**EE175ABC Final Report Template**

**Vehicle to Intersection Communication**

**EE 175AB Final Report**

**Department of Electrical and Computer Engineering, UC Riverside**

|  |  |
| --- | --- |
| **Project Team Member(s)** | Lazaque Mugerwra, Chaoyun Ma, Nick Seal |
| **Date Submitted** | 04/17/2015 |
| **Section  Professor** | Ping Liang |
| **Revision** | Revision 1.0 |
| **URL of Project Wiki/Webpage** | http:// |
| **Permanent emalis** |  |

**Summary**

This report presents the concepts of what?, experiments, and results of the construction of the scale intersection, the 3 experimental cars, and the intersection interaction system. Also included are the problems encountered, and the solutions provided, as well as the difficulties we had, and what we could and could not fix.

**Revisions**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Description of Version** | **Author(s)** | **Date Completed** | **Approval** |
| 1.0 | Initial version of the template, filled in with vital information from the project, as directed by the template. | Nick Seal,  Chaoyun Ma,  Lazaque Mugerwa | 04/17/15 |  |

**Table of Contents**

[**Revisions**](#h.gjdgxs)

[**Table of Contents**](#h.30j0zll)

**1 \* Executive Summary**

[**2 \* Introduction**](#h.3znysh7)

[2.1 \* Design Objectives and System Overview](#h.tyjcwt)

2.2 \* Backgrounds and Prior Art

[2.3 \* Development Environment and Tools](#h.4d34og8)

[2.4 \* Related Documents and Supporting Materials](#h.2s8eyo1)

[2.5 \* Definitions and Acronyms](#h.3rdcrjn)

**3 \* Design Considerations**

3.1 \* Assumptions

3.2 \* Realistic Constraints

3.3 \* System Environment and External Interfaces

[3.4 \* Industry Standards](#h.2jxsxqh)

[3.5 \* Budget and Cost Analysis](#h.z337ya)

[3.6 \* Safety](#h.3j2qqm3)

[3.7 Performance, Security, Quality, Reliability, Aesthetics etc.](#h.1y810tw)

[3.8 \* Documentation](#h.4i7ojhp)

3.9 Design Methodology

3.10 Risks and Volatile Areas

[3.11 \* Understanding of Professional and Ethical Responsibility](#h.2xcytpi)

[3.12 \* Global, Economic, Environmental and Societal Impact](#h.1ci93xb)

[3.13 \* Contemporary Engineering Issues](#h.3whwml4)

[3.14 \* Recognition of the need for and an ability to engage in lifelong learning](#h.2bn6wsx)

[3.15 \* Importance of Team Work](#h.qsh70q)

**4 Experiment Design and Feasibility Study**

[4.1 Experiment Design](#h.1pxezwc)

[4.2 Experiment Results and Feasibility](#h.49x2ik5)

[**5 Architecture**](#h.2p2csry)

5.1 System Architecture

[5.2 Rationale and Alternatives](#h.3o7alnk)

[**6 \* High Level Design**](#h.23ckvvd)

[6.1 Conceptual View](#h.ihv636)

[6.2 Hardware](#h.32hioqz)

[6.3 Software](#h.1hmsyys)

[**7 Data Structures**](#h.41mghml)

[7.1 Internal software data structure](#h.2grqrue)

[7.2 Global data structure](#h.vx1227)

[7.3 Temporary data structure](#h.3fwokq0)

[7.4 Database descriptions](#h.1v1yuxt)

[**8 \* Low Level Design**](#h.4f1mdlm)

8.1 Module i (for i = 1 to n)

[*8.1.1 Processing narrative for module i*](#h.19c6y18)

[*8.1.2 Module i interface description.*](#h.3tbugp1)

[*8.1.3 Module i processing details*](#h.28h4qwu)

[**9 User Interface Design**](#h.nmf14n)

[9.1 Application Control](#h.37m2jsg)

[9.2 Screen 1..m](#h.1mrcu09)

[9.3 Development system and components available](#h.46r0co2)

[**10 Administrative and Other Design Issues**](#h.1664s55)

[10.1 \* Project Management](#h.3q5sasy)

[10.2 Requirements traceability matrix](#h.25b2l0r)

[10.3 Packaging and installation issues](#h.kgcv8k)

[10.4 Design metrics to be used](#h.34g0dwd)

[10.5 Restrictions, limitations, and constraints](#h.1jlao46)

[**11 \* Experiment Design and Test Plan**](#h.2lwamvv)

[11.1 \* Design of Experiments](#h.111kx3o)

[11.2 \* Bug Tracking](#h.3l18frh)

[11.3 \* Quality Control](#h.206ipza)

[11.4 Performance bounds](#h.4k668n3)

[11.5 \* Identification of critical components](#h.2zbgiuw)

[11.6 \* Items Not Tested by the Experiments](#h.1egqt2p)

[**12 \* Experimental Results and Test Report**](#h.3ygebqi)

[12.1 \* Test Iteration 1](#h.2dlolyb)

[*12.1.1 Experiment 1*](#h.sqyw64)

[*12.1.2 Experiment i (repeat for i = 1, …., n)*](#h.3cqmetx)

[12.2 \* Test Iteration j (repeat for j=1,…m)](#h.1rvwp1q)

[*12.2.1 Experiment 1*](#h.4bvk7pj)

[*12.2.2 Experiment i (repeat for i = 1, …., n)*](#h.2r0uhxc)

[**13 \* Conclusion and Future Work**](#h.43ky6rz)

[13.1 \* Conclusion](#h.2iq8gzs)

[13.2 Future Work](#h.xvir7l)

[13.3 Acknowledgement](#h.3hv69ve)

[**14 \* References**](#h.1x0gk37)

[**15 Appendices**](#h.4h042r0)

# \* Executive Summary

Our system was designed to simulate a scale model of an intelligent autonomous intersection, specifically for the proof of concept. Initially, our project was designed with 3 large cars, what was meant to be a 4-way intersection, and have each car only move in straight lines. Each of our cars was suppose to simulate an autonomous vehicle, taking the human element out of the equation. This would make it possible to have a system where turning cars was not a factor, and we would be able to coordinate when the cars moved through the intersection.

Besides the 3 autonomous vehicles, we had a tower attached to a xbee communication device, which would allow our tower to coordinate with the 3 cars in the system, telling each individual car when to speed up, slow down, or stop. The tower would sense where each car was, by using ultra-sonic distance sensors, which would be factored into the towers’ calculation of which car needed to speed up/slow down. Additionally, each car was equipped with speedometers which allowed the cars to send the tower their own speed. This was also used in the calculation of the new speeds, which allowed the tower to be the “brain” of the system.

In short, the tower has attached ultrasonic sensors that detect the distance of each car in their respective lane. Additionally, each car has a speedometer that allows each car to tell the tower their own speed. After gathering all the information for each vehicle, the tower will determine if a collision is likely, and coordinate an anti-collision scenario. This is the design of the system, and is capable of producing an anti-collision system accurately, if the input information is correct.

Our system also suffers from a need to coordinate effectively, in a time sensitive manner. Due to the dynamics of our system, our tower will wait a short period after telling a specific car to increase or decrease speed, followed by a re-evaluation of the new information, to keep the most relevant information available. If no new speed is required, the system will actively re-evaluate the most current information, since ideally the system should try to keep each car from colliding, and thus keeping the current trajectories of each vehicle is better than having them change, into collision potential vectors.

In short, the first priority of our project was the creation of a scale model of an intersection, so that we could generate a proof of concept with a successful trial. After this, our priority was to streamline our system, as to make a successful trial more repeatable.

# \* Introduction

*With the project generating an autonomous system, which coordinates which vehicle should move before which other vehicle, there were no other projects that were involved with the development of our own system.*

## \* Design Objectives and System Overview

Overall, the system is a model representation of the 4-way intersection that allows the autonomous vehicles to coordinate when each car will occupy the intersection without drivers. In addition, there is the element of the system needing to be capable of calculating the new speeds dynamically, in order to keep up with the system itself.

The system is devised into 2 categories, the Tower, and the Cars. Each Car is equipped with an Xbee series 1 connected to an Arduino Uno, which allows communication. Each Car is also equipped with a hall effect sensor system, utilizing a magnet attached to the left rear tire, with a detector mounted to the side of the car. The Tower is equipped with an Arduino Mega, and an Xbee series 1, in order to communicate with each car. Additionally, the Tower is connected to 6 individual Ultrasonic Distance Sensing modules, with 2 of the sensor cascaded at 2 meters apart in 3 of the 4 lanes. This allows the Tower to see how far away the cars are at any given time.

You must provide a list of quantitative technical design objectives, e.g., accuracy of 95%, sensor range of 20 feet, 90% successful retrieval or recognition, mean square error < 0.1, response time < 1ms, dynamic range of 20Vp-p, bandwidth of 20MHz, transmission range>100m at 100mW, SNR>10dB, data transfer rate = 100Mbps, costing less than $500, etc.

The accuracy of our sensors must be within 85% accuracy at 4m, with a response time of less than 20ms. Our communication needs to occur within 100ms, in order to keep our system fast enough to keep up with the cars, which should average about 1m/s, but be capable of at most 2m/s. Our entire budget cannot exceed 300 dollars, since that is all we have to put forward toward the project.

Design of the Car system, and the implimentation of the speedometer: Lazaque

Digital communication and overall systems integration: Nick

Design of Tower, and distance sensing: Chaoyun

## \* Backgrounds and Prior Art

Scale representations of what we were attempting to develop have occurred in various places around the globe, usually utilizing different hardware for the various tasks required. In real sized experiments, vehicle to vehicle communication and vehicle to intersection communication is accomplished utilizing DSRC, which is much faster than our Xbee modules, however a module for DSRC would be much larger than our cars could support, as well as being extremely expensive. The cheapest modules we were able to find were more than 1,500 dollars. This forced us to find alternative methods for initiating the communication aspect. As for our sensing equipment, we found radar systems that would be able to measure the distance of our cars from the tower, without the need for cascaded ultrasonic sensors, however these also proved to be too expensive for our budget.

## \* Development Environment and Tools

The design environment from the coding perspective took place exclusively in the Arduino terminal provided by the Arduino client. This is where we not only designed our code, but tested how each file would work in regards to our cars. The testing environment for our hardware consisted of the lab rooms in the engineering building, the cars themselves, the tower structure, and our computers. Our tools included standard tool kits, assorted electronics, and computers, Arduino unos, an Ardunio mega, and the Arduino software.

## \* Related Documents and Supporting Materials

No reference required.

## \* Definitions and Acronyms

Beyond the definition of the Tower and the Cars as separate systems that require different amounts of hardware, there are no Acronyms or other definitions that were created in the design and construction of this project. Isn’t Xbee an acronym?

# \* Design Considerations

This section describes issues that need to be addressed or resolved prior to or while completing the design as well as issues that may influence the design process.

## \* Assumptions

For our project, we assume that the 3 cars will be approaching the intersection with the intent of moving straight. We need to assume that the cars will be controlled by autonomous systems that are capable of interacting with our intersection. We assume the roads leading to the intersection are straight, and that each car is moving at a similar speed to each other, to simulate a realistic speed limit.

## \* Realistic Constraints

Cars must move straight, the battery system must be able to manage all of the power that the circuit draws, as well as the power required to communicate between the tower and the cars. Additionally, the sensors must be angled as to provide the best view of the lane, in order to generate a realistic intersection interaction.

## System Environment and External Interfaces

The cars must be able to move straight enough for the tower to identify where the cars are, as well as communicate back to the tower. The environment must be big enough to support the lanes, the towers, and the start-up distance, which allows the car to reach their cruise velocity.

## \* Industry Standards

Zigbee’s Xbee series 1 standards: 868MHz, uses an 802.15.4 network, which worst case with a 16-bit Unicom, should be roughly 40 ms.

HC-SR04 standards: best results from testing gives a 10ms delay to pulses, gives an accuracy of about 90 percent at 2 meters. To keep to this standard, we cascaded our sensors, with 2 in each row.

The circuits onboard our cars utilized a high start-up current to initialize our DC motors, but needed to be stepped down to keep our microprocessors from frying. As such, our system was unable to support a battery charge of less than 9V. Furthermore, anything over around 12 V caused our voltage regulator to heat up dramatically.

## \* Knowledge and Skills

Describe the knowledge and skills required by this project. Complete the following for each team member:

Nick: EE 150 Digital Communications, Introduction to Robotics, CS 128 Digital Control

I learned how to program the Arduino to interact with the Xbee, as well as how to code the generation of the pulse that would set the car speed. Additionally, I learned how to integrate the software and hardware together, as well as generate a fully functional system, by combining various parts of the project that had been designed independently, and fit them together to make the system work.

Chaoyun: EE 150 Digital Communication, Embedded System, Digital Circuits,

I learnt:

• how to program Arduino

• how ultrasonic sensors work and how to integrate them with Arduino

• how to combine hardware with software so that the whole system would work

• how to design and test components individually

• how to design and test the system when each components work independently

Lazaque: ??

## \* Budget and Cost Analysis

HC-SR04 x 10 = $15, 4 x Xbee = $80, 3 x RC Car = $60, 1 x explorer board = $15,

Assorted electronics (resistors, capacitors, etc.) = $30, assorted lumber = $50,

Arduino Uno x 3 = $100 3 x Speedometer = $45

Total Costs: $395

## \* Safety

The considerations of the safety required is minimal in this project, due to the unlikelihood of a failure resulting in any sort of damage to a human being. All of our battery systems are too low voltage and current to harm any human, and if the cars were to hit a human running at full speed, it would most likely be more of an inconvenience for the car than the person. That being said, each functional lane has wires extending to the sensors mounted on the bridges, which does present a tripping hazard to those who would walk around the intersection without looking.

## Performance, Security, Quality, Reliability, Aesthetics etc.

Initially, almost every part of the project was designed with the scope of solving the problem at hand. After each part proved successful, we determined if it provided enough of a success to cover our basic needs of whether or not it could perform to the standards of our system. If so, then our system was in need of a different piece of hardware.

## \* Documentation

As updates for our project came into focus, we found that it was easier to record what ideas we had, so that we could put them into practice, and evaluate our results. In short, we would put the ideas we wanted to implement into our shared Google drive folder, in order to preserve our systems progress.

## \* Understanding of Professional and Ethical Responsibility

The implication of our work ranges from the private sector, where the option of autonomous vehicles are more of a factor, to hopefully the masses. The idea, similar to the electric car, is that hopefully as the technology becomes more and more prominent, more and more people will get behind the idea, until it is widely available for everyone. It is at this time, where our system would have the most implication, being used hopefully in cities across the world, replacing traffic lights and round abouts all together. Of course, the safety requirements for this kind of task would need to be strictly enforced, since the vehicles will be carrying people all around. This is the major problem with the system, requiring an extremely reliable base system that could guarantee a certain level of safety when transporting people from one place to another. Today, an intersection is the most likely place for an accident to occur in most modern cities. To take the human driver out of the equation could prove difficult for some people to accept. However, if the autonomous cars are intended to maneuver these intersections efficiently in large numbers, it would be much safer to have the intersections themselves coordinate which car will move when.

We addressed this system by attempting to recreate the scale system, in a way that would provide the real world scaled system to be able to exist. In short, we looked at technologies that exist already, in order to make sure that our system is capable of supporting a real-world equivalent. We also ignored factors that were only issues to our system, and would not be present in a real-world equivalent. This was because issues that would not be present in a real-world equivalent should not be considered extremely critical problems for our system to solve. Our system should focus on proving that the real world could generate an autonomous car-intersection interaction system that would be able to coordinate each car to ensure a safe interaction. Finally, the real world equivalent needs to address some of the issues that we ignored in our system, in order to prove the communication and timing of the system was possible. For example, for our system, we assumed each car was going straight through the intersection. In a realistic example, the cars should be able to communicate to the intersection which direction they will be heading. This way, the intersection can address more realistic situations, where drivers need to head in different directions.

After trying to keep the separate cars from hitting each other, it is apparent that the system is incapable of providing the safety that the real-world equivalent would need to ensure the lives of those who would ride through the intersection in the autonomous vehicles. However, the cars in our system are in need of special augmentation, in order to provide insurance that they will move in straight lines. Another feature that I would install on the real world vehicle would be a sensor system that forces the car to stop if it gets too close to another vehicle. In short, I’ve learned that if our system were real-world prototypes, we would need to increase the safety of the entire system, so that the human passengers would have some level of guaranteed safety.

Every team member contributed to this report

## \* Global, Economic, Environmental and Societal Impact

If the concept of our design project becames a commercial product, and the safety regulations were high enough to keep the passengers safe, then the users would benefit from some of the first hands-free driving systems. This means that not only would the cars be able to drive in traffic without driver interaction, it would be able to coordinate with intersections and cars that the human driver could never have perceived. This benefit not only makes the interaction at the intersection level more efficient for all vehicles, but it also makes the passengers in the cars safer as a whole. This is because the intersection understand where all the other cars are, and to what speeds they will be accelerating. In other words, the intersection can understand factors that a human driver would not be able to understand. That being said, our project has many potentially unethical side effects. The most glaring would be that the system would literally be responsible for the life/death of the passengers of the vehicles. This is where strict safety regulations would come into play, making it difficult for many people to not only trust these autonomous vehicles, but also slow to allow any level of autonomy into the vehicles themselves. Finally, besides the obvious safety issues, the people who were resistant to the idea of letting cars drive themselves would push for the system to not be implemented. I believe this is because of the curve that follows technology, it would not be until the benefits far outweighed the stubborn misinterpretations that many technology fearing people would have.

More recently in the world, there was the development of electric cars, and the autonomous car from various existing car companies. It was theorized that these kinds of self-driving cars would be implemented more commonly only when it became a cheaper alternative to the existing cars, since the standard set by today's society would be unwilling to change. It would not be until the obvious benefits pushed for the masses to make driving an electric smart car necessary that we would see the development of everyone using self-driving cars.

The technology exists for the vehicles described in our report, and that work in model form in our system to be driving around today. The only reason they would face such harsh adversity, would be the hesitation of people to climb into an autonomous vehicle, and hand control over to a self-driving car. As such, the social impact of the project is hard to determine at this point, since it would most likely be a long time before it was popular enough to warrant being used all around the world. Ideally, however, there would be a high demand for this kind of work in the future, since the technology to make vehicles autonomous exists, and the creation of smart intersections could help avoid collisions in major cities all around the world, as well as provide more efficient solutions for traffic sequencing than simple lights based on timers/sensors.

Every team member contributed to this report.

## \* Contemporary Engineering Issues

In early March, the car company Mercedes debut a version of its autonomous car in San Francisco, demonstrating that their car was capable of following the directions of human driving rules, and did not need to coordinate with any other cars, or with any intersections. This is extremely representative of our system, since our system focuses more on the communication between multiple smart vehicles and the intersection. In short, as the world turns more and more toward smart vehicles, the need for intersection interaction becomes more reliable in terms of profitability. In short, the more attention autonomous cars generate, the more likely they will be used, which would lead to more and more smart vehicles on the road. The end result of this would be that the coordination between the cars and an intersection might seem more likely, in situations with dangerous interactions and high density traffic. The obvious target would be any large scale city on earth, making it extremely likely that our product would be something that would be interested in being developed further, especially with the potential for so many different intersections with as many more smart cars looking for direction.

Despite the concept of smart cars being something that many people would not be willing to put their lives in at the moment, this is understandable, since there must be a certain threshold of safety before someone is willing to put their lives in the hands of a robot. The accuracy of the system must be airtight enough to quell these doubts. Today, people trust robots to do menial tasks, with a few exceptions, such as complex surgeries, which still require putting humans on the controls. This project, however, would remove the human element completely from behind the wheel, which could potentially yield a much safer driving environment. As the demonstration of the vehicles’ potential convinces more and more individuals, the trend would slowly result in the vast majority of people beginning to trust autonomous cars. As technology begins to make our lives easier and easier, it may be possible for humans as a group to admit that autonomous vehicles, as well as smart intersections are much more capable of understanding the environment than we are, especially in terms of how fast each car that could potentially have an accident is traveling.

Besides the social aspect of the technology we worked with, there is very little new information regarding what is feasible in our system. The sequencing of the cars does not require us to communicate in a way that has not been done before, and the way we sense where the cars are is not feasible in the real world using the technology we currently use, in fact using a radar systems and technologies that currently exist would be much better, even in scale form. They could also be used to verify speed, by taking multiple readings, at a faster speed than our original system was capable of making one reading. Also, the computational power of our Arduino Mega would be replaced in the real world by something with a much better processor. In short, the real world equivalent would be a much more efficient, and powerful system, even in scale. With these sorts of real world technologies, including DSRC(Designated Short Range Communication), a system of communication practically built for vehicle to vehicle communication, and capable of communicating with intersections, I am sure that our system in real life would be significantly safer for those operating the cars.

Every team member contributed to this essay

## \* Recognition of the need for and an ability to engage in lifelong learning

As the project began, I realized that in order to not only be useful to the team, but in order to keep myself up to date on the technologies that were being introduced at all times, I needed to keep learning. I would not be able to say I was absolutely sure that anything was set in stone, or unable to be different than the way I perceived. I, like many people, would be at my best when I am capable of dynamically learning what needs to happen next, as well as what each component is capable of. I do not believe that if I would continue to grow as a human being if I did not continue to learn as I got older. I don’t believe that this is necessarily something that applies only to the technical aspects of the system at hand, or even in the future projects I will work on, but in all areas of life. I believe that open mindedness toward learning new things is something that would make the world a little bit of a better place. Anyway, in the technical domain, keeping current with the new technologies that are emerging is something that almost every engineer would need to do to keep as sharp as possible. It is not necessarily a need that is not natural in the field of engineering, however. Most engineers, myself included, are curious about new technologies that we hear about, and have no idea how they work, but we strive to learn for ourselves.

Every team member contributed to this essay.

## \* Importance of Team Work

This project helped me understand every aspect of teamwork, how to design several parts, and come together to make everything mesh perfectly. This project taught me that teamwork can be stressful at times, relying on others to complete parts of the project that I am not responsible for, but can also be very rewarding, like when everyone comes together to work all night on a particularly nasty problem. It highlighted how important keeping tempers under control could be, and how to divide the work evenly, especially when others felt as though it would seem awkward to share their personal feelings.

Every team member contributed to this essay.

# Experiment Design and Feasibility Study

Include this section if you need to conduct experiments to evaluate the feasibility of your project ideas, alternatives, trade-offs and realistic engineering constraints, and to answer key design questions, such as what parts to use, how to collect data, whether accuracy of sensor is sufficient for the design objective, whether battery provides enough power, what hardware or software interface methods to use to connect the different modules, etc.

## Experiment Design

Early experiments with our software and hardware were meant to generate feasibility, and functionality of our system. To begin with, our system needed to be able to support the operations that we would require of our vehicles and our tower. This meant we needed to see how functional our communication and cars were. As a result, our early tests included hooking up the car to an Arduino generating a PWM, and generating a PWM on Arduino A from a signal sent to it by Arduino B. There was very little other hardware at this point.

Lazaque was in charge of getting the cars to run, Nick was in charge of getting the Tower to talk to the Cars, and Chaoyun was in charge of getting the Tower to see the Cars.

As time went on, the Tower began to take form, first as a platform with basic sensing, followed by a fully functional system that was able to regulate speed of each car. At this point the experiments became more focused on the system as a whole, seeing if the Tower could control each car individually, seeing if each car could respond without interfering with the other cars, and finally seeing if the tower was fast enough to detect the cars, and talk to the cars without confusing any of the cars for each other.

Nick and Chaoyun were in charge of getting the Tower to work with the Cars, in order to make the system functional.

## Experiment Results and Feasibility

As the experiments continued, we believed that our results indicated that our overall system was feasible. This conclusion was a result of our cars moving fast enough, and our communication system was fast enough for our system to work. The conclusion was the consensus of all of the team, with each person representing their own part of the overall project.

# Architecture

The architecture provides the top level design view of a system and provides a basis for more detailed design work. These are the top level components of the system you are building and their relationships.

## System Architecture

The development of each system was as follows:

Nick worked on communication and systems integration

Chaoyun worked on Ultrasonic Distance Sensing and systems integration

Lazaque worked on the RC Cars, and the speedometers

The system worked generally in the following way:

The tower senses how far away the Cars are, determining each car’s distance from the intersection. After the distance is recorded, the Tower then interrogates each car for their speed. The cars respond, and the Tower then calculates the time until each car will enter the intersection. After determining if a collision is imminent, the Tower will then either inform each car its new speed, or will return a no collision indicator, and re-evaluate to see if the situation has changed.

## Rationale and Alternatives

The approach we decided on was developed this way due to the time constraints we had, as well as the cost analysis of purchasing more effective modules, at a much higher monetary value. We believed the problem could be solved without the need for a more costly system. Additionally, we stopped several systems from being implemented, since we did not want redundancy in the system. Finally, as our alternatives seemed too expensive, many of the more affordable systems were not as available to us, due to our student status.

Chaoyun was responsible for the Ultrasonic Distance Sensors, and the system integration.

Lazaque was responsible for the Cars, and the Speedometers.

Nick was responsible for the Communications, and the system integration.

# \* High Level Design

This section describes in further detail elements discussed in the Architecture. Normally this section may be split into separate documents for different areas of the design.

High-level designs are most effective if they attempt to model groups of system elements from a number of different views. Typical viewpoints are:

a. Conceptual or Logical: this is the view most often used in Section 3. This view shows the logical functional elements of the system. Each component represents a similar grouping of functionality. For UML, this would be a component diagram or a package diagram..

b. Hardware: this view is for hardware functional blocks and how they interface.

c. Software: this view is the software view of the system. The components are modules, threads, processes or distributed applications.

d. Security: this view typically focuses on the components that cooperate to provide security features of the system. It is often a subset of the Conceptual view.

For many smaller applications, the conceptual view is all that is necessary. Document those views that will help you design and implement the system. If you have only a single view, and that view is discussed adequately in section 3, then this entire section can be deleted.

State clearly who is responsible for which module/task

## Conceptual View

Provide a description and diagrams of a system element or set of elements that describes a clearly defined view or model of the entire system or a subset of the system.

From a conceptual standpoint. we needed three systems to work together. Sensors (Chaoyun), Wireless Communication (Nick) and RC Car Design (Lazaque). If any of these systems did not work, our project would not work or would otherwise need major remodel. Sensors had to be capable of reading 4 meters in one direction. RC Cars had to be able to changed speed and communicate this change in speed via wireless communication.

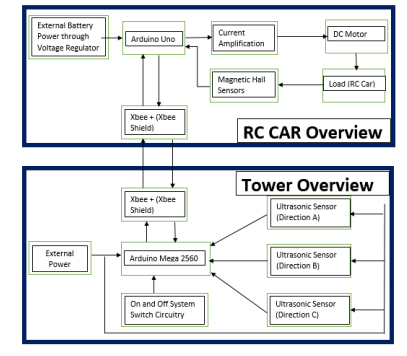
Lazaque was responsible for the circuit design which powered the motor. Lazaque chose the appropriate batteries for the project. He also wrote the code preliminary code that allows the car to change speed and record its speed.

Chaoyun integrated the sensors. Chaoyun implemented the mechanisms which stacked the sensors together extend the reliability range of the stacked sensors to 4 meters. Chaoyun measured reliability ranges of all the sensors. Chaoyun also worked with Nick to implement wireless communication between the Tower and RC Cars,

Nick was responsible for the Wireless Communication. This means learning everything about the Xbees. Speed of transmission, bandwidth of transmission. How to send information wireless from one node to another while avoiding interference. Nick also wrote the algorithm which prevents the cars from colliding with each other. He also implemented the Push Button start circuitry which allows the project to have power on and power off control.

## Hardware

The block diagram shown below shows a breakdown of the hardware.



The Block above shows an overview of the hardware.

## Software

Chaoyun Ma was responsible for the interfacing with Ultrasonic Sensors.

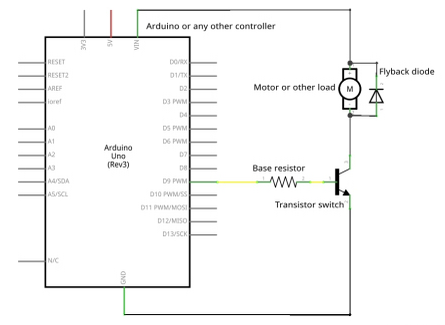
Nick was responsible for overall algorithm that prevents cars from crashing into each other.

Lazaque wrote the code to measure RC Speed and power the RC Car with PWM Signal from the Arduino Uno Microcontroller.

# \* Low Level Design

This section provides low-level design descriptions that directly support construction of modules. Normally this section may be split into separate documents for different areas of the design. For each component we now need to break it down into its fundamental units or modules. Each module or block may be hardware or software or a subsystem implemented using hardware and software

## The Circuit Amplifier Module

Lazaque did research and implemented the circuit below.

### Processing narrative for The Circuit Amplifier module

The diagram above shows the circuit amplification Schematic.

### The Circuit Amplifier Module interface description.

Input into the current amplifier circuit goes into Base Pin of the TIP 31 Transistor. The TIP’s Collector biases the DC Motor. The DC Motor is biased by both the TIP 31’s Collector and External 5 V (Vcc). A diode is connect in parallel with the DC motor to prevent any back current from destroying the Arduino.

### The Circuit Amplifier Module processing details

The DC motor requires 200mA initial start-up current and 1 Volts minimum voltage. The Arduino PWM Pin 9 can provide upto an average of 5 Volts. However the maximum current from an Arduino uno is 40mA. Far shorter than what is need. The Circuit above with transistor TIP31 has a gain of 10. The Arduino Uno cranks 20 – 30 mA with the PWM Signal. This current is then multiplied by 10. To about 200 mA. Enough to start up the DC motor and keep it running.

# \* Test Plan

## \* Design of Tests

Design tests to verify whether your design meets the design objectives. Design experiments to verify the functions of your modules and the prototype you built. The set of tests, which constitute your test plan, are to be specified, including as much detail as is possible at this stage.

**Tests and expected responses**

Specify the test and the expected results.

Test Case 1

1. The Objective of the test is to see if 3 cars can successful go through an intersection without crashing into each other. 3 Cars all have the same starting speed. However they a placed a variable distances. When an On Signal is detected wirelessly, Cars start moving towards the Tower.
2. Three Cars were set up at variable distances. Each distance at least 1.5 Meters from the tower.
3. Once On Signal is detected. Cars should start moving at preprogrammed speeds. Once Speeds have been communicated. Tower will decide which car can use an intersection or which car should slow down to avoid collision.
4. One or two cars should be asked either to speed up or slow down if there is a probable collision. If there is not probability for a collision then there won’t be any new speed commands given to the car.

State clearly who is responsible for which experiment

## \* Bug Tracking – Experimental Problems

The Biggest factor that affects reliability is inability of the cars to go in a straight direction. With the cars being unable to go through the intersection it is difficult to see whether or not experiment works. However we did manage on occasion. Maybe 1 out of 10 runs, to get at least 2 cars going through the intersection with other 3rd car slighting veering off the path with .5 meters of the tower. With 2/3 Cars having gone through the tower, we can at least get some understanding on whether or algorithm works or not based on how close the cars came to NOT colliding with each other.

## \* Quality Control

The computing algorithm is designed to find out based on speed and distance if any of the cars will be in the tower area within at least .5 seconds of each other. For a car travelling at 1 meter per second, the distance under the tower is .5 meters. This gives the car travelling at 1 meter per second a minimum of .5 seconds to clear the tower. This is what the tower calculates and looks for. For some of the successful trails we hard, one car asked to slow down while the other 2 were allowed to maintain their current speeds. Only one car is asked to slow down because if its 3 cars travelling towards each other, 2 of the cars won’t collide into each other because they are in opposite lanes, and ideally one car would not end up in another cars lane if the cars travelled straight and did not veer off into other lanes.

At other instances, because of the distances where the cars were placed, there we no new speed commands given to the car as the tower didn’t find zero to low probability of collision.

## Performance bounds

Special performance requirements are specified.

## \* Identification of critical components

Batteries. If Batteries are not fully charged to 10 Volts, we always had problems getting the cars started.

Voltage Regulators: Sometimes due to running time. The Voltage Regulator Heat Sinks would get hot, and therefore it would necessitate that we wait a while before running the RC Cars again.

On and Off Switch Signals. Sometimes, due to interference. The Wireless Xbees would not receive an   
On Signal when they should have. We solved this by always waiting about 5 seconds once all systems had been powered on.

## \* Items Not Tested

List any functions or processes not being tested and why.

Temperature. The Reliability and accuracy of the Ultrasonic Sensors depends on temperature. Without instantaneous variations in room temperature which is pretty constant. Temperature was not a huge deciding factor in Ultrasonic Measurement accuracies.

# \* Test Report

Carry out the tests designed in the section above to test your modules and prototype and present the results. Present the results of the tests and provide an analysis of the test data.

State clearly who is responsible for which test case

## Cars

### 11.1.1 Motor Control & Power System

By Lazaque Mugerwa.

1. Cars can accelerate and decelerate under Arduino’s control. The max speed is around 120 cm/s, with max PWM of 255. Cars won’t start until PWM reach 80. This part was conducted by Lazaque Mugerwa.

2. Our expected max speed ranges from 100 cm/s to 200 cm/s. We also expect various car speeds. This part meets our expectation. The power system can support cars running at full speed for about 5 min, and moderate speed for about 15 min.

3. ‣ The max speed of cars is around 120 cm/s. The batteries’ power affect car speed a great deal.

‣ Cars won’t start unless PWM is larger than 80. Motors we use need a large current to initialize. Also this feature affects batteries’ duration. With those features, we chose rechargeable batteries.

4. ‣ This part didn’t meet our original expectation at first. The max speed was below 100 cm/s. Also the older version of our cars needs large current to support. Thus we changed our car model from a 36-cm-long model to an 11-cm-long model. The new model we choose is faster (max speed is 120 cm/s) and requires less current to initialize.

‣ Lazaque later changed car’s circuit to better power the car motor and reduce risks of frying Arduino and its related circuit. (He fried 2 chips while trying to figure out a better power system.)

### 11.1.2 Motion Control (running straight) & Speed Mapping

By Lazaque Mugerwa, Nick Seal and Chaoyun Ma

1. Cars would run straight for about 200 cm then would turn right. Cars would be off line 100 cm after running 500 cm. We reduced the deviation by 50 cm. We also mapped the car speeds with PWM.

2. Cars were expected to run straight. They obviously won’t. We could only reduce the deviation by half. However, the mapping is a success.

3. ‣ Cars don’t have steering control, which makes it hard to steer cars to go straight.

‣ The mapping enabled us to do a look-up table when controlling cars’ speed.

4. ‣ Tighten up the steering arms of cars

‣ Glued all the loose steering parts to the car frames.

However, without a steering control (mainly PID control), it’s not possible for cars to go straight.

### 11.1.3 Speedometer

**a)** **Using infrared sensors (By Lazaque Mugerwa (main), Chaoyun Ma and Nick Seal)**

· Speed-reading is not accurate. The infrared detector would often omit some readings. Most of time,environment heavily affects the reading results.

· The required circuit to generate a 38kHz pulse is difficult to implement on the car. It requires too much space.

· It’s hard to position both emitter and detector right.

· Infrared sensors need to be changed to another kind of sensors, which won’t be affected by environment and easy to implement.

**b)** **Using Magnetic Hall Sensors  (By Lazaque Mugerwa)**

1. Speed readings are accurate. We tested 5 different speed on all 3 cars and all work well.

2. Error within 1 cm/s.

3. The Magnetic Hall sensors are very sensitive and stable to any environmental changes. The sensor requires little circuit and it’s very easy to implement on the car. Those features make it ideal for reading car speed.

## Wireless communication & Remote Control

### Xbee

By Nick Seal

1. Cars and Tower can speak to each other.

● The Tower can tell cars which PWM to use

● Cars can tell the Tower their speeds separately.

2. This part meets the original expectations that there are two-way communications.

3. ● The communication speed is fast. This will not hinder Tower controlling cars in real time.

● When the cars talk back at the same time, there are times when the received data get messed up. That’s because the signals step on each other. In order to solve this problem, buffer needs to be cleared each time when communication occurs.

4. ● Car A, B and C need to be distinguished. Each Xbee on the car has a unique call number. When Tower broadcasts its commands, each commands will come with a specific call number in order to specify which car would follow the command.

● Car A, B and C will talk, only when Tower asks them to. This way, there would be no stepping on each other.

● Xbees’ supply voltage is .8 – 3.4 V; and transmit current is 45 mA. Both are much lower than the car motor’s. Thus car’s power system is made up with two parts. One is for the motor, and the other is for the circuit.

### Remote Control

By Nick Seal (main) and Chaoyun Ma

1. ● Tower can state and stop all three cars at the same time.

● Tower can control car’s speed by transmitting new PWM to the car separately.

● Cars can transmit their speeds back to Tower separately.

2. The results satisfy the expectations for this function.

3. When powers up, the communication won’t work out for about 10 – 15 seconds. This is mainly caused by the difference between Arduino’s required voltage (5 V) and Xbee’s required voltage (3.1 V). The board needs time to cool down in case anything is fried.

4. ● Finding right speeds for cars is difficult. It needs to be fast and accurate. Also, the data transmitted to the cars need to be small and clear. And in the best cases, cars won’t need to spend any source to calculate anything. To meet those requirements, a look-up table (converting speed to PWM) is the best solution. It requires little memory space. The time spent on calculation and searching for reight value is o(1).

● Implementing look-up table is difficult. After carefully examined the data, we found a unique mapping to connect speed to PWM. Since the new speed can only be as large as 120, which is a 3 digit number, we use its first 2 digit as the position of its correlated PWM. In this case, there are only 9 different speed levels in a 18-item-large array.

## 11.3 Distance Measurement

By Chaoyun Ma

### 11.3.1 Ultrasonic Distance Sensor

1. Accuracy (< 1 cm) ; long distance covered (3 cm – 250 cm); fast (< 0.01s to read an object 250 cm away); cheap

2. This is ideal.

3. Each lane is 5 meters long. Thus we cascade two ultrasonic sensors at each lane.

4. Pulses will step on each other when sensors sending out signal at the same time. However, by adopting a switcher in the algorithm, this problem can be solved.

## 11.4 System Integration

By Nick Seal and Chaoyun Ma

### 11.4.1 Connecting Ultrasonic Sensors to the Tower

1. There weren’t any readings at first. After we changed the reading orders of the sensors in the program, we got good readings.

2. It meets the requirement.

3. This cascading system has a bind spot, which is the joint between two sensors. The bind spot is15cm long. This bind spot will only make the system bind for 0.2 seconds, which will not affect the overall system.

### 11.4.2 Testing The Project

1. We have 2 successful trials and 50 failures, due to car’s inability to go straight.

2. SUCCESS × 2 !

3. The reasons for so many trials are:

● Cars won’t go straight. In order to get cars enter the intersection as well as stay in the area covered by ultrasonic sensors, we angled the car slightly to the left, but this angling is purely empirical and needs many trials.

● Car speed is positive correlated to batteries’ voltage. Since the voltage decreases as time goes, car speed decreases as well, which makes the initial speed unpredictable. We won’t be able to position all three cars right to have the expected scenario.

### 11.4.3 Testing Results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| car A | | car B | | car C |  | to speed up |
| distance | speed | distance | speed | distance | speed |  |
| 95 | 94 | 96 | 78 | 307 | 102 | C to pwm 255 |
| 44 | 133 | 93 | 89 | 304 | 110 | C to pwm 255 |
| 302 | 61 | 305 | 255 | 293 | 59 | no new speed |
| 118 | 31 | 245 | 36 | 270 | 44 | C to pwm 120 |
| 84 | 60 | 101 | 52 | 246 | 62 | A to pwm 255 |
| 92 | 69 | 90 | 68 | 289 | 85 | A to pwm 255 |
| 95 | 94 | 96 | 78 | 307 | 102 | C to pwm 255 |
| 44 | 113 | 93 | 89 | 304 | 111 | C to pwm 255 |
| 76 | 115 | 60 | 88 | 283 | 112 | C to pwm 255 |
| 252 | 175 | 248 | 173 | 260 | 175 | A to pwm 255 |
| 169 | 184 | 248 | 175 | 261 | 177 | A to pwm 255 |
| 251 | 177 | 247 | 177 | 260 | 175 | A to pwm 255 |
| 253 | 177 | 247 | 179 | 259 | 175 | A to pwm 255 |
| 253 | 184 | 247 | 173 | 259 | 173 | A to pwm 255 |
| 111 | 21 | 91 | 6 | 265 | 32 | no new speed |

# Administrative and Other Design Issues

## \* Project Management

One paragraph from each team member

How was the project managed, how was tasks distributed, how was scheduled made, what project management software/method did you use, your experience in working together, what have you learned in terms of team work, time management, project management etc., from this project.

First we design the project together. After the design was finalized, each of us chose the parts that we were interested in to work on. When our individual work was done, we then integrated the whole system and tested.

We mainly used Google drive to keep track of our work. Since we can share our work online, the time schedule for us to meet was quite loose. However, we do learn that meet in person can reduce quite a lot workload and motivate us to do better. Meeting in person is easier to brainstorm, and a very good way to monitor our progress.

## Requirements traceability matrix

A matrix that traces stated modules and data structures to the Software Requirements Specification is developed.

## Packaging and installation issues

Special considerations for software packaging and installation are presented.

## Design metrics to be used

A description of all design metrics to be used during the design activity is noted here.

## Restrictions, limitations, and constraints

Special design issues that impact the design or implementation of the design are noted here.

# \* Conclusion and Future Work

## \* Conclusion

\* Must complete for each team member: Describe what you learned from this project, both in terms of technical knowledge and skill, professional and personal growth.

Our project is a success. We successfully conducted a live demonstration on April 9, 2015.

Lazaque Mugerwa did car design.

Nick Seal did wireless communication, system design and project testing .

Chaoyun Ma did distance measurement, system design and project testing.

Conclusion (Chaoyun):

Making plan is very important. Making sensible plan is much more important. The point to have plans can help keep track of things and make sure everything can be finished in time. In order to make good plans, one need to know the project thoroughly, and consult experts on it.

Both hardware and software are their own system, and they should be tested separately at first. However, the system is the perfect combination of hardware and software. Debugging the system is quite trouble-some. Patience and luck are often needed. Most importantly, only good hardware design combined with a thoroughly thought, bug-less software can make a good project.

## Future Work

Add PID controls so that cars can go straight.

## \* Acknowledgement

Professor Ping Liang,

Professor Gang Chen

Pavle Kirilov

# \* References

[1] Arduino, “Homepage”.

Available: <http://www.arduino.cc>

[2] Digi International Inc , “XBee 802.15.4 datasheet”.

Available: <http://www.digi.com/pdf/ds_xbeemultipointmodules.pdf>

[3] Digi International Inc, “XBee 802.15.4 Starter Kit Getting Started Guide”

Available: <http://www.digi.com/support/getasset?fn=90002160&tp=3>

[4] Elec Freaks, “Ultrasonic Ranging Module HC - SR04”,

Available: http://www.micropik.com/PDF/HCSR04.pdf

# Appendices

**Appendix A:** Software List

**CarA**

#include <Time.h>

#include <SoftwareSerial.h>

SoftwareSerial xbee(0,1);

int pwmPin = 9;

int revolutions = 0;

const float pi = 3.14;

unsigned long t1, t2, delta\_t = 0, time;

float R = 4.7;

float circumference = R\*pi; // R in this case equals 2 Centimeters, = 12

float rcSpeed = 0;

int speeder = 200;//this is for the car, change it to change car speed

int talk\_flag = 0;

int speed\_test;

int i = 0;

float speed\_ave[5];

float speed\_curr, dif, dif2;

void setup()

{

pinMode(pwmPin, OUTPUT);

attachInterrupt(0, indicate, FALLING); // Pin 2 Is used for INT 0

t1 = millis();

xbee.begin(9600)

}

void loop()

{

xbee.listen();

int comm = xbee.read();

if (comm == 0)

{

analogWrite(pwmPin, 0);

}

else if(comm == 1)

{

analogWrite(pwmPin, speeder);

}

else if(comm == 5)

{

int i = Serial.read();

while(i == 5)

{

i = Serial.read();

}

speeder = i;

analogWrite(pwmPin, speeder);

}

else if (comm == 2)

{

talk\_flag = 1;

}

if (comm == -1)

{

if (talk\_flag == 1)

{

speed\_test = (int) rcSpeed;

xbee.write(speed\_test);

talk\_flag = 0;

}

}

}

void indicate() // ISR

{

t2 = millis();

delta\_t = (t2 - t1);

revolutions = revolutions + 1;

rcSpeed = ((revolutions\*circumference)/(delta\_t))\*1000;

t1 = t2; revolutions = 0;

revolutions = 0;

}

**TOWER**

//#include <SoftwareSerial.h>

//SoftwareSerial xbee(0,1);

int a\_dis, b\_dis, c\_dis,a\_spe, b\_spe, c\_spe;

const int A\_far\_trig = 22;

const int A\_far\_echo = 23;

const int A\_close\_trig = 24;

const int A\_close\_echo = 25;

const int B\_far\_trig = 26;

const int B\_far\_echo = 27;

const int B\_close\_trig = 28;

const int B\_close\_echo = 29;

const int C\_far\_trig = 30;

const int C\_far\_echo = 31;

const int C\_close\_trig = 32;

const int C\_close\_echo = 33;

//int dis\_holder;

int dis\_holder\_a,dis\_holder\_b,dis\_holder\_c;

int a\_flag = 0;

int b\_flag = 0;

int c\_flag = 0;

int speed\_flag = 0;

int new\_speed;

int new\_pwm = 0;

int again = 0;

int over = 0;

//int pwm[11] = {0, 80, 100, 120, 140, 160, 180, 200, 220, 240, 255};

//int table[11] = { 0, 10, 48, 55, 70, 97, 102, 127, 135, 160, 170};

int pwm[18] = { 0, 80, 80, 80, 80, 120, 120, 140, 140, 160, 180, 180, 200, 220, 220, 220, 240, 255};

int pos;

int speed\_for\_car;

int a\_spe\_flag = 0;

int b\_spe\_flag = 0;

int c\_spe\_flag = 0;

int a\_dis\_flag = 0;

int b\_dis\_flag = 0;

int c\_dis\_flag = 0;

int push\_start = 7;

int push\_stop = 8;

//Codes for communication & commands.

int start = 1;

int halt = 0;

int ask\_a\_spe = 2;

int ask\_b\_spe = 3;

int ask\_c\_spe = 4;

int a\_delt\_spe = 5;

int b\_delt\_spe = 6;

int c\_delt\_spe = 7;

void setup()

{

//a\_dis = 0;

//xbee.begin(9600);

pinMode(push\_start, INPUT);

pinMode(push\_stop, INPUT);

pinMode(A\_far\_trig, OUTPUT);

pinMode(A\_far\_echo, INPUT);

pinMode(A\_close\_trig, OUTPUT);

pinMode(A\_close\_echo, INPUT);

pinMode(B\_far\_trig, OUTPUT);

pinMode(B\_far\_echo, INPUT);

pinMode(B\_close\_trig, OUTPUT);

pinMode(B\_close\_echo, INPUT);

pinMode(C\_far\_trig, OUTPUT);

pinMode(C\_far\_echo, INPUT);

pinMode(C\_close\_trig, OUTPUT);

pinMode(C\_close\_echo, INPUT);

Serial.begin(9600);

//Serial1.begin(4800);

pinMode(10,INPUT);

pinMode(12,INPUT);

}

void loop()

{

//Serial.println();

//Serial.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

// Basic start/stop commands.

if(digitalRead(push\_stop) == HIGH)

{

//xbee.write(halt);

Serial.write(halt);

//Serial.flush();

//a\_spe = 0;

}

else if(digitalRead(push\_start) == HIGH)

{

//xbee.write(start);

//Serial.print("trip\_1");

Serial.write(start);

//Serial.flush();

delay(1000);

a\_dis\_flag = 1;

}

if(a\_spe\_flag == 1)

{

//xbee.write(1000);

//Serial.print("trip\_2");

if(Serial.peek() < 1)

{

//Serial.print("trip\_3");

Serial.write(ask\_a\_spe);

}

else if (Serial.peek() > 0)

{

a\_spe = Serial.read();

while(Serial.available())

{

Serial.read();

}

Serial.print("Car A speed is: ");

Serial.print(a\_spe);

Serial.print('\n');

Serial.print("Car A distance is: ");

Serial.println(a\_dis);

Serial.flush();

a\_dis\_flag = 0;

a\_spe\_flag = 0;

b\_dis\_flag = 1;

}

}

if(b\_spe\_flag == 1)

{

if(Serial.peek() < 1)

{

Serial.write(ask\_b\_spe);

}

else if (Serial.peek() > 0)

{

b\_spe = Serial.read();

while(Serial.available())

{

Serial.read();

}

Serial.print("Car B speed is: ");

Serial.print(b\_spe);

Serial.print('\n');

Serial.print("Car B distance is: ");

Serial.println(b\_dis);

Serial.flush();

b\_dis\_flag = 0;

b\_spe\_flag = 0;

c\_dis\_flag = 1;

}

}

if(c\_spe\_flag == 1)

{

if(Serial.peek() < 1)

{

Serial.write(ask\_c\_spe);

}

else if(Serial.peek() > 0)

{

c\_spe = Serial.read();

while(Serial.available())

{

Serial.read();

}

Serial.print("Car C speed is: ");

Serial.print(c\_spe);

Serial.print('\n');

Serial.print("Car C distance is: ");

Serial.println(c\_dis);

Serial.flush();

c\_spe\_flag = 0;

c\_dis\_flag = 0;

speed\_flag = 1;

}

}

while(Serial.available())

{

Serial.read();

}

//Finding distance to B.

if (b\_dis\_flag == 1)

{

if (b\_flag == 0)

{

dis\_holder\_b = dis\_1(B\_far\_trig, B\_far\_echo);

if (30 < dis\_holder\_b < 200) // changed from 5 to 30

{

b\_dis = dis\_holder\_b + 200;

b\_spe\_flag = 1;

}

if( dis\_holder\_b < 30)

{

b\_flag = 1;

}

}

else if(b\_flag == 1)

{

dis\_holder\_b = dis\_1(B\_close\_trig,B\_close\_echo);

if (5 < dis\_holder\_b < 200)

{

b\_dis = dis\_holder\_b;

b\_spe\_flag = 1;

}

if( dis\_holder\_b < 5)

{

b\_flag = 2;

b\_spe\_flag = 0;

}

}

}

//Finding distance to A.

if (a\_dis\_flag == 1)

{

if (a\_flag == 0)

{

dis\_holder\_a = dis\_1(A\_far\_trig, A\_far\_echo);

if (30 < dis\_holder\_a < 200) // changed from 5 to 10

{

a\_dis = dis\_holder\_a + 200;

a\_spe\_flag = 1;

}

if( dis\_holder\_a < 35)

{

a\_flag = 1;

}

}

else if(a\_flag == 1)

{

dis\_holder\_a = dis\_1(A\_close\_trig,A\_close\_echo);

if (5 < dis\_holder\_a < 200)

{

a\_dis = dis\_holder\_a;

a\_spe\_flag = 1;

}

if( dis\_holder\_a < 5)

{

a\_flag = 2;

a\_spe\_flag = 0;

}

}

}

//Finding distance to C.

if(c\_dis\_flag == 1)

{

if (c\_flag == 0)

{

dis\_holder\_c = dis\_1(C\_far\_trig, C\_far\_echo);

if (30 < dis\_holder\_c < 200) // changed from 5 to 10

{

c\_dis = dis\_holder\_c + 200;

c\_spe\_flag = 1;

}

if( dis\_holder\_c < 35)

{

c\_flag = 1;

}

}

else if(c\_flag == 1)

{

dis\_holder\_c = dis\_1(C\_close\_trig,C\_close\_echo);

if (5 < dis\_holder\_c < 200)

{

c\_dis = dis\_holder\_c;

c\_spe\_flag = 1;

}

if( dis\_holder\_a < 5)

{

c\_flag = 2;

c\_spe\_flag = 0;

}

}

}

//collision calc

if(speed\_flag == 1)

{

new\_speed = collision\_calc(a\_spe,b\_spe,c\_spe,a\_dis,b\_dis,c\_dis);

new\_pwm = finding\_pwm\_from\_hell(new\_speed);

if(new\_speed > 0)

{

if (new\_speed > 3000)

{

Serial.println("Car C, ");

Serial.write(c\_delt\_spe);

//Serial.flush();

new\_speed = new\_speed - 3000;

}

else if(new\_speed > 2000)

{

Serial.println("Car B, ");

//new\_speed = new\_speed - 2000;

Serial.write(b\_delt\_spe);

}

else if(new\_speed > 1000)

{

Serial.println("Car A, ");

//new\_speed = new\_speed - 1000;

Serial.write(a\_delt\_spe);

}

//Serial.print("speed change to: ");

if(pos > 17)

{

new\_pwm = 255;

}

Serial.write(new\_pwm);

Serial.print("speed to: ");

Serial.println(new\_speed);

Serial.print("pwm is: ");

Serial.println(new\_pwm);

Serial.print("pos: ");

Serial.println(pos);

delay(50);

speed\_flag = 0;

new\_pwm = 0;

again = 1;

}

else if(new\_speed == 0)

{

Serial.println("no new speed");

//again = 1;

speed\_flag = 0;

new\_pwm = 0;

}

}

if(again == 1)

{

if (a\_flag == 2)

{

if(b\_flag == 2)

{

if(c\_flag == 2)

{

over = 1;

}

}

}

else

{

a\_dis\_flag = 1;

}

again = 0;

}

if(over == 1)

{

Serial.write(halt);

Serial.println("over");

}

}

///////////////////FUNCTIONS////////////////////

//distance measurement function

int dis\_1(int trig, int echo)

{

digitalWrite(trig,LOW);

delayMicroseconds(2);

digitalWrite(trig,HIGH);

delayMicroseconds(20);

digitalWrite(trig,LOW);

int pulse = pulseIn(echo,HIGH);

int distance = pulse/58;

return distance;

}

//finds new pwm for related speed.

int finding\_pwm\_from\_hell(int new\_speed)

{

speed\_for\_car = new\_speed / 1000;

pos = (new\_speed / 10) - (speed\_for\_car \* 100); //here to find the pos at look-up table

new\_pwm = pwm[pos];

return new\_pwm;

}

//collision calculation & new speed determination function

int collision\_calc(int a\_spe, int b\_spe, int c\_spe, int a\_dis, int b\_dis,int c\_dis)

{

double ti\_a, ti\_b, ti\_c, col\_1, col\_2, c\_s, c\_d, a\_s, a\_d, b\_s, b\_d;

c\_s = c\_spe;

c\_d = c\_dis;

a\_s = a\_spe;

a\_d = a\_dis;

b\_s = b\_spe;

b\_d = b\_dis;

ti\_a = a\_d/a\_s;

ti\_b = b\_d/b\_s;

ti\_c = c\_d/c\_s;

col\_1 = ti\_a - ti\_b;

col\_2 = ti\_c - ti\_b;

col\_1 = abs(col\_1);

col\_2 = abs(col\_2);

if (col\_1 < 2)

{

col\_2 = ti\_c - (ti\_b + 1);

col\_2 = abs(col\_2);

if (col\_2 < 2)

{

new\_speed = ((a\_dis/(ti\_a - 1)) + 1000);

}

else

{

new\_speed = ((b\_dis/(ti\_b+ 1)) + 2000);

}

}

else if( col\_2 < 2)

{

col\_1 = ti\_a - (ti\_b + 1);

if ( col\_1 < 2)

{

new\_speed = ((c\_dis/(ti\_c - 1)) + 3000);

}

else

{

new\_speed = ((b\_dis/(ti\_b+1))+2000);

}

}

else

{

new\_speed = halt;

}

return new\_speed;

}