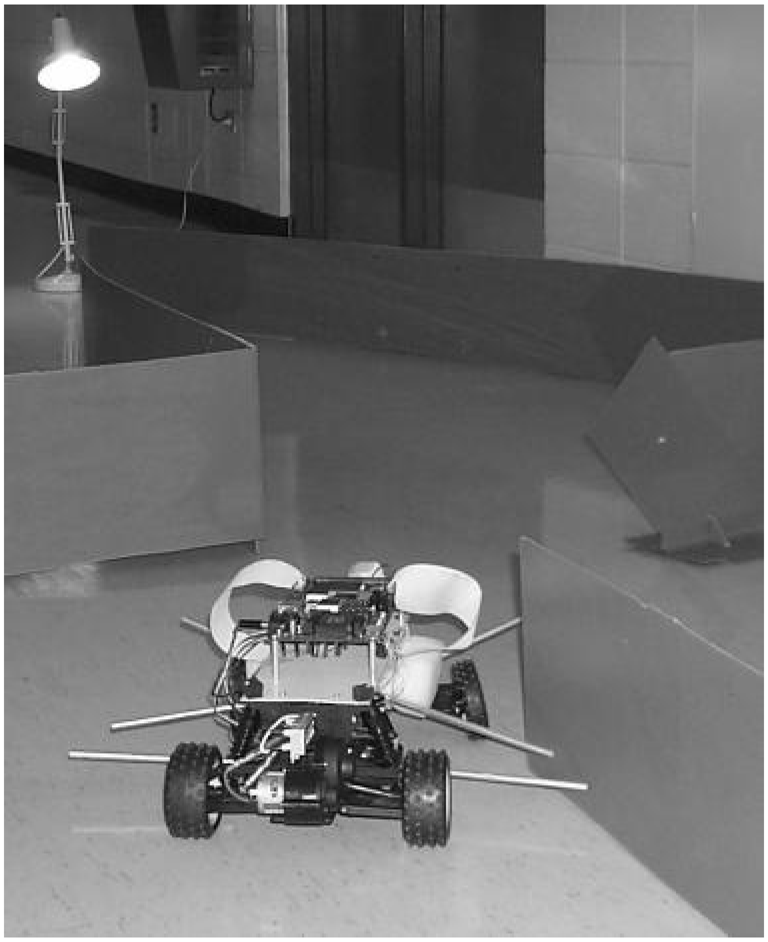
EE 552 - Final Report

December 6, 1999

Hardware Implemented Reactionary Robotic Algorithm



Bartosek, Paul	364801
Bindas, Marta	266379
Rae, Shauna	321385

Declaration of Original Content

The project and the contents of this report are entirely the original work of the authors except as follows:

Schematics for sonar circuits were modified from reference [7].

Paul Bartosek	Marta Bindas	Shauna Rae

Abstract

The final result of our project is a four wheeled vehicle with front wheel steering and rear wheel drive, which we named Hardware Implemented Reactionary Robotic Algorithm (HIRRA). It is equipped with a variety of sensors such as whiskers, sonar and photocells. These sensors help the vehicle avoid any obstacles on its path towards the brightest light in its environment, it's final destination. Once an object is detected via the whisker sensors, the car begins maneuvering around it. If the sonar detects an object the car will immediately veer away in order to avoid the object entirely. Finally if no object is in the vehicles path it will go towards the strongest light source as detected by the photocells. Once the vehicle reaches its final destination under the light, it comes to a stop.

Table of Contents

1 Achievements

We have achieved all the goals that we set for this project. Photocells detect the direction from which the strongest light is emanating. There are six whisker sensors that, upon contact, report contact with objects/walls. The sonar detects objects directly in front of the car within a range of about 86 cm. Servomotors control the speed and direction of the car quite accurately. Our VHDL code has no known bugs and, when programmed into an Altera EPF10K20RC240-4, it controls the interactions between all of these systems to create a mobile robot, which travels towards light while avoiding obstacles in its path. Total logic cells used by this system is 639. There were no features that were removed to save space on the chip, so new features could be added in the future.

2 General Description

Before going into the specifics of how this system works, it would be useful to explain what HIRRA means: Hardware Implemented Reactionary Robotic Algorithm. A reactionary robot is one that simply reacts to its environment instinctively without any memory or previous knowledge of its surroundings. The advantage to an instinctive robot is that it can react very quickly to an input stimulus versus a deliberative robot, which relies on forming a symbolic representation of its environment in order to avoid obstacles as well. Implementing this system in hardware makes it one of the fastest algorithms available.

There are three separate types of input available from the environment that our system can respond to: photocells, touch sensors and sonar. And there are two outputs available to the system: the motion of the car and the lights on the car. Four photocells are mounted on opposing corners of the car, which translates into being able to determine eight directions for the strongest light to be emanating from. Whisker, or touch, sensors tell the system when it is close to a wall on a given side, or if it has collided with an object ahead of it. The sonar detects objects directly ahead so that the car can swerve around it. Depending on the combination of inputs the control system will move the car so that it moves towards the strongest light while trying to avoid running into things. The lights provide a visible feedback as to how the FPGA is controlling the servos without the need of having the servos hooked up. This is advantageous when debugging and also looks really neat.

Determining how the system should respond to a given set of inputs is the job of the control system that is implemented on the FPGA. Since this algorithm is purely reactionary and has no memory it is really just a big lookup table. Priority is given to the whisker sensor signals, when even one of them is high the system goes into wall follow mode. In this mode the car ignores the light direction and tries to follow the wall or backup from a collision. When there is no contact read from the whiskers but there is an object detected by the sonar the car will turn either sharp right or sharp left in order to avoid it. If neither the whiskers nor the sonar detect an object the car will move in the direction from which the strongest light is detected. Finally when all of the photocells read the same light level the car will stop.

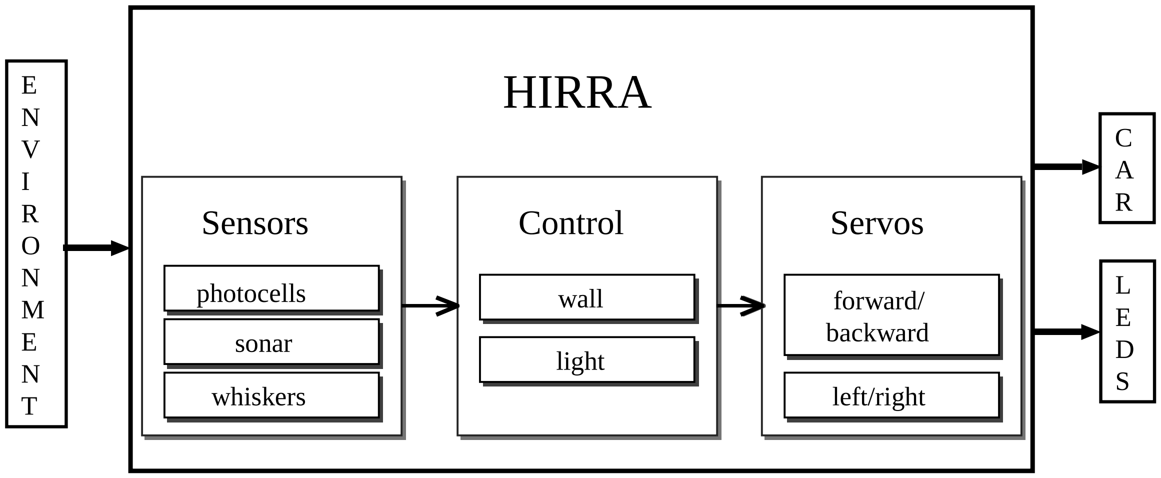


Figure 1: Data flow diagram

3 Modules and Interface

There are three levels in the design hierarchy. The top level is HIRRA. The second level consists of the Sensor module, the Control module, the Servo Control modules, and the LED display. The third level of the hierarchy consists of the components of the Sensor and the Control modules. Please refer to figure 2 below.

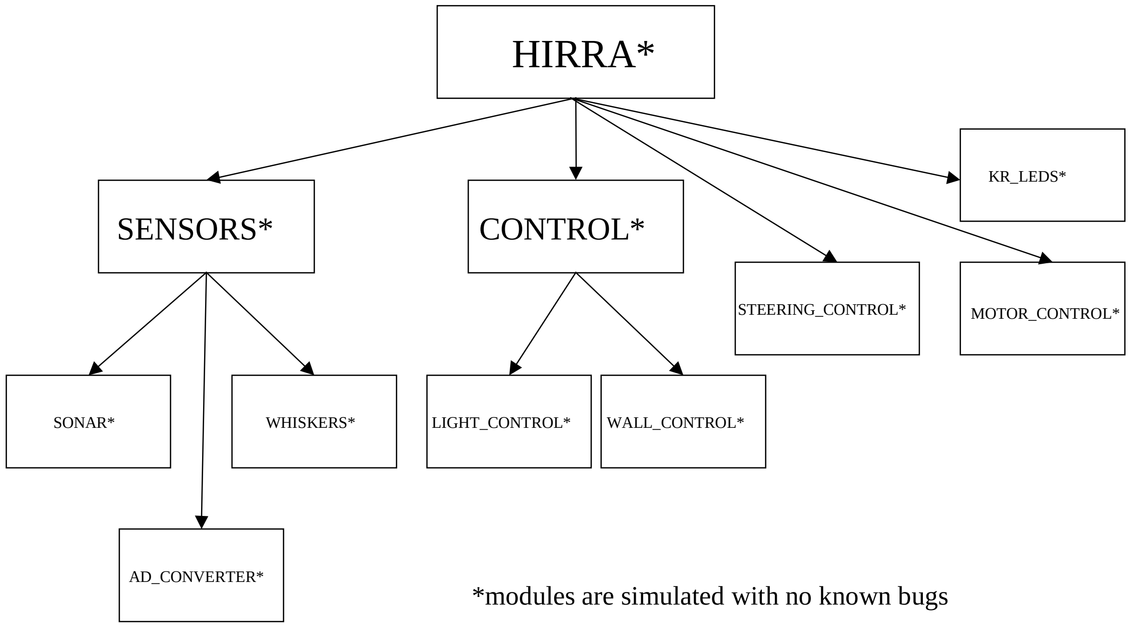


Figure 2: Diagram of Design Hierarchy

3.1 HIRRA

The top-level module is the HIRRA module. It is responsible for interfacing with all of the second level modules and for all of the input and output to and from the FPGA. All second level components are instantiated inside the HIRRA module and all required input and output signals are mapped here as well. The only other item that the top-level module is responsible for is the high-level clock divide.

3.2 Sensors

The Sensors module is responsible for interfacing the individual sensor components to the HIRRA module. The SONAR and AD_CONVERTER components are instantiated here. These components, along with the whiskers, feed raw information into the Sensor module. This module in turn preprocesses this information so that it can be easily discerned in the Control module after being passed there through the top level.

It takes information from the whiskers and outputs a *wall_contact* vector to represent if it is too close a wall, along a wall, or in collision with something as follows:

from_whiskers vector	*Whiskers in contact*	*wall_contact vector*	*meaning*
000000	none	000	no contact
000001	back right	011	wall on right
000010	right	011	wall on right
000011	right and back right	101	too close right
000100	front right	111	collision right
000110	front right and right	111	collision right
000111	all right whiskers	111	collision right
001000	front left	110	collision left
010000	left	010	wall on left
011000	front left and left	110	collision left
100000	back left	010	wall on left
110000	back left and left	100	too close left
111000	all left whiskers	110	collision left
others	any undefined combination	001	general collision

Table 1: Mapping of Whiskers to Wall Contact

It also takes information from the ADC to determine which of eight possible different directions the light is coming from or if all the light is even by comparing the intensity of the light from each of four directions. There are four different photocells used for this comparison. It only accounts for the 5 most significant bits of the eight bits representing light intensity when doing comparisons. The resulting output to the control module follow:

Results from ADC module	*Meaning*	*Output to Control*
All light equal	light overhead	stop = 1
Front Left = Front Right	light in front	light_dir = 000
Front Left = Back Left	light on left	light_dir = 001
Front Left is strongest	light in front left	light_dir = 010
Front Right = Back Right	light on right	light_dir = 011
Front Right is strongest	light in front right	light_dir = 100
Back Left = Back Right	light in back	light_dir = 101
Back Left is strongest	light in back left	light_dir = 110
Back Right is strongest	light in back right	light_dir = 111

Table 2: Determination of light Direction

It takes information from the SONAR to determine if there's an object within the threshold limit. It holds the object ahead, obj_ahead, signal high for long enough that the vehicle should maneuver around the problem area.

3.2.1 Ultrasonic Range Finder / SONAR

The ultrasonic range finder’s job is to tell the main control path when there is an object directly ahead of the car. It will also tell the motor control path to slow the car down before it rams into the object. There are two sections to the range finder: the transmitter and the receiver. The transmitter is driven directly from an output from the FPGA. The sonar is only active when it receives the enable signal from the motor servo module. While sending a pulse the FPGA outputs a 40 kHz square wave. An opto-coupler chip was included in between the FPGA and the transducer, this is necessary to isolate the transducer from the rest of the system since it put an incredible amount of noise on the power bus which interfered with the operation of the receiver circuit.

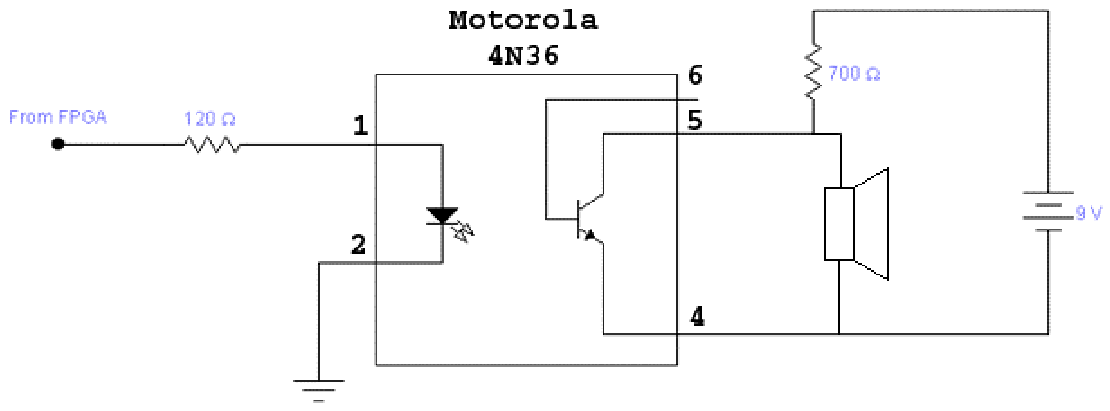


Figure 3: Ultrasonic transmitter

Detection of the returning pulse is a little bit trickier. First the signal comes from the transducer into the first op-amp stage. This serves as a combination amplifier and bandpass filter to reduce noise. The second op-amp stage is setup as an inverting amplifier, which amplifies the signal into one that will trigger the FPGA. Finally there is a clamping circuit who’s job it is to keep the voltage down to a level that will not fry the FPGA and not make us have to buy a new one to replace it. Below is a schematic of the final circuit.

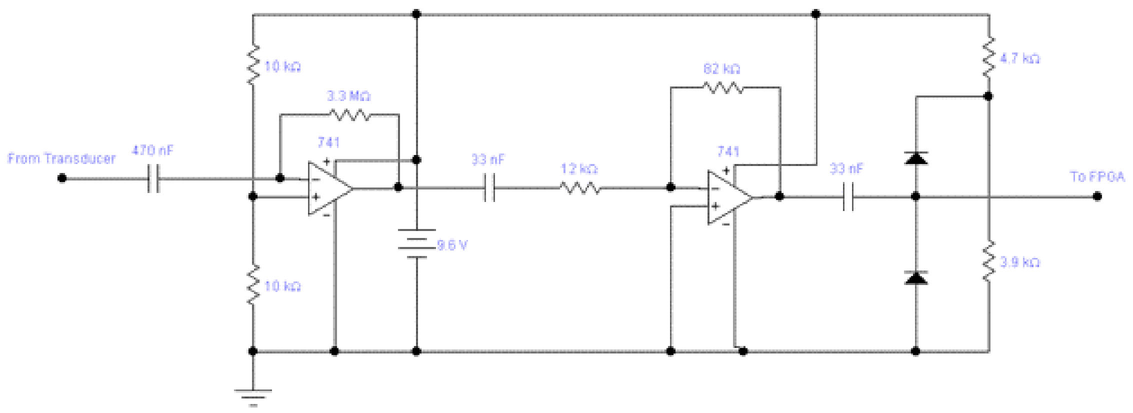


Figure 4: Ultrasonic receiver

3.2.2 Whisker Sensors

The whisker sensors are a trivial matter to construct. They currently consist of a microswitch with a straw attached to the actuator. VHDL code for our implementation of them is equally simple. Using a case-select statement we implement purely combinational logic to output the significance of any combination of whisker sensors.

A 6-bit vector represents the whiskers. Each bit is mapped to one of the whisker sensors located on the vehicle as follows:

- 000001 $\rightarrow$ back right
- 000010 $\rightarrow$ right
- 000100 $\rightarrow$ front right
- 001000 $\rightarrow$ front left
- 010000 $\rightarrow$ left
- 100000 $\rightarrow$ rear left

3.2.3 Analog to Digital Converter

The Analog to Digital converter is used to convert analog signals from the output of the photocells to digital output readable by the FPGA. The ADC requires control signals from the FPGA for correct operation. These signals include a clock, the addresses to control the multiplexer, a load address signal, a start signal, and an output enable signal. The signals that are sent to the FPGA are the end of conversion signal, or data ready, and the 8 bit digital output. Please refer to the block diagram on the first page of the datasheet for more details. Application notes from the manufacturer were consulted on proper operation.

The clock used was the same one generated in the top level module this enabled the start signal and the load address signals to be tied together because it was operating at less than 500kHz. The data ready signal operated with handshaking with the output enable line and the load address and start signal.

The following circuit was used to connect the photocells to the ADC. The op amp is used as a buffer to minimize the output impedance of the circuit since the ADC can tolerate no more than a 1kΩ source impedance. This is due to RC timing characteristics.

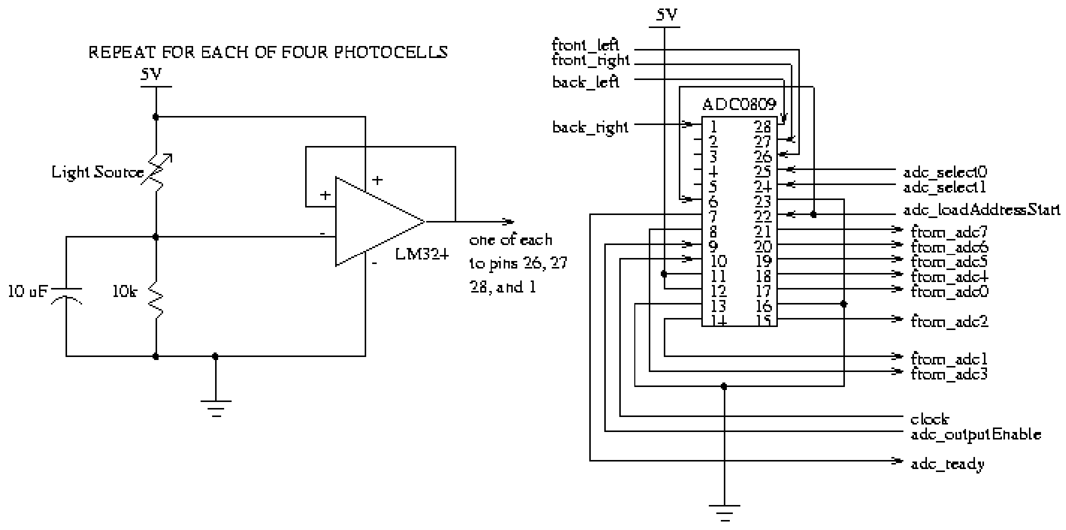


Figure 5: Photocell to ADC.

3.3 Control

The control module is responsible for interfacing the separate control modules to the rest of the system through the top level. The two components instantiated here include the *Light_Control* entity and the *Wall_Control* entity. The control module takes in the preprocessed information from the sensors and maps them to the individual control components. It uses the information about the whiskers to determine which module's output should stream to the servo controls. If there is any whisker contact, it streams the output from *Wall_Control* and vice versa. It also overrides any output from the control modules when the Stop signal goes high and outputs "zero" values to both the motor servo and the steering servo.

3.3.1 Wall Control

The *Wall_Control* module is responsible for either following walls or backing out of collision situations when there is any whisker contact. If it is backing away then it sends a signal to the Control module to tell it to continue to stream information from *Wall_Control* until it has evaded the situation. A backup timer is used for this and the timing is adjustable. The input to this module maps to the output as follows:

Input – wall_contact	***action***	***output – fwd_back***	***output – left_right***
no contact	none – light control would take over	zero	zero
general collision	backup for time and turn wheels left	back	left
wall on right	follow along the wall	fwd	zero
wall on left	follow along the wall	fwd	zero
too close right	turn away from wall	fwd	left
too close left	turn away from wall	fwd	right
collision right	back away and turn right	back	right
collision left	back away and wheels left	back	left

Table 3: Response of Wall Control to Input

3.3.2 Light Control

The Light_Control module is responsible for determining the direction of travel of the vehicle when there is no whisker contact and when the vehicle is not backing up. It takes as input the *obj_ahead* signal from the Sensor Module as well as the *light_dir* signal. The cars motion is determined by these inputs:

input – obj_ahead	***input – light_dir***	***action***	***output – fwd_back***	***output – left_right***
0	In front	Go straight forward	fwd	zero
0	In front left	Go slightly left forward	fwd	slight_left
0	In front right	Go slightly right forward	fwd	slight_right
don't care	On left	Go left forward	fwd	left
don't care	On right	Go right forward	fwd	right
don't care	In back left	Go back to point to left	back	right
don't care	In back right	Go back to point to right	back	left
don't care	In back	Go straight back	back	zero
1	In front	Go forward right hard	fwd	right
1	In front left	Go forward left hard	fwd	left
1	In front right	Go forward right hard	fwd	right

Table 4: Response of Light Control

3.4 Motor Servo

The Motor Servo module is responsible for determining the output to the servomotor controlling the cars drive motor. The servomotor requires a pulse width modulated signal, with different pulse widths representing the position of the servomotor. The servo horn turns a potentiometer that controls the current supply to the motor.

The main input to this module is the *fwd_back* signal that determines the speed of the car in the forward or reverse direction. It accepts any one of 16 possible values. The "zero" value is the one that holds the servo at a position where the car motor is stopped. A counter determines the actual width of the pulse. This counter also has a trim level, to allow fine-tuning of the position of the servo. Because minimum speed of the car is too fast for our purposes, the motor is turned on only once for every four cycles.

The servomotors interfere with the SONAR so they must be turned off, no pulse sent to them, for a SONAR reading to take place. Because of this the Motor Servo module sends a *SONAR_enable* signal to the SONAR module, when the servo is turned off.

3.5 Steering Servo

The Steering Servo module is responsible for determining the output to the servo that controls the steering of the vehicle. The servo horn moves the steering rods. Like the motor servo it requires a pulse width modulated signal to control its position.

The main input to this module is the *left_right* signal that determines the orientation of the front wheels. The mapping of the signal input to the output pulse width is the same in the Motor Servo module.

The enable signal received by this module is (global enable and not *SONAR_enable*).

3.6 Knight Rider LEDs

The Knight Rider Style LEDs were useful debugging tools. They were used to show the state of the vehicle and the changing light intensity. They showed the changing light intensity because the speed at which they switched was related to the intensity measured by the photocell pointing to the strongest light source. The representation of the states is mapped in the table below as well as the priority for the display. There were a number of inputs to this module including light intensity, fwd_back, left_right, and obj_ahead.

Priority	***State***	***Output***
1	Stopped – Light is even from all sides	All LEDs are on
2	Object Ahead – SONAR detected object with threshold	LEDs flashed at speed proportional to light intensity
3	Going straight forward	LEDs light one at a time from one end to another and then back again
3	Going forward left	LEDs light one at a time from right to left
3	Going forward right	LEDs light one at a time from left to right
3	Going straight back	LEDs light two at a time from one end to another and then back again
3	Going back left	LEDs light two at a time from right to left
3	Going back right	LEDs light two at a time from left to right

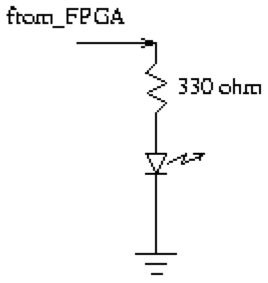


Figure 6: LED circuitry

4 Results of Experiments

4.1 SONAR

Several tests were conducted on the sonar sub-system in order to determine its operating characteristics. First of all the frequency range of the receiver had to be determined, this was done using a small speaker connected to a signal generator. This speaker was pointed toward the receiver transducer and the frequency was varied on the generator. The frequency that produced the highest signal gain was at 40.8 kHz. Half-power or 3 dB frequencies were at 41.2 kHz and 39.4 kHz. This provides a very tight reception range, which translates into good noise rejection.

Measuring the actual distance between an object and the transducers was a matter of converting the number of clock cycles between transmission and reception to a distance in centimeters. As long as the distance between the transducers and the object is significantly greater than the distance between the two transducers a proportionality constant can be used. This constant can be determined as follows:

*Dist =343m/s * clock cycles / 80000 / 2*

Assuming the speed of sound is 343 m/s and the operating clock frequency of 80 kHz, for the SONAR module, the total distance the sound travels is the speed multiplied by the time. To get the distance of the object from the vehicle, the result is then divided by two. The threshold for detecting an object ahead of the car was set at 400 clock cycles; this equates to about 86 cm.

4.2 Servo Motors

Controlling the servomotors took a bit of testing as well. First of all a servo was connected to a radio receiver that was known to be working properly with it. Then an oscilloscope was used to measure the control signal being sent from the radio to the servo. The position of the servo was varied with the transmitter and measurements were taken for a series of positions. After a few iterations of this, the position of the servo was determined to be linearly proportional to the width of the pulse being sent to the servo by the receiver. The pulse frequency is at 60 Hz and the maximum and minimum pulse widths are 2 ms and 1 ms, respectively.

After connecting the FPGA up to the servos in the car it was determined that a longer pulse width made the steering servo turn the car to the right and the speed servo move the car backwards and vice versa. This information was useful for ensuring that the car went in the desired direction.

Immediately, when the car was placed on the ground and had it go forward it was found that it went too fast even at the slowest speed. Therefore, our solution was to pulse the motor with different speeds. For 25% of the time it would receive the true signal to make it go either forward or backward. The remaining 75% of the time it would be given a signal to stop entirely, this would cause it to coast since there is no braking mechanism. Additionally this provided a window to take sonar reading without having to worry about interference from the servomotors.

4.3 Photocells and Analog to Digital Converter

The first stage in ensuring the correct operation of the light detecting sensors was to measure out the actual range of the photocells. The photocells act as variable resistors, the ones used on HIRRA vary from 5kΩ to 500kΩ, according to the specifications. Test results show that with the photocells connected as in figure 5 a voltage range of ~0V to 4.62V was obtained when varying the light from none to the brightest light available directly on the photocell. This leads to the conclusion that the range of the photocells is closer to 50Ω to 500kΩ. More importantly it was concluded that the circuit used in figure 5 would be sufficient to give a reasonable range of accuracy.

A number of tests were completed while interfacing the ADC to the FPGA. An oscilloscope was used to ensure that the ADC was receiving the desired signals. All of the signals were determined to be correct after some debugging.

In order to check that each channel was working properly on the ADC. A fixed address was sent to the ADC and the output of the ADC was fed into the FPGA and then output to LEDs. The output could have been routed directly to LEDs, but the use of the FPGA also confirmed that it was receiving the data correctly. All channels were shown to be in correct working order.

Another test was done to ensure that the output of the ADC was accurate. This was done by blocking out all light and then slowing increasing the amount of light to the photocells. As the light increase the binary value displayed on the LEDs increased one number at a time from 1 to 9. It was too difficult to control the amount of light beyond this level. This, however, shows the accuracy in the LSB’s where problems are more likely to occur.

Finally the ability for the FPGA to discern the direction of light based on the input from the photocells was tested. This was done by fixing the location of a strong light source and displaying the determined direction on one of the eight LEDs. The vehicle was rotated 360 degrees and as it was rotated the LEDs correctly displayed the location of the strongest light.

4.4 Integrated System

To verify that the vehicle would work properly, different stimuli were applied and it was verified that the output would be correct based on the LED display noted in section 3.6. This was done without connections to the servomotors. After correct operation was determined, the servos and the motor were connected and the vehicle was high mounted. This allowed monitoring that the vehicle would in fact be going in the correct direction in response to different stimulus. Once this was deemed correct, HIRRA was placed on the ground.

5 References

1. Arkin, R.C. 1998. *Behaviour Based Robotics* MIT Press, Cambridge, Massachusetts, 1998.
2. Brooks, R. 1989a. "A Robot That Walks: Emergent Behaviours from a Carefully Evolved Network," *Proceedings of the IEEE International Conference on Robotics and Automations,* May, pp. 692-94.

Web references:

3. www.csindianna.edu/robotics/stiquito.html
4. www.frc.ri.cmu.edu/robotics-faq/8.html
5. www.robotstore.com
6. www.national.com/ads-cgi/viewer.pl/an/AN/AN-247.pdf
7. www.ee.ualberta.ca/~elliott/ee552/projects/98f/2-Dmapper/