ECE 477 Final Report – Spring 2013
Team # – Team ID

Team Members:

#1: Nathan Begle	**Signature: NB_____**	**Date: 04/29/2013**
#2: Jiaqi Jiang	**Signature: JJ_____**	**Date: 04/29/2013**
#3: Chang Yoon Kim	**Signature: CYK_____**	**Date: 04/29/2013**
#4: Jordan Gaines	**Signature: JG_____**	**Date: 04/29/2013**

CRITERION	SCORE	MPY	PTS
Technical content	0 1 2 3 4 5 6 7 8 9 10	3	
Design documentation	0 1 2 3 4 5 6 7 8 9 10	3	
Technical writing style	0 1 2 3 4 5 6 7 8 9 10	2	
Contributions	0 1 2 3 4 5 6 7 8 9 10	1	
Editing	0 1 2 3 4 5 6 7 8 9 10	1	
Comments:		*TOTAL*	

Table of Contents

Abstract

The Infrarat is an autonomous toy car which utilizes an array of infrared and ultrasonic sensors to navigate the world. The vehicle communicates with an android device to control the vehicle's movement mode and transmit infrared (IR) data and send battery information to the user. The Infrarat has three operating modes: flee, follow, and manual. Flee mode results in the vehicle running away from people and obstacles at high speed. Follow mode causes the robot to follow a person using IR heat tracking. Manual mode allows the user to take manual control of the device with virtual thumb sticks on the Android device.

1.0 Project Overview and Block Diagram

The Infrarat consists of a few critical components: two high speed motors, a swiveling servo-mounted IR array, and a Bluetooth communication device. The vehicle uses two geared brushless DC motors to power the wheels and provide high speed navigation. Mounted on the vehicle is an array of four 16x4 grid IR sensors that supplies the user with directional temperature information via the I2C protocol. The IR sensors allow the vehicle to track warm bodies and follow movement with both servo control and vehicle movement. The heavy floating point calculations involved in parsing the raw temperature data is handled by an AT32UC3C2256C Atmel microcontroller running full-speed at 60 MHz. In addition to the IR sensors a pair of ultrasonic sensors mounted on the front and rear of the vehicle allow the vehicle to detect and avoid nearby objects. The Bluetooth module allows the Infrarat to communicate with an Android device. This is used to control the three modes and communicate battery charge levels and display the temperature gradient to the user. The block diagram is displayed below.

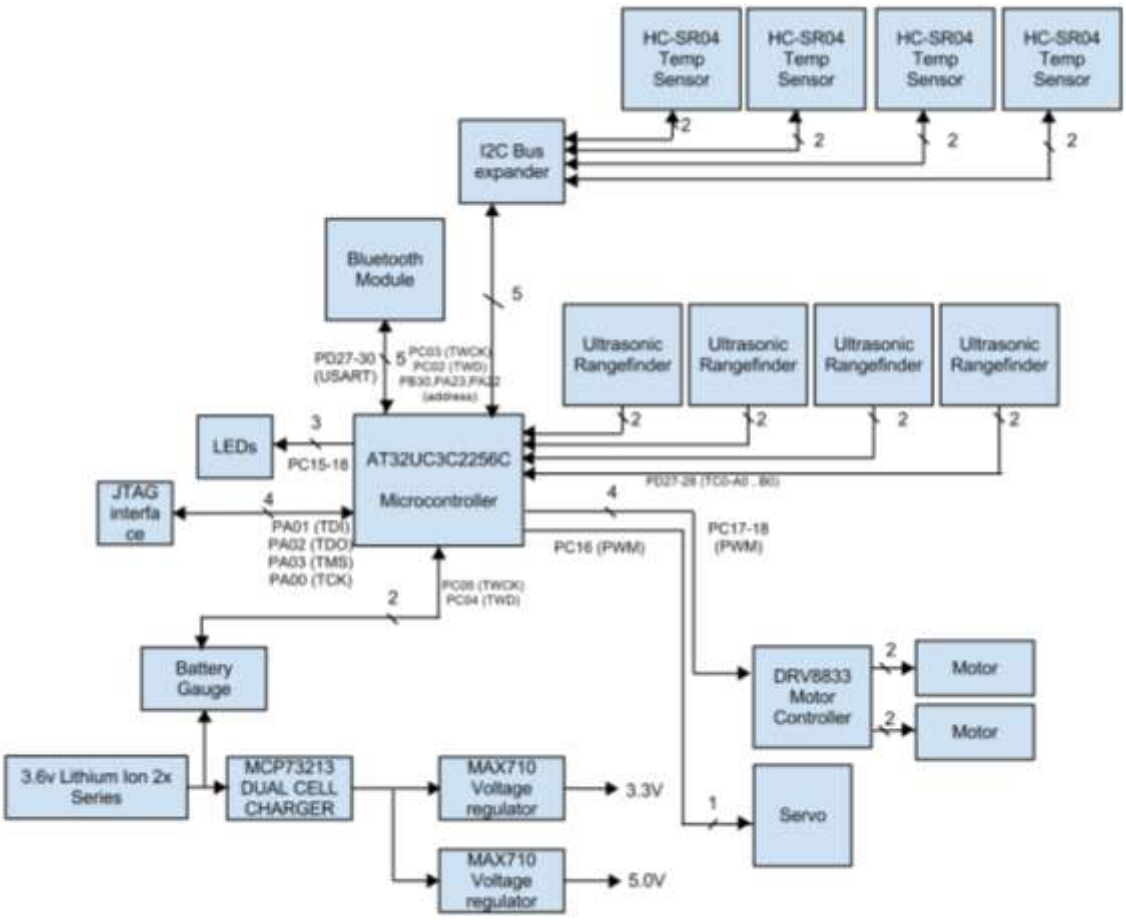


Figure 1.0: Block diagram

2.0 Team Success Criteria and Fulfillment

1. The ability to charge and regulate a rechargeable battery and show the user how much life is left

   The Infrarat is able to successfully recharge its internal battery pack via wall wart plug-in and displays fuel gauge information to the android device.

2. The ability to track a random person and follow them

   While results not as impressive as hoped, the Infrarat successfully tracks and follows any person within sensor range.

3. The ability to avoid running into objects

   The Infrarat is very effective at avoiding walls and other obstacles in all movement modes. Additional emergency stopping is activated when sensors detect dangerous proximity to walls or people.

4. The ability to show a temperature gradient from the IR sensor on an Android device

   The temperature gradient is transmitted to the android device and displayed as a colored temperature gradient as originally planned. Display updates at approximately 5 Hz.

5. The ability to flee from a person

   The vehicle is effective at fleeing from people and avoiding collision. Car wanders randomly until approaching objects or walls obstruct its movement. The car then takes action to avoid the encroaching object.

3.0 Constraint Analysis and Component Selection

Design Constraint Analysis

The chassis design had two major constraints, having to do with durability and reliability. For durability, the main issue was the IR sensor. It would be easily exposed on a servo so it could be broken off. This would pose a problem if children or small animals would be playing with it because that demographic isn't known for their light touch. The proximity sensors on the sides of the chassis also needed to be exposed but a possible workaround would have been a wire mesh over the critical parts making the more durable.

In terms of reliability, the main issue was the IR sensor. Decent IR cameras with a higher resolution can cost upwards of a few thousand dollars so a cheaper very low resolution sensor array needed to be used. The chosen IR sensor also requires many floating point calculations which can slow down the robot's decision-making logic. Another reliability issue is the battery

pack. The robot needs to last long enough on a single charge to provide adequate enjoyment which means that the motors and other components need to be as low power as possible. It also meant that a larger battery pack would be needed to be provided to ensure a longer battery life. This battery pack, while standard in that it is a lithium ion battery, is a 2-cell battery, which meant that the components involved with the battery also needed to be designed for two cells in series. This battery was specifically chosen so that the motors would be provided enough current when going at full speed, but with that benefit came some additional constraints which weren't part of the original plan. The fuel gauge needed to support 2-cell batteries, since according to our research, is not as common as single-cell fuel gauges. Not only was the right fuel gauge IC hard to find, but it was also only available in a QFN package. The charge controller was also in a similar situation, but it was slightly easier to find.

Computation Requirements

The major computational requirement for this project was calculating the temperatures using the data from the IR array sensor. The calculations require multiple floating point arithmetic operations for each pixel. The floating point calculations could be reduced if the data is rounded off and integer-based decimal point arithmetic could be used. Since the IR array is 16x4, that is 64 individual pixels that the microcontroller needs to calculate a temperature for. This needed to be done at fast enough refresh rates so that any latency between the robot and user would be negligible. We are expecting the microcontroller to need to refresh at around 30 times a second. This delay can be reduced if the refresh rate is held at around 20 times a second, since there would be fewer calculations involved in a certain window of time and less bandwidth needed for transmission. We determined that a table-lookup method would not be feasible because the data from the IR sensor is 16-bits and in addition to the IR data, there are multiple values that need to be read from the other sensors. The microcontroller has a speed of about 50 MHz and so it could handle the calculations quickly and efficiently.

The IR sensor isn't needed in flee and manual modes so the microcontroller can place priority on the proximity sensors and the motors so that it does not run into anything. The IR temperature data will still be sent to the Android phone but a delay in transmission of that data will not affect the movement of the car. The proximity sensors require fairly simple calculations that will utilize

a large amount of if statements given a distance from each proximity sensor. The distance is easily calculated because it is a ratio of the duty cycle that the sensor inputs. No floating point calculations are required as a finer range can be determined by selecting finer units for distance.

Interface Requirements

The main interface requirements deal with communicating with the IR sensor, battery gauge, and the Bluetooth module. The Bluetooth module has a UART interface or a USB interface. Since the UART protocol is readily available on most microcontrollers, it will be the chosen protocol to communicate with the module. The IR array sensor and the battery gauge both require the I^2C protocol for communication with the microcontroller. Multiple devices can share an I^2C bus so only one bus will be required. To accomplish this, an I2C switch was employed on the PCB since it was discovered that the microcontroller had limited timer capabilities over two channels. The microcontroller runs at 3.3 volts while some of the components require 5 volts, so two separate voltage regulators were placed on the PCB to satisfy that requirement.

On-Chip Peripheral Requirements

The on-chip peripheral requirements of the Infrarat consisted mainly of timers, PWMs, and digital communication protocols. The proximity sensors output a PWM signal based on the distance so four input capture channels were needed to make sure that the robot could measure the duty cycle of the sensor. These sensors will also require the need to use timers to measure the duty cycle of the signal. For this it was decided we would need at most four timers. Note that due to problems later on in development, we were only able to utilize 1 timer channel running two sensors. The timers needed to be at least 8-bits but 16-bit timers are preferred because that allowed for a higher resolution of distance from the proximity sensors. We will also need three 8-bit PWM outputs so the microcontroller can control the two motors that drive the car as well as the servo that turns the IR sensor.

The Infrarat's required digital communication protocols consist of I^2C and UART. The IR array sensor data and the battery gauge chip are accessed by sending commands through I^2C. This means that we will need at least one I^2C peripheral that can communicate with multiple devices

on the bus. The Bluetooth Module communicates with the microcontroller through the UART protocol. This means that at least one UART peripheral is required by the microcontroller.

Off-Chip Peripheral Requirements

The only off-chip peripherals that were required were a battery gauge and a motor controller. The motor controller will be controlled with the microcontroller's PWM outputs and 4 outputs to control the H-bridges in the motor controller. The motor controller needed to have two motor control outputs for the two motors on the car. The battery gauge interfaces with I^2C and will be used to measure how much power is left in the battery. This data would then be transmitted and displayed on the Android phone.

Power Constraints

Power is a major concern for the Infrarat because it runs off of batteries. It will need to be able to last long enough to provide adequate enjoyment for the user. The car needed to be fast enough to be hard enough to catch in order to make it entertaining. This means that faster and more powerful electric motors were needed which shortened the life of the car. The batteries are charged using a controller along with a wall-wart so no transformers or rectifiers will be needed on the car.

Packaging Constraints

Packaging constraints can be a major problem for the car because it needs to withstand a moderate amount of abuse. This poses a problem because the IR sensor will be sticking up and attached to a servo. Most of the car can be covered by a plastic shell but the most sensitive part will be the IR sensor assembly which will need to move freely on the top of the chassis. Another constraint is that the chassis needs to be as light weight as possible. The lighter the chassis the faster the motors will be able to drive it and the less battery power it requires.

Cost Constraints

There are no similar toy cars compared to what the Infrarat is capable of so no perfect comparison can be made. One can base the robot car off of other remote control cars. Most of the remote control cars on the market can cost from about $20 to around $200 for higher end cars.

The total cost of our car would ideally be around $170 to $180 which is well within the price range of other cars.

Component Selection Rationale

The most costly component for our car is the IR array sensor. We chose an IR array sensor from Melexis (MLX90620) because it had a respectable resolution as well as a field of view small enough as to make a good reading from a target a few meters away. The IR sensor has a resolution of 16x4 with a field of view as small as 40 degrees. This should prove sufficient for detecting a person at a range of 2-3 meters. This sensor was much better than another sensor from Omron (D6T) which had fewer pixels. The D6T had a resolution of 1x8 or 4x4 which was thought to not provide an adequate view of the surroundings. Even though the Melexis sensor required a lot of calculations to get the correct temperature reading, it allowed for a much better view of its target. This means that the following function of the car will be able to work more accurately. If a smaller array sensor was used then the person the car was tracking could easily walk out of view making the car loose track. This is an important feature of the car so it had to be favored over the calculation constraints.

An Atmel AVR ATUC128L4U microcontroller was chosen for this design because of the vast amount of PWM channels and the speed of the microcontroller. Our other option was a microcontroller from Microchip, a dsPIC30F6013A, which was thought to have an adequate speed for our calculations. A comparison table can be seen in Table 1 which shows the various requirements of our project. Another disadvantage with the PIC was that it had fewer timers than the AVR microcontroller. The prices were about the same with the Atmel microcontroller costing about $9.24 which is about $4 less than the Microchip controller. They both have the digital communications protocols that we need (UART and I^2C) but the Atmel microchip would be a little faster which allows for us to use faster refresh rates on our IR temperature sensor. It also turned out that the Atmel used a lower voltage at 3.3V compared to 5V for the Microchip microcontroller as well as having a less power consumption. This will greatly enhance battery life giving another reason to go with the Atmel microcontroller.

	Atmel [1]	*Microchip [2]*

	AT32UC3L0128	*dsPIC30F6013A*
Price	$9.24	$13.72
Speed	50 MHz	30 MHz
Power	49.5 mW	1.25 W
Pin Count	48 pins	64 pins
Dev board available in lab	No	Yes
Input capture channels	12	8
I^2C & UART	Yes	Yes
PWM output channels	35	8 (shared with input capture)

Table 1

The proximity sensor (HC-SR04) was chosen over others because it was readily available for a low price on Amazon as well as having an easy to use interface. The sensor has a long range of about 5 meters and a resolution of about 0.3 cm. This will provide an accurate "view" of the car's surroundings and have a long enough range so the car can detect and avoid obstacles. The other sensor (the PING))) from Parallax) we compared it with had a shorter range of about 3 meters and a higher price tag. With a shorter range and the car moving at a relatively fast speed, the car might not be able to react in time if an object gets in the way.

The Magician's chassis from Sparkfun was selected for the car was selected because it was extremely cheap and had a flat top which would make mounting our PCB on much simpler. It also has a zero turn radius because the wheels are controlled by two motors with only a ball acting as a front wheel. This means the car can make much sharper turns to avoid objects which will be useful at higher speeds. It was chosen over another pre built car, the i-Racer from Sparkfun, mainly because the i-Racer car didn't have a flat workable area on top as well as a small turning radius. The selected car also turned out to be cheaper than the alternative which helps keep the total cost of the car down.

4.0 Patent Liability Analysis

Results of Patent and Product Search

One result of the search for relevant patents was a patent filed by Honeywell International Inc. on November 28, 2000, US Patent 6829370. This patent has claims that are very similar to how the Infrarat detects a warm body and determines whether or not it had encountered a human. The abstract states, "A detection method and system that detects reflection from a scene in at least a portion of an upper band of the near infrared spectrum. The presence of a human body in the scene is then determined by comparing the reflection of at least one region of the scene (e.g., at least one region of the scene including a face region of the human body) to at least one other region of the scene (e.g., at least another region of the scene including one or more inanimate objects)." The first claim describes the method of detection, in the same manner as shown above in the abstract. In particular, the fourth claim would have the most potential for infringement. The claim is worded as follows, "The method of claim 1, wherein detecting reflection includes detecting reflection from the at least one region of the scene comprising a face region of the human body." The detection function in the software for the Infrarat does exactly that: detect a facial region of the human body, but also adds an additional detection area around the hips/torso area. This would fall under the Doctrine of Equivalents rather than Literal Infringement, because although the end result is the same (human detection by facial warmth recognition), the methods used to arrive at the detection is different.

The second patent is US patent 8020657, filed by iRobot Corporation and Deere & Company on October 20, 2006. The abstract is stated as follows, "Embodiments of the invention provide systems and methods for obstacle avoidance. In some embodiments, a robotically controlled vehicle capable of operating in one or more modes may be provided. Examples of such modes include teleoperation, waypoint navigation, follow, and manual mode. The vehicle may include an obstacle detection and avoidance system capable of being implemented with one or more of the vehicle modes. A control system may be provided to operate and control the vehicle in the one or more modes. The control system may include a robotic control unit and a vehicle control unit." This patent is the closest match that could be found after some time searching, and highlights what could possibly be a serious case of patent infringement if the Infrarat were to go to market. An important excerpt is as follows: "A robotically controlled vehicle comprising: a robotic control unit comprising a control application; a sensor for detecting object data representing detected obstacles or objects located outside of the robotically controlled vehicle

and for transmitting the object data to the robotic control unit, the object data being three-dimensional data." Teleoperation, manual control, and follow modes are all features that are present on the Infrarat, and in one of the later claims, it is stated that these peripherals are all controlled by a processor with memory, which is also the case for the Infrarat. The next part about obstacle detection is also what appears to be literal infringement, as the Infrarat utilizes the ultrasonic sensor peripherals to avoid smashing into objects while in operation.

The next patent in question is US patent 7211980, filed by Battelle Energy Alliance, LLC on July 5th, 2006. The abstract states, "Robot platforms, methods, and computer media are disclosed. The robot platform includes perceptors, locomotors, and a system controller, which executes instructions for a robot to follow a target in its environment. The method includes receiving a target bearing and sensing whether the robot is blocked front. If the robot is blocked in front, then the robot's motion is adjusted to avoid the nearest obstacle in front. If the robot is not blocked in front, then the method senses whether the robot is blocked toward the target bearing and if so, sets the rotational direction opposite from the target bearing, and adjusts the rotational velocity and translational velocity. If the robot is not blocked toward the target bearing, then the rotational velocity is adjusted proportional to an angle of the target bearing and the translational velocity is adjusted proportional to a distance to the nearest obstacle in front." This patent is similar to the one mentioned previously, but focuses on just the method of navigation. The patent mentions a robot platform that has 'perceptors, locomotors, and a system controller', which serve as a means for sensing and following a target in its environment. Again, this is very close to the function of the Infrarat, by use of ultrasonic and infrared sensors, though only broadly mentioned in the patent.

2.0 Analysis of Patent Liability

The first patent filed by Honeywell International Inc. (USP 6829370) contains language that describes a human detection method using sensors that sense above and near the infrared spectrum. The second claim describes the spectrum that the sensors are detecting, stating specifically, "The method of claim 1, wherein the at least a portion of the upper band of the near infrared spectrum is at least a portion within the range of 1.4 μm and above in the near infrared spectrum." The Infrarat's sensors also use the infrared spectrum for human detection. The fourth

claim states, “The method of claim 1, wherein detecting reflection includes detecting reflection from the at least one region of the scene comprising a face region of the human body.” The infrared sensors on the Infrarat are aimed at both the facial area and torso/hip area for human body detection. Claim #8 is particularly important, as it states, “The method of claim 7, wherein generating data representative of the detected reflection comprises focusing the scene on a pixel array that is sensitive to the at least a portion of the upper band of the near infrared spectrum, and generating a signal representative of the spectral power for each of a plurality of pixels of the pixel array to be used for the comparison to the at least one threshold reference reflection level.” This is exactly how the Infrared sensors on the Infrarat work. They are pixel-array infrared sensors that send out a stream of pixel-data for the microprocessor to interpret. For this claim, there are literally infringing functions.

The patent held by Battelle Energy Alliance, LLC (USP 7211980) has language in the claims that suggest infringement under the doctrine of equivalents. In this patent, a robot is described which has the ability to follow a target in its environment, which includes proximity detection to avoid walls and other obstacles. The following is from the first sentence of the first claim, “A method for a robot to follow a target in its environment, comprising: receiving a target bearing from a target tracking behavior; and sensing whether the robot is blocked in a front direction and if so, then: adjusting the robot's motion to avoid a nearest obstacle in the front direction by adjusting at least one of a rotational direction, a rotational velocity, a translational direction, and a translational velocity.” This claim suggests nearly the exact behavior of the Infrarat, except for the behavior of when the robot approaches an obstacle. Based on our software plans, the robot is not to reverse direction but to change its angle and continue with its mode of operation. The second claim goes into more detail about the reversing motion of the robot when there is an obstacle, “wherein adjusting the robot's motion to avoid a nearest obstacle comprises: sensing whether the robot is blocked on either lateral side of the robot and if so, then: setting the rotational velocity to substantially near zero; and adjusting the translational velocity to a reverse direction at a third fractional amount of the maximum translational velocity.” The patent claim above would likely be infringing under the doctrine of equivalents. The Infrarat will not make a near-complete stop when it detects an obstacle, but instead change its course and try to avoid a head-on collision while also continuing the operation of whatever mode it’s in.

The third patent in question is held by the iRobot Corporation and Deere & Company (USP 8020657). The first claim suggests that the Infrarat may actually be literally infringing the patent. The language is as follows, "A robotically controlled vehicle comprising: a robotic control unit comprising a control application; a sensor for detecting object data representing detected obstacles or objects located outside of the robotically controlled vehicle and for transmitting the object data to the robotic control unit, the object data being three-dimensional data." While the Infrarat does not transmit 3-dimensional map data back through the on-board Bluetooth module, it does contain all the other components mentioned in the first claim. A robotic control unit comprising a control application would be the equivalent of our Android device, and we have multiple sensors for representing detected obstacles located outside of the robot. For the components mentioned, several main features of the Infrarat would fall under literal infringement of this patent. The last part of the claim mentions 3-dimentional map data being transmitted to a control device, which does not happen on the Infrarat. Object data is sent via Bluetooth but it is not 3-dimensional in any sense of the word. The patent also mentions the operation of the vehicle motors, described in much the same way that the motors operate on the Infrarat. Claim number 8 states, "The robotically controlled vehicle of claim 1, wherein the robotic control unit is capable of outputting control signals to a vehicle actuator." This would be an example of Literal Infringement under this claim, since under any mode of operation, the Infrarat is outputting control signals to the motors, or in this case the vehicle actuators.

3.0 Action Recommended

Since the Infrarat uses all discrete parts, the functions of the sensors themselves may just require licensing from the manufacturers of the parts, but the actual systems involved would require dramatic changes in order to fall within the realm of the Doctrine of Equivalents rather than Literal Infringement. For example the third patent mentioned, filed by the iRobot Corporation and Deere & Company, has the claim about the available modes of operation by their robot. Two of the three modes listed on the patent are functions on the Infrarat that are literally infringing on the patent. As mentioned previously, the Infrarat has three modes of operation, follow, run away from, and manual control, which in the claim language, would correspond to the 'teleoperation', and 'follow me' mode operations. To change those modes would remove core features of the

Infrarat, so if it were to go to market, a licensing deal must be reached beforehand. The patent by Battelle Energy Alliance, LLC falls under the Doctrine of Equivalents because their object avoidance system does one thing different: it reverses and changes direction when it detects an obstacle. As mentioned previously, by slightly changing the logic of our detection systems on the Infrarat to change angle rather than completely reverse direction is one way that the potential for infringement could be avoided.

4.0 Summary

There are many claims that the Infrarat does not fit in to and therefore does not infringe upon, but there are certain claims in the patents mentioned that would require some action on the part of the Infrarat team. Several important aspects of the Infrarat robot fall under the Doctrine of Equivalents and Literal Infringement. The three patents mentioned all have some similar functionality listed in the language of the patents, and the recommended actions to those infringements have been discussed. The actual systems in place for the operation of the robot are what are infringing on several claims in certain patents, rather than the physical mechanisms of the sensors themselves (except for the infrared sensor). Some of the main features of the Infrarat can avoid any infringement by slightly changing the logic in the code, such as obstacle avoidance, but there are several core features that cannot be changed, such as the control of the motors, so a licensing deal would have to be reached.

5.0 Reliability and Safety Analysis

The most complex device on our board is the microcontroller. It has 64 pins and contains a 32 bit architecture running at 60 MHz. Therefore it is likely to be one of the more common critical components to fail on our board. The analysis came out to a MTTF of 11.19 years. This is an unexpectedly short time. To improve this, we could look into a less complex microcontroller with fewer pins, and a smaller address size while still being able to run our code effectively. This is a change we could implement on a second development cycle.

AC32 Atmel microcontroller

Parameter name	**Description**	**Value**	***Comments***

C1	Die complexity	0.56	
π_T	Temperature coeff.	.029	Assume avg operating temp of 50C
C2	Pin constant	0.25	64 pin device
πe	Env constant	4.0	Grounded Mobile
πq	Quality factor	1.0	
πL	Learning factor	1.0	
Entire design:		10.2 fails per 10^6 hours	

Another critical component is the motor controller IC. Though it functions mainly as a protected H-bridge, it draws a large amount of current to route to the motors and has potential to generate high levels of heat.

Parameter name	**Description**	**Value**	***Comments***
C1	Die complexity	0.020	Assume 100-1000 Gates
π_T	Temperature coeff.	1.5	Assume max operating temp of 100C
C2	Pin constant	0.056	16 pin device
πe	Env constant	4.0	Grounded Mobile
πq	Quality factor	10.0	
πL	Learning factor	1.0	
Entire design:		2.54 fails per 10^6 hours	

The LM2675 ICs regulate voltage to 3.3 and 5.0 volts to control all on-board components except for the wheel motors. These elements are critical to device functionality and, due to their

likelihood to generate heat, have a higher concern for failure. However, analysis gives a failure rate of only 0.154 fails per million hours which is within tolerable limits.

Parameter name	**Description**	**Value**	***Comments***
λb	Die complexity	0.002	Voltage Regulator
π_T	Temperature factor.	3.7	Assume max operating temp of 100C
π_S	Elec. stress constant	0.29	
πc	Contact constr.	1.0	
πe	Environment Factor	9.0	Grounded Mobile
πq	Quality factor	8.0	Plastic
Entire design:		0.154 fails per 10^6 hours	

5.0 Failure Mode, Effects, and Criticality Analysis (FMECA)

For this failure mode analysis criticality has been split into two levels- high and low criticality. High criticality represents failures with a possibility of user harm. And low criticality represents complete failures with no external risk created. There are two major cases in this design that can cause high level criticality. One is power-ground shorts which may create dangerous amounts of heat, and unpredictable motor control, which may cause dangerous or erratic vehicle movement. An ideal failure rate for high criticality failures would be λ less than 10^{-9}. For low criticality a rate less than 10^{-6} is acceptable.

The circuit is split into several components: Battery Recharging circuit, Bluetooth circuit, I2c Bus expander, Microcontroller, Motor Controller, Ultrasonic sensor array, and Voltage regulators. The full failure analysis and effects are covered in the FEMCA tables located in Appendix F.

6.0 Ethical and Environmental Impact Analysis

Environmental Impact Analysis

The Manufacturing process of the product will induce hazardous material since printed circuit board is required. The PCB etching will require corrosive and environmental hazardous materials such as ferric chloride, copper and other acids. To reduce the environmental damage, we tried to minimize the PCB board size and trace size while routing the circuits. Another manufacturing consideration is the materials we used to fix the IR sensors mount on the servo. Epoxy putty is very commonly used for modeling and filling gaps. We decide to fix the mount on the servo with epoxy putty instead of glue because of its lower health risks and higher flexibility. It can be easily take off after fully dried.

The major concern during normal usage of our product is the rechargeable battery life cycle. We choose to use a dual cell lithium ion rechargeable battery in our design, because it is environmentally safe. However, the disadvantage of it is the relative short cell life. Over the time, cell's capacity diminishes due to the increase of its internal resistance. Another factor which can also reduce cell life is high temperature. Poor ventilation may increase temperature and shorten the cell life. In order to prolong the life cycle of batty, a dual-cell battery charge management controller chip [12] is required to ensure that the battery can be charged timely and frequently enough. We also decide not to use a fully closed package for the Infrarat so that the temperature of battery will not increase so easily. Battery packaged by insulated tape is placed on the top of the car. Another concern during normal use is the power consumption. The Infrarat will be running on 7.2V of power when in use and will be dissipating at most 300mA of current, which is about 2W. The real power consumption may be varied depend on the operation and mode chosen by users. We also consider some special situations that will waste the power by starting the program accidentally. A de-bounce function is used to solve these problems. It will determine whether the physical start button has been pressed, and then identify the validation. If the time is too long and over 8 seconds, the program will not be launched. A sleep mode will be raised by the disconnection of Bluetooth to save the power. At the end of the product's life cycle, disposal and recycle also needs to be taken into account. Many parts we used are non-biodegradable but can be recycled such as rechargeable battery, PCB board and other semiconductor components. There are some online website [13] can help people to find recycle

spot. The user manual will also come with some friendly suggestions for recycle and reuse program.

Ethical Challenges

There are two main concerns for the ethical challenges. The first one is the safety concern of the users. Since our target users are children and pets, there are some special concerns for them. Especially in follow and flee modes, children and pets try to run away or catch the Infrarat. While running, children may fall on the ground or hit some objects. Pets may bite the body of the Infrarat which is almost fully made of plastic. To prevent these kinds of situations, the user manual will mention the car speed and some suggestion for the user safety. Another concern is the accuracy of the IR sensor data send to the Android device. Since Infrarat will use 8 sensors to track a body and detect obstacles, the microcontroller will check these sensors one by one and then send the IR sensor data to the Android device. Obviously, a time delay will occur by processing the data and sending the data to the Android device. The accuracy of data received by Android device will be very time dependent. Four IR sensors are separated into 2 groups by the mount; each group has 2 IR sensors. The front sensors and back sensors will check the temperature, and then the motor controller will align the car with the servo. Proper speed of motors and servo are very important while using follow mode. Additionally, while the Infrarat is running, Bluetooth will be always connected. If it is not connected, the Infrarat will stop running. So to give a clear reminder to our users, there will be some pop-up warning on the Android device to tell the Bluetooth connection status and Bluetooth range. A considerable package design of our product also took account in these special user considerations.

A series of tests for the battery were done under different temperature and humidity to make sure our product can work properly. We also did optimization with the IR sensors and Bluetooth module to minimize the time delay. The speed of servo and motor were tested to align the car with servo fast enough. Some optimization of code reduced the time delay as well. Each sensor has 64 pixels which are separated into three zones: left zone, center and right zone. Each zone of a sensor is assigned to a priority, the zone which is the same as one checked last time has the highest priority. The priority assignment minimized the total time of tracking heat source. In addition, due to the limitation of the motor controller itself, the backward speed of each motor is

slower than the forward speed. In order to overcome this weakness, we give the front sensor a higher priority, so our product will use go forward function more often which is a way to minimize the time delay.

7.0 Packaging Design Considerations

Commercial Product Packaging

Due to the popularity of the chassis that we've chosen for the Infrarat, there are many robots that look similar but do not have the same functionality. In fact, though there are many variants of robots with ultrasonic sensors and infrared sensors, there does not seem to be any that use the more sophisticated infrared-array sensors. This is probably due to the hefty price tag of the actual sensors themselves. A product that had a similar setup as ours, but as mentioned previously, it does not appear to have any of the advanced functionality as the Infrarat has planned.

5.1 Wizard

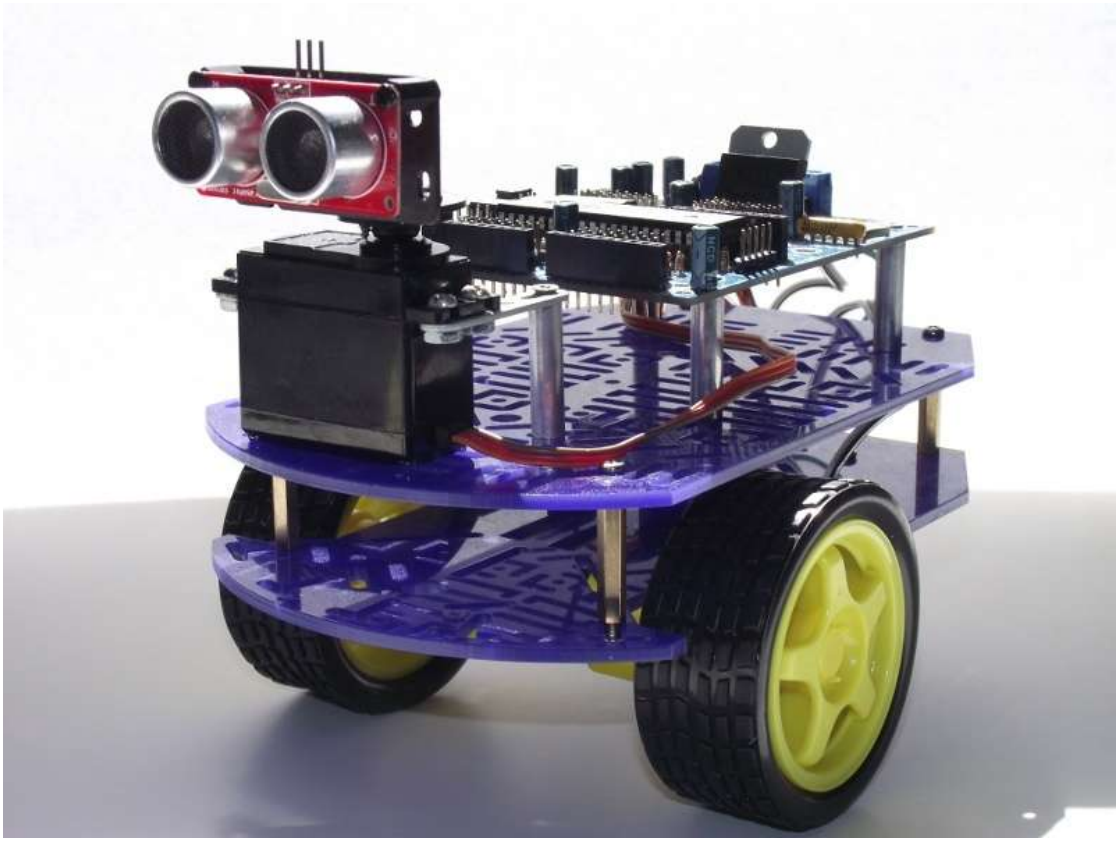


Figure 5.1: Wizard

This robot, called the ‘Wizard’, has a similar configuration as what is planned for the Infrarat. However, it does not do any of the advanced tracking that the Infrarat has to offer, instead there are two mutually exclusive options that you can get with this commercial product. One option is with an ultrasonic sensor, which is used to detect walls and prevent collision with other objects, the other option is an infrared sensor, which is only used to detect lines and guide the vehicle. As shown, it is mounted on the same ‘Magicians Chassis’, but there appears to be a lack of consideration for the balance of the vehicle. Given that our robot is meant for a slightly higher-speed application (with much stronger motors), if our servo was mounted in a similar fashion, the robot may tip over during a quick turn.

Project Packaging Specifications

As every good robot requires a reliable chassis, so does our Infrarat. In our search for a suitable platform, we came across a cheap but highly-recommended chassis package named the 'Magicians Chassis' from Sparkfun. This package features a two-level design, where two nearly identical plastic plates are stacked on top of each other, with about an inch of space available in between and on top for the critical project components. This package was chosen due to its simplicity in design and its various cutouts and mounting options.

Since the Infrarat requires quick turning without loss of balance, the servo holding the infrared sensor array will be mounted with its axis of rotation between the wheels.

PCB Footprint Layout

The initial PCB footprint was limited by the size of the chassis undercarriage. Original designs had the PCB a mere couple of inches in length with external headers strategically placed at the edges. As you will see in section 9.0, we greatly underestimated the space needed to fit our components.

8.0 Schematic Design Considerations

Theory of Operation

The circuit is split into four major subsections, Motor control, Sensor interfaces, Wireless Communication, and Power supply.

Motor control:

This section consists of a Dual H-bridge device, two high torque motors, and a servo. The H-bridge will receive PWM control signals from the microcontroller and supply power to the motors. Control signals will be at 3.3v. The wheel motor voltage supply will be at approximately 6v (Our maximum supply voltage supplied by the battery) to provide the necessary acceleration. This may be lowered to increase battery life if it does not hamper the vehicles ability to navigate quickly. The servo power supply will be 5v which is nearly its minimum allowed operating voltage.

Sensor interfaces:
Though the vehicle has many sensors, none of them are directly mounted on the PCB. The sensor interface consists mainly of traces between the microcontroller and wire plugins which will connect to the external sensors mounted on the chassis. Due to the locked I2C bus address of our IR sensors, we will be utilizing an I2C switch to expand the address space by 3 bits, allowing communication with all four IR sensors. Power from the Power regulation block will also be routed to these external interfaces. The temperature sensors run on 3.3v while the ultrasonics require 5.0 volts, requiring voltage translation for the control signals.

Wireless communication:
The wireless communication block simply consists of a self contained Bluetooth module which will be connected to the microcontroller via a UART serial connection and run at low drive 3.3 volts. The device will take a character string from the micro via uart and transmit it automatically to the connected android device, as well as receive data via Bluetooth and communicate it to the microcontroller.

Power Supply:
Power will be provided by two 3.3v lithium batteries in series. Multiple voltage regulator circuits supply 6v, 5v, and 3.3v to the rest of the circuit. A dual cell charger IC is implemented in the circuit to allow recharging of the lithium cells. To be able to monitor battery levels, a fuel gauge IC will be used and the data will be sent back to the micro.

Hardware Design Narrative
The vehicle will utilize the PWM, I2C, USART, and TC subsystems of the AT32UC3C2256C microcontroller.

The PWM subsystem is used to control both the wheel motors and the servo head. The micro will send two PWM signals to the circuit's motor control subunit and act as AIN1 and BIN1 input for the motor H-bridge [5]. This will cause the PWM signals to act as throttle to the

motors. A third PWM signal will be tied to the servo input to control the positioning of the IR servo head.

The I2C subsystem is used to interface with the four IR sensors and the battery gauge IC. The IR sensors need to be initialized by the microcontroller, and will read the sensors' internal ram via I2C requests. Slave addressing will be altered by the I2C bus extender IC due to the fixed I2C addresses of the sensors. The battery gauge IC will be on a separate bus and will read the battery status readout.

The USART subsystem will utilize its UART mode to send serial commands and data to the Bluetooth module. Data transmission will be kept at a minimal data rate to minimize power and cpu usage.

The TC subsystem is used to monitor PWM signals from the ultrasonic sensors. The Timer channels will be run in capture mode using the PWM inputs as the triggers [3]. Duty cycle can then be measured during timer interrupts.

9.0 PCB Layout Design Considerations

PCB Layout Design Considerations - Overall

The PCB design uses 2 boards: The Bluetooth board and Main board. The Main board is about 60mm*130mm which is placed on the base of the car, and the Bluetooth board contains only the Bluetooth module which is on the top of the car. The main reason why we decided to use 2 boards is that the Magicians Chassis car has a very limited space left after placing batteries on the base of the car. Second, the antenna of the Bluetooth module is sensitive to interference. [14] Therefore, to avoid placements of traces and components around the antenna, the Bluetooth module needs to be separated from the other components.

While designing the main PCB board, many considerations have to be taken into account. Our main concern is to make the device easy to debug and program. In order to accommodate this, we placed programming and debugging headers on the board, such as a USB header and a JTAG

header. Since we have a very small main board, all components on the board are placed properly to save the spaces. We initially had several isolators on our board, but further iterations removed them, allowing for a much cleaner PCB layout. We also choose 16 Mils for logic traces and 40 Mils for power and ground. For those very small pins, 12 miters traces are used to connect to other components. Due to the packaging considerations, four ultrasonic sensor headers are placed on each of the four edges of the board so that sensors can be easily placed on the car. Similarly, Four IR sensor headers are gathered together. These IR headers are bundled together and routed to the mounted IR sensors on top the servo.

The Bluetooth board is about 40mm * 60mm. On the Bluetooth board, outputs and inputs from or to the microcontroller connect to a 6 pin header.

PCB Layout Design Considerations - Microcontroller

Our main PCB design consideration for the microcontroller includes location and the size of the decoupling capacitors. Because our power supply occupies half of board, the microcontroller dose not stay in the center of board as we expected before. The datasheet recommended four decoupling capacitors of size 1nF, 4.7uF, 470pF, and 2.2uF. The decoupling capacitors are placed as close as possible to the microcontroller, resulting in them being placed on the bottom of the board opposite the micro. Most signals to the headers go underside to reduce the congestion around microcontroller. The trace size for the microcontroller is 12 Mils, since the microcontroller pins are very tiny. While laying the traces out, the PCB needs to maintain at least 12 mil trace while avoiding acute and 90 degree angles.

Several headers, such as JTAG header, used to debug, are placed in an easily accessible side of the board. There is also a reset button on the main PCB board with a pull-up resistor connected to all the reset enabled chips.

PCB Layout Design Considerations - Power Supply

The power supply part include two power regulators, one dual cell charger, one fuel gauge and a 7.2V batter supply. It occupies almost half of the main board. The Infrarat will need to supply a variety of different voltages and polarities including 5V, 3.3V from a 7.2V unregulated lithium

ion battery supply. The main board will need to be provided with 5V and 3.3V to power the systems on the PCB board. The regulator circuits are constructed on the upper half part to avoid introducing extra noise into the rest of our system and to be spatially modular for more manageable debugging.

The regulator on the left of the board outputs 3.3V and the other outputs 5V. 3.3V rail will be powering most components. 5V rail will only provide the power to the servo and ultrasonic sensors. Components are placed in groups to accommodate for different levels of power so that power traces will not run to the area where they do not needed. The battery supply is placed close to the dual cell charger to minimize voltage drops along the high current-carrying PCB traces [3]. The placement of the inductor, catch diode, and large input capacitor are the most critical parts and will be placed first.

As mentioned, all power and ground are 40 Mils wide traces, the ground rail on the topside go along the edges of board. 3.3V rail for motor controller, I2C module mostly stay on the right side of the board.

10.0 Software Design Considerations

Software Design Considerations

Two software devices that make up this project are the android device and the microcontroller. The Android device software is based around the Gingerbread release or later. It will use standard Android API as well as the Bluetooth Chat example program that can be downloaded with the Android SDK 7 or later [15]. The device runs as a master device and will only make connections if it can see the car broadcasting. Upon app startup, it will immediately begin searching for the car's broadcast signal. Once connected the manual controls and the modes can be sent to the car. The car can also send the IR data to the Android device so it can be displayed.

For the microcontroller the software gets more low level. On startup the first thing that needs to be done is setting up the clocks for the micro. The 120 MHz (RC120M) oscillator will need to be started first by writing 0x1 to the RC120MCR register in the SCIF module. The main clock is

set at 60 MHz by enabling and setting the clock source to the RC120M by writing 0x87 to the MCCTRL register. Then the CPUSEL register is set to 0x80 to enable clock division and divide it by 2 to get 60 MHz.

Once the main clocks are set up, the microcontroller can begin initializing the peripheral modules including timers, PWMs, external interrupts, I2C, UART, and GPIO. Two timers are used for the proximity sensors and will operate in input capture mode. They are used for measuring the duty cycle of the proximity sensor outputs. Both timers are set by writing a 0x1F8000 to their CMR registers. To enable the interrupts for the timers the software will need to write 0x60 to their IER register. The I2C runs master, interrupt mode and will need to run as fast as it can. This is done by writing 0x2 to the IER register and 0x1808 to the CR register. The I2C module is used to connect to the I2C bus expander to communicate with the fuel gauge and the IR sensors. The UART is run in master mode as well and will run at 115 Kbaud. The register CR needs to have the value 0x14014 and the register BRGR will need to have the value 0x20A. The UART will be used to communicate with the Bluetooth module. The external interrupt for the fuel gauge is enabled by writing 0x2 to the IER register. This interrupt will fire every time the fuel gauge detects a change of one percent in the battery charge. The GPIO pins need to be configured for whether the connected pins connect to their respective peripherals or are used as general purpose I/O. This is done by configuring the various registers for each pin. The ER and PRM1 register have a 1 when the pin is used by a peripheral function, the PMR2 register will have a value corresponding to the correct peripheral function needed. There are 8 pins that need to be set as GPIO and will all be output pins. These pins will be used for the 2 debugging LEDs, the motor controller, and the ultrasonic triggers. The PWMs is set to operate close to 300 Hz for the motors and 50 Hz for the servo. This is done by writing to the CLK register. The PWMs will be enabled by writing a 0x7 to the ENA register. The duty cycle will be updated during run-time.

The memory mappings for RAM start at address 0x0 and the flash memory starts at address 0x8000. Our program code is stored in flash and then on boot up will be copied over to RAM. We need to do this because accessing flash takes two cycles at 60 MHz which will slow down our program considerably. We want the program to execute as fast as possible so there is very

little delay in the car's reactions. For mappings of the static variables and other code pieces, the Atmel Studio C compiler determines that. Some of the static variables we will have include the IR sensor temperatures as well as the initial variables from the IR sensor used for calculations. More static variables include the ranges for the proximity sensors, the battery level, the mode flags, motor speeds, and body detection.

Our application code is mainly driven by interrupts and states. The states include flee, follow, and manual. There is also another state for follow called body detected. If the car is in this state it will have found a body and will need to follow it. If it is not in this state it will control the servo and search for a body. The follow, manual, and flee states will be used for the various modes the car will operate in. The interrupts we will use will include the receiving and transmitting interrupts for the UART and I2C interfaces and the timer interrupts for the falling and rising edges of the proximity sensor inputs. We chose to do interrupts here so the microcontroller can continue to do calculations for the temperatures and body detection. Since the data busses take a lot longer to transfer data compared to the speed of the microcontroller we will be able to do thousands of instructions between byte transfers.

From boot the microcontroller will begin initialization routines on all the subsystems. After everything is initialized, the microcontroller will enter an infinite while loop where it will execute the heart of our code. Upon entering the while loop it will first check for a Bluetooth connection. If an Android device is connected, the execution will continue, if not then the microcontroller will keep polling for a connection. Then the microcontroller will check for new data from the Bluetooth module. If there is new data changing the mode or the manual controls it will change the states accordingly. Next the microcontroller will get the new data from the IR sensors. Following that the ultrasonics will be triggered to update the range information. As the receiving I2C interrupts come in from the IR sensors, the microcontroller will save the pixel data and calculate a temperature from it. Meanwhile, the proximity sensors will be triggering the timers and the timer interrupts which the microcontroller will then calculate the distance for each proximity sensor. Once all of the calculations and data are accounted for, the microcontroller can then start sending out the temperature values over Bluetooth to the Android device. While that is being transferred, the microcontroller can begin calculations for the body detection and

direction change depending on which mode it is in. After a new direction is calculated it will be applied to the motors and the while loop will bring the execution point back to the beginning to start it all over again.

For debugging we have added a boot up self test. This will have the car move both of its wheels and move the servo. We have also added two LEDs so we can see heart beats while debugging. This will make it easy to see what is working and what is not working. One LED will be used for indicating that all data has been sent and the other will be used for indicating that the microcontroller is waiting on a connection. The latter LED will be able to tell us that the microcontroller is actually executing instead of being stuck in a buggy loop. The former LED will be used to make sure that data is being sent properly and it is not waiting indefinitely for data if there is a bug.

6.0 Software Design Narrative

Upon startup, the microcontroller will need to initialize itself first before it can do anything else. This is where the "initialize micro" module comes into play. This module includes all of the code to initialize all of the modules in the microcontroller. To speed up this process the main clock will be set first so the microcontroller is running at its optimum speed of 60 MHz. This means that the power manager and the system control interface needs to be initialized first by setting the clock to the internal 120MHz oscillator and dividing it by 2. In no particular order, the rest of the microcontroller modules will be initialized including the necessary PWMs, timers, UART bus, I2C bus, external interrupts, as well as the GPIO module. The initialize clocks module has been written but not tested and all of the other initializations have been partially written.

The next main module of the code consists of initializing the various external peripherals that need to be initialized. The Bluetooth module will be told to start broadcasting and act as a slave. The constant values from the I2C sensors will need to be retrieved and processed and the fuel gauge will need to be set up to send an interrupt signal when the battery level has changed by at least one percent. This module has not been started.

The "process peripherals" module will be placed in a while loop for operating the car. In the "get Bluetooth data" module, the microcontroller will ask the Bluetooth module if there is any data from the Android phone, if there is then the microcontroller will get it, otherwise the microcontroller will continue. The "get IR sensor data" module consists of starting the data transfers between the IR sensors. As the I2C interrupts occur, the data will be calculated in the "calculate temperatures" module and processed in the "process temperatures" module. The calculate temperatures module will include code from Melexis [16] that will be ported to our microcontroller. The process temperatures will include detecting body heat. The "trigger ultrasonics" module will include sending a trigger signal to the ultrasonic sensors. The "process ultrasonic data" module will be called when the timer interrupts are handled. The "calculate motor" module will include determining the duty cycles for the motors to change the car to the direction calculated. This module will call the set motor duty module which will set the duty cycle of the motors and move the car accordingly. The "send Bluetooth data" module will consist of starting the transfer of temperatures to the Bluetooth module. The "search for body" module will be used in follow mode which includes scanning the temperatures and looking for clumps of data that resembles body heat. If it is not seeing any body heat, the module will continue to move the servo around until body heat can be detected. If a body has been detected, it will set a variable which will hold the angle the body heat is located at from the front.

11.0 Version 2 Changes

There are many design decisions in our project which we would change if provided with a second development cycle. Below is a list of some of the major revisions we would make.

- Invest in higher quality Motors

  The original COPAL 30:1 motors we bought were fairly inexpensive, but proved to be extremely unreliable and prone to breaking. This resulted in undesirable setbacks in development time and end result.

- Utilize IR sensors with a smaller field of view

  The field of view on the temperature sensors used proved to be slightly too wide for accurate tracking. We would likely pick the lower FOV variety to allow for more focused temperature measurements at a distance.

- Increase ultrasonic sensor count

  Due to design errors, we were only able to include 2 ultrasonic rangefinders, which hampered the possibilities in the vehicle's spatial awareness.

- Invest in more aesthetically pleasing chassis

  The end result of our design, while functional, was not a pretty sight. Such a design would not be easy to market and in general hurts the image of our product.

- Begin software development earlier

  We experienced some unpleasant crunch time due to delayed software development. Making sure that we have the tools to develop code as early as possible would be a priority.

12.0 Summary and Conclusions

Over the course of the semester, we have proposed, planned, developed, and delivered our final product. The Infrarat fulfilled almost all of our initial design goals. It successfully flees from obstacles and follows heat sources, as well as successfully communicating with the android application via Bluetooth. The experience of developing a product from start to finish, with all the mistakes it entails, provided important knowledge about the engineering design process.

13.0 References

[1] Pavlidis, Ioannis (Minneapolis, MN), Symosek, Peter F. (Shoreview, MN), Fritz, Bernard S. (Eagan, MN) 2004 Near-IR human detector United States Honeywell International Inc. (Morristown, NJ) US Patent 6829370 http://www.freepatentsonline.com/6829370.html

[2] MIL-HDBK-217F Military Handbook: Reliability Prediction of Electronic Equipment. [Online]. Available: https://engineering.purdue.edu/ece477/Homework/CommonRefs/Mil-Hdbk-217F.pdf

[3] Atmel 32-bit AVR Micro Datasheet. 2012. [Online]. Available: http://www.atmel.com/Images/doc32117.pdf

[4] Maxim MAX710 Datasheet. 1997. [Online]. Available: http://datasheets.maximintegrated.com/en/ds/MAX710-MAX711.pdf

[5] TI Dual H-Bridge Motor Driver Datasheet. 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/drv8833.pdf

[6] Bruemmer, David J. (Idaho Falls, ID, US), Anderson, Matthew O. (Idaho Falls, ID, US) 2007 Robotic follow system and method United States Battelle Energy Alliance, LLC (Idaho Falls, ID, US) US Patent 7211980 http://www.freepatentsonline.com/7211980.html

[7] Allard, James (Newton, MA, US), Wienhold, Kathleen A. (Arlington, MA, US), Norris, William Robert (Rock Hill, SC, US), Catalfano, Anthony Francis (Davenport, IA, US) 2011 Systems and methods for obstacle avoidance United States Deere & Company (Moline, IL, US), iRobot Corporation (Bedford, MA, US) US Patent 8020657 http://www.freepatentsonline.com/8020657.html

[8] MIL-HDBK-217F Military Handbook: Reliability Prediction of Electronic Equipment. [Online]. Available: https://engineering.purdue.edu/ece477/Homework/CommonRefs/Mil-Hdbk-217F.pdf

[9] Atmel 32-bit AVR Micro Datasheet. 2012. [Online]. Available: http://www.atmel.com/Images/doc32117.pdf

[10] Maxim MAX710 Datasheet. 1997. [Online]. Available: http://datasheets.maximintegrated.com/en/ds/MAX710-MAX711.pdf

[11] TI Dual H-Bridge Motor Driver Datasheet. 2013. [Online]. Available: http://www.ti.com/lit/ds/symlink/drv8833.pdf

[12] MCP73213 Dual-cell battery charge management controller
Internet: http://ww1.microchip.com/downloads/en/DeviceDoc/22190b.pdf [Accessed April 11, 2013]

[13] Greener Gadgets
Internet: http://greenergadgets.org/ [Accessed April 11, 2013].

[14] RN42XVXP (2013 February 21th) Bluetooth module [online datasheet]
http://www.rovingnetworks.com/resources/download/165/RN41XV_RN42XV

[15] Google. (2013, March 22). *Get the Android SDK*. [Website]. Available: http://developer.android.com/sdk/index.html

[16] Melexis. (2013, March 22). MLX90260 *FIRray:16x4 Far Infrared Array*. [Website]. Available: http://www.melexis.com/Infrared-Thermometer-Sensors/Infrared-Thermometer-Sensors/MLX90620-776.aspx

Appendix A: Individual Contributions

A.1 Contributions of Nathan Begle:

Nathan played a major role in the general software design as well as helping interfacing some of the sensors with the microcontroller. He also played a role in laying out the PCB traces and soldering on the components when they were ready. His main role was interfacing the Bluetooth module with the microcontroller and the Android phone. He also designed and coded the Android app that was used to control the car.

During the project brainstorming and product planning, Nathan talked with the team determining the best course for implementing the idea as well as what parts would be best to use. He also played a role in determining what should be included as PSSCs and how best to approach accomplishing those goals.

During the PCB design portion of the project Nathan helped lay traces specifically for the Bluetooth board as well as helping determine where the main circuit components would go on the board. Once all of the parts and PCB had been received, Nathan helped solder many of the components on the board.

Nathan was responsible for creating the Android application and interfacing the Android phone with the Bluetooth module. He created the graphical user interface that the user would see and use making sure the design was easy to use. He was also responsible for making sure the Bluetooth connection between the Android phone and the micro was stable with low latency. He designed the analog controls in the Android app as well as displaying the Infrared temperature color gradient on the screen. He made the battery gauge and made sure the correct fuel level was being transmitted to the android phone. He also designed the interrupt process that the micro would use to transmit the data to the Bluetooth module.

During product implementation, Nathan helped brainstorm ways to get the car functioning as specified. He also helped with the implementation in the microcontroller software making sure

the idea would be implemented properly. He played a major role in implementing the flee mode as well as helping out with getting the car to follow and track a person. He implemented the manual mode which mainly required that the Android app transmitted the correct values at a reasonable speed.

During product testing, Nathan observed and tried to determine where the product was failing and helped brainstorm how to fix these problems. He helped make sure that the battery charge controller was not overheating as well as making sure it was not drawing any more current than it should have been.

For the project documents Nathan analyzed and wrote the software design considerations as well as the overall project design constraints. He also wrote the user manual for the product.

A.2 Contributions of Jordan Gaines:

Jordan contributed to many different parts of the project in different capacities. Areas include initial design conceptualization, schematic design, PCB routing, software development, and both hardware and software debugging throughout the design process.

Jordan had a role in the initial project idea, contributing the 'fleeing rat' idea, as well as the specifics of the body tracking hardware. He also aided in selecting a microcontroller family and obtaining a proper development board and routing card. His input combined with the ideas and feedback of the rest of the team had a major influence on the final design proposal.

Jordan was a major contributor to the initial schematic layouts and subsequent revisions. More specifically, he worked on the connections and pin mappings of the microcontroller, motor controller, I2C switch, voltage regulators, charge controller, and fuel gauge. He was also in charge of creating the IC package footprints for many of these components.

In addition to the schematic, Jordan was a major contributor to the second and third PCB layout iterations. Contributions included initial component layout and tracing, re-routing, header

additions, power / ground trace considerations, part footprint design, and extensive error checking. He was also tasked with implementing re-designed parts later in the project, such as voltage regulator replacement and optical isolator removal.

Jordan aided in initial component testing, such as the 3.3V and 5.0V voltage regulators and motor controller. Helped with initial motor tests and completed servo functionality tests before microcontroller programming was feasible.

Jordan worked heavily on the software component of the design. He worked on implementing initial PWM and timer code, and on debugging the problematic port to ASF libraries early in the project. He worked on ultrasonic sensor implementation and debugging, as well as the I2C communication between the IR sensors and the microcontroller. Obtaining raw IR data and calculating the temperature gradient from the IR sensor array was a major area of focus. He also contributed to the flee mode and follow mode algorithms, including IR servo tracking and motor control, as well as general debugging and aid throughout the design process.

When the project required it, Jordan worked on research and documentation, including the initial schematic presentation and conducting a safety and reliability analysis of the project design. He also worked heavily on the final report and semester report.

A.3 Contributions of Chang Yoon Kim:

A lot of the visual aspects of the project were done by Chang Yoon Kim. The website and poster were done by him, along with most of the pictures along the development of the Infrarat. The video demonstrating the PSSCs was also done entirely by Chang Yoon Kim. Much of the fitting was able to be visualized and fitted beforehand due to the accurate CAD modeling of the package. The website came as a purchasable template and was heavily altered for the team's use by Chang, especially the journals, and was kept close to the original design by Professor Meyer.

The acquisition of the infrared array sensors was made possible under Chang's suggestion that an email should be sent out asking them for a sample, though it was mentioned that the company (Melexis) does not normally do so. Had the infrared sensors not been free, four of those sensors would have cost $80 each, coming to a total of $320 which would have been over-budget and unfeasible to the project.

Around a quarter of the soldering was done by Chang Yoon Kim on the PCB, and he had a hand in making all the wires and mounting solutions in and around the packaging of the Infrarat. Specifically, the I2C circuit, most of the headers, the twizzler-like wires connected to the infrared sensors, and voltage regulator circuits. The breakout boards for the charge controller, voltage regulators, and I2C chip were acquired by Chang and their corresponding ICs soldered on to them by him for initial testing.

During the beginning developmental stage, Chang helped with the wiring of the circuits for the chips on the breakout boards so that they would be available for testing. At one point, he had driven around town during Spring Break to look for an inductor that would work with the voltage regulators.

The mounting bracket was visualized and modeled by Chang Yoon Kim, along with the actual fabrication of it. Many trips down to the metalworking shop were made, and discussion with the staff there in order for the process of fabrication to go smoothly, even though it didn't. The actual modeling of the IR mounting bracket went through 4 iterations, starting as a one-directional unit, moving on to angled, but still one-directional unit, then to a two-directional unit, then finally to the design that we have now. The bracket went through several stages of pseudo-science in order to detect at the correct angles that would intersect the warmest parts of the body. He also took the wheels down to the shop for a mounting solution to the motors.

Code-wise, he was able to contribute to a small segment of the main function, but for the most part Jordan and Nathan took the lead on that.

A.4 Contributions of Jiaqi Jiang:

Jiaqi ’s responsibility was software development, software testing and debugging through the product design. She also helped out with other aspects including component selection and PCB layout. During the design stage of the project, Jiaqi helped to select motors, voltage regulator and other components.

Jiaqi made the first PCB layout iteration. She realized the initial PBC board size was too small to for the design while making the first PCB layout iteration. The size of board was finally changed to 130mm by 160mm from 100mm by 100mm. Jiaqi also aided in routing the second and third PCB layout iterations including tracing and re-route components.

During the software development, Jiaqi implemented the initialization of PWM and GPIO. Her major task is to implement the motor controller and servo with the PWM module. She contributed to the main functions for driving motors and the initialization of servo position. She also has the responsibility to implement the fuel gauge with I2C module and make sure the fuel gauge data available for updating all the time.

Jiaqi helped to test the IR sensors and follow mode algorithm during software testing and debugging. She came out the average column temperature idea to implement the IR sensor heat tracking algorithm. She also worked on implementing and debugging the back IR sensors, optimizing algorithms of four IR sensors by testing the data of all IR sensors and helping to fix the temperature calculation mistakes as well. Jiaqi also optimized the motor movements with all IR sensors and ultrasonic sensors for the flee mode and manual mode.

Throughout the whole project every member of the team helped package the product, Jiaqi helped to place the mount on the servo. For documentation, Jiaqi wrote the PCB narrative plus preliminary PCB layout report, the ethical and environmental impact analysis report. She also helped to work on the final report.

Appendix B: Packaging

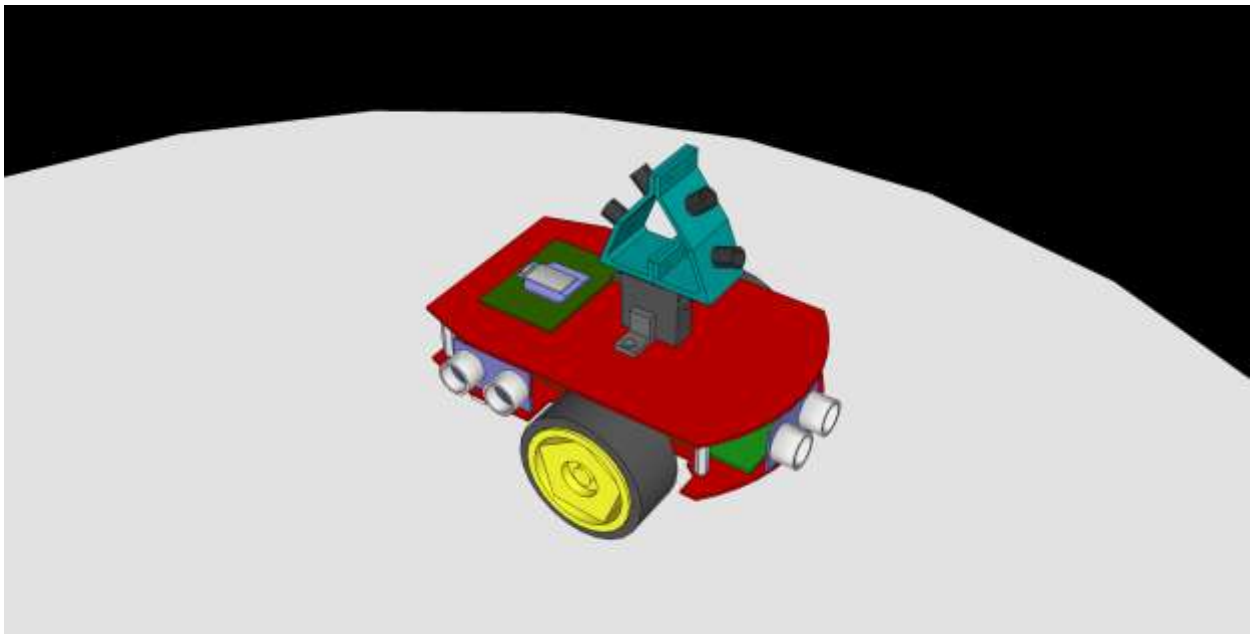


Figure B-1: Perspective view of final package

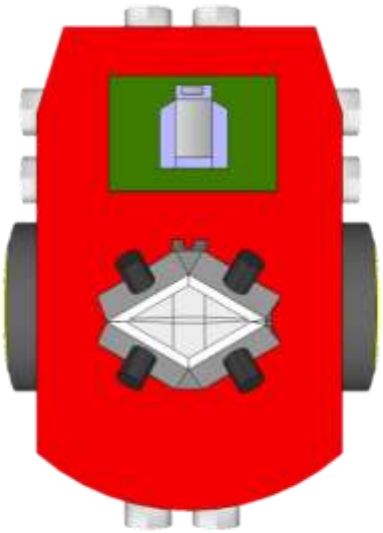


Figure B-3: Top view (beta IR mount pictured)

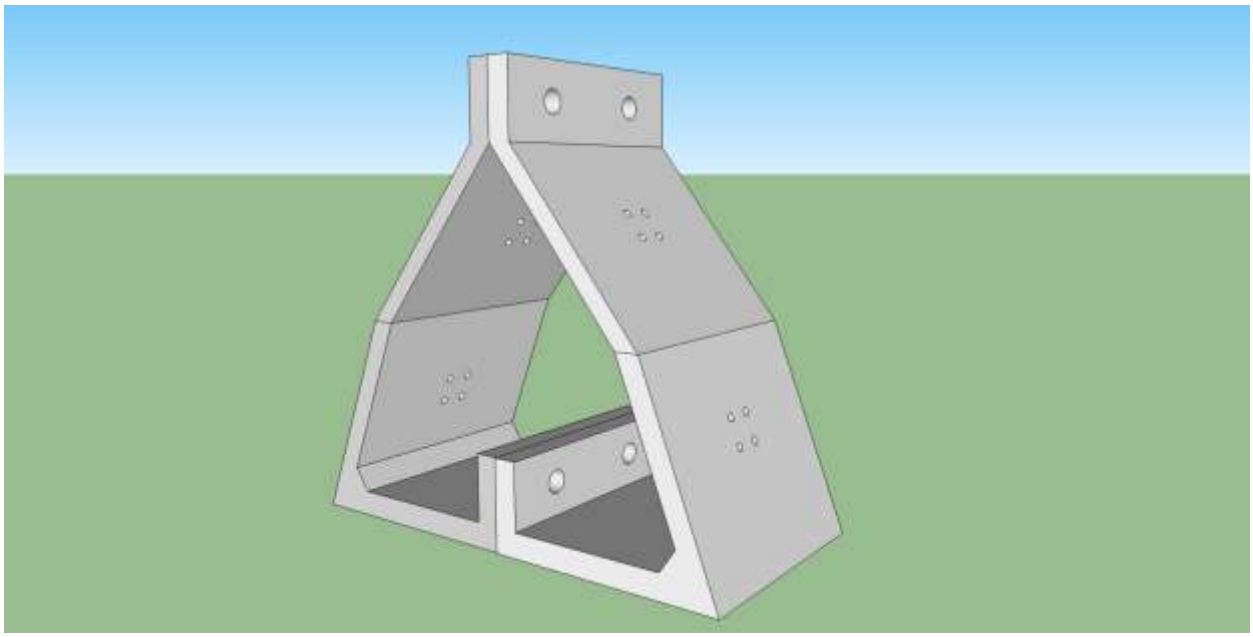


Figure B-2: IR mount

Appendix C: Schematic

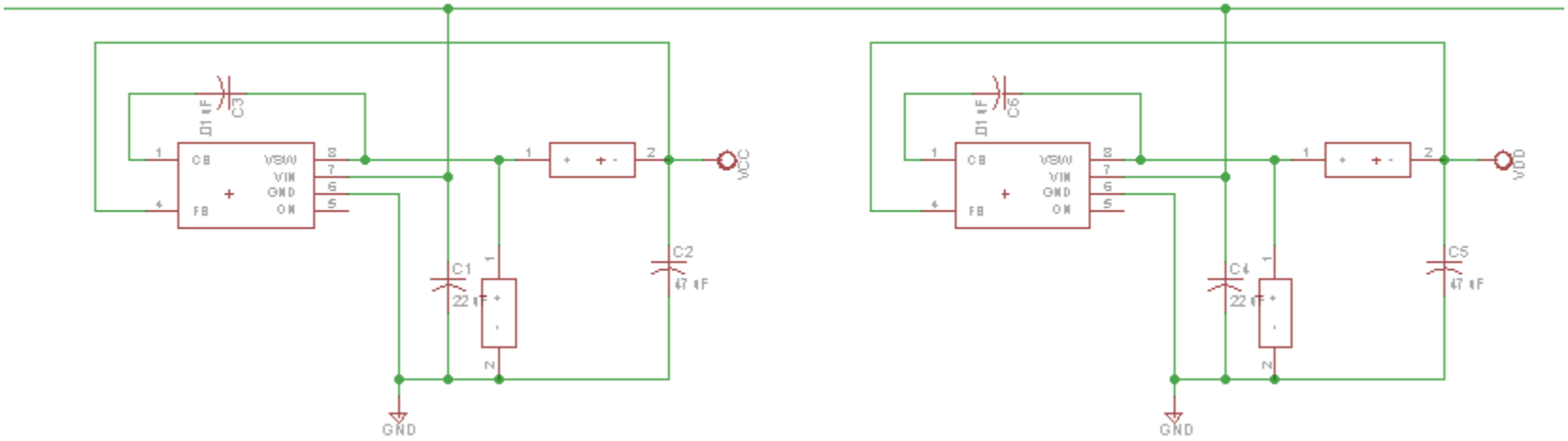


Figure C.1 Voltage regulators

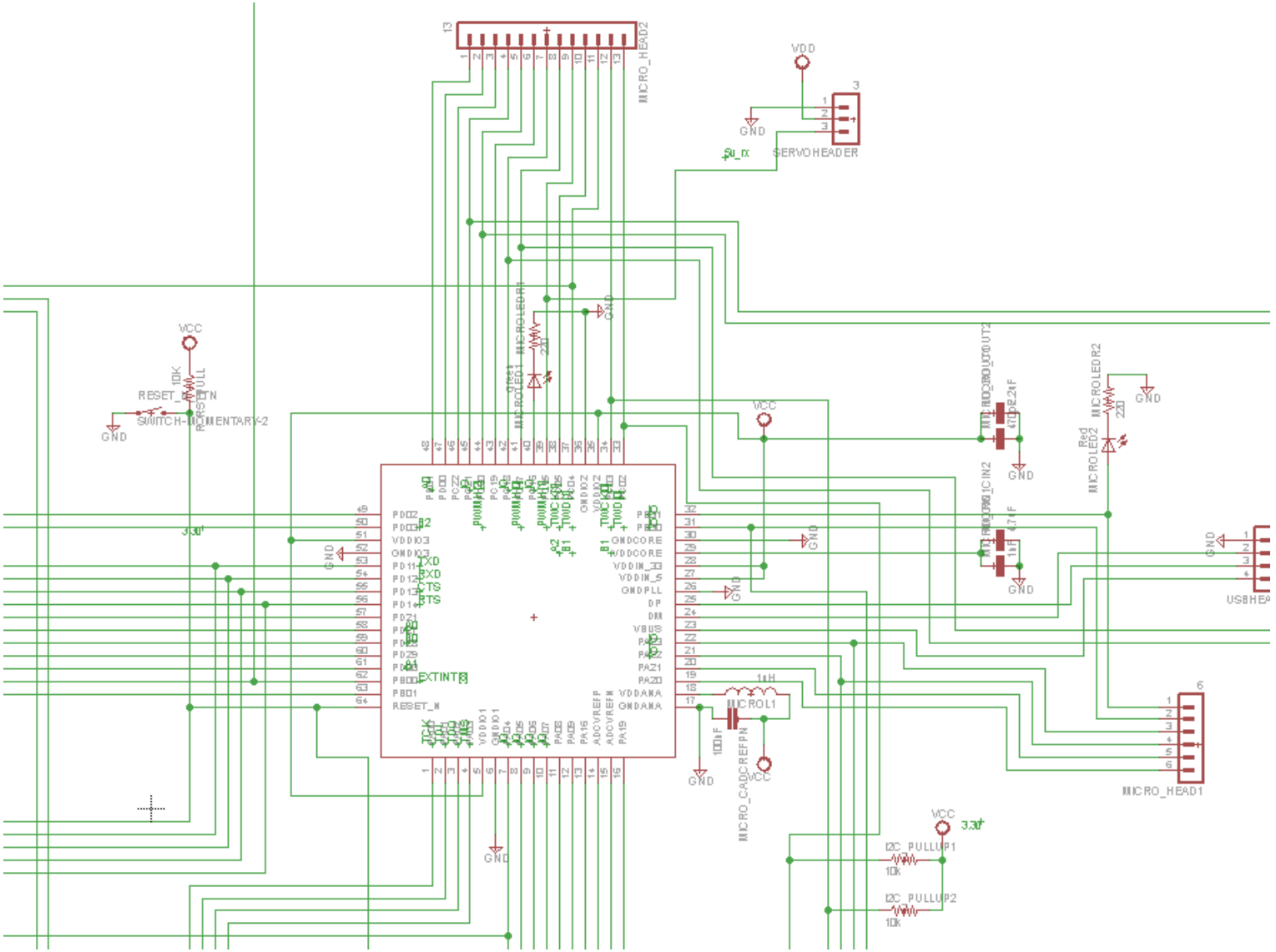


Figure C.2 Microcontroller

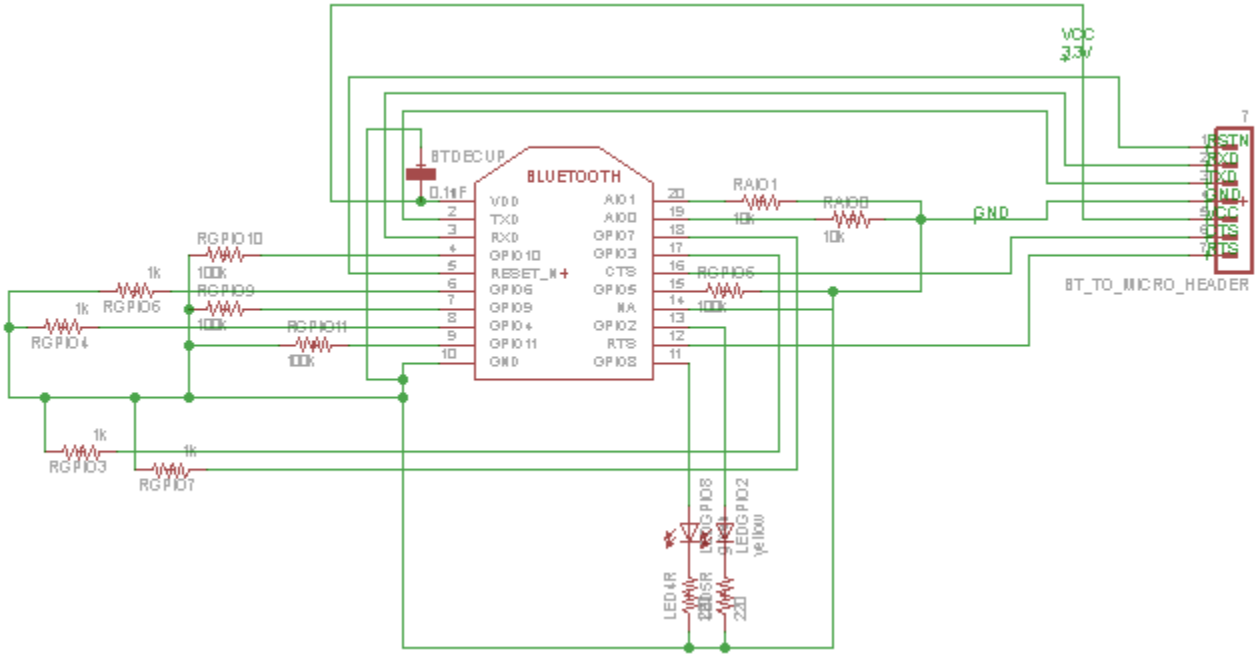


Figure C.3 Bluetooth

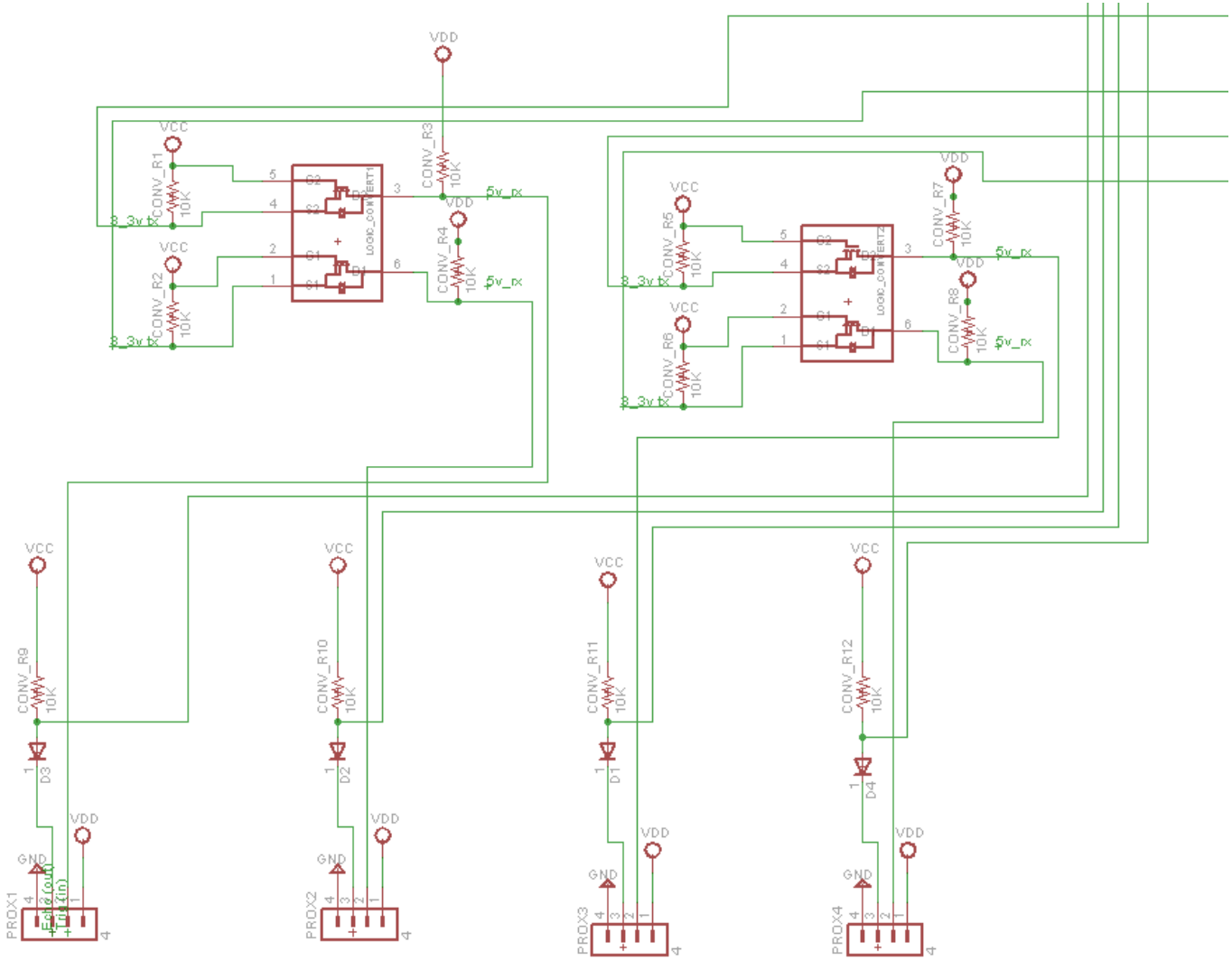


Figure C.4. Ultrasonic sensors

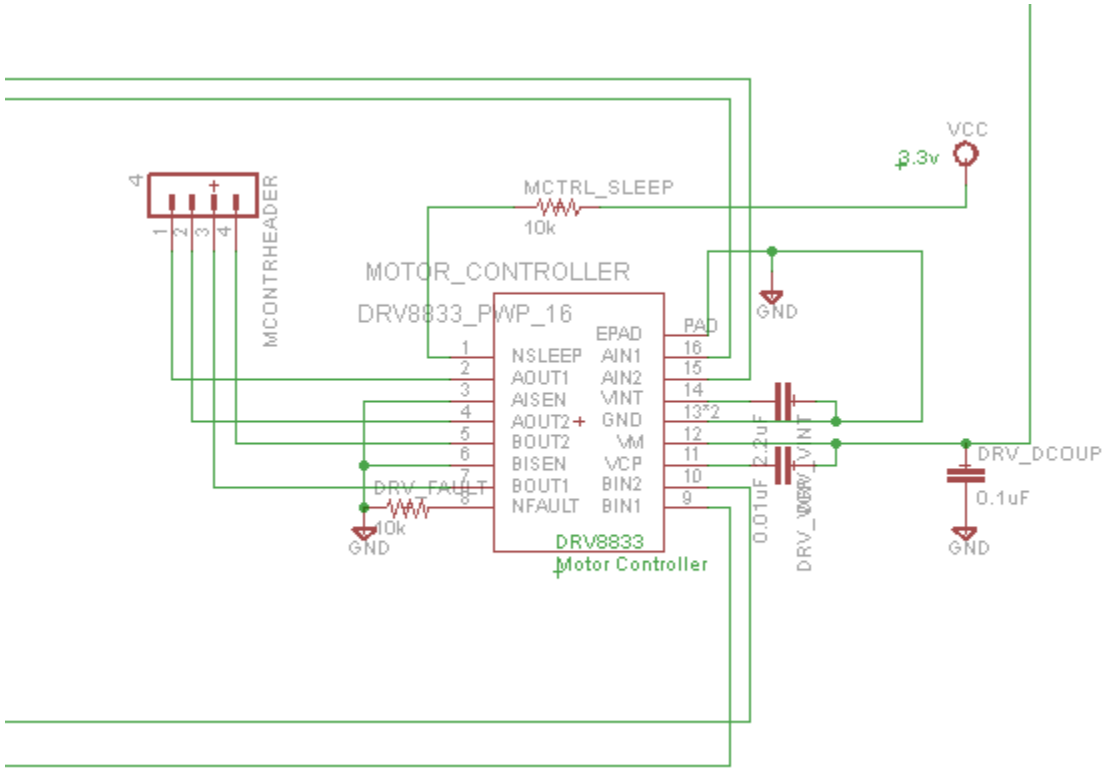


Figure C.5. Motor Controllers.

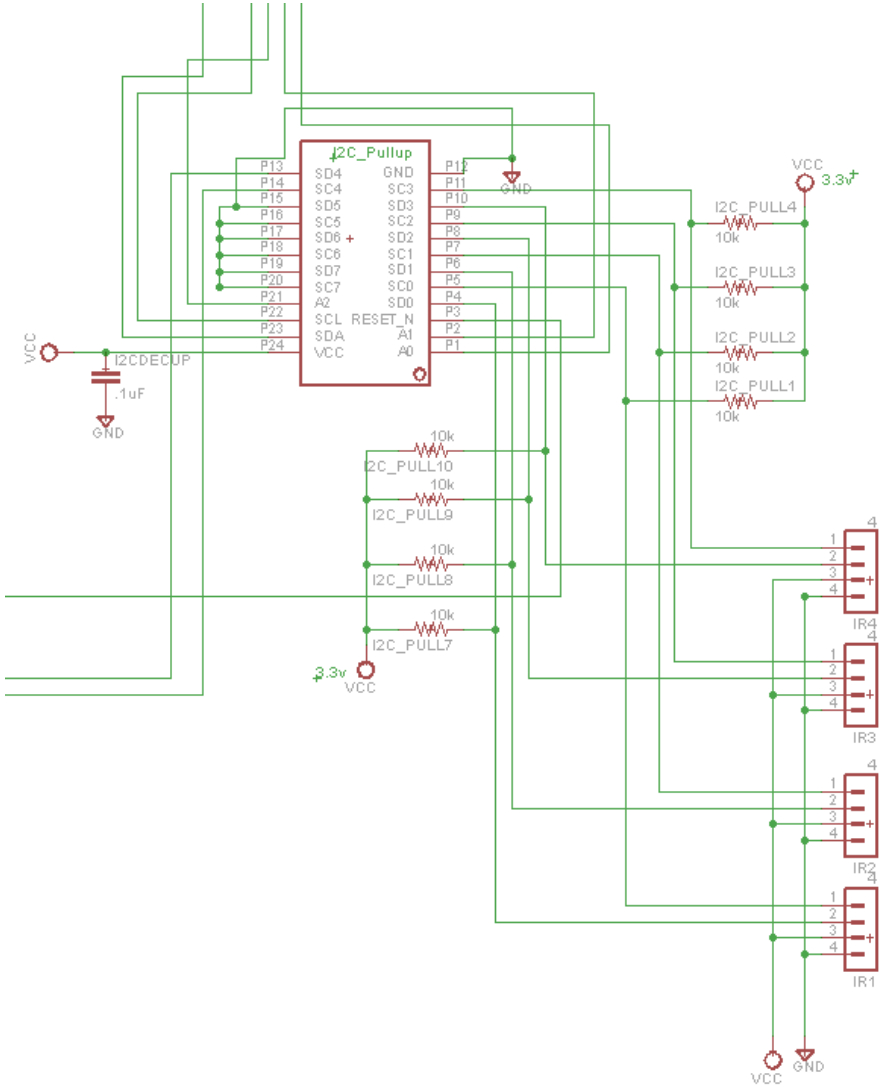
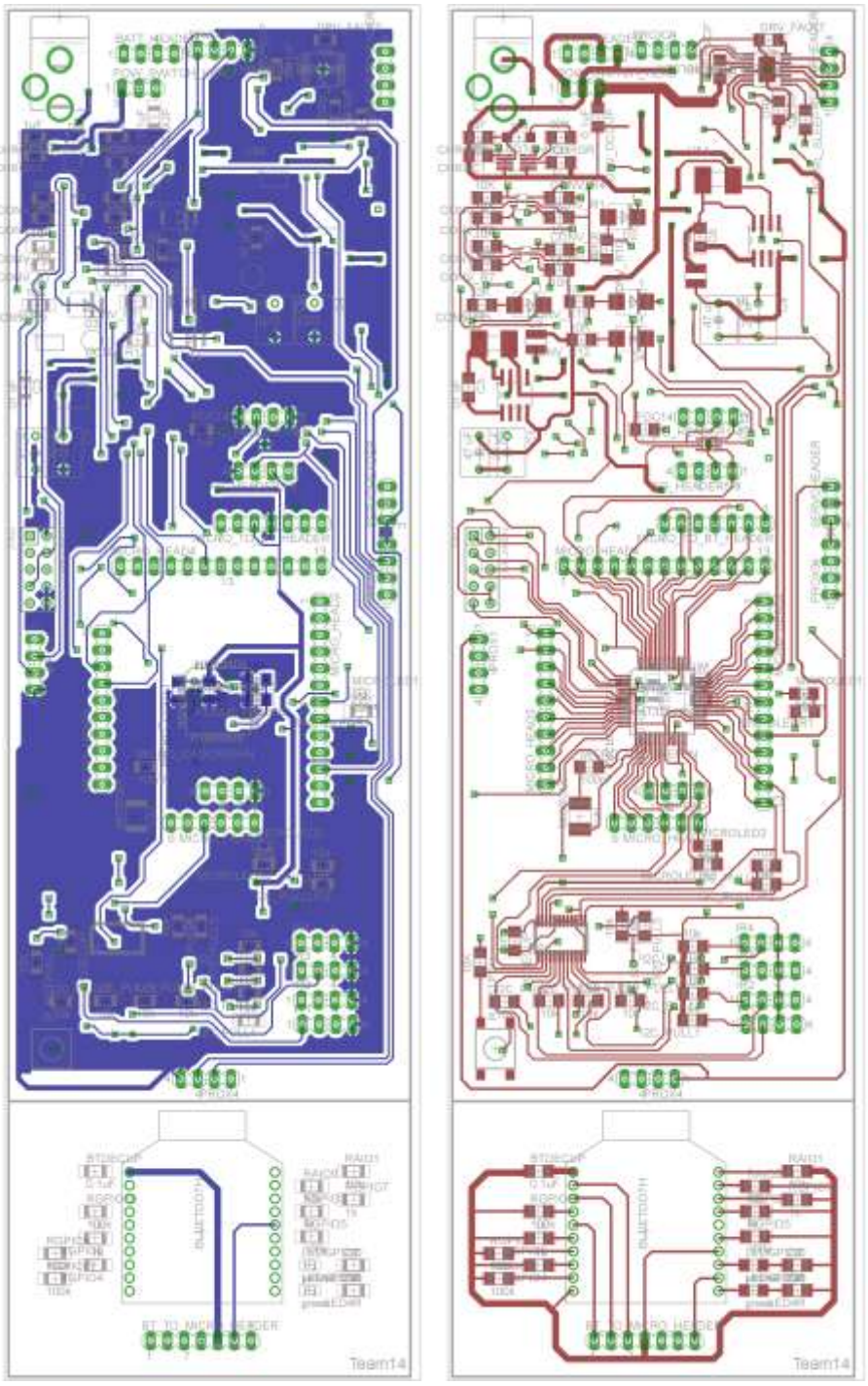


Figure C.6. I2C switch

Appendix D: PCB Layout Top and Bottom Copper



D.1 PCB Final Layout (Bottom on left. Top on right)

Appendix E: Parts List Spreadsheet

Vendor	*Manufacturer*	*Part No.*	*Description*	*Unit Cost*	**7.0**	*Total Cost*
Sparkfun Electronics	Sparkfun	ROB-10825	Car chassis	14.95	1	$14.95
Future Electronics	Melexis	MLX990620	IR array sensor	68.12	1	68.12
Amazon	ElecFreaks	HCSR04	Ultrasonic Proximity sensor	5.82	4	23.28
Mouser	Roving Networks	RN42XV	Bluetooth Module	20.50	1	20.50
Digikey	Atmel	AT32UC3L0128-AUT	32-bit microcontroller	9.24	1	9.24
Digikey	Microchip	MCP73213	Dual battery charging controller	1.94	1	1.94
Mouser	Maxim Integrated	DS2782	Fuel Gauge	6.04	1	6.04
Mouser	Maxim Integrated	MAX1616	Regulator for fuel gauge	2.00	1	2.00
Digikey	Tadiran	439-1026-ND	Lithium cell battery	12.47	2	24.94
Digikey	Texas Instruments	296-303911ND	Motor controller	2.01	1	2.01
				TOTAL		**$173.02**

Appendix F: FMECA Worksheet

Battery Recharging and monitoring circuit

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
Batteries cease to charge	Charge controller Vout fails or Battery wiring tear.	Vehicle can only be run off wall supply	No positive change in battery display	Low	
Batteries overcharge	Charge controller Vout shorts with Vsupply	Damage to battery, possibly hazardous.	Observation	High	
Charge LED held high or low	Charge controller fried	No communication of charge status	Observation	Low	
Power loss to main board	Disconnect in power switch header	Board no longer powered.	Observation	Low	

Bluetooth Circuit

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
Failure to connect to Android	Bluetooth module fries, or tear in power or ground lines	Inability to control modes of vehicle, or to visualize sensor data	Observation	Low	
UART data transfer failure	Break in UART data lines, failure in RN41 or AT32 micro	Inability to control modes of vehicle, or to visualize sensor data	Observation	Low	

I2C Expansion circuit

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
I2C Expander doesn't propagate SCL, SDA	TCA9548A internal logic failure	No communication with sensors	Observation	High	Bad sensory data can result in illogical steering.
Pull-up resistor failure	One or more pull-up resistors shorted or opened	Inability to send proper clk and data signals to I2C devices	Observation	High	

Microcontroller

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
PWM output failure	Failure of PC15, 17, 20 Output pins	Motors either disabled or stuck on full power.	Observation	High	
UART failure	Failure of PD11-PD14 output pins	No wireless communication with android device	Observation	Low	
I2C failure	Failure of PC02-3 I2C pins	Loss of IR data and fuel gauge information	Observation	Low	
Timer Input capture failure	Failure of timer channel pins	Inability to correctly measure ultrasonic sensors	Observation	High	

Motor Controller

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
Failure to control motors	Failure of AOUT/BOUT pins, or shorting of DRV_DECOUP	Unpredictable control of motors (disabled or stuck enabled)	Observation	High	
VM pulled to ground	Shorting of Decoupling Capacitor	Loss of power to the motors, and a 7 volt short in the circuit.	Observation	High	Motors will be disabled, but components may become hot to touch

Ultrasonic Sensors

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
3.3v signal to PROX	Short of voltage translation transistor	Inability to communicate with sensors	Observation	High	Sensor data needed to
Stuck voltage levels on TX/RX lines	Short across any resistor in voltage translation circuit	Inability to communicate with sensors	Observation	High	

Voltage Regulators (Applicable to either 3.3 or 5.0 volt circuits)

Failure Mode	Possible Cause	Effects	Method of Detection	Criticality	Comments
VSW pulled low	Short across Diode	No output voltage	Observation	Low	
High noise output	Short across output inductor	Possible damage to other components	Observation	Low	
VSW != 3.3v or 5.0v	Failure of regulator IC	Unreliable function of rest of circuit	Observation	Low	