# FPGA Radar Guidance:

Final Project Proposal

Brian Plancher

November 13, 2015

## 1 Overview

Imagine that NASA's Curiosity rover suffered a catastrophic failure to its guidance and communications systems. All that remains of the system is the backup IR communication receiver. The only system close enough to communicate with Curiosity is the now stationary Spirit rover which got stuck in some soft soil in 2009 (suspend disbelief that both the rover came back on-line) which could then communicate back to earth. How could NASA control the robot's movements and keep the mission going?

This project is an exploration of a solution to that problem. Namely, I want to have the stationary device locate the position of the "rover" through ultrasound location sensing and be able to send it instructions to find its orientation and as a stretch send it to a desired location by calculating a path between the "rover" and a destination.

Given the hypothetical situation I am proposing, my design is centered around producing a reliable method to provide location and path communications. NASA would want to make sure that their design worked and ensured safe passage of the "rover" to its desired location. Given that, there are a couple of ways in which this project can be modularized in order to provide mid-way success points along the path toward the ultimate stretch goal of a feedback based pathfinding algorithm.

# 2 Design

## 2.1 Project Overview

The project can be understood as two interacting systems as shown in Figure 1. The primary system is the main FPGA system which consists of 4 main blocks: the ultrasound distance block, the orientation and path calculation block, the IR transmitter block, and the VGA display block. The secondary system is the “rover” system which consists of 2 main blocks: the IR receiver block, and the motor control block. The blocks at a high level all operate as their name implies and only do that one task. The two blocks that have a bit of complexity are the Ultrasound Block which will need to control the ultrasound module to send out signals at the right time and capture the signal and then calculate the location based on the signals captured, and the Orientation and Path Calculation Block which will need to memorize locations and use those location and optionally the goal location to calculate an orientation and calculate the best path to take toward the goal.

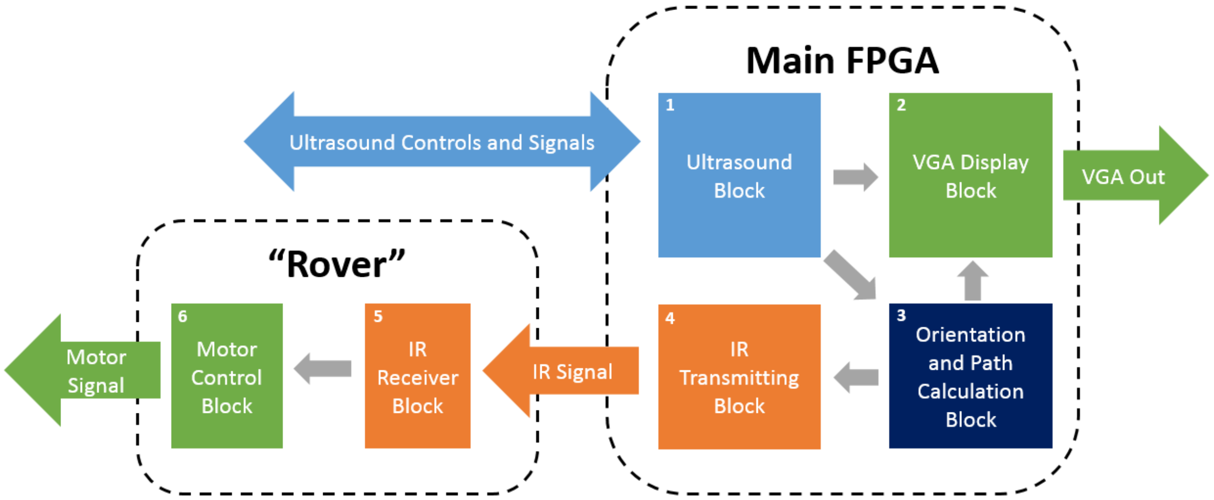

Figure 1: The high level design of the system

## 2.2 Design Decisions and Motivation

The primary motivation for this project is to explore robotics, signal processing and control systems and gain experience building systems. I am incredibly interested in the topics and am trying to pursue further graduate study in those topics. I therefore see this system as a starting point and testbed for further work. While all of the modules are necessary to complete the project, multiple modules can have their outputs hard-coded to allow for testable intermediate steps (e.g., always define the path as a straight line forward). Because I am starting from a point of limited experience with many of the technologies I am working with I made a few design decisions to increase the likelihood of completion. For example, I decided to use IR and sound communication instead of RF communication because I worked with them in earlier Labs and given the complexity of the system did not want to add another new technology. It also reduces the need for obtaining extra hardware. I am also going to work with a two-motor tank platform for the “Rover” base since I already own that platform. I also already own some Ultrasound sensors that I have been wanting to use that I am excited to take advantage of in this lab.

At the highest level, the project in its completed form is designed to allow the Main FPGA to leverage the ultrasound sensors to calculate the “Rover's” location (potentially vis-a-vis a target location) and send it a command over IR to discover its orientation (and potentially move to the desired location). The system will output VGA to show the location. Timing delays will only become important if I end up implementing the feedback control but until then most things are stable and inputs can be pipelined through as many registers as need to get the calculations done. To begin with I am going to assume away the orientation calculation and just solve the Ultrasound Block and VGA Block as all other steps are built on top of that step.

As an important note, I had originally proposed using multiple microphones and a sound source on the “Rover” to triangulate the position of the “Rover” but found out from Gim that the Nexys4 board had some limitations that would have made that implementation very difficult. Therefore I have adjusted to the Ultrasound sensor based approach.

## 3 Implementation

This section goes into more detail into each block in the system and further describes its inputs and outputs as shown below in Figure 2.

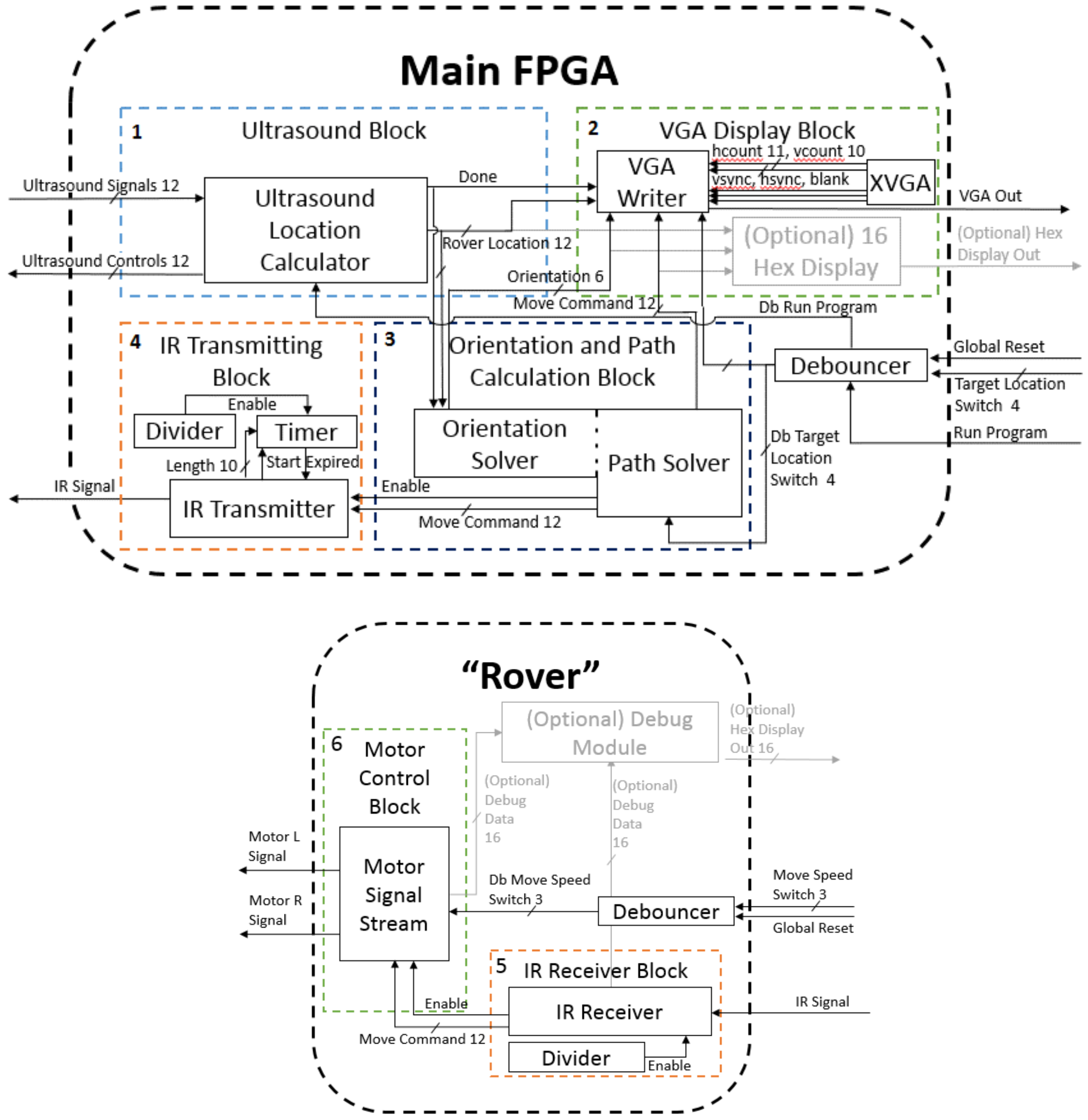

Figure 2: A detailed block diagram showing all of the links between the various modules

## 3.1 Main FPGA: Ultrasound Block

The Ultrasound Block is probably the most important block in the entire system as its accuracy will determine the precision and ability of the rest of the project. The design will be based around the specifications of the HC-SR04 ultrasound module which I will be using for the range finding. The way the module works is when it is triggered it sends out the pulse and determines the location of the nearest object in its range and then sends the echo output too high for a time proportional to the distance to the nearest location as show in Figure 3 below.

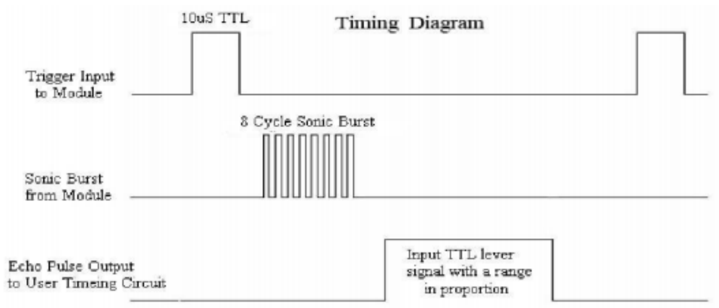
10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timeing Circuit
Input TTL lever signal with a range in proportion

Figure 3: Timing diagram of the ultrasound module *from ELEC Freaks*

The module has a field of vision of about 30 degrees from practical testing done by Cytron Technologies as show in Figure 4 below. This means that I will have to either rotate one module with a stepper motor or use multiple modules to cover a full arc from the main FPGA. Also, I will have to make sure that there are enough steps or modules to narrow down the directional angle from the main FPGA in which the rover is aligned to provide accurate instructions and positioning. Therefore, to calculate the position of the object with accuracy, I plan to take measurements every 15 degrees with a 150 degree field of vision from the source FPGA. I will then take the shortest distance received as the distance to the "Rover" $r$ and use that angle $\theta$ to calculate its position in $(x, y)$ space. Using the fact that $x = r * cos(\theta)$ and $y = r * sin(\theta)$. Since we will only use 10 pre-defined angles the *cos* and *sin* of those angles can be precomputed and stored in memory or registers for quicker executing during operation.

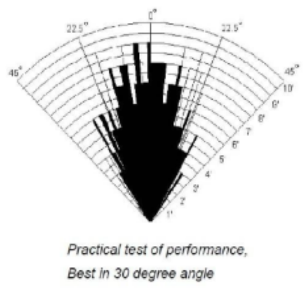

Figure 4: Practical range of the ultrasound module *from Cytron Technologies*

**Note to future implementers:** the HC-SR04 ultrasonic module operates on 5V and so I am going to need to either step up the 3.3V output from the Nexsys4 board or will need to power them through an external power supply of 5V with a regulator to ensure a constant 5Vs. Also as I discovered through a large amount of testing, some of the HC-SRO4 modules do not time out when they do not register an object in range. When this occurs they need to be power-cycled and then they will operate normally. If you do not power-cycle them they will lock out and be unusable until the FPGA is reset.

## 3.2 Main FPGA: VGA Display Block

The VGA Display Block will operate very similarly to Lab 3 and will utilize the VGA code provided to us for the majority of the functionality. There will be two tricky parts to the logic of the display. First, I want to have the display automatically adjust the scaling of the screen to account for the distances of the rover and the target (this will not be done in the initial version but is a first priority stretch goal). This will require heavy use of parameters and some simple scaling factors for everything that is displayed on the screen. Second, I want to display the orientation of the "Rover" which will be indicated by drawing an isosceles triangle for the rover pointing in the direction of its orientation. This will require some math to calculate which pixels are in range. There are a couple of different approaches I can take for this. One simple option would be to limit the orientation accuracy to a set number of degrees (e.g., every 30 digress) and then I could save a table of sins and cosines and then perform matrix multiplication to get the new end points and then calculate along the lines between the points. In any case this will take some time due to the multiplication. Luckily in all cases I can delay the update for a few clock cycles to perform the calculation as the incoming positional data is not sampled and calculated at a very high rate to begin with.

The VGA Display block will also have an optional Hex Display module which will simply be used for debug purposes and will leverage the provided code for the Hex Display.

## 3.3 Main FPGA: Orientation and Path Calculation Block

The Orientation and Path Calculation Block will be made up of one module with two modes: Orientation Mode and the optional stretch goal Path Calculation Mode. In Orientation Mode the module will first command the "Rover" to rotate 0 degrees and travel forward for a small amount of time. It will remember the initial position and then use the final position to determine the direction of the vector in which the "Rover" traveled. This direction is the orientation $\theta$ of the "Rover" with $\theta = 0$ defined as facing the right. It will relay this information to the VGA Display Block. For simplicity and to improve the speed of the program I will only calculate $\theta$ to an accuracy of 15 degrees as our range sensing will only be that accurate to begin with. In this case we can again use a table of *tan* and note that $tan\theta = (y_2 - y_1)/(x_2 - x_1)$.

Path Calculation Mode is a stretch goal, and as such, I will not go into great detail describing it. At a high level it will use vector addition to determine the correct vector that the "Rover" should travel on and then will calculate the change in angle and distance it needs to travel to move along that vector and send that information to the IR Transmitting Block. The angle $\theta$ being calculated as described above and the distance $r = \sqrt{(y_2 - y_1)^2 - (x_2 - x_1)^2}$. It will also record the previous position and command in a buffer so that after the move is completed by the "Rover" it can then re-examine what happened and adjust a scaling factor which is originally set to 1 to its outputs to adjust for overshoots and undershoots of the target. Since the "Rover" will be moving of uniform level terrain it shouldn't have to greatly adjust the scaling factor and it should quickly come to an equilibrium solution and provide accurate commands (assuming all of the electronics perform to consistent levels). Path Calculation Mode will always be preceded initially with Orientation Mode as the initial orientation is an input to the path calculation. Other even further stretch goals could include the ability to add some sophisticated path finding algorithms with obstacles to the calculation.

## 3.4 Main FPGA: IR Transmitting Block

The IR Transmitting Block will be a very simple block that encode and send the command from the Orientation and Path Calculation Block. This command will be sent over my custom protocol to the "Rover" for execution. The data will be sent as a 40kHz square wave with the start pulse lasting 2.4ms and the logical "1" lasting 1.2ms and the logical "0" lasting 500$\mu$s like in Lab5b (see Figure 5 below for example of this type of protocol).

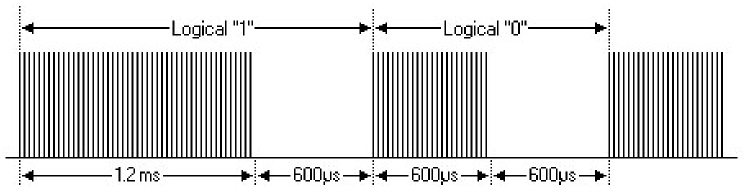

Figure 5: Timing for logical "1" and "0" over IR protocol *from Lab5b*

Like in Lab 5b, my 12 bit message is composed of a 5 and 7 bit piece of information. The 5 bit piece of information corresponds to the angle of rotation that the "Rover" needs to undergo to orient itself towards the target. The 7 bit piece of information corresponds to the distance the "Rover" needs to travel in order to reach the target. The information is sent LSB to MSB and the distance is always sent first (see Figure 6).

The IR Transmitting Block will store the data to be sent in a shift register and simply shift out the data in the correct order.

As a stretch goal I may try to implement the communication over a more sophisticated mediums (e.g., RF) that doesn't require line-of-sight for successful communication.

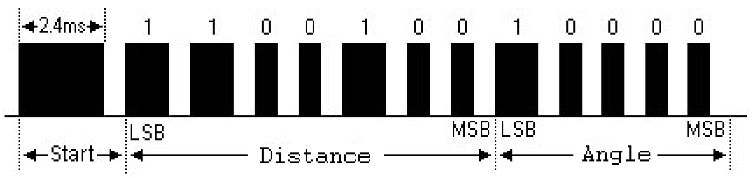

Figure 6: Ordering of data in IR Protocol *adapted from Lab5b*

**Note to future implementers:** the Lab 5b IR materials operate on 5V and so I am going to run into the same complication using the Nexsys4 as I will on the Ultrasound Block.

### 3.5 Rover: IR Receiver Block

The IR Receiver Block will operate identically to the block designed for Lab 5b as it will simply read in the message from the IR signal and store it into a 5 bit angle register and 7 bit distance register. Nothing else needs to change from Lab 5b as I retained almost all of the protocol.

As mentioned above as a stretch goal I may try to implement the communication over a more sophisticated mediums (e.g., RF) that doesn't require line-of-sight for successful communication.

**Note to future implementers:** the Lab 5b IR materials operate on 5V and so I am going to run into the same complication using the Nexsys4 as I will on the Ultrasound Block.

### 3.6 Rover: Motor Control Block

The Motor Control Block will take as input the decoded angle and distance from the IR Receiver Block and will then power the motors on the tank base for the appropriate amount of time. The motor control will be simple as the motors take in a single wire with a The measurements for the amount of time needed to go the appropriate distances will have to be determined experimentally once the project is near completion as the final force applied by the tank treads to the ground will rely on more variables than can be easily accounted for (including importantly the weight of the Nexsys4 board once it is mounted). Hopefully with a few trials a reasonable approximation can be determined and then the Orientation and Path Calculation Block logic will take over and adjust its parameters to get the "Rover" to the right final location.

**Note to future implementers:** the motors for the tank base I have may operate on 5V (further testing is needed) and so I may run into the same complication using the Nexsys4 as I will on the Ultrasound Block although there is a chance they are only 3V motors which would be nice.

## 4 Timeline and Testing

The timeline for testing and implementation is shown in Figure 7 below:

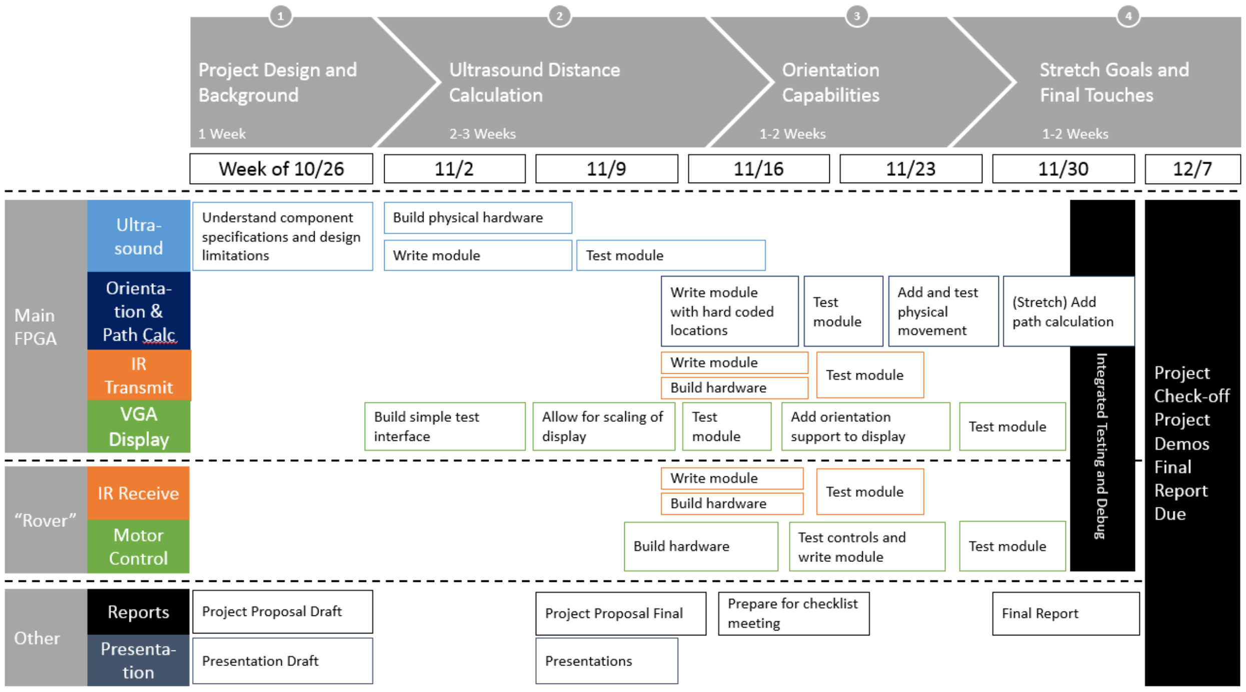

Figure 7: Proposed Schedule of Testing and Implementation

The basic testing for each module with consist of creating Verilog test benches that can be simulated with ModelSim. Combinatorial test benches will be used for the interactions and combinations of the various modules that interact. Also hex displays and the logic analyzer will be used for further debug and analysis throughout the process. User testing will also be done to ensure final real-world behaviors make sense. The VGA output can also be used to make sure that the Main FPGA is locating the objects correctly (and optionally mapping a correct path between them).

## 5 Resources

This project needs a few additional resources to be built beyond what is supplied in lab. The first main component is a “Rover” body and the Ultrasound Sensors which I happen to already have (which was part of the inspiration for the project). I will re-use the IR materials from Lab 5 to make all of the signals.

## 6 Conclusion

The FPGA Radar Guidance project will be a very interesting and fun project for me to build. I am very interested in robotics, signal processing, and control theory. I have covered many of the topics in a theoretical context but have little actual practice building systems. I am very excited to get started. Hopefully I can take the completed project and use it as a base to continue to research and experiment on new designs, control systems, and real world interactions from a robotics platform.

## 7 Bibliography

Cytron Technologies. "User’s Manual".
<https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit#>.

ELEC Freaks. "Ultrasonic Ranging Module HC - SR04."
<http://e-radionica.com/productdata/HCSR04.pdf>.

MIT 6.111. "6.111 Lab #5b."
<http://web.mit.edu/6.111/www/f2015/handouts/labs/lab5b.htm>.