Mapua Institute of Technology

School of Electrical, Electronics and Computer Engineering

Robotics and Mechatronics 4

ECE184P/B11

**Wireless Controlled Amphibious Vehicle**

**Using Zigbee**

In partial fulfillment of the requirements

in Robotics and Mechatronics 4 course

Submitted by:

Bulaong, Marc Lorenz D.

Carlos, Jennie S.

Fernandez, Krishia F.

Granada, Norbielle O.

Ibasco, Charmane Kimberly D.

Macose, Anna Marie P.

Silisilon, John Alfred III. R.

Uy, Jeffrey H.

Submitted to:

Engr. Glenn Magwili

December 2015

**ACKNOWLEDGEMENT**

After 11 weeks of doing hard work. We already come up with a prototype of a wireless controlled amphibious vehicle using Zigbee. But of course, this is not possible if we didn’t helped by people around us.

First of all, we dedicate this prototype to our Heavenly Father who guided us in doing this. We thank Him because he had helped us a lot in every problem that we encounter.

We would like to thank our beloved families that support and encourage us. Especially Macose family, who helped us in cutting and fixing some hardware problems and allowing us to work on their place.

We would like also to acknowledge our thanks to our friends who had helped and supported us while doing this project. But, as Robotics Student, we would also like to thank Engr. Glenn Magwili, our beloved Robotics and Mechatronics professor, for sharing his knowledge to us and for helping us with this project. Without his suggestions, this project will not be possible and we would not made this far. Also, Sir Moching for lending a helping hand in every problem we encountered.

We are also grateful to all the staff of Mapua Institute of Technology for allowing us to use facilities for keeping the project and for tools we borrowed that we needed on our project.

Lastly, we would never have this successfully if we don’t cooperate with each other so we would like to thank our group members or should we say new found friends in helping and contributing time and effort in this project.

To all… Thank you for inspiring us.

**OBJECTIVES**

• To be able to build a working amphibious vehicle using the all the concepts learned in the Robotics and Mechatronics Track.

• To be able to wirelessly control the amphibious vehicle through the Zigbee.

**DESCRIPTION**

**ZigBee** is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other wireless personal area networks (WPANs), such as Bluetooth or Wi-Fi. Applications include wireless light switches, electrical meters with in-home-displays, traffic management systems, and other consumer and industrial equipment that requires short-range low-rate wireless data transfer.

A **relay** is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

The mechanism behind wipers is the windshield **wiper motor**, which provides the power the wipers need. A linkage converts the rotational output of the windshield wiper motor into the back-and-forth motion of the wipers.

An electric **battery** is a device consisting of two or more electrochemical cells that convert stored chemical energy into electrical energy. Each cell has a positive terminal, or cathode, and a negative terminal, or anode. The terminal marked positive is at a higher electrical potential energy than is the terminal marked negative. The terminal marked positive is the source of electrons that when connected to an external circuit will flow and deliver energy to an external device. When a battery is connected to an external circuit, electrolytes are able to move as ions within, allowing the chemical reactions to be completed at the separate terminals and so deliver energy to the external circuit. It is the movement of those ions within the battery which allows current to flow out of the battery to perform work.

A **gizDuino** is an open source computing platform based on a simple input/output (I/O)board and the use of standard programming language; in otherwords, it is a tool for implementing a program you have designed. Gizduino is programmed using the IDE (Integrated Development Environment). With Serial RX-TX disable switch. Atmega168P ICs are low power but its also the same functions of ATmega168 IC.

**Universal Asynchronous Receiver Transmitter** is a microchip that controls a computer's interface to its attached serial devices. Specifically, it provides the computer with the RS-232C Data Terminal Equipment interface so that it can "talk" to and exchange data with modems and other serial devices**.**

**ACTUAL OPERATION**

You have to turn on the enable button on the controller in order to have the ability to choose a direction of where the vehicle would go otherwise it would not run at all. After turning the enable on, you can then choose which direction you want the vehicle to turn by pushing the designated buttons and their corresponding direction. In order to change directions, you would need to turn off and press the previous direction button before pressing another direction button. By not pushing any button or by turning off all buttons, you essentially put the vehicle to a stop. The vehicle has 5 states that it can perform which is forward, reverse, left, right and stop. Each direction has a corresponding button while the stop happens if no button has been pressed.

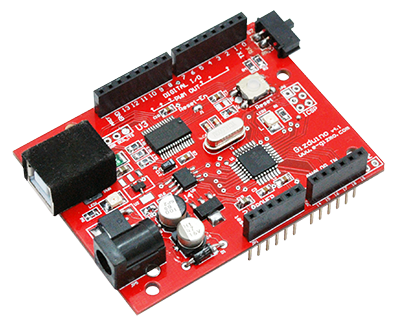
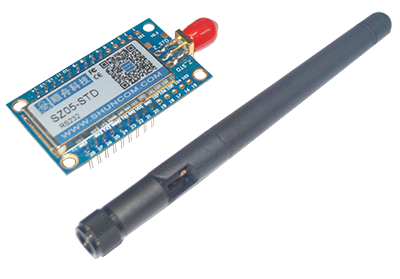
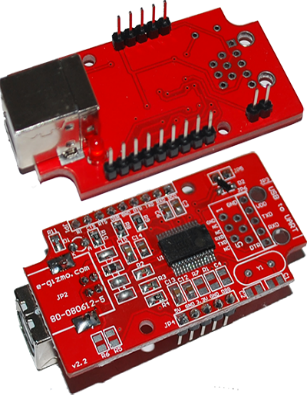
**SCOPE AND LIMITATIONS**

* The vehicle has a 200m maximum distance allowed to be controlled by the controller. Going further would cut the connection between the transmitter and the receiver.
* The Transmitter and the Receiver is powered by a power bank and needs charging every few hours of constant use.
* The Relays and the Wiper Motors are powered by Lead-Acid Batteries and needs to be charged every few hours of constant use.
* The Vehicle has a maximum capacity load of 30 kilograms. Going over it would result in slower movement of the vehicle.

**MATERIALS**

* Relays
* 2 pcs. 12V Battery
* 2 pcs. gizDuino Version 4.1
* 2 pcs. Zigbee
* 2 pcs. Power bank
* 2 pcs USB cable
* UART
* Connecting Wires
* Male to female connector
* Epoxy
* Wiper Motor
* Water Drum
* 2 pcs. Old kiddie car Wheels
* 2 pcs. Old bike wheels
* Plywood
* Paint
* Paint Brush
* Scratch Papers
* Arduino IDE
* Hyperterminal

\*Photos of some of the materials used



gizDuino v4.1 Zigbee (SZ05-STD) UART

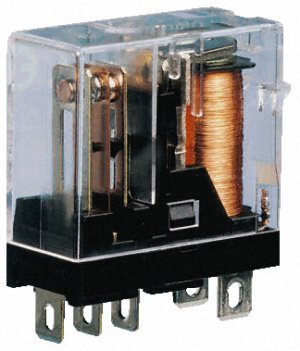
****

12V Lead- Acid Battery Wiper Motor





Wheels

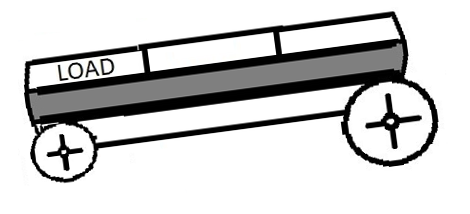


Water Drum

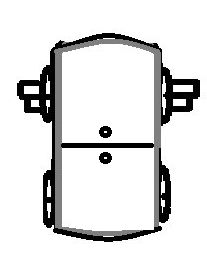
Relay

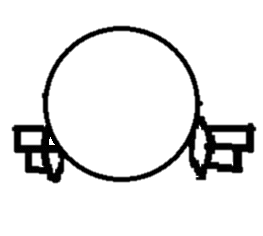
**METHODOLOGY**

**LAYOUT**



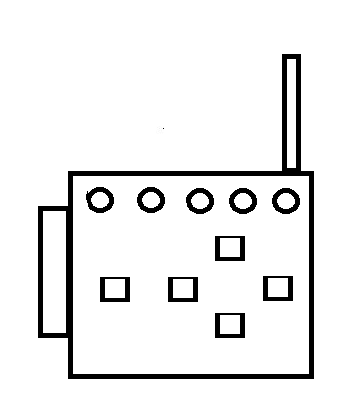
Side View







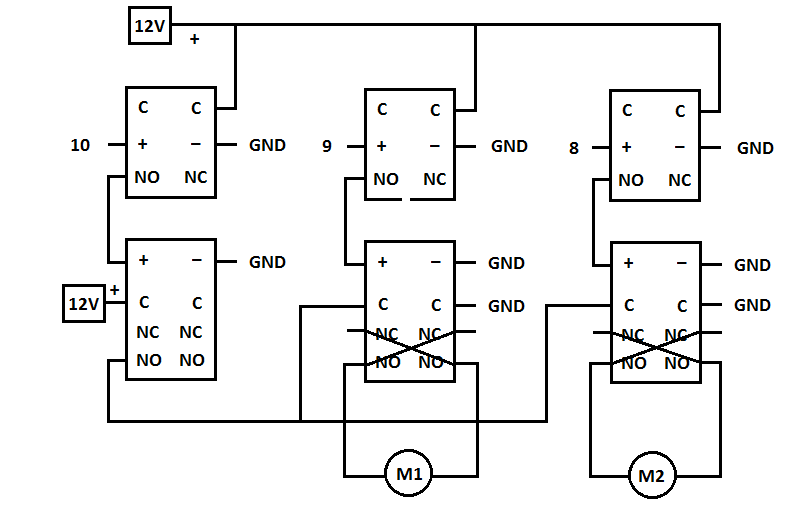
Top View Front View



Remote Control

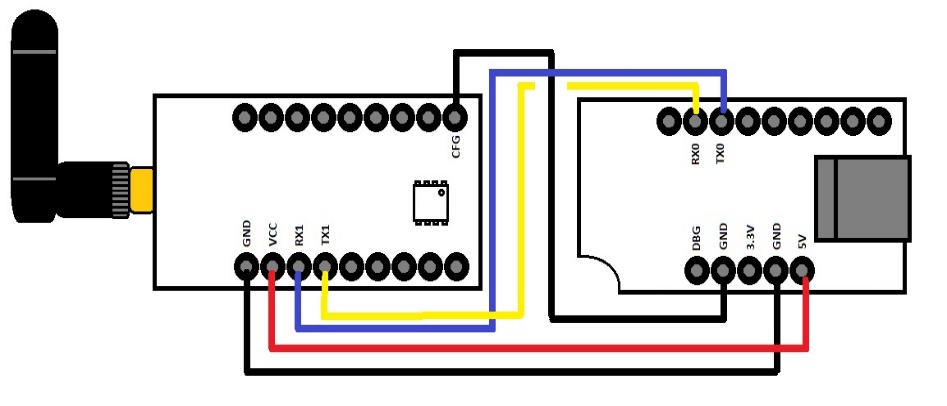
**PROCEDURE**

1. **Motor Connection**
2. Build the circuit using the schematic diagram below.

****

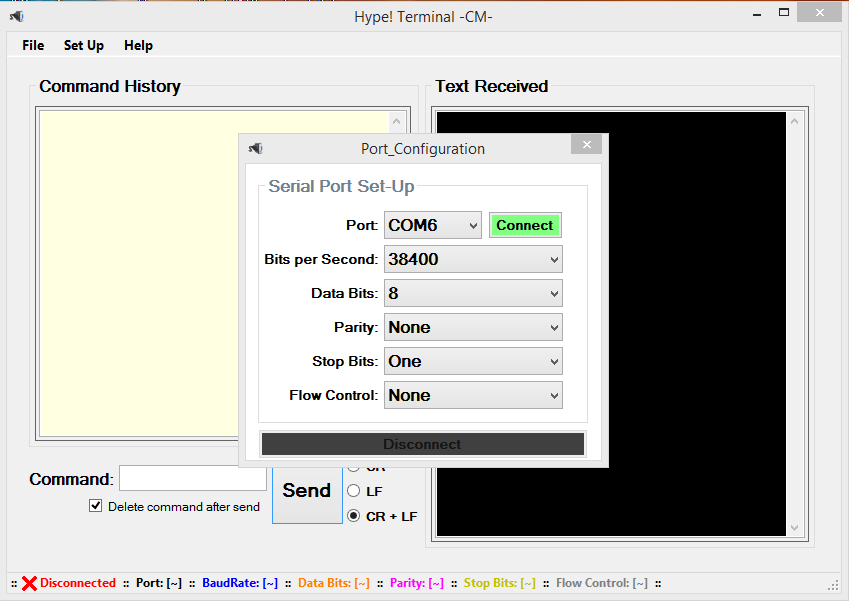
***\*Note***: 10, 9, and 8 are pins in the Gizduino board.

1. **Configuring the Zigbee**
2. Connect the RX0 port of the UART to the TX1 of the Zigbee, TX0 port of the UART to the RX1 of the Zigbee.
3. Connect the 5V port of the UART to the VCC of the Zigbee, GND of the UART to the GND of the Zigbee.
4. Connect the CFG port of the Zigbee to the GND port of the UART. (Refer to figure below)



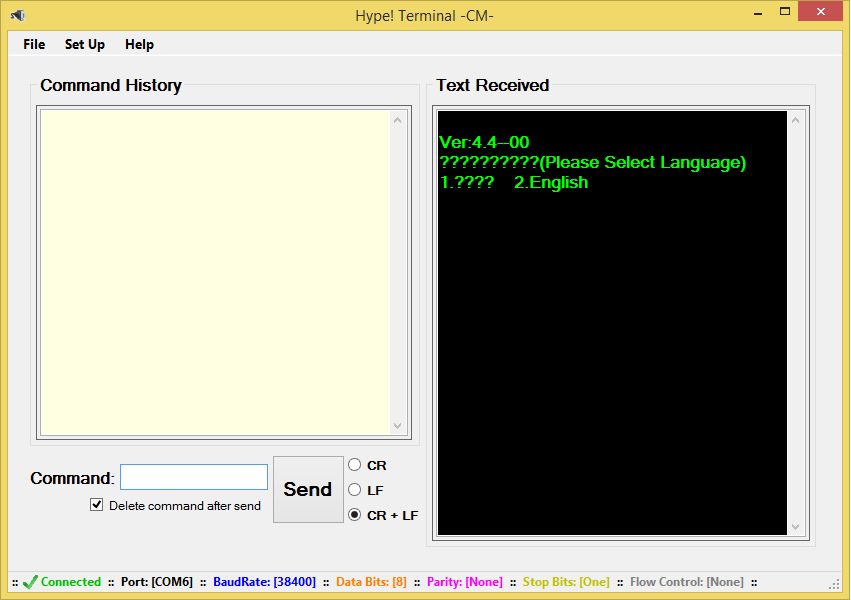
Zigbee configuration wiring diagram

1. Open the Hyper Terminal Application. Open the ***Set Up*** menu then choose the correct port where the UART is attached then change the ***Bits per Second*** to ***38400.***



Hyper Terminal Zigbee configuration

If the connection to the Zigbee has been established you will be asked for the language of configuration.

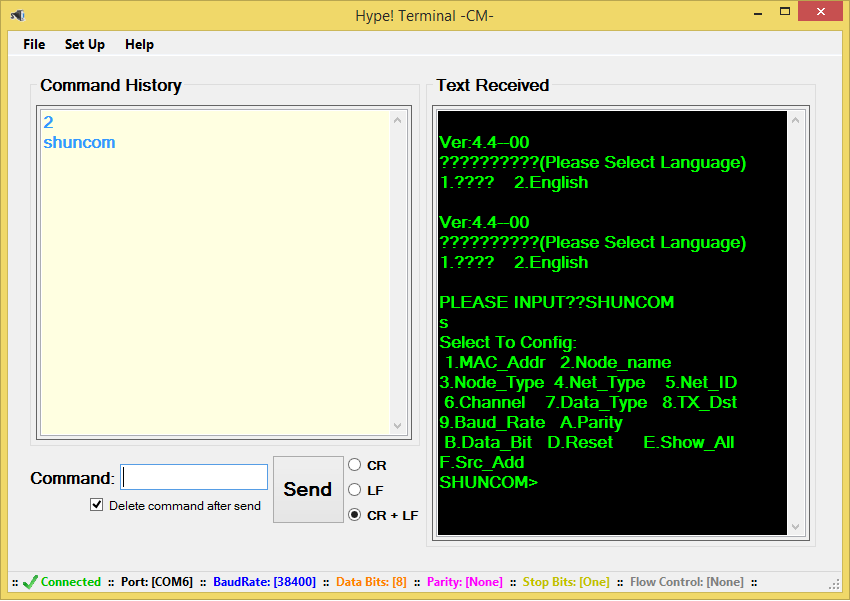


Hyper Terminal Zigbee configuration

1. Press “**2**” then enter to choose English. You will now be prompted to enter the word “**shuncom**”, after entering, you will now be able to configure the Zigbee.

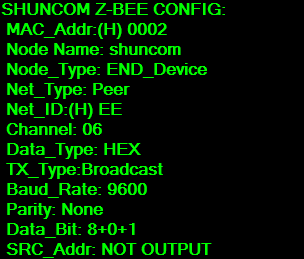
***\*Note***: - You can vary the font and the font size on the text received area on the ***Set Up*** menu

- ***Ctrl+E*** clears the text received area



Hyper Terminal Zigbee configuration

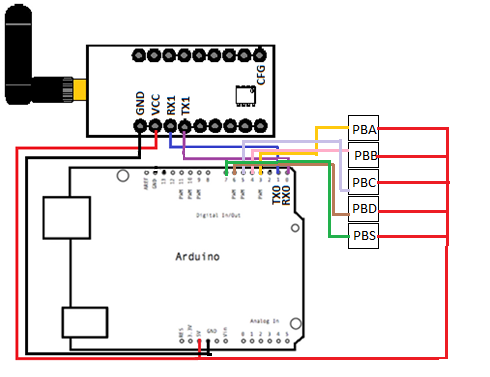
1. Configure to the set up shown



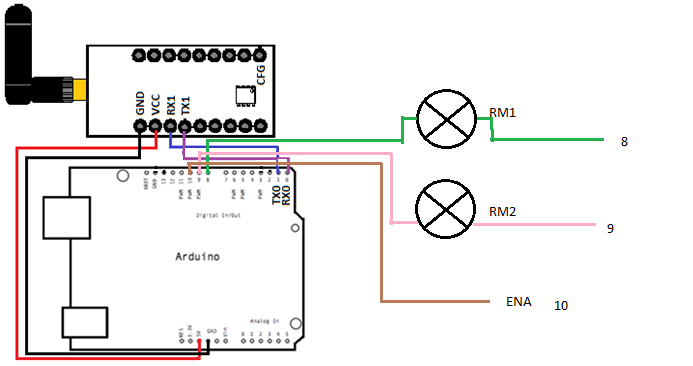
1. Configure the same set up to the other Zigbee.

***\*Note***: the *Mac Address* and the *Node Name* may be different for the Zigbees because they are irrelevant in our communication set up.

1. **Digital Control through the Zigbee**
2. ***Transmitter*.** Connect the Zigbee to the Gizduino with the TX of the duino to the RX1 of the Zigbee, RX of the duino to the TX1 of the Zigbee, 5V supply of the duino to the VCC of the Zigbee then the GND port of the duino and Zigbee.
3. Connect 5 push button to the pin 3, 4, 5, 6, 7 of the duino going to the ground port. (Refer to figure below)

****

1. ***Receiver*.** Connect the Zigbee to the Gizduino with the TX of the duino to the RX1 of the Zigbee, RX of the duino to the TX1 of the Zigbee, 5V supply of the duino to the VCC of the Zigbee then the GND port of the duino and Zigbee.

****

1. Upload ***Transmitter Program*** to the transmitter duino then ***Receiver Program*** to the receiver duino. Make sure both the sides are powered.

***\*Note***: The switch on the upper left of the duino allows upload of program but not the serial connection for wireless modules on the duino when the switch is directed inward. The switch directed outward allows the use of serial connection for wireless modules but you cannot upload programs. ***When uploading program the switch should be inward and when operating the Zigbee the switch should be outward.***

**CODES**

**//Transmitter Program**

**TRANSMITTER (REMOTE CONTROL)**

const int PBA = 3; //Forward Pushbutton

const int PBB = 4; //Reverse Pushbutton

const int PBC = 5; //Left Pushbutton

const int PBD = 6; //Right Pushbutton

const int PBS = 7; //Switch Pushbutton

int PBS1 = 0; // Forward Pushbutton State

int PBS2 = 0; //Reverse Pushbutton State

int PBS3 = 0; //Left Pushbutton State

int PBS4 = 0; //Right Pushbutton State

int PBS5 = 0; //Switch Pushbutton State

void setup() {

// put your setup code here, to run once:

Serial.begin(9600);

pinMode(PBA, INPUT);

pinMode(PBB, INPUT);

pinMode(PBC, INPUT);

pinMode(PBD, INPUT);

pinMode(PBS, INPUT);

}

void loop() {

// put your main code here, to run repeatedly:

PBS1 = digitalRead(PBA);

PBS2 = digitalRead(PBB);

PBS3 = digitalRead(PBC);

PBS4 = digitalRead(PBD);

PBS5 = digitalRead(PBS);

if (PBS1 == HIGH && PBS2 == LOW && PBS3 == LOW && PBS4 == LOW && PBS5 == HIGH) {

Serial.print(1);

Serial.println("Forward");

}

else if (PBS1 == LOW && PBS2 == HIGH && PBS3 == LOW && PBS4 == LOW && PBS5 == HIGH){

Serial.print(2);

Serial.println("Reverse");

}

else if (PBS1 == LOW && PBS2 == LOW && PBS3 == HIGH && PBS4 == LOW && PBS5 == HIGH){

Serial.print(3);

Serial.println("Left");

}

else if (PBS1 == LOW && PBS2 == LOW && PBS3 == LOW && PBS4 == HIGH && PBS5 == HIGH){

Serial.print(4);

Serial.println("Right");

}

else if (PBS1 == LOW && PBS2 == LOW && PBS3 == LOW && PBS4 == LOW && PBS5 == LOW){

Serial.print(5);

Serial.println("Do nothing");

}

else if (PBS1 == LOW && PBS2 == LOW && PBS3 == LOW && PBS4 == LOW && PBS5 == HIGH)

{

Serial.print(5);

Serial.println("Do nothing");

}

}

**//Receiver Program**

**RECEIVER (AMPHIBIOUS VEHICLE)**

const int RM1 = 8; //RM1

const int RM2 = 9; //RM2

const int ENA = 10; //ENA

int data = 0;

void setup() {

Serial.begin(9600);

// initialize the LED pin as an output:

pinMode(RM1, OUTPUT);

pinMode(RM2, OUTPUT);

pinMode(ENA, OUTPUT);

}

void loop() {

if (Serial.available()>0) {

data = Serial.parseInt();

if(data == 1){

// Forward

digitalWrite(RM1,LOW);

digitalWrite(RM2,HIGH);

digitalWrite(ENA,HIGH);

Serial.print(data);

Serial.println("FORWARD!!!");

}

else if (data == 2){

// Reverse

digitalWrite(RM1,HIGH);

digitalWrite(RM2,LOW);

digitalWrite(ENA,HIGH);

Serial.print(data);

Serial.println("REVERSE!!!");

}

else if (data == 3){

// Left

digitalWrite(RM1,LOW);

digitalWrite(RM2,LOW);

digitalWrite(ENA,HIGH);

Serial.print(data);

Serial.println("LEFT!!!");

}

else if (data == 4){

// Right

digitalWrite(RM1,HIGH);

digitalWrite(RM2,HIGH);

digitalWrite(ENA,HIGH);

Serial.print(data);

Serial.println("RIGHT!!!");

}

else if (data == 5){

// OFF ALL

digitalWrite(RM1,LOW);

digitalWrite(RM2,LOW);

digitalWrite(ENA, LOW);

Serial.println(data);

Serial.println("Do nothing");

}

}

}

**ACTUAL PICTURES**



Amphibious Vehicle without cover



Amphibious Vehicle with cover



Remote Control

**LAND TESTING**

****





****

**WATER TESTING**

****

****

****

****

****

**ECE184P-B11-GROUP1**