جامعة الأمير محمد بن فهد

PRINCE MOHAMMAD BIN FAHD UNIVERSITY

College of Engineering

Department of Electrical Engineering

Fall 2018-19

Senior Design Project Report

Environmental Monitoring Vehicle

In partial fulfillment of the requirements for the Degree of Bachelor of Science in Electrical Engineering

Team Members

	Student Name	Student ID
1	Hassan Rumaeh	201401856
2	Mahdi Al Nasir	201302938
3	Abdullah AL Buhumidah	201001504
4	Bakr AL Takroni	201002370

Project Advisors:

Advisor Name: Dr. Jawad Al Asad

Abstract

Nowadays, water and air pollution has been increased lately in Saudi Arabia, which has led or causes a harm on fishes and plants, also as the pollution increases the air quality and level of oxygen will decrease. Therefore, after looking up to the results we have been designed and implemented The Environmental Monitoring Vehicle. Basically, The Environmental Monitoring Vehicle it works as an amphibious, so, it will collect the readings from land and underwater, and upon the results that has been saved to SD card actions will be taken.

Table of *Contents*

1. Introduction

1.1 Project Definition

To design an environmental monitoring vehicle that will measure environmental variables inside and outside the water. Environmental monitoring vehicle will allow to drive on the land and float in the water. Figure 1.1 shows the final project:

Figure 1.1: Environmental Monitoring Vehicle

1.2 Project Objectives

1- Helps to monitor the pollution
2- Effective for environmental organization
3- To develop an amphibious vehicle for confined and narrow areas using suitable steering configuration and maneuvering system.

1.3 Project Specifications

1- Input voltage 14.7 V
2- Measures Temperature, Pressure and Oxygen
3- 1.5 km wireless connection
4- Operation time up to 15 Minutes

1.4 Product Architecture and Components

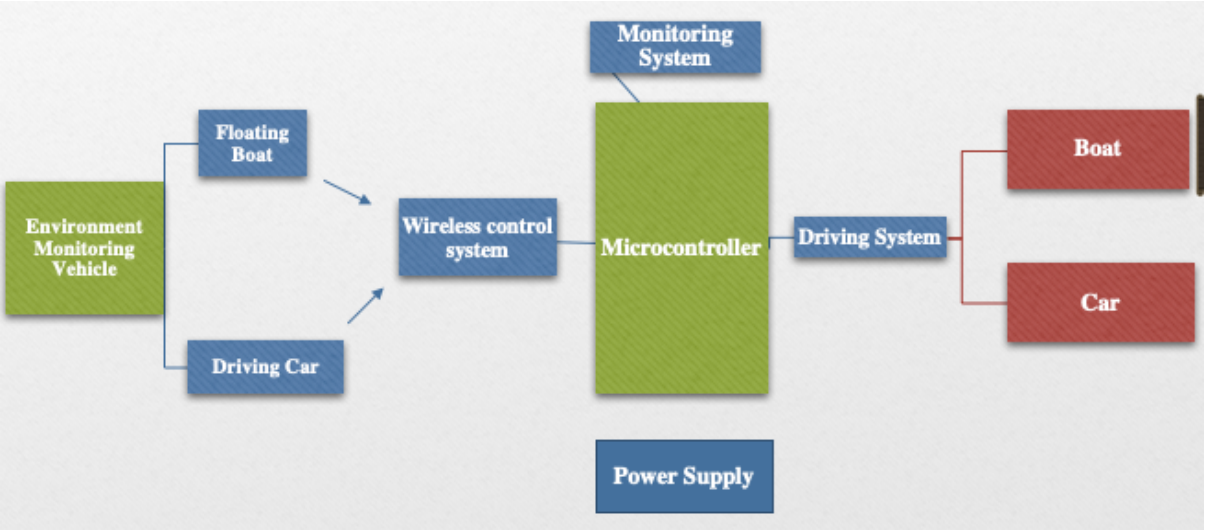


Figure 1.2 Product Architecture

Table 1.1: shows the total Budget for the project

Items	Quantity	Cost
Body	1	700 SR
Battery	3	600 SR
Arduino Uno Mega Kit	1	120 SR
Motors	3	470 SR
Xbee Comm.	2	196 SR
Controller	1	674 SR
Wheels	3	60 SR
Oxygen Sensor	1	600 SR
Temperature / pressure sensor	1	80 SR
Total	19	3500 SR

1.5 Applications

- **Educational purpose**

Environmental Monitoring Robot can be used as an educational instrument, because, one of the main objectives is to measure the pressure, temperature and oxygen underwater. So, students can get the measures data from the control station and analysis the data.

- **Underwater Maintenance**

Environmental Monitoring Robot could be very useful for underwater maintenance, because it gives a live video to the control station while it is underwater. Therefore, if there is any leaking or crack underwater the Environmental Monitoring Robot will give a live feedback to the control station.

- **Environmental Purpose**

Environmental Monitoring Robot can be useful and very effective on land also because, it also measures the pressure, temperature and level of oxygen outside the water.

Chapter 2. Literature Review

2.1 Project Background

Environmental Monitoring Vehicle is a robot that will improve the quality of the environment inside and outside the water. The measurement data will be divided in two parts. The first part is, monitoring the air quality outside the water. The air quality is very important nowadays, because, the manufactories will affect the people especially those people who are very sensitive and have health issues such as asthma. The second part is, monitoring the oxygen inside the water, also it will measure using the levels of temperature and pressure. Temperature consider as an important factor because, when the temperatures decrease that will affect the growth and reproduction of living organisms. As a result, animals and plants grow faster at warmer temperatures. Also, pressure is important as well because, if the level of pressure decrease inside the water that will reduce water flow to a trickle and it will take a long time to fill a kettle or a cistern. Finally, oxygen is very important because, the oxygen will have an impact on the plants and animals that live inside the water.

2.2 Previous Work

Project 1: Remote Controlled Submarine, *Prince Muhammed Bin Fahad University Spring* 2016 [1]

Project definition: To design and build a submarine that will gather data underwater and send it to an on-shore control station.

Objectives:

- Send live feedback to detect defects on surfaces underwater
- Can measure temperature underwater
- Can measure depth from pressure underwater

Project 2: Robots to Study Lake Tahoe, *University of Nevada July, 2016* [2]

Project definition:

To design a platform that can be used on the surface of a lake to quantify the water quality changes in the nearshore environment.

Objectives:

- Aimed to monitor the environment for water quality (turbidity and Oxygen)

Project 3: Autonomous Targeting Vehicle (ATV), *Purdue University, Spring* 2011 [3]

Project definition:

To designated GPS waypoints and to visually locate and follow targets.

Objectives:

- Sensor data to accurately determine location of the robot
- It can be used autonomous or controlled wirelessly

2.3 Comparative Study

Below table shows a comparison between the previous projects and our proposed approach

Table 2.1: A comparison between the previous projects and our project [1] [2] [3]

Projects	**Remote Controlled Submarine**	**Robots to Study Lake Tahoe**	**Autonomous Targeting Vehicle**	**Our Project**
Functionality/serviced	Underwater	Floating on water	On land	Amphibious
Communication	Wired	Wirelessly	Autonomous	Wirelessly
Live Stream	Yes	No	Yes	Yes
Depth	5 m	10 m	-	5m
Operating Time	15 Min.	15 Min.	Not Mentioned	Minimum 15 min.

Chapter 3. System Design

3.1 Design Constraints

Our project is designed to be on the water in order to take some environment variable, and it takes an action to save the environment. On the other hand, the driving car is the second subsystem، and it is designed to measure the environment variable on the land، especially، in our area we have several climatic factors and that need to be monitored.

Table 3.1: Design Constraints

Type	The problem	Our solution
Economic Constraints	High price in Waterproof Components	No alternatives
Waterproofing the components	Not available in local market	Order online
Body Design	How to modify the boat to Amphibious Vehicles	By design an aluminum beam that fits motor
Safety	Electronic, components, power supply, and cable	Designing a space that fit with components

3.1.1 Design Constraints: Economic

Our project is characterized by its high quality and reasonable cost. The system has been designed suitable for the project specifications with lower cost. All the components that have been used are high efficient, reliable, and have good quality. The power components have been chosen from the best brands with high quality, good performance, and reasonable costs. The components have been used in this project are not consumed much power, so they can serve as long as possible.

3.1.2 Design Constraints: Waterproofing the Components

One of major tasks that we have been considering a lot, which is, preventing water leakage into the electronic components. However, after looking up and searching we have found a waterproof component such as motors and ESC (Electronic Speed Control) that can go inside the water without damaging them.

3.1.3 Design Constraints: Safety

Using high current power sources can be dangerous especially on the water. Any leakage could spoil the battery and could contaminate the water. Since we have two motors are attached to an aluminum beam, and the beam is attached to the vehicle body, we had two holes or slots in each side for the beam, so we need to filled these two gaps with silicon in order to avoid water leakage.

3.1.4 Design Constraints: Body Design

The design of the project has to be light and hard, so the design of our body has good construction. It’s made of hard plastic and light weight. The hard plastic can decrease the percentage of the damages if the body crash with an object and the light weight helps the body to be stable on the water and easy movement.

3.2 Design Methodology

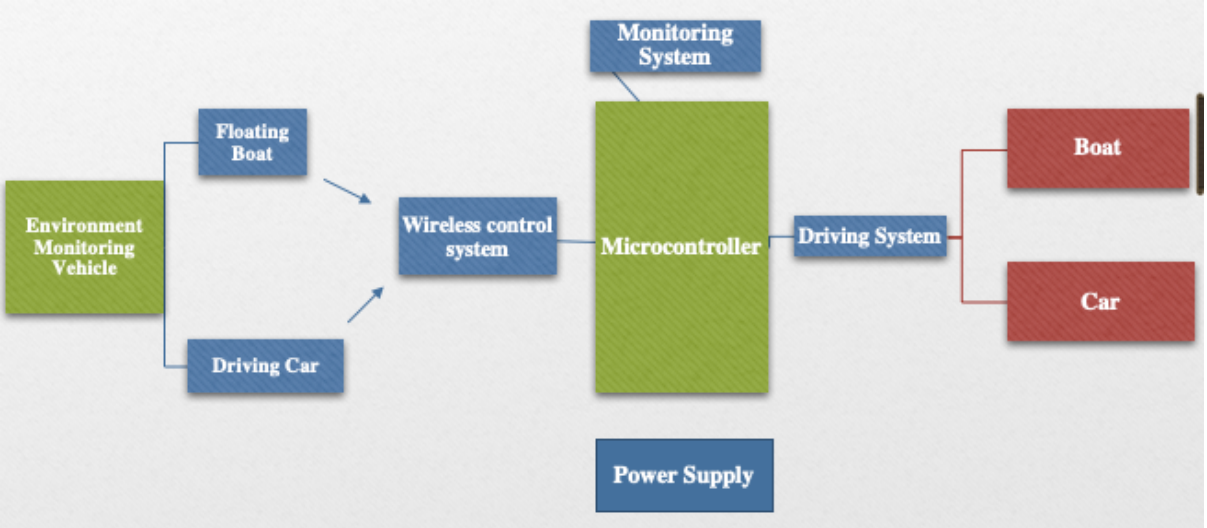


Figure 3.1 Project Architecture

As shown in the figure; we have four subsystems in our design methodology.

- Subsystem 1: Floating Boat
- Subsystem 2: Driving Car
- Subsystem 3: Wireless Control System
- Subsystem 4: Mentoring System

Environmental monitoring vehicle has been designed according to the specification mentioned earlier. The environmental monitoring vehicle will be controlled by a wireless controller. The controller is programmed using Arduino; we send a signal by the controller and we receive the signal by the Xbee receiver, that is placed inside the boat.
However, we used two Xbees' and we configured them, one as transmitter and the other as receiver. The motors speed will be controlled by the waterproof ESC. On the other hand, we will gather the natural variables using highly selected sensors and the data will be saved in an SD card.
The phases below determine the objectives of this project:

Senior Design 1:

- Read and learn about previous amphibious robot, system components and implementation challenges.
- Discover available resources at PMU (components, instruments, Labs, software) and relevant to projects.
- Identify all functional subsystems required in the project.
- Identify various alternatives to implement each subsystem and select appropriate ones.
- Select components required for each subsystem.
- Study about the controller system and implementation.
- Download full version of Arduino in our computer.
- Understand Arduino in detail.
- Order the required components (Motors, sensors, speed control, servo and battery)

Senior Design 2:

- Design each subsystem and build it.
- Test subsystems, analyze results and make necessary improvements
- Use project room and/or Lab equipment for testing and support from Lab Engineers.
- Integrate all subsystems together and test
- Program Arduino.

3.3 Product Subsystems and Components

In the third part, we will discuss the subsystems of our projects and how we chose our components according to our needs.

3.3.1 Subsystem 1: Floating Boat

Subsystem 1 which is Floating boat, it has a Brushless DC motor with high efficiency. It will be controlled by a wireless controller using XBee, and it has input volt 14.7V. This system is going to be on the water and monitor some environment variables (temperature, oxygen, and pressure) and the data will be saves in an SD card.

According to the specification in Table 3.2, we have selected the brushless for the subsystem one which Floating Boat

Table 3.2: Comparison between Motors

	Brushless	Stepper motor
Efficiency	High Efficiency	Low Efficiency
RPM/V	3180 RPM	1325 RPM
Water Resist	Yes	No
Cost	45$	30 $

Figure 3.2.1: Brushless Motor

Figure 3.2.2: Stepper Motor

3.3.2 Subsystem 2: Driving Car

Subsystem 2 which is driving car, we redesign the boat in order to make it as an amphibious. The driving car has two brushed motors attached directly to the wheels. Of course, driving car is going to be on the land and it is controlled by wireless controller. The function of this system is to measure the environment variables, such as the level of oxygen, temperature, and pressure, the data will be saved in an SD card.
According to the specification in Table 3.3 we have selected the Brushed motor for the subsystem two which is Driving Car.

Table 3.3: Comparison between Motors for Subsystem Two Driving Car

	Brushed motor	**Brushed motor**
Performance	Low	High
RPM/V	4910 RPM	4600 RPM
Cost	32 $	29$
Water Resistance	No	Yes

Figure 3.3.1: Brushless motor

Figure 3.3.2: Brushed motor

According to the specification in Table 3.4 we have selected the Multistar High Capacity 4SBattery for both subsystems

Table 3.4: Comparison between Battery for both subsystems (Floating Boat) & (Driving Car).

	Tenergy LiPO	**Multistar High Capacity 4S**
Voltage	11.1 V	7.4 V
Capacity	12000 mAh	6200 mAh
Discharge	10C	10C
Cost	24$	37$

3.3.3 Subsystem 3: Wireless Communication System

This system is made to control the whole subsystems of the project by using controller with Xbee module. Every push button in the controller has its function to send the order to the receiver and that will be via the Xbee. Xbee has good range which is 1.5 km that allow to the user to control the environmental monitoring vehicle from good distance.
In Table 3.5, we choose the X-bee, because, it has a higher range than the Bluetooth module.

Table 3.5: Comparison between wireless modules for the system

	Bluetooth	**X-bee**
Max Range	10 meters	1500 meters
Input voltage	3.7 V	2.8-3.4 V
Cost	8 $	26$

Figure 3.4.1: Xbee Module

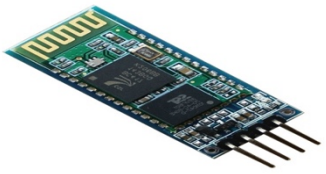


Figure 3.4.2: Bluetooth Module

3.3.4 Subsystem 4: Mentoring System

Of course, our project is environmental monitoring vehicle, which means that this subsystem is considered as the heart of our project. We considered confined areas that have pollution, in order to reduce the pollution and save the environmental, we have decided to make the monitoring system to measure oxygen, temperature, and pressure inside and outside the water. In addition, the collected readings of the will be saved in an SD card.
Below tables shows the difference between sensors, and according to the specification we have chosen the labeled one. Regarding the Oxygen sensor in table 3.7, we borrowed it from PMU lab.

Table 3.6: Comparison between pressure and temperature sensors

	Pressure sensor	**Temperature sensor**	**Pressure/Temperature sensor**
Voltage	3.3-5 v	3.3-5V	3.3-5V
Accuracy	+/-0.25%	+/-0.5%	+/-0.25%
Cost	25 SR	50 SR	20 SR

Table 3.7: Comparison between oxygen sensors

	Dissolved Oxygen sensor	**Atlas Dissolved Oxygen Sensor**
Voltage	3.3-5 v	3.3-5V
Response	+/-0.25%	98% full response, within 90 seconds (25°C)
Cost	806 SR	625 SR

3.4 Implementation

3.4.1 Implementing Subsystem 1 (Floating Boat)

Before the implementation phase we have considered all the alternative components and we have chosen the most reliable and efficient material and components. In the implementation phase, we did many configurations before we built all the things together to make sure that every part will work as we expected.

However, for implementing subsystem one which is floating boat, we have considered three main characteristics which are, material of the body, the weight and the shape. Firstly, the body supposed to be designable in order to manipulate it to be an amphibious. Secondly, the weight the material must be light in order to float without any obstruction. Finally, the shape of the body should look like a boat and should has enough space to fit the components. Figure 3.5 and Figure 3.6 shows the body structure along with float boat motor.

Figure 3.5: The structural of the boat

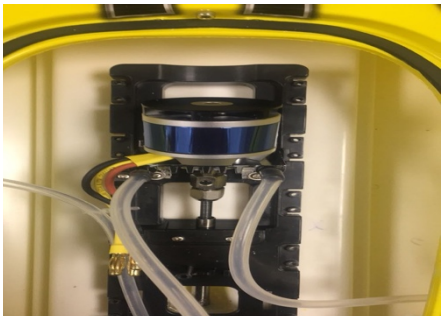


Figure 3.6: Motor for the floating boat.

3.4.2 Implementing subsystem 2 (Driving Car).

The objective of this subsystem is to make the structure that we picked in Subsystem 1 (Floating Boat) capable of driving offshore, and the main things to make this happen are the Motors with wheels and how to install them in the boat. So, we have chosen two Brushed motors for the Subsystem 2 (Driving car), and we have designed one supporting wheel (without motor) to be attached in the front of the boat. Also, we designed an aluminum beam that the Brushed motors can fit in. Figure 3.7 shows the Brushed motor we picked for the driving system. Figure 3.8 shows the fabrication of the aluminum beam we designed for the motors, as a result the aluminum beam ready to be attached in the vehicle. Figure 3.9 shows the supporting wheel and the aluminum beam attached to the vehicle.

Figure 3.7: The Brushed motor for the driving car

Figure 3.8: The Aluminum Beam

Figure 3.9: The supporting wheel for the front of the vehicle.

The idea of the aluminum beam is to be attached at the bottom of the boat to hold the driving motors, and the shape we chose for the boat is flatty from the bottom which helped us a lot in attaching the beam. We started with fabricating the aluminum beam and making holes in it that matches the holes we did in the bottom of the boat. Figure 3.10 shows team members fabricating the aluminum beam. Figure 3.11, shows after completing the task and insert the brushed motors inside the aluminum beam. Figure 3.12 shows team member attaching the supporting wheel and the beam on the boat, also Figure 3.13 show after successfully completing the task.

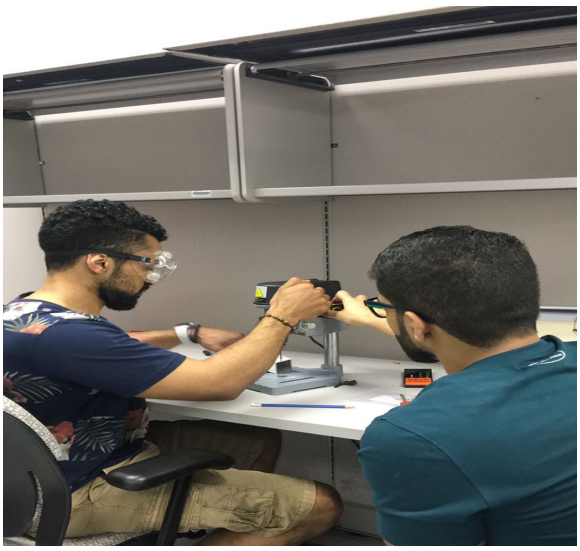


Figure 3.10: Fabricating aluminum beam

Figure 3.11: After completing the task

Figure 3.12: Attaching the supporting wheel and the beam

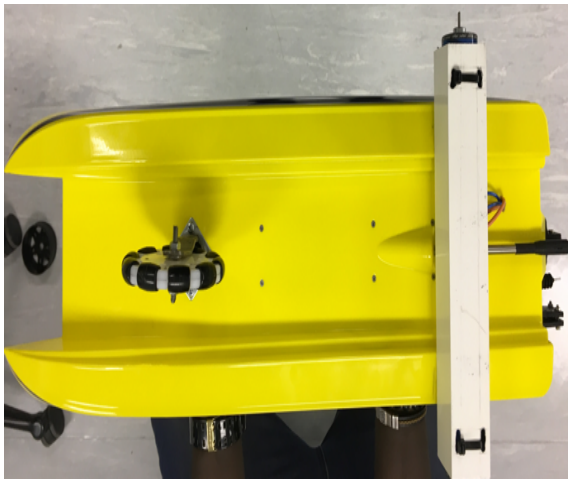


Figure 3.13: After completing the task

After successfully implementing the subsystem 2 (Driving Car) and integrate it with first subsystem (Floating Boat). See Figure 3.14

Figure 3.14 After Subsystem 1 & Subsystem 2

3.4.3 Implementing subsystem 3 (Wireless Control System)

The objective of this subsystem is to control the first subsystem (floating boat) and the second Subsystem (Driving car). Figure 3.15, X-bee module.
To create a network with XBee modules, the user must know XBee roles, which are:

- Coordinator: required in every network In charge of setting up the network. Can never sleep.
- Router: multiple may exist. Can relay signals from other routers/end points, can never sleep.

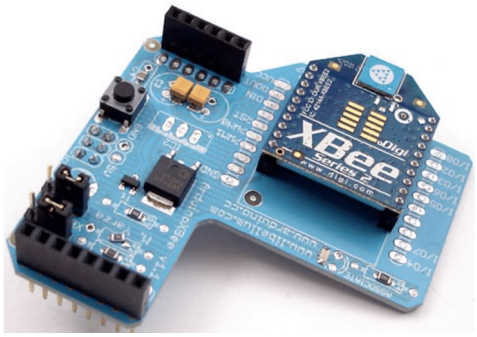


Figure 3.15 Xbee Module

Therefore, every network must have a coordinator, and may have router or end points. When one is working with XBee, they may need to update or change the firmware on the radios occasionally. Based on the type of communication needed, appropriate firmware should be uploaded to the radio modules. To do so, X-CTU program is needed. Below, in Figure 3.16, is the front page of the XCTU software. The steps required to configure the XBee units are shown in Figure 3.17 and Figure 3.18.

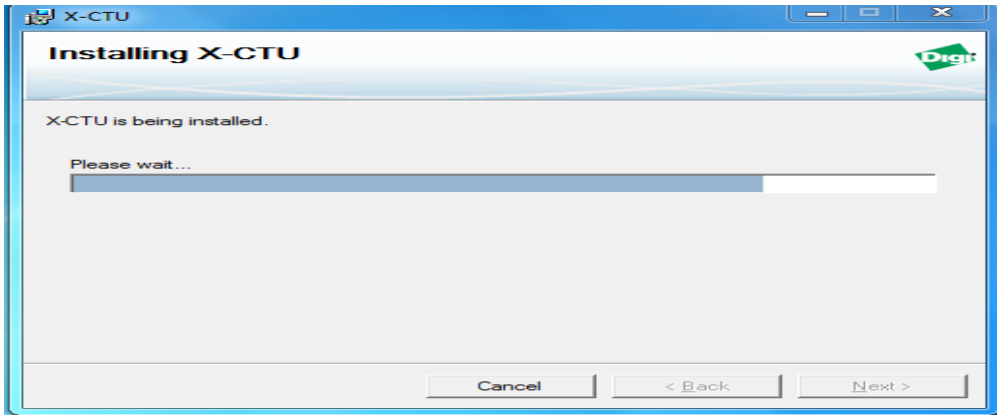


Figure 3.16: XCTU Software Install.

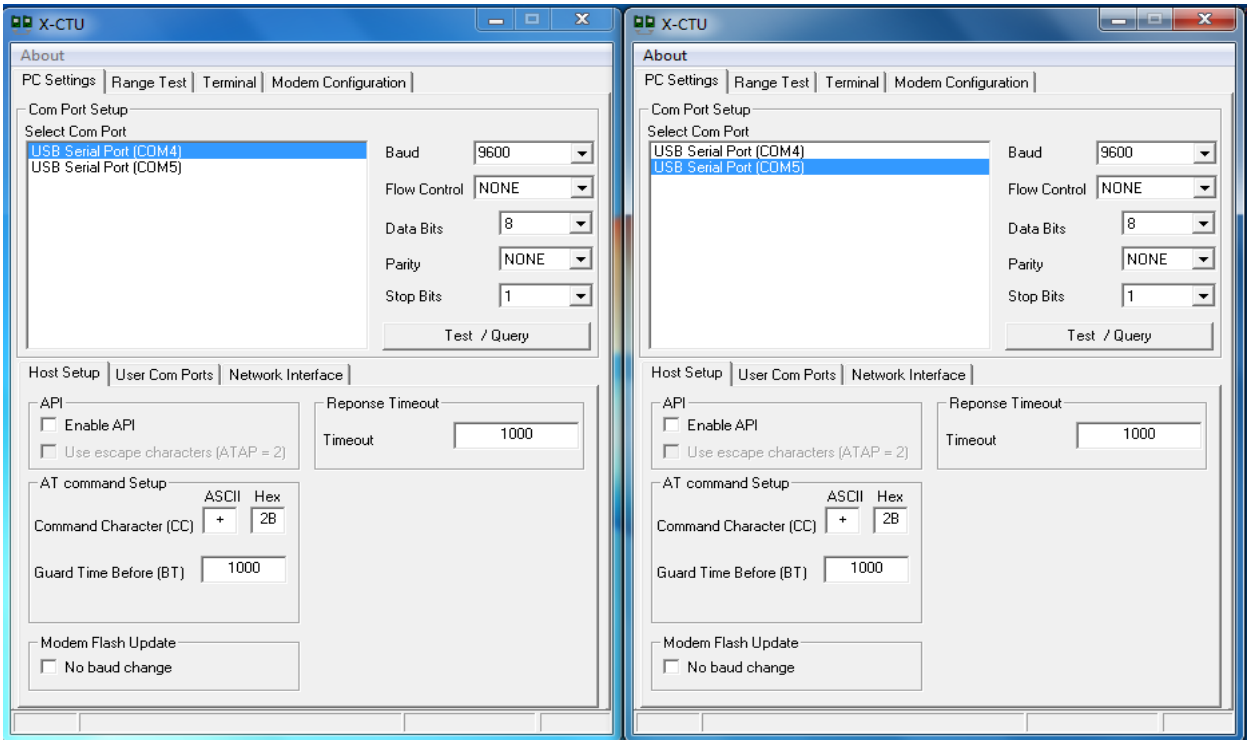


Figure 3.17: XCTU Configuration Steps.

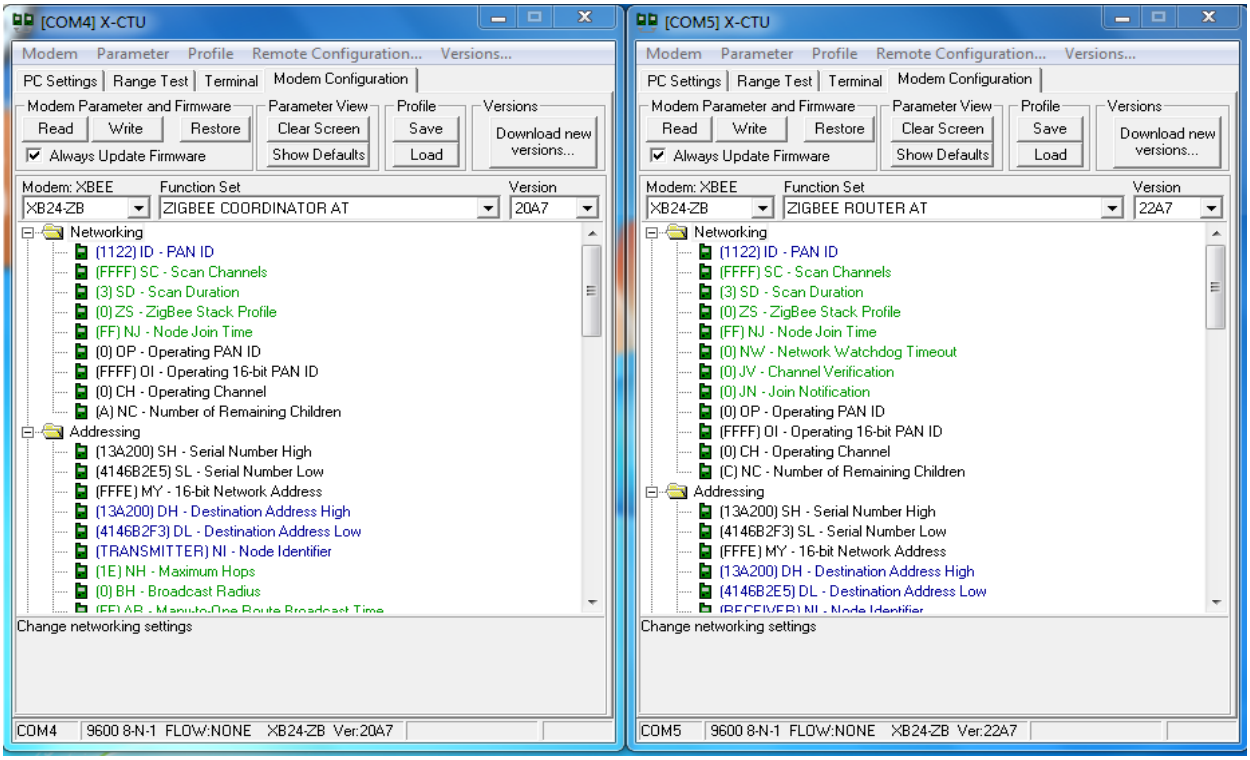


Figure 3.18: XCTU Configuration Steps.

After successfully configuring two Xbee's one as receiver Figure 3.19 and the other one as transmitter Figure 3.20. However, after that we develop an Arduino code to transmit the signal from the Controller to the receiver to give an order to the motors. And we set the buttons accordingly as shown in figure 3.21.

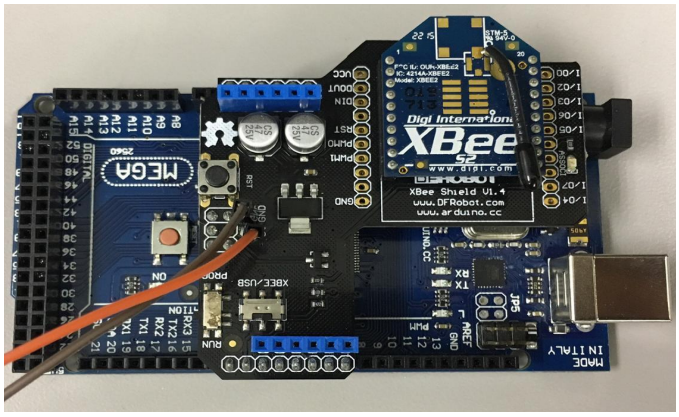


Figure 3.19: Xbee Receiver

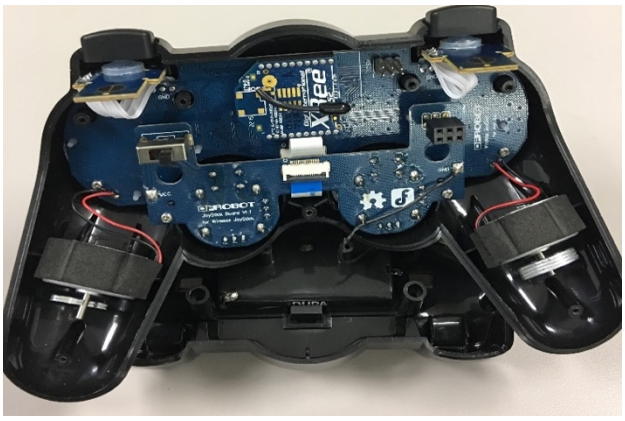


Figure 3.20: Xbee Transmitter

Figure 3.21: Controller Digital and analog with Gestures

3.4.4 Implementing subsystem 4 (Monitoring System)

The last subsystem which is, monitoring system that come with a critical question; where is the best place to install the sensors, we have thought about it many times, and finally we agreed to place the sensors outside the boat in order to get accurate measurements and save place inside the boat for other components. Figure 3.22, shows red antenna and black rope, the red antenna for the pressure and temperature sensors, while the black rope is oxygen sensor. The main idea here is, while the vehicle is on the land the red antenna will be vertical as shown in figure 3.22, but if the vehicle is in the water the red antaean will rotate 180 degrees as shown in figure 3.23 to get the measurements or readings from the water.

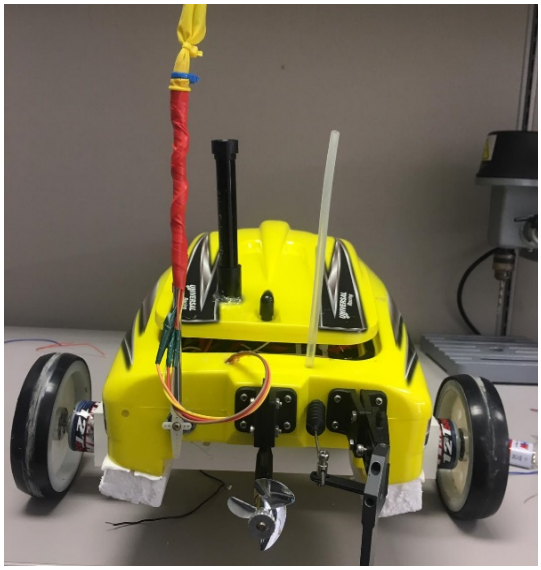


Figure 3.22: Shows T, P&O, while on the land.

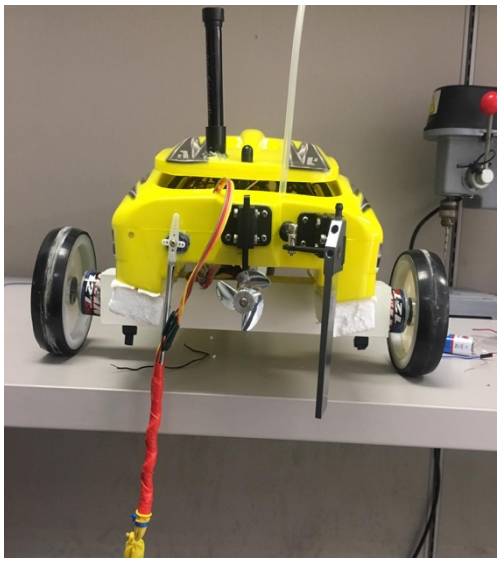


Figure 3.23: Shows T, P&O, while in water.

Chapter 4. System Testing and Analysis

4.1 Subsystem 1: The Floating Boat

The objective of this subsystem is to get the proper motors, shape and material for the vehicle to be able to float on the water smoothly. We picked the Brushless Motor because of the high efficiency and also brushless' rpm is good enough for our vehicle's movement on the water. We have chosen the body, so that the body will be fabricated for other subsystems' set ups, and we put the weight of the structural in our consideration. We achieved greater performance on the water by the reinforced plastic because the plastic is lighter than the other metals and it is easy to fabricate, also, plastic is way cheaper. Figure 4.1 shows our Brushless motor for the subsystem1. Our battery supplies a voltage of 14.7 V and the motor minimum current approximately 2.7A and the other power distribution will be discussed in the Subsystem 2.

Brushless Motor Specifications:

- It operates from a voltage of 7V to 18V
- It requires a minimum current of 2.7A.
- Max current rate is 94A
- Its diameter: 36mm
- Its length 60mm
- Shaft diameter is 5mm
- Its weight is 280g**.**

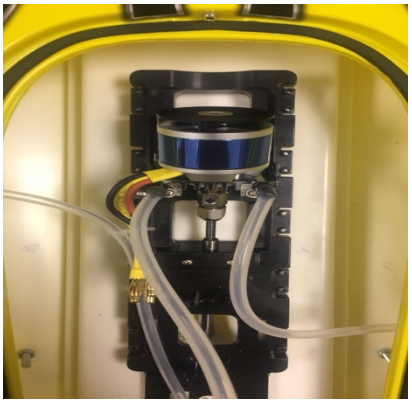


Figure4.1: Brushed Motor

We have run through many tests in order to fix and improve our first system which is (Floating Boat). Figure 4.2, shows a one of the tests we have done it is very clear that the vehicle float on the water successfully, which means that we have selected the right material and components for the first subsystem.

Figure. 4.2: First subsystem (Floating Boat)

4.2 Subsystem 2: The Driving Car.

For this subsystem which is (Driving Car), we have chosen a high torque waterproof brushed motor. As discussed in chapter 3, we have redesign the boat in order to install the aluminum beam that fit brushed motors.

Brushed Motor Specifications

- It operates from a voltage of 7V to 14V
- Minimum current required 2.9A
- It efficiency 85%.
- waterproof
- It has rpm 4600
- Its diameter: 36mm
- Its length 60mm
- Shaft diameter is 5mm
- Its weight is 280g**.**

Figure 4.3: Brushed Motor Attached to the wheel

Figure 4.4 and Figure 4.5, clearly shows the environmental monitoring vehicle while testing the second subsystem which is (Driving Car). While testing the second subsystem, we observed that the aluminum beam needs to be adjusted, and we also needed to decrease the speed of the motors.

Figure 4.4: The Monitoring vehicle

Figure 4.5: The Monitoring vehicle

4.3 Subsystem 3: Wireless Control System

As discussed in previous chapters, after comparing between alternatives we found the most suitable communication module is Zigbee (Xbee S2).

The specifications for the Xbee:

- It has input range around 400 meters and output range 1500 meter
- The operating voltage: 2.8-3.4 V
- Weight: 3 g
- Pulse width modulator (PWM): 2 outputs

Figure 4.6: Xbee Module

After integrating both subsystems (Floating Boat) and (Driving Car), we needed to control them. Therefore, after configure the Xbees' modules as Transmitter and Receiver, and developed an Arduino code (see Appendix D), we tested them on the on both systems which are (Floating Boat) and (Driving Car). As a result, we observed that they responded to the sent signal from the controller to the receiver. Figure 4.7 clearly shows the received data from the controller using Arduino serial monitor. The data shown in the serial monitor it describes the speed value for brushed motor, it is very clear as we press a button in the controller the values are increasing by ten, and when we press the other button the values are decreasing. On the other hand, in Figure 4.8 it is the same case for the brushless motor, when UP button is pressed the Brushless value is increased, and when DOWN button is pressed the Brushless value decreased.

Figure 4.7: Values for Brushed Motor

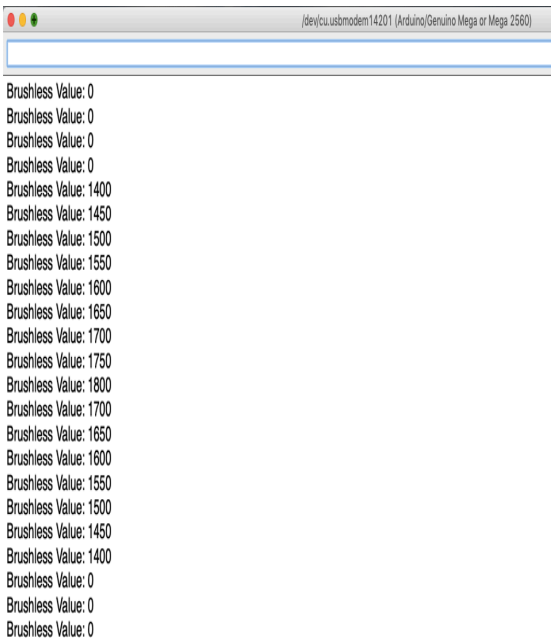


Figure 4.8: Values for Brushless Motor

4.4 Subsystem 4: Mentoring System

The monitoring system consider as the heart of our project, because mainly our project depends on it. After looking up through sensors we selected the best that satisfy our needs, the sensors are Oxygen, Temperature, and Pressure sensor we placed them at the top roof of the boat, in order to get accurate results. After placing the sensors, the sensors will take the measured data and save them in SD card automatically after pressing a command in the controller. Moreover, we can adjust the time period for taking the data such as in every five minutes and that will have a good advantage to make the readings accurate.

The specifications for the SD Card

- VCC: 3.3-5V
- Size:20x28mm
- Interface: SPI
- Compatible: MicroSD

Figure 4.9 SD Card Arduino

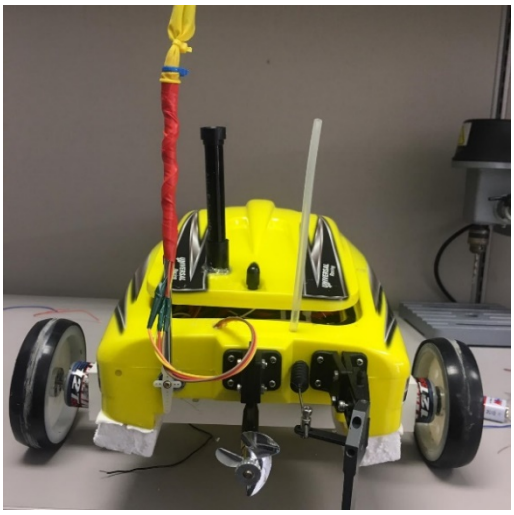


Figure4.10: Sensors After installation

Figure 4.10, shows the placing of the sensors the red antenna is the pressure and temperature sensor while the black one is for the oxygen. However, after placing the sensors we have tested and we have successfully completed the task, as a result figure 4.11 shows the stored data from SD card after plugging the SD in the computer.

DATA.TXT

Pressure: 0.00 hPa	Temperature: 0.00 C	O2: - mgl
Pressure: 0.00 hPa	Temperature: 0.00 C	O2: - mgl
Pressure: 0.00 hPa	Temperature: 0.00 C	O2: - mgl
Pressure: 0.00 hPa	Temperature: 0.00 C	O2: - mgl
Pressure: 506.38 hPa	Temperature: 24.27 C	O2: 1.40 mgl
Pressure: 506.27 hPa	Temperature: 24.23 C	O2: 1.40 mgl
Pressure: 506.20 hPa	Temperature: 24.22 C	O2: 1.45 mgl
Pressure: 569.36 hPa	Temperature: 73.17 C	O2: 1.55 mgl
Pressure: 481.05 hPa	Temperature: 20.40 C	O2: 1.40 mgl
Pressure: 481.17 hPa	Temperature: 20.40 C	O2: 1.40 mgl
Pressure: 481.15 hPa	Temperature: 20.42 C	O2: 1.45 mgl
Pressure: 481.28 hPa	Temperature: 20.41 C	O2: 1.45 mgl
Pressure: 481.36 hPa	Temperature: 20.44 C	O2: 1.40 mgl
Pressure: 481.13 hPa	Temperature: 20.44 C	O2: 1.55 mgl
Pressure: 482.44 hPa	Temperature: 20.40 C	O2: 1.40 mgl
Pressure: 0.00 hPa	Temperature: 0.00 C	O2: - mgl
Pressure: 0.00 hPa	Temperature: 0.00 C	O2: - mgl

Figure 4.11: Stored Data from SD Card

4.3 Overall Results, Analysis & Discussion

4.3.1 Float Boat

Preparation:
To test the brushless, we tried with a special code that specify the speed of the motor. We charged the battery fully to have a better result, then we observed the Max an Min for the motors.

Test results:
Basically, when we start the test, we were impressed by the speed of the motor. Then we tried to decrease the speed using the Arduino code.

4.3.2 Driving Car

Preparation:
After we design a beam and chose the brushed motor, we connect the brushed motors into the Arduino joining with the electronic speed control to observe the Max and Min RPM. By the way we decided that the floating boat and driving car have separated batteries to reduce power consumption and the let the vehicle run much longer time.
Test:
First of all, when we test the car in land we found that there is some lagging of the wiring such as loss the connection between the wires so we soldered them. Finally, we were able to control it.

4.3.3 Wireless Communication System

Preparation:
After configuring the two Xbees' we develop an Arduino code for both transmitter and Receiver
Test:
After uploading the codes, we tested if there is any data dent to the receiver, we found the data delivered, but after a delay which causes a problem. Therefore, we had to redevelop the code to come over the issue.

4.3.4 Mentoring System

Preparation
We made sure that we have selected the right sensors that will achieve its job correctly. Also, before installing the sensors on the vehicle we run a quick test on the sensors in order to see how accurate the readings are, see figure 4.12 it shows the collected data through Arduino serial monitor.
Test:
As a result, after running the test as shown in figure 4.13 it shows the implementation of the circuit, and figure 4.12 shows the collected data. After the observation of the data and how accurate they are, we proceed to install them on the vehicle.

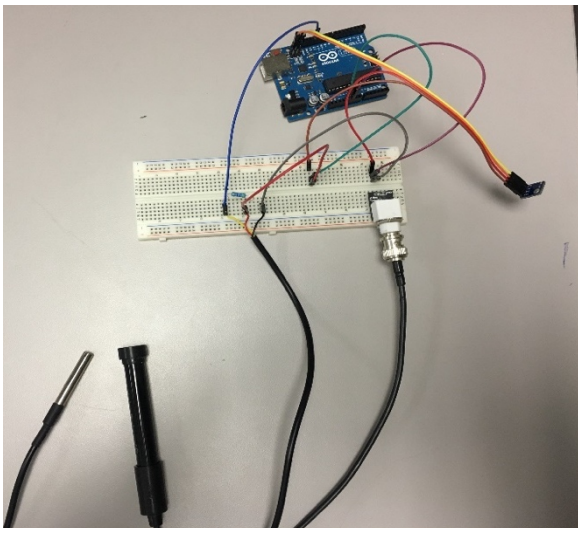


Figure 4.13: Circuit Implementation

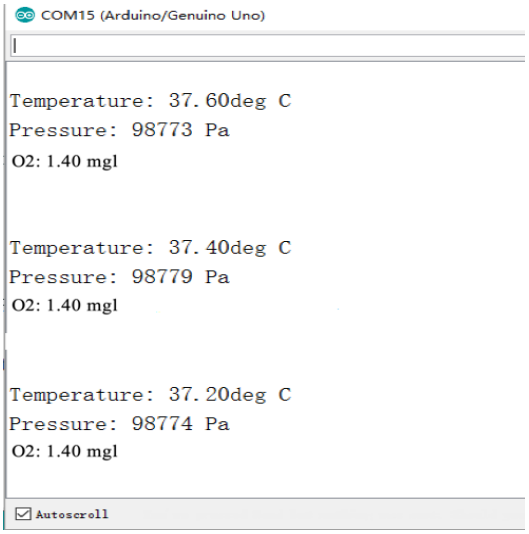


Figure 4.12: Results

5. Project Management

5.1Project Plan

	Project Management: Plan	
	Electrical Engineering Department	
	EEEN4311: Design Methodology & Project Management	
	Instructor: Dr. Chedly B. Yahya	Spring 2018

Project Title: Environmental Monitoring Vehicle		
Team:	Hassan Al Rumaeh (HA) Abdullah Al Buhumaidah (AS)	Mahdi Al Nasir (MN) Baker Al Takroni (BT)

SN	Tasks & Responsibilities		Begin**	End**	% Completion
1	Test and refine design of SS2 (Driving car)	HA, MN	Sep 2	Sep 15	100%
2	Design SS3 and verify equipment for (Wireless Control System)	BT, AS	Sep 16	Sep 19	100%
3	Implement SS3 (Wireless Control System)	MN, BT	Sep 24	Oct 1	100%
4	Test and refine SS3(Wireless Control system for Motors)	HA, AS	1 Oct.	Oct 6	100%
5	Implement design subsystem SS4 (Monitoring Natural Variables)	ALL	8 Oct.	Oct 15	100%
6	Test and refine design for SS4 (Monitoring Natural Variables)	MN, HA	Oct 16	Oct 22	100%
7	Integrate SS1(Driving Car), SS2(Float Boat) and SS3 (Wireless Control system for Motors)	AS, MN	Oct 31	Nov 3	100%
8	Test and troubleshoot SS1(Driving Car), SS2(Float Boat) and SS3 (Wireless Control system for Motors)	MN, BT	Nov 5	Nov 10	100%
9	Prepare midterm presentations	ALL	Nov 11	Nov 12	100%
10	Integrate and test SS3 & SS4	HA, AS	Nov 15	Nov 16	100%
9	Test and troubleshoot	HA, BT	Nov 16	Nov 20	100%
10	Integrate all subsystems	ALL	Nov 21	Nov 26	100%
11	Test and make final changes	ALL	Nov 27	Dec 7	100%
12	Writing final report	ALL	Nov 21	Dec 6	100%
13	Prepare Demo. (prototype)	ALL	Dec 8	Dec10	100%
14	Prepare a video	ALL	Dec 10	Dec 11	100%
** Dates: From week 1 Sep2 until week 14 Dec6			% Completion by set end date		

5.2 Contribution of Team Members

Table 5.1: Contribution of Team Members

Task	Baker	Abdullah	Hassan	Mahdi	Task Total
Search & acquire components	**25%**	**15%**	**20%**	**45%**	**100%**
Design Subsystems	**20%**	**35%**	**25%**	**15%**	**100%**
Test Subsystems	**30%**	**25%**	**25%**	**20%**	**100%**
Write Reports & Presentations	**25%**	**25%**	**30%**	**20%**	**100%**

5.3 Project Execution Monitoring

- Meetings with Advisors

In every two weeks, all the group members had a meeting with the advisor Dr.Jawad Alasad to discuss the latest updates of the project and getting the advices and recommendations from him for the project. One of his recommendations, is to have a ready-made structure for our vehicle, also he is the one who choose the features like measuring the variables inside the water, which are the oxygen, pressure and the temperature inside the water. He also invited us to his own swimming pool to test our project there, due to the closing of the PMU sport center. So, the meetings were very helpful to us and letting the advisor updated for everything to direct and advise us.

- Meetings with Group Members

The meetings between the group members were fluctuating between twice a week or more depends on the tasks that needs to be done. Usually, the tasks are divided for the group members equally, and the meetings are helpful to help each other if anyone has an issue with his part. One of the discussion was to discuss what will the shape of our design, and we did choose a very good material and shape to present our project.

- Design of Subsystems

For every subsystem, there was four or more tasks. So as a result, each member assigned to work on one part of each subsystem, by this methodology every member will work on everything in the whole project. The main idea is to divide the work between the members, and all of the members should work in every part, to increase the understanding and the creativity. The core of this project was done as a team.

5.4 Challenges and Decision Making

- Challenges

The first challenge that we had was about how to design the structure of the vehicle. The first idea was to 3D printing the vehicle. But we considered to look for another choice because it will cost a lot, and the designing of the vehicle to print it was a difficult and time consuming to everyone in the team. Then we had the problem of placing the subsystems on the structure. Also, the availability of the components was an issue to us. We had more challenges like, isolating the system from the water since it is an amphibious system so the isolation was an important thing to do. Also, the coding wasn't easy because we had many features and functions, doing them and integrate them together was a big challenge for our project. Also, the power consumption of the motors was very high and the batteries wasn't stay that long.

- Decision Making

For the first challenge which is about the structure of the vehicle we and our advisor Dr.Jawad Alasad met and discussed about the problem and he suggested to have a ready-made structure for our vehicle to save time and effort. So, we considered a viper glass structure due to its low weight also it is strong and solid. In addition, the advantage of it was cheaper than the 3D. For the second challenge which is placing the subsystems in that structure, we had the benefit from PMU labs to fabricate the structure of the vehicle to make holes in it and soldering the equipment's. Also, we found that the solution for the availability of the components is to order them online ahead of time, to have them early enough to test them and approve using them. We as a team members and our advisor had a meeting about the isolation part and how should it be done, and we considered the silicon to be our isolation material due to its performance in previous projects and it was a very good choice. The coding issue was very hard to solve because of the multiplicity of the functions, so coding them separately and integrate them later on was the hardest part, but after researching and going back to our microprocessor course experiments we found that building it all together from the beginning will be easier, and it worked. For the power consumption and battery draining we considered to have separate battery for each system and we did it and the battery's draining significantly improved.

5.5 Project Bill of Materials and Budget

Table 5.2: shows the total Budget for the project

Items	Quantity	Cost
Body	1	700 SR
Battery	3	600 SR
Arduino Uno Mega Kit	1	120 SR
Motors	5	470 SR
Xbee Comm.	2	196 SR
Joystick	1	600 SR
Wheels	4	60 SR
Temperature / pressure sensor	1	80 SR
Total	19	2826 SR

Chapter 6. Project Analysis

6.1 Life-long Learning

Designing and implementing subsystem one and subsystem two which are (Floating Boat) and (Driving Car) had affected us in very effective way, we have learned how to apply teamwork principle by discussing any issue as a team and provide solutions. For example, design subsystem one which is Floating Boat, we were thinking to do the Floating Boat using the 3D printer, but unfortunately there is not 3D printer available locally, so we had to discuss this issue and find an alternative solution, and as a team we found a suitable alternative which is readymade body boat that can be easily redesigned for subsystem two which is Driving Car. Also, we have explored that there are many types of motors and each one of them has its own characteristics so we made a research using the internet about motors type and which one of them is suitable for our design. However, during this course we have gained the value of time and it is important to divide our work into many tasks with specific time period assigned to group members.

6.2 Impact of Engineering Solutions

The main purpose of our engineering design to have better and healthier environment, as we all know that the water pollution, air pollution, high temperature and pressure could have a major harm on the human begins fishes or the plants, so, as a future engineer you have provide an engineering solution which is the Environmental Monitoring Vehicle. Basically, our project will collect precise readings of the temperature, pressure and oxygen inside and outside the water, after collecting the data we will take an action accordingly.

6.3 Contemporary Issues Addressed

Water Pollution and Air pollution are the most hazardous environmental issues in Saudi Arabia. First, air pollution in Saudi Arabia is originally come crude oil. While they are refining crude oil, it creates different kinds of greenhouse gases that create air pollution. On the other hand, during the process of desalination of sea water, machines heat sea water and gather the water vapor, which is gathered to be pure water for agricultural and industrial use. However, after the process of desalination, sea water becomes polluted with high concentration of salt. Aftermath of desalination, this water does not only contain high salt concentration, but it also contains high concentration of metal. This water cannot be used in agriculture and city, so mostly they throw it away to the sea. As a result, it damages ecology of the sea.

Chapter 7. Conclusions and Future Recommendations

7.1 Conclusions

In summary, after completing the senior design course we have successfully met all the requirements, also we followed our project management plan and executed all tasks assigned to be completed this Fall semester 2018/2019. As a result, we designed and implemented the Environmental Monitoring Vehicle, of course after looking at the previous projects we have got some ideas from them and we developed their ideas and come up with our project. However, after completing our project we have explored new experience and learned many things such as, time management which consider as an essential thing, planning and set tasks which contributes to help us completing the required tasks and finally teamwork, we had weekly meetings together in order to discuss what have been done so far, also we had to consult our adviser and give him the latest updates on our project.

7. 2 Future Recommendations

We highly recommend the following things to improve or system:

- Design the vehicle

As stated before, we had some issue using the 3D printer it was not available locally, because, it will be much better to design our own body according to our own specifications.

- Add different supporting wheel

The main idea of the supporting wheel is, to make the vehicle get in the desired direction easily, but the one we have sometimes it fails to do its job.

- Add additional sensor for gas leaking

It will add another advantage in the land to our project, and it will be very effective for confined places, such as pipelines.

- Send data wirelessly to control station

One of the important feature that could be added to our project, sending data to a control station wirelessly instead of waiting for the **v**ehicl**e** to come and get the data out from the SD card, data could be send wirelessly.

8. References

[1] Remote Controlled Submarine, *Prince Muhammed Bin Fahad University, Spring 2016*, www.pmu.edu.sa.

[2] Lifeguard Boat, *Prince Muhammed Bin Fahad University, Fall 2017*, www.pmu.edu.sa.

[3] Robots to Study Lake Tahoe, *University of Nevada July, 2016* http://www.kostasalexis.com/autonomous-robots-camp.html .

[4] Autonomous Targeting Vehicle (ATV), *Purdue University*, Spring 2011 www.purdue.edu/uns/html3month/2005/050502.Hirleman.car.html

[5] Amphibious *Robot* *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.95.5901&rep=rep1&type=pdf*

[6] https://www.dfrobot.com/wiki/index.php/Wirless_GamePad_V2.0_(SKU:DFR0182)#Specification

[7] https://www.arduino.cc/en/Reference/Libraries

[8] www.pmu.edu.sa.

Appendix A: Progress Reports

	Project Management: Progress Report	
	Electrical Engineering Department	
	EEEN4311: Design Methodology & Project Management (Fall 2018-19)	
Weeks 1-6	**Progress Report No. 1**	**Date: Oct 11 , 2018**

Project Title	**Environmental Monitoring Vehicle**	
TEAM	**Hassan Al Rumaeh (HA)**	**Mahdi Al Nasir (MN)**
	Abdullah Al Buhumaidah (AS)	**Baker Al Takroni (BT)**

Tasks		**Description (brief but clear)**	**Date**	**% Completion**
Completed Weeks 1 to 6	**1**	Test and refine design of SS2 (Driving car)	Sep 15	**100%**
	2	Design SS3 and verify equipment for (Wireless Control System)	Sep 19	**100%**
	3	Implement SS3 (Wireless Control System)	Oct 1	**100%**
	4	Test and refine SS3(Wireless Control system for Motors)	Oct 6	**100%**
	5			
In Progress Weeks 7 & 8	**6**	Implement design subsystem SS4 (Monitoring Natural Variables)	Oct 15	**30%**
	7	Implement design subsystem SS4 (Monitoring Natural Variables)	Oct 22	**0%**
	8	Test and refine design for SS4 (Monitoring Natural Variables)	Oct 31	**0%**
	9			
	10			
Comments	During this weekend, we will meet and try to catch up with our plan, specially for (Task 4).			
Deadlines Thu	**ProgRpt1 (Wks 1- 6) Th Oct 11**		**ProgRpt3 (Wks 9, 10) Th Nov 8**	
	ProgRpt2 (Wks 7, 8) Th Oct 25		**ProgRpt4 (Wks 11, 12) Th Nov 22**	

Project Management: Progress Report

Electrical Engineering Department

EEEN4311: Design Methodology & Project Management (Fall 2018-19)

Weeks 7-8	Progress Report No. 2	Date: Oct 25 , 2018

Project Title	Environmental Monitoring Vehicle	
TEAM	Hassan Al Rumaeh (HA)	Mahdi Al Nasir (MN)
	Abdullah Al Buhumaidah (AS)	Baker Al Takroni (BT)

Tasks		Description (brief but clear)	Date	% Completion
Completed Weeks 1 to 6	1	Implement design subsystem SS4 (Monitoring Natural Variables)	Oct 15	100%
	2	Test and refine design for SS4 (Monitoring Natural Variables)	Oct 22	100%
	3	Integrate both SS3 and SS4	Oct 31	100%
In Progress Weeks 7 & 8	6	Integrate both SS3 and SS4	Oct 31	40%%
	7	Prepare midterm presentations	Nov 7	20%
	8	Test and troubleshoot SS3, SS2 and SS3	Nov 11	0%
	9	Writing Final Report		
	10			
Comments				

Deadlines Thu				
	ProgRpt1 (Wks 1- 6) Th Oct 11			ProgRpt3 (Wks 9, 10) Th Nov 8
	ProgRpt2 (Wks 7, 8) Th Oct 25			ProgRpt4 (Wks 11, 12) Th Nov 22

Project Management: Progress Report

Electrical Engineering Department

EEEN4311: Design Methodology & Project Management (Fall 2018-19)

Weeks 9-10	Progress Report No. 3	Date: Nov 8 , 2018

Project Title	Environmental Monitoring Vehicle	
TEAM	Hassan Al Rumaeh (HA)	Mahdi Al Nasir (MN)
	Abdullah Al Buhumaidah (AS)	Baker Al Takroni (BT)

Tasks		Description (brief but clear)	Date	% Completion
Completed Weeks 9 & 10	1	Test and refine design for SS4 (Monitoring Natural Variables)	Oct 22	100%
	2	Integrate SS1(Driving Car), SS2(Float Boat) and SS3 (Wireless Control system for Motors)	Oct 31	100%
	3	Test and troubleshoot SS1(Driving Car), SS2(Float Boat) and SS3 (Wireless Control system for Motors)	Nov 5	100%
In Progress Weeks 11 & 12	6	Prepare midterm presentations	Nov 12	40%
	7	Integrate and test SS3 & SS4	Nov 15	25%
	8	Test and troubleshoot	Nov 16	15%
	9	Integrate all subsystems	Nov 21	0%
	10	Writing final report	-	35%
Comments				
Deadlines Thu	ProgRpt1 (Wks 1- 6) Th Oct 11			ProgRpt3 (Wks 9, 10) Th Nov 8
	ProgRpt2 (Wks 7, 8) Th Oct 25			ProgRpt4 (Wks 11, 12) Th Nov 22

Project Management: Progress Report

Electrical Engineering Department

EEEN4311: Design Methodology & Project Management (Fall 2018-19)

Weeks 11-12	Progress Report No. 4	Date: Nov 22 , 2018

Project Title	Environmental Monitoring Vehicle	
TEAM	Hassan Al Rumaeh (HA)	Mahdi Al Nasir (MN)
	Abdullah Al Buhumaidah (AS)	Baker Al Takroni (BT)

Tasks		Description (brief but clear)	Date	% Completion
Completed Weeks 9 & 10	1	Prepare midterm presentations	Nov 11	100%
	2	Integrate and test SS3 & SS4	Nov 15	100%
	3	Test and troubleshoot	Nov 16	100%
	4	Integrate all subsystems	Nov 21	100%
	5	Writing final report	Nov 21	100%
In Progress Weeks 11 & 12	6	Test and make final changes	Nov 27	100%
	8	Prepare a video	-	100%
	9	Prepare Demo. (prototype)	-	100%
	10	Writing final report	Nov 21	100%
Comments				

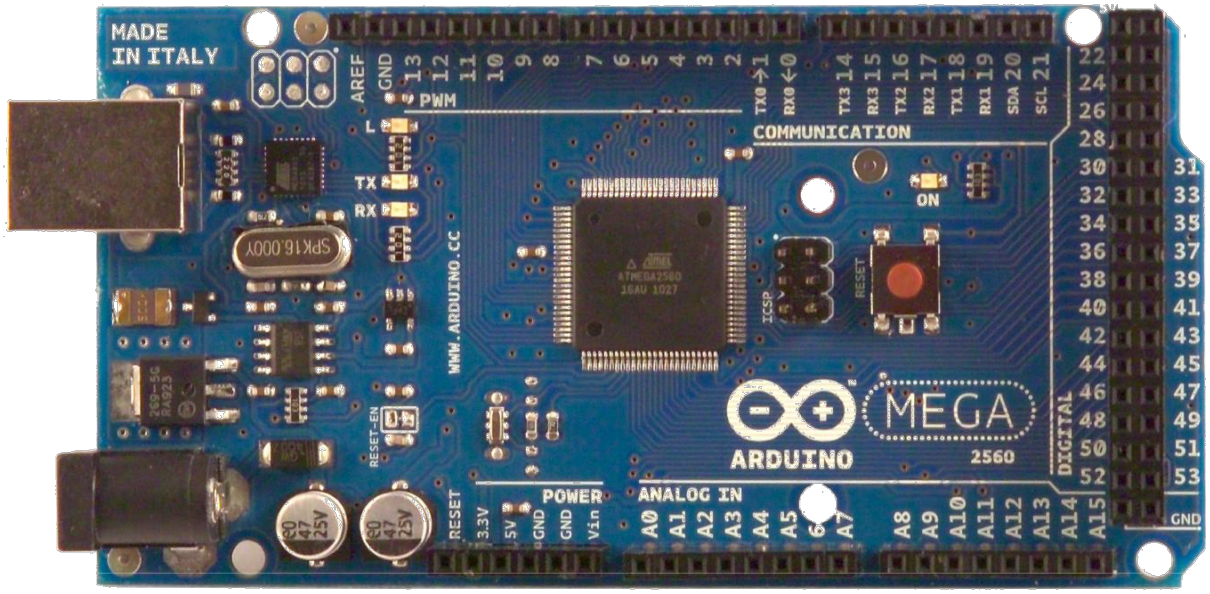
Deadlines Thu				
	ProgRpt1 (Wks 1- 6) Th Oct 11			ProgRpt3 (Wks 9, 10) Th Nov 8
	ProgRpt2 (Wks 7, 8) Th Oct 25			ProgRpt4 (Wks 11, 12) Th Nov 22

Appendix B: Bill of Materials

Items	Quantity	Cost
Body	1	-
Battery	2	SR 131
Arduino Uno Mega Kit	1	SR 120
Motors	6	SR 340
Xbee Comm.	2	SR195
Joystick	1	SR 100
Wheels	4	SR 60
Total	19	SR 1171

Appendix C: Microcontroller Data sheet

Arduino MEGA 2560



CE

Product Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Index

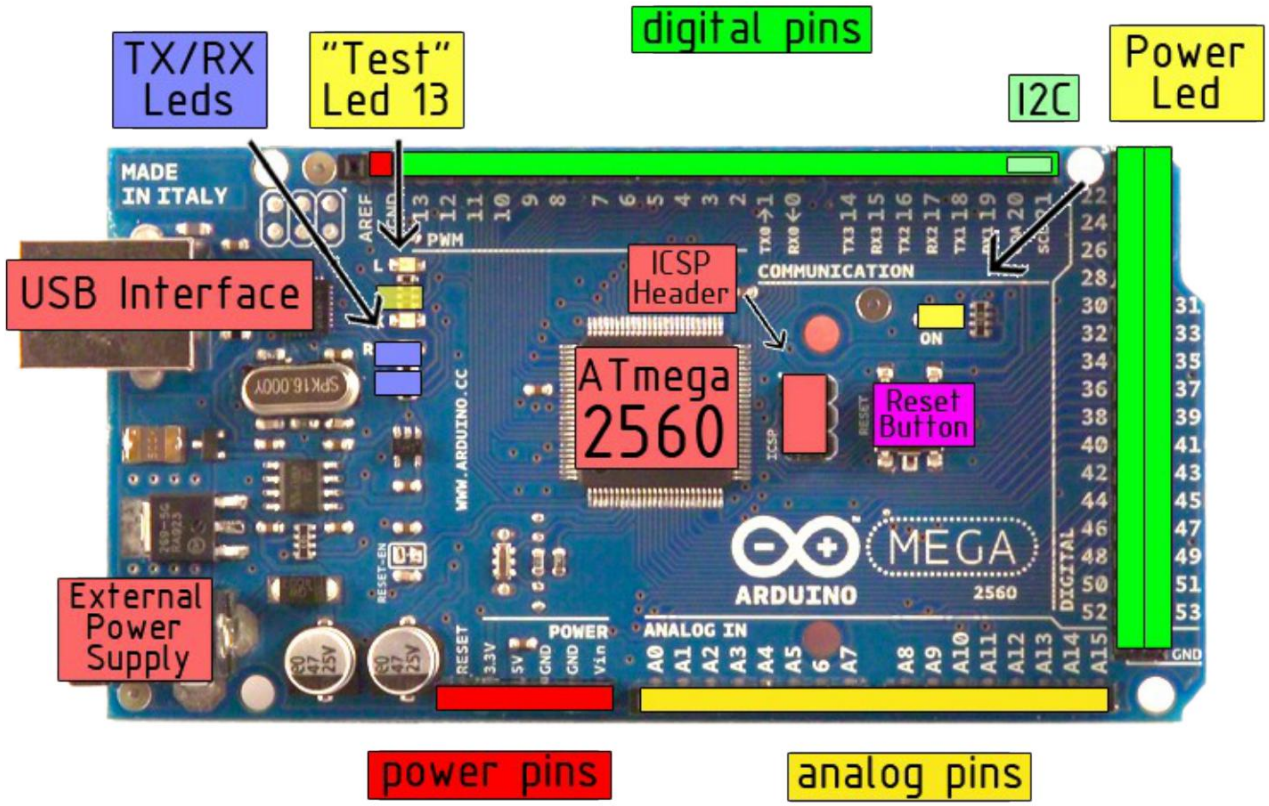
Technical Specification

EAGLE files: arduino-mega2560-reference-design.zip Schematic: arduino-mega2560-schematic.pdf

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

the board



Power

The Arduino Mega2560 can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode() , digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **I^2C: 20 (SDA) and 21 (SCL).** Support I_2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I_2C pins on the Duemilanove.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

· **AREF.** Reference voltage for the analog inputs. Used with analogReference().

- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega's digital pins.

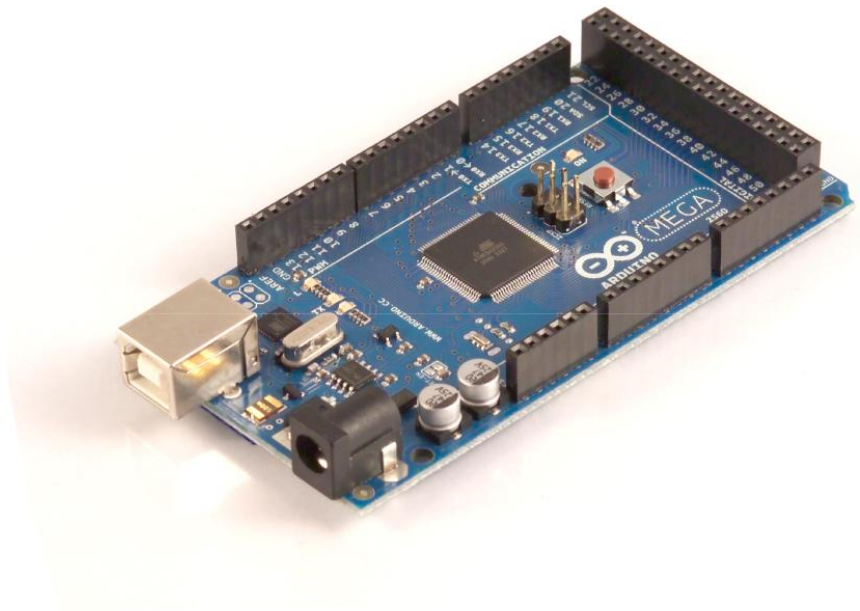
The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the ATmega2560 datasheet.

Programming

The Arduino Mega2560 can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Atmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.



Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Mega contains a trace that can be cut to disable the auto -reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

The maximum length and width of the Mega PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila. **Please note that I^2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).**

How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

Linux Install

Windows Install

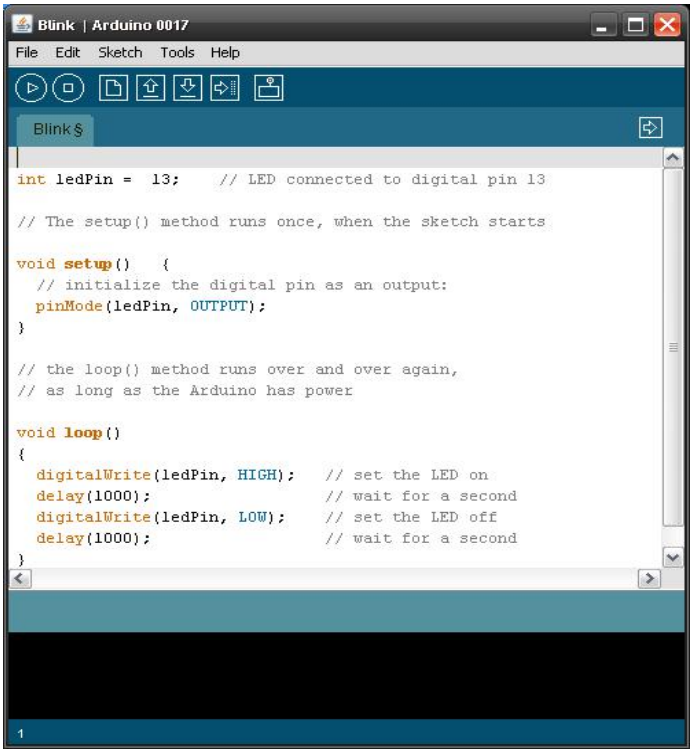
Mac Install

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

Blink led

Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

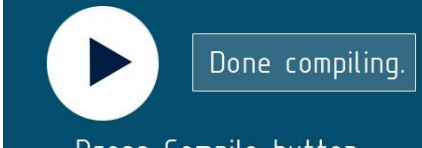
File>Sketchbook>
Arduino-
0017>Examples>
Digital>Blink



Once you have your skecth you'll see something very close to the screenshot on the right.

In **Tools>Board** select MEGA

Now you have to go to **Tools>SerialPort** and select the right serial port, the one arduino is attached to.

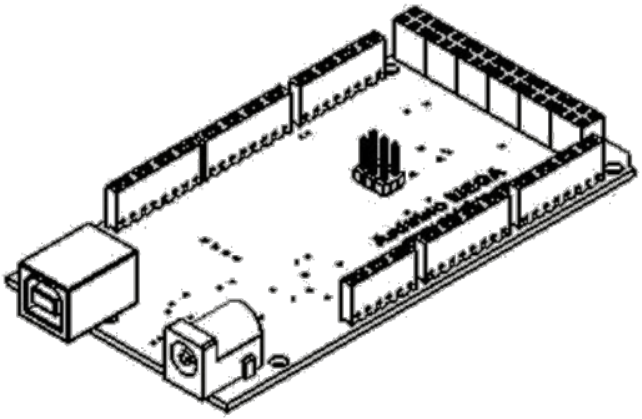


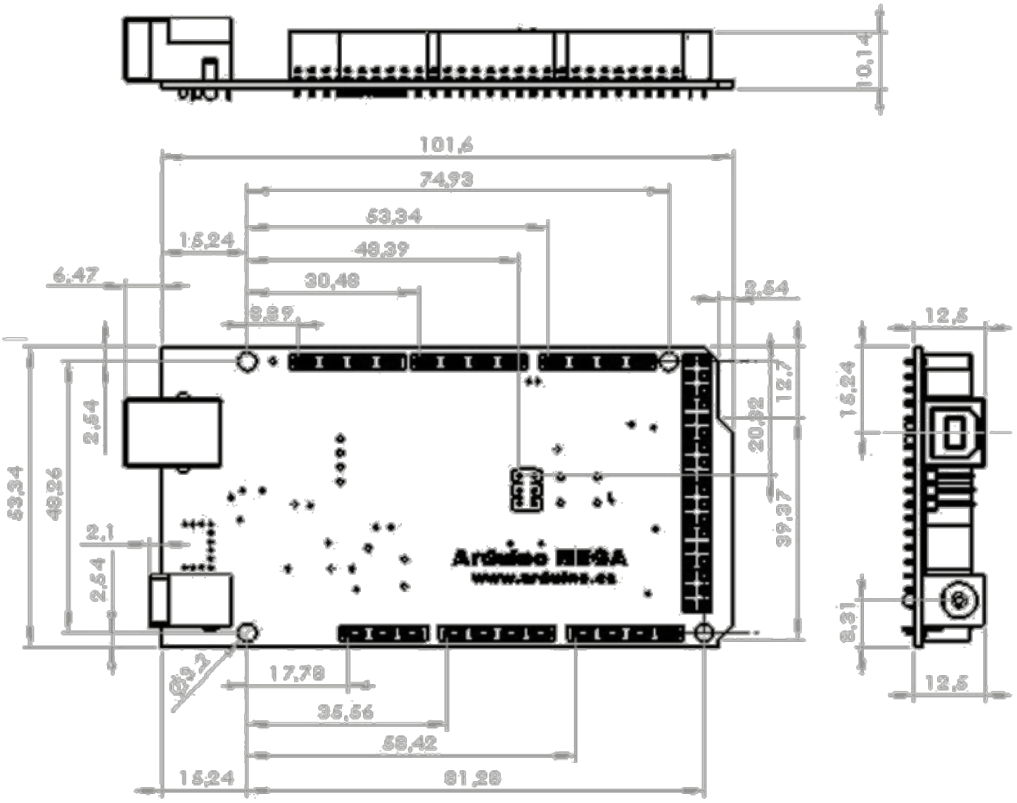
Press Compile button (to check for errors)

Upload

TX RX Flashing

Blinking Led!




10,16
101,6
74,93
53,34
15,24
48,39
6,47
30,48
8,89
2,54
12,5
12,7
15,24
20,32
39,37
53,34
48,26
2,54
2,1
2,54
Arduino MEGA
www.arduino.cc
8,31
Ø3,2
17,78
35,56
58,42
15,24
81,28
12,5

Terms & Conditions

1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

Enviromental Policies

The producer of Arduino ™ has joined the Impatto Zero® policy of LifeGate.it. For each Arduino board produced is created / looked after half squared Km of Costa Rica's forest'

Appendix D: ESC 1060 Data Sheet

HOBBYWING®

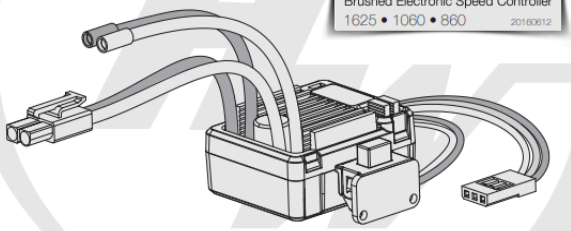
USER MANUAL

QUICRUN

Brushed Electronic Speed Controller

1625 • 1060 • 860

20160612



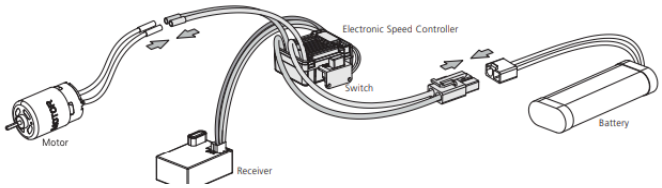
Congratulations and thanks for purchasing Hobbywing QUICRUN series electronic speed controller (ESC) for brushed motor. The power system for RC model can be very dangerous, so please read this manual carefully. Since we have no control over the installation, application, use or maintenance of this product, in no case shall we be liable for any damages, losses or costs.

01 Features

- Water-proof and dust-proof, suitable for all-weather condition races.
- Small size with built-in capacitor module.
- Three running modes: Fwd/Br, Fwd/Rev/Br and Fwd/Rev, fits for various vehicles.
  Note 1: Fwd =Forward, Br=Brake, Rev=Reverse.
  Note 2: QUICRUN-WP-1625-BRUSHED ESC only has the Fwd / Rev / Br mode.
- Great current endurance capability.
- Great built-in BEC output capacity.
- Automatic throttle range calibration, easy to use.
- Easy to set the ESC parameters with jumpers.
- Multiple protections: Low voltage cut-off protection for battery / Over-heat protection / Throttle signal loss protection.

02 Begin to Use the New Brushed ESC

1 Connections



Turn off the ESC switch, wire the battery, motor, ESC, servo, receiver according to the following diagram. Recheck the wiring to ensure all connections are correct before getting into the next step.

1) Once the power is wrongly connected (that means the battery polarity is mistakenly reversed), irreparable damage may occur to the ESC and batteries. Therefore, please pay close attention to the battery polarity.
2) Please swap the two wire connections if the motor rotate in the opposite direction.

Specifications

Model		QuicRun-WP-1625-BRUSHED	QuicRun-WP-1060-BRUSHED	QuicRun-WP-860-DUAL-BRUSHED
Fwd. Cont. / Peak Current		25A/100A	60A/360A	60A/360A
Rev. Cont. / Peak Current		25A/100A	30A/180A	30A/180A
Voltage Range		2-3S Lipo or 5-9 NiMH		2-4S Lipo or 5-12 NiMH
Cars Applicable		1/18 &1/16: Touring Car, Buggy, Monster, Truggy	1/10: Touring Car, Buggy, Short Course Truck, Monster, Truggy, Rock Crawler and Tank	1/8: Touring Car, Buggy, Short Course Truck, Monster, Truggy, Rock Crawler and Tank
Motor Limit *	2S Lipo or 6 NiMH	280, 370 or 380 Size Motor: RPM<30000 @7.2V	540 or 550 Size Motor: ≥12T or RPM<30000 @7.2V	540, 550, 775 Size Motor: ≥12T or RPM<30000 @7.2V
	3S Lipo or 9 NiMH	280, 370 or 380 Size Motor: RPM<20000 @7.2V	540 or 550 Size Motor: ≥18T or RPM< 20000 @7.2V	540, 550, 775 Size Motor: ≥18T or RPM<20000 @7.2V
	4S Lipo or 12 NiMH	Not Available		540, 550, 775 Size Motor: ≥24T or RPM<15000 @7.2V
Resistance		Fwd 0.003Ω, Rev 0.003Ω	Fwd 0.001Ω, Rev 0.002Ω	Fwd 0.001Ω, Rev 0.002Ω
BEC Output		1A / 6V (Linear Mode)	3A / 6V (Switch Mode)	3A / 5V (Switch Mode)
Dimension / Weight		34x24x14mm / 23.5g	36.5x32x18mm / 39 g	46x36x26.3mm / 73g
Cooling Fan			Without cooling fan	With cooling fan. It is supplied from receiver.
Running Modes		Forward / Reverse / Brake	Forward / Reverse / Brake, Forward / Brake, Forward / Reverse	Forward / Reverse / Brake, Forward / Brake, Forward / Reverse, Boat

**Note: WP-860-DUAL-BRUSHED has two outputs to drive 2 motors. When driving 2 motors simultaneously, the Turns of the motors need to be increased.*

2 Set the Throttle Range

Turn on the transmitter, and set parameters (of the throttle channel) like "D/R, "EPA", "ATL" to 100% (if there is no LCD display on the transmitter, please adjust the corresponding knob to its limit). Set the throttle trim to 0 (if there is no display, then adjust the knob to the neutral position). For FUTABA™ and similar transmitters, set the throttle direction to "REV", while the throttle direction of others to "NOR". Please disable the built-in ABS brake function in your transmitter.

Besides, we strongly recommend users to enable the "Fail Safe (F/S)" function of the transmitter, set the "F/S" of the throttle channel to the Shutdown mode or set the protection value to the neutral position, so the car can be stopped if the receiver fails to get the radio signals from the transmitter.

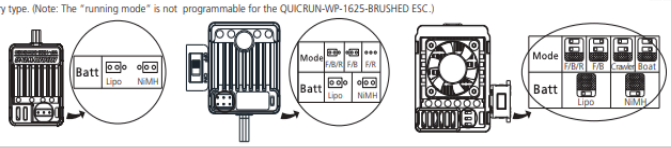
Calibrate the throttle range: Turn on the ESC switch, set the throttle stick to the neutral point and then wait 3 seconds for the completion of throttle range self-calibration; Beep sound emits if the self-calibration is sucessfully passed, then the ESC is ready to run.

The Meaning of Beep Sound	LED Status (in Running)	Throttle Stick Position
• 1 short Beep: The battery is NiMH • 2 short Beeps: The battery is 2S Lipo • 3 short Beeps: The battery is 3S Lipo • 4 short Beeps: The battery is 4S Lipo • 1 long Beep: Self-test and throttle range calibration is OK, the ESC is ready to run.	• When the throttle stick is in neutral range, red LED is off • Partial throttle forward, partial brake or partial reverse, red LED blinks • Full throttle forward, maximum brake or full throttle reverse,redLED is solid on	Turn on the switch Neutral point

03 Set the ESC Parameters

How to Set Parameters:

1. QUICRUN-WP-1625 / 1060-BRUSHED ESC uses the jumper cap to set running mode & battery type. (Note: The "running mode" is not programmable for the QUICRUN-WP-1625-BRUSHED ESC.)
   Way to set: We suggest users use the tweezers to set parameters by plugging / unplugging the jumper cap (as shown in the picture beside);
   For example, if want set the battery type to the "LiPo" mode, you only need to plug the jumper cap into left two pins of the battery pin header.
2. QUICRUN-WP-860-DUAL-BRUSHED ESC uses the dial switch to set running mode & battery type.
   Way to set: recommend using the tweezers to set parameters by flipping the DIP switch (for detailed explanation, please refer to the following picture); if want to set the battery type to the "LiPo" mode, you only need to flip the battery dial switch to the left position.



Programmable Items WP-1625-BRUSHED WP-1060-BRUSHED WP-860-DUAL-BRUSHED

1. Running Mode: 3 Options (Fwd / Br / Rev, Fwd / Br, Fwd / Rev). The "Fwd / Br / Rev" is the default option. Fwd=Forward, Br=Brake, Rev=Reverse

"Fwd / Br / Rev" mode indicates the vehicle can go forward, backward and brake. This mode uses "Double-click" method to make the vehicle reverse. When moving the throttle stick from the neutral zone to backward zone for the 1st time, the ESC begins to brake the motor and the motor slows down but still running, so the backward action is NOT performed immediately. When the throttle stick is moved to the backward zone again, if the motor speed slows down to zero (i.e. stopped), the backward action will happen. This "Double-click" method prevents mistakenly reversing action when the brake function is frequently used in steering. Therefore, this mode is often used in daily practice. For the "Fwd / Br" mode, the vehicle can go forward and brake, but no reversing, so this mode is often used in competitions. And the "Fwd / Rev" mode uses "Single-click" method to make the vehicle reverse, when moving the throttle stick from neutral zone to backward zone, the vehicle reverses immediately, so this mode is usually used for rock crawler. *(Note:WP-1625-BRUSHED has no optional running mode except the default "Fwd / Br / Rev" mod.)*

"Boat" mode: this mode used some brand-new algorithm that is specially designed for RC boats. (Only the WP-860-DUAL-BRUSHED ESC has this option in its programmable items.)

2. Battery Type: 2 Options (Lipo, NiMH), the "Lipo" is the default option.

04 Protection Features

1. Low Voltage Cutoff Protection: If the voltage of battery pack is lower than the threshold for 2 seconds, the ESC will enter the protection mode, so the motor speed will be lowered (when voltage is lower than the 1st trigger point) till stopped (when voltage is lower than the 2nd trigger point). When the car stops, the red LED blinks to indicate the low voltage cut-off protection has been activated.

2S Lipo	3S Lipo	4S Lipo	5-9 NiMH
When the voltage is below 6.5V, the output power will be halved. When the voltage is lower than 6.0V, the output will be cut off and won't be resumed again.	When the voltage is below 9.75V, the output power will be halved. When the voltage is lower than 9.0V, the output will be cut off and won't be resumed again.	When the voltage is below 13.0V, the output power will be halved. When the voltage is lower than12.0V, the output will be cut off and won't be resumed again.	When the voltage is below 4.5V, the output power will be halved. When the voltage is lower than 4.0V, the output will be cut off and won't be resumed again.

Note: when setting the WP-860-DUAL-BRUSHED ESC to the" Boat" mode, the motor will stop running when the LVC protection is activated. Please move the throttle stick to neutral position, after that the motor can be started up again but the output power will be halved.

2. Over-heat Protection: When the internal temperature of the ESC is higher than 100 Celsius degrees, this protection will be activated and the output power will be reduced till cut off. The RED LED blinks when the vehicle stops, and the ESC will not resume output power until its temperature is below 80 Celsius degrees.

3. Throttle signal loss protection: The ESC will cut off the output power if the throttle signal has been lost for 0.1 second. The "Fail Save" function of the radio system is strongly recommended to be activated.

05 Troubleshooting

Troubles	Possible Causes	Solutions
After power on, no LED lights up, no self-test and no beep sound.	No power is drawn to the ESC; The switch of the ESC is broken.	Check the connections between battery and ESC. Re-solder the connectors if needed; Change the ESC switch.
After turn on, the RED LED blinks but the motor doesn't work.	Throttle wire is wrongly plugged or into the incorrect channel; The ESC can't successfully complete the throttle range self-calibration.	Plug the throttle signal wire correctly (in right direction) into the throttle channel (usually Ch2) of the receiver; Set the "TRIM" of throttle channel to 0 or turn the knob to its neutral position.
The car runs backwards when accelerating forward on the transmitter.	The direction setting of the throttle channel is incorrect in the transmitter or the motor wires are wrongly connected.	Reverse the direction of the throttle channel, from the original "NOR" to "REV" or "REV" to "NOR"; Swap the wires between the ESC and motor.
The vehicle can't reach to the full speed even at the full throttle, and the RED LED doesn't keep lighting.	There are some incorrect settings in the transmitter.	Set D/R, EPA, ATL to 100% for the throttle channel or turn the knobs to maximum value. Set TRIM to 0 or turn the knob to its neutral position.
Vehicle can't reverse.	The corresponding jumper is plugged into the wrong position; Neutral point of the throttle is drifted or deviated.	Insert the jumper into the right location; Set the "TRIM"of the throttle channel to 0 or turn the knob to its neutral point.
Motor suddenly stops running.	The throttle signal is lost; The low voltage cutoff protection or thermal protection (i.e. over heat protection) of the ESC is activated.	Check the connections between ESC and receiver. Check the transmitter and receiver). Check whether the battery voltage of the transmitter is too low; The RED LED on the ESC blinks, denoting the ESC is under low voltage cutoff protection or over-heat protection. Please check the ESC temperature, if it is too hot, please let the ESC cool down. If the battery voltage is low, please change the battery.
The vehicle neither go forward no reverse, but the LED indicators work normally.	The connection between ESC and motor is interrupted; The motor is damaged.	Check the connectors between the motor and ESC to ensure all connections are firm and reliable; Replace a new motor.
The motor accelerates rapidly at the startup moment, but has lockout or cogging problem.	The discharge capacity of the battery is not strong enough; The motor rotates too fast, and the gear ratio is too aggressive; Something wrong with the driveline of the vehicle.	Change a battery with better discharge capability; Use a motor with lower RPM, or smaller pinion to soften the gear ratio; Check the driveline of the vehicle.

Appendix E: Arduino Code

Transmitter code:

```
#define BRUSHLESS_UP 5
#define BRUSHLESS_DOWN 7
#define SERVO_1 A5
#define SERVO_2 A3
#define BRUSHED_1_UP 13
#define BRUSHED_1_DOWN 14
#define BRUSHED_2_UP 16
#define BRUSHED_2_DOWN 15
#define ONE 9
#define TWO 11
#define THREE 12
#define FOUR 10
#define SELECT 3
#define START 4
#define RIGHT 8
#define virbrationMotorPin 2
void setup()
{
  Serial1.begin(9600);
  pinMode(virbrationMotorPin, OUTPUT);
  digitalWrite(virbrationMotorPin, LOW); // Stop shacking of the
gamepad
  pinMode(TWO, INPUT_PULLUP);
  pinMode(FOUR, INPUT_PULLUP);
  pinMode(ONE, INPUT_PULLUP);
  pinMode(THREE, INPUT_PULLUP);

}
unsigned long timer = 0;
int BrushlessValue = 0,
             Brushed_1_Value = 500,
             Brushed_2_Value = 550,
             BrushlessLastValue = 500,
             Brushed_1_LastValue = 500,
             Brushed_2_LastValue = 550 ;
unsigned char Servo_1_Value = 90,
```

```
              Servo_2_Value = 90,
              Servo_1_LastValue = 90,
              Servo_2_LastValue = 90;
boolean PIEZO = 0;
long BRUSHED_1_Seconds=millis(),BRUSHED_2_Seconds=millis();
void loop()
{
  UpdateValues();
  SendValues();
  delay(100);
}
void UpdateValues() {

  if (digitalRead(BRUSHED_2_UP) == LOW &&  (millis()-
BRUSHED_2_Seconds) > 100) {
   Brushed_2_Value += 25;
   Brushed_2_Value = constrain(Brushed_2_Value, 0, 800);
   BRUSHED_2_Seconds=millis();
  }
  else if (digitalRead(BRUSHED_2_DOWN) == LOW &&  (millis()-
BRUSHED_2_Seconds) > 100) {
   Brushed_2_Value -= 50;
   Brushed_2_Value = constrain(Brushed_2_Value, 0, 800);
   BRUSHED_2_Seconds=millis();
  }
  if (digitalRead(BRUSHED_1_UP) == LOW  &&  (millis()-
BRUSHED_1_Seconds) > 100 ) {
   Brushed_1_Value += 25;
   Brushed_1_Value = constrain(Brushed_1_Value, 0, 800);
   BRUSHED_1_Seconds=millis();
  }
  else if (digitalRead(BRUSHED_1_DOWN) == LOW &&  (millis()-
BRUSHED_1_Seconds) > 100) {
   Brushed_1_Value -= 50;
   Brushed_1_Value = constrain(Brushed_1_Value, 0, 800);
   BRUSHED_1_Seconds=millis();
  }

  if (digitalRead(BRUSHLESS_UP) == LOW) {
  BrushlessValue += 50;
  BrushlessValue = constrain(BrushlessValue, 0, 1000);
  }
```

```
  else if (digitalRead(BRUSHLESS_DOWN) == LOW) {
  BrushlessValue -= 50;
  BrushlessValue = constrain(BrushlessValue, 0, 1000);
  }

  if (digitalRead(TWO) == LOW){
  Servo_1_Value =180;
  Servo_1_Value = constrain(Servo_1_Value, 0, 180);
  Serial.print(",TWO PRESSED:");
  Serial.println(Servo_1_Value);
 }
 else if (digitalRead(FOUR) == LOW ){
  Servo_1_Value =0;
  Servo_1_Value = constrain(Servo_1_Value, 0, 180);
  Serial.print(",FOUR PRESSED:");
  Serial.println(Servo_1_Value);
  Serial.print(",");
 }
 else {
  Servo_1_Value=90;
  }
 if (digitalRead(ONE) == LOW ){
  Servo_2_Value =150;
  Servo_2_Value = constrain(Servo_2_Value, 0, 180);
  }
  else if (digitalRead(THREE) == LOW ){
  Servo_2_Value =30;
  Servo_2_Value = constrain(Servo_2_Value, 0, 180);
  }
 else {
  Servo_2_Value=90;
  }
}
void SendValues() {
  if (Brushed_1_Value != Brushed_1_LastValue ) {
  Serial1.print(",B");
  Serial1.print(Brushed_1_Value);
  Serial1.print(",");
  Serial.print(",B");
  Serial.print(Brushed_1_Value);
  Serial.print(",");
    Brushed_1_LastValue = Brushed_1_Value;
```

```
  }
  if (Brushed_2_Value != Brushed_2_LastValue ) {
  Serial1.print(",b");
  Serial1.print(Brushed_2_Value);
  Serial1.print(",");
  Serial.print(",b");
  Serial.print(Brushed_2_Value);
  Serial.print(",");
  Brushed_2_LastValue = Brushed_2_Value;
}
   if (BrushlessValue != BrushlessLastValue ){
  Serial1.print(",L");
  Serial1.print(BrushlessValue);
  Serial1.print(",");
  Serial.print(",L");
  Serial.print(BrushlessValue);
  Serial.print(",");
  BrushlessLastValue = BrushlessValue;
 }
 if (Servo_1_Value != Servo_1_LastValue ){
  Serial1.print(",S");
  Serial1.print(Servo_1_Value);
  Serial1.print(",");
  Serial.print(",Servo");
  Serial.print(Servo_1_Value);
  Serial.print(",");
  Servo_1_LastValue = Servo_1_Value;
 }
 if (Servo_2_Value != Servo_2_LastValue ){
  Serial1.print(",s");
  Serial1.print(Servo_2_Value);
  Serial1.print(",");
  Serial.print(",Servo2");
  Serial.print(Servo_2_Value);
  Serial.print(",");
 Servo_2_LastValue = Servo_2_Value;
 }
 if (digitalRead(START) == LOW) {
 Serial1.print(",D,");
 Serial.print(",D,");
}
 else if (digitalRead(SELECT) == LOW) {
```

```
 Serial1.print(",d,");
 Serial.print(",d,");
  }
   if (digitalRead(RIGHT) == LOW) {
  Serial1.print(",P,");
  Brushed_2_LastValue =550;
  Brushed_2_Value=550;
  Brushed_1_LastValue =500;
  Brushed_1_Value=500;
}
}
```

Receiver Code:

```
#include <SPI.h>
#include <SD.h>
#include <Servo.h>
#include <Wire.h>
#include <SFE_BMP180.h>
#define PIEZO_PIN 7
#define BRUSHLES_PIN    8
#define BRUSHED_1_PIN   4
#define BRUSHED_2_PIN   5
#define SERVO_1_PIN     6
#define SERVO_2_PIN     7
#define READING_PERIOD 5  // 60 seconds
Servo Brushles;
Servo Brushed_1;
Servo Brushed_2;
Servo Servo_1;
Servo Servo_2;
SFE_BMP180 bmp180;

boolean sensor_string_complete = false;
String sensorstring = "";
void setup() {
Brushles.attach(BRUSHLES_PIN);
Brushed_1.attach(BRUSHED_1_PIN);
```

```
Brushed_2.attach(BRUSHED_2_PIN);
Servo_1.attach(SERVO_1_PIN);
Servo_2.attach(SERVO_2_PIN);
bmp180.begin();
Serial.begin(9600);
Serial1.begin(38400);
Brushles.writeMicroseconds(1000);
Brushed_1.writeMicroseconds(1500);
Brushed_2.writeMicroseconds(1500);
Servo_1.write(90);
Servo_2.write(90);
sensorstring.reserve(30);

if (  SD.begin()){Serial.println("Opened: ");
}else{  Serial.println("Not Opened: ");}
delay(3000);
}
int Brushed_1_LastValue=0,Brushed_2_LastValue=0;
double T ,P;
long Reading_period_Seonds = millis();
boolean Data_log_flag =0;
void serialEvent1() {
sensorstring = Serial1.readStringUntil(13);
sensor_string_complete = true;
}
void loop () {
if (millis()-Reading_period_Seonds >= (READING_PERIOD*1000) &&
Data_log_flag ==1){
   File myFile = SD.open("Data.TXT", FILE_WRITE);
   delay(200);
if (myFile){

delay (100);

  TEMP_PRE( );

  myFile.print("Pressure: ");
  myFile.print(P);
  myFile.print(" hPa\t\t");

  myFile.print("Temperature: ");
  myFile.print(T);
```

```
  myFile.print(" C\t\t");

  myFile.print("O2: ");
  if (sensor_string_complete == true) {
  myFile.print(sensorstring);
  sensorstring = "";
  sensor_string_complete = false;
        }
        else {
    myFile.print("-");
    }

  myFile.println(" mgl");

  Reading_period_Seonds = millis();
  myFile.close();
  }
}
delay(100);
while (Serial.available()>0) {

unsigned int value=0;
char Command='a' , c = 'a';
while (c != ',') {

  c = Serial.read();
  delay(1);
  if ( c == 'L' || c == 'B' || c == 'b' || c == 'S' || c ==
's'|| c == 'P'|| c == 'D'|| c == 'd') {
    Command = c;
  }
  else if (c != ',') {
    value = (value * 10) + (c - 48) ;
    }
  }

  if (Command == 'L') {
    value = value + 1000;
    Brushles.writeMicroseconds(value);

  }
  else if (Command == 'B') {
```

```
      value = value + 1000;

      if (value <=1500 && Brushed_1_LastValue>1500)
      {
        Brushed_1.writeMicroseconds(1000);
        delay(1);
        Brushed_1.writeMicroseconds(1500);
       }

    Brushed_1.writeMicroseconds(value);
    Brushed_1_LastValue = value;
   }
    else if (Command == 'b') {
      value = value + 1000;

     if (value <=1550 && Brushed_2_LastValue>1550)
     {
     Brushed_2.writeMicroseconds(1000);
     delay(1);
     Brushed_2.writeMicroseconds(1550);
     }
     Brushed_2.writeMicroseconds(value);
     Brushed_2_LastValue = value;
     }
else if (Command == 'P') {
 Brushed_2.writeMicroseconds(1000);
Brushed_1.writeMicroseconds(1500);

delay(200);
Brushed_2.writeMicroseconds(1500);
Brushed_2_LastValue = 1550;
Brushed_1_LastValue = 1500;
}
  else if (Command == 'S') {
    Servo_1.write(value);
  }
  else if (Command == 's')
  {
  Servo_2.write(value);
  }
  else if (Command == 'D') {
  Serial.println("RECEIVED D");
```

```
  Data_log_flag=1;
  }
   else if (Command == 'd') {
Data_log_flag=0;
  }
}
}
void TEMP_PRE( ){
  char Status;
Status = bmp180.startTemperature();
if (Status != 0) {
  Status = bmp180.getTemperature(T);
  if (Status) {
    Status = bmp180.startPressure(3);
    if (Status) {
      delay(Status);
      Status = bmp180.getPressure(P, T);
      }
    }
  }
}
```