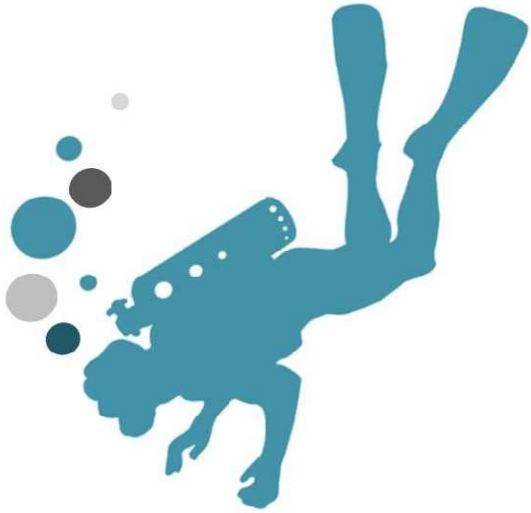
INGENIØRHØJSKOLEN I KØBENHAVN

DELPHINUS

Development of a Digital Communication Device for Diving



Document: Final report

Date: 2 December 2010

Version: 1.0

Members of the team

Pawel Gorski *(PL) 101024*

Mechatronics

David Pinilla Ramiro *(ES) 101001*

Telecommunications

Marcin Samsonowski *(PL) 101028*

Computer science

Luna Vaquero Aso *(ES) 100970*

Industrial Design

Tamanna Zirak *(NL) 100891*

Human Technology

Supervisor: Emil Piper

EPS 2010, Autumn

Group 7

I Abstract

The main of the project was to improve the digital communication device for divers from the previous project group. To achieve the aim, there are several improvement made in comparison to the previous project group.

Therefore is the concept, Delphinus developed. Delphinus size is smaller. The design and interface are more user friendly.

Delphinus can be tied up with two straps on the arm. The interaction with the user takes place using visualization on the OLED display. When sending and receiving messages, it can be controlled by using the touch screen.

In this report there is more detailed information about divers aspects of Delphinus and how the process was when developing Delphinus in a multidisciplinary team.

II Table of contents

III List of figures

IV List of tables

V Preface

This project has been running more than three times now. The project started when the diving club Seahorse came up with the idea to develop a new wireless digital communication device for divers. In comparison to other devices that exist, it had to be smarter, more user friendly and cheaper.

During the project several groups focused on different parts, like design, programming or material. The previous project group made a CAD model. They focused more on technical and business aspect. The focus of the group this time was redesigning a user friendly casing & interface, electronics and also business, but studying divers as users of Delphinus rather than customers. The team was divided in two subgroups, Design & Business and Electronic & Programming.

For the project group this was the first time to work in a multidisciplinary team. In EPS it was fun to work with the people from different disciplines and cultures. It was difficult, but we learned a lot about what is really necessary when working in a multidisciplinary team and how to contribute in the future. The contribution this time was the following below.

Contribution by the members of the projectgroup

Pawel used his electronic and programming skills on how to use ATMEGA, PWM, USART and crystals. He is very passionate with his work. It has been very nice to talk and discuss with Pawel about different subjects.

David as telecommunication engineer, contributed in the electronic part. He attended the needs of the microcontroller and display. He researched the appropriate one, the programming or constructing the board (PCB) where they are connected to. David is a real shaper; he kept us motivating and made us laugh even when we had hard times. He is very open.

At the beginning, Marcin, with his experience in group work was chosen by group for team leader. He was responsible for the direction of system development. His overall view of the whole project was helpful in making decisions. He was participating in electronic part of project, despite of different field of study. He was trying to find a link between electronics and marketing and design. He made research and worked out an extended software design part.

Luna as industrial designer contributed by making the casing and interface user friendly. She can work very well independent and is also a good team player. She works hard and is humble.

Tamanna gave the users insight to the project as Human Technology. She was the secretary in the project, kept everyone informed. She worked on interface, requirements specification and did some marketing research. She took responsibility during the project.

Acknowledgement

We would like to thank our supervisor, Emil Piper for guiding us during this project. He gave us feedback, brought us to ideas and made us think if we were going towards our goal.

We would like to thank our home universities for this opportunity. And at last but not least IHK and everyone who is not mentioned but was involved in EPS.

Signature

Pawel Gorski	David Pinilla Ramiro	Marcin Samsonowkski
Luna Vaquero Aso	Tamanna Zirak	

Information

University	:	Copenhagen College of Engineering, Lautrupvang 15, 2750 Ballerup.
Course	:	European Project Semester 2010 Autumn.
Project proposal	:	Digital communication device for divers.
Project group	:	7
Hand in date	:	2 December – 12 pm
Supervisor	:	Emil Piper (Epi@novonordisk.com/epi@ihk.dk)

1 Introduction

The people always needed to communicate each other. Nature gave us tools to communicate in short distances in the air. But also nature gave us propel to explore other environments. So we use our senses to communicate. In our environment we can see, smell, taste, hear, feel but also need our body language while talking to understand each other properly.

But when the environment changes then we are limited to use our senses and skills, we need to communicate different in order to understand each other. In an environment like in the sea, we only could use body language. For example divers stay close to each other and use common hand signs to communicate. But how do animals communicate under water on long distance?
Fishes use (colour) signs, gestures, vibration and sound to communicate. Another animal that also uses sound are the dolphins. So we need to invent items which can give us additional skills which normally were reserved for animals to communicate under water.

For many years it was difficult to use technology for communication between people under water and also very expensive. The particular features of water like easily conduction of electro-magnetic waves don't allow using radio waves which are in common use in the air. Finally electronics and physics solve these problems. Now we know that the best way to sending and receiving messages under water is by acoustic waves like dolphins do. We cannot speak under water because transmission sound waves between air and water are very difficult.

We need a device which can generate sound underwater very similar to common speakers. Using changing voltage we are making vibrations and these vibrations are transmitted by water molecules. Having a vibrations sensitive microphones we can receive a message. These waves have to be translated to microcomputer and then showed to the human. It has to be shown in a good design and with user friendly interface.

All this things conduct to make a communication device for divers which we call “Delphinus”. Delphinus consists an AMOLED display, ATMEGA16L microcontroller, battery, a new casing and an interface.

2 Method

There has been several method used to accomplish the tasks in the project. In this chapter is the method explained the participants, the tools that are needed to achieve the aim and an overview of the project process. The data processing of the results is made in the different chapters of the report and some are in *appendix.*

The task of the project was making a working prototype (a wireless digital communication device for divers). Therefore old reports from previous semester projects were utilized to take information out and improve the application benefits.

In this project there are several method used to achieve the aim. The methods are used in different state of the process.

Desk research

The project work was divided in different parts, programming, electronics, design and marketing. The project team was divided in two sub-groups depending on which skills the members had and would want to learn. Each member did desk research for the different parts and studied, analyzed the data and edited it. *The project management* document in appendix shows more detail how the teamwork was divided.

The project started gathering and analyzing information and data from different ways. This documentation was extract from previous reports (old EPS projects), internet, diving videos, books, articles and other information provided by our supervisor.

Morphological chart

The next step consisted on generate and evaluate new ideas using some of the tools and techniques to promote innovation and design: because the aim was to design cheaper device which now exist on the market. *The morphological chart* is in the appendix.

Functional Modeling

The functional modeling is used to analyze alternatives ways of device's construction (its hardware) and how it acts. It also is used for making product requirements, when testing the interaction; the functional model was edited. This functional modeling is made in the document requirement specification (appendix) and is called differentially, because it is used for a system, conceptual model.

Generating and Evaluating Alternatives (Brainstorming)

During brainstorming there are ideas generated and afterwards evaluated. Brainstorming was used for further development of the mock-up model than previous project group made. The advantage of this method was: all members of the team are involved to generate ideas. One of the outcome were for example the first model, appendix chapter 4.2. During brainstorming was another the interface, chapter 5.

Participants

The project members were focused in different parts. Below can be seen how the project group is divided in two subgroups. Subgroup one is in charge of the electronic and programming section. Subgroup two takes charge of developing the casing and doing the marketing research.

Subgroup 1_ Electronic and Programming section		
Name	Specialist area	Skills
Pawel Gorski	Mechatronics	Microcontrollers and electronics.
David Pinilla Ramiro	Telecommunications	Programming, electronics and signal processing.
Marcin Samsonowski	Computer Science	Software design and development, programming, algorithms, image processing, graphical user interface (GUI), project management,
Luna Vaquero Aso	Industrial Design	Sketching, 3D modeling, ergonomics aspects, materials and manufacturing process, marketing.
Tamanna Zirak	Human Technology	Make a link between human and technology: translate the needs of users into a product. Interaction (user testing), make (specify) requirements for a product, usability, ergonomics aspects and some marketing.

Tools

The following tools were needed:

- Laptops
- Programming software
- Electronic stuff (lab)
- Microcontroller
- Microprocessor
- Display
- 3D modeling software: Autodesk Inventor Professional, 3DS Max Studio
- 3D Printer
- Project room
- Post-its

An overview of the process was:

- Gathering information and data
- Analyzing data
- Choosing a useful part of work that is done
- Choosing a appropriate parts and devices for our project
- Doing marketing research
- Making requirements
- Generate and evaluate new idea's
- Designing a structure of a system
- Programming and assembling
- Testing the prototype in different kind of aspects
- Preparing the documentation of the project
- Preparing the final presentation

3 Marketing

The previous group made also marketing research. *Results of marketing research* can be found in the appendix.

While doing the marketing research first it was the plan to introduce the product to military and all different kind of divers. So the target group would remain big.

But when doing more research about the divers as users rather than customers, it was obvious the communication between them differs. For example cave divers communicate different then military. The research is processed in the chapter *user profile.*

So conclusion about the marketing changes now, it can be said:

- The market is limited: the target group is small and the device is expensive.
- There had to be a segment chosen: the interaction of divers with different purposes is different. Delphinus has to be made for the biggest target group and if it's useful for other divers, they also could use it. More about segment can be found in the chapter *user profile*.
- Use strategy to make improvement to design and interface. So it is more user friendly.
- Introduce Delphinus at working prototype status to potential customers (for example a business-to-business or military) who are interested to invest.
    - Sell via online shopping because divers are living worldwide.

4 User profile

This chapter gives information about the segmentation of Delphinus, the characteristic of the users and the interaction.

Diving and about divers

Scuba diving (Self Contained Underwater Breathing apparatus) is a form of underwater diving in which a divers uses a scuba set, see Figure 4 .1 to breath underwater for recreation, commercial or industrial reasons. Scuba divers have their own breathing gas they can stay longer underwater than others who are free diving.

Figure 4.1, Diver equipment (2010, http://visual.merriam-webster.com)

The maximum depth of recreational diving is between 30 m and 40 m. If a diver gets deeper it will be unsafe because they need specialized training and equipment for technical diving.

When deep diving there are possibilities to get the caisson disease. The cause of the disturbances is that during the stay in the atmosphere with high pressure there is more nitrogen gas dissolve in the blood and in the tissue than in normal circumstances. When going to circumstances where the pressure is normal, the nitrogen gas won't remove fast enough through the longs. The nitrogen gas raises in to bubbles in the blood vessel, so it blocks the blood circulation. If there the bubbles raise in brain, it can end deadly.

Nitrogen gas does not totally dissolve in water, but when the pressure its high the dissolution increases. When going down en diving deep nitrogen dissolve in blood and from there to the other part tissues of the body. When the dive is deeper, nitrogen dissolves faster.

So diving should be not for a long time and not to deep (less dissolution of nitrogen), and slowly rising up (the body gets enough time to remove the gas through the longs)

Divers could get divers symptoms like from dizziness to dying. There is for the disease a decompression room, where they remove slowly the nitrogen gas.

Another danger would be that oxygen get poisoned if the diver goes deep under the pressure of more than 1,6 bar.

Barotraumas, another negative effect that could occurs when diving. It is caused by the environment pressure and the pressure in air keeping holes in the body.

So to prevent long overpressure there is a overpressure of 0,6 till 0,8 bar needed, but exception the values could be lower.

Segment

People who are technical diving are professional divers. They have a good condition; their body can take the different pressure under water. They know what kind of risks there are. They have the right mentality and had a special training and planning on deep diving. They have to make decompression stops, otherwise they will get diseases.

- Professional diving are:
  - **Cave diving** needs special training and skills. Needs to set up the wire to cave entrance. There is poor visibility, less freedom of movement and bad view.
  - **For building underwater constructions**, technical construction, civil engineering or maintenance of fish farming.
  - **Military**, operations, building underwater constructions, ship repair underwater and secret infiltration.
  - **Search & Rescue**, for example fire department, gathering proof when researching.
  - **Scientific diving**, researching about marine biology and underwater archeology.
  - **Diving at night**

    Diving at night requires other configuration of device than diving during daylight. The circumstance under water at night is different. Not only that its dark, also because for example you have different kind of fishes, other animal creature that hunt at night, or behave differently than during daylight.

    At night the divers uses their lamp instead of hand signals to communicate under water.

Other diving purposes are:

- **Diving for fun** for example **'Go with the flow':** the divers get taken away by the flow.
- **Training at diving clubs**, everyone who want learn diving.
- **Underwater photography/movie industry**

Chosen segment for Delphinus

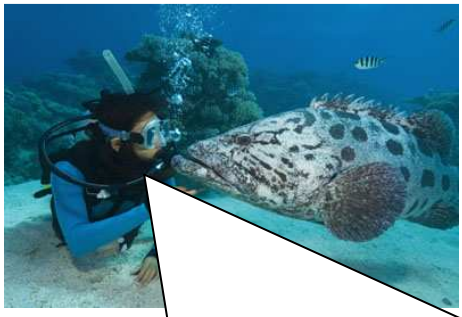
The main customers of Delphinus will be diving clubs where people are training to dive and are areas around the world where people dive. Delphinus can be used as a safe way to communicate underwater between divers. The customers can earn the investment for Delphinus back from the membership of the diving club. Other divers that are interested also can buy the device.

It is chosen for this segment because:

- The project started from the diving club Seahorse.
- Professional diving needs other communication than for example 'Go with the flow'.
- It is easier to sell to military, but making a rough estimate how useful Delphinus could be. It could be 50 % if they use some of the 52 command hand signals. But it can't really be estimated. Because in order to understand the military as 'user' there should be possibility to contact the military to win information about what kind operations they have and which information they need to send each other.
- **In short:** there is more information required as 'users' about the other divers in order to have a good communication. If the characteristic are known, it is recommended to start paper prototyping and involve the professional divers to get feedback. Only then it could be said what the professionals divers require and want when communicating underwater.

Characteristics of a diver

To understand what kind of users Delphinus now has. There is a persona set up. A persona represent the user, how they are in daily live and what kind characteristic they have. In this chapter there is a fictive story made about Jørgen Rasmussen, who represent the users of Delphinus.



My name is Jørgen Rasmussen. I like to be in shape, so I work out a lot. Beside of working out I now have been sport diving for 3 years.

I like different kind animals under water, so I always wanted to dive once. I am really impressed by the animal creature in the water. So once when we went diving, there were like hundred fishes with different color and shapes swimming around us. Its a totally completely different World under water. For me it is an adventure. I also made some pictures for my memories.

After diving for the first time I had first special training at the diving club, Søhesten. Now each year my friend and me go for holiday where there the sea or lakes are close by, where we could dive with other divers. We often discuss on PADI-diving society which place are nice to dive and give each other tips.

I now want to discover new things, so I want to learn deep diving. It is difficult, because you have to plan it very well, need a good training because the pressure and the risk are bigger. But maybe in two years I will be prepared for it and go deep diving for about 40 meter.

5 Casing

In this chapter the results of brainstorming will be shown in concepts, chosen material and manufacturing process.

5.1 Concept

In comparison to the previous design the new design should have bendable display and a better *shape*, smaller and smoother. The internal *components* and *sealing* should be better compiled. The *strap* should have better security.

Therefore it is the method morphological used for redesigning the previous digital communication device. The ideas from the morphological charts are raised from brainstorming and discussions if the meet the below requirements. The chosen redesign can be seen in the morphological chart.

The requirements when redesigning were:

- Bendable display
- The shape has to fit a *variety of users* on their wrist/arm.
- It has to be *lightweight* to be comfortable to wear.
- The device has no *sharp-edges.*
- The thickness has to be so less than the mock-up model, so it looks *smoother.*
- The display of Delphinus needs scratch *resistant* in case of accidents with surrounding objects.

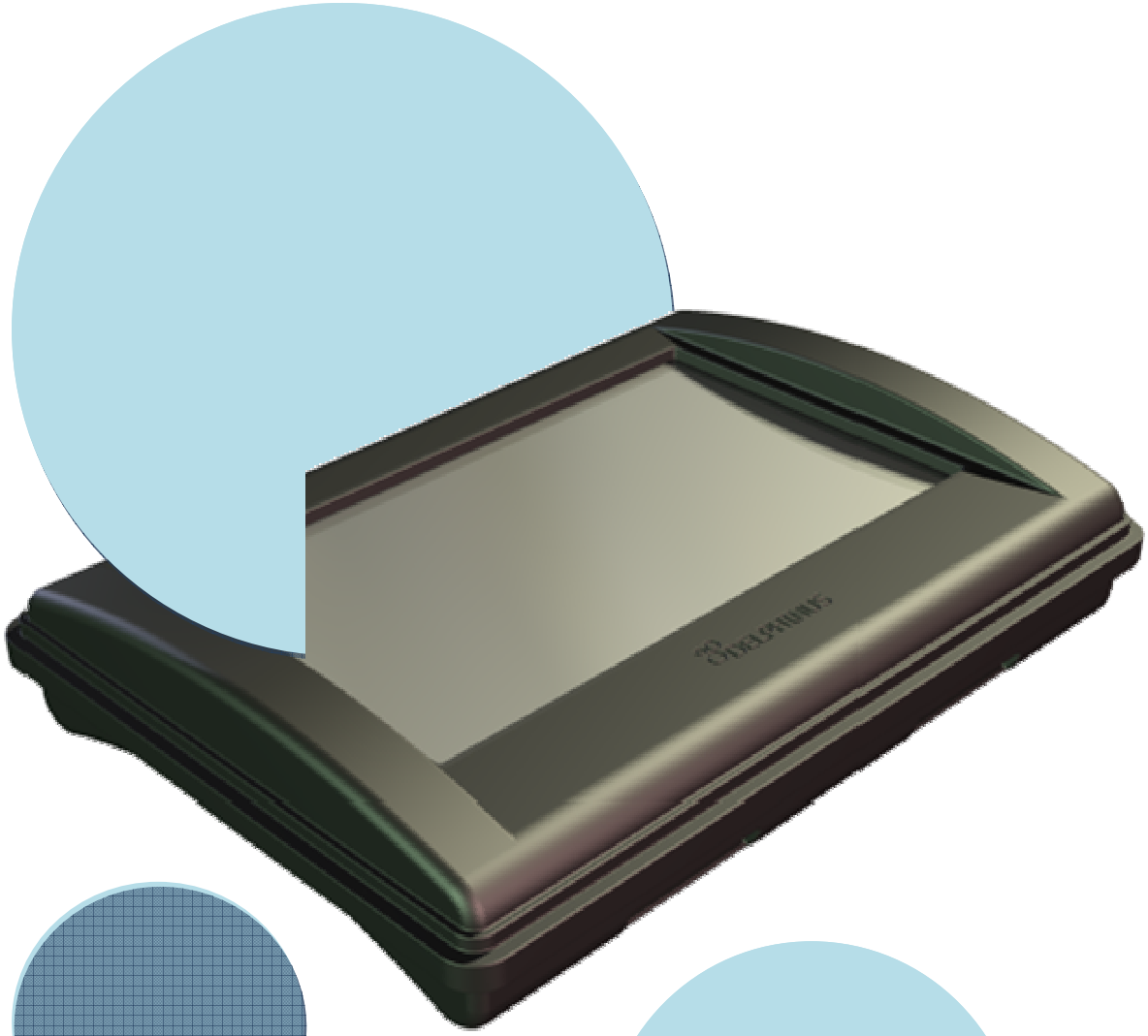
- The *sealing* is needs to be flexible to compile the components
- The assemblies must be *accessible* for repair and replacement.
- There will be *wireless charging* used to avoid physical connections.
- Delphinus needs a good strap for *security* on the arm.
- In order to be *water resistant,* the casing should be as one piece.
- When using the device in different environment it needs a good level of *chemical resistant.*
- The product is good *visible* in diving water.
- Delphinus should be *compact* within minimum depth of 10 m and with minimum length of 25 meters between divers.

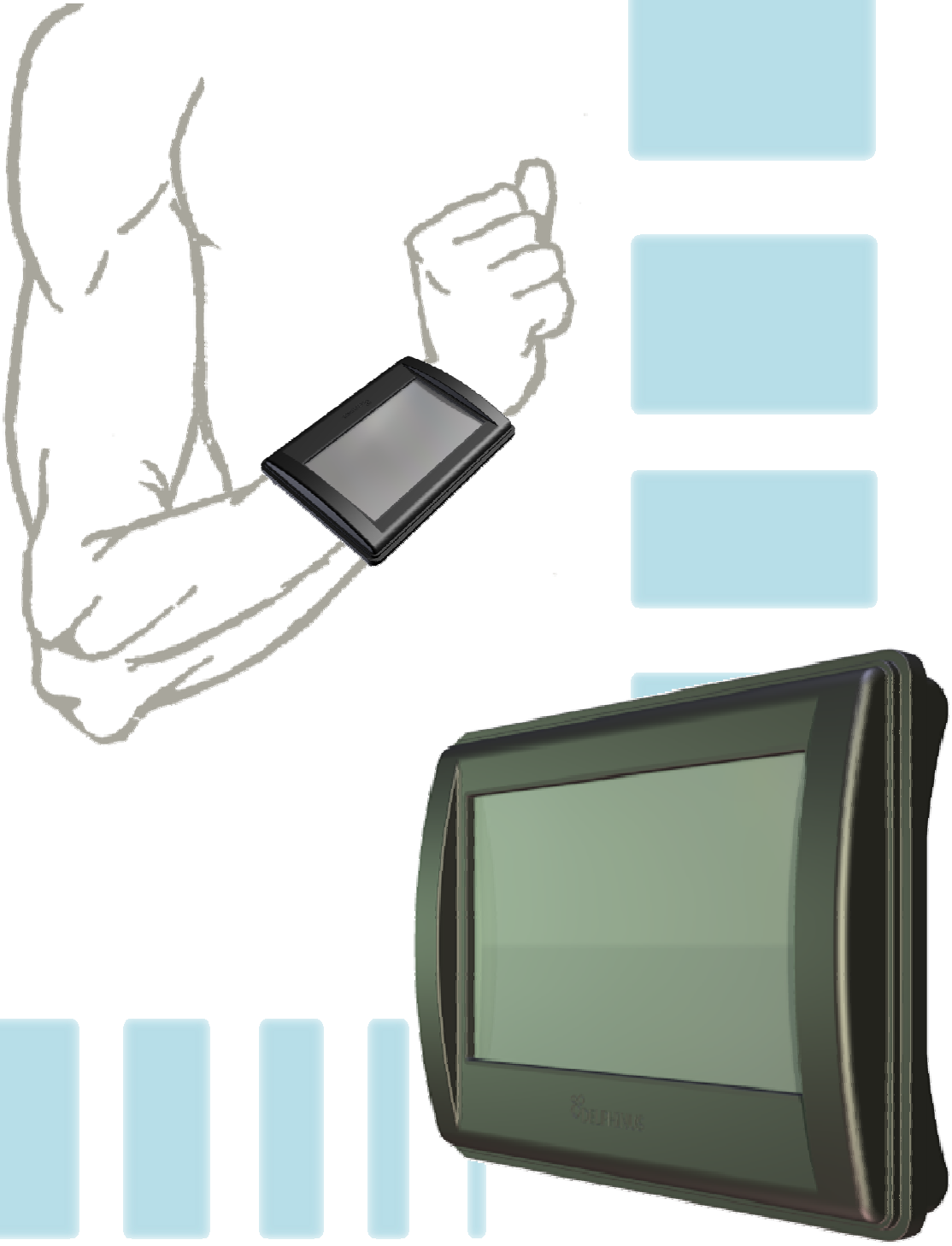
Chosen design

- There is a flat OLED display chosen. The bendable display is not available. Only prototype exists.
- There is a square and flat shape chosen because to compile the component sits easier and it is scratch resistant. The flat surface makes the device looks smoother and smarter.
- The component will be installed under the display, so there will be less thickness.
- The device could be used at both positions.
- In comparison to other straps, the click buckle is the most user friendly, because it is easier to adjust the length.
- For the sealing there is chosen the same system than waterproof boxes on the market use.

More info about the pros and cons of the design from the morphological is available in appendix chapter 4.1.

5.2 CAD model Sketch

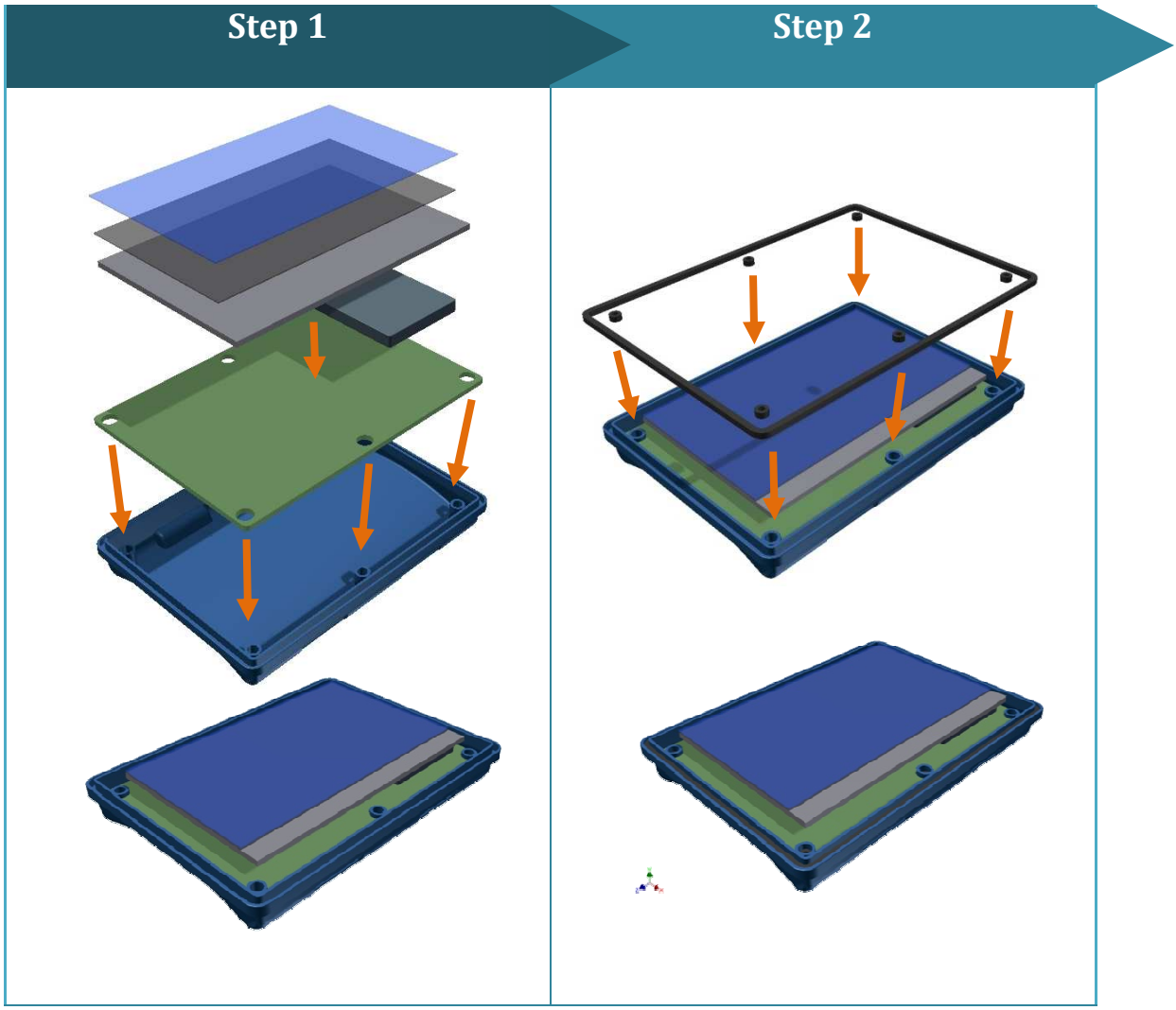


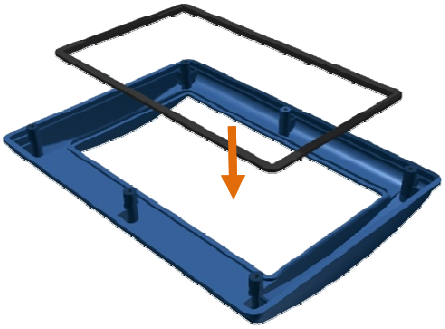
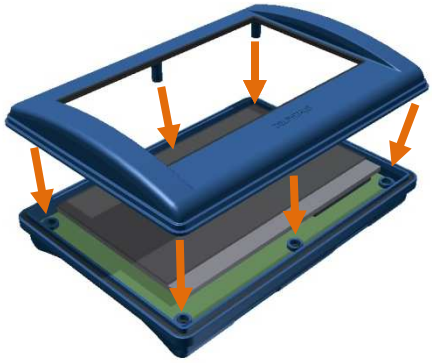
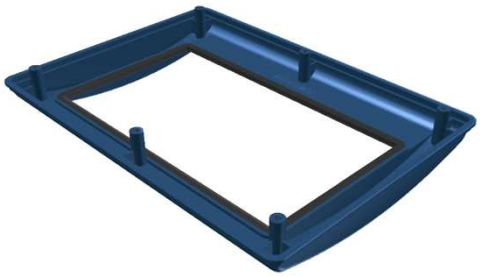


5.3 Assembly process

In this chapter the assembly process is showed. The process happens by doing six steps. The map in figure 5.1 shows how the assemblies are compiled.

1. •The mainboard and the electronical parts are inserted into the back casing
2. •The o-ring is fitted in the back casing
   •Screw rings are fixed in place. This step is repeated six times.
3. •The waterproof bezel is sealed with the upper casing
4. •The bezel is sealed with the screen protector
5. •Both casings are joined with the screws. This step is repeated six times
6. •The strap is passed by the hooks. This step is repeated twice


Step 1
Step 2

Step 3	Step 4
	
	

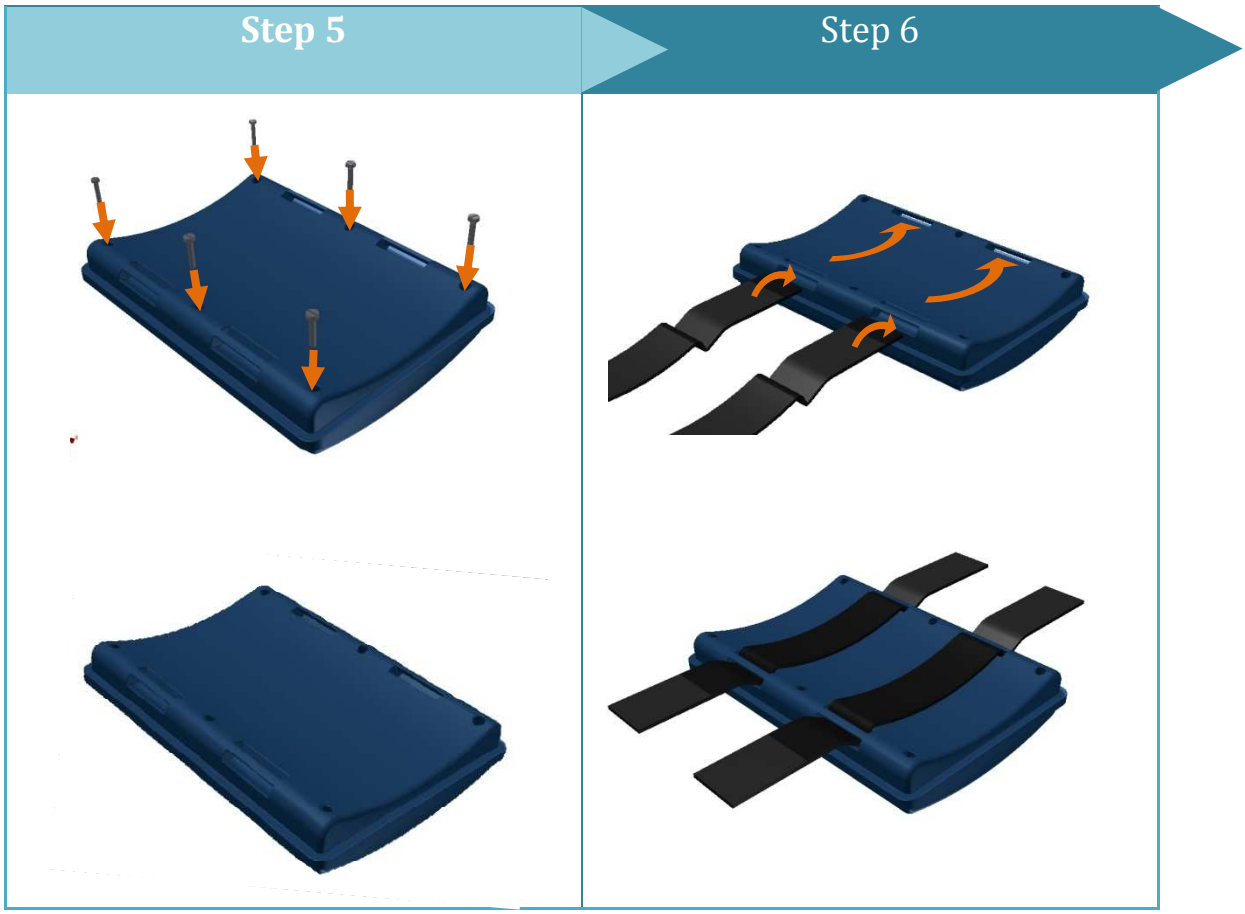


Figure 5.1 C, Assembly

5.4 Sealing components

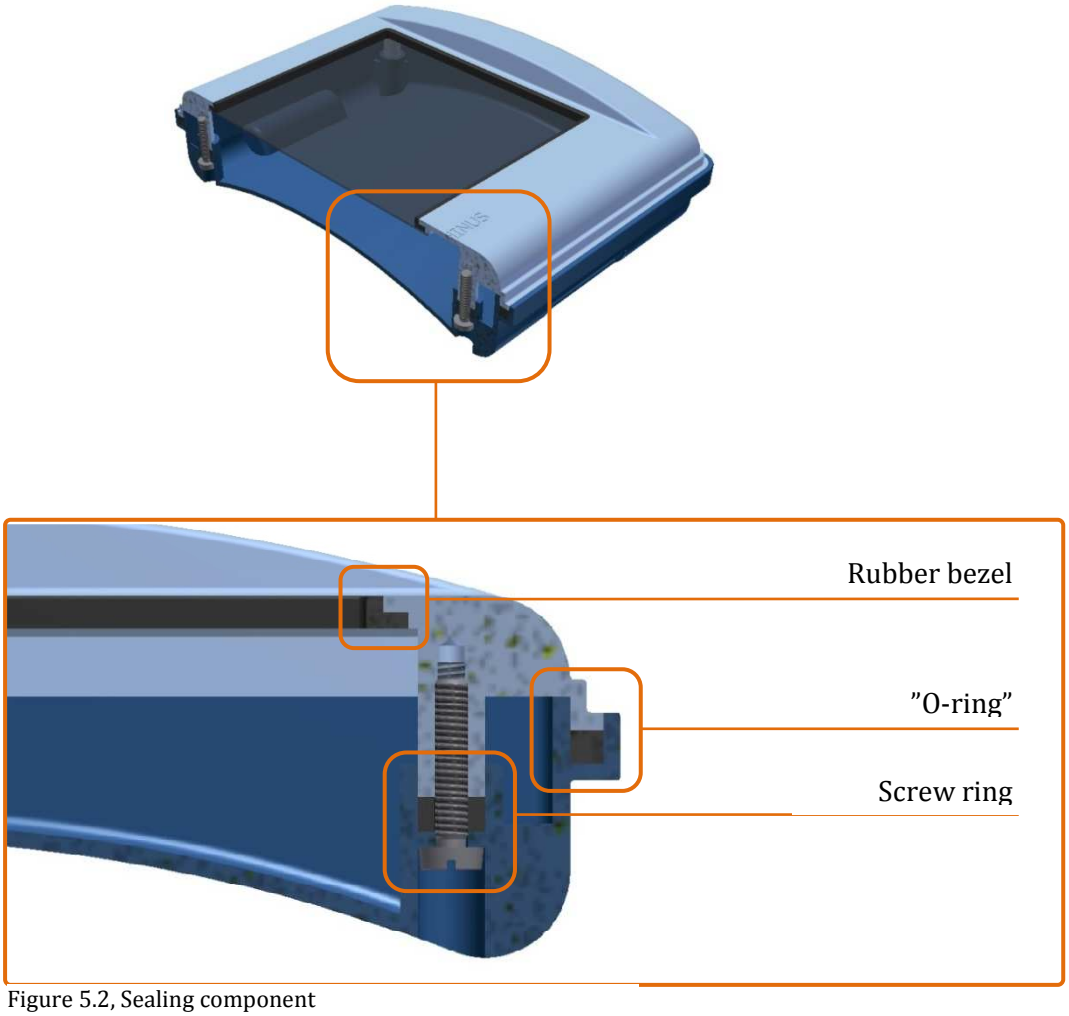


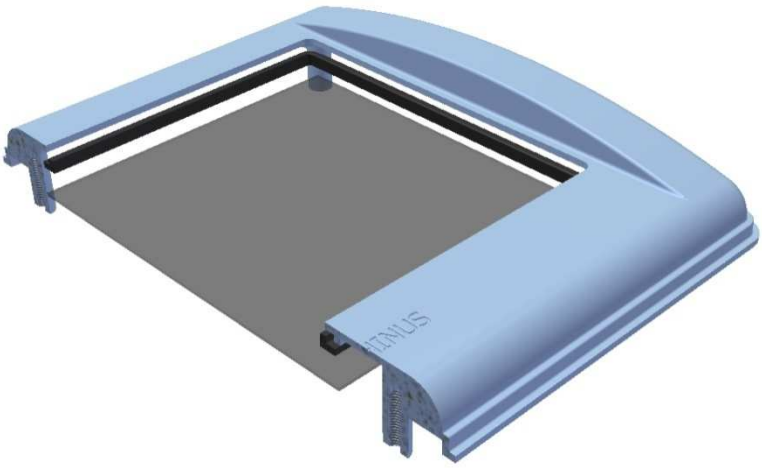
Figure 5.2, Sealing component

In this chapter the sealing of the components are explained, figure 5.2.

There is more information about the sealing of the first model added in the appendix, chapter 4.2.3.

1_ Rubber bezel

A bezel is placed between the top housing and the screen protector. It is a seal with flexible adhesive.

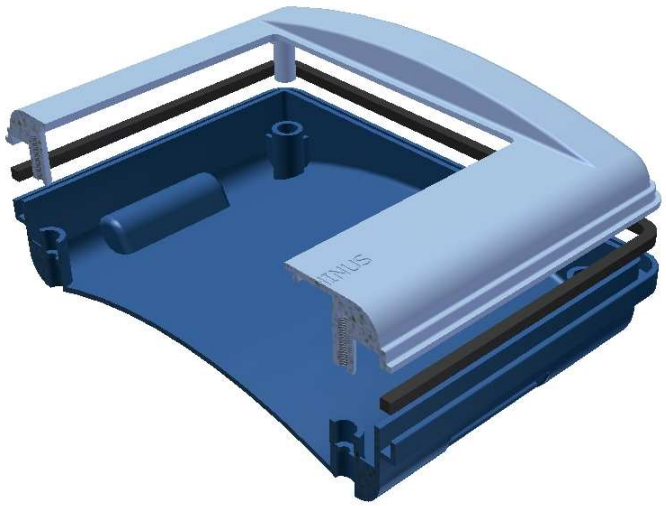


2_ "0-ring"

The same system that is commonly used in waterproof boxes on the market will be utilized for embedding the casing.

One of the housing has a U-shaped section. A rubber piece similar than an o-ring but with square section instead of circular is placed into the U-section.

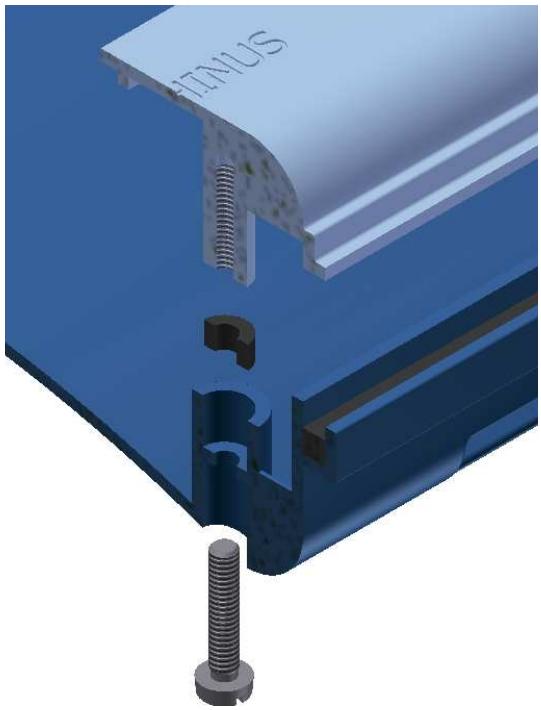
Thereby there are many contact surfaces preventing the water for entering inside.



3_ Screw ring

Delphinus will be used in environments with high pressures. It means the casing is subjected to high forces. For that reason, clipping systems aren't enough and screws are needed. The final design has six screws. Four of them are in the corners of the rectangular shape. The remaining two place in the middle, avoiding harmful effects from pressure.

A rubber ring is inserted between the casings allowing a perfect adjustment.



5.5 Materials and processes

Casing

The previous group had for the casing ABS. It is checked if there were better options. With the help of CES EduPack 2010 it is found out that ABS is the good choice. CES is also used to choose the other materials.

ABS is easily molded, because its characteristic. It is tough, resilient, and opaque. The process to build these pieces will be injection molding. It is the best way to produce small, precise, polymer components with complex shapes. Besides that the surface finish has a good quality.

It has good chemical and temperature resistance and high impact at low temperatures.

Acrylic-styrene-acrylonitrile has very high gloss; its natural color is off-white but others are available. It has good thermal and chemical resistance.

It is the stiffest of the thermoplastics and has excellent resistance to acids, alkalis, salts, and many solvents. It can be extruded, compression molded or formed to sheet. It is typical used for divers purposes like safety helmets, creational vehicles, refrigerator and fittings as in communication equipment or small housing appliances.

Sealing components (display and casing)

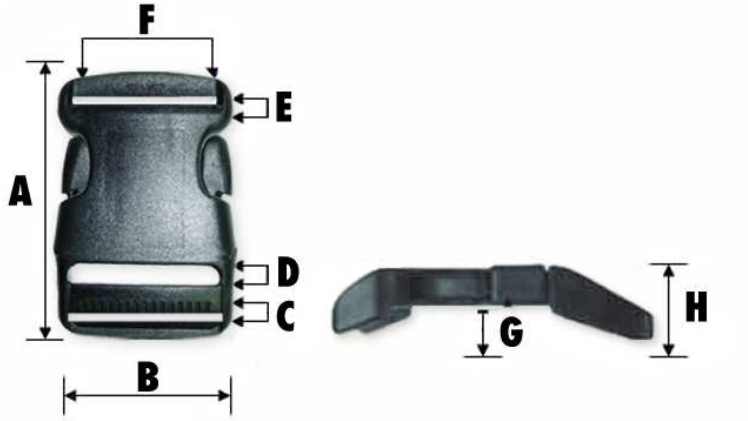
For the sealing there were two options. First option was polyurethane. This material is flexible and resistant to water, but not to acids. The second option is silicone. Silicones are high cost materials with good performance. It is chemically inert, do not absorb water and can be used in surgical or food processing equipment and seals. It could be produced differently.

Between these two materials, the best material for sealing the display and casing is silicone. Because there is not much silicone needed, it would not be expensive.

Strap

The strap must adjust for all size of users. The strap can be bought from the suppliers, http://www.strapworks.com.

The strap has a thickness of 1,5 mm and a breaking strength of 306 kg. The width of the strap is 19 mm and the length is 610 mm. Figure 5.3 shows the dimensions of the buckle in inches.



Size	A	B	C	D	E	F	G	H
3/4"	2.15"	1.07"	.15"	.15"	.12"	.79"	N/A	.36"

Figure 5.3, Buckle

The Side Release Buckle Straps is suitable for light duty outdoor applications. Heavyweight polypropylene webbing has excellent UV protect and does not absorb water quickly giving it better resistant to mildew and rot resistance. However, polypropylene does not have good abrasion resistance, so it is not recommended for use against rough edges. (2010, www.strapworks.com)

The dimensions and components from the final model is available in appendix, chapter 4.3.

5.6 IP code

It is a requisite during the development of an electrical device to assign an IP standard, which indicates the level of protection of its enclosure from solid object and liquids (explaining in appendix chapter 4.5).

IP65 and IP54 are the most commonly used in the manufacture of electrical products. An IP65 rated product will be fully protected against dust and airborne particles whilst also be protected against water jets which would allow the machine to be washed down, while an IP54 rating offers dust protection (but not total) and protects against splashing of water but not wash-down. Delphinus must be totally waterproof. It will be used in continual submersion underwater and even in high pressures. In this way it is needed a higher IP rating. IP68 would provide complete dust and underwater protection.

To have qualified for IP68 the casing has had to be sealed so as to prevent dust and water from invading the sensitive internal components. Unfortunately this means if the device has to be repaired or any of its components has to be replaced then a technical expert would need to be called. The user cannot repair it himself.

5.7 CE marking

The CE mark can be applied by the manufacturer through a "self certifying" procedure that verify that products are designed to the appropriate standards.

The manufacturer is responsible for non-compliance and liable for any damage caused by the product. If the manufacturer (or his authorized representative) is not based within the EU, the importer is responsible for the product in Europe.

If a product is not in compliance with the directives, it may be restricted, prohibited from sale or even withdrawn from the market.

Further information about the steps to CE marking a product is in the appendix chapter 4.4.

6 Interface

6.1 Concept

While making the interface there has been changes made. Some parts of the concepts are explained in this chapter. The prototype is explained in the next chapter.

Figure 6.1, Main menu

Figure 6.1 shows the main menu. By touching the screen this screen always appears.

The two buttons, menu and settings can be clicked. The two other buttons has to be dragged.

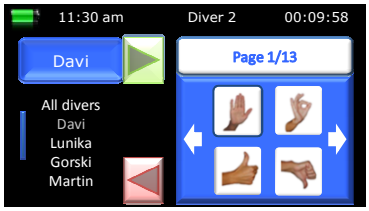


Figure 6.2 A, Sending

Figure 6.2 B, Receiving

Figure 6.2 shows the first interface of how to receive a message. At the left side is the receiver list. At the right the received message and info about the diver and time can be seen. The red cross is for deleting the message and that means going back to the main menu, figure 6.1.

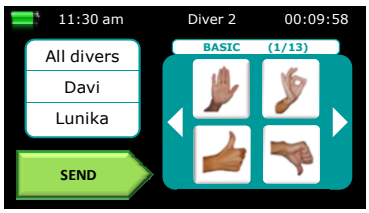


Figure 6.3, Sending

Figure 6.3 is a second version of figure 6.2. The cancel button wasn't necessary. If the user does not use the touch screen, it would go off.

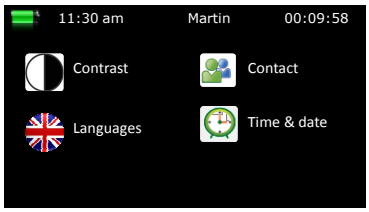


Figure 6.4, Settings

Figure 6.4 show the settings from the main menu. The settings is especially necessary when setting timer and adding contacts. In settings a diver, user from another Delphinus could be added and saved to the receiver list.

Signs

The signs for Delphinus are used from the website www.ukdivers.net. They are sorted in categories.

The categories Other, Training and Air are joined because they are small category. Instead there are two other categories added. These categories are scale and difficulties.

The category scale has a scale from zero to 9 and main of the other is to show what kind of difficulties the user has.

Figure 6.5 A, Basic

Figure 6.5 B, Distress

The sign in figure 6.5 A and 6.5 B are changed. There are some layers added to show the movement of the hand.

6.2 Interaction

Delphinus is always on. To use Delphinus the touch screen should be touched. The users can use the touch screen with their gloves because the touch screen is resistive so it works with pressure.

Battery

Before using Delphinus it has to be charged. There will be always feedback from the battery level, figure 6.6. During charging the battery shows that is charging, at low level the colour changes and when it is almost empty it gives a warning to charge it.

Figure 6.6, Battery levels

Main menu

Figure 6.7 shows the main menu of the interface. **Red button:** It has to be slid to the right direction where the arrow points. This message will be received by all divers. **Yellow button:** It has to be slid to the left direction where the arrow points. All the divers gets a message that the user needs helps. **Blue button:** With this button there can be send messages. **Setting:** In this section the language, time, timer, contrast can be set. There are also the possibilities to see which divers are added in the divers list, add more and to delete some.

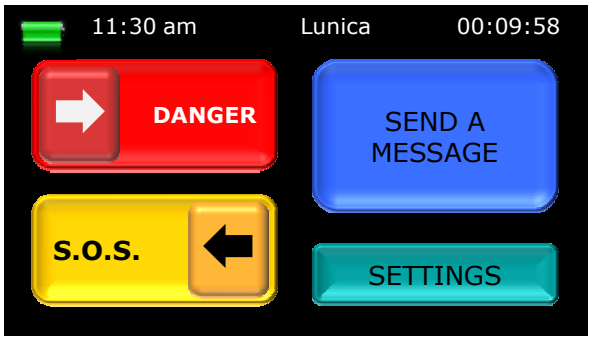


Figure 6.7, Main menu

Send a message

To send a message the user have to choose a receiver, to whom he want to send it. The second step is to choose a category of the message the user wants to send. After this the user can choose a sign. The user will be now in the screen to send the message. The process of sending the message is shown with a squared timber.

Between the steps of sending a message the user always can go back to his previous step.

Another option is to go 'Home'. In this case the user will be back at the main screen. There also is the possibility to use the two emergency buttons all the time as it can be seen in figure 6.8.

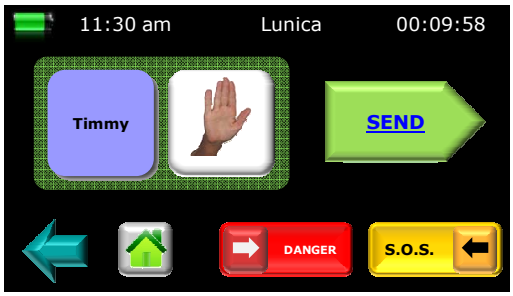


Figure 6.8, Send

Receiving a message

When receiving a message Delphinus vibrates and it is turned on. The message and the diver info is shown on the screen. With the reply button (figure 6.9) one step is passed over, because there is no need to choose the receiver.

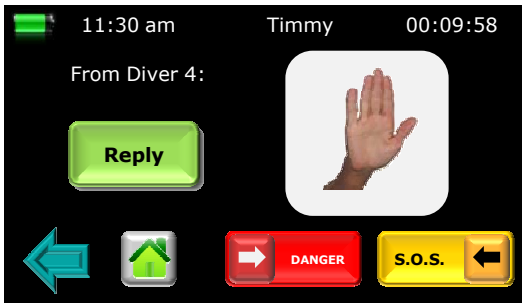


Figure 6.9, Receiving

Interruption

There is possibility for interruption when: choosing a receiver, category, or choosing the sign and when sending message. At all time the incoming message has to be shown. When the button to go back id used, the user gets back to the step where he was before the interruption. Only if there was a sign selected, the user gets to screen of sending the message.

7 Electronics

7.1 Display

Display options

First of all, it must be known that there are a lot of different types of displays. We were hesitating whether use LCD (Liquid Crystal Display) or OLED technology. An OLED (Organic Light Emitting Diode) is a new family of screens in which the emissive electroluminescent layer is a thin film of organic compounds which emit light by itself in response to an electric current.

In terms of the advantages of using this type of display comparing to LCD technology, these are several. In low ambient light conditions such as dark rooms, an OLED screen can achieve a higher contrast ratio than an LCD screen using either cold cathode fluorescent lamps or the more recently developed led backlight. OLEDs can enable a greater artificial contrast ratio (both dynamic range and static, measured in purely dark conditions) and viewing angle compared to LCDs because OLED pixels directly emit light. Their colors appear correct and entire, even as the viewing angle approaches 90 degrees from normal. This is a factor to consider in our application, because of the underwater environment. They are also thinner than any LCD screen.

And finally, while LCDs filter the light emitted from a backlight (allowing a small fraction of light through so they cannot show true black), an inactive OLED element produces no light and consumes no power, so they have better power efficiency (2010, http://en.wikipedia.org). On the other hand, we found that LCD displays are more available in the market. There are much more devices in this type of screens, so it is easier to find a LCD out than an OLED one, which is a very important point to have in mind as well.

The OLED we considered to buy was limited to only two models: Bolymin bl043acrnb$ and Densitron p0430wqlc-t (2010, http://es.rs-online.com). The rest of OLED displays were either too small for our interests or too large. However, we found a lot of accurate LCDs in the meaning of the dimensions. The best examples of products with this said technology are Sharp lq043t1dg01 and Electronic Assembly a dip240j-7klwtp (http://dk.mouser.com).

The LCD from Sharp is a display with touch screen included in it, such as Densitron's model. The other two products, however, are only screens. We decided to use a simple display, without touch screen, because it is easier to find an accurate keypad after choosing a right display. Once we have the display, we can find a keypad out, join it to the screen, and program it helping of the microcontroller. So we should select one of the displays between the Bolymin and Electronic Assembly models. The price of both of them is very similar, so we thought that we should take advantage of the OLED qualities.

The choice between displays

Therefore, the best option for was *Bolymin bl043acrnb$_display*. The document is attached to the appendix. It is able to see an image in figure 7.1.

Figure 7.1, Bolymin display

Focusing on its general characteristics, they are listed as following:

- Type: full color AMOLED (Active Matrix OLED) display. RGB 16M color.
- 4.3" screen.
- Active area: 95.0 x 53.8 mm.
- Resolution: 480 x 272.
- Voltage supply: +2.8 V.
- Interface: 8-bit serial RGB & 24-bit parallel RGB.

The product specification is attached in the datasheet document *displaySheet.pdf* in appendix which gathers all the technical information about the display (2010, http://docs-europe.origin.electrocomponents.com).

Since our project has such specifications, this display fits accurately to them. The OLED color properties make it easy to see the screen in underwater and in dark conditions. It has low battery consumption, so the diver will can use the device for a long time without worry. Finally, the price is not much higher than other similar displays. It costs 177.19 €, while LCDs are also about that amounts (depending on the specific model), but this one has better features.

An active matrix OLED display consists of a matrix of OLED pixels that generate light upon electrical activation that have been deposited or integrated onto a thin film transistor (TFT) array, which functions as a series of switches to control the current flowing to each individual pixel (http://en.wikipedia.org/wiki/Active-matrix_OLED).

RGB is the acronym of Red-Green-Blue. It is a standard of screens and displays interface that manages each pixel in the image as a mixture of intensities of the three basic colors. In our case, each color is indicated with 8 bits, so one pixel is defined by 24 bits, which allows a total of 16,777,216 colors (approximately 16 million of possible colors).

Anyway, we must choose between 8-bit serial or 24-bit parallel interface. The difference lies in the way of transferring the data. The parallel interface is faster than the 8-bit serial one, since it sends three times the amount of data of the last one in the same time. The problem is that our microcontroller is limited by the amount of ports, so we use the 8-bit serial interface.
In terms of the screen, this display has a 4.3 inches screen, with an active area dimensions of 9.5 centimeters by 5 centimeters, approximately. We thought they were the accurate size for the display because it should fit to a diver's forearm, so it could not be too long or wide. However, it is needed a large enough screen that allows big buttons in the menus, in order to make it easier for the user to identify letters and images underwater, where the seeing capability is, sometimes, very low.

As it can be seen in the display specifications file displaySheet.pdf, attached in the appendix chapter, there are different chronograms, time schedules, and examples of some useful instructions to handle the data between microcontroller and display to achieve this can show the wished images. Although that has been the basic document where we have informed about the display, it has not been enough because it does not explained how to use those instructions, how it is the right way to communicate with the ATMEGA16 or what kind of instructions the display driver uses.

We had to research much more in order to find that information out. After investigate, we found the datasheet of the HX5116-A, the display driver (http://www.andilcd.de). It talks about the displays functions or modes, the serial and parallel interface and the power specification.

7.2 Connections

Connector

Looking at the display datasheet, there is a useful sketch figure 7.2, which indicates the most important pins and connections between the display and the controller device, in our case, the ATMEGA16L.

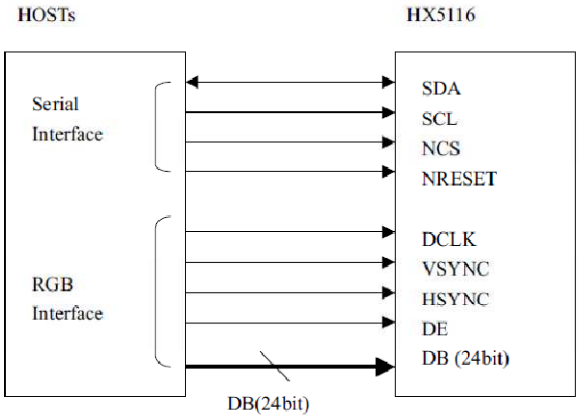


Figure 7.2, Connector

As it is been seen, there are two types of interfaces: the serial one and the RGB interface. The serial interface is used, as we can see in the driver datasheet, to fix the mode that the programmer is going to utilize. There are many different formats to use through the RGB interface, as it is watched at the table 7.1.

Table 7.1, Interface Format Select

IFS[3:0]	Interface Format Select
0000	8-bit serial RGB (SYNC)
0001	8-bit serial RGB (DE)
0010	24-bit parallel RGB (SYNC)
0011	24-bit parallel RGB (DE)
0100	8-bit RGBDummy 24.54MHz
0101	8-bit RGBDummy 27MHz
0110	CCIR601 mode A 24.54MHz
0111	CCIR601 mode B 24.54MHz
1000	CCIR601 mode A 27MHz
1001	CCIR601 mode B 27MHz
1010	CCIR656 mode A 27MHz
1011	CCIR656 mode B 27MHz

However, there are only two considered options, and examples of their instructions, in the datasheet. These are the 8-bit serial RGB (DE) and the 24-bit parallel RGB (DE). Therefore, one of them will be used, because there is more information than for others.

In order to select it, we must use the serial port of the microcontroller. This is the USART (Universal Asynchronous/Synchronous Receiver Transmitter). Next section will talk about software, so an explanation of this application will be included there.

As it was said, the figure 8.2 includes the most important pins in terms of the interfaces. Nevertheless, every pin is rightly indicated in the display specification sheet. There are 71 pins in total, but not all of them are used. Depending on the kind of interface we use, we should connect different pins to the microcontroller. For instance, if 8-bit serial RGB interface was finally used, we should connect eight ports from the microcontroller to the display, instead the necessary 24 ports for the 24-bit parallel RGB. The benefits of using this last interface are that we would achieve more speed in the data transfer. However, if 8-bit serial RGB is used, the advantage is that only that amount of ports is needed. Our microcontroller has four different

ports with ten input/output pins each one. Many of them will be used for ADC or DAC functions, USART, timer or PWM, and also for general outputs, like high or low level voltage in the display, what limits us in terms of ports. Therefore, 8-bit serial RGB will be used.

We also must take care about the power supply pins. These are intended to determine some characteristics about the voltage or power, and it is not only important, but also mandatory to fulfill these requirements: analog ground, negative and positive voltage for OLED, power supply for analog and digital circuit and external ground. It is important as well an external capacitor in the pin number 31, with a 1µF capacitance.

Anyway, a scheme of the display is included in the last page of the datasheet. There, we can see detailed dimensions of the screen and the device, and some important notes to consider in the connections, it means, that it is needed and specific socket to wire the display to the microcontroller. It is called HIROSE FH-26-71S-0.3SHW, whose datasheet is attached in the appendix chapter, with the name *fh26-71s-0.3shw.pdf.* There were several problems while looking information for about this connector, since the display is not popular and the connector is very special, but the document is enough to make the connection possible. Apart of some features and other data about this device, there are some instructions which explain how the connector has to be handled. Connect it to the display is easy. However, in order to join the microcontroller to the screen, it is needed a PCB board, because the 71 pins are very close between them and it was not possible for us to solder cables on the chip. So we should manufacture a board with tracks on it which will allow to wire the socket. In one side our FH-26-71S-0.3SHW is connected. In the opposite extreme the tracks are wider, so we can solder our microcontroller to the right connections without problems.

In terms of the most important connections in our device, it is needed to emphasize those which are drawn in the previous scheme, in figure 7.2. The SDA pin, although it is a both directional one, we will use it only in one way because it is only to set the display mode, so the USART from the microcontroller will send data to the display through this pin. The other three pins in the serial interface will received the accurate signals from general I/O ports. Finally, as it was commented previously, we used the 8-bit serial RGB mode to program the display. So the DB signal is a block with 8 bits from general I/O ports as well. The other pins in the RGB interface are digital signals, so I/O ports are enough to manage them. The only special pin in this interface is DCLK. This signal is very important because the data transfer is made synchronously, so we should be careful to keep this synchronism in the right time. This signal is created by our microcontroller, using the PWM or timer functionality. With this application we can make a periodic square signal with the wished period. Looking at the different chronograms in the display datasheet we can decide the length of this period. All these connections are included in figure 7.3. Besides, there is another different scheme of the connections between display and microcontroller in the appendix chapter.

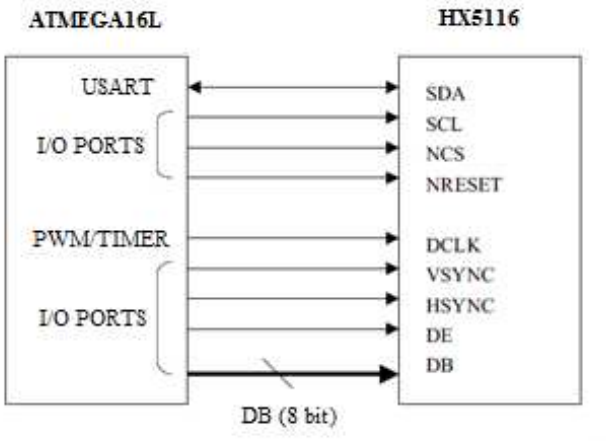


Figure 7.3, ATMEGA16L

7.3 PCB

PCB (Printed Circuit Board) is used to mechanically support and electrically connect electronic components using conductive pathways, tracks or signal traces etched from copper sheets laminated onto a non-conductive *substrate*. It is also referred to as printed wiring board (PWB) or etched wiring board. A PCB populated with electronic components is a printed circuit assembly (PCA), also known as a printed circuit board assembly (PCBA).

PCBs are inexpensive, and can be highly reliable. They require much more layout effort and higher initial cost than either wire wrap or point-to-point construction, but are much cheaper and faster for high-volume production. Much of the electronics industry's PCB design, assembly, and quality control needs are set by standards that are published by the IPC organization (2010, http://en.wikipedia.org).

As it was said, our connector has 71 pins. Distance between them is 0.3 mm, as it can be seen in the datasheet (*appendix, fh26-71s-0.3shw.pdf).* This is a very short space, so it is very difficult to solder cables from the microcontroller so close. In order to solve this problem, we had to make a PCB board with 71 tracks which get farer, so we could work more comfortably.

The first step while creating your own board is to draw your sketch. In this case, we must look at the exact dimensions in the connector specification. Only two of them are interesting for us, distance between pins (0.3 mm) and width of the pins (0.12 mm). We must draw 71 tracks, like it is shown in figure 7.4:

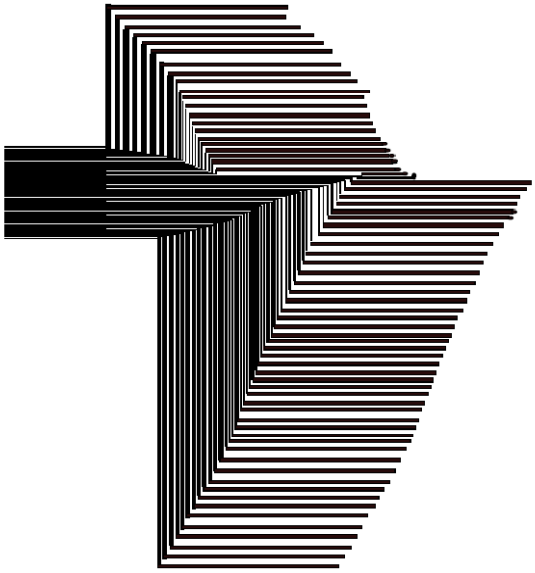


Figure 7.4, PCB circuit

In the upper image are not well seen the different lines because they are very close to be distinguished, so next figure (figure 7.5) shows a zoom in the thinnest lines at the left of the sketch:

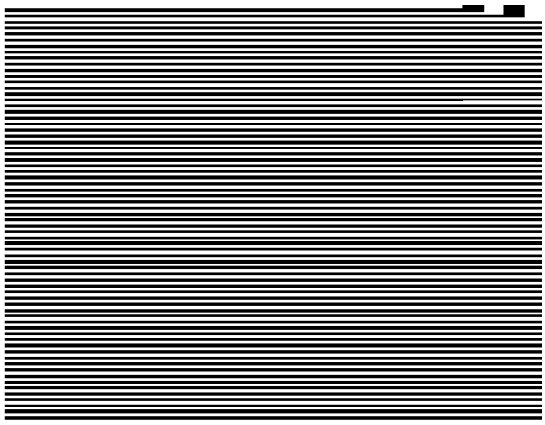


Figure 7.5, Zoom PCB circuit

Once we have the drawing of the desired circuit, you must print it out in a special transparent paper, with the real dimensions of the device in a scale 1:1. This paper will be used to copy the tracks on the board, which is provided by the store in the school. A process must be followed in

order to achieve the circuit will be recorded on the board. After this process, the tracks are ready to drive current.

7.4 Programming

First of all, it is necessary to know the environment in which we are working. Our display is controlled by the ATMEGA16L, a microcontroller from ATMEL Corporation. This company has its own family of controllers, and they use their own tools. AVR Studio 4 is a program for Windows that allows program in codes language C for a lot of different microprocessors and controllers. A screenshot of this program is watched in figure 7.6:

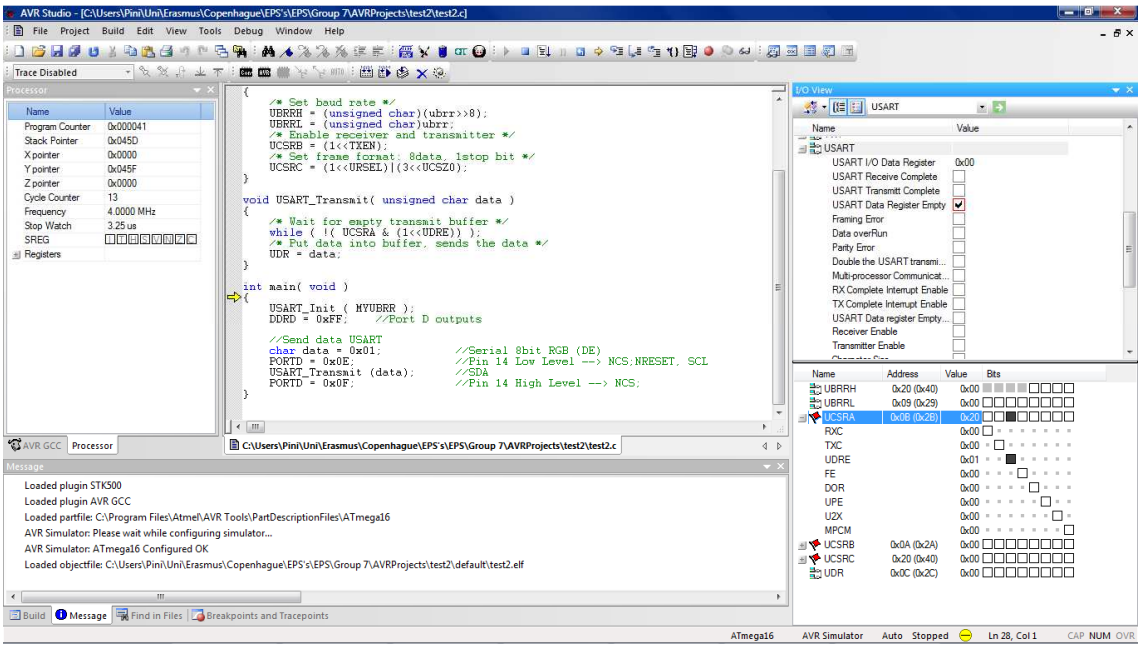


Figure 7.6, Screenshot

The possibilities of this tool are very wide, but in our case we just work with the main and basic features.

As it has been seen, the display requires some specific instructions which are sent through the USART port of the microcontroller. This device has been studied more in detail in its own chapter, but it must be known than there is a concrete functionality of the ATMEGA16L to transmit data in a serial mode. USART (Universal Synchronous/Asynchronous Receiver Transmitter) allows sending data like a bits streaming. Some of the features of this device are:

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous Operation
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty, and RX Complete
- Multi-processor Communication Mode
- Double Speed Asynchronous Communication Mode

Many of them are not necessary in our project, since we only use the USART to set the display mode. In our case, the USART has been set with 8 data bits, 1 stop bit, no parity check and 9600 bps (bits per second) Baud Rate. In the appendixes it is possible to see the code destined to configure the display with the necessary parameters.

As important as the USART are also the ports which are used to send the data to the display. They are input/output ports of general purpose, but if we utilize them in the right way we can

send data in hexadecimal format that makes an image appear in the screen. As well as the USART, there is a simple example attached in the appendixes about how they work.

7.5 Touch screen

A touch screen is a kind of screen that reacts to touch. There are 4 types of touch screens – depending on techniques of construction using: infrared, surface acoustic wave, capacitive and resistive.

Infrared

Interruption (when touch on the screen) in beam of infrared light, which is being emitted by network of LED diodes, which are situated on the edges of the screen.

Technology uses an array of X-Y infrared LED and photo detector pairs around the edges of the screen to detect a disruption in the pattern of LED beams. These LED beams cross each other in vertical and horizontal patterns. This helps the sensors pick up the exact location of the touch. A major benefit of such a system is that it can detect essentially any input including a finger, gloved finger, stylus or pen. It is generally used in outdoor applications and point-of-sale systems which can't rely on a conductor (such as a bare finger) to activate the touch screen. Unlike capacitive touch screens, infrared touch screens do not require any patterning on the glass which increases durability and optical clarity of the overall system.

Surface acoustic wave

There are disorders of acoustic wave, which is being propagated on the screen. Technology uses ultrasonic waves that pass over the touch screen panel. When the panel is touched, a portion of the wave is absorbed. This change in the ultrasonic waves registers the position of the touch event and sends this information to the controller for processing.

Capacitive

Changes of electric capacity of the screen. Screen doesn't react with pencil or hand with glove. This technology allows using multi-touch. Panel consists of an insulator such as glass, coated with a transparent conductor such as indium tin oxide. As the human body is also a conductor, touching the surface of the screen results in a distortion of the screen's electrostatic field, measurable as a change in capacitance. Different technologies may be used to determine the location of the touch.

Surface capacitance - only one side of the insulator is coated with a conductive layer.

Projected capacitance - permits more accurate and flexible operation, by etching the conductive layer.

Resistive

Changes of electric resistance between transparent electrodes which are set on the screen. Panel is composed of several layers, the most important of which are two thin, metallic, electrically conductive layers separated by a narrow gap. When an object, such as a finger, presses down on a point on the panel's outer surface the two metallic layers become connected at that point.

Usually used in palmtop technologies. LCD screen is being bonded together with touch-sensitive part, which consists of 2 transparent electrodes, which are letting in over 85% of light.

Best in different categories:

- Durability: Surface Acoustic Wave, 5-Wire Resistive, Infrared
- Stability: 4-Wire Resistive, Surface Acoustic Wave, 5-Wire Resistive, Infrared
- Transparency: Surface Acoustic Wave, Infrared, Capacitive
- Touch:
    - Anything - 4-Wire Resistive, 5-Wire Resistive
    - Finger, pen - Surface Acoustic Wave, Infrared
    - Conductive – Capacitive
- Waterproof: 4-Wire Resistive, 5-Wire Resistive, Capacitive

7.6 Battery

Delphinus needs a chargeable battery, which allows using Delphinus underwater during some hours. Like this project has been developed by other groups previously, we can take some studies from them, for instance, this one. They used an accurate battery for this purpose. The name of the model is 054562, and the following table 7.1 reflects the battery features:

Table 7.2, Battery features

Model	Typical capacity(mAh)	Minimum capacity(mAh)	Nominal Voltage(v)	Approx. Weight (g)	Dimension (mm) Thickness x Width x Length
054562	500	100	3.7	12.0	4.85 x 34.5 x 61.5

Next image 7.7 shows the battery's dimensions:

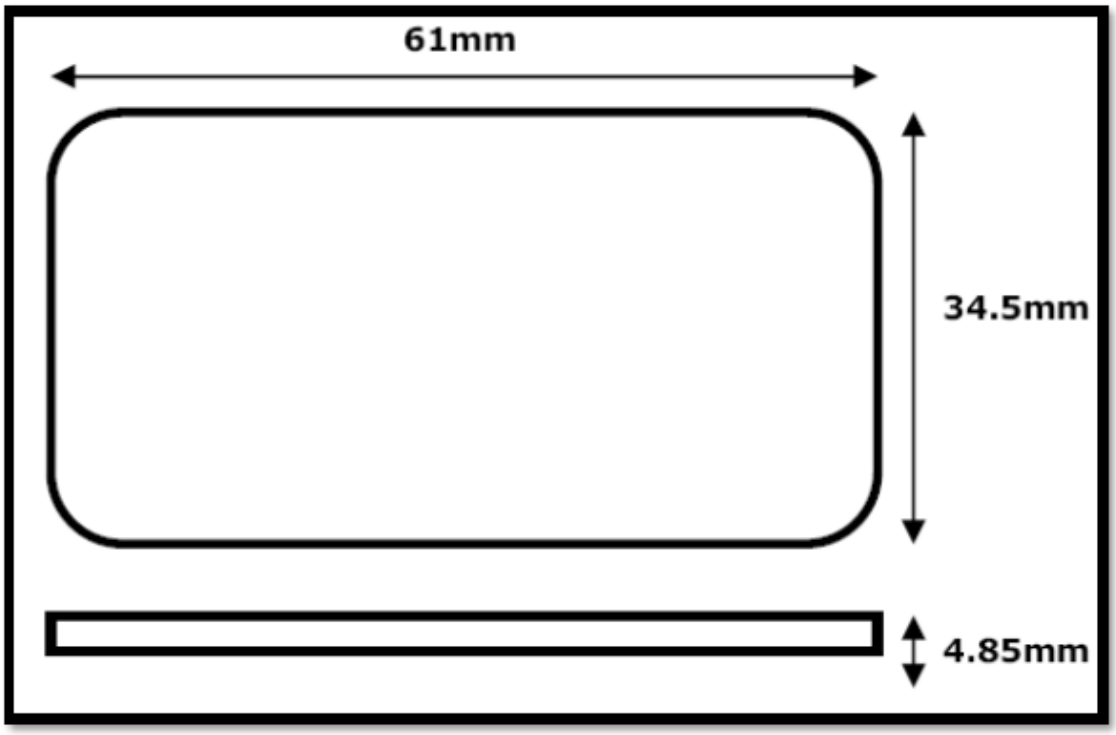


Figure 7.7, Battery dimensions

As it is possible to see, the battery is small enough to fit inside the casing and it does not take much space. To read more about it, there are projects from previous semesters available.

7.7 Microcontroller

Figure 7.8 shows the microcontroller that is used in the project, ATMEGA16L .

Figure 7.8, ATMEGA16L

To make the project working there was a microcontroller needed. To choose microcontroller it was considered :

- Its frequency;
- Recommended level of voltage from power supply;
- How big is the memory;
- Total amount of pins (ports);
- Total amounts of interruptions;
- If it's possible to program it in system (ISP)
- Quantity and sort of peripherals.

ATMEGA16L

The best option is the ATMEGA16L because they are equipped with:

- 16 Kbytes of In-System Self-programmable Flash program memory;
- 512 Bytes EEPROM;
- 1 Kbyte Internal SRAM;
- Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes;
- One 16-bit Timer/Counter with Separate Prescaler, Compare Mode;
- Real Time Counter with Separate Oscillator;

- Four PWM Channels;
- 8-channel, 10-bit ADC;
- 8 Single-ended Channels;
- 7 Differential Channels in TQFP Package Only;
- 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x;
- Byte-oriented Two-wire Serial Interface;
- Programmable Serial USART;
- Master/slave SPI Serial Interface;
- Programmable Watchdog Timer with Separate On-chip Oscillator;
- On-chip Analog Comparator;
- 32 programmable I/O lines;
- Operating voltages 2,7-5.5 V;
- 0-8Mhz of speed.

The power consumption is like following:

- Active mode 1.1mA;
- Idle mode 0.35mA;
- Power down mode < 1uA;

On figure the description of ATMEGA ports can be seen.

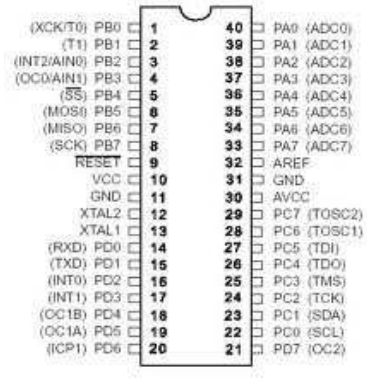


Figure 7.9 ATMEGA ports

Despite their internal assets they are cheap and easy to find. To program this kind of microcontrollers one can use assembler language and C language. Assembler is very close to machine language so it is hard to understand for human beings. Assembler is also fast and allows controlling assets of microcontrollers. We choose C language because it's great compromise between low and high level programming languages.

Using C we can maintain speed advantages with "human appearance" of language at the same time. What is interesting is that AVR microcontrollers have an architecture and set of orders dedicated to compilers of C language. The outcoming code from C compilator is a little bit longer and slower than the same code written directly in assembler, but the time needed to write code and checking if any mistakes exist is much shorter than using high level programming language.

For ATMEGA microcontrollers there are few programming environments but the most popular are:

- AVRSTUDIO;
- CODEVISION.

It is possible to download a trial version of CODEVISION, but size of code is limited to 3kbytes. AVRSTUDIO is for free and has a good support and without any limitations.

To make it work it should be installed on existing C compiler like WINAVR because AVRSTUDIO doesn't have its own compiler. The most useful function of AVRSTUDIO is simulation of programs. By clicking on proper peripheral or port one can see what's happening during process.

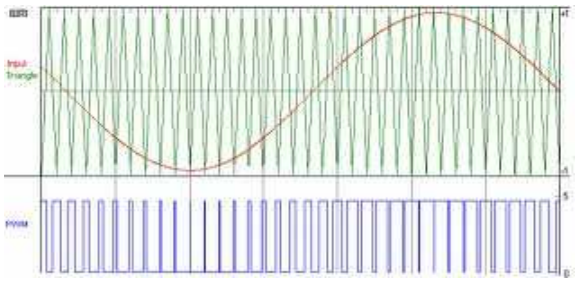


Figure 7.10, PWM

To generate and send massages good option is to use pulse width modulation which is abbreviated to PWM. PWM gives us a possibility to connect the digital world with analog(natural) world, figure 7.10.

Thanks of the PWM we can set an overall frequency, and then change the "on time" and "off time" of the output inside each timer cycle. If the duty cycle is longer gives analog representation closer to VCC voltage. If the duty cycle is shorter the analog representation is closer to GND.

Thanks of that we can use frequency to create 0-1 communication. Depends on frequency we have 0 with lower frequency and 1 with higher frequency.

Stk 500

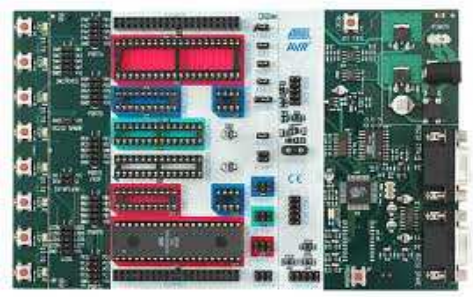


Figure 7.11, STK 500

To program our microcontroller we use STK500, figure 7.11. This is programmer and testing circuit board from Atmel Corporation. It has sockets for many microcontrollers from Atmel. It is useful tool for tests because it has led's and switches. Leeds can be lighted by setting 0 on ports and turned off by setting 1 on ports. Switches when pressed give a logical value 0. To connect STK500 with computer, RS232-USB converter is needed. We get one without drivers, but after researching we found universal drivers for every RS232-USB converter which can be downloaded for free from internet.

Our prototype needs connection with the computer to control it. It can be done using MAX232, and figure 7.12 shows connections.

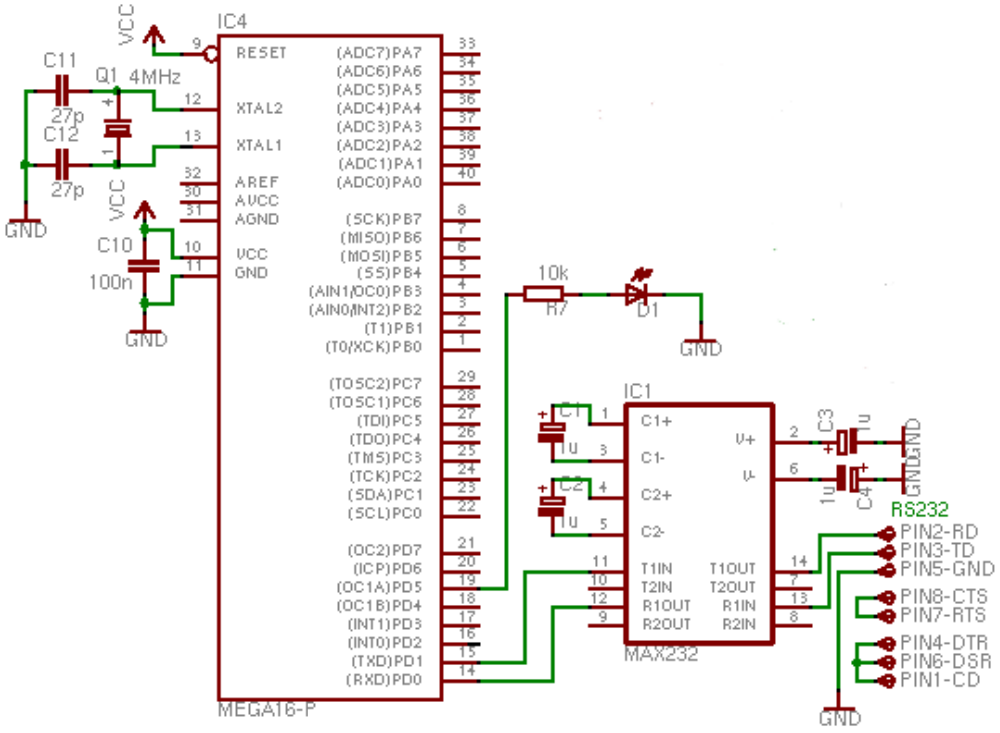


Figure 7.12, MAX232

Microcontroller need also stable 5 Volts voltage. It can be done by using 7805 chip, figure 7.13.

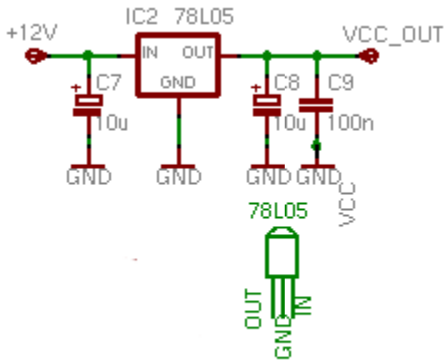


Figure 7.13, 7805 chip

8 Software design

8.1 High-level programming and software design

Project concerns microcontroller, display and other low-level devices. There is no place for high-level programming (e.g. object-oriented programming) and advanced design. However, having in our team specialist of high-level programming, helped us to discover, how useful can be a view from the different abstract view. Part called electronics and programming seems to concern low-level subjects – microcontrollers and so on. Electronic or communication experience can be vital here. We can consider the project in other perspective. Object-oriented programming won't help us in soldering etc., but it can be very useful in system design.

To describe my thought in the easiest way, let's say only a few sentences about one of designed diagrams – class diagram. It's very common in object-oriented programming, but there is no point in making it for structural or low-level programming. But the diagram was made only with one class. It looks silly, but let's think about it. We have there names of functions needed, attributes, which we will be using... Do you still think, it's totally unnecessary?

We think it won't be exaggerated, if we say, that computer system was already done in majority. We have done the diagrams, which represent all functionality of the system. Everything was planned. The application which allows us to simulate a flow of system was done. It's not so hard to transfer it into working system. Almost all the things that can be done in phase of software design were completed. In object-oriented programming the only thing to do now would be filling appropriate functions (in simulation application) with operations, to execute specific commands in separate places of program. To make it work, it 's important to translate high-level code into lower level or make some wrappers or libraries which can be more user-friendly than

low-level code – in this case you can utilize existing code and linking mentioned wrappers / libraries into it.

To be honest, having such design of system, it's much easier to make it even in low-level programming. Extended design phase protects from making errors. It's proved, that the more time was spent on designing, the less errors will be done later, the less money and time will be spent on finding and fixing mistakes and changing the concept. Whether the object-oriented design will be used like it is done, or will only be help in later work, I'm sure it is very essential for the project – even if some features or ideas will be changed.

We present only a few of many possibilities of design such system. It's still informal in some way – it must be to stay flexible. For example sometimes we are talking about turning off and turning on the device. It's used in some of possibilities. In final version we probably won't have option to turn off and on the device – we don't have any external buttons. It will turn off automatically after some time without using, or when because of low energy, and it will turn on again when under charging. But it's only the matter of detail – it doesn't change almost anything in effect.

The next things are differences in names and words used. Sometimes we have different names for the same things on different diagrams. It doesn't matter because it is still comprehensible.

8.2 UML

8.2.1 A few words about UML

Unified Modeling Language – Formal language, which is commonly used for modeling various systems. Very useful to describe and model a part of existing reality. The main area where UML is in common use is modeling computer systems. Usually when we are talking about UML we think about its graphical representation. Each element from this language corresponds to the symbol. These symbols are linked together on the diagram and it all makes a realistic model of complex system. We have many kinds of diagrams in UML (they are listed below). Using UML is almost inseparable step during developing of software (especially high-level, object-oriented programming and so on).

In these sentences We were trying to explain a basic reasons of using UML in our project. Usually it's one of the first steps in computer system life-time. Even if we won't finish whole project – it should be a great benefit for the next groups, to have completed one of the first phases of real programming and software design.

We would like to mention, that there are many ways of designing and developing a digital communication device for diving. We are only trying to show our vision, which in our opinion is a way worth to try. All the diagrams are appropriate for this vision mentioned, and were made for specific algorithms. If algorithms, rules and way of thinking about this product – will change, then our diagrams won't be ideally appropriate for other conceptions. However we believe that having those algorithms, diagrams etc. will be very useful even if almost every kind of changes will be done.

Only a few most interesting (for this solution) diagrams from listed below were designed.

Diagrams in UML 2.2

- Structure diagram
  - Class diagram
  - Component diagram
  - Object diagram
  - Profile diagram
  - Composite structure diagram
  - Deployment diagram
  - Package diagram
- Behavior diagram
  - Activity diagram
  - Use Case diagram
  - State Machine diagram
  - Interaction diagram
    - Sequence diagram
    - Communication diagram
    - Interaction overview diagram
    - Timing diagram

8.2.2 State Machine diagram

State Machine diagram (or state diagram) in figure 8.1 describes the behavior of systems. One of the requirements is to have finite number of states, and of course this condition is met in our system. The idea of state diagram is to show states in which the system can act, and variety of events that can occur in specified states.

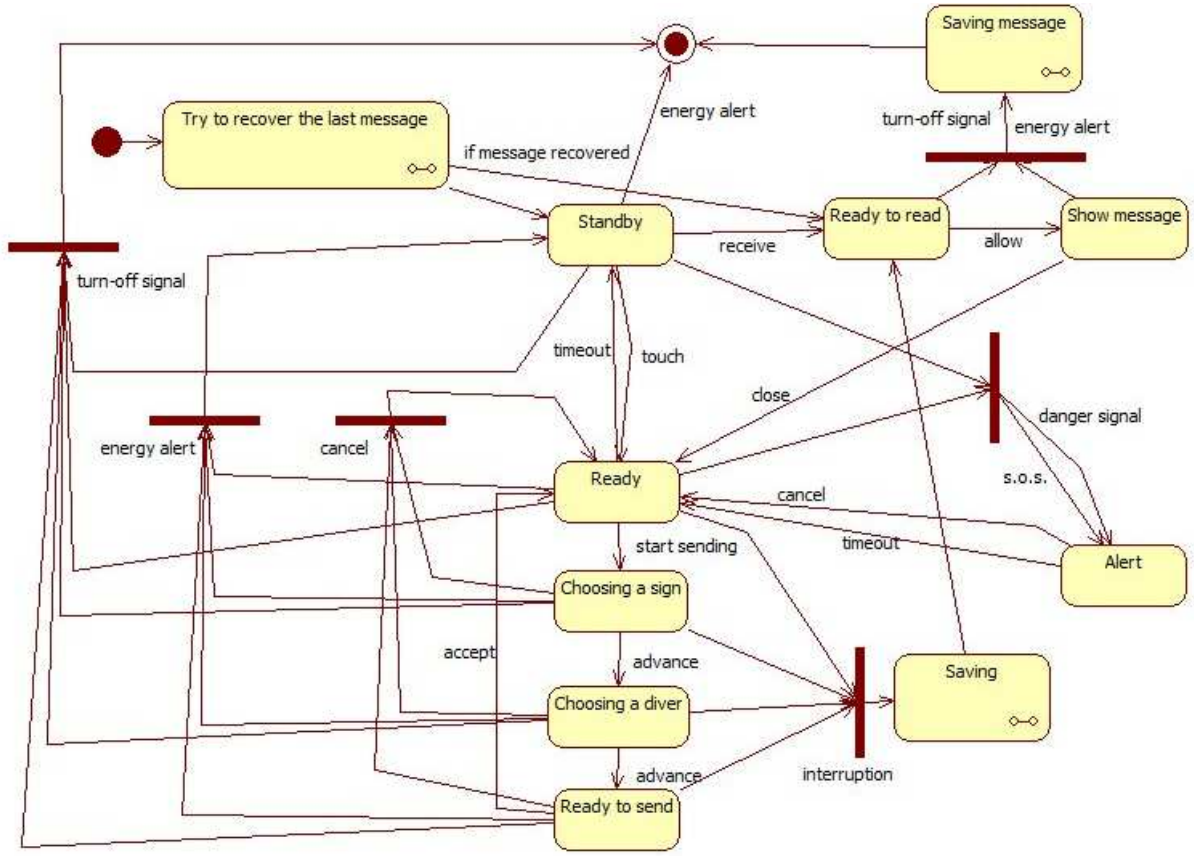


Figure 8.1, State diagram

LEGEND:

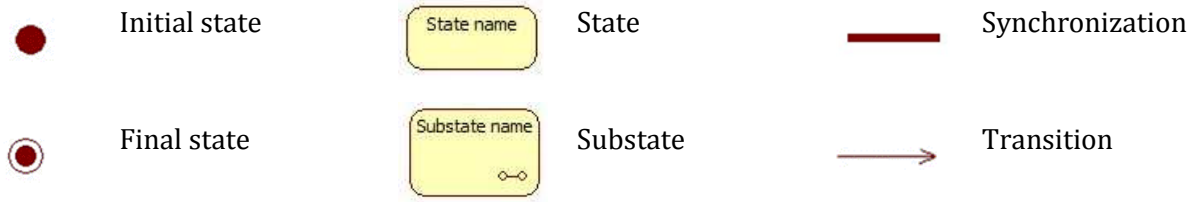


Figure 8.2, State diagram - legend

Initial state

- We are starting in initial state.

Try to recover the last message substate

- If there is any message saved (only if the message was received, but then a low-battery or turn-off signal occurred), according to action *if message recovered* – the next state is *Ready to read*.
- If we don't have any message saved – the first state after turning-on the device will be *standby* state.
- It's one of substates – or conditional states. It's not the state in which our system can wait for new actions – it's only the moment, when the condition is being checked. It's only to make decision if next step should be *Ready to read* or *Standby*.

Standby state

- It's the lowest energy consumption level. A good proposal is to only show dark display with two large buttons on the screen – SOS and danger. They are our very important signals – it must be easy to find them.
- When SOS or danger button clicked – it's changing to *alert* state

- When a *turn-off* signal will appear (when user will turn the device off) we are proceeding to *final* state.
- The same situation appears when we get *low energy alert*.
- When a message will be received, next state is *ready to read*.
- After touching the screen in this state, it goes to *ready* state - it becomes light (in standby state the screen can be dark – to limit energy consumption).

Ready to read state

- When it will be allowed we are going to read the message – it means showing the sign of message and sender information. Maybe this state is not needed – it depends of conception. In other way message can immediately appear on the display when received – but then it shouldn't use timer to hide message after fixed time – there is possibility, that diver won't even look at the message. As it was said it's only one of many different ways to plan the system.
- If the turn-off signal or energy alert will appear, reading a message is stopped, next state is *saving message* state.

Show message state

- After reading the message, it's time to move to ready state – with light screen and general information. Implementation of it can be done in many ways. For example we can use a timer, to close the message automatically after fixed time (e.g. 10 sec.) but then, it's not a good idea to remove previous state (ready to read state) with message box about message received. At least part of receiving message should be done in manual way – to protect of totally automatic processing – without any interaction we are not sure, if user saw the message or not.

- In case of receiving turn-off signal or low battery alert, proceed to *saving message* state

Saving message substate

- In overall saving message is unacceptable. In our system receiving has higher priority than sending. Even if we are in process of sending, message received will interrupt our sending. But there are some exceptions. When receiving a message, there is a possibility, that our device will be turning off (energy etc.). We must be prepared for such situation. Restoring the message which was sent, can be useful in critical situations – e.g. if it was the last message sent by a diver, who is missing. Because of such reasons – the message, which was sent, but wasn't read will be saved until the device will be turned on again.
- After saving device is turning-off – it was the only reason why to save the message.

Ready state

- The main state of our system. A central point where the system transfers after many actions.
- After fixed time (e.g. 30 sec.) of doing nothing, state changes into *standby* state. Timeout can be replaced – for example we can use a button with sleep function.
- From this state we can start sending a message. Clicking a button "send message" transfers us to *choose a sign* state – it's the first thing, what we need to do to send a message.
- It's one of 2 states, where we can click SOS or *danger* button. The reason why there are only 2 states with this function, is that we don't need to interrupt receiving a message, but in all cases (at other states) it's enough to click cancel button and get to *ready* state where we have such possibility. As was mentioned, it can be easily changed – e.g. providing SOS / *danger* functionality in all states.

- When interruption occurs, proceed to saving state. In this situation we won't save nothing, but *ready* state is part of sending, so from this sequence reaction is the same. In *saving* state we have conditional actions, so if get there from ready *state*, we will save nothing – it's done in easy and clear way.
- Energy alert leads us to standby state – we can have some limits – e.g. 10% of total battery charge. It's to less for normal work, but it's enough to stay in standby state. It gives us basic functionality for longer.
- Turn-off signal leads us to final state.

Choosing a sign state

- When achieving this state, system is restoring the information (if any) about last sign saved (*saving* substate) and making the sign lastly chosen a default one. It makes interruption during sending message less disturbing.
- As was partly mentioned, when interruption occurs, state is changing into *saving*.
- When a sign will be chosen we can proceed to next part of sending message – *choosing a diver* state.
- When the cancel button clicked, sign is not saved and the system is going back to *ready* state.
- When energy alert occurs, system is moving to *standby* state. The reasons why it's standby and not e.g. *final* state, were mentioned before.
- When turn-off signal appears, application goes to *final* state.

Choosing a diver state

- When achieving this state, system is trying to restore the information about last diver saved (*saving* substate) and making the diver lastly chosen a default one.

- When interruption occurs, state is changing into *saving*.
- When a diver will be chosen user can proceed to next part of sending message – the last state in sending process called *ready to send*.
- When the cancel button clicked, sign and diver are not saved and the system is going back to *ready* state.
- When energy alert occurs, system is moving to *standby* state.
- When turn-off signal appears, application goes to *final* state.

Ready to send state

- It's the last state in process of sending the message. We can accept and send or cancel our decisions from previous states.
- When user accepts decision, message is being sent, and system moves back to ready state. We are back now in central point of system.
- When interruption occurs, state is changing into *saving*.
- When the cancel button is clicked, nothing is saved and the system is going back to *ready* state.
- When energy alert occurs, system is moving to *standby* state, like in previous states.
- When turn-off signal appears, application goes to *final* state.

Saving substate

- When interruption appears during sending the message, system gets to this state. After saving choices, system makes that options default ones – save's that for next sending.
- Depends on a place, from which we get here, there are a few ways of saving.
- If previous state was *ready*, we are saving nothing.

- If previous state was *choosing a sign*, we can save a sign here. Of course it is not sure, that current sign was a proper one, for sending, but it's important, that user was moving in the direction of needed sign. Probably then it will bring forward finding the correct sign next time.
- If previous state was *choosing a diver*, we can save here a sign and diver. The reason was mentioned in previous point.
- If previous state was *ready to send*, the situation is analogous to previous one. The difference is that a diver was already chosen, so it's more possible, that the default diver will be the one, which user is looking for.
- When data were saved, we can finally proceed to *ready to read* state. It's obvious, that a message was the reason of moving to saving state, so going to read a message is the only thing to do now (in our system we assume, that receiving a message is the only kind of interruption).

Alert state

- System is in this state after choosing SOS or danger message. It means that message is being sent for all other divers without any confirmation and other options.
- Again it's matter of implementation – how to confirm or cancel choosing one of that 2 most important signals. One way is to wait for fixed time for confirmation (second touch on the appropriate part of the screen). For example if user won't click on the screen second time in 5 seconds, it means that first touch was only accidental – we have *cancel* action then. Next option is to do it without any timer. There are many different ways of planning actions for this state.
- When system is in this state, user can click cancel – it means, we are going back to *ready* state without sending any message.

Final state

- It's the state appearing when user is turning-off the device or when there is no more energy.

8.2.3 Use Case Diagram

Use case diagram is very common kind of UML diagrams. Inherent feature of UML diagrams is subjectivity. Maybe UML is a formal language, but it's graphic representation or rather interpretation of diagrams and signs is very symbolic. Use case diagram is one of the most ambiguous kinds of diagrams in interpretation and design. The idea of this one is to show all available cases of use of the system. Functionality provided by the system, actors, goals, actions, dependencies – all those parts should be taken into consideration during making this kind of diagram.

LEGEND:

Figure 8.3, Use Case diagram - legend

It's very hard to explain the differences between all kinds of dependencies and connections on this diagram. Usually it's only the intuitive action of designer. Sometimes it's easy to feel and recognize what the designer meant when doing the diagram. Very often it's not. It's almost impossible to make a diagram, which suits the opinions of all people judging the quality of it.

In this case, it won't be easy to explain the difference between kinds of dependency. We will try to do it, but even for us, the rules which we will talk about, are not 100% true in all cases. As was said, graphical representation of UML is very informal – we can talk about formal rules, but it's not enough to decide in all cases.

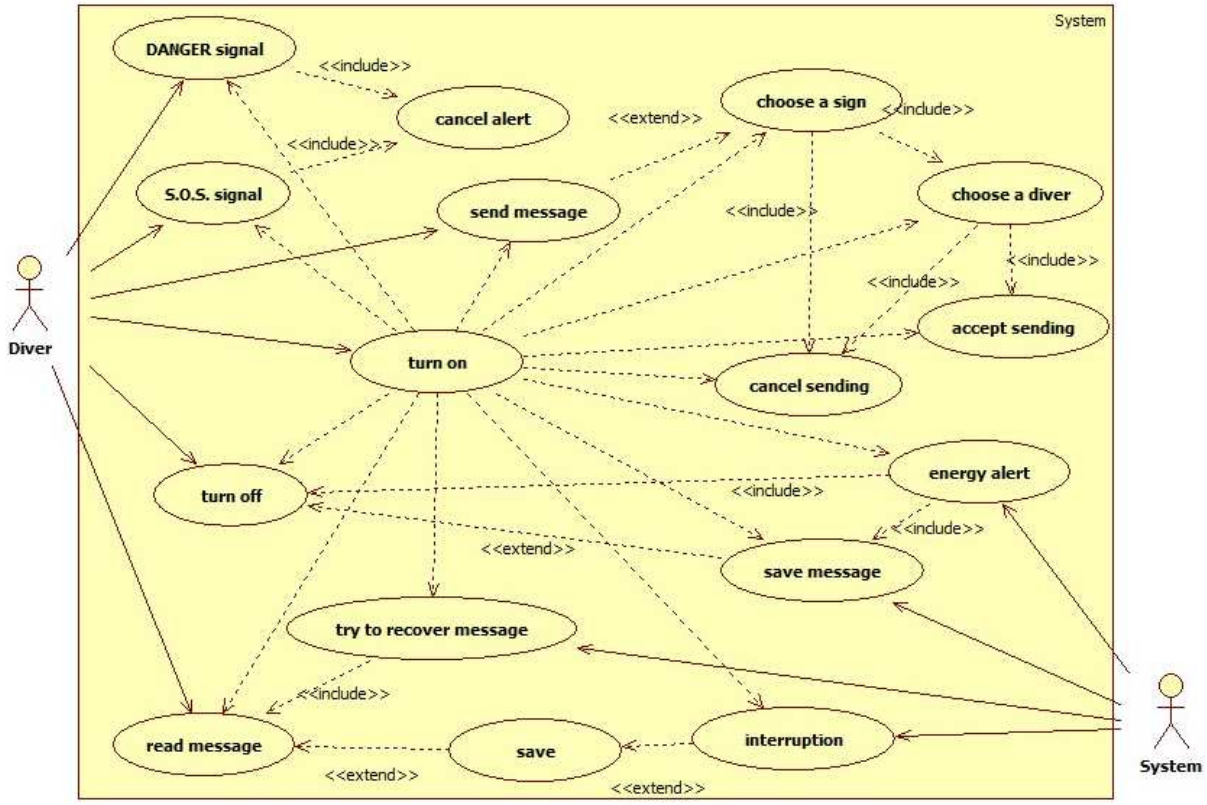


Figure 8.4, Use Case diagram

The most reasonable way of explanation for me is to say that directed association usually concerns actors. Dependency is a connection between reason and possible result – it's only one of many options. Include dependency means that something can occur but doesn't need to, but it's still stronger. Extend dependency means that moving to the state is the natural outcome of action.

In some cases it's better to not put all of the arrows or information on the diagram – it can be not necessary (e.g. something appears from transitiveness) and even obscures the diagram.

It's a good place to write a few words about this instance of use case diagram. We have 2 actors here. First is a user – diver. Second is a system – we have some actions, that are not triggered by user, so we need other actor (e.g. interruption is triggered by system).

We have many cases of use – a diver can e.g. read a message, but it demands turning on the system, before showing message (turning on is a diver action too).

Cancelling alert doesn't have any sense without choosing danger or SOS signal before it. So it's not the truly separate state. It includes choosing danger and SOS messages, but even if user chooses SOS signal, it's not sure, that he will cancel it. It's one of the examples of – how it was said–very hard to describe - include relation. Off course that you can find some definitions of types of dependencies, but they won't be always true.

We have some extend relations. E.g. after saving the options selected, we always have reading a message. The reason to save options is that a message came and we want to see it. It is always like this situation. In effect – reading message is extension of saving. But it can occur after recovering message too (without saving selections). In this case it's included relation – showing message doesn't need to appear after recovering message.

Few examples of notation and its meaning seems to be enough to understand what every part of this diagram is about. When UML diagrams are being used in reasonable way, they can be vital for project development from cradle to grave.

8.2.4 Class diagram

Class diagram is very important type of UML diagrams. It's one of most frequently used diagrams. It describes the structure of system – classes, their attributes, relationships (generalization, specialization etc.) packages, interfaces etc. It's one of the structure diagrams.

As it was written, it's dedicated for object-oriented modeling – as like all UML diagrams, but it's more object-oriented than all the other diagrams. So what is the reason to use it for our application? We can simply use only those features of this kind of diagram, which give us the benefits. For example we won't have more classes than one (because our system is not object oriented) but methods and attributes are described in this class – it can be useful.

This box describes class named Device. It's divided into 3 parts.

The first part is the class name. Let's say that in this case it can be the system name.

Figure 8.5, Class diagram

Second part is for attributes.

Third part shows methods (or functions) implemented in the class.

Before attributes and methods we have a sign + or –. It's the notation describing the visibility property. E.g. '+' means that visibility is public – all the other classes can see it. Next we have '–',

it means that something can be seen only in current class. There are two more signs with visibility meaning, but it's not necessary to explain it here ('#' - protected, ~- package visibility). It's hard to translate the meaning in this case. Better would be, if we agree, that if something is public, it corresponds the actions triggered by user. If something is private, it will be only used by system. So for example attributes (last sign chosen and so on) should be private, because system will deal with it in automatic way – we don't need to know about it. If there is time for loading last message – system will do it without our help. Function for choosing a sign must be public, because user triggers it – clicks the button and then the function starts.

8.3 Simulation Application

The reason of making such application was to have possibility to simulate flow of designed system. It's important to at first test only the algorithm and functionality, and after that, if our thinking and ideas are in a good way, then it's easier to start programming. It's important especially with low-level programming, when errors occurred during work with hardware, are very hard to find and fix. If the algorithm and design is done well, we will save much time in next phases of developing the system – we will have less problems.

The application was written to describe the idea presented on UML state diagram. Program was done in high-level and object-oriented programming language (c#). This application is the simulation but it can be a frame for the system too. It exactly shows how the system should work, so it's enough to fill appropriate functions with operations needed, and it will be done (in high-level way). Filling the schema and translating it into low-level is all that is needed to do with the system to make it work.

If we would be making a truly object-oriented system, we would do diagrams and application in other way. Here we were trying to find a compromise between the object-oriented programming (which is very convenient and effective method) and low-level programming (working with electronics and hardware). Those factors caused that in effect we get something between these 2 techniques mentioned. When it was reasonable we were making it more high-level, when not – more low-level. The aim here was to make as much benefits as we can – not to make perfectly object-oriented code or something like that.

For example the state diagram is done like for object-oriented system, because it has sense – it shows how the system should work. A class diagram is not like object-oriented – there is no point in making many classes in artificial way – in effect we won't have more than 1 class.

There are some differences in names of objects (or something other) between diagrams – for example between state diagram and simulate application. We decided that it's better way to show – e.g. on the diagram we need short names but in application it can be more descriptively. It is still clear and comprehensible.

One of the things that I'm still working with is the application. We want to add some different interfaces in each state to make the simulation more conveying the true sense of system's flow.

Code of application was enclosed in appendix.

9 Conclusion

During project we encountered some problems. For our display we needed an adaptor to connect display with the microcontroller. When we wanted to do the PCB it turned out later that resolution of printer is not enough to do proper PCB. So it would be better if we considered methods of connection between microcontroller and display before buying it. Also it is unique and brand new OLED so it was hard to find any examples of its usage in books and internet also. In some sources we read that better printouts could be done by using plastic transparent paper instead of normal transparent paper.

Communication between divers but from the point of view of the ATMEGA16 could be done by using the PWM or A/D converter. The easiest and recommended way is to use PWM. PWM is sending the square waves with different frequencies which mean 0 or 1. To receive this information using of USART is recommended. But it's very slowly way of communication. Prototype should consist of devices like ATMEGA16L, OLED display, voltage stabilizator for ATMEGA, voltage converter for display because it needs a negative voltage to operate an external crystal and communication part with sensors and amplifiers.

If we want to connect our prototype to PC we can use MAX232 serial converter for serial communication via RS232. For debugging of our code we had JTAG also from Atmel. Other interesting thing is the way of programming our OLED. We found two ways of doing this. First way is simple and hard to do because it takes a lot of time. This is the way of giving orders to each pixels alone using C language. The second is to use a graphical program which can generate a code in C language automatically. It is simple and is very similar to using Paint. We found many programs like this and the matter is to choose a proper one. These programs are for example:

- ProGFX graphic engine;
- FastLCD;
- Bitmap2lcd;
- Micro LCD;

Especially the first one could be proper to our task.

10 Reference

Marketing

- 2010. *Dykcenter.* [online] Available at: <http://www.dykmag.net> [Accessed 21 September 2010]
- 2010. *Great dive locations around the World.* [Online] Available at: < http://www.scubish.com> [Accessed 17 September 2010]
- Team 9, (2009) *Final report.* (21.3.1 - Porters 5 forces, page 114)
- Team 9, (2009) *Final report.* (Appendix - Page 74 of 80)
- Team 9, (2009) *Final report.* (21.4.3 - Pricing strategy)
- Team 9, (2009) *Final report.* (Appendix - Page 77 of 80)
- 2010, *Underwater Communication.*
- 2010. *Ocean Reef Alpha Under Water Cell Phone (UWCP).* [Online] Available at: <http://www.scuba.com/scuba-gear-168/Communication.html> [Accessed 3 October 2010]
- 2010. *Underwater Technological Center*. [Online] Available at: < http://www.utc-digital.com/ > [Accessed 1 October 2010]
- *Ocean Technology Systems.* [Online] Available at: < http://www.oceantechnologysystems.com/> [Accessed 1 October 2010]
- *Scuba market.* [Online] Available at: < http://www.scubamarket.com/index.php?/english/Home/vmchk.html> [Accessed 10 October 2010]
- 2010. [Online] Available at: < http://visual.merriam-webster.com> [Accessed 4 November 2010]

Interface

- *Scuba Diving Communications* . [Online] Available at: < http://www.scubadiver.cc/handsignals/uwcommunications.htm> [Accessed 8 October 2010]
- *Underwater signals.* [Online] Available at: <http://www.ukdivers.net/comms/signals.htm> [Accessed 8 October 2010]
- *Hand signals*. [videos online] Available at: <http://www.scubadiver.cc/handsignals/uwbasic.htm> [Accessed 18 November 2010]

Design

- 2010. [Online] Available at: < http://www.strapworks.com. > [Accessed 4 November 2010]
- Jakob Nielsen. 2010. *Ten usability heuristics*. [Online] Available at:< www.useit.com > [Accessed 2 November 2010]
- Steve Krug. 2nd. Berkeley: New Riders.

Electronics references

- [Online] Available at: <http://es.wikipedia.org/wiki/Diodo_org%C3%A1nico_de_emisi%C3%B3n_de_luz>
- [Online] Available at: <http://www.sparkfun.com/commerce/product_info.php?products_id=8625>

- 2010. [Online] Available at: <http://en.wikipedia.org/wiki/organic_light-emitting_diode>
- [Online] Available at: <http://es.rs-
>2010,online.com/web/search/searchbrowseaction.html?method=getproduct&r=6746759
#header >
- 2010. [Online] Available at: <http://es.farnell.com/densitron/p0430wqlc-t/display-am-oled-4-3/dp/1589476>
- 2010. [Online] Available at: <http://dk.mouser.com/productdetail/sharp-microelectronics/lq043t1dg01/?qs=va1bon3yxcrr16n8s2voua%3d%3d>
- 2010. [Online] Available at: <http://docs-europe.origin.electrocomponents.com/webdocs/0d6c/0900766b80d6c0d7.pdf
- 2010. [Online] Available at: <http://www.crystalfontz.com/product/CFAL12864Z-Y-B4.html>
- 2010. *OLED element.* Online] Available at: <http://en.wikipedia.org>
- 2010. *Bolymin bl043acrnb$*. [Online] Available at: <2010, ttp://es.rs-online.com>
- 2010. *Sharp lq043t1dg01.* [Online] Available at: <http://dk.mouser.com>
- 2010. *DisplaySheet.pdf* . Online] Available at: <http://docs-europe.origin.electrocomponents.com>
- 2010. Online] Available at: <http://en.wikipedia.org/wiki/Active-matrix_OLED>
- 2010. *Datasheet of the HX5116-A.* Online] Available at: <http://www.andilcd.de>

Others

- AVR stk500 user guide
- Documentation of Atmega16 microcontroller
- ANSI C programming language"- B.W. KERNIGHAN,D.M.RITCHIE
- 2007, Embedded C programming and the Atmel AVR, Barnett, R.h.