen therapy, is prescribed in accordance with the data saturation collected in the hospital or clinic for several tests. However, it has been shown that these data do not respond effectively to the needs of the patient in his normal activities. A particular study showed that patients experienced several events of desaturation during a standard oxygen prescription [5]. Thus, it is difficult for the health care providers prescribe one a standard treatment, since there is no specific treatment that can ensure full coverage of the patient's needs. Another study showed that many patients are treated with higher flow rates than necessary (e.g. patients at home) and some lower flow rates than would be optimal [6].

To overcome these problems it was proposed a portable device that adjusts automatically the oxygen flow according to the data saturation obtained of the patient during his daily activities.

3.2. State of the Art

Historically, due to the simplicity of the continuous oxygen delivery, this has been a widely used method. It was first administered oxygen in hospitals, where the oxygen flow allowed a low state of patients' hypoxia and their distribution was not a problem. Since the cost of delivery of oxygen inside the hospital was not concern, it was not investigated techniques for lightweight, long-lasting, cost-effective of this method. Over the years, this concern has earned respect, thus, with the aim of making health care less expensive, many patients (COPD) went to treatment at home. With this fact, the cost, availability and capacity of oxygen delivery systems have become issues of debate and research to clinicians, providers and patients. Increasingly, the development of devices for the oxygen therapy is carried out with efficiency and effectiveness [7]. Therefore, there are devices capable of maintaining a level of patient's oxygenation suitable for various activity levels. This oxygenation is carried out intermittently allowing a saving of oxygen while keeping the patient in a state of complete harmony.

The following chapter pretends to summarize the oxygen supply systems and the oxygen conserving devices that currently exist for delivery of oxygen to patients suffering from respiratory failure.

3.2.1. Oxygen Supply Systems

Currently, the LTOT at home uses one of three types of oxygen supply systems: oxygen concentrators, oxygen cylinders liquid or gaseous.

3.2.1.1. Oxygen Concentrators

Oxygen concentrators appeared in the early 1970s [8]. This is the most economical to use oxygen at home. Despite the extra expense in energy, the concentrators are still much cheaper than the cylinders of oxygen. For the individual consumer, the monthly cost of the concentrator is equivalent to 25% of the cost of the cylinder [3].

This machine separates the oxygen from nitrogen from ambient air through a series of filters and concentrates on providing the flow of O_2 from 1 to 5 lpm. It is a lightweight unit with wheel base, comprising motor and battery. Their advantage is that they work independently of other systems (unlike gas cylinders they do not need to be refilled periodically from a base unit). However, they need to be connected to the main electricity supply, a major drawback that limits patient mobility [8].

In general an oxygen concentrator has an on/off switch to start and stop the oxygen concentrator, a flow adjusting knob that gives more or less oxygen, a tubing to carry the oxygen to the patient and a humidifier bottle to mix a mist of sterile distilled water with the oxygen [9].

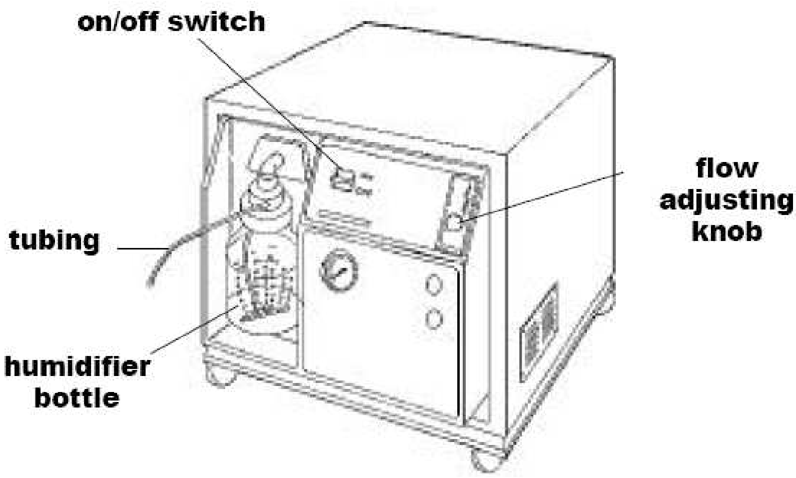


Figure 3.1. Oxygen concentrator. From [9]

3.2.1.2. Oxygen Cylinders Liquid

Liquid oxygen was first used for home oxygen therapy purposes in 1965 [8]. The liquid oxygen is light and it may be stored at home in containers under low pressures and it is possible to make your home refueling in smaller units and portable.

When the liquid oxygen goes out of the matrix immediately turns into gas, so, it is easily stored in the laptop bag, allowing greater mobility to the patient [3].

Liquid oxygen systems have been shown to improve quality of life for patients requiring home oxygen therapy by affording them greater mobility and independence in activities of daily living at home and in the community and ensuring adequate oxygen saturation.

This oxygen supply system presents a reservoir that stores liquid oxygen, a liquid level gauge to show how much oxygen is left in the reservoir, a pressure gauge button that turns on the liquid level gauge, a tubing to carry the oxygen to the patient, a humidifier bottle to mix a mist of distilled water with the oxygen, a flow adjusting knob that gives more or less oxygen and it shows how many liters of oxygen will be released by the tank in each minute [9].

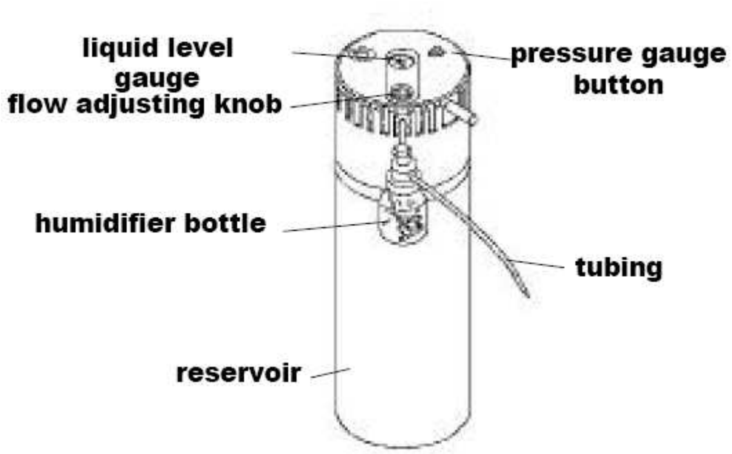


Figure 3.2. Large reservoir with oxygen liquid. From [9]

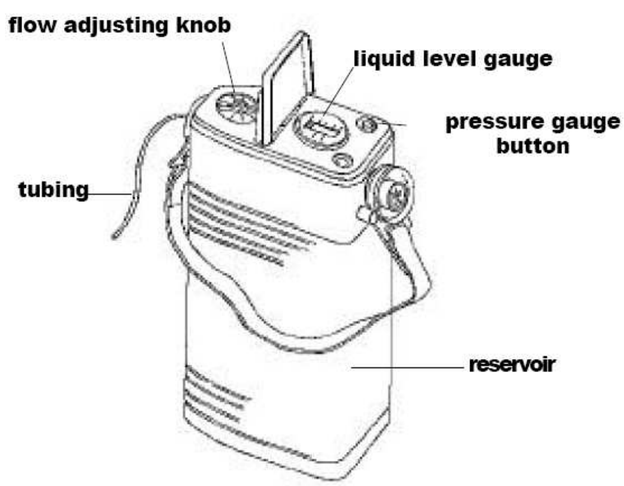


Figure 3.3. Small and portable reservoir to refill oxygen from the larger reservoir. From [9]

3.2.1.3. Oxygen Cylinders Gaseous

The oxygen cylinder of compressed gas under pressure was the first delivery systems used. Though still widely used, they have a major drawback: their size and weight. Although this problem has been partly overcome with the introduction of portable models (small cylinders of 600ml), gas cylinders can seriously hinder patient mobility, even around the home [8].

These systems have built the equipment for the control of pressure, flow and monitoring of the oxygen available. The pressure gauge shows how much oxygen is left in the tank in pounds per square inch (psi). The flow adjusting knob regulates the gas flow rate in lpm and the regulator or flow meter controls how much oxygen the patient is getting. There are two types of regulators that can be used on tank oxygen: the dial regulator and the ball regulator.

Beyond these components, this system also presents a metal tank that stays in the house, where the oxygen is stored as a gas under high pressure, a tubing to carry oxygen to the patient, an on/off valve to starts and stop the oxygen flow, a humidifier bottle and a safety stand to keep the tank standing up.

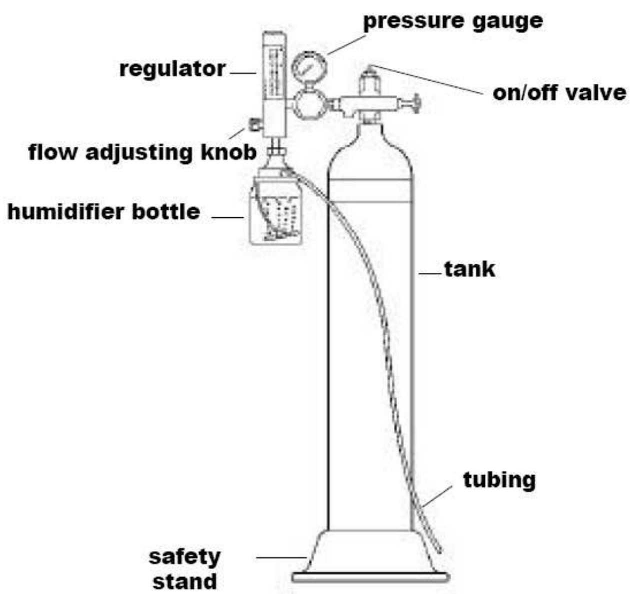


Figure 3.4. Oxygen cylinders gaseous reservoir. From [9]

The following table shows the characteristics of each oxygen suppliers systems.

	Oxygen Concentrator	Oxygen cylinders gaseous	Oxygen cylinders liquid
Indications	Patients with restricted mobility and low oxygen flow requirements.	Complements stationary systems and allows patients to be mobile.	Mobile patients
Advantages	Unlimited volume of gas; Easy to use.	Allows patients to move if it is portable cylinders.	Allows patients to move; Satisfactory duration; Refillable from base unit.
Disadvantages	Maximum flow limited to 5 L / min; Require electricity supply; Are not portable.	Heavy and large; Dangerous, can not suffer loss; Not refillable.	Risk of burns during recharge (stored below zero ºC)
Cost	Low	Medium	High

Table 3.1. Main Characteristics of oxygen sources.

3.2.2. Oxygen Conserving Devices

The oxygen-conserving devices appeared in the mid-1980s in order to increase the duration of oxygen cylinders or reduce the energy consumption of electric concentrators. These, as the name suggests, are intended to economize oxygen while the patients are properly oxygenated, offering increased mobility with improved comfort and efficiency [7].

Nowadays there are three different types of oxygen conserving devices such as a nasal cannula with reservoir, the catheter transtracheal and Oxygen Demand Delivery Systems (DODS). Each method attempts to reduce oxygen wastage during exhalation and each has its own advantages and drawbacks.

3.2.2.1. Reservoir Cannula

To improve the efficacy of standard nasal cannulas it was introduced in the mid-1980s the reservoir cannulas [8]. The oxygen conserver cannula consists of nasal prongs and an attached reservoir. The reservoir has a membrane that is activated when the patient exhales, in order to store the oxygen (that is usually wasted) inside. This oxygen is delivered at the beginning of inhalation – the optimal time to deliver supplemental oxygen – and the patient inspires almost pure oxygen [10]. This fact allows the healthcare professional to reduce the oxygen flow rates required to obtain target oxygen saturation.

Two types of these conserving devices are available. Both of these two configurations have the same principle of operation and provide similar oxygen saturations and savings.

The oxymizer places the reservoir directly under the noise so it introduce oxygen to the patient from a proximal moustache (moustache-configure). For patients this type of conserving device is more comfortable but some patient does not like it because it is too much obtrusive [11].

The oxymizer pendant incorporates the tubular conduit and reservoir bag that rests against the patient's chest (pendant-configure). For patients this device is not comfortable but it is much less obtrusive than the oxymizer because the reservoir bag can be concealed ender clothing [11].

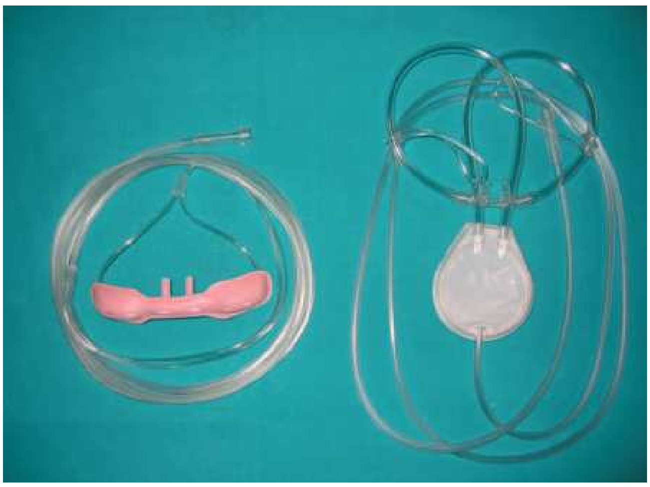


Figure 3.5. Reservoir cannulas: Oxymizer (left) and Oxymizer Pendant (right). From [8]

3.2.2.2. Catheter Transtracheal

The transtracheal system delivers oxygen directly into the trachea through a catheter that is introduced through a percutaneous route. The hypothesis of this method was derived from the idea that, if the oxygen was introduced intrabronchially as close to blood-gas interface, the oxygen delivery to the blood could be much more efficient [10], [11].

Therefore, the oxygen storage occurs in the upper airways, which act as a reservoir, towards the end of exhalation when air flow is at a virtual standstill, *i.e.,* during the expiratory pause after exhalation and before the succeeding inhalation and additionally, the dead space could be bypassed [10], [11].

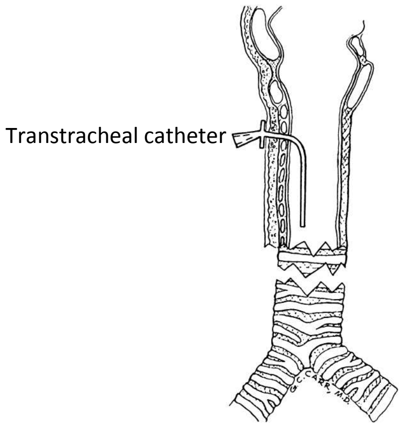


Figure 3.6. A transtracheal catheter (left) and it is positioned into the trachea (right). From [11], [13]

This transtracheal system of oxygen supply requires the implementation of a tracheostomy. This method is frequently rejected by the patients, mostly because it is an invasive method and because it needs special care for a good recovery. Complications such as subcutaneous emphysema, cellulitis, bleeding, etc are associated with transtracheal catheters that occur in the insertion area [11].

3.2.2.3. DODS

DODSs were developed at the beginning of the 1980s [8]. It is clear that the advances in oxygen technology are helping change the way we provide oxygen at home. These devices are now a normal part of LTOT therapy at home and an integral part of most modern oxygen technologies.

Pulsed or demand oxygen devices use an electronic sensor that detects negative inspiratory pressure and delivers accurate oxygen pulses rapidly, by opening a solenoid valve. This means that oxygen is only delivered during the inhalation phase of the respiratory cycle and wastage during exhalation is avoided, so adequate oxygen saturation can be maintained [8], [10].

The DODS uses two different strategies in order to improve gas exchange while reducing the amount of oxygen that accumulates in the dead space.

The pulsed oxygen devices use one of these strategies. It delivers a fixed bolus of oxygen, whose volume can be changed according to the device settings, only at the beginning of inhalation. The other strategy is used by the demand oxygen devices. This devices also delivers a smaller bolus of oxygen at the beginning of inhalation, however, it maintains a continuous flow during the inspiratory phase, being this flow generally at lower concentrations than in conventional continuous systems [8].

Figure 3.7. Demand oxygen delivery systems: valve fitted on liquid oxygen tank (left) and independent valve (right). From [8]

The major advantages of the DODS are that it prolongs the duration of the oxygen sources and therefore the oxygen conservation, avoiding its wastage. The Disadvantages of these systems include the need for a complex mechanicoelectrical device that may breakdown or not working properly and it requires the use of batteries, which will need constant replacement or recharge. Moreover, some patients feel discomfort with the nasal prongs and the firing noise [10].

The following table presents, the characteristics of each oxygen conserving devices.

	Catheter Transtracheal	Reservoir Cannula	DODS
How it works	Reduces dead space; Some storage.	Stores oxygen during exhalation and delivers it at beginning of inhalation.	Only delivers oxygen during inhalation (at beginning mainly).
Advantages	Portable; Improves lung function parameters; Unobtrusive.	Portable; Easy to use; Disposable; Least subject to mechanical failure.	Portable; Adapted to liquid oxygen tanks; Maintains O_2 savings ratio over usual delivery ranges.
Disadvantages	Invasive; Needs to be replaced periodically; Catheter clogs; Surgical complications.	Not very comfortable; Most obtrusive.	Reduced efficacy at high respiratory rates; Most subject to mechanical failure.
Cost	High	Low	Medium

Table 3.2. Main characteristics of oxygen conserving devices.

3.3. Benefits of Automatic Oxygen Delivery

The main disadvantages of oxygen supply systems have been described previously. Currently, the most common oxygen supply systems to patients using LTOT is gaseous oxygen cylinders.

This is the most expensive supplier system of LTOT and that usually cause great concern to patients and family, because the cylinders are heavy, empty itself rapidly and constantly need to be replaced with full cylinders, thus the duration of each cylinder is low so that the patient has greater dependence on it, less mobility and freedom [3].

It is a system very expensive because, beyond the gas prices, the whole strategy of applications, transport, delivery of oxygen to the residence of the patient and the difficulties and high cost of these procedures face up this system. Therefore, these actions need to be consistently very well orchestrated by the health system which dispenses the LTOT, by the company supplier and the oxygen users for the provision of LTOT should be smooth and uninterrupted [3].

The automatic system to be developed should incorporate any oxygen supply systems. However, in accordance with what was described in the preceding paragraph and in order to overcome these objections, the oxygen supply system that will more likely to be used will be oxygen cylinders liquid. Besides the fact that this gas is the more expensive the whole process of delivery to patient’s home makes the system cheaper than the cylinders of oxygen. It is a benefit for the company that makes the distribution of oxygen, in this case it is GASIN. This company collaborates with this project, since it gains by each patient/client and not by the time that they spent in each residence. Thus, using oxygen cylinders liquid system, the company moves less often to the patient's home, because according to this system the oxygen can be stored in a base unit, the patient will refill his portable cylinder when necessary and the replacement is not done so periodically since these cylinders present satisfactory duration. Beyond the benefits of the company, the patient benefits from this system because it is much more convenient, it ensures greater autonomy and mobility, and this gas has greater purity.

About oxygen conserving devices, there are many inherent disadvantages. The fact that they are bulky, uncomfortable, socially unacceptable (reservoir devices), or require surgery (transtracheal) or a mechanicoelectrial failure that does not allow the detection of inspiration (DODS) limit the use of these devices. These systems are only effective in accordance with its sensitivity to the respiratory cycle, the quantity of O_2 delivered and the delivery speed. They are devices that allow a saving of O_2, but the amount of O_2 that they debit is constant, i.e. the debit is not made according to the needs of patients in each moment.

To overcome these objections appears the Automatic Oxygen Delivery portable (increased patient's mobility), that proposes a self-regulating device that automatically adjusts the oxygen level administered to patients based on the patient's oxygen saturation reading. So, they will spend more time within the desired oxygen saturation (SpO_2) range as well as experience fewer hypoxic events.

Moreover, it is intended that this adjusting to real patient's needs expand the duration of the oxygen's reservoir, which not only reduces costs associated with it but also enables the patient walking longer.

Chapter 4. Theoretical Background

An important and necessary component to this project was the knowledge acquired along this year without which it would not be possible to accomplish it. Many of the knowledge acquired in the field of firmware programming have been performed for the first time by the students because during the Biomedical Engineering's course there was no contact with that area. This limitation was overcome with the help supplied by some ISA's engineers. Knowledge in the area of human physiology has been reviewed since it had already been studied.

The acquisition of new concepts and review of others was the first step for the realization and development of the project so, that students become familiar with issues relating to human physiology, C programming and microcontrollers.

The following chapter presents a brief description of these concepts and technologies for a better technical understanding of other chapters of this thesis. Further demonstrates the research work carried out during the development of the project.

4.1. Study of Oxyhemoglobin [1]

To better understand and apply the pulse oximetry in practice, it was acquired knowledge of the principles of transport and the oxyhemoglobin dissociation. The transport of O_2 in the blood occurs in two ways: dissolved in plasma or, mostly, present on hemoglobin (red blood cells/erythrocytes).

The hemoglobin molecule is composed of four polypeptide chains called globins (α and β) and four groups heme with a central atom of iron. Each group heme is responsible for establishing and transport the oxygen. So, one hemoglobin molecule can combine with four molecules of oxygen.

When the iron's heme is in the reduced form, it can share electrons and bond with oxygen to form oxyhemoglobin (HbO_2). When the dissociation of oxyhemoglobin occurs in the tissue capillaries the hemoglobin (without oxygen) is called deoxyhemoglobin (Hb).

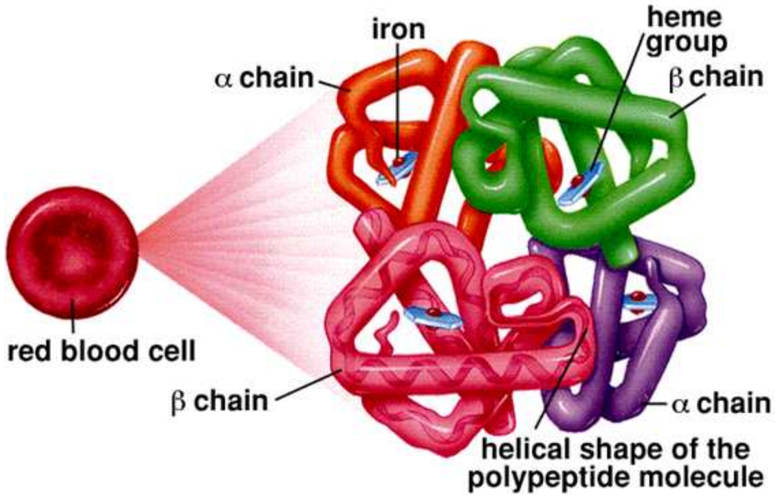


Figure 4.1. Hemoglobin's structure. From [14]

The process of transport and transfer of oxygen is carried out differently according to where the exchange of gases takes place, i.e. it is made in the pulmonary capillaries or the tissue capillaries. The following diagrams show the process for each one of them.

- **Pulmonary capillaries**

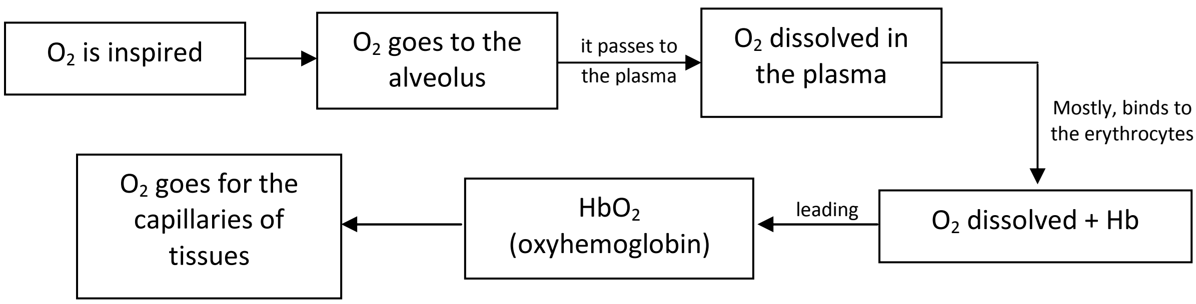


Figure 4.2. Scheme relatives of the O_2 exchange in the pulmonary capillaries.

In the capillaries of the lungs the loading reaction occurs, this is, the deoxyhemoglobin and the oxygen combine to form oxyhemoglobin.

- **Tissues capillaries**

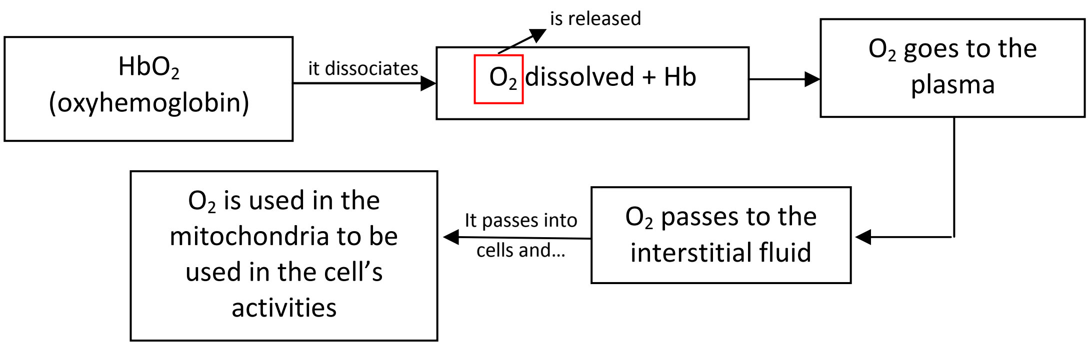


Figure 4.3. Scheme relatives of the O_2 exchange in the tissues capillaries.

In the tissue capillaries the unloading reaction occurs, this is, the oxyhemoglobin dissociates to yield deoxyhemoglobin and free oxygen molecules.

The operation principle of pulse oximetry is based on the exposure of oxyhemoglobin (oxygenated hemoglobin) and deoxyhemoglobin (deoxygenated hemoglobin) to the light. It is first necessary to clarify what is happening in hemoglobin during the gas exchange explained previously. For that it is require analyzing the oxyhemoglobin dissociation curve and the factors that influence the ability of this protein in the retention of oxygen.

In the following graph is shown the percentage of oxyhemoglobin saturation and the amount of oxygen in the blood, for different values of PaO_2.

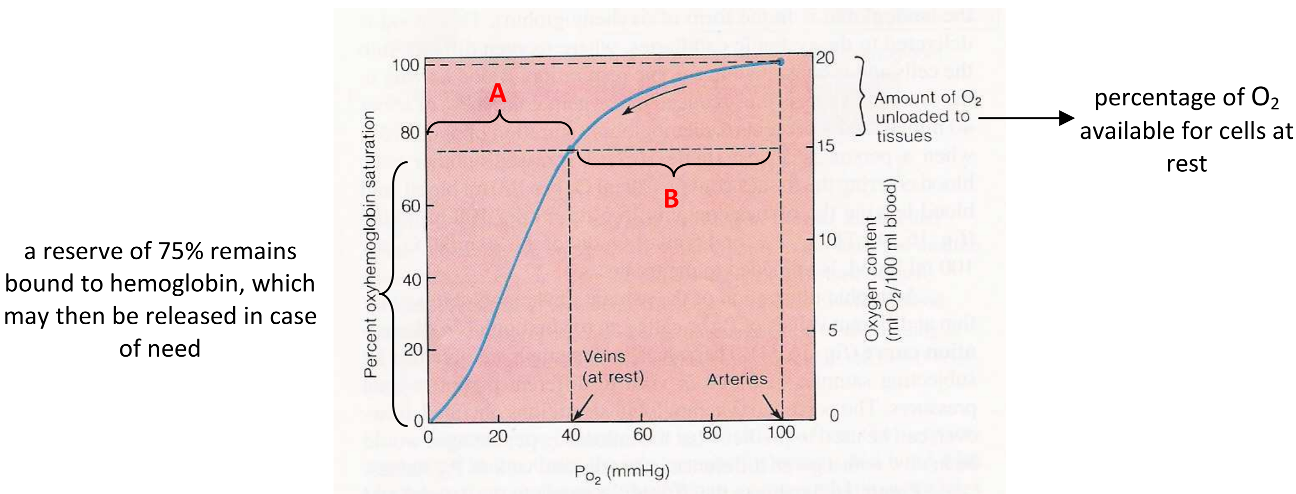


Figure 4.4. Oxyhemoglobin dissociation curve. Adapted from [1]

According to the graph it is noted that the percentage of oxyhemoglobin saturation decreases about 25%, once the blood passes through the tissues, the arteries to the veins (B). A large amount of oxyhemoglobin (75%) which remains in the venous blood serves as a reserve of oxygen, to be used in situations of high consumption.

In zone A, to small changes in PaO_2 there is great variation in the saturation of hemoglobin. In contrast, in zone B, to large variations of PaO_2 there is a small change of the saturation of hemoglobin.

Relatively the binding capacity of the hemoglobin and consequent percentage of oxygen saturation of hemoglobin depends on several factors such changes in temperature, pH and in DPG.

4.2. C Programming Language

The C language was created in 1972 in Telephone Bell Laboratories by Dennis Ritchie and Brian Kernighan and it was purpose to enable the writing of an operating system, the Unix [15]. Considered an intermediate language, higher-level language to assembly, and lower than the other languages, the C language is very powerful and versatile. It is a language known for its balance between structure and efficiency, eliminating the shortcomings of language assembly.

The C programming language has become rapidly one of the most important and popular language, especially for being very powerful, portable flexible and by the standardization of compilers available. It is now standardized in the Computing Industry as a well-known system implementation language and today is being used to develop new languages.

The advantage of the C language on the Assembly is on the fact that the development of codes involved situations more complex than those supports by a single machine language instruction [16].

The disadvantage of the C language is on the consumption of memory. Since the compiler cannot always cover all possible code optimizations, in many cases it has an increase of instructions, consuming a little more memory program [16].

4.2.1. Basic Structure of C programs

All programs developed in C follow a determined composition/structure. In the following paragraphs is intended to describe briefly each structure belonging to the C program and the functions associated with each part.

```
< pre-processor directives >
< global variables declaration  >;
main()
{
      < local variables declaration >; /* comments */
      < instructions >;
}
< others functions >
```

Figure 4.5. Basic structure of C programs. From [17]

- **Pre-processor Directives**

Each program contains a header with the directives of the pre-processor, i.e. commands, which define the value of symbolic constant, declaration of variables, including libraries, declaration of routines, etc.

When a program written in C is compiled, it is actually implemented two operations: the operation of pre-processing and the operation of compilation.

The function of the pre-processor corresponds to processing the instructions/directives, i.e. it allows to execute and transform the commands (all starting with #) in C code [15].

The two most common Pre-Processor Directives are #define and #include. The first substitutes a text for a specified identifier while the second includes the text of an external file into the program [18].

- **Variables Declaration**

A global variable must be defined before being used and any instruction, i.e. at the beginning of the program. The local variables must be declared into specific functions. The definition of a variable tells the compiler what type of data that is attributed to the name given to this variable. In a declaration, several attributes can be defined:

− **Basic Type:** char, int, float and double.

− **Signedness**: unsigned, signed.

− **Size:** short, long.

− **Type Qualifiers:** const (variable cannot be modified).

− **Pointer Types:** by means of the * type declarator.

− **Arrays:** by means of the [ ] type declarator, following variable name.

- **Main Function**

This function tells the compiler where the program should be started. Therefore, this function must be unique, appearing only once in each program. The program ends when it is closed the implementation of main function.

- **Expressions**

A C program is mostly constituted by expressions. An expression is every piece of code that combines operators and operands in order to obtain a single value. This block of instructions is inside the functions.

- **Comments**

The comments are not interpreted by the compiler, are only for documentation and clarification of the programmer allowing an easy understanding of source code.

4.3. Microcontrollers

A microcontroller differs from a microprocessor in several important ways. A microprocessor is simply the “heart” of a computer. In order for a microprocessor to be used, other components such as memory, or components for receiving and sending data must be added to it. By contrast, the microcontroller was designed to be a complete computer on a single chip that contains data and program memories, input/output interfaces and peripheral modules.

This high integration makes these devices more efficient, faster, with higher capacity, more resources for connectivity and control and lower power consumption. This particularity makes them suitable to typical applications.

The PIC (Peripheral Interface Controller) microcontroller was the subject of study and work during the development of the project. It is a microcontroller developed by Microchip which has a good diversity of resources, processing capabilities, cost and flexibility of applications. The PIC is available in a wide range of models to better fit the requirements of specific projects, differing by the number of I/O’s lines and the contents of the device. In the project was used a PIC24H microcontroller (16-bit), since this was incorporated in the *Leonardo* Platform.

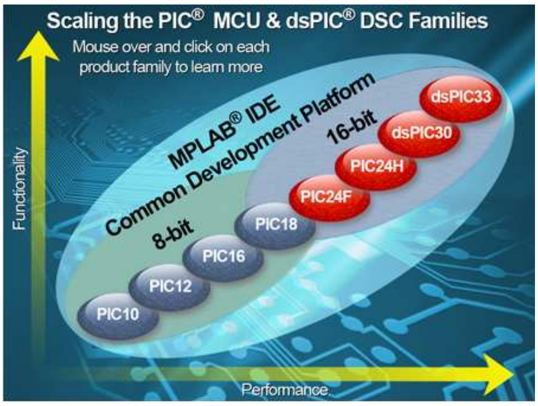


Figure 4.6. Microcontrollers Microchip PIC models. From [16]

A microcontroller is made of basic units which will be described in the topics below.

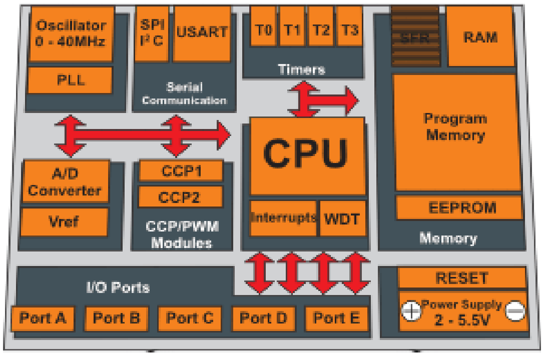


Figure 4.7. Schematics of a Microcontroller. From [19]

4.3.1. Central Processing Unit

The Central Processing Unit (CPU) is the main part of a microcontroller. This part is responsible for finding and fetching the instructions for decoding and execution. The instructions of a program stored in memory are fetched by CPU and decode into a set of actions that perform an assigned task. These actions may involve transfer of data from one memory location to another, from memory into ports and various calculations, so the CPU must be connected to all parts of the microcontroller [20].

4.3.2. Memory Unit

In a microcontroller, the program instructions are usually stored in a memory. The memory of a microcontroller is divided into different parts, each one with a specified address. From the address you can access each part of memory for manipulate data, just that way it guarantees access to the right location.

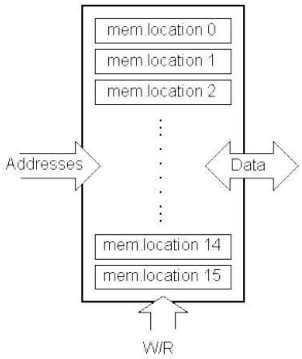


Figure 4.8. Simplified model of microcontroller's memory. From [20]

The memory of microcontrollers is divided into two blocks, one for data and one for the program. The EEPROM and RAM memory constitute the data's block and FLASH memory constitutes the program's block.

The program memory has been performed in FLASH technology, which stores the necessary instructions to allow the programming of microcontroller many times as necessary before and after it is installed into a device.

The EEPROM memory is usually used for storing important parameters but it is not directly part of the memory space. It is accessed indirectly through registers during program execution. The RAM memory stores general purpose, leaving the programmer to control it and special registers (specific function for each bit).

- **Buffer**

The buffer is a part of data's block memory (RAM memory) and it is a place of temporary memory used for reading and writing data. It is possible to store data received by the UART and data to be transmitted via UART. The buffers can be implemented in software and are typically used when there is a difference between the rate at which data is received and the rate at which they can be processed. These are three components of the buffer to take into account: the tail, the head and the data. This last is being used to determine the capacity of the buffer's memory. In the process of sending data from the buffer (TX), is the tail that moves up to match the head, this tells that there is no more data to transmit. To receive data to the buffer (RX) is the head that moves for successively store the data in its positions.

4.3.3. Input-Output (I/O) Ports

One of the most important features of the microcontroller is a number of input/output pins used in order to monitor or control other components or devices. These pins are grouped into I/O ports and each pin of the port can be set up as an input, output or an input/output to send or receive data to or from the port.

Each I/O port can have 8 or 16 I / O pins and each port has two registers that control its operation, a *data register* and a *direction register*.

The data register is called the PORT register. This register reads the state of the pin. If the pin is high means that the pin is on a maximum voltage (e.g. 3.3V); if the pin is low means that the pin is on a minimum voltage (e.g. 0V). Each I/O port has its own PORT register, e.g. to port A the data register associated the name PORTA.

The data direction register is called the TRIS register. Each I/O port has its own TRIS register named after the port letter for example TRISA is for Port A.

Each pin of the port can be set up as an input or an output by controlling a bit within the TRIS register. If at the appropriate place in TRIS register a logical "1" is written, then that pin is an input pin, and if the opposite is true (logical "0"), it's an output pin [20].

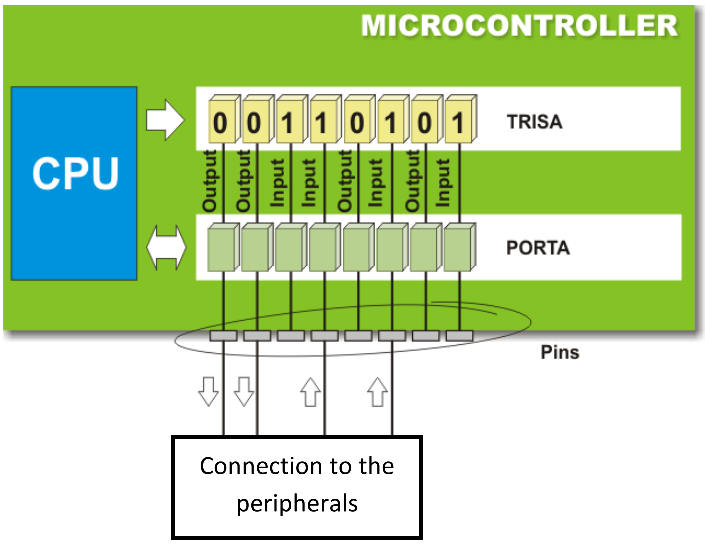


Figure 4.9 Schematics of I/O ports. From [19]

Not all I/O ports' pins are necessarily being used as digital, they can have multiple peripheral functions like timers, serial communication, oscillator, etc., i.e. they are pin multiplexed. So, it is necessary to assign the right function to the I/O ports' pins as well as to set their direction (input or output).

4.3.4. Timers

The microcontroller timer units are based in a counter register whose value is continually increasing every time a given pulse occurs, which is supplied by a calibrated oscillator with a known and precise frequency. Thus, the process is automatically increased and it is possible to establish a relation between the temporal dimension and the counter variable, because the counter determines the difference between two different moments of time.

Timer units have a wide range of applications, so the programmer will take advantage of this characteristic for his needs the best way he finds.

4.3.5. Serial Communication [22]

Serial communication is to generate the transmission and reception of data from one point to another using a digital channel. The data bits are sending in a sequence along at the same line with a certain speed and frequency of transmission. This type of communication contrasts with the parallel communication, in which the data bits are transmitting together.

Generally, serialized data are not uniformly transmitted through a channel, i.e. they are sent followed by a pause until the message has been fully transmitted. Two basic techniques are employed to ensure correct synchronization, the synchronous and asynchronous communication.

In synchronous systems, separate channels are used to transmit data and timing information. The timing channel transmits clock pulses to the receiver, which reads the data channel and stores the bit value. The receiver reads the new data channel only when arise a new clock pulse and thus ensures the synchronization.

In asynchronous systems, information is transmitting by a single channel. Then, for the communication is established correctly the transmitter and receiver must be set in advance to avoid errors between communications. Thus, it is possible to communicate between devices, send an unlimited amount of data with a reduced number of channels.

In order to make integration easier, most PIC Microcontrollers have serial I/O modules available, and the UART module is one of them. It will be given a brief overview on this communication module.

4.3.5.1. UART Module

The Universal Asynchronous Receiver/Transmitter (UART) module is the key component of the serial communications subsystem of a computer.

The UART module implements only the asynchronous communication that communicates with peripheral devices and personal computers using protocols such as the RS-232 [21], [31].

In asynchronous systems the exchange of information is made by the pins UxTX and UxTR, for transmission and reception respectively. This means that both can occur at the same time i.e. full duplex operate.

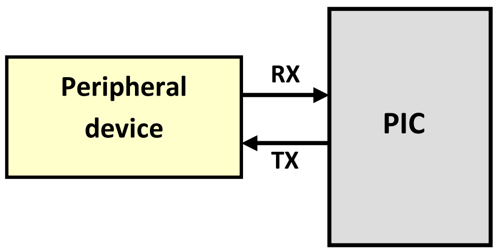


Figure 4.10. Illustrative scheme of the connection lines exchanged between a peripheral device and a PIC in Asynchronous Serial Communication.

The UART uses the communication protocol NRZ (Non-Return-to-Zero), by sending 1 Start Bit, 8 or 9 Date Bits and 1 or 2 Stop Bits. In the most common format, the data is sent in small packets of 10 or 11 bits, which 8 of them are the message [21].

The signal has a logic value '1', when the channel is idle. Data transmission begins when the signal has a logic level '0' (start bit) to advised the receiver that the signal's transmission was started. The "start bit" starts an internal timer in the receptor, warning that will be necessary clocks pulse. Followed by the start bit, 8 bits of data are sent according to specified transmission rate (baud rate – number of bits per second that can be transmitted in a given route). The data's transmission is completed with the parity and stop bits [21].

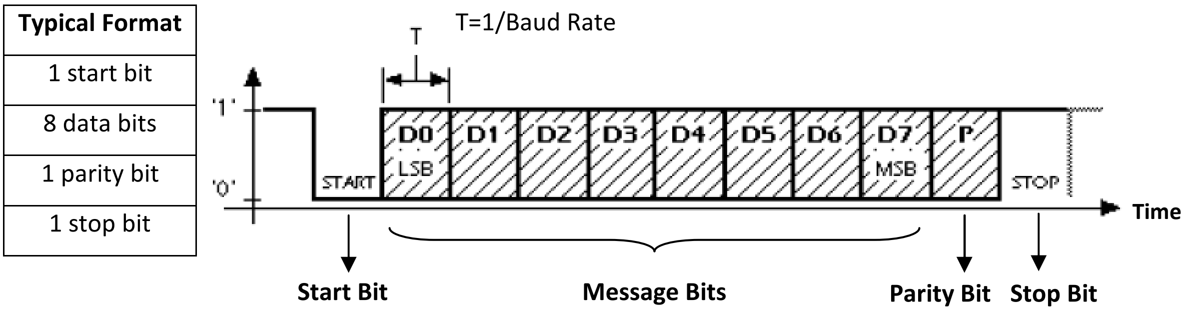


Figure 4.11. Illustrative scheme of a typical format of Asynchronous Serial Communication. From [22]

RS-232 Protocol [22]

Recommended Standard RS-232 is a common interface for serial data communication between equipments, established in the early'60s, by the "Electronic Industries Association (EIA).

The RS-232 interface specifies the tension, timings and functions of signal, a protocol for information exchange and mechanical connections. The RS-232 standard defines a set of specifications for the serial transmission of data between two devices called DTE (Data Terminal Equipment) and DCE (Data Communication Equipment). The DTE is a device that made the processing of the data, while the DCE is a device that generates the data. Normally, a cable connection between DTE and DCE devices contains links in parallel, not requiring changes in the connection pins.

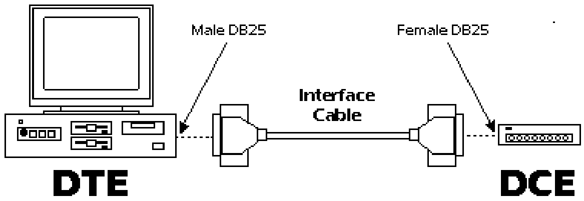


Figure 4.12. Connection between the DTE and DCE devices.

The RS-232 protocol operates for both positive and negative values, but to have a valid signal the lower and upper limits must exceed the-3V and +3 V. Signal voltage between -3 volts and -15 volts with respect to land are considered logical level "1" (mark condition), and voltage between +3 volts and +15 volts are considered logical level "0" (space requirement). The range of voltage between -3 volts and +3 volts is considered a region of transition for which the state of the signal is undefined. If occurs transmission of data the transmit data line is +15 V corresponding to logic condition “0”; if there is no data transmission the voltage in the transmit data line is −15V corresponding to logic condition “1” [22].

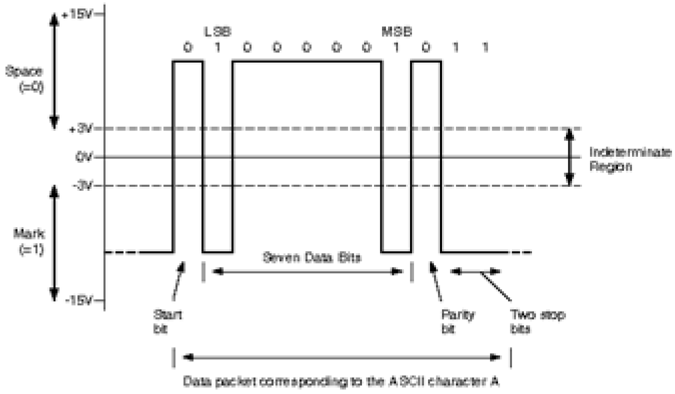


Figure 4.13. Illustrative scheme of a typical operation of RS-232 protocol. Adapted from [22]

Transístor to Transístor Logic (TTL) Signals

TTL is common semiconductor technology for building discrete digital logic integrated circuits. Typically, TTL refers to a signal level where the voltage is 0 to 3.3 volts as opposed to RS-232 where the signals are +/- 15 volts. Most digital devices use TTL levels. So the first step to connect digital equipment to an interface is to convert the values +/- 15 RS-232 to TTL levels (0/ 3.3 volts) and vice versa. This is done by level converters that allow converting an input signal in another signal with different voltages [22].

4.3.6. Oscillator

The circuit of the oscillator is used to provide a clock signal (clock) to the microcontroller. This clock is necessary for the microcontroller can execute a program or the instructions of a program. The oscillator system of the PIC24 includes certain characteristics such as four oscillator settings, Phase-Locked Loop (PLL) system, switching between different sources of clock, energy-saving system (doze mode), Fail-Safe Clock Monitor (FSCM) system, non-volatile configuration bits to select the source clock [23].

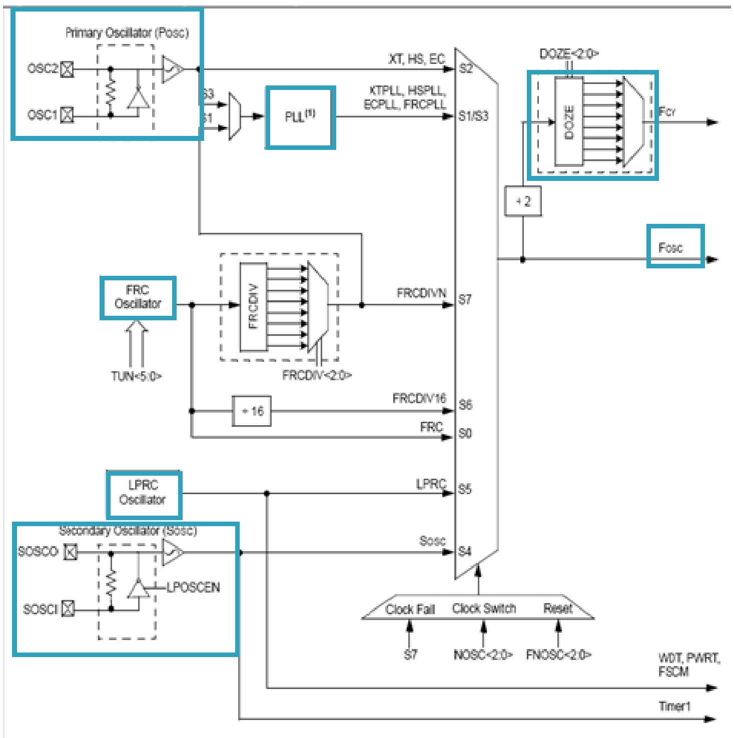


Figure 4.14. Oscillator system block diagram. From [23]

The frequency of the system clock (FOSC) can be obtained for various configurations of oscillator:

- Primary Oscillator (POSC) by pins OSC1 and OSC2 with or without PLL:

- XT (crystals and ceramic resonator) – frequencies of 3 to 10MHz.
- HS (High-Speed Crystal) – frequencies of 10 to 40MHz.
- EC (External Clock) – frequencies of 0.8 to 64MHz.

- Secondary Oscillator (SOSC) by pins SOSCI and SOSCO
- FRC Oscillator with an option to the clock divider or PLL
- LPRC Oscillator

It should be noted that in this PIC an instruction requires two clocks, and the value of FCY (frequency of instruction) is half of FOSC. This is a value that must be taken into account because it is the value that tells the frequency at which the bits are sampled, in this case is frequency at which instructions are executed in the PIC [23].

The primary oscillator and FRC may optionally use PLL. This system allows to multiply the oscillator's frequency keeping the wave's phase relationship to obtain higher operating speeds. It is needed to configure the PLL to run the PIC at the maximum clock speed from the chosen oscillator. To configure the PLL system is necessary to take into account certain parameters. The diagram below illustrates the PLL's actuation process [23].

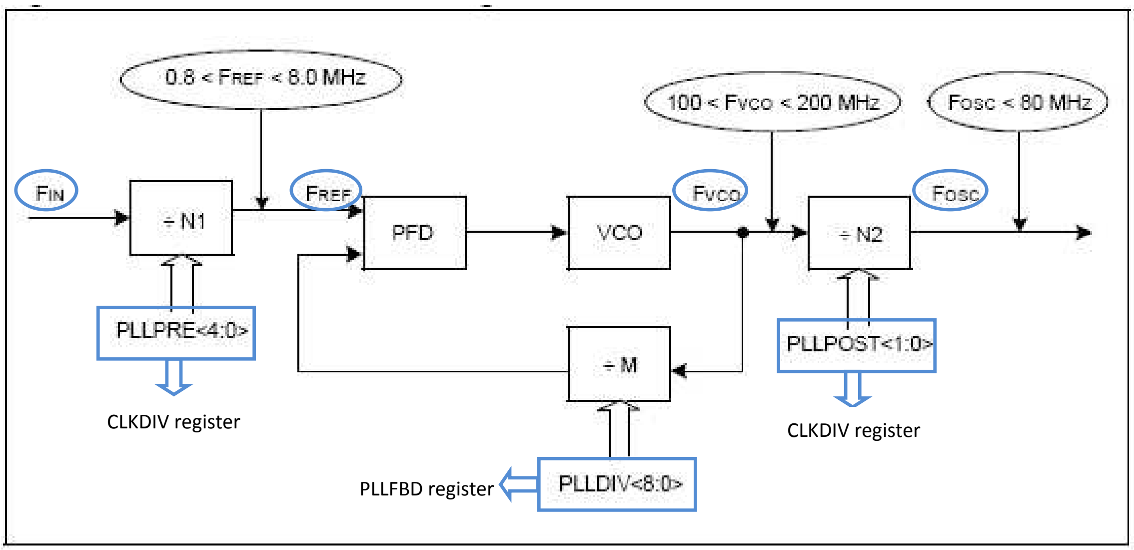


Figure 4.15. PIC24H PLL block diagram. From [23]

Therefore, to obtain the value of FOSC and consequently the value of FCY, it is necessary to submit the value of the input frequency (Fin) to a series of calculations and it must meet some requirements. The settings of each parameter (N1, N2, M) are made from special registers (see attachement B).

4.3.7. Interrupt System

Interrupts are a mechanism of a microcontroller which enables it to respond to some events at the moment when they occur, regardless of what microcontroller is doing at the time. The interruption allows the PIC intercept external events to the program running. It temporarily stops to run the code in operation, monitor the event with an appropriate subroutine and continues to run the code exactly where it was left off [21]. Normally, the flag interruption has the logic value "0”. When occurs an interruption, the flag receives the logic value "1" and the microcontroller does not responds to another interrupt. Upon completion the interruption the flag has to receive the logic value “0” again, this task has to be done by the programmer.

The microcontroller is a multifunctional system consisting of several modules that need a correct configuration and the interrupts are one of these modules (see attachment B).

4.3.8. Analog Module

The binary numbers (zero and one) understood by the microcontroller is different from the format of the signals sent by the peripheral, so it is necessary convert this information into binary values to be understood by the microcontroller. The module capable of performing this conversion is called A/D converter (analog/digital converter). Furthermore, this block allows the transmission of information to the CPU to be processed immediately [19].

4.3.9. Output Compare

The PIC also presents a compare module, which includes different modes of operation, a total of seven. These modules allow to generate an output pulse or a sequence of pulses through the change of state of the pin-out during the comparison of successive events. The method used to control the proportional valve will be the electronic mode PWM (Pulse Width Modulation) unprotected by default (without PWM mode fault protection). This mode is similar to the PWM mode with a default protection, however, this allows activation by an external signal [24].

The Pulse Width Modulation (PWM) is used to generate an analog output signal continuously variable from a digital signal. It works ranging the average stress through the generation of a signal of constant frequency through the modeling of the wave's cyclic ratio (duty cycle).

The duty cycle is defined by the time that the signal remains high ("1") opposing the time it is low ("0"). For example if the maximum voltage is 5V (high level), the minimum is 0V (low level) and it is necessary an output signal of 2.5V, the signal must be half the time high and half the time low, i.e. the ratio cyclic should be 50% [24].

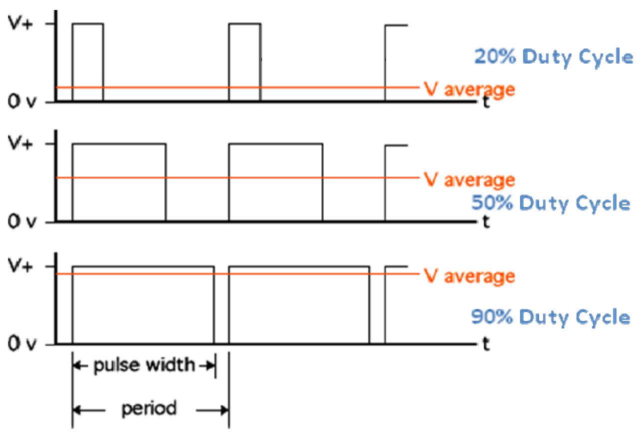


Figure 4.16. Illustrative scheme of examples of the duty cycle operation. Adapted from [32]

4.4. Programming Microcontrollers in C

Traditionally, most of embedded programs have been written in programming assembly language. However, due to the many microcontrollers' suppliers, most new microcontrollers can be programmed using the high-level C programming language as well as assembly.

Compared to an equivalent assembly program, a well-written C program has three major advantages: portability, readability and modularity.

The portability allows the embedded programmer to remove the specifics about the target hardware and focus on a functional implementation. With some care and some amount of changes of the programming source (re-programmed), the program C may be moved to a different hardware [25].

The readability of the high-level language allows for easier reading (and writing) due to the similarity with the programmer's native tongue. Associated with the readability is maintainability, this allows to maintain a program when written in a high-level language because they are easier to read [25].

The modularity allows the embedded programmer to collect one's embedded source into modules which allows re-using of source code in different projects. This programming concept can decrease a project's development time [25].

There is a consequence regarding the compiler's effectiveness despite the advantages of programming using a high-level language. The compiler cannot always translate the C source code to the machine code that may result in reduced code efficiency when programming in a high-level language. To overcome or minimize this consequence the programmer should select a microcontroller which was designed to be programmed using C language and, according to the compiler, determine the best mode to write code efficiently [25].

4.5. Development Cycle

To develop a code the firmware programmer needs to define his Development Environment. So, the process for writing a program is often described as a development cycle by main steps, because it is rare that all the steps from design to implementation can be done flawlessly the first time they all link together forming a cycle. To produce an application that performs correctly, the code has to be written, tested and then modified.

In this process the main steps that form a cyclical operation are: the Edit/Create/Design code, that creates the high level design; Compile/Assemble/Link code, that converts the source code to machine code; Program device and Analyze/Debug firmware that tests the code. This cycle is represented in the following figure [26].

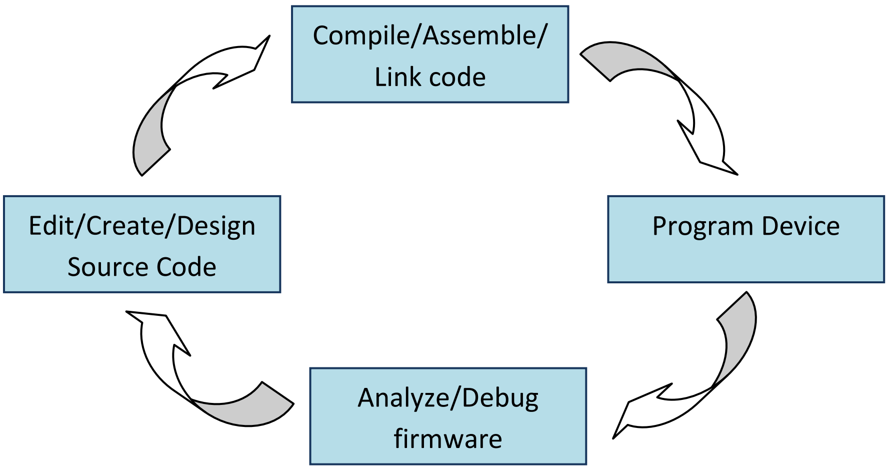


Figure 4.17. The development cycle. Adapted from [26]

Now it is possible identify the Development Environment's components once it is known the Development Process' flow. These components are a Compiler, a Programmer/Debugger and an Integrated Development Environment (IDE) software.

- **Compiler**

According to the microncontrollers's architecture two types of compilers can be used. This is for 8-bit PIC microcontrollers the compiler uses is the HI-TECH PICC-18 ™, for 16-bit PIC microcontrollers the compiler used is the MPLAB C30.

A compiler is a piece of software that translates a high-level programming language to a lower level language, e.g. C language to assembly language. The compiler uses a number of relatively simple rules in order to simplify and optimize assembler code, making the code understandable. One classification of compilers is by the platform on which their generated code executes. So, for this kind of compilers where the code is running on a platform other than the one whose code is being built is called cross-compiler [26].

To select the compiler it is most important to take account the full-feature compliance with the ANSI C language, because this allows to minimize the user integration impact into different compilers. The MPLAB C30 is a full-featured ANSI compliant C compiler for PIC 16-bit microcontrollers. It is fully compatible with Microchip's MPLAB IDE, allowing source level debugging with the MPLAB ICD2 [21].

- **Programmer/Debugger**

After the process of compile / assemble / link and built the code with no errors it is necessary to store the final firmware in microcontroller's memory. This process of transfer of the code developed into the chip is called programming, after that the program will run automatically. A microcontroller can be programmed with the In-Circuit Debugger, which consists on a hardware piece that stands between the developer's PC and the system to be tested. They are connected by wires via RS-232 serial or USB interface [26].

This device allows the programmer to debug the code at the same time it is running on the PIC, making the developing task much easier and if something goes wrong there is no output and the programmer can find the problem.

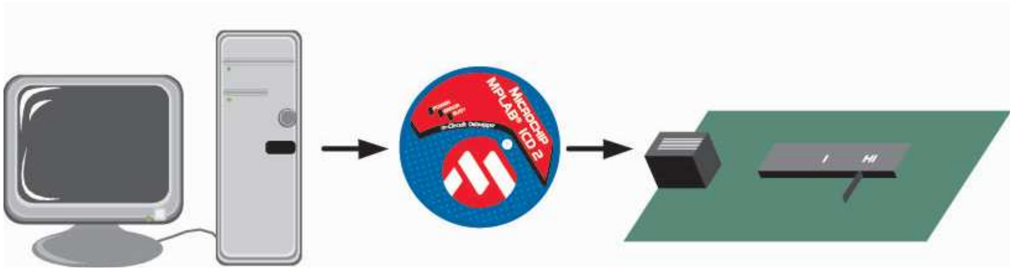


Figure 4.18. Schematics of the connection between MPLAB ICD2 and two devices (e.g. PC and microcontroller). From [27]

During the developing process it was used a MPLAB ICD2 from Microchip, the same manufacturer of PIC microcontrollers. The features that it offers to the programmer are real time and single-step code execution, breakpoint setting and registers and variables watch or modify [26].

- **Integrated Development Environment [26], [28]**

Integrated Development Environment or IDE is the main part of the programmer's Development Environment. This is a software application that combines features and tools that provides comprehensive facilities to computer programmers for software development and test the final code. IDEs are designed to maximize programmer productivity by providing an integrated and user-friendly single interface.

The MPLAB IDE is a PIC microcontroller IDE from Microchip that helps the programmer to focus on important details and correlate information from the design phase thought the debugging, optimization and the programming phases.

It has built-in components to configure the system for a variety of software and hardware tools. They include the Project Manager, Editor, Assembler/Linker and Debugging Tools.

Project Manager

The project manager organizes the source files that compose the firmware code, to be edited. So they can be sent to the language tools for assembly or compilation, and ultimately to a linker. This provides integration and communication between the IDE and the language tools.

Editor

The editor, beyond being a fundamental programmer's text editor, is also a support to the debugger module, so it is an indispensable component in the IDE software. Source code editors have features specifically designed to simplify and speed up the development and debugging firmware. In case of MPLAB IDE it has not so many features like others editors, but due the simplicity C language it is not require a power full text-editing feature.

Assembler/Linker

The assembler converts the source files into intermediate modules of machine code so that it can be programmed into the microcontroller. The assembler can use a single file, or if the project comprises several source files, can be used with a linker. The linker is responsible for linking these several source files, modified by assembler, into a single executable program. It is also responsible for positioning the compiled code into memory areas of the embedded controller, and ensures that the modules function with each other.

Debugger

Debuggers, typically, comprises sophisticated functions such as running a program step by step (single-stepping), pausing the program to examine the current state (breaking) by breakpoints, and watch/modify the values of some variables. It works in conjunction with the editor to reference information from the target being debugged back to the source code.

Chapter 5. System Architecture

5.1. Project Requirements

5.1.1. Scenery – Oxygen Therapy at Home

The oxygen is part of the air we breathe and is essential for life. When you breathe, the air enters in our lungs and from there the oxygen is transported by blood to all our body.

Patients living with hypoxemia, i.e. disability abnormal concentration of oxygen in arterial blood, have significant impairment physical, mental and social with deterioration in the quality of life. Moreover, these patients have several complications, with numerous hospitalizations and consequent increase in economic cost for all systems health.

Studies in patients with COPD and other lung diseases shown that LTOT at home, i.e. that the administration of oxygen at home, reduces tissue hypoxia during the activities of daily; increases the survival of patients by improving the physiological variables and clinical symptoms; increases the quality of life by increasing exercise tolerance, reducing the need for hospitalization, and improves neuropsychiatric symptoms resulting from chronic hypoxemia. All of these advantages evidence that this treatment, performed at home, provides benefits, both for patients and for the healthcare professionals. So, this project would be developed for application in home care.

5.1.2. Vital Signs Acquisition

Vital signs acquisition is an important aspect on the vital signs monitoring in failure respiratory patients. This control is necessary to avoid situations of possible risk to the patient.

The acquisition of the patient's vital signs will be made by pulse oximetry that measure the patient’s SpO_2 and an accelerometer that measure the activity of the patient.

To understand the importance of vital signs measured at each sensor and validity of them it is made an explication of the operation's principle of each one.

5.1.2.1. Pulse Oximeter

The pulse oximeter provides an indirect and noninvasive measure of hemoglobin's oxygen saturation in the blood, as well as changes in the volume of blood which allows the value of the heart rate.

This technique is characterized by being of high reliability, and allows instant verification of the patient's blood oxygenation status, yielding results in a continuous mode.

The measurement of oxygen saturation is based on the principle that establishes the transmission and reception of light, red (600-750nm) and infrared (850-1000nm), performed by LEDs and photoreceptors respectively [12].

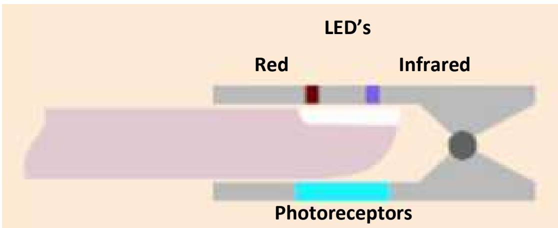


Figure 5.1. Principle of oximeter's operation. Adapted from [12]

The light infrared and red (alternately connected and disconnected) pass through the tissues of the body and subsequently they are detected by photoreceptor that measures the intensity of light received.

The operation of the oximeter is based upon two physical principles: the spectrophotometry and the plethysmography.

The first one, is based on the exposure of oxyhemoglobin (oxygenated hemoglobin) and deoxyhemoglobin (hemoglobin deoxygenated) to light infrared and red. This means that each of these hemoglobin's forms retain different amounts of light of each wave lengths employees, i.e. oxyhemoglobin absorbs more infrared light, while deoxyhemoglobin absorbs more red light, as seen in the chart below [12].

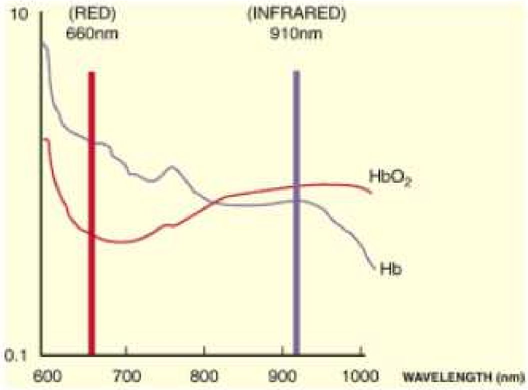


Figure 5.2. Spectrum of absorption of hemoglobin from oxyhemoglobin. From [29]

The second one is directly related with the spectrophotometry, because the volume of blood in tissues changes the absorption of light during the pulse. This is, the blood rich in oxygen absorbs most amount of light, while the blood low in oxygen absorbs less amount of light. So, during the diastole the amount of oxyhemoglobin is greater which increases the light absorption. During the systole the amount of deoxyhemoglobin is greater then there's less absorption of light. [12].

According to the maximum and minimum values obtained of the light transmitted/absorbed the oximeter estimates the saturation of arterial hemoglobin in the oxygen carrying (in percentage) [12].

During patient monitoring, the oximeter software selects the appropriate coefficients for the calculation of SpO_2. Measurements of oxygen saturation are converted to a real value from a ratio. The result of this calculation indicates the value of oximetry as a percentage.

The solution of the pulse oximetry continuously monitors the state of oxygenation of a patient. Depending on the patient’s data obtained by oximetry and using the parameters pre-set by the physician mentioned above, the negative feedback system responds by adjusting the treatment to real needs, avoiding situations of hypoxia.

In addition to the steps defined allow an adjustment of the oxygen flow to the patient’s SpO_2, this solution also allow greater savings oxygen, thus avoiding its waste.

5.1.2.2. Accelerometer

The accelerometers are sensors or transducers that measure acceleration. The acceleration is a measure of how quickly the speed varies and can be obtained using two or three directions, using accelerometers bi or triaxial, respectively.

These sensors respond to both the frequency and the intensity of the movement, and its use in the assessment of physical activity based on relationships established between the output of the accelerometer and the energy used in studies examining the movement or ergonomic studies.

The accelerometer is used in medical and therapeutic field like the assessment of human movement, the detection of sleep disorders, the detection of physiological variables, among other applications [30].

In the assessment of human movement and medical rehabilitation, it can be used to determine the risk of falling in elderly which the underlying causes are related to a gait and balance disabilities. So, it is possible to assess the physical conditions of individuals suffering from various types of diseases [30].

5.2. System Specifications

The following chapter describes the system architecture as well as specific information on each of the system's elements, which allows the reader to better understand the function of each component.

The system, developed under the project, presents its architecture a number of components, all integrated with the *Leonardo* Platform, which contains the microcontroller PIC24HJ128GP306.

As problems arose with the RS-232 communication between the pulse oximeter Oxilink™ from Smiths-Medical and the *Leonardo* Platform, it was necessary to make an intermediate connection using an iDemo Board. Thus, the pulse oximeter communicates with the iDemo Board through the RS-232 and the signals coming from this board is sent to the *Leonardo* Platform to be processed later.

The XYZ accelerometer Plux communicates with the *Leonardo* Platform through the 3 analog inputs.

The PIC microcontroller operates the valve via PWM mode that provides a voltage to the Driver Board. This driver board receives the voltage signal and converts it into current signal to the electronic proportional valve. This valve is directly connected to the oxygen source, so when it receives the current signal by the driver board, it controls the oxygen flow from the oxygen source, changing its diameter. The other side of the valve is link to the patient through an interface such as a nasal cannula, allowing the delivery of the oxygen.

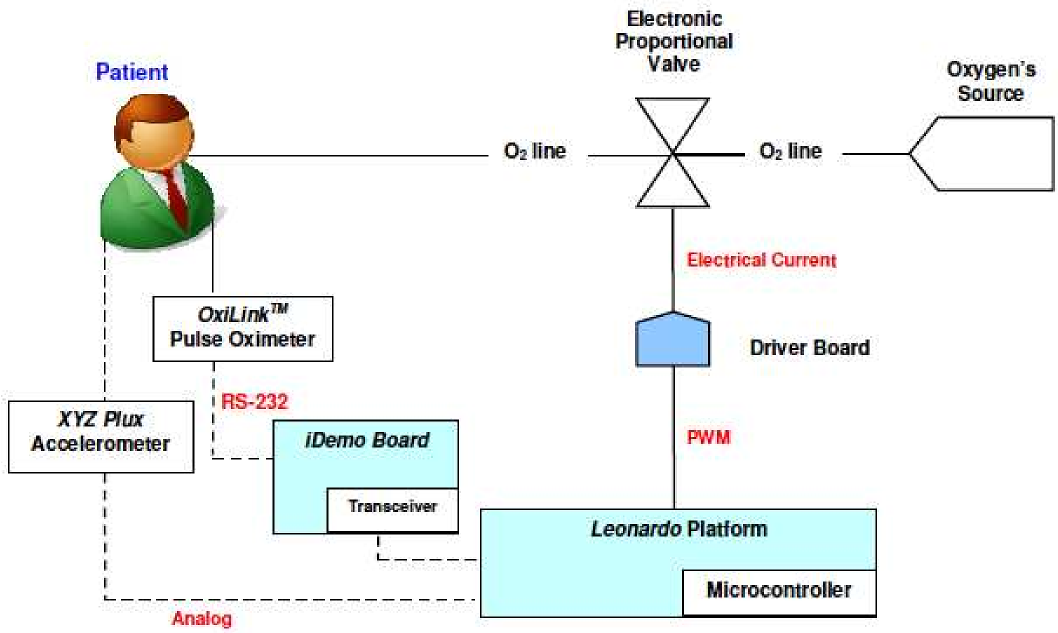


Figure 5.3. Physical architecture of the system to implement.

5.2.1. Acquisition and Processing Module

The acquisition and processing module is responsible for the vital signs data collection. The vital signs of interest are the pulse oximetry and later the accelerometry to detect the movements/posture of the patient. The sensors are placed directly over the patient and continuously measure his physiologic parameters and transmit them to this module to which they are attached via wire.

The diagram below shows how the acquisition and processing module is organized.

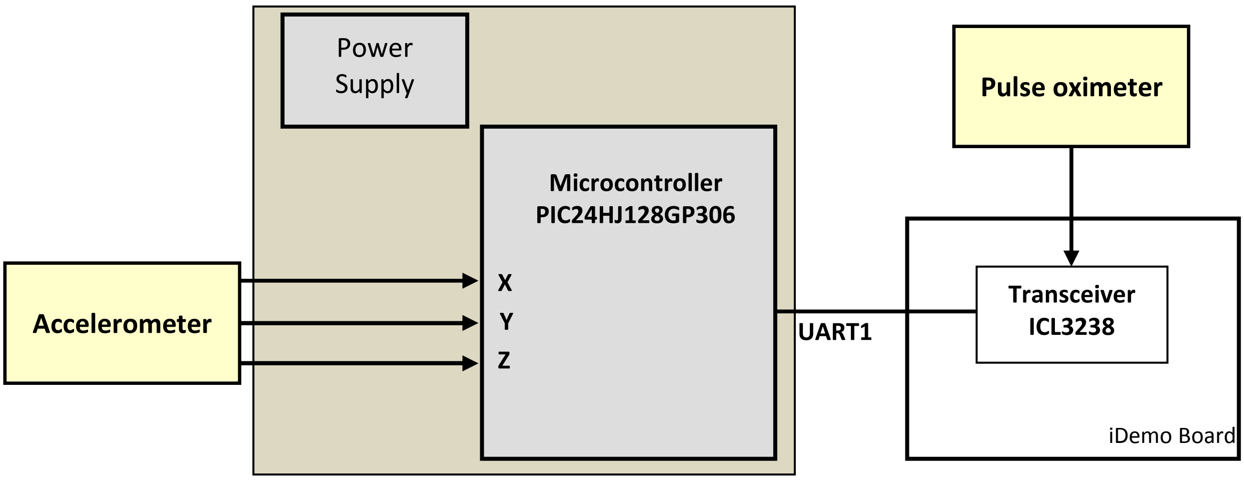


Figure 5.4. Illustrative scheme of the acquisition module.

5.2.1.1. iDemo Board

The incorporation of this board was not planned for this project. However, we had to use it since the communication between the oximeter and the *Leonardo* Platform was not achieved, because the oximeter communicates by RS-232 and the microcontroller's platform only receives TTL signals.

Several solutions were tried such as to connect the pins of the RS-232 directly in the corresponding sections of the *Leonardo* Platform (GND, 12V power, SYNC, TX, negative power), but it didn't solve the problem; to integrate a transceiver on the *Leonardo* Platform in order to convert the signal, but it was unsuccessful too due to the protocol RS-232. According to the specifications of the protocol previously described (see subchapter 4.3.5.1) to have a valid signal, the limits (upper and lower) must exceed +3V and be lower then -3V. In this case, there is no problem with the upper limit; however, the lower limit is not situated below -3V as it is supposed to be.

The solution was to incorporate an iDemo Board in the system. This intermediate connection solved the problem mentioned. This is because the iDemo Board converts the RS-232 signals from the oximeter to TTL signals through the transceiver ICL3238. These signals are subsequently sent by the iDemo Board for the *Leonardo* Platform (directly on the UxRX pin). Now the system is using the iDemo Board and the *Leonardo* Platform but it is intended to optimize this model at a later stage.

The figure below shows the iDemo Board and the *Leonardo* Platform, the components that allow to receive the data (RS-232) and to converter the input signal to a TTL signal (transceiver). It also shows the connection between both of the boards that allows the sending of the signals from the iDemo Board to the UxRX pin's *Leonardo* Platform.

The signal from the oximeter enters on the iDemo Board's DB9 connector of the pin 3 - $R2_{in}$ – passing through the ICL 3238 transceiver and the signal converted goes out by the transceiver's pin 20 - $R2_{out}$ (see attachment A2). This signal from the pin 20 is sent to the pin 34 (UxRX) of the PIC24 (see attachment A1) through a direct connection (red wire in the figure).

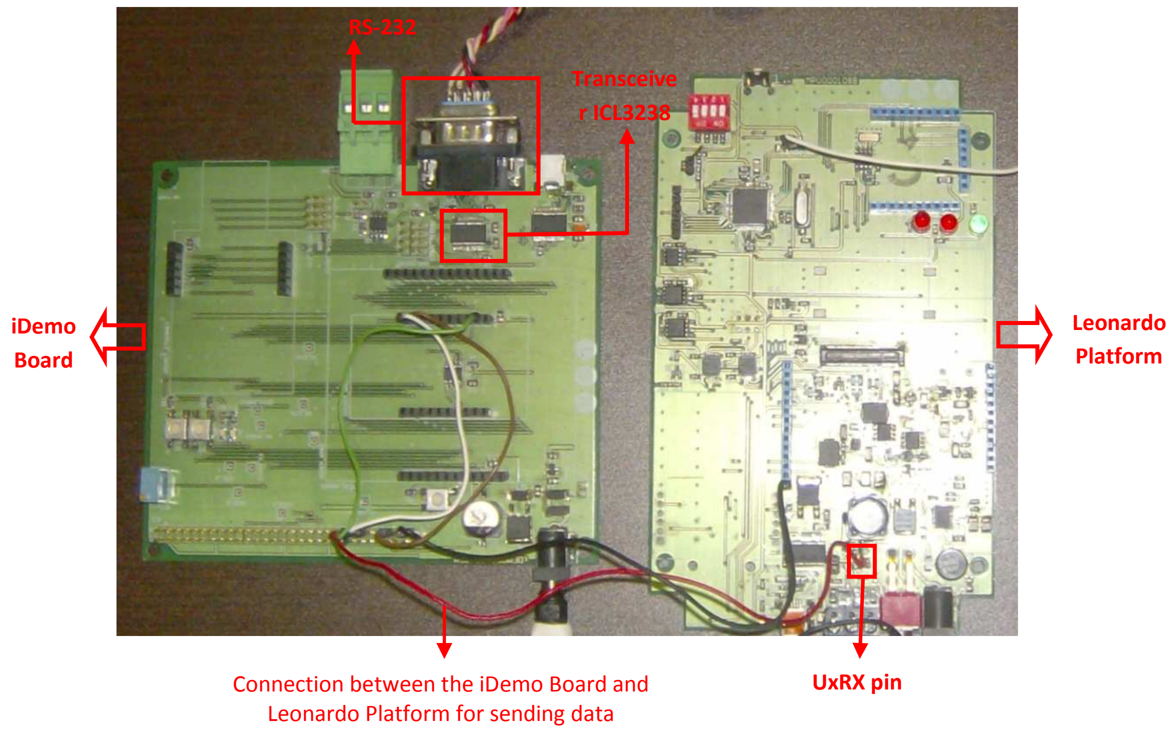


Figure 5.5. Connection between the iDemo Board and *Leonardo* Platform.

5.2.1.2. *Leonardo* Platform

The *Leonardo* Platform available on the company and developed under another project was chosen to implement this system, and it has incorporated the microcontroller PIC24HJ128GP306. This microcontroller receives the data from the accelerometer and the pulse oximeter and according them it acts directly on the diameter of the valve.

The *Leonardo* Platform's PIC has 2 modules UART, the UART1 and UART2 enabling serial communication with diverse peripheral devices through transmitting and receiving information by the pins UxTX and UxRX respectively. It has been used the UART1 that communicates through RS-485. At the begging of the development of this project the connection between the oximeter (RS-232) and the *Leonardo* platform was made through an RS-485/RS-232 converter.

However, as there were problems with the communication between them, it had to be implemented the iDemo Board, so, it was not necessary to use the converter. As said before, the signals from the iDemo Board are received directly by the *Leonardo* Platform's UxRX. The UART uses the communication protocol NRZ (Non-Return-to-Zero) and the most common form of communication is (8, N, 1) (see subchapter 4.3.5.1)

The PIC24 also presents an output compare, being the PWM mode used to control the electronic proportional valve's diameter (see subchapter 4.3.5.1).

Besides the PIC to have a key role, the *Leonardo* Platform has other important features. It has a panic bottom to allow for patient safety. If the system fails the patient presses the bottom and the system will open the valve completely and delivery 7 lpm of oxygen thus prevents that the patient feels the lack of oxygen in order to prevent developing serious consequences.

This fact happens according with PIC's 53 pin variation state (see attachment A1). This is, usually this pin has a logic value "0", that means it is a 0V and acts as output. When the button is pressed, this pin is 3.3 V and it acts as an input, which causes transmission of information to the PIC (see attachment B). According with this information and another configurations the PIC acts on the valve, allowing it's opening completely hence delivery of 7lpm.

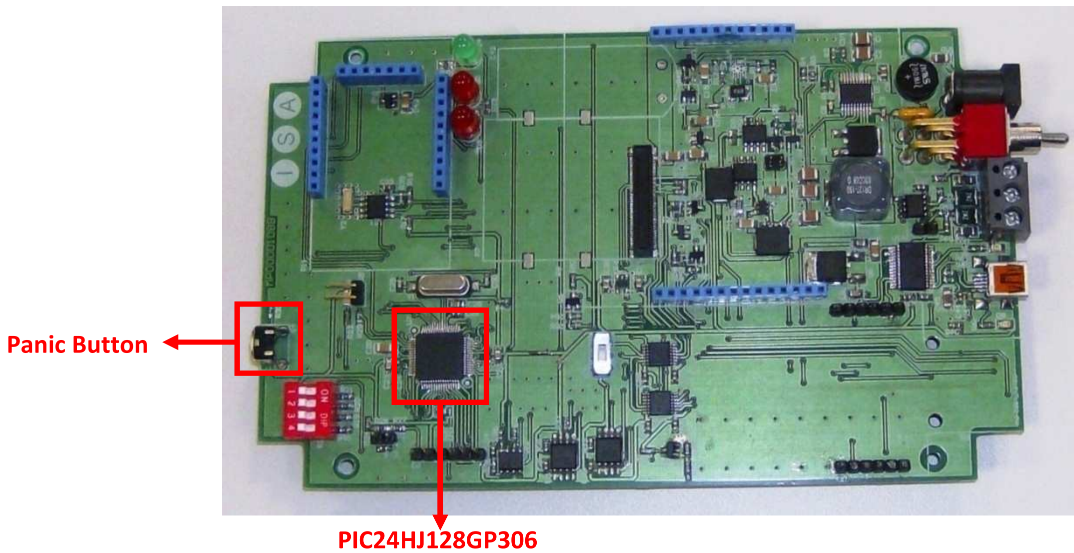


Figure 5.6. *Leonardo* Platform.

5.2.1.3. Sensors: Pulse Oximeter and Accelerometer

- **Pulse Oximeter**

In order to incorporate a more appropriate oximeter to the requirements of the project, mentioned in the previous paragraph, several options Original Equipment Manufactured (OEM) were analyzed having arisen as the solution the OxiLink oximeter™ 3150 from the company Smiths Medical.

Figure 5.7. *OxiLinkTM,* from Smiths Medical.

The OxiLink™ OEM oximeter has incorporated a micropower board and it interfaces with a host using a single asynchronous serial channel by RS232 communications.

It provides spot checking or monitoring of a patient's %SpO_2, pulse rate, signal strength, bargraph, plethysmogram and status bits data. In addition to averaged SpO_2 values, this device transfers instantaneous SpO_2 values which may be useful in a variety applications (in sleep study applications for example) presenting a good performance. This oximeter can be used with neonatal, pediatric, and adult patients [33].

Theory of Operation

The OxiLink™ uses two wavelengths of light, one red with 660nm and one infrared with 905nm to determine the vital signs. This lights passes through the tissues and the signal strength resulting from each light depends on several factors. These results allow to obtain the SpO_2 [33].

Since it was not possible a direct connection between the oximeter and the *Leonardo* Platform, it was used an iDemo Board to connect with it through a standard RS-232 communication. The iDemo works as an intermediate link to enable the data transmission in the format that the PIC24 can receive and process them.

To ensure proper connection between these two devices, it had to make some changes.

Connection between the oximeter and the iDemo board

In order to make the connection between the oximeter and the iDemo Board, it was necessary to establish some change on the pin out of both outputs. This is because the plugs of the oximeter and the iDemo Board are the same, so the connection between them cannot be possible. As was mentioned on the chapter about UART module, the cable that allows connection between two devices has to provide links in parallel, which does not happen in this case, avoiding the connections between them are correct.

Previously, it was necessary to study the DB9 connectors of both devices to make the links correctly. The tables below demonstrate the functions of each connector pins [33].

Pin descriptions:

- **RS-232 DB9 Connector – Oximeter**

DB9 Pin	Description
1	Not connected
2	Oximetry data output – TX
3	Negative supply voltage – V^-
4	Power Input – V^+
5	GND
6	Not connected
7	SYNC
8	Not connected
9	Optional Power Input

Table 4.1. RS-232 DB9 Connector – Oximeter

- **RS-232 DB9 Connector – idemo Board**

DB9 Pin	Description
1	T2out - DCD
2	T3out - TX
3	R2in
4	R1in
5	GND
6	T5out
7	R3in
8	T4out - CTS
9	T1out

Table 4.2. RS-232 DB9 Connector – iDemo Board

After knowing the descriptions of the pins, it was possible to make the connections between them. So, the following table shows the connections made between pins of both connectors.

Oximeter Pin	iDemo Board Pin
2	3
3	2
4	1
5	5
7	8

Table 4.3. Connections between oximeter and iDemo Board pins.

The following figure shows the connections made between the oximeter and the iDemo Board.

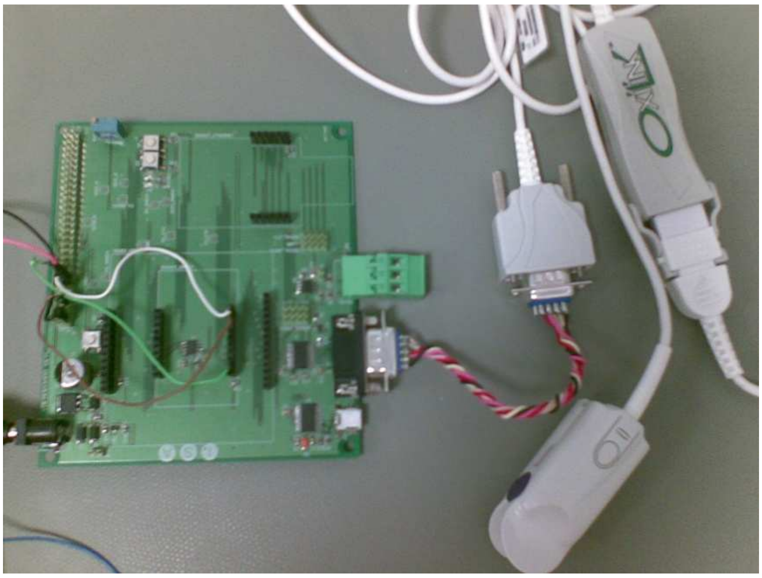


Figure 5.8. Connection between iDemo Board and oximeter.

After made the connection between the iDemo Boaard and the oximeter and in order to establish the data transmission with the microcontroller (PIC24) of the *Leonardo* Platform, it is necessary to convert the RS-232 levels down to TTL, as it was mentioned before. The ICL3238 device from the iDemo board has been selected for this purpose. The principle of receiver's operation is explained in the following paragraph.

Operation mode of ICL3238

As was mentioned above, it is used a RS-232 driver (ICL3238), an integrated capable of converting the +/- 15 volt typical of RS-232 interface to signals TTL of 0/3.3 volt compatible with the ports of the PIC.

Analysing the iDemo Board – Module RS-232 in the attachment A2, it is possible to note that the transmission signal from the oximeter enters in the pin 3 of the iDemo Board's connector and consequently it is connected to pin 9 of ICL3238. In accordance with this connection, two situations can be possible:

- The pin 20 (output) of the ICL will present a signal of 0V when the pin 9 (input) is +15V (this voltage comes from the RS-232 signal).

- The pin 20 (output) of the ICL will present a signal of 3.3V when the pin 9 (input) is -15V (this voltage comes from the RS-232 signal).

After converting the signal to TTL, it is sent to the microcontroller's *Leonardo* Platform by UART1 module, thus it is possible to receive, in real-time, data corresponding to the blood oxygen saturation of the patient. The data is sent from the oximeter at a baud rate of 4800, 8 bits, one stop bit, and no parity [33].

Data analysis obtain by the oximeter

The analysis of data obtained by pulse oximetry parameters and using pre-established by the doctor, is made by the microcontroller. This system will ensure a flow of oxygen delivered by the valve suitable to the needs of the patient.

The values of oxygen saturation in healthy adult are between 90 - 99%. However, for individuals with respiratory failure the range of saturation values changes [1].

Usually, patients that make oxygen therapy with values of SpO_2 like 90% the oxygen volume is constantly delivery with a flow rate of 2 lpm. However, with this system, this flow could be regulated through levels, i.e., defining intervals of oximetry and correspond them to volumes of oxygen flow.

Therefore, if the SpO_2 decreases (state of hypoxia, below 90%), the system should increase the flow of oxygen per minute. In situations where it is set the administered 2 lpm and the SpO_2 of the patient increases, the rate of O_2 is set in the same proportion and the system should decrease.

To make sure that the patient's finger is well placed in the oximeter and that there are no errors, exists a luminous indication to alert the user. A LED (*Light-emitting diode*) is implemented in the *Leonardo* Platform to indicate that everything is in order. It is has two states: if the LED is "off" it means that everything is going the way it is supposed to and the SpO_2 is read, if the LED is "on" the system lets the user know that it is ready to start reading the signal as soon as it detects that the patient is set with is finger placed in the sensor. By doing so, the system will deliver constant oxygen flow, i.e. the flow will be 2lpm, avoiding future consequences for the patient.

- **Accelerometer**

The accelerometer to incorporate is the ADXL330 from Analog Devices. This is a tri-axial accelerometer, with a range ± 3g, which allows extracting information about acceleration in both static (such as gravity) and dynamic (such as movement, shock and vibration). This device is connected by cable by tree analog inputs. However, the incorporation of the accelerometer in the system will be performed only at a later stage of work [34].

Figure 5.9. *ADXL330,* from Plux. From [34]

5.2.2. Hardware's Actuation Module

This module is responsible for the hardware's actuation, i.e. actuation on electronic proportional valve.

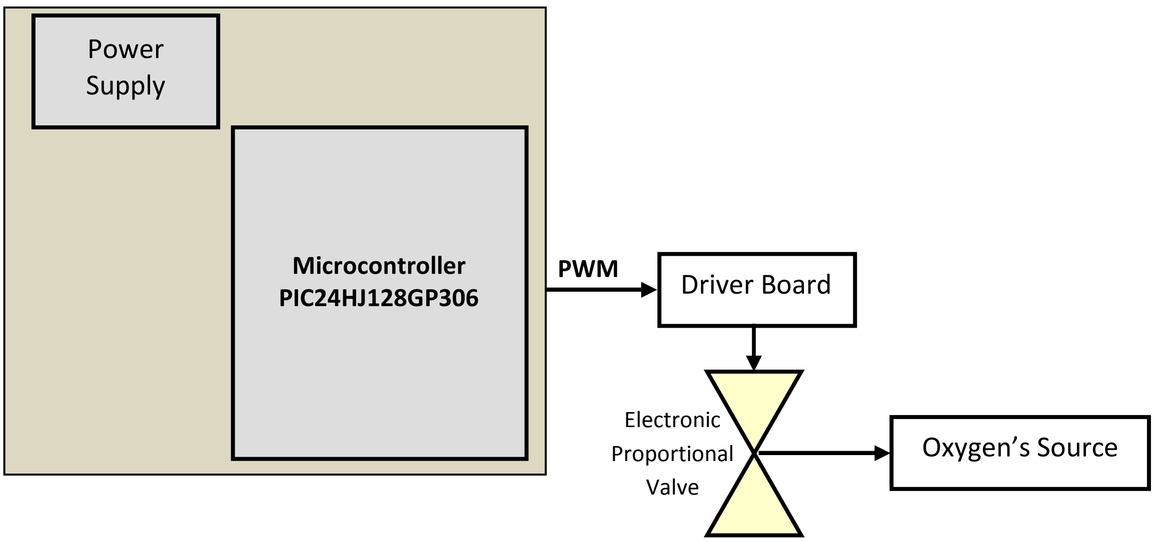


Figure 5.10. Illustrative scheme of the transmission module.

5.2.2.1. Electronic Proportional Valve [35]

The valve has a key role in this project and the electronic proportional valve miniature with a driver board, from Kelly Pneumatics was chosen. With this equipment, it can control the oxygen flow by making it suitable for the needs of each patient. This device is controlled by the feedback given by accelerometry and by oximetry from the patient.

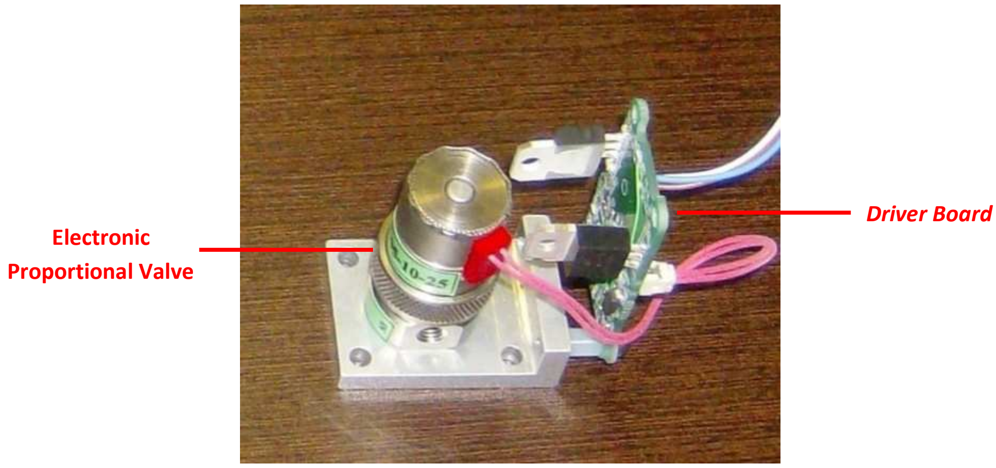


Figure 5.11. Electronic proportional valve in miniature with a driver board, from Kelly Pneumatics.

The electronic proportional valve is specially calibrated by the manufacturer in accordance with the requirements of the system. The physiological range of interest for the flow of oxygen is 0-7 lpm, so the valve had to be produced with this range of flow allowed. The voltage on the valve is also calibrated and it varies between 0 and 3.3V that suit the range of the flow mentioned.

To connect to the ports of entry and exit of the valve it has been to acquire two adapters advised by the supplier of the valve. The following figure shows the valve with the adapters connected. One allows the connection by a pipe to the oxygen source and the other allows the connection of the valve to the patient, so that it can receive the oxygen that the valve charge.

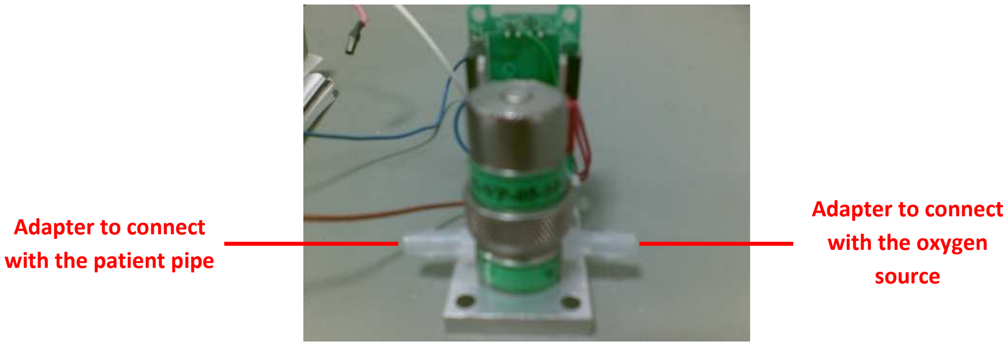


Figure 5.12. Electronic proportional valve in miniature with the adapters.

The driver board generates a current (see subchapter 5.2.2.2) and the variation of this in the solenoid causes a change in the diameter of the valve's orifice hence a variation in the rate of oxygen.

When there is no current applied to the solenoid, the valve is closed (normally closed proportional valve). If the solenoid current increases, the oxygen flow increases, and, if the solenoid current decreases, there is a decrease in the oxygen flow. This relation is approximately linear, i.e., the oxygen flow is proportional to the inlet current.

To obtain the linear regression for the relation between the output flow (oxygen flow needed) and the voltage it was informed by the valve manufacturer three calibration points: the oxygen flow starts at 1 volt, is 7 lpm at 3.3 volts and is 12.5 lpm at 5 volts. This linear function allows calculating the correct voltage to control the driver board and the valve. From this voltage, there is a % of duty cycle associated to use in Pulse Width Modulation, made by the PIC's PWM mode. It is then necessary to calculate the value of the PWM mode's register (OCxRS) relatively the period of the timer.

5.2.2.2. Driver Board [36]

The driver board is the hardware for the valve's actuation. It is powered with 12 VDC from *Leonardo* Platform and is specially developed by Kelly Pneumatics, forming an integrated system.

The driver board, and consequently the valve, is act by the PIC microcontroller's PWM module, which varies the voltage output. However, the voltage control is not as rigorous as the current control, due to variations caused by the temperature increase.

With the passage of current, there is an increase in temperature (Joule Effect) causing a decrease in resistance. Therefore, is necessary to increase the voltage to maintain the same current and hence the same output. Consequently, the driver board converts the input signal, a voltage signal, in an amplified current output proportional to the voltage received. So, the valve operates in current actuation, compensating the variations caused by the Joule Effect.

In this manner it is possible to vary the valve's diameter and consequently the flow of oxygen required.

Figure 5.13. Driver Board, the hardware actuation of the proportional valve, from Kelly Pneumatics. From [36]

5.2.2.3. Oxygen's Source

As was mentioned, there are presently three oxygen sources available for the oxygen therapy. About 80% of the market for oxygen sources regards to oxygen cylinders gaseous; the remaining 20% regards to the oxygen cylinders liquid and oxygen concentrator. The system implemented has to incorporate any of these systems. However, in the future there will be a tendency to replace the oxygen cylinders gaseous to the liquid systems and concentrators. The portable concentrators, being cheaper, may have tended to dominate up until now; however, it is advantageous the use of the oxygen cylinders liquid because it has both a higher rate of oxygen's purity and a portable system.

Figure 5.14. Oxygen's source: oxygen cylinders gaseous; oxygen cylinders liquid; oxygen concentrator. From [4]

Chapter 6. Firmware Design

This chapter presents the work carried out by the student during the project progress. The previous chapters are considered intermediate steps that have in common the objective to support the development of the Module for signals' Acquisition Firmware. Only the firmware for the vital data's acquisition through the oximeter was developed, since, as it was already mentioned, the accelerometer will be incorporated later. This chapter presents the main aim of this project – Firmware Design – that was the last stage to be performed after the acquisition of all knowledge necessary to achieve it. Here, there will be mentioned concepts discussed above that will only be fully understood by the reader if it is familiarized with the information presented in the previous chapters.

As the reader may have understood, for the development of this chapter it was necessary to program a microcontroller (PIC24). The firmware developed allows the microcontroller to receive the patients' data saturation from pulse oximetry, and it also allows processing them according to a pre-set parameters and then sending a response corresponding to the quantity of oxygen (lpm) that is necessary to delivery to a patient according with the reading of saturation previously made.

So, the following chapter is divided into two sections. The first section provides relevant information about the principal settings of the modules that constitute a microcontroller defined by the programmer during the firmware development. They include the UART module, oscillator and interrupts.

The second section is subdivided into two more parts. First part presents the firmware acquisition and filtering of the vital signs obtained by pulse oximetry and the second part presents the firmware for processing these data.

6.1. Microcontroller's Modules Configuration

As mentioned in subchapter 4.3 the microcontroller is made of several modules, each with its specific function.

In order to configure the program for reception and processing the data from the oximeter, it has to be reviewed the modules of the microcontroller (PIC24) that the actions above are carried out correctly. These settings are made according to the registers of each module. These registers and examples are given in attachment B for better understanding of the reader.

The oscillator is configured according to the source clock and oscillator characteristics. Its bits' configuration will be made through the specification of FOSCEL (selection of the oscillator source) and FOSC (configuration of the oscillator) registers. *Special function registers* allow the control of the execution time and the state of the oscillator. They are the register OSCON (oscillator's control), the CLKDIV (doze mode control, the selection of PLL pre and postscaler and postscaler's FRC), the PLLFBD (selection of the PLL divider) and the OSCTUN (attenuate frequencies of FRC).

In short, the programming of the PIC oscillator was made as follows:

➯ Initially the PIC's PLL feature is disabled. Otherwise the microcontroller cannot start it's execution.

➯ When the PIC is running the PLL settings are changed

⇩

- PLL values (PLLDIV, PLLPOS and PLLPRE) will be within the range, and the PLL feature is enabled.

⇩

- The PIC will now run in the required clock speed.

The I/O ports are associated with main registers. Through these registers is possible to configure pins' direction and access its logical state.

The UART module and its correspondent interruptions also require some configuration. The following diagram shows the UART and interrupts modules' operation.

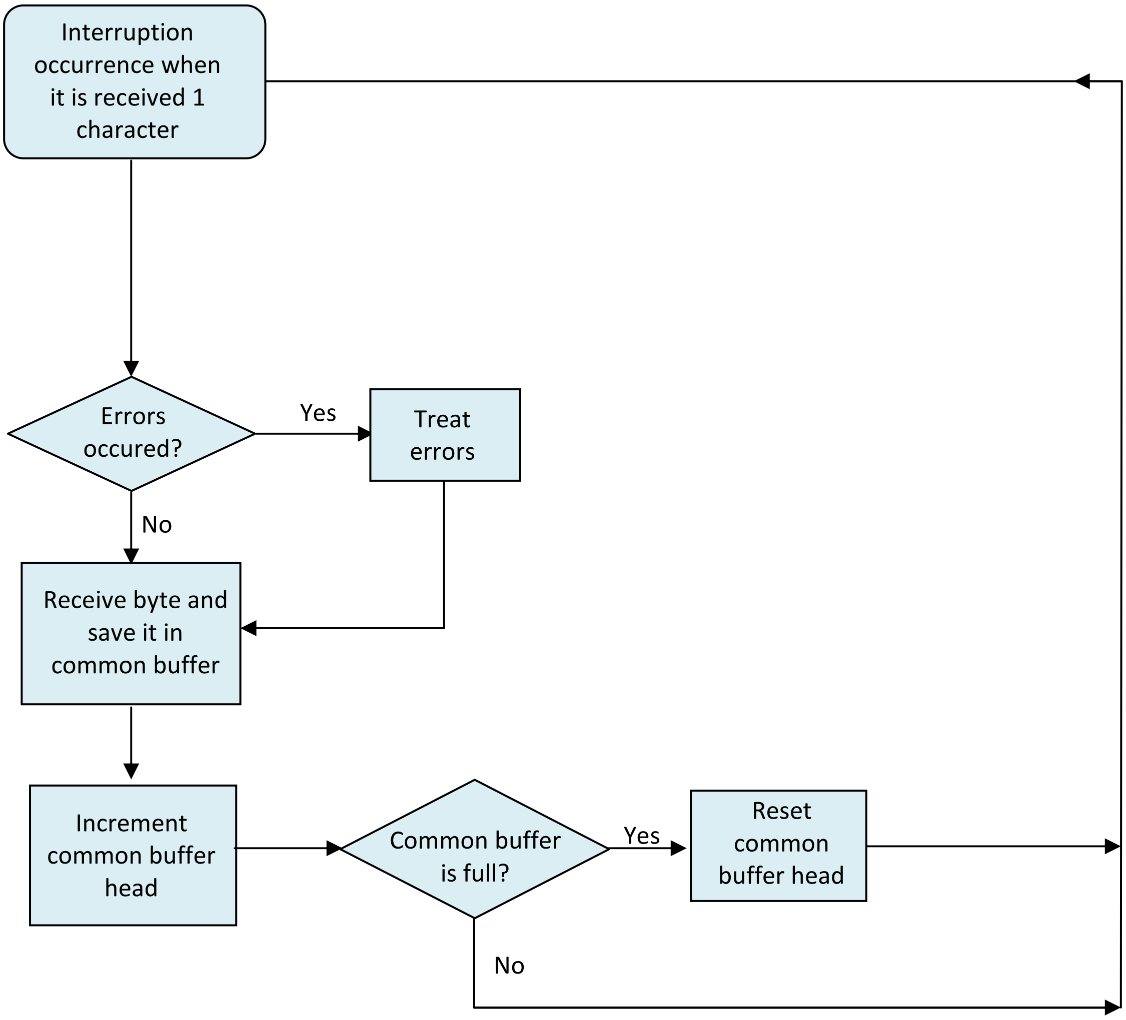


Figure 6.1. Flowchart of the UART interrupts handling routines.

Each time one byte is received, the system generates a UART interrupt and its correspondent routine is executed. This routine consists in error verification and data saving. The routine also verifies for buffer overrun. When the interrupt routine ends, the system continues to run the code exactly where it was left off.

6.2. Firmware Development

After reading the previous chapters as well as the acquisition of information reported, it is possible to understand easily what is treated in the following sections. That is to say, that the concepts that will be referenced below, should be familiarized for an easier understanding of this issue.

With the knowledge of the oximeter operation mode, it was possible to establish certain rules for the development of the firmware to be done effectively.

Next subchapters will present the acquisition, filtering and processing firmware.

6.2.1. Acquisition and Filtering Vital Signs Firmware [33]

As was mentioned in the specifications of the oximeter, this device provides several parameters to be used in a variety of applications. However, for this project the parameter to be determined is only the value of patient oxygen saturation in real-time. So, it was necessary to develop a firmware program capable to receive all the data that is provided by the oximeter and then implement the filters to get just the value of saturation intended. The following algorithm demonstrates the system management to perform the steps above.

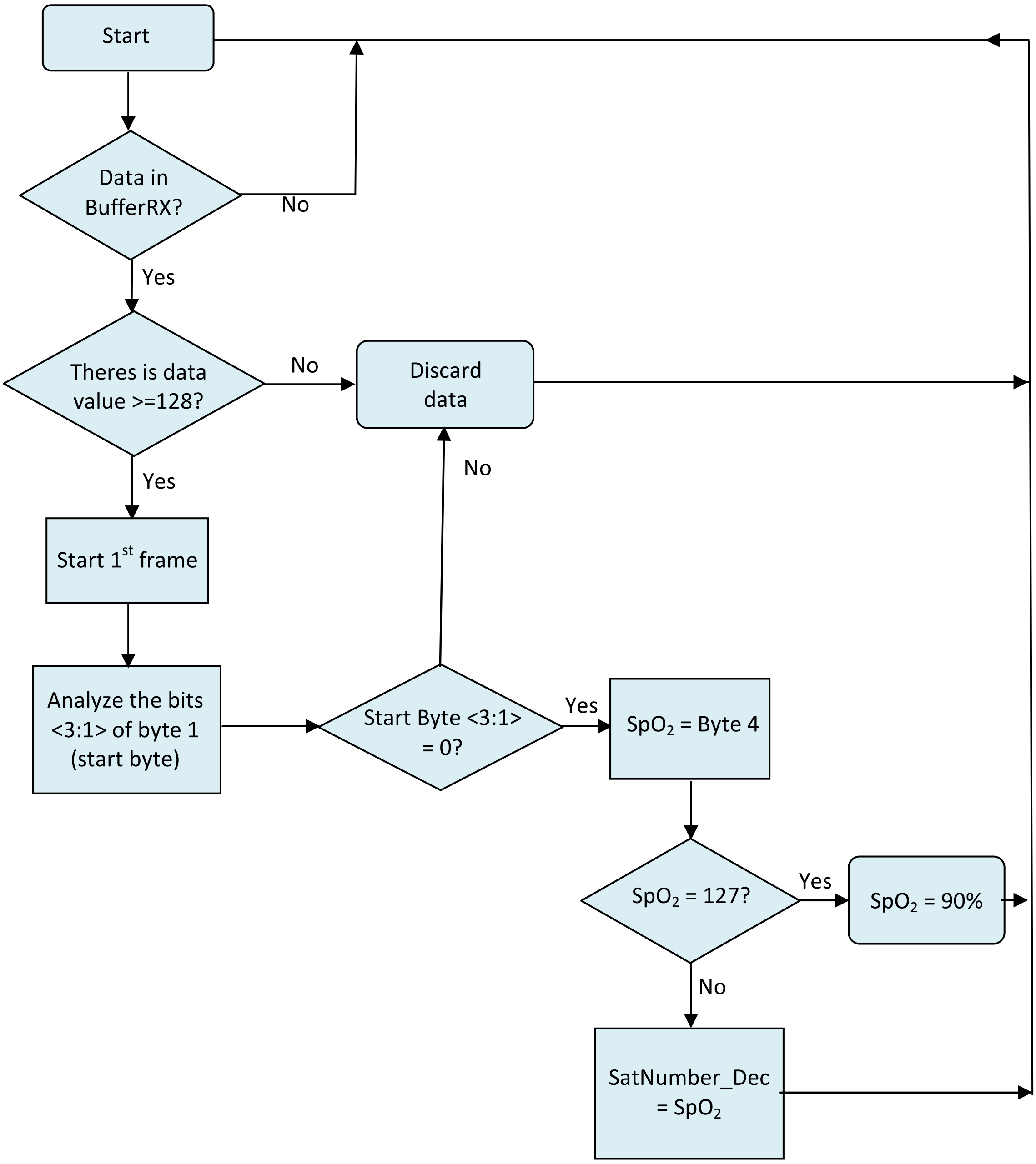


Figure 6.2. Flowchart of the data's filtering process obtained by pulse oximetry.

In this way, the first step of this procedure consist on examine each data received in order to verify that there are no errors in saturation readings and verify the existence of data in the buffer. The data of the buffer should be analyzed (see attachment B). If the decimal value of each one is less than 128, the discard data; if the decimal value of the data analysis is higher or equal than 128 there is the beginning of the first frame (consisting of four bytes). The resulting frame has to be examined in detail to obtain the data of interest for the project.

Frame

Byte 1	*Start Byte*
Byte 2	“Don’t’ care”
Byte 3	Signal quality
Byte 4	SpO_2 value

Table 6.1. Frame structure. Adapted from [33]

The analysis of the bits’ value <3:1> of the Start Byte indicates the contents of the byte 4, i.e., the value of oxygen saturation (see attachment B).

On the one hand, if the bits’ content <3:1> is the value 0 means that the byte 4 is in the presence of saturation average value. In this case this value is stored in a variable (e.g. SpO_2) to be examined again (see attachment B). On the other, if the bits’ content <3:1> is different from 0 the system discard the data present in the byte 4 because it does not match the information that is necessary (see attachment B). So, in this case the system goes back to check the <3:1> bits until it finds the value 0.

When is detected the saturation average value from the byte 4, this value has to be examined again. This is, according to the manufacturer's instructions (see attachment B), whenever the detected value of saturation is different from 127 (decimal value) it corresponds to a real saturation value, so it can be stored and processed in order to get the value of the corresponding oxygen flow. If the value of saturation found is equal to 127, it should be eliminated, once it may mean that the patient’s finger is not placed in the oximeter or that there is any reading error. So, in this case, what the system does is to consider a value of saturation of 90% by default and deliver oxygen flow of 2lpm to prevent serious consequences for the patient.

6.2.2. Processing Firmware

In the next paragraphs, using the information previously acquired, it will explain the operation of the processing firmware developed.

The algorithm, with the configuration defined in the following flowchart, was developed for the reception and processing of data from the oximeter. With the system implemented, the oxygen flow is regulated by levels, i.e., there are defined intervals of oximetry that result in certain volumes of oxygen delivery.

These intervals were defined by Doctor Moita. According to the physiological study conducted by him and his professional experience, he says that these changes (+/- ½ lpm if the SpO_2 decreases or increases 2%) are enough to ensure a state of adequate oxygenation.

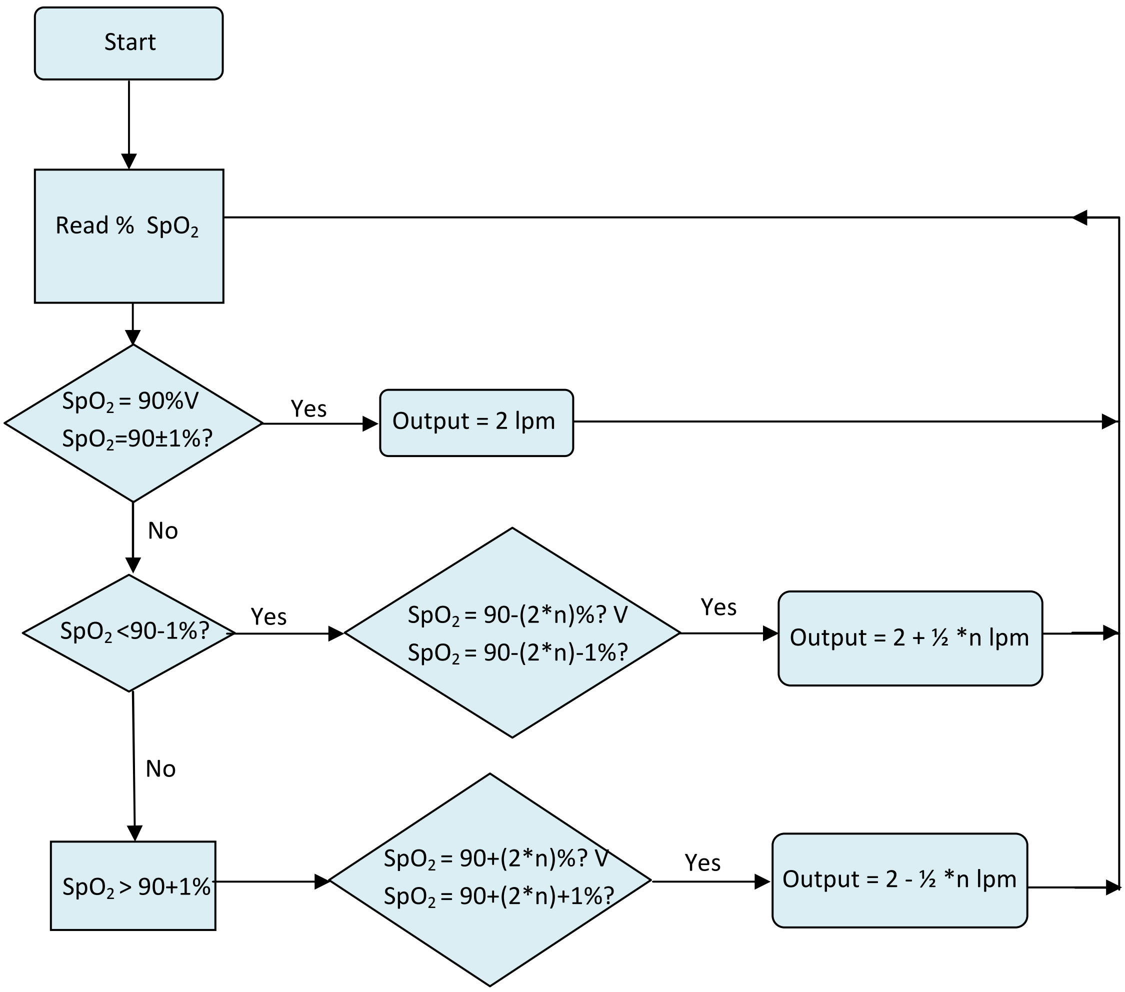


Figure 6.3. Flowchart of the control oxygen flow according to SpO_2 data from pulse oximetry (n is the number of times that the SpO_2 range 2%).

In this way, the microcontroller was programmed to operate according to different situations. The reading of SpO_2 is done on a continuous mode to allow instant verification of the patient's arterial oxygenation status and a real adjustment of the O_2 volume. So, by knowing the value of the patient's saturation, in %, obtained through the acquisition and filtering firmware, it is necessary to determine the flow of oxygen to deliver in lpm.

The standard value of saturation is 90%, thus the analysis of data is done in comparison with this value.

For values of SpO_2 in the range of 89-91% the delivery of oxygen is 2 liters per minute (lpm).

If occurs a fall in SpO_2 of 2% or 2% +1, the system should increase the rate of ½ lpm multiplied by *n*, a number of times that decreases 2%, i.e. the output becomes 2lpm basal + ½ * n.

In situations where it is administered 2lpm and the SpO_2 of the patient increases, the system should reduce the delivery of O_2 in the same proportion as in the case of the situation of decreasing saturation.

In addition to the levels defined (decrease / increase of 2% and 2% +1 in SpO_2 correspond equal oxygen's volumes) to permit an adequate real delivery of oxygen to the patient's SpO_2, they also allow greater savings oxygen, avoiding its waste.

In order to prove this study, it was elaborated the following tables, so that the flow is regulated in levels, i.e., defining intervals of oximetry and corresponding them to certain oxygen's volumes to delivery.

n	SpO_2 = 90-(2*n)%	SpO_2 = 90-(2*n)-1%	Output = 2 + ½ *n (lpm)
0	90	89	2
1	88	87	2,5
2	86	85	3
3	84	83	3,5
4	82	81	4

Table 6.2. Registration of a SpO_2 values decreasing and values of the O_2 volume corresponding.

n	SpO_2 = 90-(2*n)%	SpO_2 = 90-(2*n)-1%	Output = 2 + ½ *n (lpm)
0	90	91	2
1	92	93	1,5
2	94	95	1
3	96	97	0,5
4	98	99	0

Table 6.3. Registration of a SpO_2 values increasing and values of the O_2 volume corresponding.

Chapter 7. Final Tests and Results

The inclusion of this chapter is important for the conclusion of the project. It presents the results of the tests carried out with the prototype implemented. This chapter is divided into two sections. One section presents the acquisition module operation tests and results; the other section presents the tests made with the hardware's actuation module and results.

7.1. Acquisition and Processing Module Tests

This section presents the results that are obtained by the acquisition and processing module regarding to the person's values of saturation and the oxygen flow relative to the data acquired.

In accordance to the algorithm previously shown and explained, tests were performed in a laboratory to verify the credibility of that algorithm. For that, were realized several tests, based on two key cases. In the first case, it was guaranteed the correct presence of the person finger in the oximeter sensor to determine the percentage of oxygen saturation. In the second case, the individual did not put the finger on the oximeter. Another possible situation was also tested, this is when the person is in a state of panic. So, when the person presses the panic button, available on the *Leonardo* Platform, the system responds by providing the maximum amount of oxygen that is capable (7lpm). The following figures show the results obtained and it will be briefly explained each step in each case.

- **First case: the patient's finger is well placed in the oximeter**

First it was necessary to verify the presence of data in the buffer. Later, it is necessary to find the first frame and review their values to obtain the correct value of saturation (see chapter 6).

In examining each value in this buffer, the system first checks if each data is a decimal value greater or equal than 128. If the system find a value in this condition means that starts the first frame (4 following values). The system analyses the frame in issue. If it finds the saturation value intended the system passes to the next step, otherwise return to new research and new data analysis.

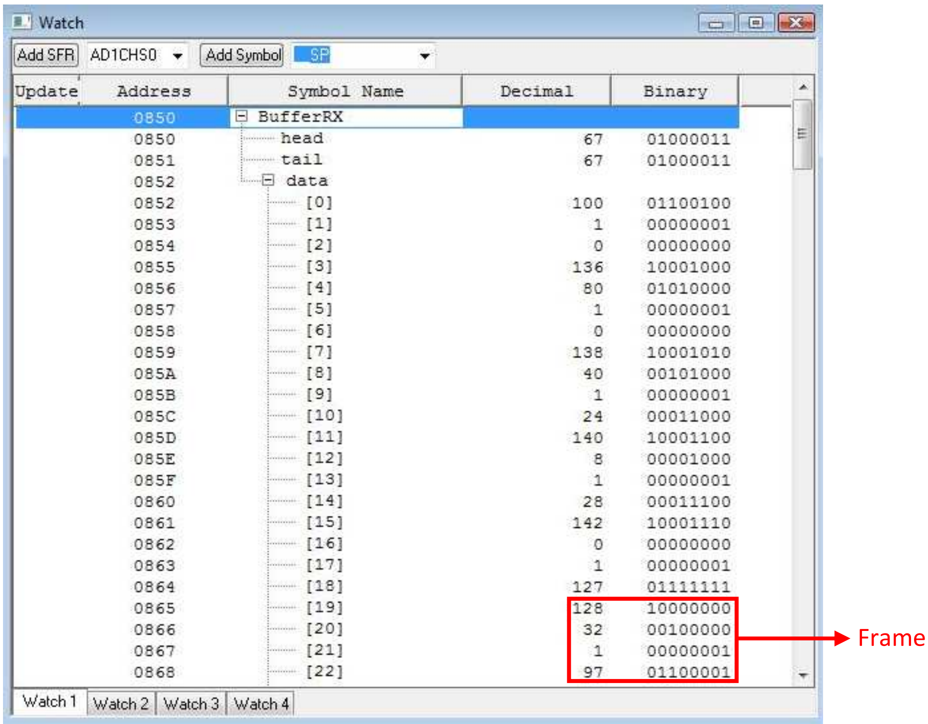


Figure 7.1. Demonstration of a data presence in the buffer and the first frame in the first case.

In the case shown above and analyzing the data from the buffer, it was concluded that the person that made this test had a saturation of 97%. This value is stored in the variable SatNumber_Dec (SatNumber_Dec = SpO_2). According to this data, it is calculated the corresponding oxygen flow (0.5 lpm in this case) and a value of voltage 1.1643 V) that will do the valve actuation to provide the respective flow.

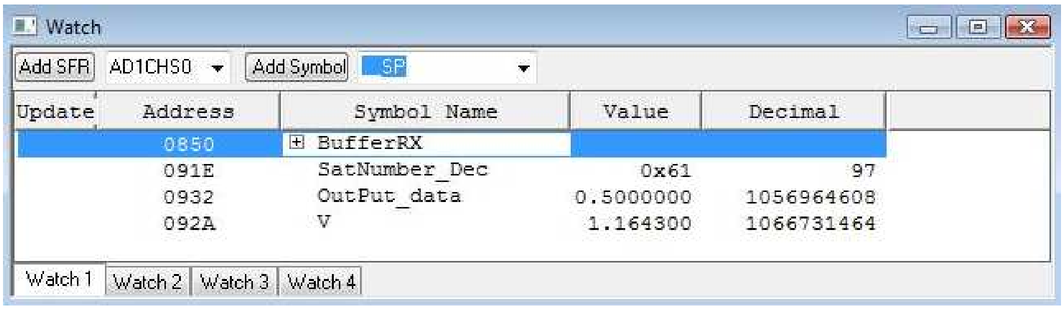


Figure 7.2. Demonstration of the saturation value obtained and the respective output and voltage in the first case.

- **Second case: the patient's finger is not placed in the oximeter**

In the second case, the operation of the algorithm is the same that the previous, with the difference that in this case what the system has to detect is, in accordance with the manufacturer's oximeter datasheet, the value 127 in the buffer, which indicates that the patient's finger is not placed on the oximeter or it is badly positioned.

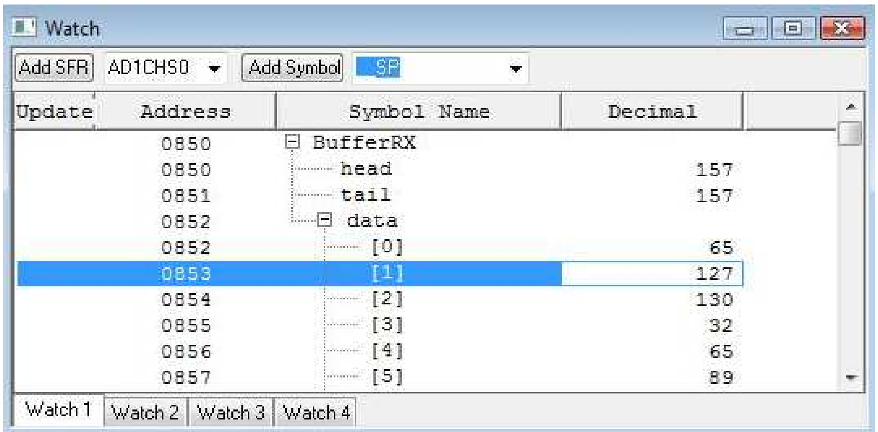


Figure 7.3. Demonstration of a data presence in the buffer and the 127 data concerning on the condition of the second case.

So, in this case the system places the value of saturation equal to 90% (baseline) for the oxygen flow to be 2lpm. The figure for this case shows the data above mentioned, as well as the value of voltage corresponding to allow the proper valve opening and thus the output is performed efficiently.

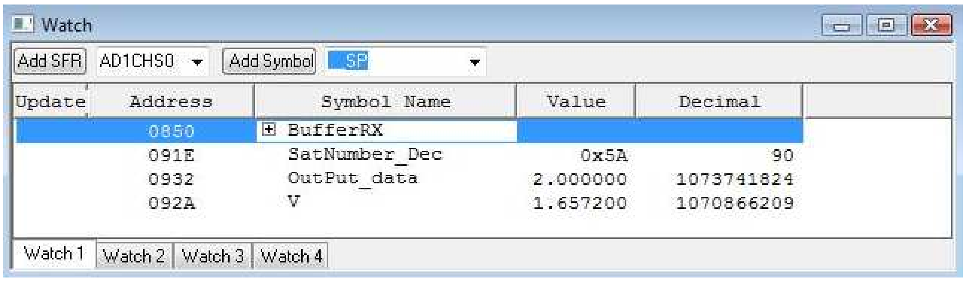


Figure 7.4. Demonstration of the saturation value obtained, the respective output and the voltage value in the second case.

In the last situation mentioned, when the panic button is pressed by the person, the system will provide a voltage of 3.3 V which makes the valve to open in totality and then delivery a oxygen flow of 7lpm, according with the manufacturer's valve information (see subchapter 5.2.2.1).

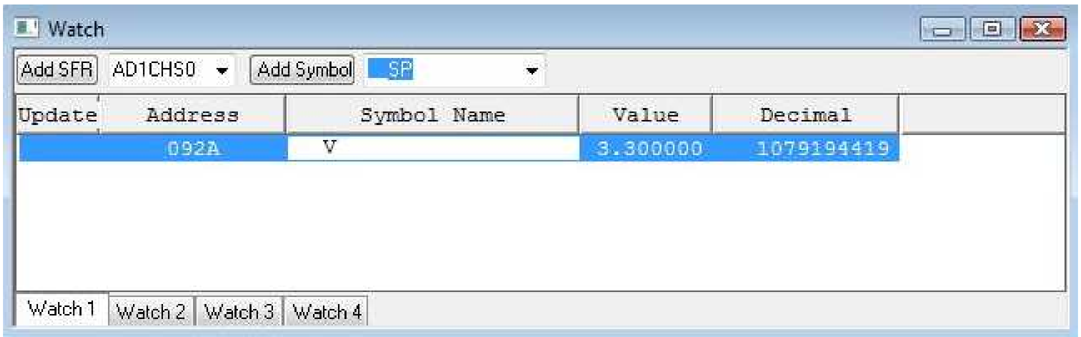


Figure 7.5. Demonstration of the voltage value obtained when the panic bottom is pressed.

7.2. Hardware's Module Tests

This section presents the results that are obtained by the hardware module. According to the data coming from the acquisition and processing module (SpO_2, output and v), the hardware module acts the proportional valve's diameter for the oxygen flow to be carried out effectively, thus ensuring that patients receive the amount of oxygen needed.

To verify the proper functioning of the prototype implemented to control the flow in real-time, tests were performed in a hospital. The tests were carry out in two phases.

In the first phase the tests were done to a patient with respiratory failure. From them it was obtained only saturation values, since the purpose of these tests, which was to measure and verify the output values corresponding to oxygen saturation measured with a flow meter, it was not possible. This fact was due that the adapters was not available to connect the proportional valve's output and input. So, the oxygen charged by the source on the wall, could not pass through the valve and because this, the oxygen could not be measured by the flow meter.

In the second phase the tests were made to a healthy individual, from them it was obtained saturation value and the corresponding output values.

According to the above problem, to obtain a final prototype assembled in totality, it was used a flexible tube of 4 mm internal diameter, supplied by the company GASIN, able to connect to the adapters of the proportional valve. After, it was made the assembly of the equipment already purchased: *Leonardo* Platform, iDemo Board, oximeter, valve, driver board and also the adapters.

In the adapters of the valve's ports were connected the tubes. In the valve's gateway that was connected the tube of 4 mm internal diameter which comes from the hospital oxygen source, i.e. the oxygen supply ramp. As shown by the following figure, the source has a regulator for establishing the output rate.

Figure 7.6. The oxygen supply ramp with the cannula placed at the exit.

In the valve's output port was connected the flow meter. This device is also provided by GASIN and has a function to check the flow of oxygen delivered, by a sphere placed in a graduated tube. These connections ensure the passage of oxygen from the source to the valve and from the valve until the flow meter.

Figure 7.7. The electronic proportional valve with the tube (blue) placed in the gateway and the flow meter in the output port (white).

Figure 7.8. The flow meter.

After the assembly of the prototype, the tests started at the hospital.

- **First phase: SpO_2 measurement and validation of the results**

As was mentioned before, the first phase of testing were not used the adapters, but all other equipment was used and assembly.

The patient, who submitted to the tests of the prototype, was a male person, with 69 years old, with DPOC whose only treatment was carried out overnight with a constant flow of oxygen 2lpm. During the day the patient does not oxygen therapy and normally he presents an oxygen saturation values between 90% and 94%.

It was proceeded to perform the tests, by running the program developed. It was acquired the person's saturation value by pulse oximetry and it according them it was confirmed that the saturation value of this patient ranged between 93% and 94%. If the patient did not have the finger placed correctly in the oximeter the saturation value was 90% as would be foreseen.

Regarding to the oxygen flow corresponding to these values of saturation, it could not measured them with the flow meter, since that the adapters was not available as it was mentioned above, there was leakage of oxygen. This fact did not allow the oxygen, coming from the ramp wall, could get to the valve and consequently it was measured by the flow meter to the valve outlet.

In accordance to the results of saturation obtained, the doctor that accompanying the patient, Doctor Joaquim Moita, validated the tests, concluding that the firmware developed for the prototype implemented is able to receive correctly the saturation values of the patients.

- **Second phase: SpO_2/ oxygen flow measurement and validation of the results**

In the second phase of testing, it was used the proportional valve's adapters acquired by GASIN. At this phase, the individual, who submitted the tests, was a female person, with 23 years old and without any respiratory failure. These tests were performed in a healthy individual, since previously had been confirmed that the measured values of saturation are viable.

Thus, these tests were performed in order to verify if, based on data of SpO_2, it was obtained the value of oxygen flow correspondent, measured by flow meter.

Initially, by running the program developed, it was acquired the person's saturation value by pulse oximetry and according to it, was obtained the value of oxygen flow needed.

It was directed to the valve, the oxygen supplied by the ramp, but in a higher output value than the required by the individual once we want to test the valve delivering the right amount of oxygen and the way it handles the flow of the oxygen from the ramp. In the output it was verified that the oxygen flow respected the value determined by the acquisition module, which was obtained by the adjustment of the valve's output.

At a second stage, it was tested the prototype operation in case of the person's finger is not placed in the oximeter sensor. In this case, the result was positive, obtaining oxygen flow output of 2 lpm, as would be expected.

However, for these values to be respected it was necessary to provide a slightly higher flow than desired to the valve, once that at the output of the valve, the isolation between it and the flow meter was not the best.

So, in the absence of the finger and when 2 lpm was provided at the valve's input, it was obtained a slightly lower flow (about 1.5 lpm) at the valve's output.

This happens because the tube is not totally connected to the flow's meter adapter, this flow meter adapter is suitable to larger tubes. This was also the problem that was encountered in the acquisition of the adapters for the valve's ports.

In this way, it could be said that the measurement of the flow is not entirely accurate, being affected by the loss of oxygen and also by the user's reading on the analog flow meter (the position of the ball on the flow meter graduated scale is not very clear). However, in accordance to the indications of the physician, the control of the flow doesn't need to be very exact nor to have high accuracy, so, the variation may occur in the range ± 0.5 lpm, according to the indications of Doctor Joaquim Moita.

The efficiency of the system was subsequently tested combining the two types of test, i.e. it was acquired the saturation's value of a healthy individual by pulse oximetry and according to it, it was obtained the value of oxygen flow (about 97% which corresponds to a flow of 0.5 lpm). Then the sensor was removed from the finger, detecting its absence, the system automatically increased the flow to 2 lpm.

In the presence of the results obtained, the system was in accordance with the expected by clinicians. The results of the tests made are show at the table below.

	Position of the finger	SpO_2	Output
1st Phase	finger placed on the oximeter	93% - 94%	–
	finger not placed on the oximeter	90%	–
2nd Phase	finger placed on the oximeter	97%	0.5 lpm
	finger not placed on the oximeter	90%	2lpm

Table 7.1. Tests' results in the hospital.

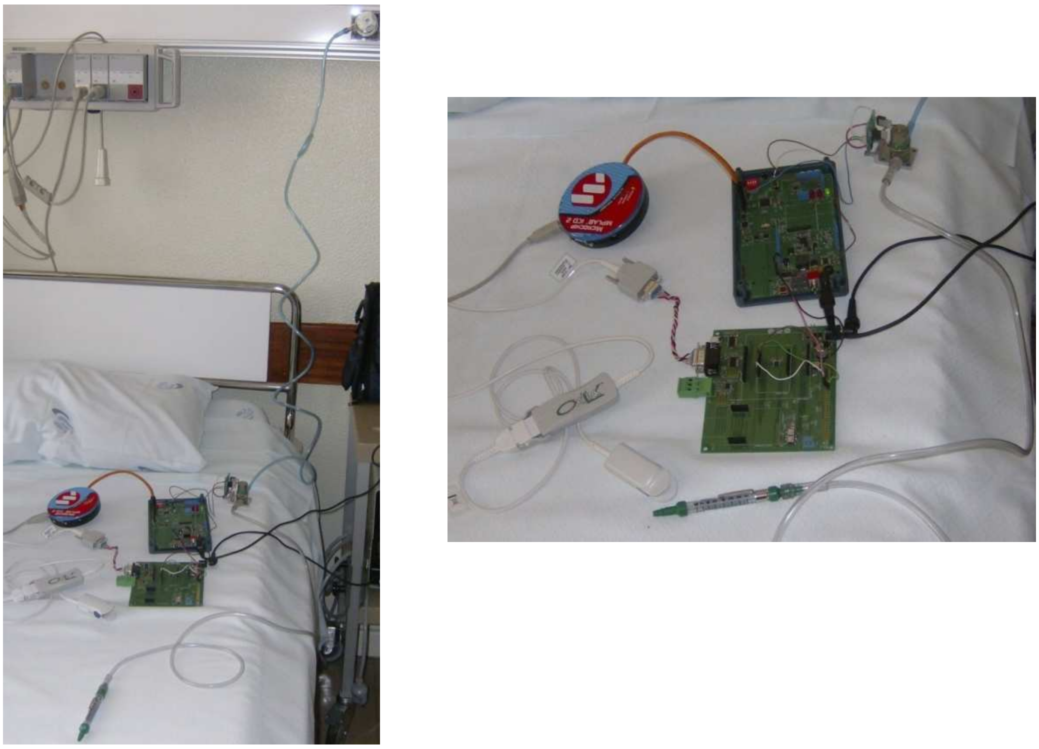


Figure 7.9. System's prototype, which is visible the Platform *Leonardo* with the MPLAB ICD2 connected, the iDemo board connected to the oximeter by RS-232 and the valve with the flow meter on the output port and the gateway cannula connected to the hospital oxygen's source on the wall.

Chapter 8. Conclusion

During this thesis was presented the development work during the past academic year in the Remote Vital Signs Monitoring with Specific Application to Respiratory Failure project. To achieve the initially objective proposed several intermediate steps were taken, such as knowledge acquisition, analysis of the project requirements, study of the system specifications and finally the firmware development.

This thesis has, as a conclusion, a brief summary of what has been developed until now and what can still be done in future work. Finally, there is a final assessment of the project's benefits for general society and also its future potential, the results that fully meet the objectives initially proposed and considerations of the project's influence on the student's personal development.

8.1. Project Status

The main objective of this project was to develop a system able to continuously monitor the vital signs of the patients with respiratory failure allowing them a constant oxygenation according to the vital data detected while allowing the mobility of the patient. This project is an innovative system having no connection with other projects previously developed. It appears to complement the gaps in the systems currently available on the market and thereby improving health care of patients.

This project represents an attempt to resolve the issue of improving the health care quality of the patients with respiratory failure by creating a system that will benefit not only medical professionals but also the patients, since it is a system that continuously monitors the patient's vital signs in real-time and it appropriates the amount of oxygen in accordance. This is achieved by the two modules implemented: acquisition and processing module; hardware's actuation module.

This system ensures an adequate oxygen therapy to patients that use this treatment, ensuring that in home care their physiological needs are pleasures, and on the other side it prevents the patient from staying in hospital.

8.2. Future Work

This section provides future development suggestions that can be studied, tested and possibly implemented.

With the proposed system it is intended for the patient to have greater mobility and flexibility. So, a task to be developed in future is to carry the prototype implemented for a portable system. For that, it is necessary to assess the energy consumption of the system and ensure the portability of it.

The accelerometer is a device widely used in medicine in the assessment of human movement and physical activity, providing the frequency and intensity of the patient's movements. This device will be a great asset to this system because the physical activity is directly linked with the consumption of oxygen, as so, the higher the activity of the patient implies higher consumption of oxygen by it.

First, it is necessary to make a mediation of the patient physical activity to obtain a range of default data. From this, it is established a correlation between the acceleration and the value of SpO_2, adjusting the therapy to patient's movement. It is then necessary to develop an algorithm capable of receiving data from the accelerometer, analyzes them and together with the data obtained by the oximeter, the algorithm sends an output appropriate to guarantee adequate oxygenation to the patient.

Another suggestion for future task is the incorporation of oxygen's conserving device, that allows to close the valve while the patient is in expiration process. Like this we can assure a more effective oxygen saving. A device similar to this one is the respiratory effort band which can also detect and identify phases of the respiratory cycle, allowing the system debiting oxygen only during inhalation of the patient. This is possible because the respiratory movements, of the chest or abdomen, stress the crystal and generate a small voltage proportional to the movement or tension in the elastic belt around the body.

8.3. Final Appreciation

This section will be considered different points of view as the final assessment of the project. From an engineering standpoint, this system is an innovative and revolutionary in the field of medical therapy for patients with respiratory failure. It is a very suitable for the treatment of chronic lung disease, there is no current system on the market similar.

From the clinical point of view, this system ensures both a continuous oxygenation of the patient according to their vital needs, monitored in real time and secondly a saving of oxygen, which prevents excessive costs and simultaneously improve the quality of care health. The results obtained by testing fully meet the original target at once from the patient's data obtained by oximetry it was possible to adjust the flow of oxygen (lpm).

From a personal standpoint, the final result of the project was positive for the student. On the one hand, it was forced to remember and review previously acquired knowledge on the other hand the new learning of other subjects was required to develop the project so that the student acquired more knowledge in other areas not yet explored. Beyond the academic level, the student had the opportunity to engage with professional World and it learn how to work in a team, how to combine ideas to develop an innovative and valid solution. Many difficulties have arisen over the project, which was overcome with the help and dedication of the entire team that was part of this project.

REFERENCES

[1] Fox, S. I. (1996). *Human Physiology.* 9th ed. New York: McGRAW-HILL.

[2] Pereira, A. C., Uva, A. S., & Cordeiro, C. R. (n.d.). *DIA MUNDIAL DA D.P.O.C. (DOENÇA PULMONAR OBSTRUTIVA CRÓNICA)*. Retrieved October 13, 2008, from ONDR: Observatório Nacional das Doenças Respiratórias: http://www.ondr.org/index.html

[3]Machado, M. C. (n.d.). *OXIGENOTERAPIA DOMICILIAR PROLONGADA.* Retrieved October 15, 2008, from UNIVERSIDADE FEDERAL DE SÃO PAULO - UNIFESP: http://www.unifesp.br/dmed/pneumo/pneumologia.htm

[4]Viegas, C. A., Adde, F. V., Paschoal, I. A., Godoy, I., & Machado, M. C. (2000, Dezembro). *Oxigenoterapia domiciliar prolongada (ODP).* Retrieved October 18, 2008, from Sociedade Brasileira de Pneumologia e Tisiologia: http://www.scielo.br

[5] Pilling, Jeffrey, & Michael C. (1999). *Ambulatory Oximetry Monitoring in Patients with Severe COPD.* 116;314-321, Retrieved October 19, 2008, from http://chestjournal.org/cgi/content/abstract/116/2/314

[6] Guyatt, Gordon H., Douglas A. McKim, Peggy Austin, & Robert Bryan. (2000). *Appropriateness of Domiciliary Oxygen Delivery*. 18;1303-1308, Retrieved October 20, 2008, from http://www.chestjournal.org/content/118/5/1303.full.html

[7] McCoy, R. (2000, January). Oxygen-Conserving Techniques and Devices. *Respiratory Care* , pp. 95-104. Retrieved October 27, 2008.

[8] Castillo, Diego, Rosa G., & Perez C. (2007). *Oxygen-Conserving Devices: A Forgotten Resource*. 43, 01. Retrieved October 31, 2008, from http://www.archbronconeumol.org

[9] The Emily Center. (2004). *Using Oxygen at Home*. Retrieved November 2, 2008, from http://www.phoenixchildrens.com

[10] Barker, A F, L W Burgher, & A L Plummer. (1994). *Oxygen conserving methods for adults.* 105; 248-252. Retrieved October 29, 2008, from http://www.chestjournal.org/content/105/1/248

[11] Tiep, B L, &M I Lewis. (1987). *Oxygen conservation and oxygen-conserving devices in chronic lung disease. A review*. 92;263-272. Retrieved November 6, 2008, from http://www.chestjournal.org/ content/ 92/2/263

[12] Silveira, C. (n.d.). *Capitulo 1_A Internaçao Domiciliar.* Retrieved October 20, 2008, from PósDesign: http://www.posdesign.com.br

[13] *Critical Care*. (2009). Retrieved Octobre 21, 2008, from Cookmedical: http://www.cookmedical.com/cc/

[14] UC Davis- University of California. (2009, June 9). *PROTEINS*. Retrieved June 11, 2009, from BIOLOGICAL SCIENCE 10V / NEMATOLOGY 10V: http://trc.ucdavis.edu/biosci10v/bis10v/index.htm

[15] Damas, Luís. (2002). *Linguagem C.* 7ª ed. Lisboa: FCA-Editora de Informática.

[16] Corteletti, D. (2006, Octobre). *Introdução à programação de microcontroladores.* Retrieved November 11, 2008, from http://amsfrancisco.planetaclix.pt/textos.htm

[17] Sistemas Industriais Inteligentes. (2002). *Curso de Linguaguem Computacional C/C++.* Retrieved November 11, 2008, from http://s2i.das.ufsc.br/

[18] Dornelles, A. (1997, November). *Fundamentos de Linguagem C.* Retrieved November 12, 2008, from Departamento de Engenharia de Computação e Automação: http://www.dca.ufrn.br/

[19] mikroElektronika . (2009). *PIC Microcontrollers*. Retrieved November 15, 2008, from mikroElektronika : http://www.mikroe.com/en/books/

[20] Hellebuyck, C. (2003). *Programming PIC Microcontrollers with PicBasic.* USA: Elsever Science. Retrieved November 16, 2008, from http://www.melabs.com/products/books/

[21] Tanzilli , S., G. Tiziano & N. Marcio. *Curso prático sobre PICmicro™.* Retrieved November 18, 2008, from http://www.ime.eb.br/~pinho/micro/apostila/pic/cursopic/brasiliano/index.htm.

[22]Canzian, E. *Comunicação Serial - RS232.* Retrieved November 18, 2008, from http://www.coinfo.cefetpb.edu.br

[23] Microchip. (2008). *Oscillator Data Sheet*.

[24] Microchip. (2008). *Output Compare*.

[25] Cox, T. (2004). *Migrating from Assembly to C for 8-bit Microcontrollers.* Retrieved November 19, 2008

[26] Microchip. (2007). *MPLAB® ICD 2 User's Guide*. Retrieved November 20, 2008, from http://ww1.microchip.com.

[27] Microchip. (2003). *MPLAB®ICD 2 In-Circuit Debugger/Programmer.* Retrieved November 20, 2008, from http://ww1.microchip.com.

[28] Microchip. (2007). Integrated Development Environment. Retrieved November 21, 2008, from http://www.acaltechnology.com/

[29] Barker, S., Hay, W., Miyasaka, K., & Poets, C. (2002, September 10). *Principles of Pulse Oximetry Technology:* . Retrieved April 18, 2009, from Oximetry.org : http://www.oximeter.org/pulseox/principles.htm

[30] Figueiredo, L. J., Gafaniz, A. R., Lopes, G. S., & Pereira, R. (2007, December 19). *Aplicações de Acelerómetros.* Retrieved October 24, 2008

[31] Microchip. (2008). *UART Data Sheet.*

[32] Benkovic, S. (n.d.). *PWM*. Retrieved March 5, 2009, from MicroMouse Hints, Ideas, Inspiration: http://www.micromouseinfo.com

[33] Smiths Medical PM, Inc. (2006, May). *Technical Description for OxiLink™ OEM Oximeter.* Retrieved May 18, 2009

[34] Analog Devices. (2006). *ADXL330 Data Sheet*.

[35] Kelly Pneumatics. (2008). *Miniature Proportional Valve.* Retrieved October 17, 2008, from Kelly Pneumatics: http://www.kpiwebsite.com/downloads.html

[36] Kelly Pneumatics. (2008). *Proportional Valve Amplifier Board.* Retrieved October 17, 2008, from Kelly Pneumatics: http://www.kpiwebsite.com/downloads.html

Other bibliography consulted

- Lewarski, J. S. (2006). *OXYGEN CONSERVING DEVICES: Optimizing the Benefit*. Retrieved October 27, 2008.
- Petty, T. L., McCoy, R.W. & Dennis E. Doherty. *Long Term Oxygen Therapy (LTOT)*. Retrieved October 28, 2008, from www.nlhep.org
- Chatburn, R. L., Lewarski J. S. & McCoy R. W. (2006). *Nocturnal Oxygenation Using a Pulsed-Dose Oxygen-Conserving Device Compared to Continuous Flow*. 51, (3):252-256. Retrieved October 29, 2008, from http://www.websciences.org
- National Heart Lung and Blood Institute. (March, 2009). *Chronic Obstructive Pulmonary Disease (COPD)*. Retrieved October 20, 2008, from http://www.nhlbi.nih.gov

- Pilling, J. & Michael C. (1999). *Ambulatory Oximetry Monitoring in Patients with Severe COPD: A Preliminary Study Ambulatory Oximetry Monitoring in Patients*. 116;314-321. Retrieved October 29, 2008, from http://chestjournal.org/cgi/content/abstract/116/2/314

- Barker, S. (September 10, 2002). *Principles of Pulse Oximetry Technology*. Retrieved November 5, 2008, from http://www.oximeter.org/pulseox/principles.htm

- Kamat, V. & Anaesth I. J. *Pulse Oximetry Indian*. 46. 4 (2002), 261-268, Retrieved April 18, 2009, from http://medind.nic.in/iad/t02/i4/iadt02i4p261.pdf

- Smiths Medical PM, Inc. (2005). *Functional and Fractional Oxygen Saturation - A Technical Overview*; Retrieved April 17, 2009, from http://www.smiths-medical.com

- Cardozo, J. S. (2008). *New AASM Recommendations for Sensors: A Simple Guide for the Sleep Technologist*. 3rd ed. Retrieved March 13, 2009, from http://www.sleepdt.com

- Microchip. (2007). *MPLAB C Compiler for PIC24 MCUs and dsPIC DSCs*. Retrieved November 20, 2008, from http://ww1.microchip.com.

- Ferreira, R. S. (Novembre 17, 2003). *Microcontrolador PIC*. Retrieved November 25, 2008, from.http://www.dpi.ufv.br/

- Sickle,T. V.(2001). *Programming Microcontrollers in C.* 2nd ed. USA: LLH Technology Publishing. Retrieved November 21, 2008.

- Nogueira, L. (April 18, 2009). *RS-232*. Retrieved November 23, 2008, from http://ldev.wordpress.com/2009/04/18/rs-232/

- Microchip, "Introduction Data Sheet." 2008.

- Microchip, "I/O Ports Data Sheet." 2008.

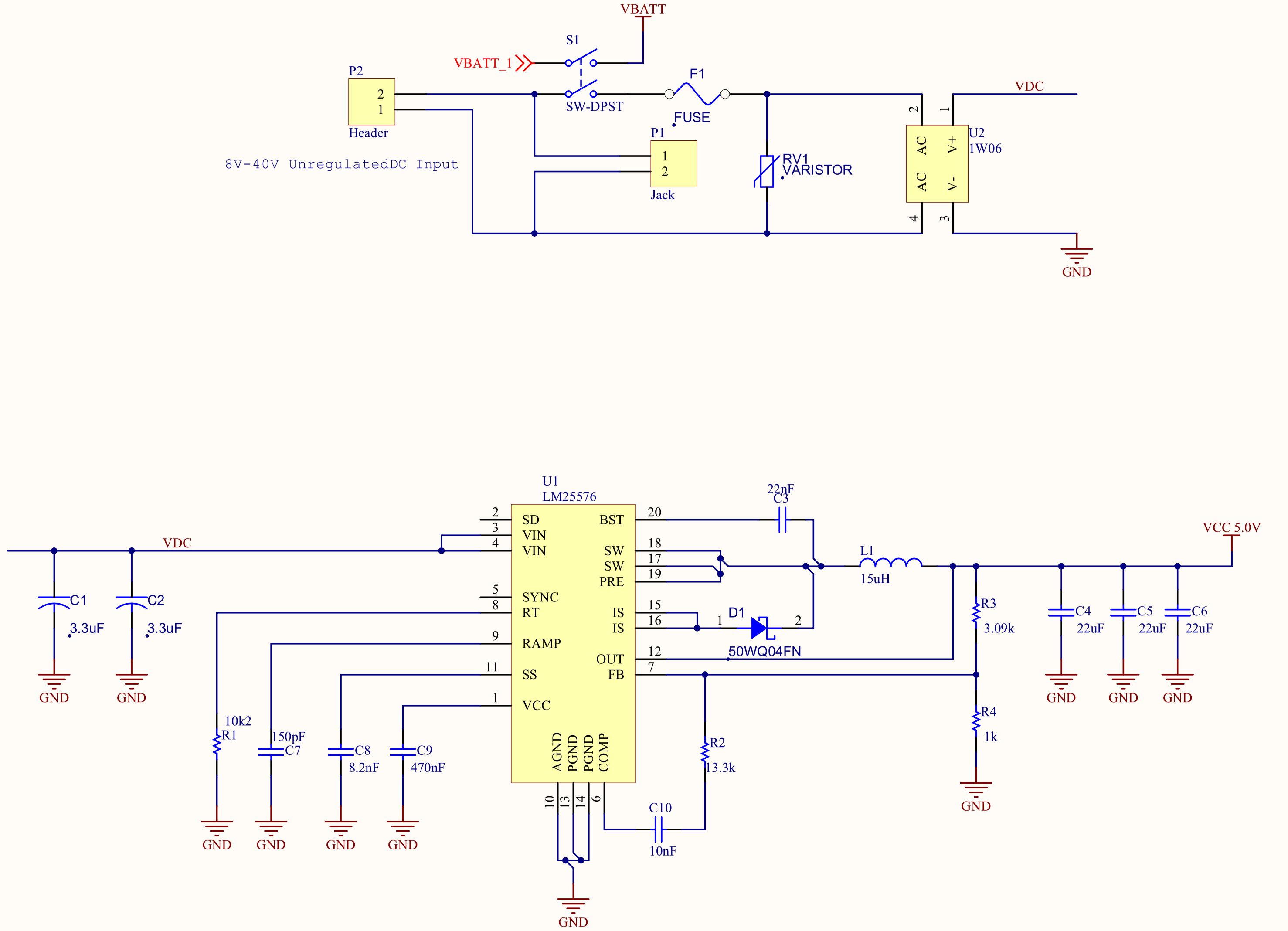
- Intersil, "ICL3238 Data Sheet." July, 2004.

- Microchip, "PIC24HJXXXGPX06/X08/X10 Data Sheet." 2008.

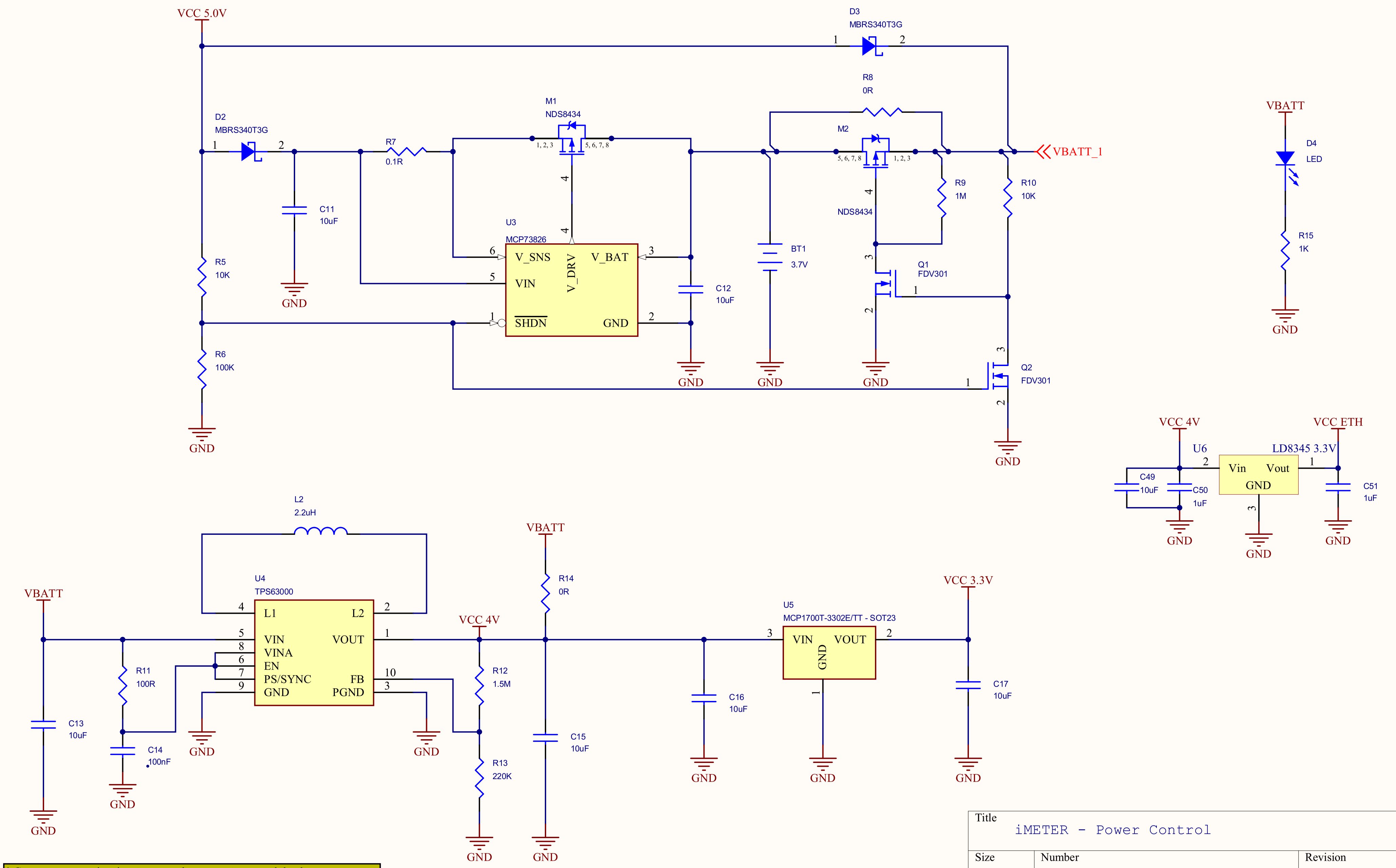
ATTACHMENTS

ATTACHMENT A – Schematic Documents

A 1 – *Leonardo* Platform Schematic

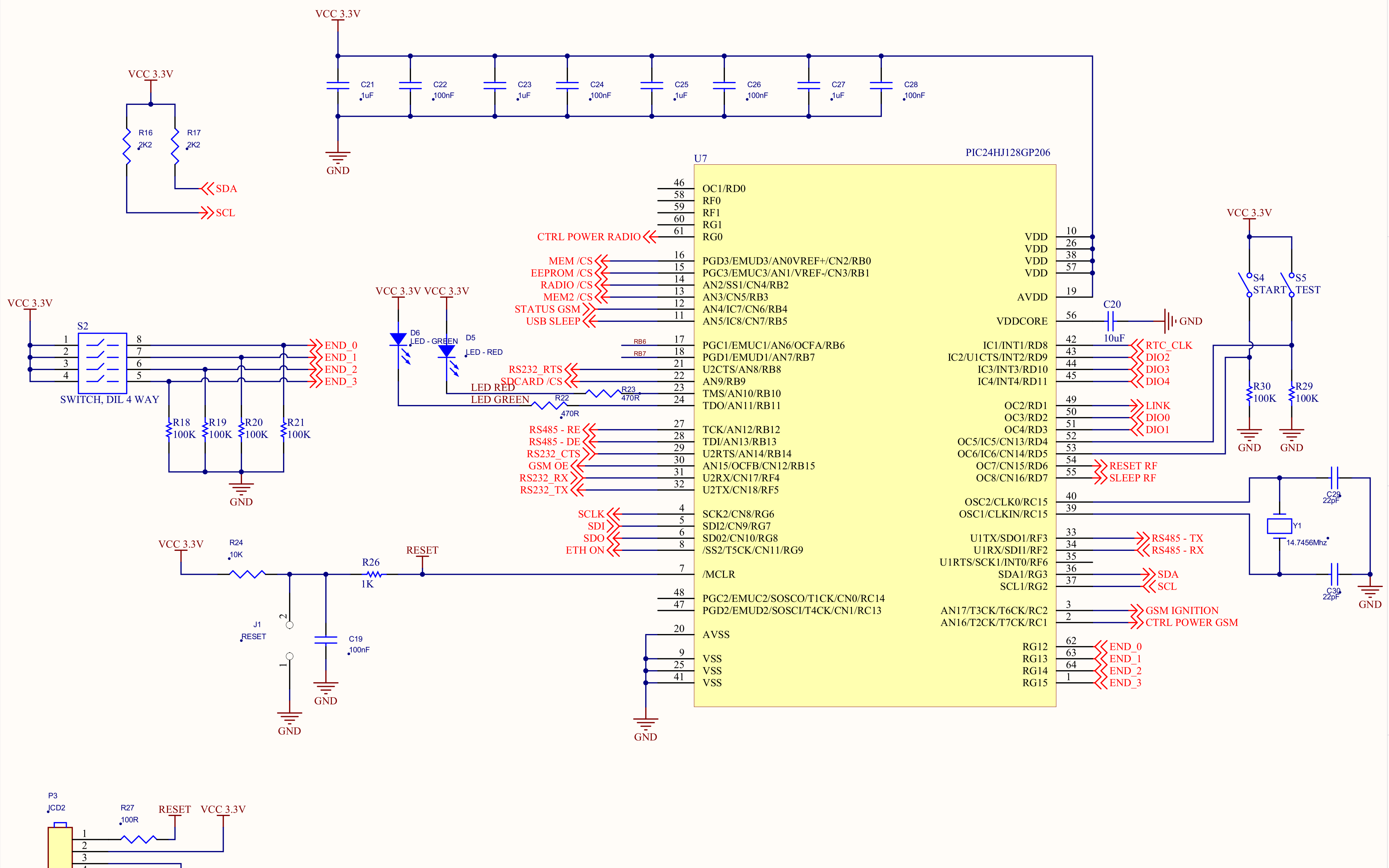


Title	iMETER - Power Supply		
Size	Number		Revision
A4			V2.0
Date:	30-04-2008	Sheet of	1 / 8
File:	D:\I S A\..\1 - POWER SUPPLY.SchDoc	Drawn By:	

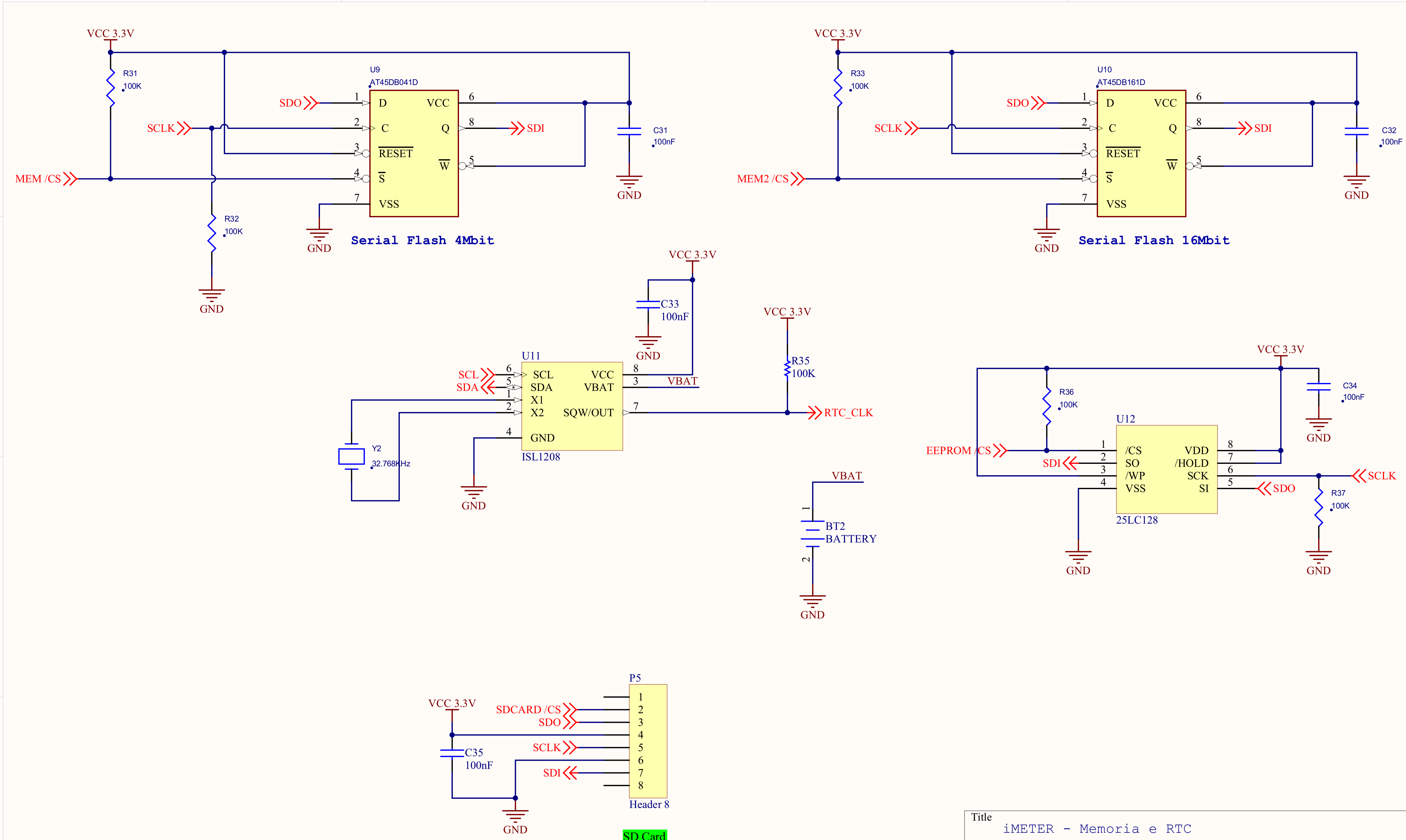


* Componentes opcionais, não montados na versão normal da placa

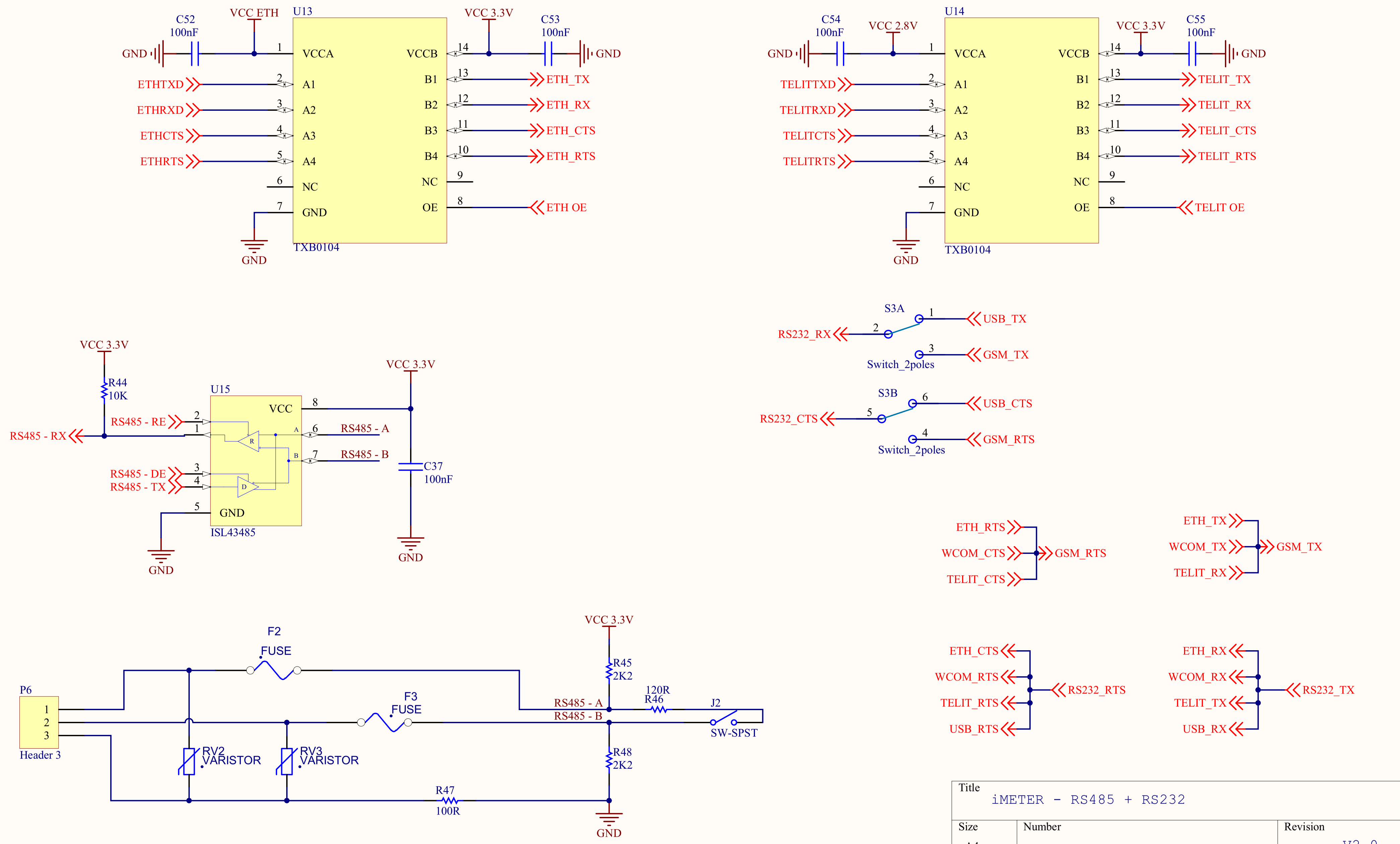
Title	iMETER - Power Control		
Size: A4	Number	Revision: V2.0	
Date: 30-04-2008		Sheet of: 2 / 8	
File: D:\I S A\..\2 - CTRL POWER.SchDoc		Drawn By:	



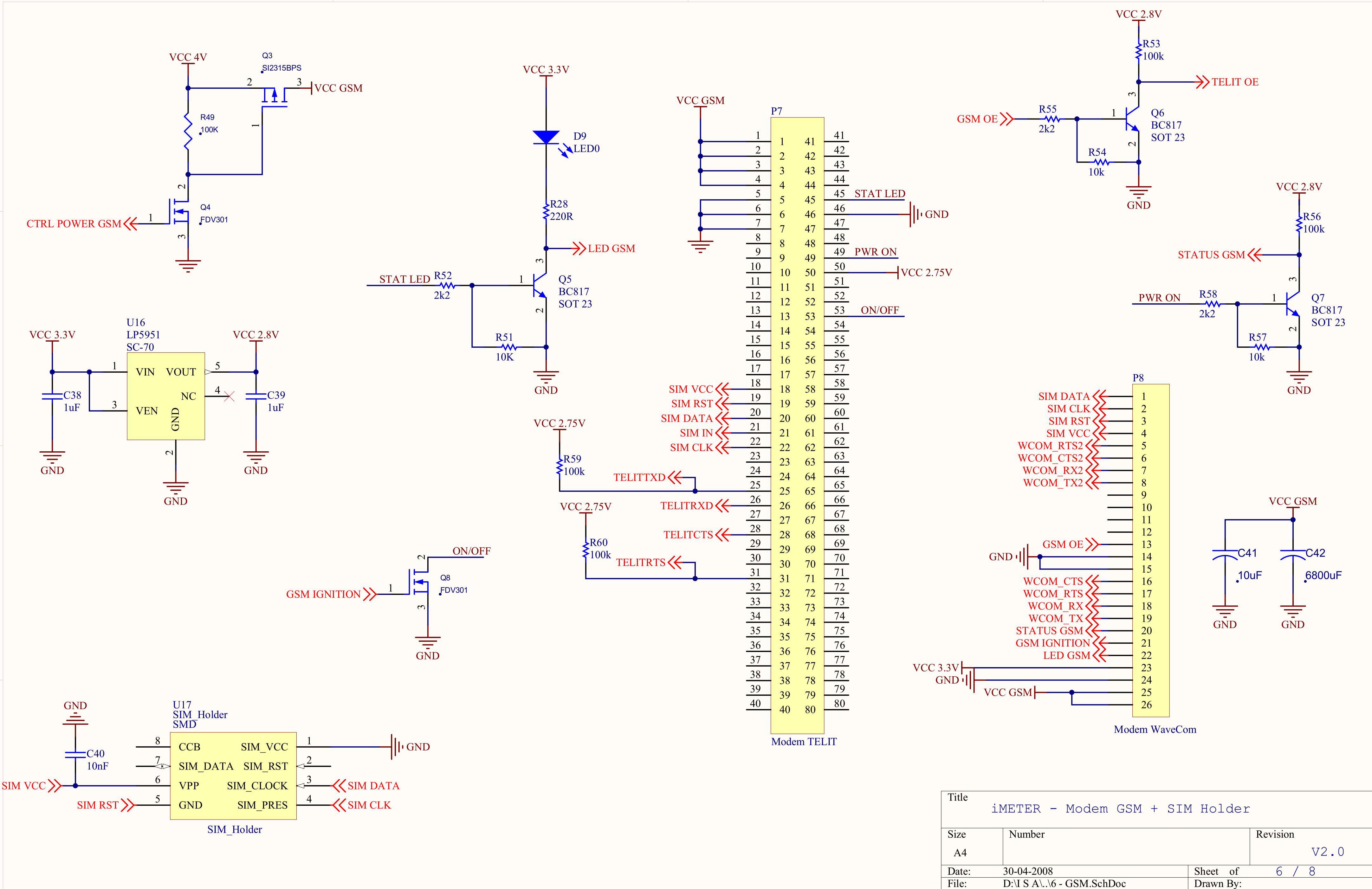
Title	iMETER - Microcontrolador	
Size: A4	Number	Revision: V2.0
Date: 30-04-2008		Sheet 3 of 8
File: D:\I S A\..\3 - MICROCONTROLER.SchDoc		Drawn By:

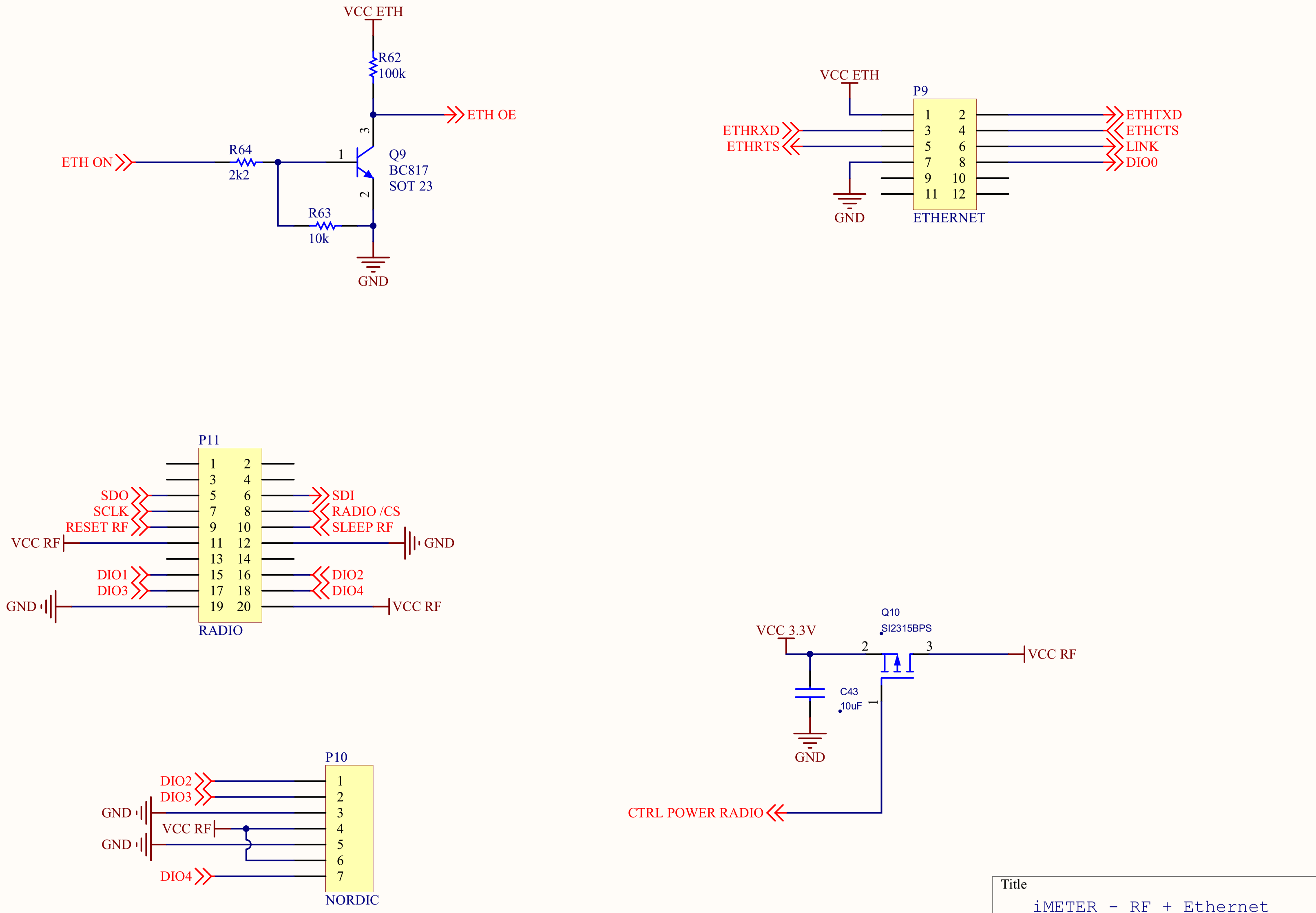


Title	iMETER - Memoria e RTC	
Size: A4	Number	Revision: V2.0
Date: 30-04-2008		Sheet 4 of 8
File: D:\I S A\..\4 - Mem + RTC.SchDoc		Drawn By:

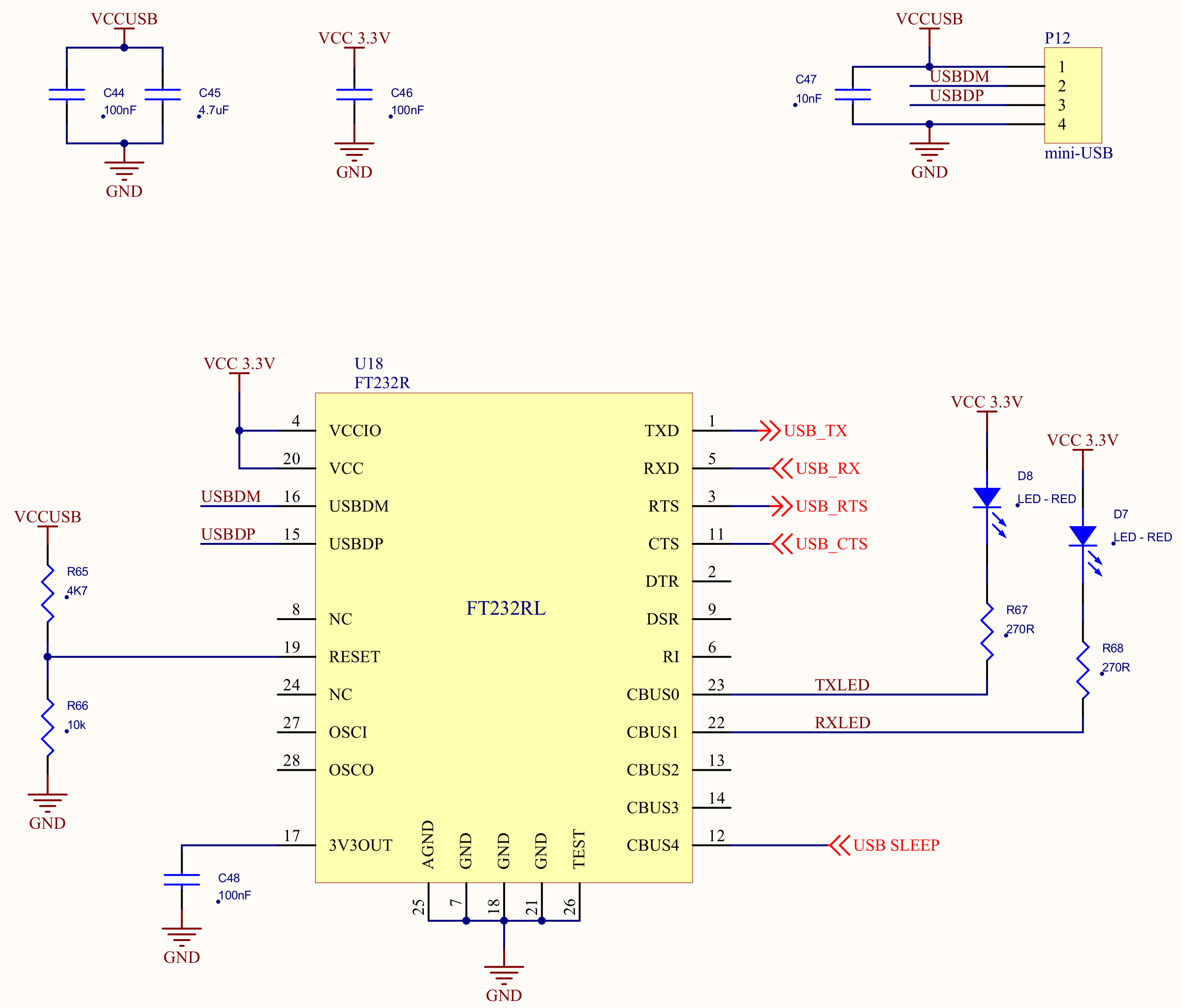


Title	iMETER - RS485 + RS232		
Size: A4	Number	Revision: V2.0	
Date: 30-04-2008		Sheet 4 of 8	
File: D:\I S A\..\5 - RS485 + RS232.SchDoc		Drawn By:	


VCC 4V
Q3
SI2315BPS
VCC GSM
R49
100K
Q4
FDV301
CTRL POWER GSM
GND
U16
LP5951
SC-70
VCC 3.3V
VCC 2.8V
VIN
VOUT
NC
VEN
GND
C38
1uF
C39
1uF
VCC 3.3V
D9
LED0
R28
220R
LED GSM
STAT LED
R52
2k2
Q5
BC817
SOT 23
R51
10K
VCC 2.75V
R59
100k
R60
100k
TELITTXD
TELITRXD
TELITCTS
TELITRTS
SIM VCC
SIM RST
SIM DATA
SIM IN
SIM CLK
P7
Modem TELIT
PWR ON
ON/OFF
Q8
FDV301
GSM IGNITION
U17
SIM_Holder
SMD
CCB
SIM_DATA
VPP
SIM_VCC
SIM_RST
SIM_CLOCK
SIM_PRES
C40
10nF
SIM_Holder
VCC 2.8V
R53
100k
TELIT OE
GSM OE
R55
2k2
Q6
BC817
SOT 23
R54
10k
R56
100k
STATUS GSM
R58
2k2
Q7
BC817
SOT 23
R57
10k
P8
Modem WaveCom
WCOM_RTS2
WCOM_CTS2
WCOM_RX2
WCOM_TX2
WCOM_CTS
WCOM_RTS
WCOM_RX
WCOM_TX
GSM IGNITION
C41
10uF
C42
6800uF
Title
iMETER - Modem GSM + SIM Holder
Size
A4
Number
Revision
V2.0
Date: 30-04-2008
Sheet of 6 / 8
File: D:\I S A\..\6 - GSM.SchDoc
Drawn By:



Title	iMETER - RF + Ethernet		
Size	Number		Revision
A4			V2.0
Date:	30-04-2008	Sheet of	7 / 8
File:	D:\I S A\..\7 - RF + Ethernet.SchDoc	Drawn By:	

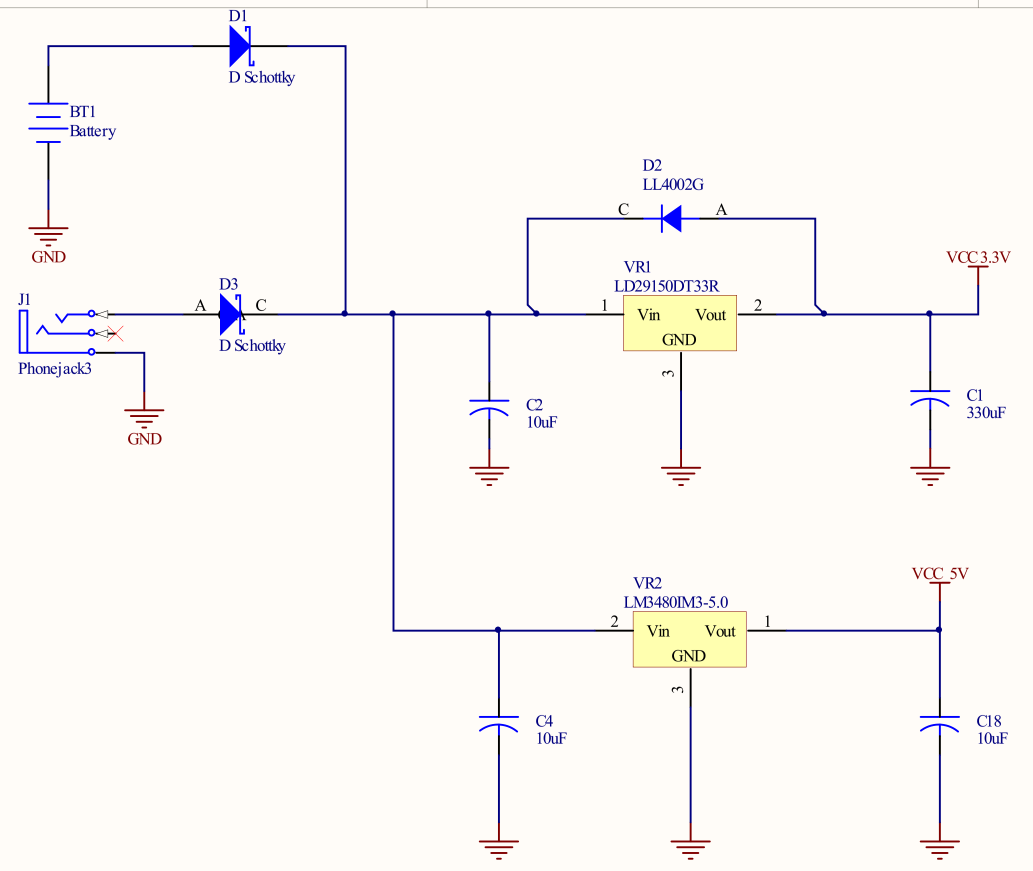


Title			
iMETER - USB			
Size	Number		Revision
A4			V2.0
Date:	30-04-2008	Sheet of	8 / 8
File:	D:\I S A\..\8 - USB.SchDoc	Drawn By:	

ATTACHMENTS

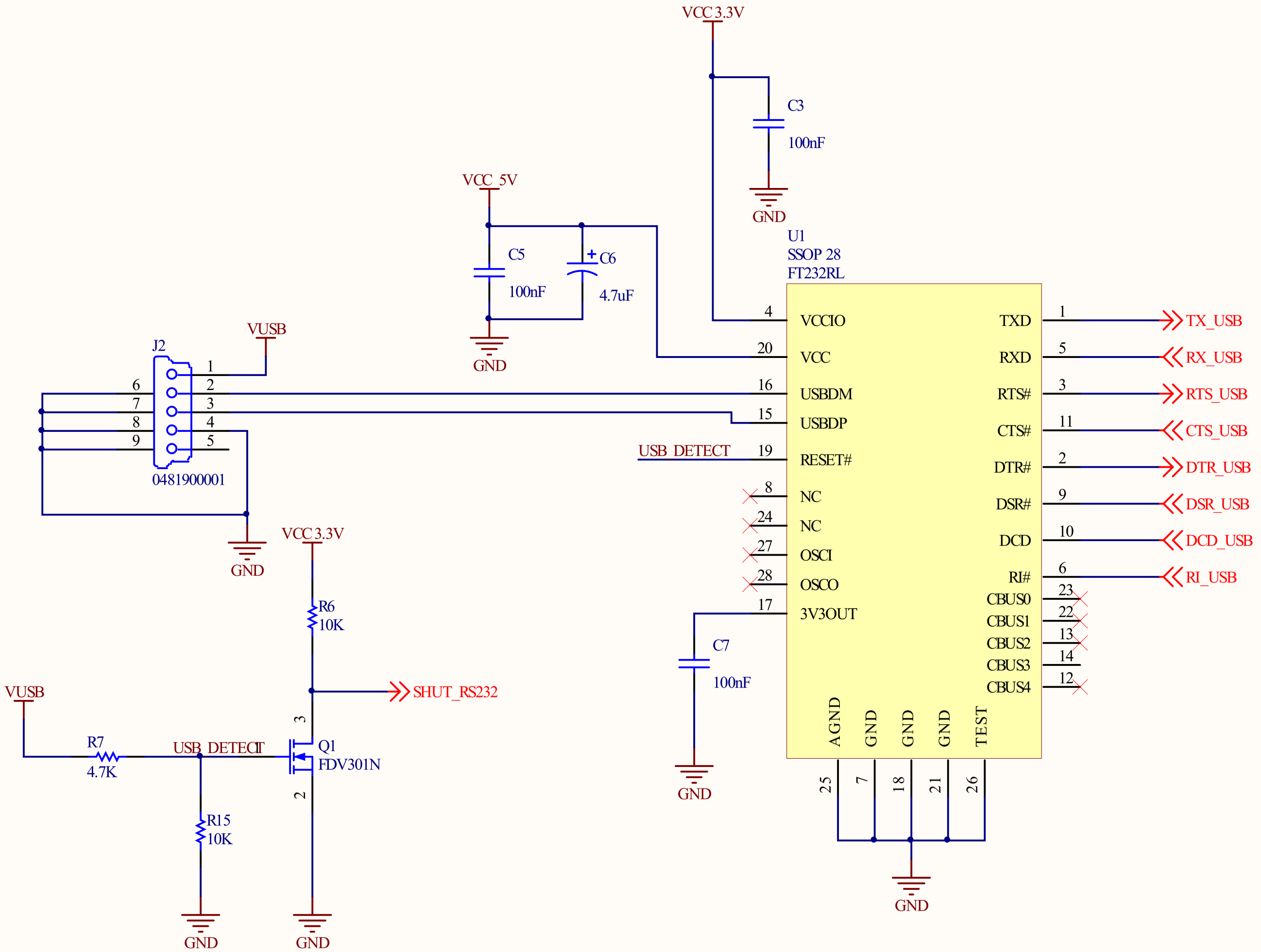
ATTACHMENT A – Schematic Documents

A 2 – iDemo Board Schematic



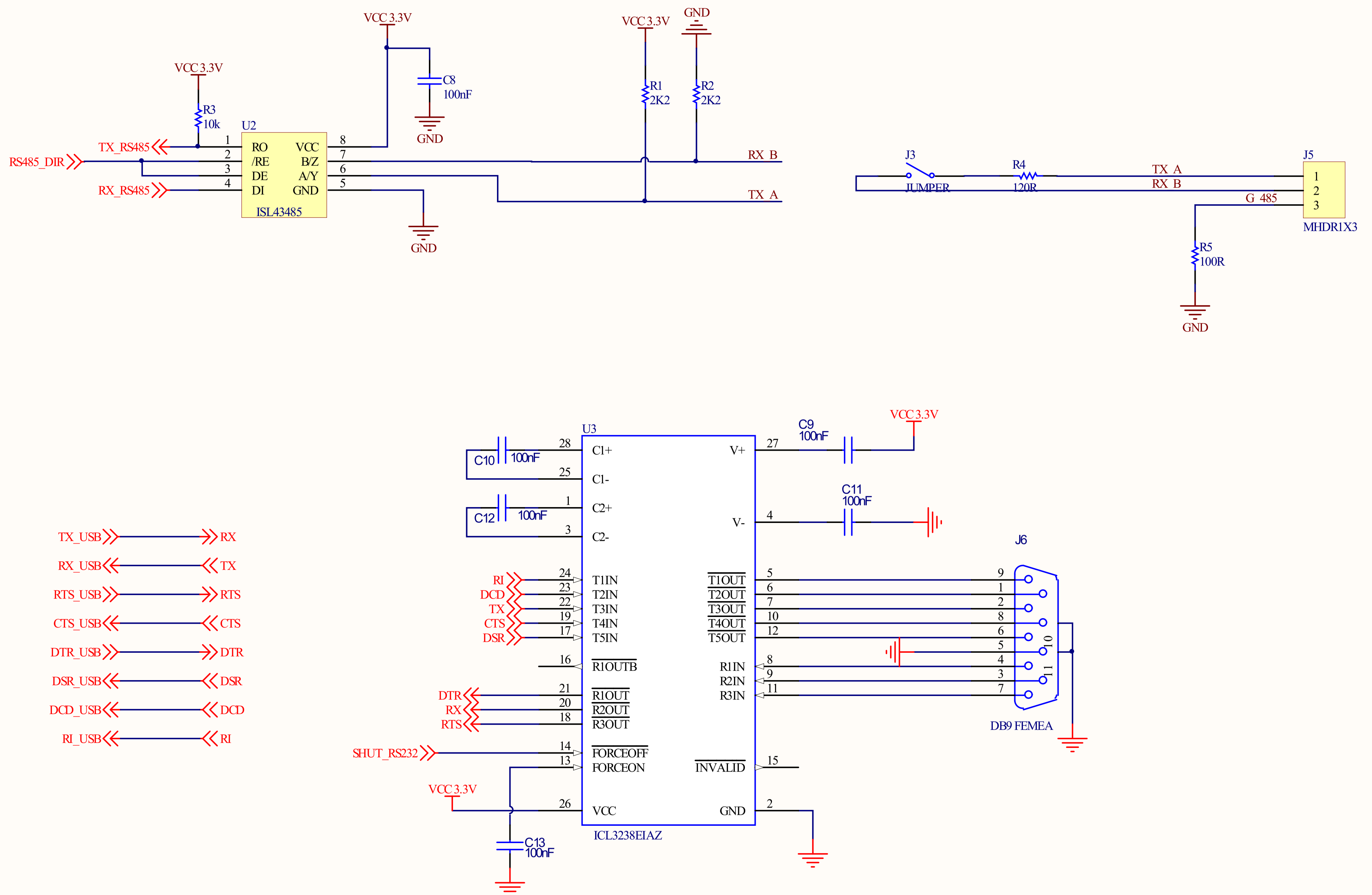
Projecto: iDEMOBOARD		
Módulo: Power Supply		
Número:	Desenhado por: Artur Vieira	
Data: 29/10/2008	Revisão: 0.1 Substitui:	Sheet: 1 of: 5

ISA
Intelligent Sensing Anywhere



Projecto: iDEMOBOARD	
Módulo: USB	
Número:	Desenhado por: Artur Vieira
Data: 29/10/2008 Revisão: 0.1 Substitui:	Sheet: 2 of: 5

ISA Intelligent Sensing Anywhere



Projecto: iDEMOBOARD

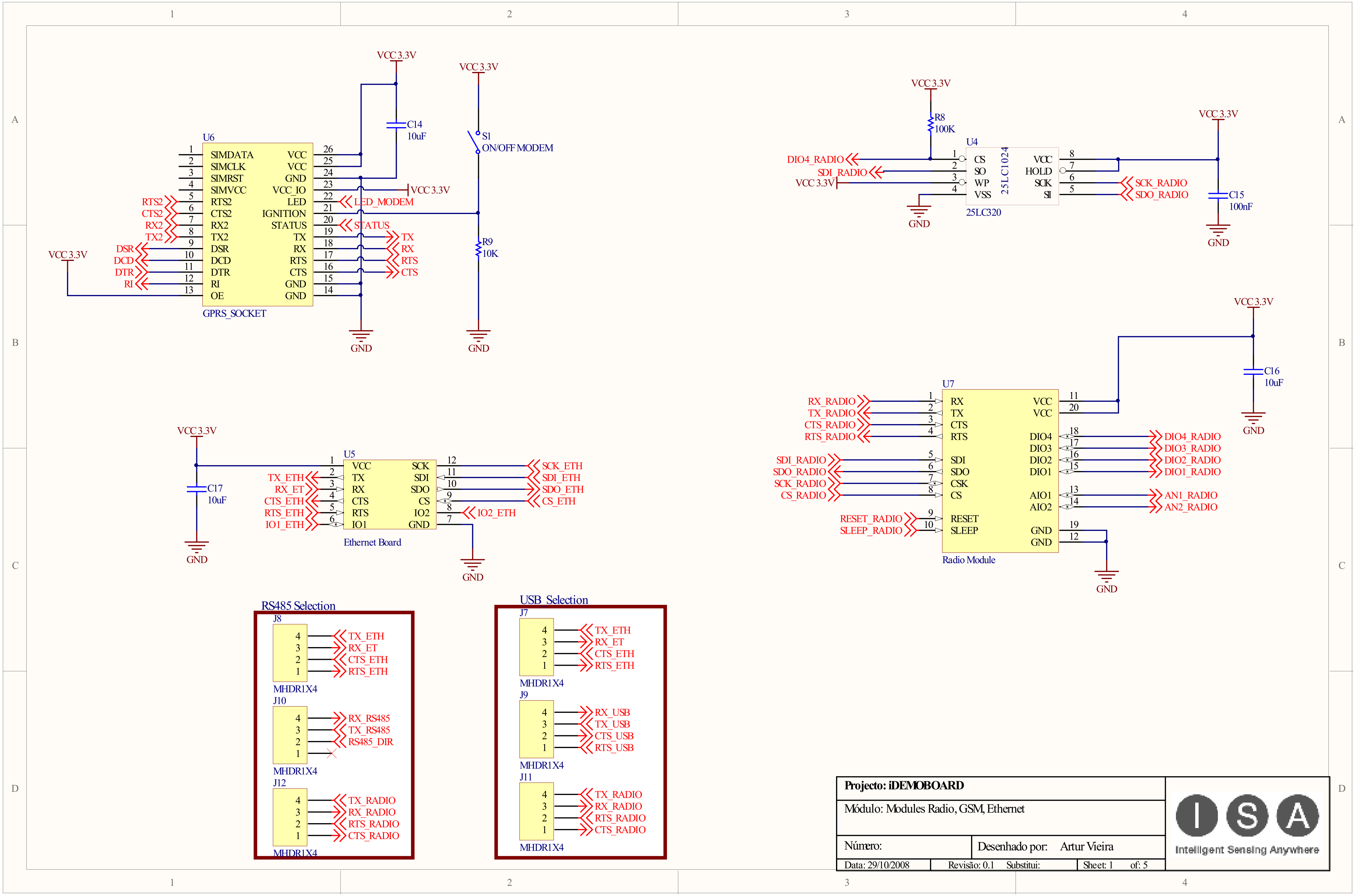
Módulo: RS232 / RS485

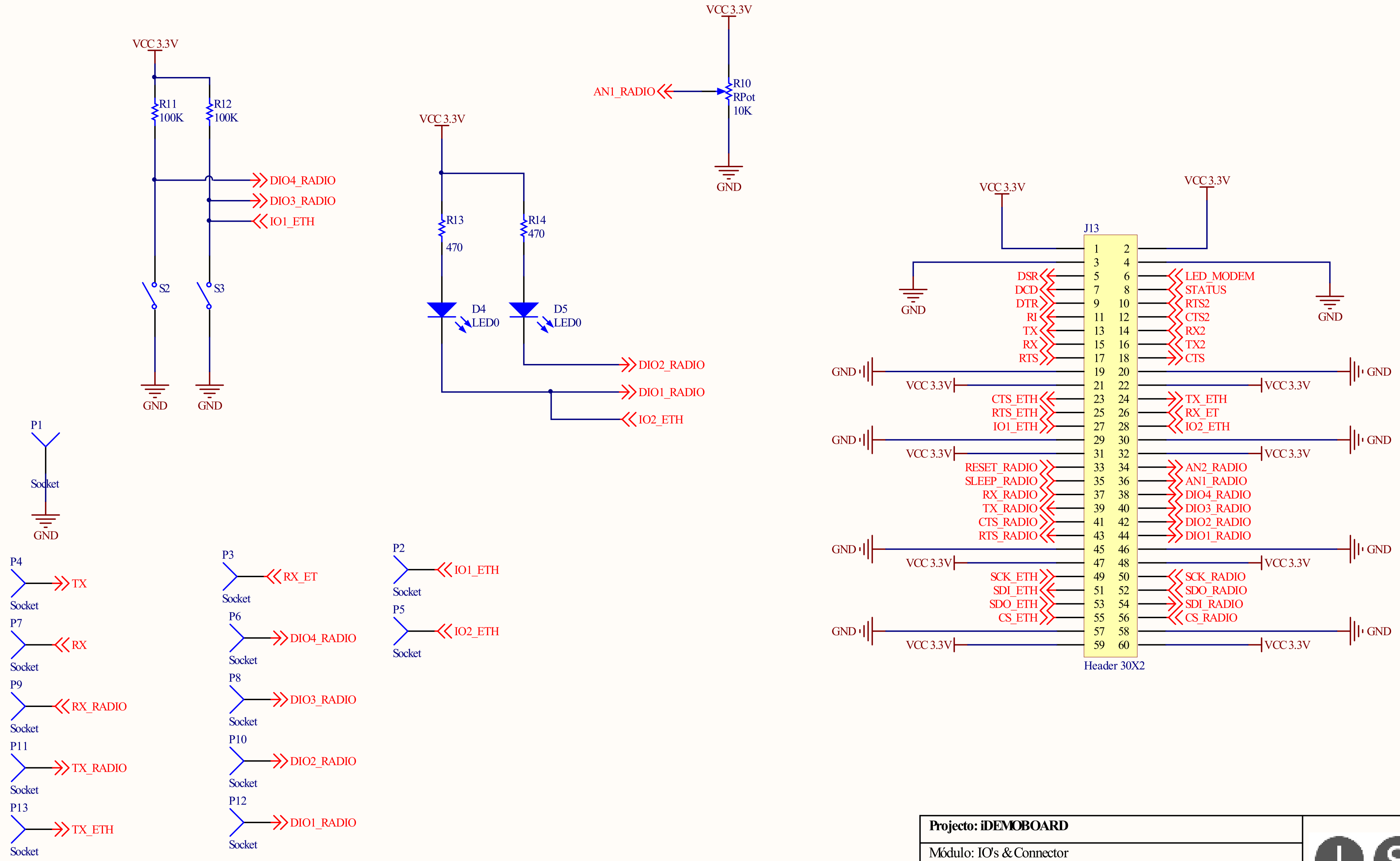
Número:

Desenhado por: Artur Vieira

Data: 29/10/2008 | Revisão: 0.1 | Substitui: | Sheet: 3 of: 5

Intelligent Sensing Anywhere


U6
GPRS_SOCKET
SIMDATA
SIMCLK
SIMRST
SIMVCC
RTS2
CTS2
RX2
TX2
DSR
DCD
DTR
RI
OE
VCC
VCC
GND
VCC_IO
LED
IGNITION
STATUS
TX
RX
RTS
CTS
GND
GND
VCC3.3V
C14
10uF
S1
ON/OFF MODEM
R9
10K
LED_MODEM
STATUS
GND
U5
Ethernet Board
VCC
TX
RX
CTS
RTS
IO1
SCK
SDI
SDO
CS
IO2
GND
TX_ETH
RX_ET
CTS_ETH
RTS_ETH
IO1_ETH
SCK_ETH
SDI_ETH
SDO_ETH
CS_ETH
IO2_ETH
C17
10uF
U4
25LC1024
25LC320
CS
SO
WP
VSS
VCC
HOLD
SCK
SI
R8
100K
DIO4_RADIO
SDI_RADIO
SCK_RADIO
SDO_RADIO
C15
100nF
U7
Radio Module
RX
TX
CTS
RTS
SDI
SDO
CSK
CS
RESET
SLEEP
DIO4
DIO3
DIO2
DIO1
AIO1
AIO2
RX_RADIO
TX_RADIO
CTS_RADIO
RTS_RADIO
CS_RADIO
RESET_RADIO
SLEEP_RADIO
DIO3_RADIO
DIO2_RADIO
DIO1_RADIO
AN1_RADIO
AN2_RADIO
C16
10uF
RS485 Selection
J8
J10
J12
MHDR1X4
RX_RS485
TX_RS485
RS485_DIR
USB Selection
J7
J9
J11
RX_USB
TX_USB
CTS_USB
RTS_USB
Projecto: iDEMOBOARD
Módulo: Modules Radio, GSM, Ethernet
Número:
Desenhado por: Artur Vieira
Data: 29/10/2008
Revisão: 0.1
Substitui:
Sheet: 1
of: 5
Intelligent Sensing Anywhere



Projecto: iDEMOBOARD			
Módulo: IO's & Connector			
Número:	Desenhado por: Artur Vieira		
Data: 29/10/2008	Revisão: 0.1	Substitui:	Sheet: 5 of: 5

ISA Intelligent Sensing Anywhere

ATTACHMENT B

Communication Protocol: OxiLink™ to the Host

The structure of the data packet is shown in the following table:

Byte	Bit(s)	Meaning
0	0	Pulse beep signal
	1 2	Slow data address, bit 0 Slow data address, bit 1
	3	Slow data address, bit 2
	4	Always 0
	5	Always 0
	6	1: if bytes 1, 2, 3 are used for the software revision number, 0: if bytes 1, 2, 3 are used for Oximetry data
	7	Always 1: sync bit, indicating the first byte in the packet (all other bytes is the packet have bit seven set to zero).
1	0…6	Real time plethysmogram waveform in the range 0 ...100
	7	Always 0
2	0…3 4	Real time Bar Graph, in the range of 0 …15. Zero value of the Bar Graph indicates no finger in sensor. Always 0
	5	Always 0
	6	Pulse Rate most significant bit (bit 7).
	7	Always 0
3	0…6	Value of these bits depends on the address field in byte 0, see table below
	7	Always 0

The following table indicates the value of bits 0…6 of byte 3 of the data packet depending on the

Address Field	Meaning of bits 0…6 of byte 3
0	SpO2 in the range of 0…99, 127 – invalid SpO2.
1	Bits 0…6 of Pulse Rate (note: bit 7 is located in byte 2). Pulse Rate range is 30…254 bpm, 255 – invalid Pulse Rate.
2	Signal Strength in the range 0…8
3	Alarm / Alert Condition: 0 – no alarms / alerts; 1 – sensor is unplugged; 2 – no finger in sensor or sensor problem; 3 – searching for pulse; 4 – searching too long; 5 – lost pulse alarm.
4	Instantaneous SpO2; it is sent with each pulse detected; it equals to zero in between of the pulses. Host system can use instantaneous SpO2 to build its application-specific averaging algorithm.
5…7	Used for production tests only

Microcontroller's Configuration

- ### Oscillator Registers

Register FOSCEL

U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—
bit 15							bit 8

R/P	U	U	U	U	R/P	R/P	R/P
IESO	—	—	—	—	FNOSC<2:0>		
bit 7							bit 0

bit 2-0 **FNOSC<2:0>**: Initial Oscillator Source Selection bits
111 = Fast RC Oscillator with Divide by N (FRCDIVN)
110 = Fast RC Oscillator with Divide by 16 (FRCDIV16)
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (Sosc)
011 = Primary Oscillator with PLL (XTPLL, HSPLL, ECPLL)
010 = Primary Oscillator (XT, HS, EC)
001 = Fast RC Oscillator with PLL (FRCPLL)
000 = Fast RC Oscillator (FRC)

For example:

#define FOSCSELREG_INIT 0xFF02 // define primary oscillator (1111111100000010)
#define FOSCSEL_CONFIG _FOSCSEL(FOSCSELREG_INIT)

Register FOSC

U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—
bit 15							bit 8

R/P	R/P	R/P	U	U	R/P	R/P	R/P
FCKSM<1:0>		IOL1WAY[(1)]	—	—	OSCIOFNC	POSCMD<1:0>	
bit 7							bit 0

bit 1-0 **POSCMD<1:0>**: Primary Oscillator Mode Selection bits
11 = Primary Oscillator disabled
10 = HS Crystal Oscillator mode
01 = XT Crystal Oscillator mode
00 = EC (External Clock) mode

For example:

#define FOSCREG_INIT 0xFF46 //define the oscillator mode (1111111101000110) – HS
#define FOSC_CONFIG _FOSC(FOSCREG_INIT)

If the selected oscillator is a primary oscillator (as is the case of this PIC) or FRC oscillator, it is occasionally necessary activated the PLL system. For that, the parameters are configured by the CLKDIV, PLLFBD e OSCON registers.

Register CLKDIV

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
ROI	DOZE<2:0>			DOZEN[(1)]	FRCDIV<2:0>		
bit 15							bit 8

R/W-0	R/W-1	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PLLPOST<1:0>		—	PLLPRE<4:0>				
bit 7							bit 0

PLLPOST<1:0>: PLL VCO Output Divider Select bits (also denoted as 'N2', PLL postscaler)
00 = Output divided by 2
01 = Output divided by 4 (default)
10 = Reserved
11 = Output divided by 8

PLLPRE<4:0>: PLL Phase Detector Input Divider Select bits (also denoted as 'N1', PLL prescaler)
11111 = Input divided by 33
.
.
.
00001 = Input divided by 3
00000 = Input divided by 2 (default)

For example:

```
#define PLLPOST_VALUE 0                  // PLLPOST=2
CLKDIVbits.PLLPOST=PLLPOST_VALUE;        // N2=input/2

#define PLLPRE_VALUE 2                   // PLLPRE=4
CLKDIVbits.PLLPRE=PLLPRE_VALUE;          // N1=input/4
```

Register PLLFBD

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	PLLDIV<8>
bit 15							bit 8

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
PLLDIV<7:0>							
bit 7							bit 0

PLLDIV<8:0>: PLL Feedback Divisor bits (also denoted as 'M', PLL multiplier)
111111111 = 513
.
.
.
000110000 = 50 (default)
.
.
.
000000010 = 4
000000001 = 3
000000000 = 2

For example:

```
#define PLLDIV_VALUE 41                  //PLLDIV=43
PLLFBDbits.PLLDIV=PLLDIV_VALUE;          // M=input*43
```

Register OSCON

U-0	R-y	R-y	R-y	U-0	R/W-y	R/W-y	R/W-y
—	COSC<2:0>			—	NOSC<2:0>		
bit 15							bit 8

R/S-0	R/W-0	R-0	U-0	R/C-0	U-0	R/W-0	R/W-0
CLKLOCK	IOLOCK(1)	LOCK	—	CF	—	LPOSCEN	OSWEN
bit 7							bit 0

NOSC<2:0>: New Oscillator Selection bits
111 = Fast RC Oscillator with Divide by N (FRCDIVN)
110 = Fast RC Oscillator with Divide by 16 (FRCDIV16)
101 = Low-Power RC Oscillator (LPRC)
100 = Secondary Oscillator (Sosc)
011 = Primary Oscillator with PLL (XTPLL, HSPLL, ECPLL)
010 = Primary Oscillator (XT, HS, EC)
001 = Fast RC Oscillator with PLL (FRCPLL)
000 = Fast RC Oscillator (FRC)

For example:

```
#define OSCCONH_NEWVALUE 0x23           // binary-00100011 (bit 15-bit8)
#define OSCCONL_NEWVALUE 0x01           //binary- 00000001 (bit 8-0)
_builtin_write_OSCCONH(OSCCONH_NEWVALUE);
_builtin_write_OSCCONL(OSCCONL_NEWVALUE); //it allows to open the
```

register, to write and to block the register - no loss of information

FOSC Calculation

$$F_{OSC} = F_{IN} \times \left(\frac{M}{N1 \times N2}\right) = F_{IN} \times \left(\frac{(PLLDIV + 2)}{(PLLPRE + 2) \times 2(PLLPOST + 1)}\right)$$

Where,

$N1$ = PLLPRE + 2

$N2$ = 2 x (PLLPOST + 1)

M = PLLDIV + 2

- ## I/O ports Registers

↳ The **TRISx register** allows associate that pin I / O acts as an exit or entry:

0 ⇨ the pin acts as exit

1 ⇨ the pin acts as entry

For example:

```
//it defines the pin RB12 as entry and the pin RB13 as exit
TRISBbits.TRISB12 = 1;
TRISBbits.TRISB13 = 0;
```

↳ The **PORTx register** is used to read the state of the pin I / O:

0 ⇨ means that the pin is 0V

1 ⇨ means that the pin is 3.3V

For example:

```
// it reads the state of the RD5 pin and verify if it is 3.3V
PORTDbits.RD5==1
```

Note: the Vcc of PIC24 is 3.3V

↳ The **LATx register** allows to write the state of the pin:

0 ⇨ put the pin to 0V

1 ⇨ put the pin to 3.3V

For example:

```
//Pin B12= 0V
LATBbits.LATB12 = 0;
```

- ## UART Registers

The configuration of the UART module is based on five control registers, but only those that were used are presented bellow.

Register UxMODE – UARTx Mode Register

UARTx enable bit
1 = UARTx is enable
0 = UARTx is disabled

High Baud Rate select bit
1 = High speed
0 = Low speed

R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0[2]	R/W-0[2]
UARTEN	—	USIDL	IREN[1]	RTSMD	—	UEN<1:0>	
bit 15							bit 8

R/W-0 HC	R/W-0	R/W-0 HC	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	LPBACK	ABAUD	URXINV	BRGH	PDSEL<1:0>		STSEL
bit 7							bit 0

Auto Baud enable bit
1 = Enable baud rate
0 = Disabled baud rate

Parity and data selection bits
00 = 8-bit data, no parity
01 = 8-bit data, even parity
10 = 8-bit data, odd parity
11 = 9-bit data, no parity

Stop selection bit
1 = 2 Stop bits
0 = 1 Stop bit

For example:

```
//UARTx is enable
U1MODEbits.UARTEN=1

//High baud rate
U1MODEbits.BRGH=0

//Disabled baud rate
U1MODEbits.ABAUD=0

//8-bit data, no parity
U1MODEbits.PDSEL=0
```

Register UxSTA – UARTx Status and Control Register

R/W-0	R/W-0	R/W-0	U-0	R/W-0 HC	R/W-0	R-0	R-1
UTXISEL1	UTXINV[1]	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL<1:0>		ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

Receive Interrupt mode selection bit
11 = Interruption when receipt buffer is full (4 characters)
10 = Interruption when receipt buffer is ¾ full (3 characters)
0x = Interruption when a character is received

For example:

```
//Interrupted after 1 Rx
U1STAbits.URXISEL=0
```

Register UxBRG – UARTx Baud Rate register

- It saves the baud rate value of data received/transmitted.

For example:

```
// Select a baud rate of 4800 bits/s for UART1
U1BRG=4800;
```

- ## Interrupts Registers

Register IEC0 – Interrupt Enable Control register 0

- Control the activation of interruptions

UART1 Receiver Interrupt enable bit
1 = Interrupt request enable
0 = Interrupt request disable

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	DMA1IE	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPI1EIE	T3IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IE	OC2IE	IC2IE	DMA0IE	T1IE	OC1IE	IC1IE	INT0IE
bit 7							bit 0

For example:

```
// Interrupt enable
IEC0bits.U1RXIE=1;
```

Register IFS0 – Interrupt Flag Status register 0

- Control of the interruption flags state

UART1 Receiver Interrupt Flag Status bit
1 = Interrupt request has occurred
0 = Interrupt request has not occurred

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	DMA1IF	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPI1EIF	T3IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IF	OC2IF	IC2IF	DMA01IF	T1IF	OC1IF	IC1IF	INT0IF
bit 7							bit 0

For example:

```
// Interrupt has not occurred
IFS0bits.U1RXIF=0;
```