University of Central Florida

Department of Computer & Electrical Engineering
Orlando, FL 32811

GRID

(Gesture Recognition Interface Device)

Pamela Garcia
Evianis Cruz
Martin Rodriguez
Landon Splitter

Table of Contents

1.0 Executive Summary

In the past few decades' technology has made some great improvements in the use and operation of personal and commercial computing devices. In fact the growth in the realm of technology has been better than any other field. With these improvements much has changed in what and how much people are able to do on a personal PC. The only thing that has remained relatively the same over all these changes is the control device for these personal machines. For the most part the mouse has not changed too much from the way it has been in the past. It has improved in the fact that it is more sleek or easier to use, faster in the response or better fitting so the user does not hurt them with extended use. Although these improvements are great it still remains that, if you've seen one mouse, you've pretty much seen them all. Well GRID is here to change that.

GRID is a Gesture Recognition Interface Device. GRID seeks to take the world of control devices to the next step using image tracking and gesture recognition to interface with the host computer. The device has been embedded within a glove that fits over the users hands and uses a camera module to track near-IR LEDs on the glove. It also has other instruments like an accelerometer and gyroscope which has been used to determine if the user has made a specific gesture, if so then it will send the commands via Bluetooth to the host computer where the command will be translated into action. Other features, which have been added, are a removable PCB design that will allow the user to use either hand as the main control device. This makes GRID usable for both right hand and left hand users.

The original desire of GRID was to design a well-fitted glove and light for the user so that there would be no issues with fatigue while using the device, which was achieved. It will function and carry out all the operations that a typical mouse would with the added features of a gesture design. What that means is there will be special "gestures" the user can do in multimedia programs to control the features of that program, as well as a few other special features for application specific actions.

GRID is the next step forward in consumer control devices, with its sleek design and universal user abilities as well as the application specific actions it will have the ability to run. As technology progresses so should the control devices.

2.0 Project Description

2.1. Motivation and Goals

The motivation behind GRID was to create an intuitive input system that would be easy and fun to use for computer applications that are not particularly keystroke-intensive, such as surfing the web and playing certain video games. Few commercial products, hobby projects, and past courses in particular inspired the group members. In particular, the group was inspired by Iron Man hand repusor gadget. In the Iron Man movie, Tony Stark lifts his arm, palm facing toward his target. This replusor device releases a powerful blast of energy, blowing his opponents off their feet. The motivation of the group is not to blow opponents off their feet but find an easy, intuitive, and enjoyable way to using the computer. The project is not something that was worked on for a period of two semesters and then forget about it but we hope this project inspire better design and eventually real-world use.

After looking into the idea of implementing a wireless-wearable mouse the team learned that two students in MIT have work on a design very similar to the idea in mind. Glove Mouse from Tony Hyun Kim and Nevada Sanchez has designed wireless gloves. Their project consists of two sport gloves an LED on the back of the index finger, picked up by a webcam to act like a cursor, along with bottoms under the index and middle fingers activated by the thumb. The group decided to expand on this idea by adding a gesture library.

Another reason the group decided to implement this project was because the different image processing techniques that would need to be researched and implemented in which the group wanted to learn to master by the end of Senior Design 2. The team has taken many classes throughout their pursuing degrees in Electrical and Computer Engineering in which theories such as infrared, wireless technology, and programming has been taught. In which the group wishes to expand their knowledge in such theories. The group members will have a chance to learn from different aspects from this project. As a team it is expected to work collaboratively to meet the project requirements and achieve measurable and significant results.

2.2. Objectives

The main goal of GRID is to create a wireless gesture input system that enables a user to use a computer by performing intuitive hand and finger motions in the air. While wearing a glove controller on the right hand or left hand, the user can move the cursor by forming a pointing gesture and click by pressing onto the push buttons with thumb. Wearing the glove device on the users desired hand allows the user to perform gestures that will be interpreted for zooming in, zooming out, and refresh through a combination of tilting the hand and touching different portions of the fingers with the thumb and index fingers. The controllers communicate to the computer wirelessly.

The glove controllers that rely on various finger and thumb touch actions. We decided to improve upon this concept by incorporating accelerometers to detect the tilt of the hand as well, thus permitting wrist movements to control the computer with a combination of intuitive gesture actions.

The project’s objectives to satisfy the goals should follow the following guidelines:

- High efficiency and low cost
- User Friendly
- Comfortable and lightweight glove

2.3. Project Requirement and Specifications

To improve the user wireless, wearable mouse experience the quality of the gloves, operation range, and battery lifetime was considered. As explained earlier the glove will have a mounted near-IR LED in the index finger when picked up by the camera module to act as cursor or gesture recognition its accuracy is of 16-pixel clutter and optimum operation range up to 15 ft. from the computer, to allow the user mobility while still operating the computer. The real time image processing will be done using an IR camera interfaced with the microcontroller via Bluetooth controlling a servo in which response time/gesture recognition should be less than 1 second. The size of the circuitry was considered with this consideration the weight of the gloves are less than 3 lbs. each glove. The battery run time on the slide on circuitry is necessary efficiently to last 8 hours, to support constant user use during the hours of operation of the gloves while requiring no more than 2 hours of charging time. Below is summary for the project requirements and specifications:

- Accuracy: 16 pixel cluster
- Optimum Operation Range: up to 15 ft.
- Weight: Less than 3 pounds each glove
- Battery Lifetime: 8 hours of continuous use
- Recharge Time: Less than 2 hours
- Response time/gesture recognition – Less than 1 second

During the implementation of the GRID design we may have choosen to revise the project requirements and specifications, but overall this project requirements should remain true.

3.0 Project Research

3.1. Image Processing

3.1.1. Overview

Image processing spans from face detection on your digital camera to manufacturing robots; which make use of computer vision technologies. In this project it was considered an Image Processing algorithms for the purpose of object tracking and recognition to track the movement of the user's hands, which will allow user interaction. This will be implemented using an FPGA.

3.1.2. Object Tracking

The purpose of video tracking is to locate a moving object in time and space using a camera. An algorithm analyzes the video frames and outputs the movement of targets between the frames. Video tracking systems consists of two major components:

- Target representation and localization algorithm- this process is totally dependent on the algorithm.
- Filtering and Data Association algorithm- this process involves prior information about the object and scene and it deals with the objects dynamics.

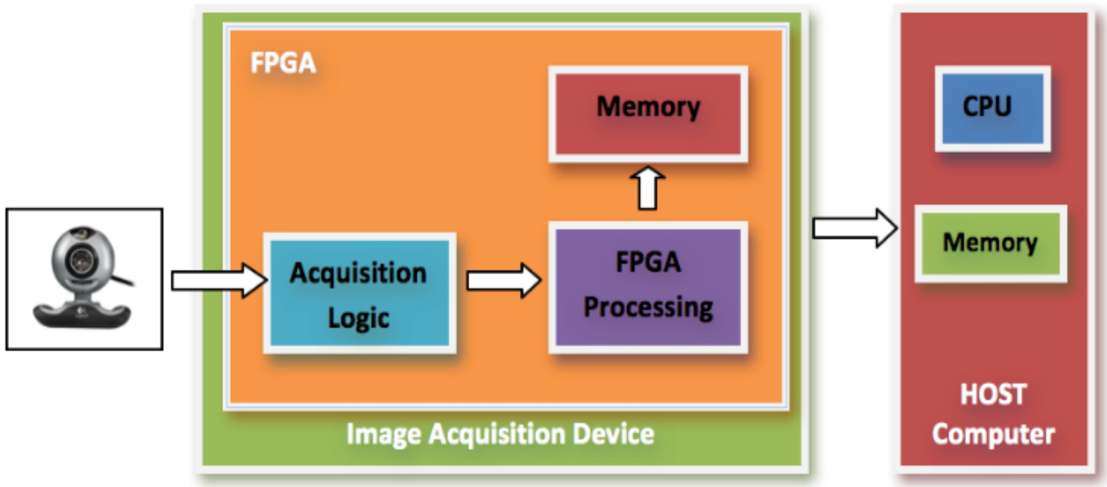
The function of the tracking algorithm is to estimate the motion parameters that are achieved by analyzing individual frames. These parameters identify several factors such as location, speed of target, direction changes, time in motion, and information about shape and size of target. The information in this section was gathered from [1].

The video-tracking algorithm is responsible for analyzing the video data and producing x and y coordinates for each LED. The algorithm works by filtering the video for pixels with the appropriate shade (red) and light. The x and y position of the pixels that are passed are then averaged to find their center of mass. The output is then smoothed to suppress the effect of noise in the video. Finally, these coordinates are transformed to fit the screen. The video will take up 720 x 480 display. The details for image processing system will be explained in greater details in this paper. This system is broken down into two sections: Image Acquisition Algorithm and Object Tracking Algorithm.

3.1.3. Image Processing System

To begin, the frames are acquired individually from the camera and fed into the FPGA. Then, the individual frames will undertake a process of segmentation, thresholding and filtering. Once the frames are subjected to this process following this, the object will be tracked by making a comparison between the background frames and the new processed updated frame containing the new location of the target. This process can be taught as comparing two frames, the current frame to our newly processed frame. As a result, tracking a moving object using an FPGA implementation was found to be appropriate for the purpose of object tracking. Figure 1 shown below shows the block diagram of the FPGA pre-processing.

Figure 1: Block diagram of FPGA pre-processing



3.1.4. Image Acquisition

The video streaming is captured using a webcam, which produces stream of RGB pixels. The Philips webcam will be mounted on a stand with a fixed background that contains the object to be track for the purposes of this project the gloves will contain NIR LEDs in the thumb and index fingers on both hands, which is the object to be tracked. The webcam will continuously stream video through the FPGA. One important parameter to consider is the frame rate vs. spatial resolution of the camera. For the purposes of this project it only requires low resolution since the camera will acquire frames at high rate. The size of the video frame will be set to 640x480 pixels that is a common format used by non-megapixel network cameras.
At the output of the FPGA the video obtained will be read in the computer using Matlab. This software will process the video and convert it into image frames at a rate of 10 frames per second.

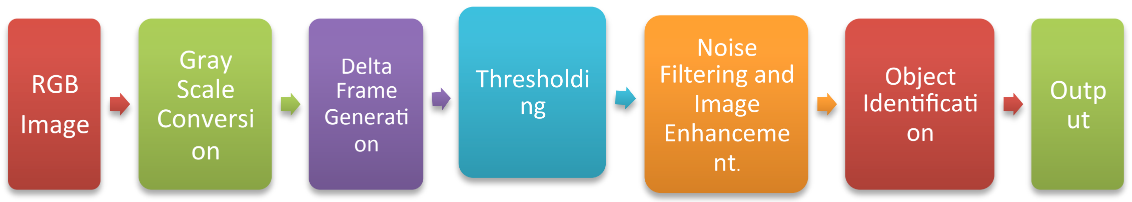
3.1.4.1. Frame Generation

At the output of the FPGA the video is fed in the Matlab program. The program reads the file and converts it to frames. The frames are produced at the rate of 10 frames per second. Considering a 10 second video when converted to frames it will be a total of 100 frames produced in RGB format. These frames are then stored as individual files 11 files that is a total of 100 files. The files are then arranged in the order of their occurrence in the video. The first frame is selected as the Base – Background Frame. The remaining bitmap files are used for the process of Object Recognition and Tracking.

3.1.4.2. Image Recognition Algorithm

This algorithm incorporates various parts that are used for object recognition such as, Grayscale Conversion, Delta Frame Generation, Thresholding, Noise Filtering and Image Enhancement. Figure 2 shown below shows the Image Recognition Algorithm Flow that will be explained in greater detail in this section.

Figure 2: Image Recognition Algorithm Flow



- **<u>Grayscale Conversion:</u>**

At the output of the FPGA passes the data through Matlab, which will convert the video into frames. Once these frames have been generated and the background and object of interest has been selected, for our purposes the NIR LEDs, these RGB format frames will be converted to gray scale. Then, a pixel value will be obtained of these RGB to gray scale format frames through a weighted sum. If an object is purely red then it will have a low value for the weighted sum therefore will appear dark in the gray scale by subtracting the gray scale image from the red component of the original, then the object can be extracted. These RGB frames have standard pixels values of 640x480 pixels. All objects fewer than 300 pixels will be removed and this will allow us to separate the infrared objects from the background. The information in this section was gathered from [1].

- **<u>Delta Frame Generation:</u>**

After, the gray scale conversion has been accomplished then, the gray scale image from the original RGB image will be subtracted. This resultant frame is called the Delta Frame. This technique of image subtraction removes the background and brings the object of interest (NIR LEDs) into attention, giving information about its shape and size. Another advantage of using this technique is that it reduces the number of pixels that the system will have to process. The information in this section was gathered from [1].

- **<u>Thresholding:</u>**

A key component to vision applications is to be able to separate out the regions of the image which corresponds to the object of interest, from the regions that corresponds to the background. Thresholding provides a convenient way to perform this segmentation provided different colors or intensities in the foreground and background regions of the image.

The input to the thresholding operation will be the delta frame gray scale image. The output is a binary image representing the segmentation between the foreground and background regions of the image. The foreground, which corresponds to the object pixel, is given a value of “1”. The background, which corresponds to the black pixel, is given a value of “0”. Then, a comparison will be made between the pixel intensity to the threshold value if the pixel intensity is higher than the threshold value, the pixel is set to white at the output. If the pixel intensity is less than the threshold value, the pixel is set to black at the output. The information in this section was gathered from [1].

- **Noise Filtering**

The medium filter will be used to reduce the noise in an image. There are two types of filter to consider when preserving the details of the image: medium filter and mean filter. For the purposes of this project a medium filter will be used because is useful in preserving the details in the image. The medium filter takes into consideration each pixel value in the image it also looks at the neighboring pixel values. Once, the pixel value and neighboring pixel values have been determine it replaces the pixel value with the medium of the neighboring pixel values. In summary, the medium filter allows high spatial frequency details to pass and at the same time is very efficient at removing noise on images. The information in this section was gathered from [1].

To conclude, this image recognition algorithm helps detect the object and gives us information about its shape and size. Then, this information will be used about the object of interest to then track the object. The object-tracking algorithm is going to be considered in section 3.1.5.

3.1.5. Object Tracking Algorithm

This algorithm incorporates various parts that are used for object tracking such as, frame rate, determining object position, and relative tracking.

- **Frame Rate:**

Using the image recognition algorithm, which provides information about the object shape and size, then it will be tracked by selecting the frames acquired from the video. A video stream will be sent via Matlab at the rate of 1 frame per second. This means the video will continue to be streaming until 100 frames are produced, which are equivalent to 10 seconds of video. The rate that has been chosen completely depends on gathering the complete motion of the object. A frame rate of 1 frame per second will be considered for the purposes of this project. A frame rate of 4 frames per second to might also be considered to reduce complications.

- **Finding Centroid of Object:**

As to finding the center of gravity, which is, used for the purposes of tracking the target, as it is a geometric property of any object, this is used in many applications. The center of gravity is taking the average location of the weight of an object. Using this method can help describe the motion of the object through space in terms of the translation of the point of the object from one place to another.

Generally speaking, trying to find the center of mass for the purposes of this project the gloves can be a bit challenging because the mass may not be uniformly distributed throughout the object. To simplify this problem it is assume that the object is composed of uniform material. The image will undertake the process of the image-processing algorithm explained earlier to acquire a noise free enhanced image. An operator then scans the entire length of the image frame for the first white pixel. This is a clear indication of the 2D position of the object within that time frame. This is iterative process and it repeated over all the frames. The information in this section was gathered from [1].

- **Relative Tracking:**

Using the update location of target in which it was considered the ideal frame rate (1 frame per second) explained earlier and a fixed background, then the frame will be subjected to a procedure for object recognition. This includes a noise free enhance image which will only contain the object. This tracking procedure will analyze each frame in search for the first white pixel, which represents the object. This point is then plotted on a new image as the first position of the object. Successive frames are gathered, subtracted from the background, filtered, enhanced and then the points are computed. This process will continue over and over in which the updated locations of the pixel points are collected to provide the approximate path taken by the object. This repetitive procedure that acquires the frames and plots the individual points in a new image is known as, object path. The information in this section was gathered from [1].

In conclusion, the object-tracking algorithm plots the objects path. This algorithm is very important to tracking the gloves position onto the screen.

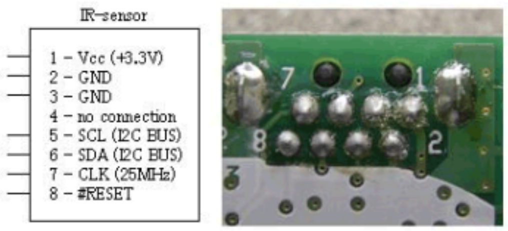
3.1.6. Wii IR Camera System

Initially, the group proposed using an FPGA for the purposes of object tracking. Due to time constraints the group look for other possibilities to track the IR-LED on the index finger. The group used the PixArt IR camera sensor, which, is commonly found in the Wii remote.

The PixArt IR camera sensor is capable of tracking up to four IR spots. Its image processing provides location data at 1024x768 resolution. The PixArt IR camera was physically extracted from the Wii remote due to the budget restrictions of the group. The IR camera is interfaced with the microcontroller via Bluetooth controlling a servo. The IR camera has an integrated processor, which output the X and Y positions and size of the near IR LEDs that it detects. The details for the near IR LEDs will be explained later in this paper. This is very beneficial for tracking the position of the cursor.

We needed a circuit to interface the IR camera to the MSP430. The parts that are needed for this circuit are two capacitors, a 25 MHz crystal, and a 1 Meg Ohm resistor. Below is the IR camera pin assignment:

Figure 3: IR Camera Pins



3.2. LEDs

3.2.1. Wavelength

Wavelength refers to the spatial period of a wave, in other words the distance over which the wave repeats itself [17]. Our eyes are only sensitive to the visible light spectrum, which goes from about 400nm to 700nm. Infrared radiation has a longer wavelength than visible red light and therefore the unaided eye cannot see it. It is important to have a basic understanding of wavelength before studying which LEDs was adequate for this project.

LEDs come in a variety of wavelengths and based on the role that this device will play in the project a decision can be made on which LED family will better suit the purpose of this project. The purpose of the LEDs on this project is to make the index finger stand out from the background noise for better object tracking. The group looked further into the infrared family of LEDs since any LEDs within the visible wavelengths will have too much interference with the background. In order to use these LEDs we a set of filters was needed, which will be explained in more detail in sections 3.3.5. Even within the infrared family, the LEDs come in a variety of wavelengths: near-infrared, short-wavelength infrared, mid-wavelength infrared, long-wavelength infrared and far-infrared. Characteristics of these infrared groups will be explained in table 1.

Table 1: Infrared LED Information

Name	**Wavelength (μm)**	**Photon Energy (meV)**	**Applications**
Near-Infrared (NIR)	0.75 - 1.4	0.9 - 1.7	Fiber optic telecommunications & night vision
Short-wavelength Infrared	1.4 - 3	0.4 - 0.9	Long-distance Telecommunications
Mid-wavelength Infrared	3 - 8	150 - 400	Guided missile technology
Long Wavelength Infrared	8 - 15	80 - 150	Thermal Imaging
Far-Infrared (FIR)	15 - 1000	1.2 - 80	Spectroscopy

In conclusion the LEDs played an important role in this project. They will make the index finger stand out from the residual background noise after the visible light/IR filter is installed. After looking at the information provided on the table above, the group came to a conclusion that the near-infrared LEDs best suits the purpose of our project since they are often used for night vision which is the closest application to the one used in this project. The data to create table 1 was compiled from [16].

3.2.2. Viewing Angle

Another factor that the group took into account when using LEDs is the viewing angle or degree. The viewing angle or degree refers to the spreading of the light emitted by the LED. For example a wide angle means that the light emitted by the LED will not travel far because it will instead spread out over a large area. A good example of this is a flashlight, these devices usually have a wide viewing angle. An LED with a small angle will have less dispersion of light and more energy in the spot. A good example of a narrow viewing angle device would be a laser. This information can be found in the LEDs data sheets and as expected it will influence in the price of the LEDs. Figure 4 below shows the spatial distribution of LEDs.

Figure 4: LED Viewing Angle- Degree

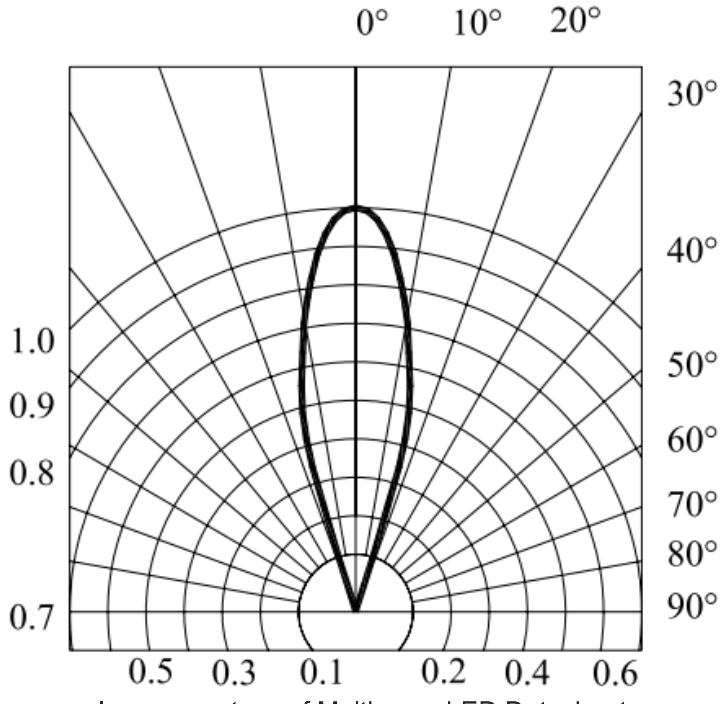


Image courtesy of Multicomp LED Datasheet.

It is evident by looking at the figure above that the viewing angle will affect the spot size of the LED. For this project we needed LEDs that have a bright spot, so each LED can be distinguished from one another. An advantage of using a visible light filter is that it will make the viewing angle of the LED less critical; it will do this by blocking the background noise. Moreover it is important to choose a narrow enough angle for the camera to recognize each LED separately as this is critical for the gesture recognition portion of this project.

3.3. Camera

3.3.1. CCD vs. CMOS

There are two major types of cameras: CCD (charge couple device) and CMOS (complementary metal oxide semiconductor). In order to chose the camera that was going to be use for this project we needed to consider the specifications of the device. Both of these cameras use a photo diode sensor, which converts light into electric charge and processes it into electronic signals. Below in Table 2 we included a summary of the features to consider when choosing a camera that best suits the purpose of our project:

Table 2:Feature and Performance Comparison between CCD and CMOS cameras

Feature	CCD	CMOS
Cost	Low	High
Fill Factor	High	Moderate
Signal out of chip	Voltage (analog)	Bits (digital)
Signal out of camera	Bits (digital)	Bits (digital)
System Noise	Low	Moderate
Speed	Moderate- High	Higher
Biasing and clocking	Multiple, higher voltage	Single, low voltage
Responsivity	Moderate	Slightly better

The content of this table is from Teledyne Dalsa [32].

Now that we have consider the feature and performance between a CCD and CMOS cameras we will go into the specifications of the camera that will achieve the purpose of this project. Later in this paper we will go into the implementation of a small camera system capable of streaming live video images from an image sensor (CCD or CMOS) to a PC through the FPGA. Below we will consider 2 types of cameras:

- HP HD-3110 5.7 Megapixel Webcam,
  - Sensor: CMOS
  - USB 2.0 Interface
  - Resolution: 352 x 288, 424 x 240, 432 x 240, 640 x 360, 640 x 480, 800 x 448, 800 x 600. 960 x 544, 960 x 720, and 1280 x 720
  - Transmission Frame Rate: 30frame/sec
  - Dimensions (L x W x H): 2.93 x 2.76 x 2.58" / 7.44 x 7.01 x 6.55 cm
  - Unit Weight: 0.22 lb / 0.1 kg

- USB Digital PC Web Camera with Microphone CMOS VGA sensor features:
  - Sensor: CMOS
  - Frame Rate: 30 FPS
  - Lens: High definition 3P(1.3M/2.0MP); 5P(5.0 MP)
  - Pixels: 5.0-megapixel
  - Resolution: 320 x 240, 680 x 480, 600 x 800, 1280 x 1024

- Interface: USB 2.0
- Transmission Frame Rate: up to 30frame/sec
- Video Format: 24-bit true color
- Supported operating system: Support Win98/2000/XP/Vista/Win7/Mac system (XP, Vista and Win7 support plug and play)
- Focus range: 3cm to infinity focus range
- CMOS Chip Type: High-resolution Color CMOS image sensor
- Unit weight: 66.3g

It was also considered using an image sensor for getting the data (image frames) ready for calculation in the FPGA. The reason we considered using an image sensor is that using the USB on the FPGA can be least practical thing to do. When using a webcam we needed to make sure to have an open-source driver for the webcam. Webcams use proprietary, undocumented, protocols and compression in which we didn't want to waste our time doing reverse engineering to the webcam.
It was also considered using an image sensor for getting the data (image frames) ready for calculation in the FPGA. The reason for considering using an image sensor is that using the USB on the FPGA can be least practical thing to do. When using a webcam it is important to make sure to have an open-source driver for the webcam. Webcams use proprietary, undocumented, protocols and compression in which time can be we wasted trying to do reverse engineering to the webcam.
It was also considered using a VGA camera because is the most basic camera compared to the megapixel camera, which has more pixels, meaning more colors and details. The image sensor we considered is the Agilent ADCS-2021:

- CMOS Image Sensor
- VGA Resolution – 640x480
- Frame Rate – 15 frames per second

In addition, the data bus can be configured as 8 to 10 bits wide. All of its functionality is controlled via internal registers, which are accessed via an I^2C bus. The contents of this section was gathered from [33] and [34].

3.3.2. Resolution

Resolution is how pixels are stored in computer memory, they may be thought of as mathematical points. Once image is displayed, the space between the points needs to be filled in. The number of spots that can be resolved on the display depends on the quality of the system. For our purposes we didn't require a camera with high resolution (640x480). We considered various resolutions for the purpose of this project and mapping its centroid to the location of the screen.

3.3.3. Frame Rate

The frame rate of a camera is the number of pictures it can take and how quickly it can transfer those to the computer's screen. The frame rate is constrained by three factors: the rate at which the screen is refreshed, specified frame rate and the time required to calculate and draw the scene [11]. Frame rate is measured in frames per second. For this project we needed to stream video from the webcam and to do so we had to consider a camera with a decent frame rate, 15 fps is the absolute bare minimum. Most cameras that have been considered within the budget of this project stream at 30 fps, that means it can take 30 pictures every second and then transfer them to the computer screen or in our projects case the FPGA, which is a good rate.

Figure 5: Frame Rate

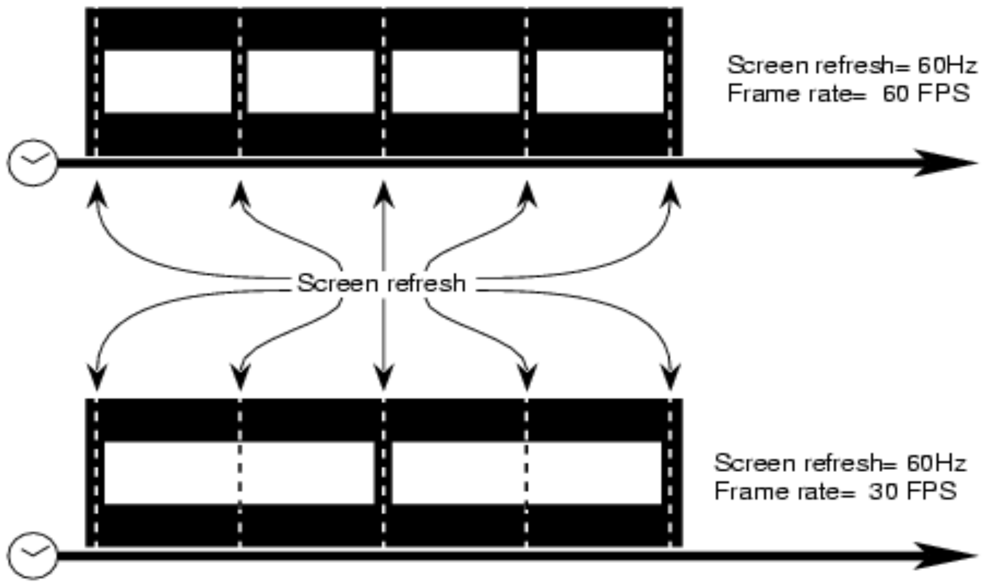


Image courtesy of Silicon Graphics International Corp. Permission:4

Above in figure 5 we see the same film taken at different speeds. The one on the top is taken at 60 frames per second and it is evident that a new image is there every time the screen refreshes. In the film underneath although the screen refreshes as frequently there is no new image every time the screen refreshes. This can cause a blur effect on the video.

3.3.4. Spectral Response

Spectral response or spectral sensitivity refers to the efficiency of light detection in a camera as a function of the wavelength of the signal. This information is hard to find since most companies do not specify this on lower end products like the webcam that was planned to be used in this project. For educational purposes a figure of the spectral response of a high-end camera, Cannon 40D is attached below:

Figure 6: Spectral response for a Canon 40D

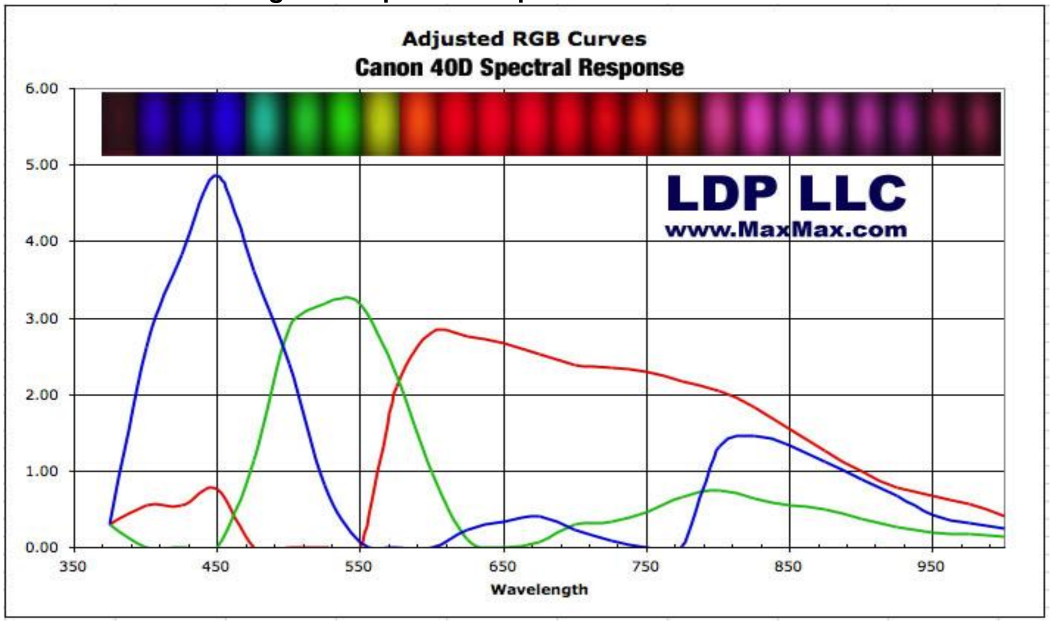


Image courtesy of LDP LLC. Permission:8

At the top of figure 6 there is the visible spectrum colors in order of lowest to highest wavelength and the wavelength values in nanometers are listed in the horizontal lower axis of the graph. It is evident how this camera peaks at about 450nm which is the wavelength for the color blue. This means that this particular camera is most sensitive to blue colors. We can also see peaks at 550nm and 625nm which are green and red respectively. The other peaks or "leakage" is due to imperfections in the color filter. This is important to know because the goal of this project is to have a camera that is most sensitive to near-infrared wavelengths so the camera can pick up easier the gestures performed by the glove.

3.3.5. Visible Light Filter

In this part of the project it is required that the camera blocks out visible light and will detect near-infrared wavelengths (0.75μm-1.4μm). Below there is a figure of the transmission of a visible light filter. Ideally it will transmit about ninety percent of the wavelengths past the 750 nm, which is the highest end of the visible spectrum.

Figure 7: Wavelength Transmission of a Visible Light Filter

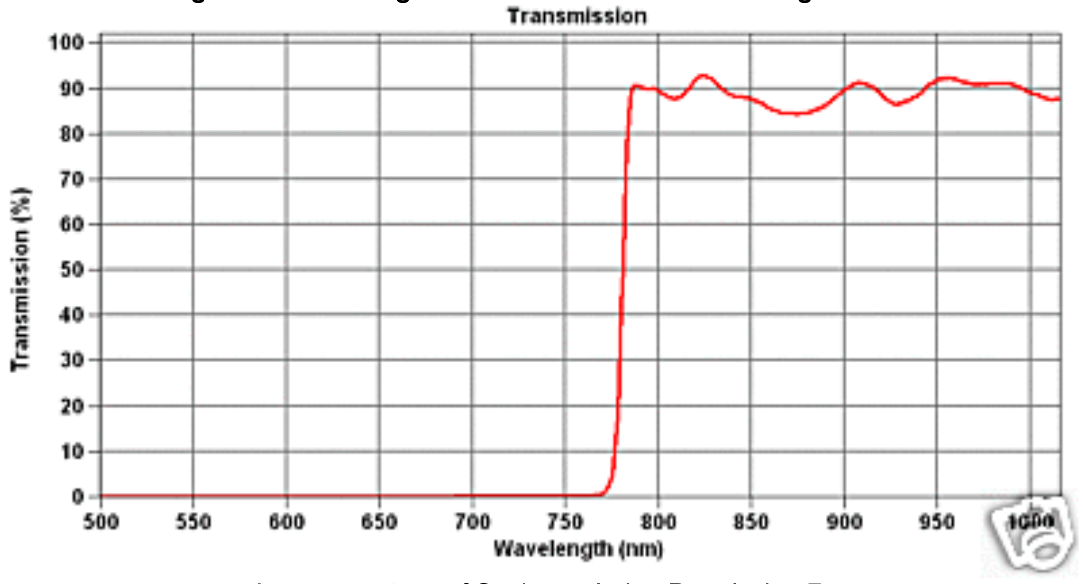


Image courtesy of Sypherus Labs. Permission:7

As a result of doing research the group found out that infrared cameras are more expensive than regular webcams. The group also found out that camera sensors are naturally very sensitive to the near-infrared wavelengths [13] and for this reason they come with a built-in IR filter. The modification that had to be performed to the camera to work in the desired wavelengths was to add a visible light filter to block out the visible light wavelengths and to remove the IR filter to allow the near-infrared wavelengths to come through the sensor. The IR filter could be a separate filter or a coating in a lens depending on the camera used. The visible light filter can be engineered by using the magnetic disk of a floppy disk or film.

Figure 8: Infrared LEDs seen with the IR filter removed and the visible light filter

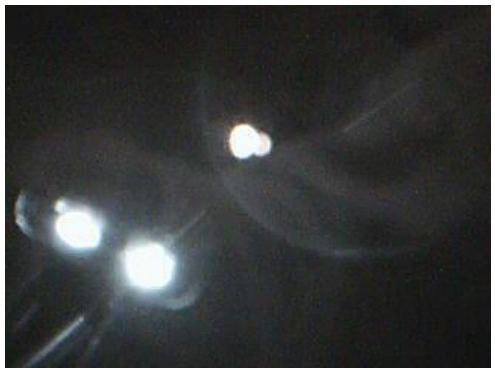


Image courtesy of Sypherus Labs. Permission:7

As mentioned before the goal of this filter is to block out background noise from the object of interest, which are the near-infrared LEDs for better object tracking. The figure above was obtained from an experiment [14] where the IR filter was a coating and was sanded down and a piece of the magnetic disk of a floppy disk was placed as the visible light filter. Although this experiment was used for different purposes the images capture the idea of what was needed to be performed in this project. The image above reflects the modified camera looking at three near-infrared LEDs. It is evident that this type of filter works in this experiment because the LEDs are much brighter than anything in the background which will make it easier to track the figures of interest.

3.4. FPGA vs. DSP Processor

3.4.1. Overview

Digital signal processing is usually implemented on two types of hardware devices: Digital Signal Processors (DSP Processors) and Field Programmable Gate Arrays (FPGAs). DSP processors are a specialized form of reprogrammable microprocessor, while FPGAs are semiconductor devices that are based around a matrix of configurable logic blocks connected via programmable interconnects that could be programmed depending on the application requirements [25]. For the purposes of this project the group studied and compared different parameters such as performance, power consumption, form factor and size, and cost which influenced the decision to use one device over the other one.

A DSP Processors can be continuously reprogrammed according to the requirements of different tasks, some related with the actual signal processing part and others related with the control or communication protocol. In a DSP processor there are “threats” which is when tasks share all the different resources like core registers, internal and external memory, direct memory access and I/O peripherals. These shared resources often interact in unpredicted or bizarre ways that may cause unexpected delays resulting in system failures in real-time applications where DSP processors are usually used.

FPGAs have flexible and reconfigurable hardware architecture. They allow a large amount of parallel processing and pipelining of dataflow. Latest FPGAs support high clock speeds, provide plenty of processing power, and large on-chip RAM. Pipelining in FPGAs is efficient and effective; this is possible due to the high I/O capability that allows FPGAs to access multiple RAM banks simultaneously. FPGAs use multiple memory banks for partitioning and pipelining of algorithms providing significant advantages in performance over the DSP processors such reducing performance bottleneck. The information on this overview was gathered from [19] and [25].

3.4.2. Performance

In this project cost and performance are two very important aspects that the group considered when choosing the image-processing device. The group wanted a device that was able to process the image within the budget goals. High speed and high performance are very important for the development and implementation of sophisticated DSP based electronic systems like the one that was used in this project. The first thing to take into account in the development of a DSP application is the sampling rate of the system that was to be designed. DSP processor is more suitable for low sampling rate DSP applications, otherwise if sampling rate of the system under consideration is higher than a few MHz then FPGA is the better as a hardware platform for implementation of the DSP application. An advantage of DSP processors over FPGAs is that they can handle conditional operations better. If there is no conditional operation in the DSP application to be developed then FPGA is a better choice. For the purpose of our project we worked with conditional operations but we are more interested in working at high clock speeds and using the parallel processing functions that the FPGA offers. Also if the system uses floating points then the DSP processor will be a better option. Both FPGAs and DSP processor provide libraries for common DSP functions like FIR filter design and FFTs but more complex functions are not usually available, if any of these libraries was to be used then this would be a parameter to consider. In our project we didn’t use the built in libraries. The information from this section was gathered from [22].

3.4.3. Cost

The cost of FPGAs and DSP processors are directly related to performance. In table 3 below we can see the difference in cost of DSP processors and FPGAs based on the MMAC performance. MMAC stands for “Millions of Multiply-Accumulate Operations per second. For the purposes of this project we were expected to stay within the medium MMAC performance range.

Table 3: Cost-Performance Relationship

MMAC Performance Range	Device Cost Range	Minimum cents per MMAC for DSP Processors	Minimum cents per MMAC for FPGAs
Low (>300MMAC)	< $10	1.8	--------
Medium (300-1000 MMAC)	$10 - $30	1.6	1.4
	$30 - $100	3	2.8
High (>1000 MMAC)	$100 - $300	5.8	2.9
	> $300	----------	4.2-20

3.4.4. Development Software

It is said that programming FPGAs is difficult, as this requires familiarity and expertise in a hardware oriented languages such as Verilog or VHDL [18]. In this project we wanted to explore other options to program FPGAs like Simulink HDL code generator or even the Labview FPGA module. The DSP processor can take a standard C program and run it. This C code supports a high level of branching and decision-making. It seems like it all depends on the skill and expertise of the programmer. Our group members do not have experience working or programming FPGA but we felt like this is an important part of engineering that we wanted to explore further.

3.4.5. Power Consumption

When developing a DSP applications power consumption and dissipation are also important parameters to be considered, especially for portable DSP based electronics applications. In our project we would have the FPGA circuit connected to the computer but it will have its own power supply, which was most likely to be a rechargeable battery. Systems using DSP processors must drive heavily loaded buses that are used to connect the DSP chips to the memory chips. The extra clock cycles needed to fetch the instructions and the operands from off-chip memory on these buses add more to the total power requirements needed to execute an algorithm. The power consumption of a chip is directly proportional to its clock frequency and due to the ability of an FPGA to split an incoming data stream and process it as several parallel data streams at lower clock rates also becomes an important part of the total power usage. In summary FPGAs are capable of providing much higher throughput than DSP processors due to their highly flexible architecture. This information was gathered from [21].

3.4.6. Form Factor and Size

DSP processors and FPGAs are about the same physical size, the main difference is what is inside the chip. Another advantage of FPGAs over DSP processors is that they are In-System Programmable (ISP), which means they can also be reconfigured on the board during system operation. This means the FPGA can be reconfigured on-the-fly to switch or toggle from one function to another. This capability adds functionality and processing power to a minimum-chip DSP system controlled with an internal or an external controller. A DSP is optimized for use of external memory, so a large data set can be used in the processing. FPGAs have a limited amount of internal storage so need to operate on smaller data sets. Nevertheless FPGA modules with external memory can be used to eliminate this restriction. Since the code for our project was not written when this research was performed we weren't certain if would take advantage of this ISP capability but it was an option to consider when choosing one device over the other. The information from this section was gathered from [21].

3.4.7. Conclusion

In conclusion lower sampling rates and increased complexity suit the DSP approach; higher sampling rates, especially combined with strict, repetitive tasks, suit the FPGA. The decision of choosing one device over the other depends a lot on the designer and the tasks that the devices would be performed on the DSP system. For the purposes of our project we consider using an FPGA. While comparing these two devices we learned that FPGAs have a lot of features that could be useful to the video processing part of the G.R.I.D. design.

3.5. FPGA Development Software

3.5.1. HDL Overview

In this project is important to have an understanding of what HDL is since an FPGA will be used to process the image. HDL is an acronym for Hardware description language and is an efficient computer design tool for the modern design and synthesis of digital systems. As technology advances, designs grow larger and larger every day. This language provides circuit designers the needed logic descriptions to be analyzed at high level, in other words HDL helps the designers with the verification before fabrication of actual hardware.

This language is based on text expressions of the spatial/ temporal structure and the behavior of electronic systems. The notion of time is a unique attribute that makes this language so useful over other soft programming languages. HDL provides different types or styles of descriptions for the same circuit functionality and these include: structural, behavioral, switch-level, data-flow, mixed type/style or mixed language. The two main hardware description languages are Verilog and VHDL. In the sections 3.5.2 and 3.5.3 below we will cover some of major differences between the two languages. The information in this and the following sections of VHDL and Verilog was gathered from [25] and [26].

3.5.2. VHDL

VHDL stands for Very high speed Integrated circuit hardware description language. VHDL is considered to be better at the system level where multiple entity/architecture pairs lead to flexibility and ease in writing code for complex systems. Below is a description of some important aspects that should be considered when choosing a hardware description language:

- **Data Types**- VHDL is a type-oriented language. These types are built in or the user can create and define them. The flexibility of users being able to create their own types is a powerful tool to write code effectively. The user created types also support flexible coding. It also supports multidimensional arrays. Another type it supports is the physical type, which is more synthesizable or targeted design code.
- **Ease of Learning**- Its is said that for beginners, VHDL may seem harder than Verilog due to the rigid type requirements. However advanced users may think these rigid type requirements are easier to handle.
- **Libraries and Packages**- Packages are used to target certain designs. They include procedures and functions and can be made available to any module that needs to use it. These are attached to the standard VHDL library. For example if a system to be modeled/designed includes arithmetic functions, a package can be used that includes those functions.
- **Operators**- An extensive set of operators is available in VHDL but it does not have predefined unary operators.
- **Procedures and Tasks**- In VHDL concurrent procedure calls are allowed. VHDL differs to Verilog in this sense because it allows a function to be written

inside the procedure's body. This feature makes it easier to describe a complex system.

3.5.3. Verilog

Verilog adopted its name from the logic simulator that originally used it. Verilog is considered to be better when describing a system at the gate or transistor level due to its use of predefined primitives at this level.

- **Data Types-** Compared to VHDL, Verilog data types are very simple and easy to use. All types are defined by the language, therefore there are no user defined types. It is said that beginners consider these simple data types as an advantage over VHDL. Moreover Verilog cannot handle objects with multidimensional array types.
- **Ease of Learning-** Verilog is considered to be easier to learn than VHDL. Users can just write the module without worrying about what library or packages should be attached. Also Verilog is thought to be easier for most people because many statements are similar to those in C language and most people are familiar with C language over hardware description languages.
- **Libraries and Packages:-** There is no concept of libraries or packages in Verilog.
- **Operators-** An extensive set of operators is also available in Verilog and unlike VHDL, Verilog does have pre-defined unary operators.
- **Procedures and Tasks-** In Verilog concurrent task calls are allowed. Functions, however, are not allowed to be written in the task's body.

3.5.4. HDL code generators

Using HDL could be tedious and required some time. To avoid this time there are graphical development software that facilitate the design and coding in HDL. Below in sections 3.5.4.1 and 3.5.4.2 the two leaders in the HDL coding industry will be explored further.

3.5.4.1. Simulink: HDL Coder and HDL Verifier

Discovering this tool was a very good advantage for this project. Simulink is very powerful graphical development software developed by MathWorks. It is used for modeling, simulating and analyzing multi-domain dynamic electronic systems. Its graphical block diagramming tool and a customizable set of block libraries make it easier to use for people that are not experts in programming but do understand electronic circuits. One of the reasons why our group benefited from using this software is because it offers tight integration with the rest of the MATLAB environment and can either drive MATLAB or be scripted from it. In the specific case of this project we would use it with a specific set of add-ons: the HDL Coder and HDL Verifier.

HDL coder is a tool that generates portable, synthesizable Verilog and HDL code from MATLAB functions, Simulink models and state-flow charts. This generated

HDL code can be used for FPGA programming. The generated HDL code can be used for FPGA programming. This code generator is compatible with Xilinx and Altera FPGAs. It gives you the flexibility to control HDL architecture and implementation, highlight critical paths and generate estimates hardware resource utilization. It also provides traceability that will connect the various parts of the Verilog of VHDL code with the Simulink model. Creating HDL code with this software can be done in a few steps. First the designer has to model the target design using a combination of MATLAB code, Simulink blocks and Stateflow charts. Then optimize these models to meet speed design objectives, generate the HDL code using the HDL Workflow Advisor for MATLAB and Simulink and finally verify the generated code with HDL Verifier. The information from this section was gathered from [23] and [24]. Below we will break this process into sections and explain further how Simulink performs each task:

Code Generation:
When generating HDL or Verilog code from MATLAB the HDL Workflow advisor will convert the MATLAB code from floating-point to fixed-point and generate the HDL code desired. This tool lets the designer model an algorithm at high level using the MATLAB structure while providing options for generating HDL code that is optimized for hardware implementation. Another advantage of this is that most already written libraries in MATLAB can be used. If the code is being generated from Simulink the HDL Workflow Advisor will generate the HDL code from Simulink and Stateflow. In Simulink there are libraries with more than 200 blocks that can be used to model an algorithm.

Code Optimization:
This can be achieved by using pipelining, streaming, and resource sharing. For example in Simulink one can implement multichannel designs and serialization techniques which are common to multimedia and DSP applications which pertain to this project.

FPGA Design:
HDL Workflow Advisor in HDL coder automates the workflow for implementing MATLAB algorithms and models into the two compatible FPGAs (Xilinx and Altera). Using HDL Workflow Advisor integrates all the steps of the FPGA design process:

- Verifying Simulink Model for HDL code generation compatibility
- Generating HDL code, an HDL test bench and co-simulation model
- Performing synthesis and timing analysis through integration with Xilinx ISE and Altera Quartus II
- Estimating resources used in the design
- Back annotating the Simulink model with critical path timing

Verifying the HDL Code:
HDL Coder works together with HDL Verifier generate two types of co-simulation models:

- HDL co-simulation model, for performing HDL co-simulation with Simulink and an HDL simulator
- FPGA-in-the-loop (FIL) co-simulation model, for verifying your design with Simulink and an FPGA board

Tracing the HDL Code:

HDL Coder documents the generated code in an HTML format that contains hyperlinked HDL code and a table of generated HDL files. These hyperlinks connect the generated code to the corresponding part of the MATLAB algorithm or Simulink block that created that part of the code. It also allows you to insert user comments and descriptions to improve code readability

Key Features and Conclusion:

After taking a deeper look at this software we thing it would be great if it is used in this project. If is very user friendly and does not require prior experience in HDL programming. Below there is a summary of all the key features that are offered with this type of software:

- Target-independent, synthesizable VHDL and Verilog code
- Code generation support for MATLAB functions, System objects, and Simulink blocks
- Mealy and Moore finite-state machines and control logic implementations using Stateflow
- Workflow advisor for programming Xilinx and Altera application boards
- Resource sharing and retiming for area-speed tradeoffs
- Code-to-model and model-to-code traceability
- Legacy code integration

3.5.4.2. Labview FPGA Module: DSP Design

This program is very much like Simulink in the aspect that it is used with the to facilitate FPGA programming to inexperienced HDL programmers. Some of the advantages of using this module over traditional HDL language is that it allows the user to prototype real-time FPGA-based digital signal processing subsystems, effortlessly integrate complex FPGA based math and signal processing libraries, design signal processing blocks , import third-party IP blocks and finally investigate design trade-offs early in the design process.

LabVIEW DSP Design for the LabVIEW FPGA Module reduces the complexity of designing real-time DSP subsystems for high-speed field-programmable gate array (FPGA) applications such as RF and communications. By using a stream-based graphical abstraction, the designer can rapidly implement an algorithm, explore design trade-offs, and generate an optimized FPGA implementation. The designer can then integrate the resulting implementation as a modular part of a larger LabVIEW FPGA-based application. Below it is explained in more detail how this module performs some of its important functions. The information covered in this section was gathered from [27].

Quick Prototype Real-Time FPGA-Based DSP Subsystems:
LabVIEW DSP Design offers an intuitive language that extends LabVIEW FPGA by providing a target-independent description of the DSP algorithm. Multi-rate DSP algorithms are described by combining high-level functional blocks that explicitly describe sample counts at the inputs and outputs of each block. LabVIEW DSP Design combines the sample counts of the inputs and outputs of each block with known cycle-level timing to determine optimal scheduling, memory, and FPGA resource utilization. You can compile the resulting algorithms into reusable blocks and integrate them into larger LabVIEW FPGA-based applications for execution on compatible NI reconfigurable I/O FPGA devices.

Design Trade-Offs:
With LabVIEW DSP Design, the user can effectively understand and optimize these design aspects early in the design process. In addition to cycle-accurate simulation for algorithmic verification, this module helps the designer to achieve a detailed look at execution scheduling. Using the Scheduling view, you can analyze trade-offs between latency and resource utilization. By enabling pipelining and changing memory buffer sizes, you can profoundly impact how high-speed, multi-rate applications execute.

3.6. FPGA Chip

3.6.1. Overview

Modern programmable hardware allows users to send data to a programmable chip to tell the chip how to wire itself. Modern hardware compilers can take this high level description of an electronic circuit and translate it into configuring bit strings, which are used to configure a programmable chip (FPGA). Inside a chip there are a lot of programmable logic gates, which allow placing a whole electronic system on a single chip. This is the reason why electronics today are more sophisticated, powerful and cheaper than before. A disadvantage of this is that electronics become harder to design. Some sources say that "it's almost as easy to program hardware as to program software" but in order to program hardware well the designer needs to understand the principles of digital electronic design for example multiplexors, flip flops, buffers, and counters among others which could be challenging.

Figure 9: FPGA Preferred Vendors 2012

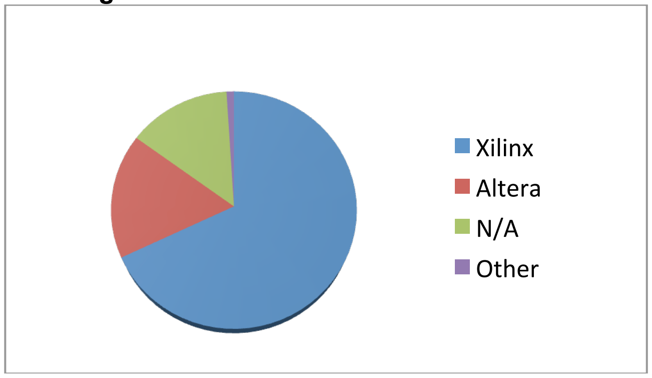


Image courtesy of Synopsys. Permission:6

The information in figure 9 above is obtained from *FPGA-based Pricing Methodology Manual (FPMM) by Synopsys, Inc.* By looking at the numbers on the graph is it evident that Xilinx is the leader in the FPGA market although it is said that Altera has more tools that will help the programmer especially if they are not seasoned in hardware description languages. In the next sections 3.6.2 and 3.6.3 we will discuss the major differences and form these we will come to a decision on which FPGA chip to use for our project.

3.6.2. Architecture of the Chip

To understand a little more about the design of the FPGA the designer needs to understand the basic architecture of the chip. A set of figures and descriptions will be used for an easy understanding of such complex device. In section 3.4.1 is explained that FPGAs are semiconductor devices that are based around a matrix of configurable logic blocks connected via programmable interconnects that could be programmed depending on the application requirements. In figure 10 below a simple FPGA structure is shown. Two input/output pads fit into the height of one row or the width of one column. All the routing channels have the same width or number of wires.

Figure 10: FPGA Structure

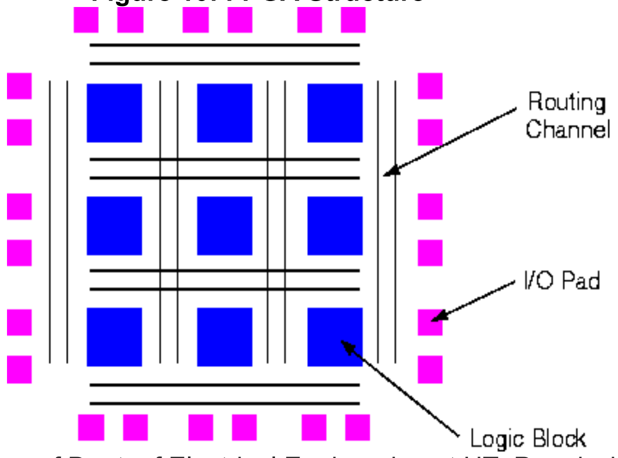


Image courtesy of Dept. of Electrical Engineering at UT. Permission:5

Each circuit has to be mapped into the smallest square that can accommodate it. For example a circuit containing 8 logic blocks and 6 input/outputs pads could be mapped into an FPGA consisting of a 3x3 array of logic blocks like the one shown above. If more blocks or input/output pads are needed then a bigger configuration will be needed.
A logic block inside an FPGA usually consists of a few logical cells, these cells are often called ALM, LE, or Slice. A typical cell consists of a 4-input look-up table (LUT), a Full adder and a flip-flop, as shown below in figure 11. As seen in the figure the LUTs are split into two 3-input LUTs. In normal mode these are combined into a 4-input LUT through the first (left) mux. In arithmetic mode, the outputs are fed to the full adder. The selection of mode is then programmed or sent to the middle multiplexer. The output can be selected to be synchronous or asynchronous depending on the programming of the right most mux. In practice, entire or parts of the FA are put as functions into the LUTs in order to save space.

Figure 11: Logic Block Structure

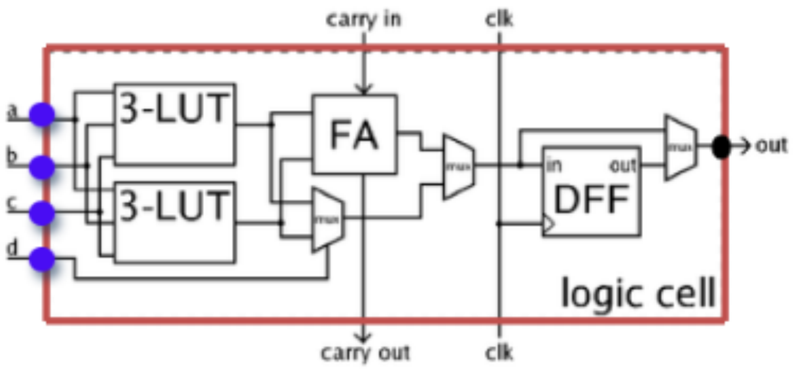


Image courtesy of Dept. of Electrical Engineering at UT. Permission:5

In the figure above we can see the blue circles correspond to the inputs and the black circle corresponds to the output. Figures 12 and 13 are color coded so the reader can match up the location of the FPGA logic block pins and correlate both figures. In figure 12 below it is clear that each input is accessible from one side of the logic block, while the output pin can connect to routing wires in both the channel to the right and the channel below the logic block.

Figure 12: Logic Block Pin Locations

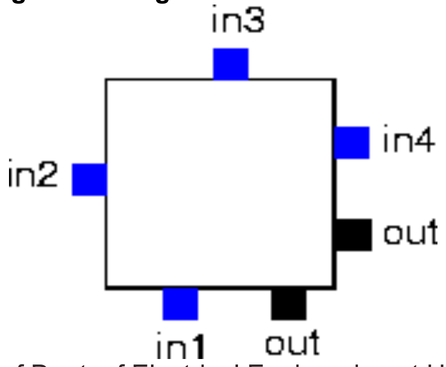


Image courtesy of Dept. of Electrical Engineering at UT. Permission:5

Each logic block input and output pins can connect to any one of the wiring segments in the channel bordering to it. In the same way, an input/output pad can connect to any one of the wiring segments in the channel adjoining to it. As seen in the picture below a channel can have multiple wires, also called channel width. For example, an input/output pad at the bottom of the chip can connect to any of the wires in the horizontal channel immediately below it as seen in figure 13.

Figure 13: Logic Block Pin to Routing Channel Interconnect

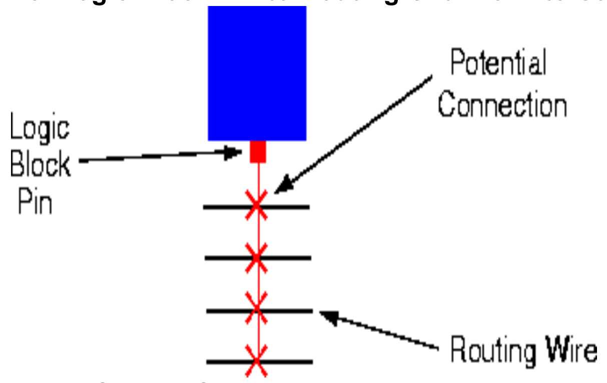


Image courtesy of Dept. of Electrical Engineering at UT. Permission:5

The red crosses in the figure above point at the possible connections that the input/output pad can have. There are various choices but it is important to know that only one wire segment may be connected to a logic block input pin.
Moreover FPGAs are typically unsegmented, this means that each wiring segment passes through only one logic block before it reaches a switch box. By turning on some of the programmable switches within a switch box, longer or shorter paths can be constructed. Some FPGAs architectures use longer routing lines that pass through multiple logic blocks to provide high speed interconnects. In the figure 14 below it is clear where these switch boxes are located.

Figure 14: Unsegmented FPGA Routing

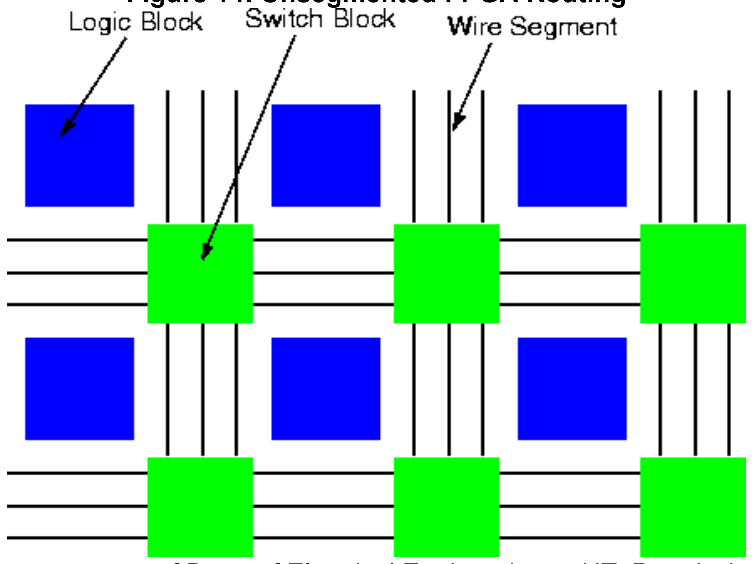


Image courtesy of Dept. of Electrical Engineering at UT. Permission:5

In every section where the vertical and horizontal channels intersect there is a switch box. Since in this particular figure we can see a width of 3 because a channel has 3 wires the switch box will have three programmable switches that allow it to connect to three other wires in adjoining channel segments. This configuration is often called planar or domain-based switch box topology. One more thing that need to be understood from this topology is that since each channel has multiple wires these wires correspond to the same wire number in the adjoining channel. For example wire 1 in the vertical channel will only connect to wire 1 in the horizontal channel, same applies for wires two and three. This may seem confusing but in figure 15 there is a zoomed in view of the channel connections and the switch box. The programmable switch would be set to best suit the desired application which for this project was digital signal processing.

Figure 15: Switch Box Topology

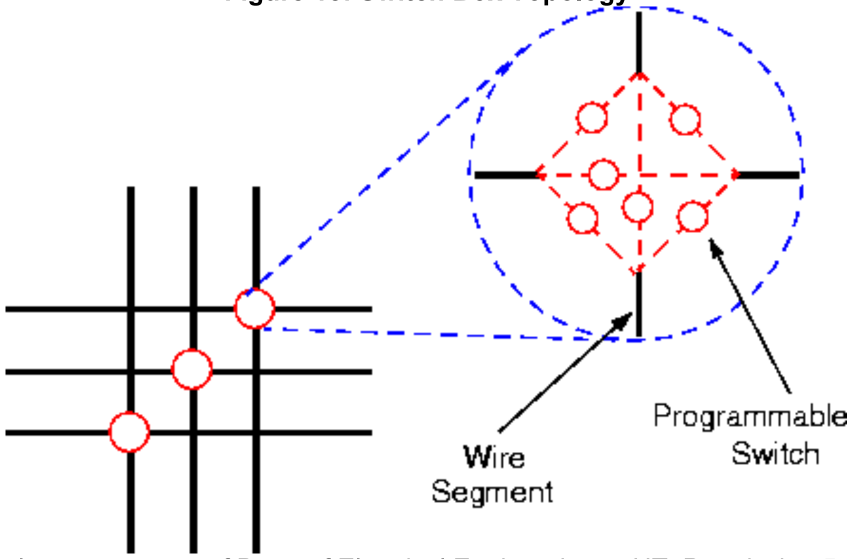


Image courtesy of Dept. of Electrical Engineering at UT. Permission:5

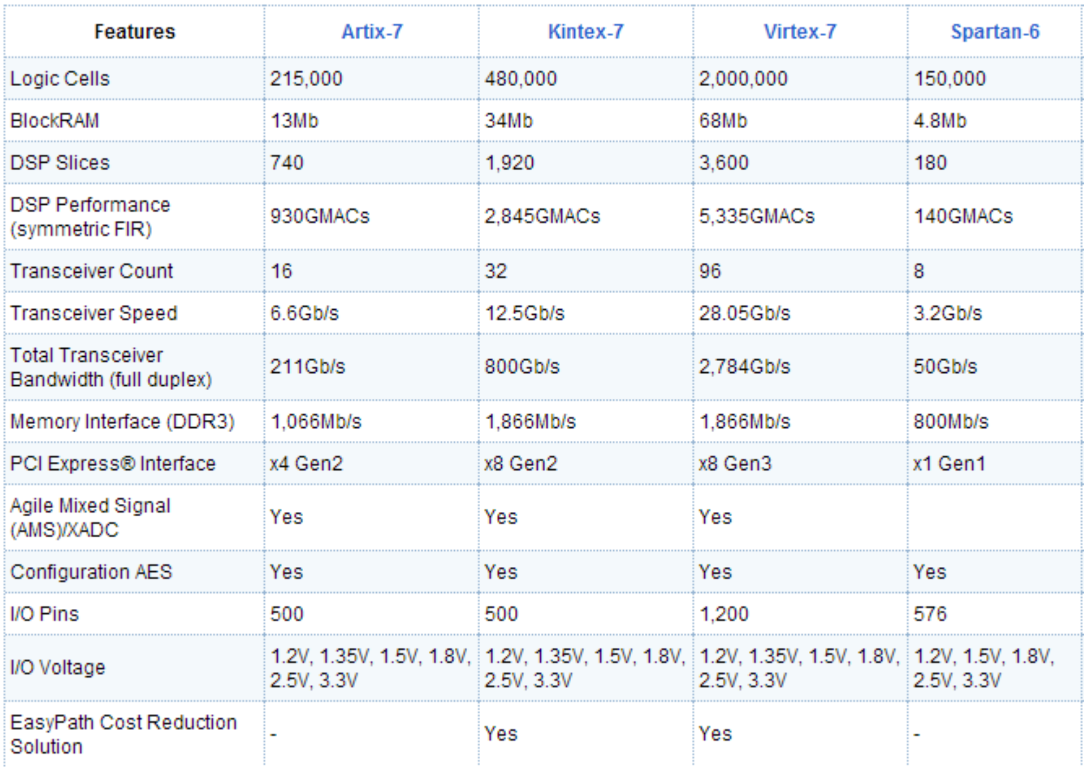
3.6.3. Xilinx Chip

Xilinx are the pioneers in FPGAs. They invented the first commercially viable FPGA in 1985 [28]. Since then they control over half of the FPGA market. Xilinx offers a variety of FPGA families that vary from low-end products to high-end devices. The different familiess of FPGAs provided by Xilinx are:

- **Spartan**- Provide optimal balance of low risk, low cost and low power for cost sensitive applications.
- **Artix & Kintex**- Provides performance and power balance for mid-range applications.
- **Virtex-** Provides high bandwidth and high performance for high end designs.

These families of FPGAs are listed in Table 4 below. The table discusses many features that are critical for the decision making on which FPGA to use.

Table 4: Xilinx FPGAs Features



Features	Artix-7	Kintex-7	Virtex-7	Spartan-6
Logic Cells	215,000	480,000	2,000,000	150,000
BlockRAM	13Mb	34Mb	68Mb	4.8Mb
DSP Slices	740	1,920	3,600	180
DSP Performance (symmetric FIR)	930GMACs	2,845GMACs	5,335GMACs	140GMACs
Transceiver Count	16	32	96	8
Transceiver Speed	6.6Gb/s	12.5Gb/s	28.05Gb/s	3.2Gb/s
Total Transceiver Bandwidth (full duplex)	211Gb/s	800Gb/s	2,784Gb/s	50Gb/s
Memory Interface (DDR3)	1,066Mb/s	1,866Mb/s	1,866Mb/s	800Mb/s
PCI Express® Interface	x4 Gen2	x8 Gen2	x8 Gen3	x1 Gen1
Agile Mixed Signal (AMS)/XADC	Yes	Yes	Yes	
Configuration AES	Yes	Yes	Yes	Yes
I/O Pins	500	500	1,200	576
I/O Voltage	1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.3V	1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.3V	1.2V, 1.35V, 1.5V, 1.8V, 2.5V, 3.3V	1.2V, 1.5V, 1.8V, 2.5V, 3.3V
EasyPath Cost Reduction Solution	-	Yes	Yes	-

Image courtesy of Xilinx All-Programmable. Permission:9

The table above was obtained from the Xilinx website [28]. Some of the features the group will take into account to make a decision are: the amount of logic cells, RAM, DSP slices, Memory interface, and I/O pins. From the Xilinx families mentioned above the group would be leaning towards the Spartan-6 because it is a low cost design that will perform the video processing required in this project. Some of the benefits of the Spartan-6 low cost family of FPGAs are:

- Designed for low cost
- Low static and dynamic power
- Multi-voltage, multi-standards Select IO interface banks
- Low-cost PCI technology support compatible with the 33MHz, 32- and 64-bit specification
- Efficient DSP48A1 slices
- Integrated memory controller blocks
- Abundant logic resources with increased logic capacity
- Block RAM with a wide range of granularity
- Clock management Tile (CMT) for enhanced performance
- Simplified configuration- to support low-cost standards
- Enhancement security for design protection

- Faster embedded processing with enhanced, low cost MicroBlaze soft processor
- Industry-leading IP and reference designs

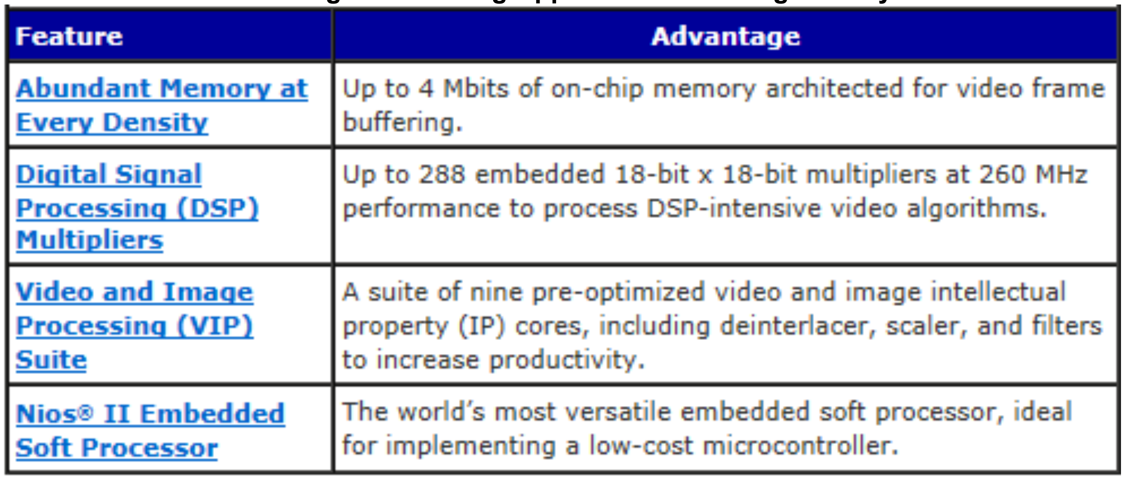
3.6.4. Altera Chip

Altera are the second leaders in the FPGA market. They control about thirty percent of the market. Altera also offers a variety of FPGA families for different purposes. The FPGA families that Altera offers are:

- **Cyclone**- Provides low cost and low power design.
- **Arria**- Provides optimal balance of performance, power and price for mid-range applications.
- **Stratix**- Provides high bandwidth for high end applications.

For this project, the low end Cyclone model will suffice. Below there is a table taken from the Altera website [29] that describes the advantages of using a Cyclone III from video and image processing applications like the ones used in this project.

Table 5: Video and Image Processing Application Advantages of Cyclone III FPGAs



Feature	Advantage
Abundant Memory at Every Density	Up to 4 Mbits of on-chip memory architected for video frame buffering.
Digital Signal Processing (DSP) Multipliers	Up to 288 embedded 18-bit x 18-bit multipliers at 260 MHz performance to process DSP-intensive video algorithms.
Video and Image Processing (VIP) Suite	A suite of nine pre-optimized video and image intellectual property (IP) cores, including deinterlacer, scaler, and filters to increase productivity.
Nios® II Embedded Soft Processor	The world's most versatile embedded soft processor, ideal for implementing a low-cost microcontroller.

Image courtesy of Altera. Permission:10

It is very important that the device provides plenty of memory and that is capable of handling the video streaming through it. Another advantage from the Altera FPGA chips is that they are easier to program than Xilinx. Some of the benefits offered by the Altera Cyclone III are :

- Low power consumption which will extend battery life for portable applications and reduce or eliminate cooling system costs
- Routing architecture optimized for design separation flow with Quartus II software
- Internal oscillator enables system monitor and health check capabilities
- High memory-to-logic and multiplier-to-logic ratio
- High Input/Output count, low and mid-range density devices for user I/O constrained applications
- Robust clock management and I/O interfaces

- Remote system upgrade without the aid of external controller
- Dedicated cyclical redundancy code checker circuitry to detect single-event offset issues
- Nios II embedded processor offers low cost and custom-fit embedded processing solutions

Supports high-speed external memory interfaces such as DDR, DDR2. SDR SDRAM and QDRII SRAM

3.6.5. Conclusion

Choosing to use a device over another is a trade-off. There are great features both Xilinx and Altera FPGAs but the group has to determine which device offers the most benefits to the project. As a result of doing research the group found out that Altera FPGAs are easier to program since Altera provides various user friendly tools for unseasoned HDL programmers. In the other hand Xilinx FPGAs are harder to program but they are offered at a lower cost. As a result of the projects cost restriction the group is leaning towards using the Xilinx Spartan-6. Although this is a low end device it will support the processing required for this project.

3.7. FPGA Power Supply

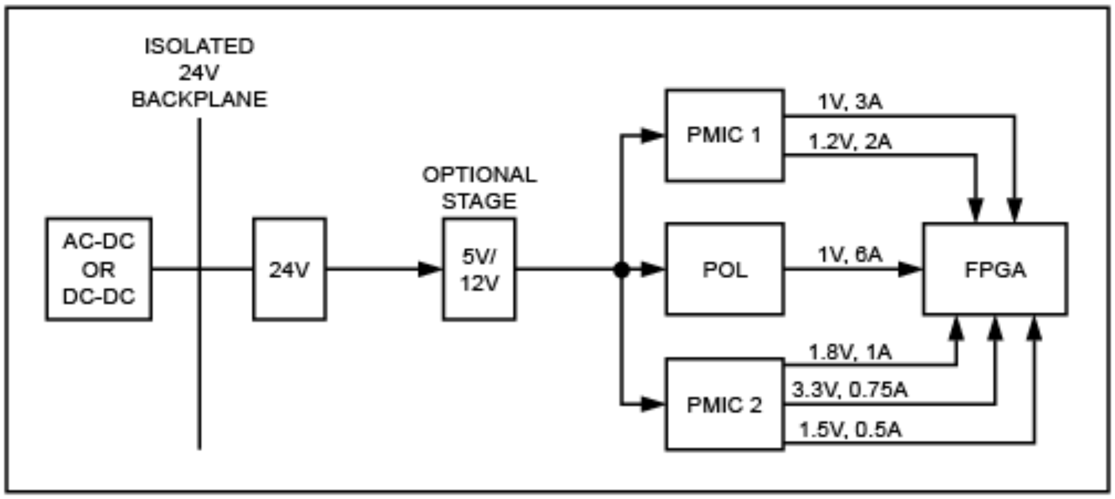
3.7.1. Overview

FPGAs today operate at lower voltages and higher currents than their antecedents. Consequently, power supply requirements may be more demanding, requiring special attention to features considered less important in past generations. Things to consider regarding the power supply of an FPGA output voltage, sequencing, power on, and soft-start requirements. Failure to consider these parameters can result in unreliable power up or potential damage to the FPGA.

3.7.2. Output Voltage Requirements

When designing power supplies for FPGAs the first criteria to consider are the voltage requirements for the different supply rails. Most of the FPGAs have specifications for the CORE and IO voltage rails, and many require additional auxiliary rails that may power internal clocks, phase lock loops (PPL) or transceivers. In addition, as process technology nodes become smaller in FPGAs, tighter tolerances are needed on the voltage-supply rails. Figure 16 shown below shows why regulators with 1% regulation accuracy across line and load and process-voltage-temperature (PVT) variations are so critical:

Figure 16: Voltage Regulators and PVT Variations



"Copyright Maxim Integrated Products (http://www.maximintegrated.com). Used by permission."

FPGAs generally specify several allowable voltage levels for the IO, the external digital circuitry determines the voltage selected. To provide flexibility, the FPGA will generally provide multiple IO banks that can be powered separately allowing the FPGA to interface with various logic families. Table 6 shown below shows the voltage levels and tolerances for several different FPGAs:

Table 6: FPGAs Power Specifications

FPGA	Core	Core Tolerance	Auxiliary Power	Auxiliary Tolerance	IO Voltage	IO Tolerance
Spartan-6	1.2	5%	2.5 or 3.3	5%	1.2 to 3.3	1.1 to 3.45
Virtex-6	1.0	5%	2.5 or 3.3	5%	1.2 to 2.5	1.14 to 2.625
Cyclone	1.2	40 mV	2.5	5%	1.2 to 3.3	5%
Arria2	0.9	30 mV	2.5 (VCCA_PPL) 0.90 (VCCD_PPL)	5% (VCCA_PPL) 30mV (VCCD_PPL)	1.2 to 3.3	5%

The content of this table is from EE Times Design [2].

Most of the voltage supplies the internal logic configuration blocks of the FPGA and is where many of the internal digital path processes occur. As such, the current demanded by the core will vary greatly depending on the percent utilization of the FPGA.

Over time the voltages used to power the core have been steadily dropping. Modern cores like the Arria2 can operate off of voltages as low as 0.9V. Lower core voltages are permitted by improved geometry silicon processes, and are valuable in keeping the power dissipated in the FPGA to a reasonable level. With process technologies designed to operate at lower voltage levels, keeping within the core voltage tolerance requirements has become more challenging for the power supply designer. The information in this section was gathered from [8] and [9].

As part of this project good candidates are (in order of complexity):

- The Altera Cyclone family.
- The Xilinx Spartan-II family or the Spartan-3 family.

3.8. Microcontrollers

3.8.1. Overview

Microcontrollers or MCUs are small scale computers on an integrated circuit which contains a processor core, memory, and a variety of General Purpose Input/output pins (GPIO). These devices, because of their limited structure, are used in embedded system design, commonly to control a system, where the inputs(s) are generally external sensor data, and the output(s) are signals to external devices. Microcontrollers range in their number of GPIO, clock speeds, integrated communications modules, the size of their available memory, as well as their word and address bus sizes. Additionally, most microcontrollers feature a variety of programmable timers, which can be used in scenarios where certain actions must be performed with very high degree of accuracy in time (Pulse Width Modulation), and analog to digital converters for processing analog signals. The market share of these devices is very large, with about 55% of all CPUs sold in the world being 8-bit microcontrollers.

3.8.2. Limitations

The biggest limitation of microcontrollers is speed. Although Clock speeds vary from model to model, most MCUs run in the order of tens of Mega Hertz. In cases where high throughput is required MCUs just cannot process the data quickly enough. As an example, image/video processing requires pixel by pixel processing on a matrix the size of the resolution of each image. For a video stream of 30 frames per second at a resolution of 640x480 each pixel needs to be processed in no more than .1us in order to maintain the frame-rate. Depending on the nature of the processing algorithm the number of instructions required may vary from tens to hundreds. Furthermore, MCUs use a RISC architecture, which would suggest a one-clock cycle per basic instruction, but at the expense of numerous instructions for more complex operations. This kind of work would require clock speeds in the Giga Hertz range, but because of the interrupt based nature of MCUs any interrupt might be enough for the system to lag.

Memory size is another drawback of microcontrollers. The msp430g2553 offers 512B while the Arduino UNO offers 2MB. It is possible however, to interface a microcontroller with external memory devices. In this case, there is a speed vs. size compromise. External communications are done through I^2C or SPI. The speed of these protocols is usually much slower than the MCUs internal memory. Fast I^2C operates at 400 KHz, therefore while fetch instruction might take only a couple of cycles from internal memory, the MCU might idle for many cycles before data is ready to be used. This results in large overhead. For computational heavy algorithms all variables must be stored within the MCU for optimal efficiency, only using the external memory to log data after the process.

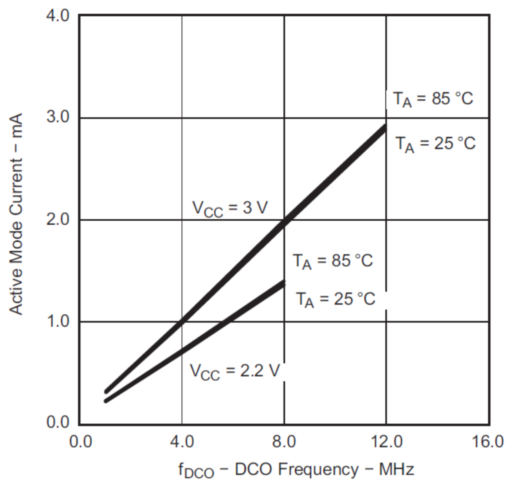
3.8.3. Advantages

MCUs excel in tasks that do not require extensive bit crunching. They are good for controlling various types of displays, doing simple analysis on sensor data, and matching input conditions to outputs. One of the features that make microcontrollers attractive to the market is their low cost. Most embedded applications are simple enough that they do not require a microprocessor for the job, so MCU are very appealing as they are suited for simple task, and keep production cost down. For more intensive applications, it is not uncommon to see systems with multiple microcontrollers communicating with each other to accomplish the task. Again, because of their low cost, even adding multiple MCU and increasing the complexity of the design is favored over a stronger, but more costly processor core.

3.8.4. The MSP430G2553

Due to the nature of our project, low power is critical in order to meet the design specifications. The MSP430 one of the most flexible MCU on the market with multiple clock sources, programmable modulation for each source, and low-power modes. As seen in figure 17 there is a direct relation between power consumption and the frequency of operation. We will take advantage of the MSP430's digitally controlled oscillator (DCO) in order to scale the speed in accordance to the current task in order to maximize power efficiency.

Figure 17: MSP430g2553 current vs DCO frequency



Furthermore, the MSP430g2553 features several low-power modes which a user can enter and exit in one clock cycle time. This feature allows us to enter a low power mode during periods of inactivity, and because of the fast wake up time there is no impact on performance. The various low power modes are detailed in table 7.

Table 7: MSP430g2553 low power mode functions

Parameter	**Test Conditions**	**T_A**	**V_{cc}**	**MIN TYP MAX**	**UNIT**
Low-power mode 0 (LMP0) current.	MCLK = 0 MHz, SMCLK = DCO = 1 MHz, ACLK = 32,768 Hz. BCSCTL1 = CALBC1_1MHZ, DCOCTL = CALDCO_1MHZ, CPUOFF = 1, SCGO = 0, OSCOFF = 0	25°C	2.2 V	56	μA
Low-power mode 2 (LPM2) current.	MCLK = 0 MHz, SMCLK = 0, DCO = 1 MHz, ACLK = 32,768 Hz. BCSCTL1 = CALBC1_1MHZ, DCOCTL = CALDCO_1MHZ, CPUOFF = 1, SCGO = 0, SCG1 = 1 OSCOFF = 0	25°C	2.2 V	22	μA
Low-power mode 3 (LPM3) current.(LFXT1)	MCLK = 0 MHz, SMCLK = 0 MHz, DCO = 0 MHz, ACLK = 32768 Hz. BCSCTL1 = CALBC1_1MHZ, DCOCTL = CALDCO_1MHZ, CPUOFF = 1, SCGO = 0, OSCOFF = 0	25°C	2.2	0.7 1.5	μA
Low-power mode 3 (LPM3) current.(VLO)	MCLK = 0 MHz, SMCLK = 0 MHz, DCO = 0 MHz, ACLK from internal LF oscillator (VLO). CPUOFF = 1, SCG0 = 1, SCG1 = 1, OSCOFF = 0.	25°C	2.2 V	0.5 0.7	μA
Low-power mode 4 (LPM4) current.	MCLK = 0 MHz, SMCLK = 0 MHz, DCO = 0 MHz, ACLK = 0 Hz. CPUOFF = 1, SCG0 = 1, SCG1 = 1, OSCOFF = 1	25°C	2.2 V	0.1 0.5	μA
“ “	“ “	85°C	2.2 V	0.8 1.7	

Another reason for this device choice is it’s incredibly low cost. The chip itself cost less than a dollar for a single unit, with a price drop for bulk purchases.

Furthermore, development cost is also incredibly small, thanks to the TI's Launchpad. The Texas Instrument Launchpad is a starter development kit that includes: the development board, msp43g2553, a 32 kHz external crystal, and extra headers for prototyping, all for 4.30$, and with free shipping. TI also offers a code limited version of their IDE Code Composer Studio free of charge.

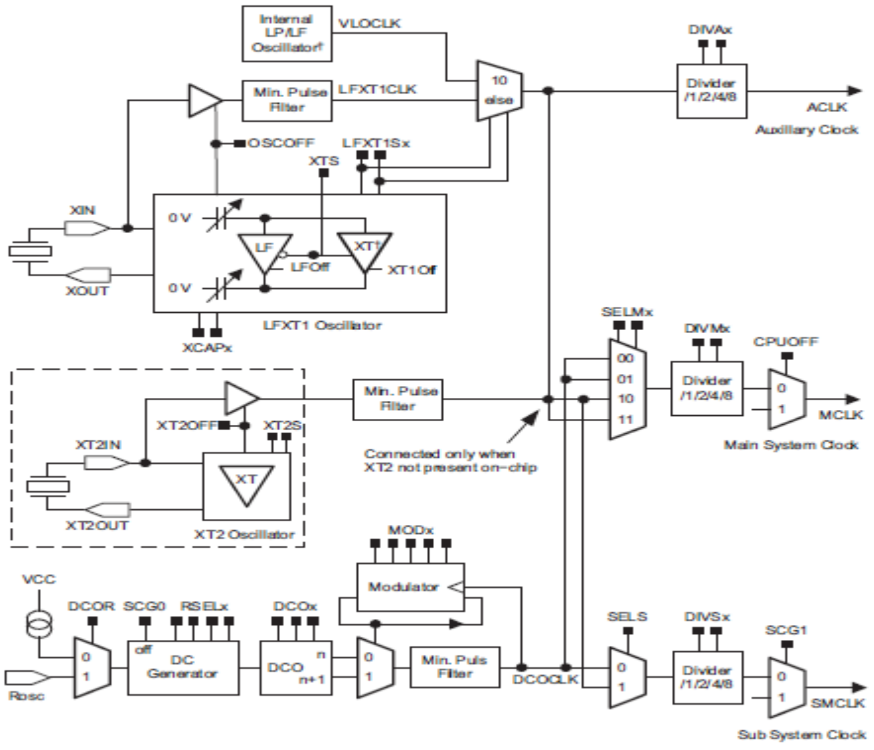
3.8.4.1. Development Software

Code composer Studio allows the user to write all code in C or in assembly. Because of our experience with C we choose it as the preferred language for the microcontroller. In order to properly communicate with the msp430g2553 we looked at the header file msp430g2553.h in order to see register definitions and interrupt vector definitions, as well as example code found on TI's website.
Code Composer studio also facilitates debugging through its own debugging environment. This feature gives the user full control of the chip at any given time, even allowing access to current variable values. All of the msp430g2553 control registers are also available for viewing at any one instance in the code. This includes: interrupt enable register, interrupt flags status, GIOP pin configuration for port1 and port2, also status and configuration registers for each of the modules (I.E. Timers, ten bit analog to digital converter, USCI). For added control the user can choose to "dive into the instruction" in which case it will (for the case of a function call) dive into the function and perform the code within. The other case is "jump over", in which case is will perform all lines of code within the function, but only the register statuses at the end of the call will be available for viewing.

3.8.4.2. Device features

The msp430g2553 is a surprisingly flexible microcontroller. In addition to its obvious advantage over the competition in regard to power consumption, it has several other features which make it the preferred choice. Firstly, the clock system, the msp430g2553 features three clock signals: The Master clock, the Sub-main clock, and the Auxiliary clock. All clocks are independent of each other, and can be the clock source for the various modules on board. The clock module block diagram is shown below in figure 18.

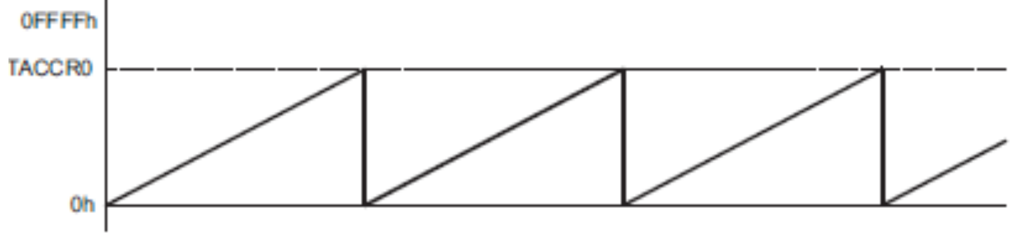
Figure 18: MSP430g2553 clock module block diagram.



Why is this convenient? Not all of the functions of the microcontroller are sourced by the same wave generator. This allows the user to turn off entire sections of the clock module while they are idling, and resume operation if required. Furthermore each source passes through a divider. This block scales the incoming signal in frequency allowing user to ramp-up/ramp-down processing speed as per demands.

Another important feature is the presence of two 16-bit timers A, and B. Because they can be sourced from the SMCLK, or the ACLK, they do not run at the speed of the CPU, and thus can be configured independently. This is in important feature for applications where events are recorded in pre-defined timed intervals. For our project, timer A and B provide an accurate way to establish a time-frame in which to read the MPU-6050's register, and then compare that signal to a template for gesture recognition.

Figure 19: Timer raises an interrupt flag once the value in TACCR0 is reached.



Lastly, it features 16KB of internal flash memory. Considering the sampling rate at 50 Hz for both the accelerometer and gyroscope, as well as a time-window for allowable measurements of two seconds, the most memory needed would be 5KB approximately, slightly over half of the allowable range. That is for storing full 16-bit readings on each axis for both the accelerometer and gyroscope.

3.8.5. Arduino Uno

The Arduino Uno is a development package based on the 8-bit ATmega328 microchip, and runs off a wiring-base language similar to C++, but with some slight modifications. Because Arduino is meant as an electronic solution for those who might not be familiar with embedded design, it functions in a way that is more accessible for beginners. Furthermore, it has been used extensively by hobbyists since its release, and thus, there is a wealth of information, as well as libraries of existing code for support. At the same time, the Arduino trades some of the more powerful features, like low power modes, and register access, in order to maintain its ease-of-use. The Arduino Uno is comparable Texas Instrument's msp430 Launchpad as presented in table 8.

Table 8: Arduino UNO (ATmega323) and TI Launchpad (MSP430g2553) hardware comparison

	TI Launchpad	**Arduino Uno**
Microcontroller	Msp430G2553	ATMega328
Data Bus	16-bit	8-bit
Speed	16 MHz	16 Mhz
Storage	16 KB	32 KB
RAM	512 B	2 Kb

3.9. Power Supply

3.9.1. Overview

The objective of this section is to examine the operation conditions of the microcontroller to provide a stable, low voltage supply to the MSP430. The operation of the MSP430 must not be affected by the power supply. The power supply itself must be dependable and stable. The power supply should not cause problems during development. Below are the recommended operating conditions for the MSP430 Power Requirements:

- Typical Input Voltage Range

Table 9: Recommended Operating Conditions

	Min	**MAX**
Supply Voltage during program execution, V_{cc}	1.8 V	3.6V

- Typical Operating Current
  - Active Mode 300μA to 5μA
  - Sleep Mode 100nA

Now that the MSP430 power requirements have been considered, then the rechargeable batteries will need to meet the specifications to power the microcontroller. The power requirements information was gathered from [2]. Part of the project requirements proposed for this project the ease for the user to interface between the computer and mouse therefore, the rechargeable batteries can meet this project requirements because is a choice for portable applications and also its high capacity-to-size ratio which diminishes the weight. Most popular rechargeable batteries are Nickel-Cadmium (NiCd), Nickel-Metal-Hydrate (NiMH), and Lithium-Ion (Li-Ion) in which charging circuits depend on the battery's chemistry. This section will discuss the battery's chemistry to meet the power requirements for the MSP430:

- **Nickel-Candium rechargeable batteries** - has a well-established chemistry. A nickel-candium battery has a terminal voltage of around 1.2 volts during discharge, which decrease little until nearby the end of discharge. Nickel-Candium batteries are made in a variety of sizes and capacities, from portable sealed types interchangeable with carbon-zinc dry cells, to large ventilated cells used for standby power and motive power. Compared with other types of rechargeable cells they offer good cycle life and capacity, good performance at low temperatures, and work well at high discharge rates (using the cell capacity in one hour or less). However, the materials are more costly than types such as the lead acid battery, and the cells have higher self-discharge rates than some other types. Sealed Ni-Cd batteries require no maintenance.

- **Nickel Metal Hydride Rechargeable Battery Packs** - steady increases in performance have made this chemistry an excellent choice for small, lightweight, portable, and handheld rechargeable battery applications. In the last few years lithium technologies have steadily worn away at the market share for NiMh. Among the advantages of NiMH are 30% to 50% more capacity than a NiCd cell the same size, less prone to memory effect than NiCd, and competitively priced (especially compared to lithium-ion/lithium polymer). NiMH, however, does have some down sides. NiMHs only yield about 500 cycles, charging is more complex than NiCd, they may require passive safety devices for protection against short circuits and overheating, and NiMH loses its charge about twice as fast as NiCd. Contact House of Batteries for more information about custom rechargeable battery pack manufacturing.

- **Lithium-Ion (Li-Ion) rechargeable batteries** - is a progressively popular technology for rechargeable battery packs in which it offers better energy density over nickel and lead based chemistries at an affordable price. Lithium ion can have up to twice the energy density of nickel-cadmium. There is potential for higher energy densities. In terms of discharge the lithium-ion characteristics are reasonable and behave very similar to nickel-cadmium. The high cell voltage of 3.6 volts allows battery pack designs with only one cell. In today's applications such as, mobile phones run on a single cell. A nickel-based pack would require three 1.2-volt cells connected in series. The information in this section is gathered from [3].

Table 10: Comparison of Typical Rechargeable Batteries

Type	**NiCd**	**NiMH**	**Li-Ion**
Specific Energy Density (Wh/kg)	45-80	60-120	**150-190**
Nominal Voltage (V)	1.2	1.2	**3.6**
Load Current (C)			
- peak	**20C**	**5C**	**>2C**
- best result	**1C**	**0.5C or lower**	**1C or lower**

"The content of this table is from batteryuniversity.com [3]."

3.9.2. Power Options

As spoken earlier in this paper there are a number of possibilities for powering the MSP430. There are three major places from which to draw power: batteries, the wall AC, and the USB port of a computer.

For the purposes of this project rechargeable battery is the best choice. For the most part, a typical battery holder with a couple of wires can be connected to Vcc and Vss. There are pins for this purpose; soldering female header pins to the wires or using jumpers make battery use simple. Before making this connection, however, the MSP430 power requirements need to be considered. The MSP430 can't handle voltages higher than about 4 V.

First power option, good combinations of batteries must be considered for the purposes of powering the MSP430. For this design we will be using a nominal 3.8 V lithium-ion rechargeable battery. In general, batteries need no other external parts, though it may be a good idea to have both filtering capacitors on the MSP430 in this case as explained in the chip configuration.

Second power option, if some application requires higher voltage it can be considered battery combination to achieve higher voltage battery or use the USB power (5V), the voltage needs to be stepped down to an acceptable level. A voltage regulator can be use for the purposes of using the USB port. These components are cheap, robust, and work very well when used properly. Some can even handle such a wide range of inputs that you could connect most any common power source to it and have the output you need. As part of this paper, the details for using a voltage regulator will not be explained is something that can consider if everything else fails.

Third power option, is the wall outlet, a transformer with a rectifier will be needed. Anything in the 1.5-3.6 V range will work directly with the MSP430. There are many, many more ways to power the microcontroller. After studying the chemistry of various choices for rechargeable batteries and powering options it has been concluded that for the purposes of this project lithium-ion rechargeable batteries will best suit the purposes of this project. As explained earlier, lithium-ion is becoming the choice for portable applications because of its high capacity-to-size ratio and a low self-discharge characteristic. As to charging the lithium-ion batteries many solutions are available such as such as power management ICs, MCU controlled or even logic devices. When considering safe charging, time efficient and low cost solution the MCU controlled charging method can be considered as a recharge solution. The capacity of a battery, C, which is expressed in mA-hours, this measures the lifetime of the battery between charges. The batteries current has the units C-rate. Considering an example where given a battery with capacity of 500 mA-hours, then the battery will have a C-Rate of 500mA. The current equivalent to 1C is 500 mA and for 0.1C is 50 mA.

Three stages are involved in the charging process of a Li-Ion battery:

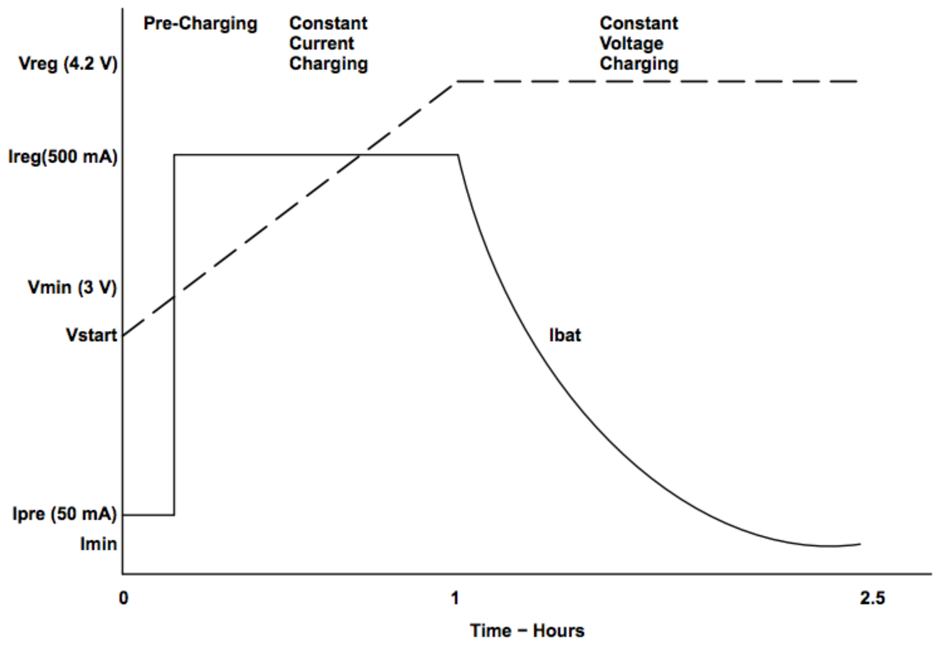
- **Slow Charge** - the battery is charged with a constant low current of 0.1C, the voltage in battery will be less than 2.5 V. Some batteries such as the Nickel-Candium suffer from a phenomenon called memory effect in which the battery is recharged with out being fully discharged which can then cause a reduction in the battery capacity. The lithium-Ion batteries do not suffer from memory effect and therefore, the battery doesn’t have to be fully discharged before charging. This slow charging stage is rarely used during the charging process of lithium-ion battery.

- **Fast Charge** - Constant current charging stage-using current of 1C is one of the most important stages during the recharge process. The lithium-ion batteries are fully charged at voltage of 4.1 V or 4.2 V. The battery is charged with a constant current of 1C until the battery is fully charged (4.1 V or 4.2V). The firmware continuously checks the charging current by sensing the voltage at the current sense resistor and adjusts the duty cycle of PWM output from the MCU.

- **Constant voltage charging stage** – While constantly checking the battery’s voltage and when the voltage reaches 4.1 or 4.2V, the charger will switch to constant voltage charging mode. The voltage charging mode, is when reached 4.1 or 4.2 V it will be charged with a constant voltage source at a fixed battery voltage of 4.1 or 4.2 V. The battery voltage will be monitored constantly and maintained at 4.1V by controlling the duty cycle of the PWM output. While this process is under taken the charging current will start to fall due to internal cell resistance. When the charging current falls below 0.1C, the charging process must stop.

Figure 21 shown below show the relationship of Current vs. Voltage during the charging progress of the Lithium-Ion explained in this section:

Figure 20: Current vs. Voltage during the charging progress of the Lithium-Ion



"Copyright Texas Instruments Products (http://www.ti.com). Used by permission."

When the battery is fully charged, most of the electrical energy is converted to thermal energy. Overcharging batteries can cause overheating, explosion due to outgassing of the electrolyte and severely reduce battery life. Li-Ion batteries are extremely sensitive to overcharging and hence it is critical that the final voltage be controlled to within ±50 mV of 4.1 or 4.2V. A battery charger design needs to be able to determine a fully charged battery to avoid overcharging. A few methods to determine a fully charged condition are:

• During the constant voltage charging stage, when the current drops to 0.1C, a fully charged condition is reached.
• Determine the battery temperature to avoid overheating
• Use a safe timing method: As long as the charging time is longer than a predetermined time, the battery can be considered as fully charged.

Given the fact that the microcontroller being considered accepts voltage in the range from 1.8V-3.6V. To keep the cost per unit down and still get good power output and battery life, lithium-ion batteries are the best option. With the voltage and cost restriction, we are limited to either using cell voltage of 3.6V or 1.2-volt cells connected in series. The information in this section was gathered from [4] and [5]. Below is considered the UltraFire 14500 900mah 3.6V rechargeable Li-Ion battery (pair) plus charger combo product features:

- 900mAh 14500 Li-ion Rechargeable Battery
- Holds 85% of it's charge over 12month period
- Can be charged over 500 times
- Lighter than standard alkaline battery

3.9.3. Linear Regulator

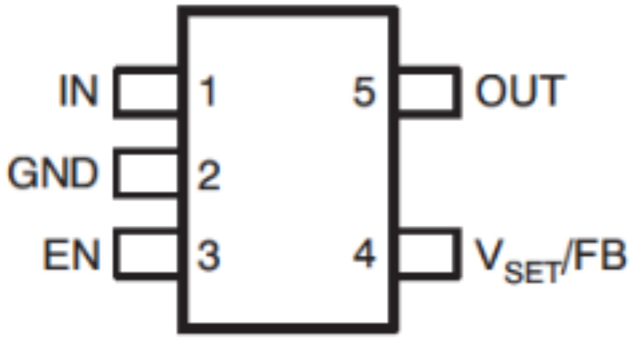
Purpose of linear regulator is to minimize MSP430 power consumption and extend battery life. A linear regulator is a system used to maintain a steady voltage. As the load varies the resistance of the regulator varies in agreement resulting in a constant output voltage. The linear regulator is made to is made to act like a variable resistor. This is done repetitive by adjusting a voltage divider network to maintain a constant output voltage. It dissipates the difference between the input and regulated voltages as waste heat. In contrast, a switching regulator uses an active device that switches on and off to maintain an average value of output. Because the regulated voltage of a linear regulator must always be lower than input voltage, efficiency is limited and the input voltage must be high enough to always allow the active device to drop some voltage.

As part of the design for the power supply it will be considered a linear regulator that will be placed the regulating device between the source and the regulated load, a series regulator, or may place the regulating device in parallel with the load, shunt regulator. Simple linear regulators may only contain a Zener diode and a series resistor; more complicated regulators include separate stages of voltage reference, error amplifier and power pass element. Because a linear voltage regulator is a common element of many devices, integrated circuit regulators are very common. For the purposes of this project it will consider the TPS780xx linear regulator to building a 3.3 V power supply. Below are key features of the TPS780xx family are considered:

- 150-mA low dropout regulator with pin-selectable dual level output
- Low dropout: 250-mV at 150 mA
- 3 percet typical accuracy
- Low total quiescent current: 500-nA
- Adjustable or fixed output voltages ranging from 1.22 V to 5.25 V

As part of the battery-powered system it will be taken into account TI's new ultra-low power TPS780xx low dropout regulator with selectable dual-level output voltages allow designers to dynamically shift to a lower voltage when the microprocessor is in sleep mode. Even if the MSP430 is operated at 7 MHz when active and placed into low-power mode when not active, this can significantly extend the battery life. When the MSP430 is in low power mode its operating currents at inputs of 3.3 V and 2.2 V are 2.13 μA and 1.3 μA, respectively. The TPS780xx series is designed to be compatible with the TI MSP430 shown in Figure 22 below:

Figure 21: TPS780xx Top View



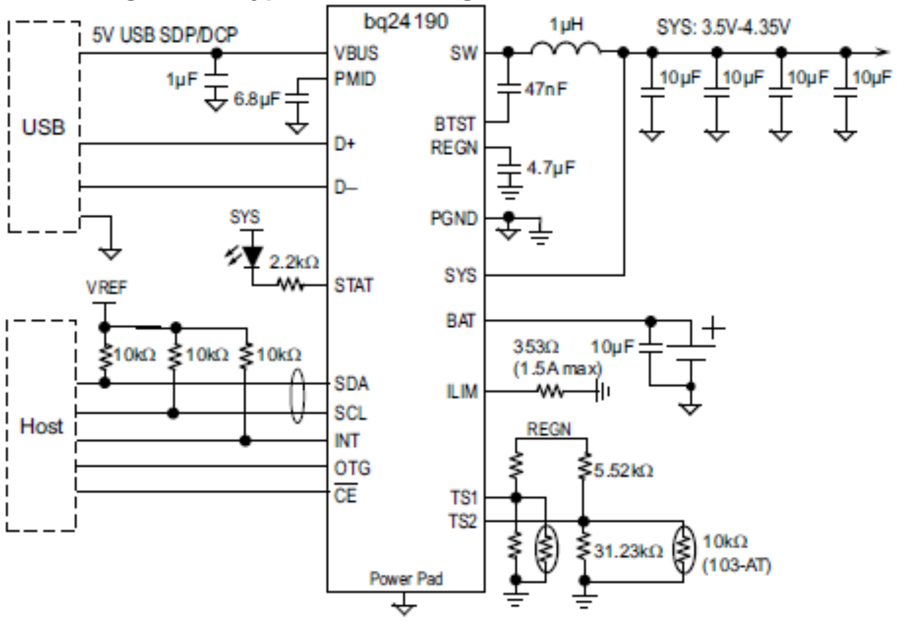
TI's Datasheet

In conclusion, to design an MSP430 battery-power system, it should be pay close attention to the proper operating voltage of the microcontroller. Overall, in this section it was considered the operating conditions of the microcontroller, the chemistry of the batteries that best suits the purpose of this project, and extending battery's life by using a linear regulator. The information in this section was gathered from [6] and [7].

3.9.4. Recharge Circuitry

As a safety precaution, as well as to extent the lifespan of the battery, a charge controller regulates the current flow in and out of the battery. For charging a battery the initial stages is a constant-source mode until the battery voltage rises to the "float" voltage. When the battery reaches the float voltage the controller switches to constant-voltage mode until the current decrease to a minimum charge (Signifying a full charge). A close estimate of the charge time of a lithium-ion battery can be obtained using the standard CC/CV algorithms by dividing the battery capacity in ampere-hours by the constant-current mode charge current in amperes and multiplying that quantity by the charge time, 1.3 hours. Although it is possible to use a standalone microcontroller, a voltmeter and a ammeter, and implement a PID controller in code to regulate the pulse width modulated signal to the transistor gate, dedicated charge ICs are available at relatively low cost. In fact, some of these ICs are designed with a USB connection as the DC supply voltage. One such example is the BQ24193, a battery charge management and system power path management device, by Texas Instruments. It can be set up over the I^2C bus eliminating the need for additional line in the design. A clear advantage of using this device, as opposed to designing a charge controller using an 8-bit microcontroller plus external sensors, it's it easy set up and low maintenance. According to the datasheet the device initiates and completes a charging cycle without software control, completely independent. Furthermore it starts in a pre-conditioning phase before moving to constant current and eventually to constant voltage. Finally it provides programmable thresholds for the minimum current in the CV phase, as well as over-voltage/over-current protection.

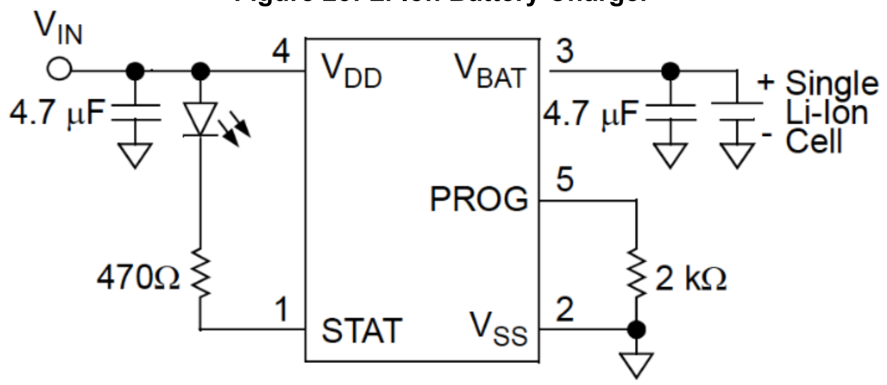
Figure 22: Typical connecting circuit for the BQ24193



As spoken earlier in this paper the microcontroller will be powered using lithium-ion rechargeable batteries. If the user wishes to use the mouse for longer periods of time a recharge circuitry would be necessary charge the circuit, therefor another method is proposed for charging the microcontroller. The USB LiPo Charger (MCP73831/2) is a basic charging circuit that allows the user to charge 3.7V Li-Ion or Li-Polymer cells at a rate of 500mA or 100mA per hour. It is designed to charge single-cell Li-Ion or Li-Polymer batteries. The USB port allows the user to connect to the computer power supply.

The board incorporates a charging circuit, status LED, selectable solder jumper for 500mA or 100mA charging current, external LED footprint, USB input, mounting holes, and various holes for your own connectors. The USB port allows the user to connect to the computer power supply. There is also a 'SYS OUT', which allows the user to connect the charging circuit directly to the batteries so it will not be necessary to disconnect the charger each time you want to use it. Figure 24 shown below shows the schematic for the Li-Ion Battery Charger.

Figure 23: Li-Ion Battery Charger



Datasheet

This charger is used to charge 3.7 V batteries but the operating conditions for the MSP430 in an input voltage between 1.8 V and 3.6 V, therefore a voltage divider will need to be added to meet the power requirements for the microcontroller and recharge circuitry. The recharge circuit is subject to change if adding the resistors in series to get a desired voltage of 3.6 V doesn't seem to work, then another recharge circuitry will need to designed. Overall, this design remains true until proven otherwise. The information in this section was gathered from [8].

3.9.5. Battery Charger

Designing a system for Li-Ion batteries requires special attention to the charging circuitry to guarantee a complete charging of the battery. For the puposes of this project the group used the bq24090 Single-Input, SingleCell Li-IonandLi-Pol Battery Charger. This device is targeted at space-limited portable applications. The devices operate from either a USB port or AC adapter.

The bq24090 has a single power output that charges the battery. Below are key features of the bq24090 which were considered:

- **Charging**
- 1% Charge Voltage Accuracy
- 10% Charge Current Accuracy
- Pin Selectable USB 100mA and 500mA Maximum Input Current Limit
- **Protection**
- 6.6V Over-Voltage Protection
- Fixed 10 Hour Safety Timer
- **System**
- Status Indication – Charging/Done

This charger was used to charge a 3.8 V Li-Ion battery in which its single output power was regulated to meet the operating conditions for both the MSP430 and Stellaris. In the next sections we will get into the details on devices that regulates the output voltage of the charger to meet the power requirements of the microcontroller.

3.9.6. Buck-Boost DC/DC Converter

The LTC3531 are synchronous buck-boost DC/DC converters that operate from input voltages above, below or equal to the output voltage. These converters provide high conversion efficiency over a wide range of load currents. Below are key features of the LTC3531 which were considered:

- Regulated Output with Input Above Below or Equal to the Outpu
- Single Inductor

- Up to 90% Efficiency
- VIN Range: 1.8V to 5.5V
- 200mA at 3.3Vout from 3.6V Input and125mA at 3Vout from 2.5V Input

The LTC3531 was used to regulate the input voltage of 3.8 V to 3.3 V to meet the power requirements for the MSP430. The details for the design of the power supply are under the design summary of this paper.

3.9.7. Charge Pump DC/DC Converter

The MCP1252 are inductorless, positive-regulated charge pump DC/DC converters. The devices generate a regulated fixed of 3.3V or adjustable output voltage. They are specifically designed for applications requiring low noise and high efficiency and are able to deliver up to 120 mA output current. The devices allow the input voltage to be lower or higher than the output voltage, by automatically switching between buck/boost operations. Below are key features of the MCP1252 which were considered:

- Inductorless, Buck/Boost, DC/DC Converter
- LowPower: 80µA(Typical)
- HighOutputVoltageAccuracy: ±2.5% (VOUT Fixed)
- 120mAOutputCurrent
- Selectable Output Voltage (3.3V or 5.0V) or Adjustable Output Voltage

The MCP1252 was used to regulate the input voltage of 3.8 V to 3.3 V to meet the power requirements for the Stellaris. The details for the design of the power supply are under the design summary of this paper.

3.10. Bluetooth Module

3.10.1. Overview

Bluetooth is a wireless communication standard for exchanging data securely over short distances. After pairing the devices enter and dynamic master and slave environment in which said devices can switch roles, by agreement. Because Bluetooth is a multi-device environment, interfacing with a computer is made easy as only one transceiver is needed. In a Wi-Fi set-up, each device will need a wireless adaptor, and a wireless router and/or wireless access point will manage communications. The main reason as to why Bluetooth makes sense for our project is its accessibility, unlike Wi-Fi, that requires configuration of hardware and software, the set-up procedure for Bluetooth is minimal. Also Wi-Fi transmit higher power signals that, although produce much better range, also consume more power. The added range is not essential, but the extra power consumption is critical for the battery life.

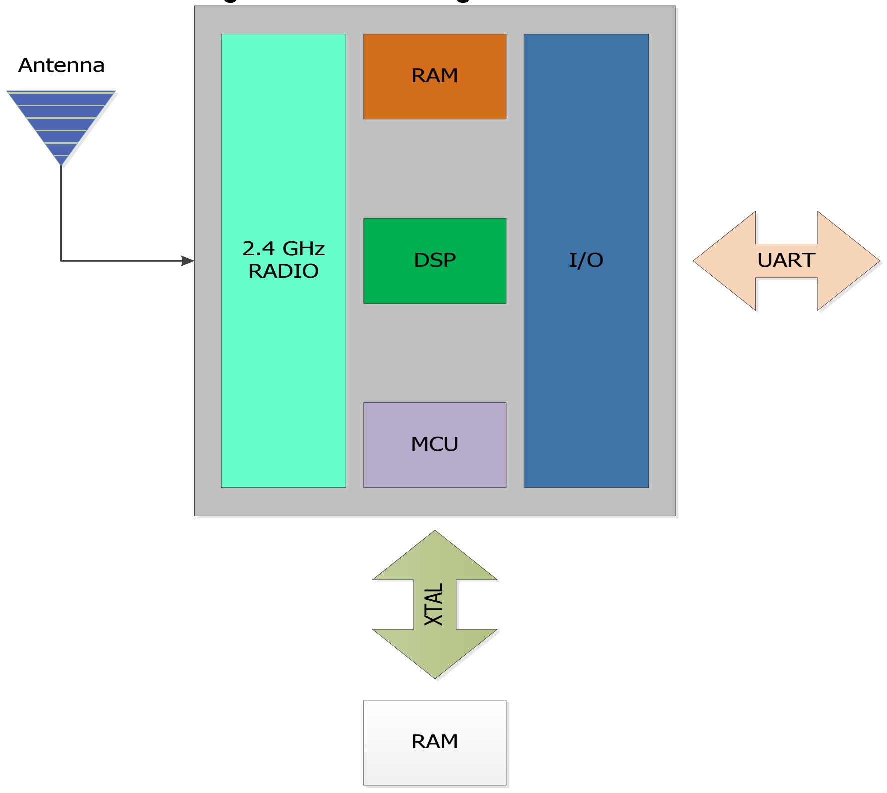
3.10.2. Bluetooth Modules

Bluetooth technology is extensively used for a variety of applications, especially in the portable electronics market. Because of its popularity, there is a wide selection of modules available for our project. In the selection process we focused on power consumption, and ease of connectivity. Features like range and operating temperature were not as critical because the design specifications are not very demanding in this regard. The first module considered was the Bluegiga WT-12. This module offers a range of up to 30 meters, with a nominal output power of +4 dBm, data rates of up to 3 Mbps, and can be interfaced via UART or SPI to the microcontroller. One nice thing about this module is a metal shielding to prevent RF interface. We suspected that, because of the nature of the project, there would be heavy RF are traffic, and this shielding would play an important role. However, after testing another module, which lacked this feature, we came to the conclusion that, for the project, the shielding is unnecessary as the devices performed similarly. Although this module meets/exceeds most of our design specifications, it is fairly expensive (25$) in comparison to other modules like the RS232 TTL transceiver module (10$). Furthermore, our design makes use of communications at 9600 baud, within 15 meters, and well within the maximum temperature ratings; any low-end module should suffice. The additional expense did not make sense for the design since other (cheaper) modules performed well enough, albeit not as capable as the Bluegiga WT-12. The other model that was reviewed was the RS232 TTL, and was chosen for the reasons that follow.

3.10.3. RS232 TTL Transceiver Module

The RS232 TTL Transceiver module is a 3.3v operated (operating voltage of the msp430g2553), Bluetooth Spec v2.0+EDR compliant, class 2 type output power device that allows wireless connection to a computer. It is also programmable to a variety of baud rates, as well as other settings like pass code and device identification name. This module is specifically made to interface a Bluetooth enabled computer and a microcontroller, as it cannot establish communications between two embedded devices. One advantage of this device over other Bluetooth modules is its usage of UART, a simple communication protocol. This feature makes interfacing the RS232 TTL transceiver module with the msp430g2553 simple, with minimal communication error at 9600 baud. Configuration of the device is accessed through the logic level of a single pin upon power-up. Once is this mode, simple commands terminated by \r\n are sent over UART. By default the module is configured to 9600 baud, one start bit, 8 data bits, one stop bit, and no parity. A basic block diagram is shown in figure 25.

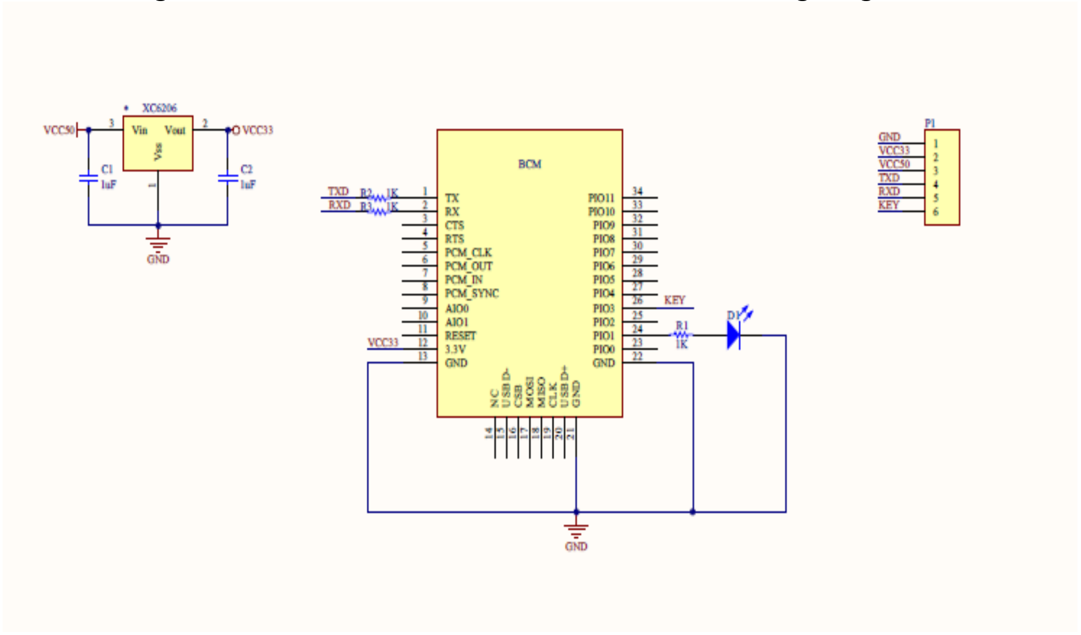
Figure 24: Block Diagram of Trasceiver



3.10.4. Board Schematics

As can be seen in figure 26 below, the layout for the device is very bare bones. This is beneficial because it reduces the bill of materials needed for the project (Cost) as well reduce the possibility of errors during assembly, since none of the group members have prior experience designing and soldering a custom PCB board. This device was chosen for its functionality, but also because its layout simplicity. Lastly the actual chip is very inexpensive; it can be found in online sites such as eBay, and amazon for less than 10$ with shipping and handling included. As stated in the design specifications, Low cost is one of the design goals, and this module helps achieve that goal, without resulting in a notable performance drop in the project.

Figure 25: Schematic of Bluetooth module with voltage regulator



3.11. PCB

As of now two PCBs were ordered for the camera and glove. In this section the pricing of the PCBs were considered in which choosing the right PCB vendor was critical for the project's success. There are many vendors to choose from when it comes to ordering this section; therefore it is extensive and detailed. The group decided to set criteria for choosing the PCB vendor for the project. Because this project requires particular specifications, a set of criteria were developed which required the vendor to comply with this specifications to master the end result the camera and glove. Later in this paper the details for the budget of the group will be considered but is important to mention it in this section. It was determined that price and completion time were considered as part of the criteria for this project.

Because the project is set to a limited budget, it was needed to ensure that the vendor offers PCBs within the designated price range. The maximum amount of money allotted for the PCB is $200. However, the group set a goal to find a PCB that is well under the maximum allotted price range. By setting a specific maximum price range, the group was able to quickly eliminate vendors that do not meet the proposed requirements for the PCB design.

Another criteria to consider is the time it takes for the PCB to be delivered which would eventually was an important parameter to consider. The project is restricted in schedule, therefore it was vital that a vendor is found to have a quick turnaround time. Finding a vendor that has a fast turnaround will allow the team to build this part of the project as early as possible. The PCB is the foundation for the project; therefore, it is important that the team begins working with it so that issues that could arise can be found earlier. Without the PCB, the project would not succeed. This is why finding the right vendor is so important.

Also, the team has a standard size for its PCB, which is 12 square inches and 4 layers. By setting a PCB size for this project, it diminishes any chance of

miscommunication regarding the ordering of the PCB. The size is important to be considered due to the project requirements in which the wireless-wearable mouse should weight less than 3 lbs. Setting this standard size for the PCB allows the team to make sure that it fits the other design elements of the project. When searching for the prices of the PCB, using the size requirements is easy to draw a prize by selecting the size of the PCB. Therefore, the exact price is known for the PCB from the specific vendor.
Out of the four group members no one is familiar with the PCB vendors. In order to best vendor for the needs, the team decided that it was beneficial to research five different vendors. Using developed criteria, the team was able to narrow down the pros and cons of each vendor, as well as any additional companies needed to contact, such as for assembly.

3.11.1. Express PCB

The first vendor to consider is ExpressPCB, which offers several convenient services. This vendor includes several packages, which includes standard, miniboard, production, protopro, 4 layer productions, and 4 layer protopro. The option considered is the 4 layer production. This package allows the group to capture the 4 layers needed for the project's PCB. The 4 layer production allows a more compact design, as well as eases the amount of work involved for layout. Express PCB also assembles the PCB, making it less work on the design end as far as contacting a company that assembles PCB's. They also offer the PCB size that the group is looking for. The total lead-time for Express PCB was 10 days.
The pricing was determined using a set formula. They particular vendor has a 10 day option, which adds option, \$273 + (.68 * Number of boards * Board area in square inches) + (.5 * Number of boards) – Quantity Discount + Shipping. Express PCF requires an order of at least 2 PCBs in which the team will make an order of two PCBs. The total cost for the PCB without shipping is \$290. The team hasd a set budget of \$200 for the PCB, which wouldn't meet the group budget needs, the group decided to explore other options.

3.11.2. PCB4Less

The website PCB4less.com is a very known PCB manufacturer and offers a short lead-time of 5 days. They offer the option to test the PCBs before sending them out to the customer. By testing their products before delivery they can validate that all specifications are met and the device is fully functional. This feature assured the group that the PCB was build correctly and reduce testing time. This website offers all of the services that Express PCB does however the way they price their products is very different. PCB4less has a very complex quote system.

3.11.3. PCB Express

Another vendor which the group explored is PCBExpress is another excellent website to order PCBs from. This particular vendor is not associated with the vendor mention earlier ExpressPCB even though the names can be very similar.

This vendor requires that at least two PCBs at a time be ordered. This vendor also assembles the PCB like the other vendors researched. They offer the correct PCB size for the project, and have a simple ordering system. The total lead-time was 3 days.
For the group's specifications, the total cost will come to $500 without shipping. Because of PCB Express' high cost, the group decided that this was not a valid option for this project. The price exceeds the allotted amount by almost $200, an amount that is too high for the group to even consider expanding the PCB budget.

3.11.4. Ultimate PCB

After much research on PCB vendors the group stumbled upon the website Ultimatepcb.com. This website was found to be very simple to navigate and offers a much more user friendly quoting system. An interesting feature about this vendor is that they offer many customization options to the costumers. One of the things that called the groups attention is that they allow the customers to order single PCBs rather than bulk orders. This is a great advantage to the group since only two boards are needed for this project. Ultimate PCB also offers a variety of sizes which one of those are ideal for this project. The total turnaround time was 5 days.

Although this vendor has many great features to offer for this project they are out of the budget. The price of the PCB from this vendor averaged $230 without shipping. Therefore, it goes over the price range and all of the group's criteria.

3.11.5. Imagineering Inc.

Imagineering Inc. is another PCB vendor that was researched for this project. The website for this vendor is also very user friendly and offers many customization options. Although they meet all the PCB requirements for this group they are outside of the price range for the group. The minimum number of boards the group can order is 5 which also exceeds the number of boards needed. This vendor also charges a tooling fee and he total price after the $200 tool charge was approximately $635. The lead-time is 7 days. It was after the group discovered the high price of their products that this vendor was eliminated from the options.

3.11.6. 4PCB.com

Finally the group decided to research 4PCB.com. This website is very famous as they offer special deals on generic size PCBs. Some of the advantages of using this website is that they offer a free PCB layout software called PCB Artist. In this software you can layout the board and upload it so they can print it. Not only that but the group will be able to order multiple boards at a low price. The boards come with white legends, 2-Layers and are 60 square in. The boards are also lead free, routed to overall dimensions and have all the wholes plated. the lead time for this vendor is 5 days. For the price and many other reasons the group will choose 4PCB.com as the PCB vendor of choice.

3.12. Host Computer

For this project it was require that we had a host computer to integrate with, this project is after all a control device for a computing system. For the exact specifications of the required system, it requires a PC that meets the requirements for and has installed Microsoft Windows 7 or later version, and has multiple free and operating USB ports. Because of the nature of the device being created there should not be other requirements place specifically on the host computer. With all processing of data happening on the MSP430 and Stellaris the host computer should not have any computational load larger than that of a typical optical USB driven mouse.

3.12.1. Drivers

For any external device to interface with a PC it must have a driver which tells the system how the device behaves and how it is suppose to interact with the system. These drivers hold the specific instructions that can be passed then from the device to the host system and vise versa. For this project the host system will be a Windows OS system. The Windows kernel-mode I/O manager manages the communication between applications and the interfaces provided by device drivers. Because devices operate at speeds that may not match the operating system, the communication between the operating system and device drivers is primarily done through I/O request packets (IRPs). These packets are similar to network packets or Windows message packets. They are passed from operating system to specific drivers and from one driver to another.

The Windows I/O system provides a layered driver model called stacks. Typically IRPs go from one driver to another in the same stack to facilitate communication. For example, a mouse driver would need to communicate to a USB hub, which in turn would need to communicate to a USB host controller, which would then need to communicate through a PCI bus to the rest of the computer hardware. The stack consists of mouse driver, USB hub, USB host controller, and the PCI bus. This communication is coordinated by having each driver in the stack send and receive IRPs. The communication of the IRPs to and from the GRID device driver must be synchronized properly for the whole stack to operate efficiently. If the driver is part of a stack and does not properly receive, handle, and pass on the information, it has the ability to cause system crashes.

3.12.2. Integration

To integrate GRID to the host computer there are several items to consider. First of which is how is GRID going to be connected to and communicate with the host computer. To transmit their input, typical cabled mice use a thin electrical cord terminating in a standard connector, such as RS-232C, PS/2, ADB or USB. Cordless mice instead transmit data via infrared radiation (see IrDA) or radio (including Bluetooth), although many such cordless interfaces are themselves connected through the aforementioned wired serial buses. GRID was coded using Bluetooth signal to communicate from the gloves and use PS/2 within the device driver to communicate to the host computer. In default mode (called stream mode) a PS/2 mouse communicates motion, and the state of each button, by means of 3-byte packets. For any motion, button press or button release event, a PS/2 mouse sends, over a bi-directional serial port, a sequence of three bytes, with the following format:

Table 11: 3-byte packet

	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Byte 1	YV	XV	YS	XS	1	MB	RB	LB
Byte 2	X movement							
Byte 3	Y movement							

XS and YS represent the sign bits of the movement vectors, XV and YV indicate an overflow in the respective vector component, and LB, MB and RB indicate the status of the left, middle and right mouse buttons (1 = pressed). PS/2 mice also understand several commands for reset and self-test, switching between different operating modes, and changing the resolution of the reported motion vectors. Additionally, GRID has an extended version of this protocol to incorporate information gathered by both the gyroscope and accelerometer. This allows the user to execute hand gestures that GRID will interpret and pass to the computer as events to be handled by the device driver.

Another item to consider when integrating GRID is timing. A mouse is usually initialized to generate movement packets at a particular rate. The default rate is 100 packets per second if the mouse is being moved. A mouse also generates a packet if a button is either pressed or released. This sample rate should provide a good rate of transmission though if there are problems the rate can be increased or decreased as appropriate.

3.13. Gesture Recognition Data Acquisition

3.13.1. Overview

Gesture recognition is the process of interpreting human actions/motions through the use of mathematical algorithms. Common uses include algorithms to interpret sign language, as well as facial expressions. In our project we hope to be able to use sensor data (accelerometer, and gyroscope) to capture movement, be able to detect pre-defined motions, and translate them into computer actions.

3.13.2. Accelerometers

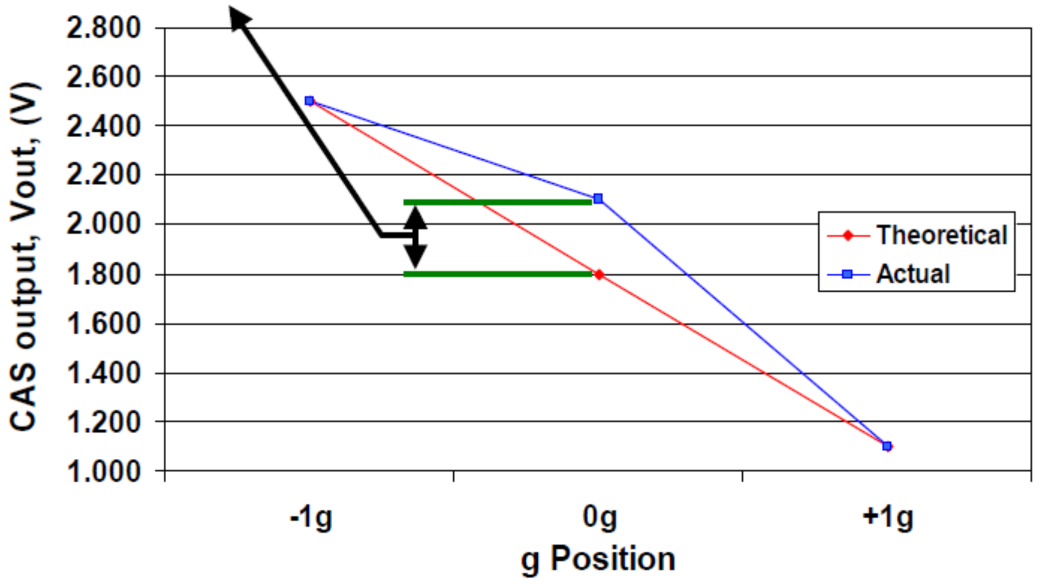
Accelerometers are electromechanical devices capable of measuring acceleration forces. They are sensitive to static forces, like gravity, as well as dynamic forces, like shaking the accelerometer. In our project we mainly care about dynamic forces. Measures of static forces like gravity give us insight on the orientation of the object by simple analysis of the vector component being measured with respect to 1g, a unit of acceleration due to Earth's gravity at sea level. However, we will have access to a gyroscope, which provides more accurate and complete information on the orientation of the object. There are several parameters to differentiate accelerometers; the ones most important for the design are: Number of axes, maximum swing, sensitivity, linearity, and bandwidth.

The number of axes, like the name suggest, is the number of axes in Euclidean space the accelerometer is sensitive to. 2-axis would mean the accelerometer can measure changes in velocity in two perpendicular directions, X and Y. Because natural human actions and mannerism are not restricted to a two dimensional plane, if we hope to identify gestures with a high recognition rate, we need only look at 3-axis accelerometers.

Maximum swing refers to the range of values the accelerometer can measure. As an example in order to measure tilt using Earth gravity, a rage of ±1.5g would be sufficient. According to Texas Instruments an Indy car driver in a corner feels 3gs, a Bobsled rider in a corner is subject to 5gs, and most humans are unable to maintain consciousness under more than 7gs. From these numbers, we do not expect to encounter acceleration readings higher than 3gs for hand motion as any higher would be unnaturally fast.

Sensitivity relates to a large signal variation over small ranges in acceleration, resulting in more accurate measurements. This aspect of the accelerometer is the most crucial for our project. In order to extract the features of the signal more easily, and with high recognition rate, there needs to be a clear distinction between signal values from accelerations of little variation. Linearity is also important since our model assumes a linear relation between the output signal of the accelerometer and the external input. Figure 27 shows how realizable accelerometers deviate from a linear relation.

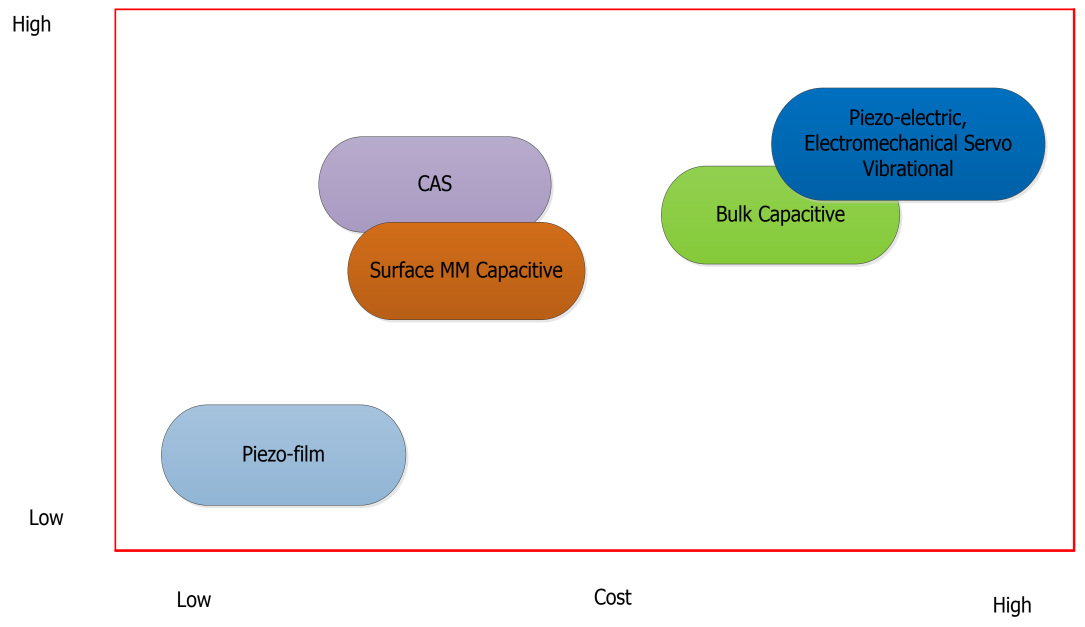
Figure 26: Deviation of Accelerometers



Bandwidth is a measure of the number of times a reliable acceleration sample can be read from the accelerometer. Hand gestures produce relatively slow changing signals, and thus, do not require a high sample rate in order to accurately record the motion. Furthermore, as will be explained in section **5.1,** the recognition algorithm is of order N^2(For signals of the same length) and the MCU, having limited processing power and memory capacity, would not be able to complete the algorithms within a second. Since one of our design specifications calls for real-time response, we sacrifice some accuracy in terms of the number of samples we read in order to save computing time.

There are various mechanisms that govern the functionality of the accelerometer: Capacitive and Piezoelectric to name a few. In a Capacitive set up a micro machined feature produces a capacitance; the change in capacitance due to acceleration is measured. Piezoelectric depends on a piezoelectric crystal that produces an output voltage under compression or stretching. Figure 28 shows the different types of accelerometer and their price vs. performance.

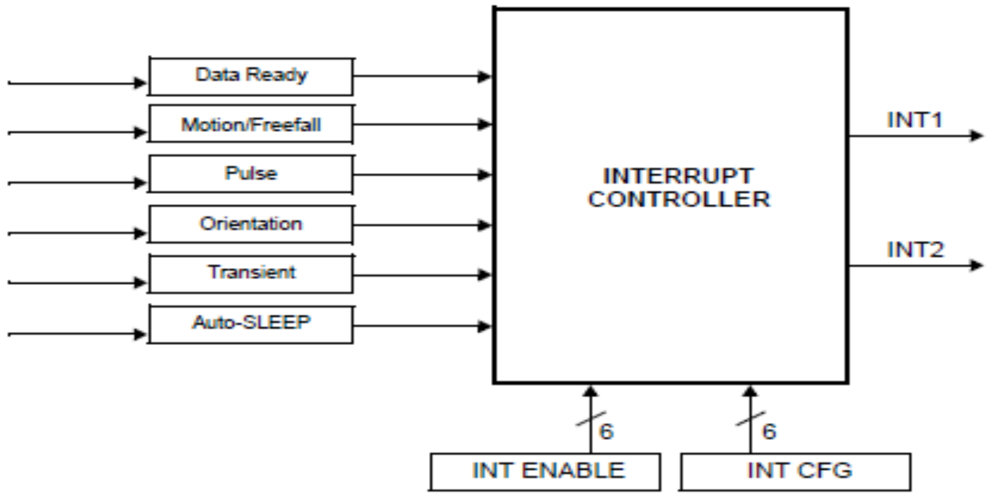
Figure 27: Cost vs. performance of the available accelerometer technologies.



3.13.2.1. MMA8452Q Accelerometer

One of the devices that we looked at was the MMA8452Q accelerometer from Sparkfun. The MMA8452Q is a digital accelerometer with a maximum resolution of 12 bits, and using a capacitive micro-machined mechanism. It operates at user programmable scales of ±2g/ ±4g/±8g as well as user programmable interrupts via two external pins. Additionally the output data rates range from 1.56 Hz to 800 Hz, much more than needed for the design. An important feature of this device is its low-power capabilities using motion interrupts, which also offloads the MCU by not requiring polling of the accelerometer. There are six configurable interrupt sources in the MMA8452Q: Data Ready, Motion/Free-fall/Pulse/Orientation/Transient/Auto-Sleep. Each can be enabled or disabled through software. The interrupt generation architecture is more clearly depicted in figure 29.

Figure 28: Interrupt controller block diagram.

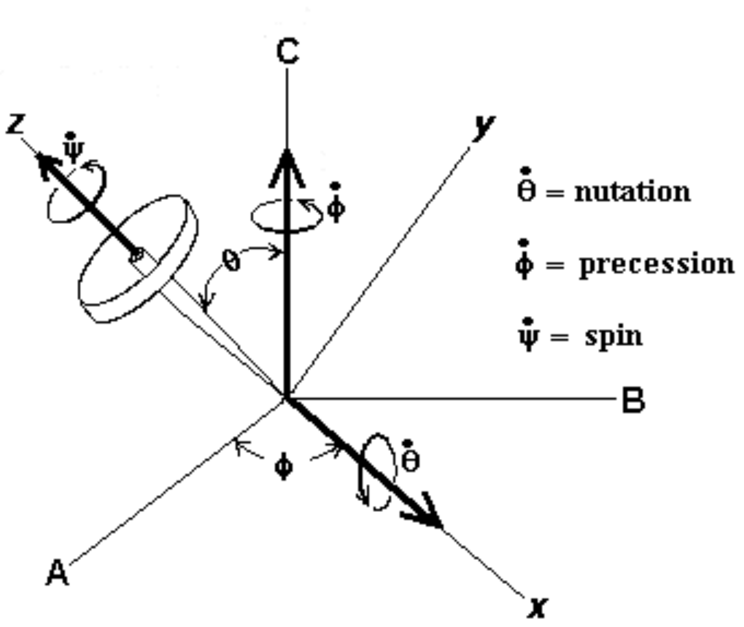


Advantages of this device include its relative low cost (10$ at Sparkfun), high sensitivity (1024 counts/g in 2g mode), as well as low pin count. The chip itself only has 16 external pins, with some being tied together to ground or Vdd. The device provided a way to check the transducer functionality without any external stimulus via a self-test. Like most controllers there is a trade-off between resolution and power. The highest resolution can be achieved at 1.56Hz by oversampling at the cost of an increase in the power requirements. Overall the MMA8452Q is a flexible and powerful chip that meets the project requirements.

3.13.3. Gyroscopes

A gyroscope consist of a disk with a high rate of spin mounted on two gimbals, or rings, as to minimize external torque. Due to the disk high moment of inertia and angular momentum, the system resists changes in orientation for as long as the disk keeps rotating. Since the gyroscope maintains its orientation, by suspending it in a special cage, we can then proceed to measure the orientation of the housing with respect to the spinning axis of the gyroscope. Some of the uses for gyroscopes include: computer pointing devices, navigational tools, and portable electronics. There are several different technologies associated with the fabrication of gyroscopes, among them are: micro electro-mechanical systems (MEMS), fiber optic gyroscopes (FOG), and vibrating structure gyroscopes (VSG) to name a few. A general depiction of the forces at work in a gyroscope can be seen in figure 30.

Figure 29: General gyroscope depiction



Because of their low cost and accessibility, we will focus on MEMS gyroscopes for our design. Furthermore, for ease of use, as well as to lower the pre-processing of the signal by the microcontroller, we have limited the groups selection to triple-axis digital output gyroscopes. The first model researched was the ITG-3200 triple-axis digital-output gyroscope by InvenSense.

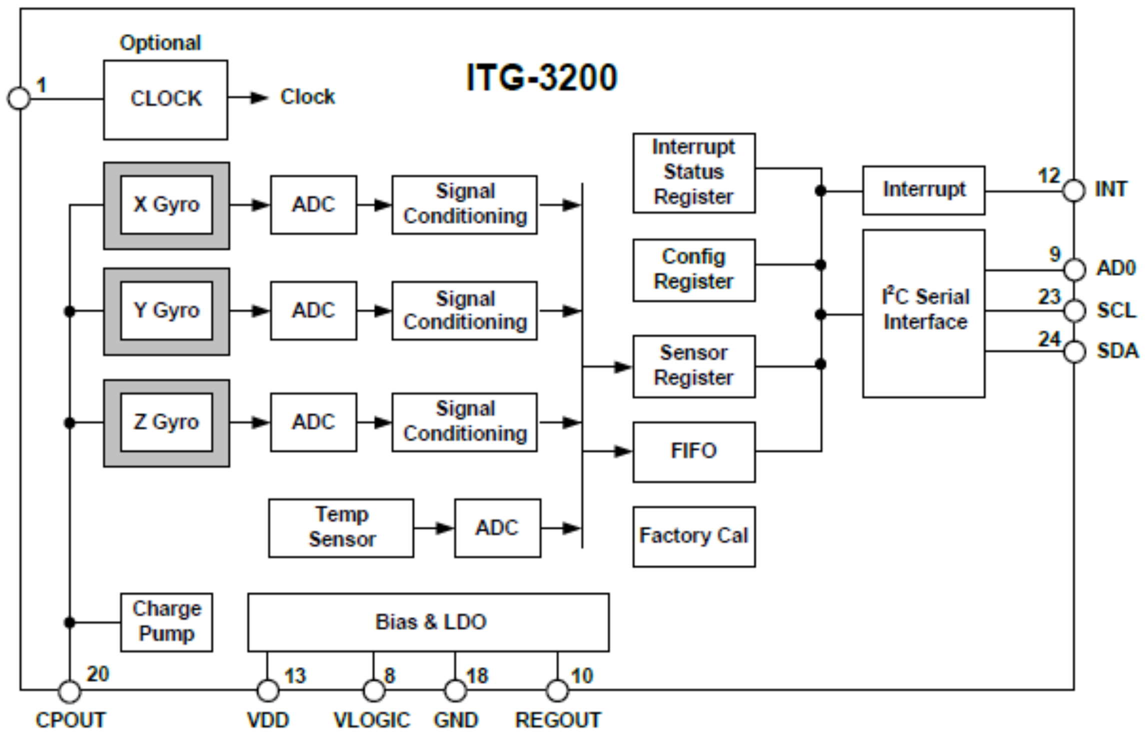
3.13.3.1. ITG-3200 Digital Gyroscope

The ITG-3200 is a single chip gyroscope solution in an extremely small 4x4x0.9mm package. It features enhanced bias and sensitivity temperature stability, reducing the need for user calibration. Additionally the low frequency noise of the device is lower than previous generations making for a more accurate device. It comes equipped with a 16-bit analog-to-digital converter for digitizing the analog gyroscope outputs, and is capable of two wire Fast-Mode I2C (400kHz) communications. The most appealing feature of this chip is that, due to gyroscope technology breakthroughs implemented on the chip, it manages to deliver a 50% power reduction when compared to competing multi-chip gyroscope solutions. According to the ITG-3200 product specification, the operating current is 6.5mA while the standby current is just 5uA at a reference voltage of 3.3v, resulting in minimal power consumption.

Power dissipation can be controlled through the power management register. This design depends on the user to indicate when to “listen in” for gyroscope gesture, therefore we will take advantage of this by putting the gyroscope to sleep mode. The ITG-3200 digital gyroscope is flexible enough as to allow “stand by” mode on independent axis.

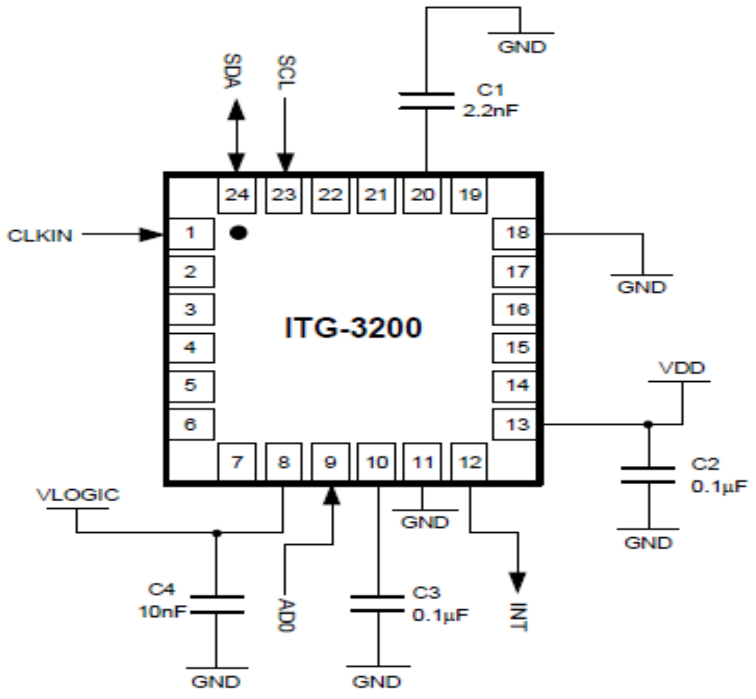
The controller will function based on events, or interrupts, and the ITG-3200 provides an interrupt pin. Interrupts are configured and used through the interrupt configuration and interrupt status registers, 23 and 26 respectively. The interrupt configuration register allows the user to define: the logic level on the pin, an open drain or push pull configuration, for data to latch until the interrupt is clear or a 50us pulse, the clearing method, and finally the enable bits for the interrupt sources. Through the interrupt status register we can understand the source of the interrupt; however, it is limited to only raw data ready or PLL ready.

Figure 30: ITG-3200 block diagram.



This gyroscope package, much like the MMA8452Q, is of a low pin count, reducing the bill of materials, and thus the overall cost. The cost of the chip is also very affordable. Sparkfun sells it for less than 30$, but it can be found in online sites, like eBay, for 20$. Additionally InvenSense offers of to 6 chips as free-samples for student projects like this one (Academic). A typical operating circuit for the ITG-3200 can be seen on figure 32.

Figure 31: Typical circuit configuration for the ITG-3200.



3.13.3.2. MPU 6050

The MPU-6050, also by InvenSense, features a three-axis accelerometer, as well as a three-axis gyroscope. It is set apart from the other contenders in its inclusion of a digital motion processor. At this time, documentation on how to take advantage of the DMP is scarce, but according to InvenSense the device is capable of motion detection like: Free-fall, high gs in any of the three axes, panning, zooming scrolling, zero-motion, tap as well as shake.

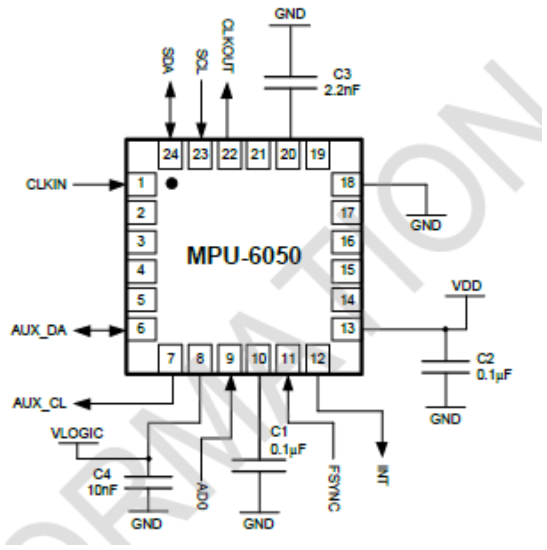
The option to add an external magnetometer, or any other digital sensor, and interface it with the MPU 6050 gives it an edge for future upgradability. The device gives the user two options to access the external sensor. The first is "pass-through mode", in which the MCU communicates directly with the external sensor via I2C. The second option is that MPU serves as a Master for the external sensor and store the output data in its own data registers. An advantage of the latter option is that the "data ready" interrupt includes external sensor data, which minimizes the number independent read operations required. All communication between the sensors, as well as with the MCU, is done via I2C minimizing the number of lines.

Like the other devices we looked at, the MPU 6050 uses three 16-bit analog-to-digital converters for the gyroscope outputs, and another 3 analog to digital converts for the accelerometer. It is clear that this device does not sacrifice speed to incorporate both a gyroscope and an accelerometer as they function independently of each other (No multiplexing of A\D). The gyroscope's and gyroscope's full scale range are user programmable, their range being: ±250,

±500, ±1000, ±2000/sec for the gyro and ±2g, ±4g, ±8g, ±16g for the accelerometer. Both scales exceed the minimum required for accurate hand gesture recognition. The sampling rate of both gyroscope and accelerometer outputs is user defined, and can be subject to a digitally-programmable low pass filter. The MPU can be programmed to send an interrupt signal to the MCU when a write operation has been completed to all the output registers. In the case where sampling rates differ the MPU will send the slower sample multiple times until a new value is written. These values can be held in a 1024 byte FIFO buffer that allows the system processor to read data in burst, and then enter a low-power mode while the MPU collects more data. An Interrupt can also be enabled to indicate when the FIFO buffer is full.

The MPU 6050 provides a better solution for sensor data acquisition than the independent ITG-3200 and the MMA8452Q. The device provides advantages in key areas like: size on board, bill of materials, cost (the MPU 6050 is priced lower than the ITG-3200 and the MMA8452Q), and power consumption. Lastly, it's more convenient for the program/programmer to have all the data in on place, as opposed to accessing different slave addresses. Our program will take advantage of the FIFO buffer to burst read sensor data within a given time-frame, a process not possible on the competing devices.

Figure 32: MPU-6050 typical operating circuit.

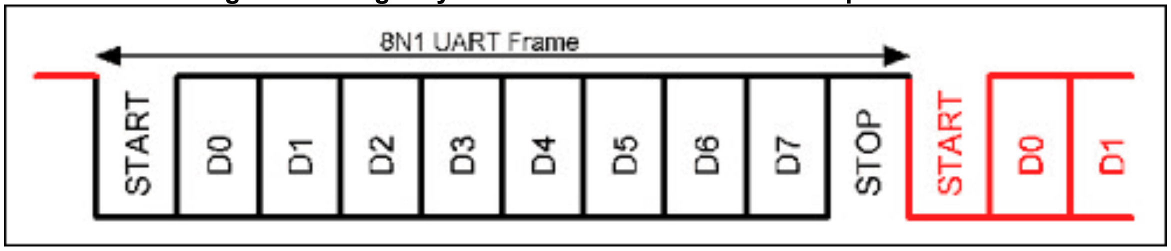


3.14. Communication Protocols

3.14.1. The UART protocol

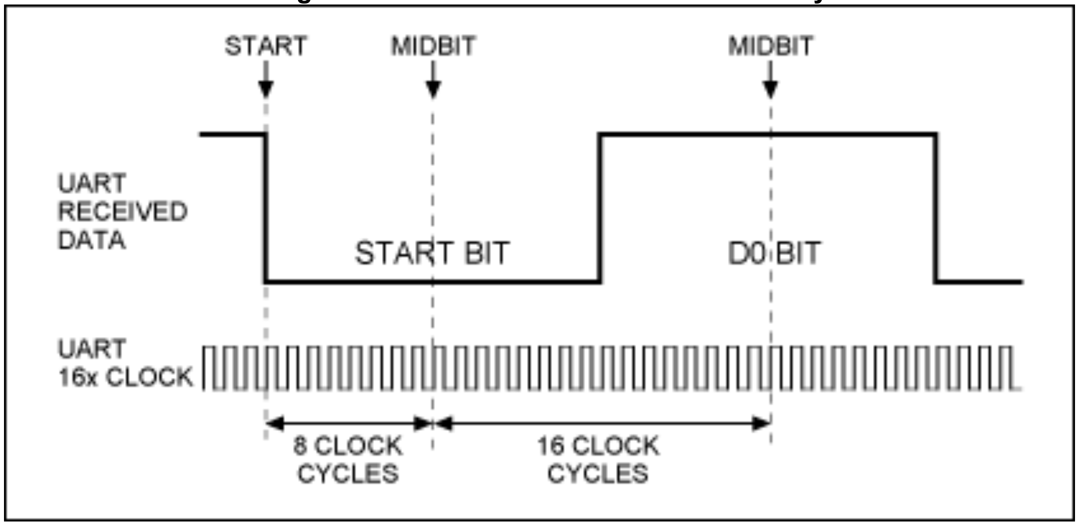
UART or Universal Asynchronous Receiver\Transmitter protocol is a standard serial communication subsystem of a computer. UART transmit a byte of data by sending individual bits in sequential fashion. In asynchronous communications a transmission starts with a "start bit", captured by the receiving device. In our case since the TXD and RXD lines idle "high", a start bit is signified by pulling the line "low". Once the receiver captures this event, it offsets its timer by one and a half the bit time for the given clock source and baud rate. The timer will generate and interrupt during the middle of the first transmitted bit to be read. This process is repeated until the 8th bit is read. As it's evident, the receiver and transmitter must both agree on a specific baud rate, otherwise bit times will be off resulting in an unsuccessful transmission or false data. A single byte UART transmission can be seen in figure 34.

Figure 33: single byte serial transmission in UART protocol.



Timing requirements are more clearly depicted in figure 35, and from then it can be seen that the clock source must have little variation over the operating condition to avoid sampling during transitional periods.

Figure 34: UART bit time over MCU clock cycles.



Another source of error is the selection of higher baud rates with respect to the microcontroller clock speed. As the baud rate approaches the MCU clock, discrete time intervals may not land during the desired sampling time. As such the msp430g2553 allows UART speeds of at most one-third of the module's clock source. The msp430g2553 also provides a modulation step to minimize timing error. The modulation register values are calculated using the following formulas:

$$N = \frac{f_{BRCLK}}{Baud\ Rate} \quad UCBRx = INT(N) \quad UCBRSx = round\ \left(\left(N - INT(N)\right) \times 8\right)$$

The msp430g2553 dedicated UART module runs independent of the CPU and can be sourced from the ACLK or SMCLK. An advantage the group will exploit by entering low power mode and only sourcing the clock signal used by this module. Furthermore it minimizes the transmission error by including a variable modulation stage to compensate for timing errors. Typical timing errors of various configurations of source clock and baud raters are given in table 12. Lastly UART protocol uses a two wire interface: TXD, and RXD. Due to the limited availability of GPIO in the mspg2553 the RS232 TTL Transceiver module provided the best wireless solution for our project.

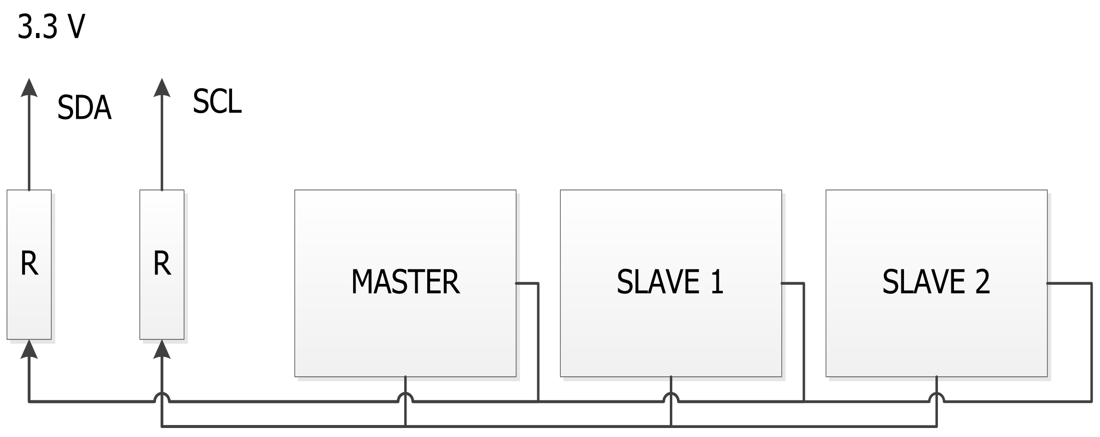
Table 12: typical transmission errors given various BRCLK clock speeds as well as baud rates

BRCLK frequency [Hz]	Baud Rate [Baud]	UCBRx	UCB RSx	UC BR Fx	Maximum TX Error [%]	Maximum RX Error [%]
32,768	1200	27	2	0	-2.8 – 1.4	-5.9 – 2.0
32,768	2400	13	6	0	-4.8 – 6.0	-9.7 – 8.3
32,768	4800	6	7	0	-12.1 – 5.7	-13.4 – 19.0
32,768	9600	3	3	0	-21.1 – 15.2	-44.3 – 21.3
1,048,576	9600	109	2	0	-0.2 – 0.7	-1.0 – 0.8
1,048,576	19200	54	5	0	-1.1 – 1.0	-1.5 – 2.5
1,048,576	38400	27	2	0	-2.8 – 1.4	-5.9 – 2.0
1,048,576	5600	18	6	0	-3.9 – 1.1	-4.6 – 5.7
1,048,576	115200	9	1	0	-1.1 – 10.7	-11.5 – 11.3
1,048,576	128000	8	1	0	-8.9 – 7.5	-13.8 – 14.8
1,048,576	256000	4	1	0	-2.3 – 25.4	-13.4 – 38.8

3.14.2. The I^{2}C Protocol

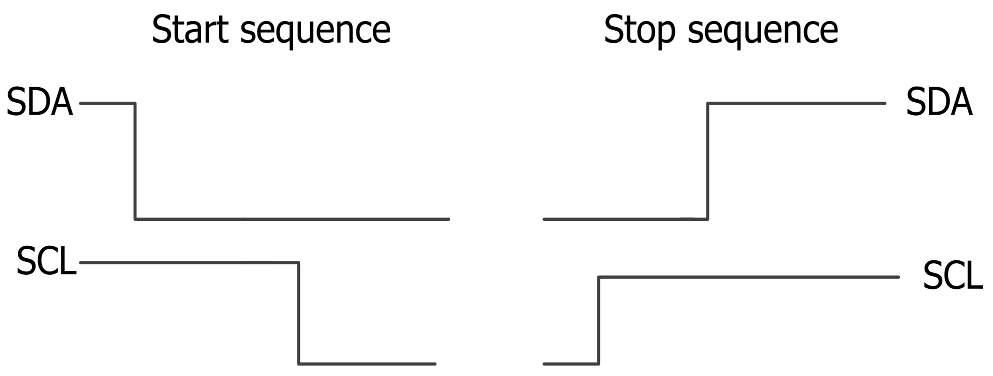
The communications protocol used for interfacing the StellarisLM4F120 with any of the aforementioned digital sensors is I^{2}C. I^{2}C, also known as a two-wire interface, is a multi-master, multi-slave environment meaning that, should the design make use of the ITG-3200 and the MMA8452Q, no additional bus lines are required (No additional pins on the StellarisLM4F120). The I^{2}C bus consists of two bidirectional open-drain lines, the Serial Data Line (SDA), and the serial clock line (SCL), with both lines needing pull-up resistors for proper functionality. Through some testing it appears that the value of the pull-up resistor is not critical; the bus operated with values in the range of 2k to 47k.

Figure 35: I^{2}C connection diagram.



Under normal transmission the SCL line is driven by the master, with the SDA controlled by the master for a "write" operation and by the slave for a "read" operation. No transmission can be initiated by any of the slave devices over the I^{2}C bus. The I^{2}C protocol allows for multiple masters but, since the design is a one master environment, it is not relevant to the project and omitted in this paper. Transmission is initiated by the master with a start sequence, defined as a transition from high to low on the SDA line while the SCL line is high. Following the start sequence data is transferred in 8 bit packets. All data bits must be stable while the SCL is high for a successful transfer; meaning changes in the SDA line must occur while the SCL line is low. Terminating a transfer is also up to the master. In order to cease communications and free the I^{2}C bus a stop sequence must be issued. The stop sequence consists of a transition from low to high on the SDA line while the SCL line is high. During a "read" operation, and in the event the slave is not ready to send data, it can hold the SCL line low. This is called "clock stretching" and allows the slave to place the data on its shift registers when ready. Furthermore, clock stretching prevents the master from continuously sending out clock pulses with no response by the slave.

Figure 36: I^{2}C Start and Stop sequences.



The first sequence of 8 bits is made up of 7 data bits, which refer to the slave address to be accessed, and on read/write bit to specify the type of transfer. The I^2C protocol allows for 10-bit addressing mode, but is not relevant to the project and thus omitted in this paper. Once the master sends the last data bit, and if the transmission was successful, the slave must send an ACK (acknowledge) signal by pulling the SDA line low. After the slave address is sent and acknowledged, the next 8-bit sequence specifies the slave register to be accessed. For a write operation, the register number is first sent, followed by the first byte of data (data written to the register specified), subsequent data transfers are written to the slaves registers in ascending order. A read operation is slightly more complicated. In order to read from a given register from the slave, the master must first write that register address, thus a reading sequence starts with a write operation. Once the register address has been transmitted the master sends a repeated start bit, followed by the slave address with the read/write bit set. The slave proceeds to continuously send data until a stop sequence is sent.

Figure 37: I^2C read sequence starts with a write operation (R/W set to write).

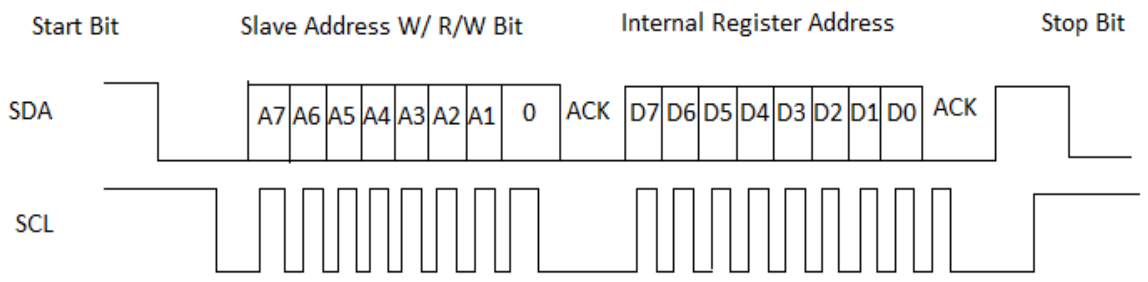
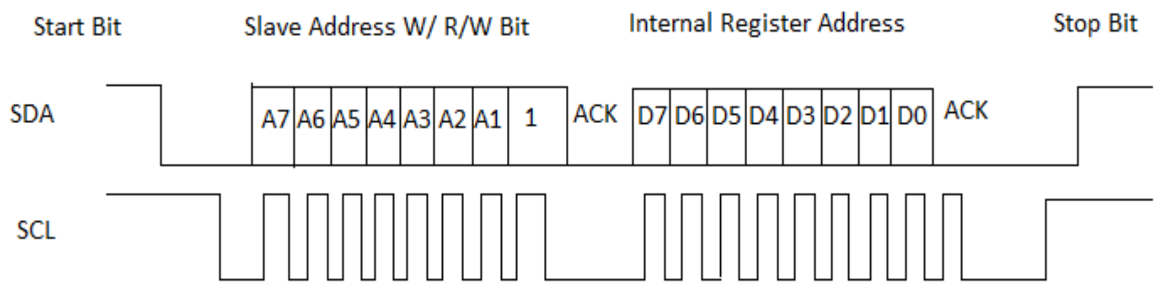


Figure 38: I^2C after register has been written, begin to read (R/W set to read).



The main drawback of this serial communication protocol is the relatively slow transmission speeds. All the digital sensors that have been researched transmit in “fast-mode” which clocks at 400 KHz. Luckily, hand-based gestures are relatively slow-changing signals, therefore this max speed of 400 KHz is much faster than the aforementioned sensors sampling rate.

3.15. Gesture Recognition Algorithm

3.15.1. Dynamic Time Warping

Dynamic Time Warping is a well-known algorithm/technique that outputs a score based on the similarities between two given (time-dependent) sequences, under certain restrictions. Dynamic Time Warping matches points in the sequences in a non-linear fashion. It also has been extensively used to compare speech patterns in automatic speech recognition scenarios. It is expected that similar success can be achieved in pattern recognition via gyroscope and accelerometer measurements.

The main requirement for this algorithm to function is that both data sequences (signals) be sampled at equidistant points in time. Since it is up to the user to configure the sampling rate of the gyroscope and accelerometer, this requirement can be met using any of the aforementioned devices. This approach is relatively simple in complexity (Compared to Hidden Markov models), and can also be adapted to meet the hardware limitations in a microcontroller. In the paragraphs that follow, the mathematical background of the model will be explained and adapted to a microcontroller friendly design.
Prior to diving into the algorithm, the user must first define the "cost function". "Cost" refers to the measure of the difference between matching points in the sequences. There are several ways to compute this difference, some are: Manhattan difference, and Euclidean distance.

$D_e = \sqrt{(Q_1^2 - P_1^2)} \quad D_m = |Q_1 - P_1|$
Definition of the Euclidean and Manhattan distances.

The first step in computing the score of the DTW algorithm is to calculate the difference, or Cost matrix. The difference matrix is obtained by aligning one signal in the X-axis and the other in the Y-axis and writing the output value of the cost function on every matrix index C[i,j] corresponding to inputs Q[i], and P[j]. As previously mentioned the cost measure can be the absolute value of the difference (preferred), or the root of the squared difference. The msp430g2553, being of a RISC structure, takes multiple cycles to compute a more complex operation like multiplication and square roots, therefore using the absolute value of the difference is less computationally taxing on the system, and a better option for this project. The difference matrix for two simple signals in depicted in figure 40.

Figure 39: Difference (Local cost) matrix for two signals S_1= {1, 2, 3, 4, 5} S_2= {1, 3, 2, 5, 6}

Signal 1					
5	4	2	3	0	1
4	3	1	2	1	2
3	2	0	1	2	3
2	1	1	0	3	4
1	0	2	1	4	5
	1	3	2	5	6
			Signal 2		

After the difference matrix is found the next step is to find the alignment path, or warping path. The warping path runs through the "valley" of the local cost matrix, since these points represent very high similarity. At this point the user is at liberty to apply a set of conditions for the warping path for optimizing performance, and accuracy.

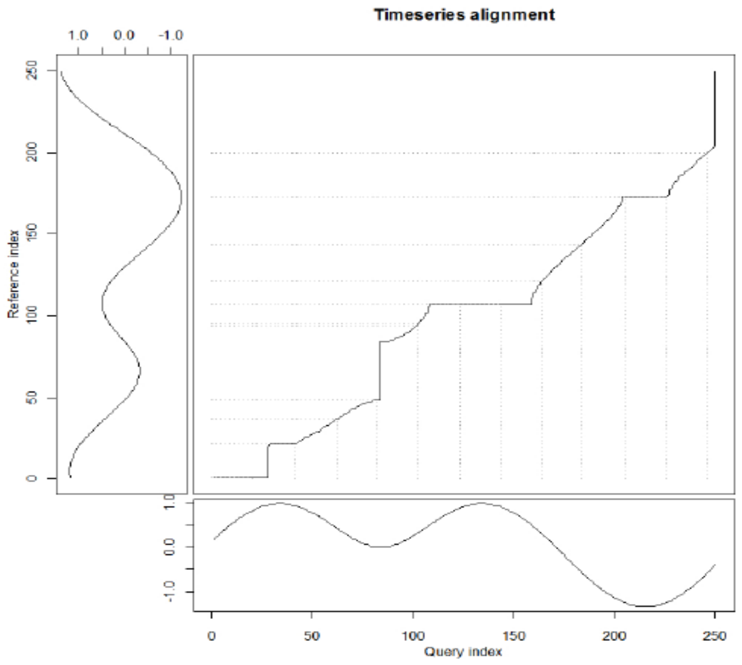
1. **Boundary condition**: p_1 = (1, 1) and p_k = (N, M). This condition ensures that the starting and ending points of the warping path match those of the first and last of the aligned sequences.
2. **Monotonicity condition:** $n_1 \leq n_2 \leq \ldots \leq n_k$ and $m_1 \leq m_2 \leq \ldots \leq m_k$. This condition preserves the time-ordering of the points.
3. **Step size condition:** This criteria limits the warping path from long jumps (shifts in time) while aligning sequences. There are several versions of the step size condition, but the basic step size condition is formulated **as $p_{l+1} - p_l$ in {(1, 1), (1, 0), (0, 1)}**.

The heart of the algorithm is finding an optimal warping path, as we will use this path as a measure for similarity. Testing all possible warping paths in order to find the optimal on would be computationally challenging due to the exponential relation between the lengths of the signals and the total number of warping paths. To overcome this challenge, the standard dynamic time warping algorithm employs dynamic programming to reduce the complexity to *O(MN*). To do this the algorithm builds the "accumulated cost matrix, or global cost matrix" defined as follows:

1. **First row:** $\boldsymbol{D(1, j) = \sum_{k=1}^{j} C(x_1, y_k)\, , j \in [1, M]}$.
2. **First column:** $\boldsymbol{D(i, 1) = \sum_{k=1}^{i} C(x_k, y_1)\, , ij \in [1, N]}$.
3. **All other elements:** $\boldsymbol{D(i, j) = \min\{D(i-1, j-1),\ D(i-1, j),\ D(i, j-1)\} + C(x_i, y_j), i \in [1, N], j \in [1, M]}$.

Like the local cost matrix, the time to build this accumulated cost matrix is of order NM. The warping path can be found by backtracking from point p_{end} = (M, N) to the p_{start} = (1, 1) following the greedy strategy, which will be discussed shortly.

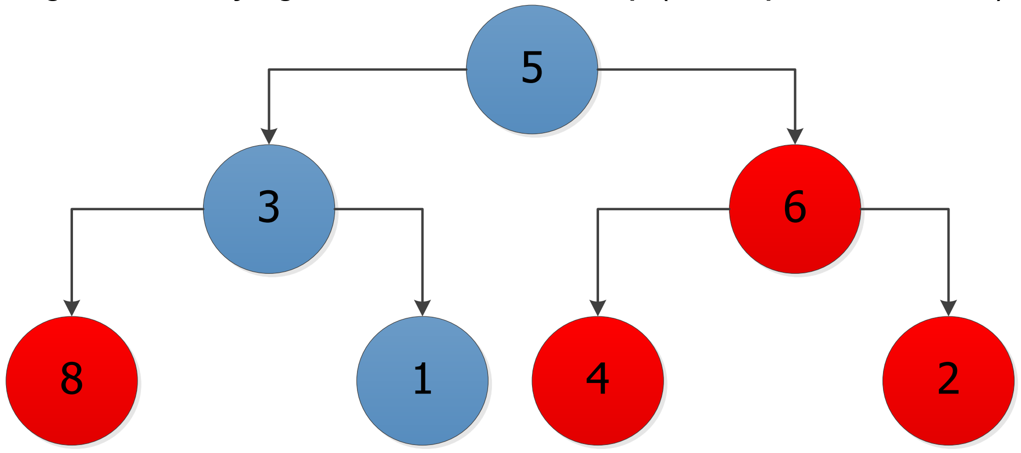
Figure 40: Optimal warping path of two time-dependent sequences.



3.15.2. The Greedy Algorithm

The act of finding the shortest warping path is an optimization problem. The "greedy" algorithm provides an intuitive way for finding the optimal path through the difference, or cost matrix. The algorithm is limited in the sense that is only takes into account the current phase, without regard for future consequences. The basic premise is that by choosing a local minimum at each step, the end result will be a global minimum. A simple example of the workings of this algorithm is counting money. In the scenario where one needs to count to 6.39, in US dollars, the algorithm will take the largest bill that does not overshoot the amount at each step in order to reach the value using the least number of bills. However, this approach would not work in a monetary system with coin values of 1, 7, and 10. In such a system, using the greedy algorithm the result would be 5 coins of value 1, and 1 of value 10. Clearly this is not the optimal solution as one could have chosen two coins of value 7 and 1 of value 1. In a similar fashion to the restrictions imposed on the warping path, several requirements can be established to improve the accuracy of the greedy algorithm. These additional criterions can be seen in the Dijkstra, Kruskal, and Prim algorithms. We expect that using the standard greedy algorithm starting that the last index of the accumulated matrix, and backtracking to the start of the matrix we can obtain an optimal warping path through the Cost matrix relating the two signals.

Figure 41: Greedy algorithm choices at each step. (Blue depicts choice made)



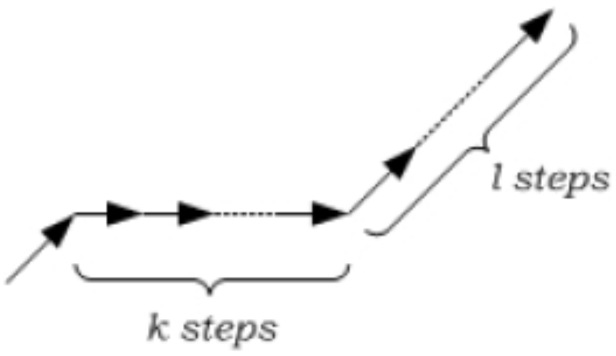
3.15.3. DTW Optimization

There are several studied ways to improve the accuracy and the computing time of the dynamic time warping algorithm. An obvious, but efficient way of reducing the order of the algorithm is to down-sample, or use linear approximations of the time signals. Because the order of the algorithm is N M, where N and M are the lengths of the individual sequences, a reduction by ½ of the number of points in each sequence would result in a 16x speed up. In the design however, the group fully controls the sampling rate of the gyroscope and accelerometer, so no pre-processing steps were taken with regard to the number of samples.

3.15.4. Step Function (Slope Constraints)

As previously stated when there are no differences between the signals being tested, the optimal warping path will run through the diagonal *i=j* of the Cost matrix. As the time-series start to deviate from one another the optimal warping path also diverges from the diagonal by matching similar time-axis fluctuations. While the DTW algorithm obtains the optimal alignment of the time-series, sometimes it might create unrealistic correspondences between features by aligning very short features of one series with very long ones of the other. In order to prevent this phenomenon the warping path is subject to constrains on each step. The constrains may vary, but generally after moving in the same direction for *k* consecutive points the path is forced to step *l* points in the diagonal directions.

Figure 42: Example of Slope Constraints



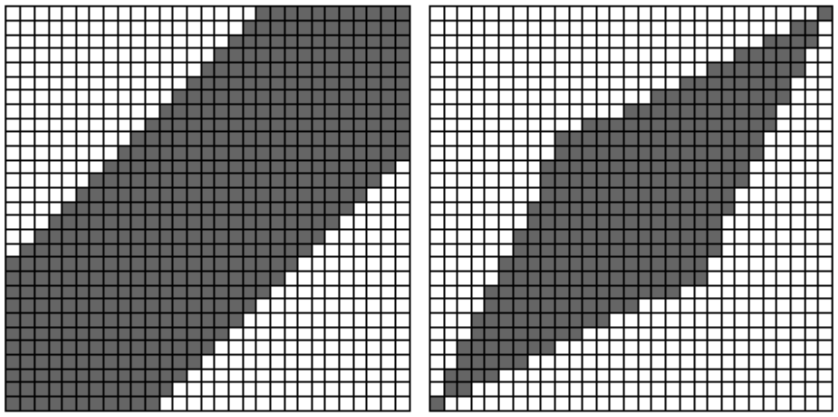
3.15.5. Weighting

Another proposed improvement to the classic dynamic time warping algorithm is the introduction of weighting. The measure of the distance between time series as the cost function, which basically is a summation of pairwise distance between corresponding points S[i], Q[j]. By adding weights to the cost of each distance based on the step direction we can penalize or favor certain types of point-to point correspondence. This method has been tested by using symmetric weighting for Japanese speech recognition with favorable results. I our design we don't expect to include this feature, unless recognition accuracy is not satisfactory upon testing.

3.15.6. Global path constraints

Two of the most widely used constraints are: the Sakoe-Chuba Band, and the Itakura Parallelogram. These global restrictions essentially reduce the area of the accumulated cost matrix through which the DTW algorithm can operate. The gray segments of figure 44 show the available steps for the warping path. This limiting approach introduces a speed up in the computing time of the DTW algorithm by a constant factor. The constraints work well in domains where the optimal path is expected to be close to a linear warp, and passes through the cost matrix diagonally in a relatively straight line. For cases unlike what previously mention, it is possible the algorithm will fail to obtain an optimal warp path if it is not bounded by the band/parallelogram limits.

Figure 43: Sakoe-Chuba Band (Left) and the Itakura Parallelogram (Right)



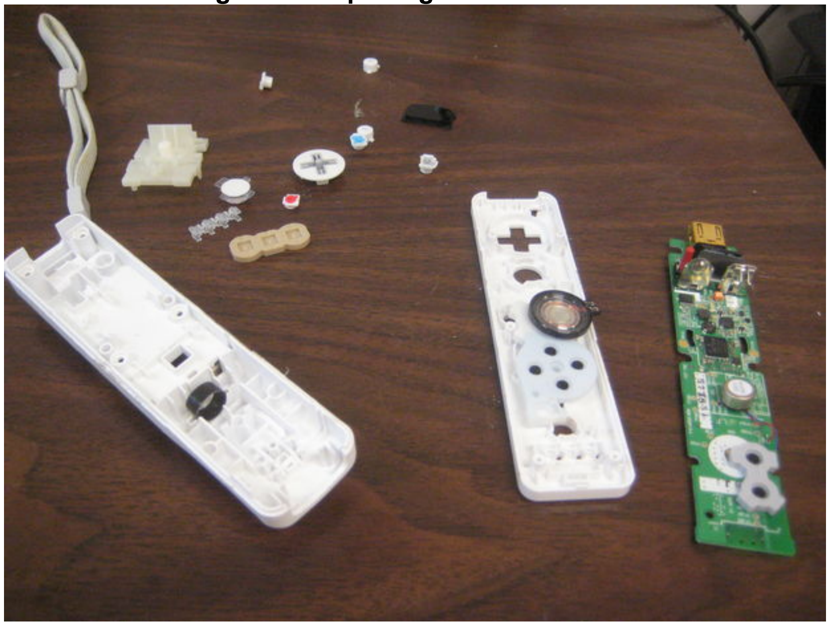
4.0 Project Hardware and Software Design

4.1. Camera

4.1.1. Extracting the Camera Module

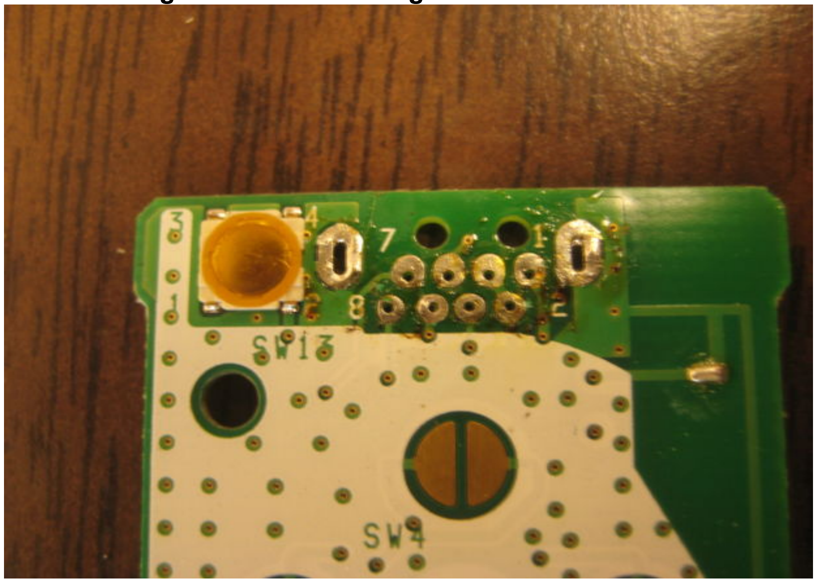
In this project the group decided to use the Pixart IR sensor found inside the Wii remote controllers. In order to obtain this module the group drilled the outter screws of the controller and removed the circuit board. The picture below show the wii remote opened with all its parts.

Figure 44: Opening the Wii Remote



Once the IR sensor was indentified the group desoldered it very carefully. Excessive amounts of heat could result into damaging the camera module. This module consisted of 7 pins and 2 latch pins. Below there is a picture of the Wii controller PCB and the apot where the camera was originally placed.

Figure 45: Desoldering the Camera Module



4.2. Microcontroller

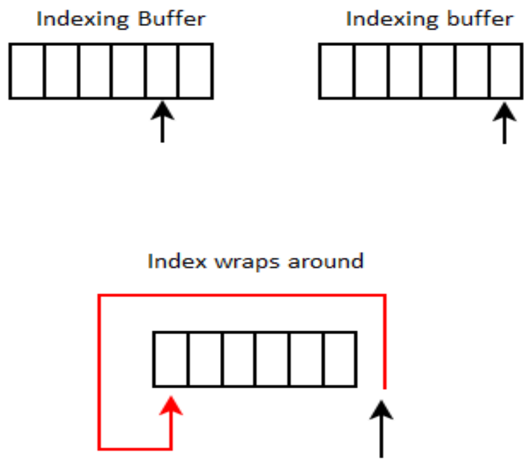
4.2.1. Program Overview

In order to meet the original specifications, the design makes use of the MPU-6050 as well as the RS232 TTL transceiver module. The StellarisLM4F120 communicates with the MPU-6050 using the I^2C bus, while the RS232 TTL transceiver module uses UART. Upon power up the Stellaris LM4F120 will have to initialize both buses and modules. The initialization procedure is as follows:

1. Set up clock system on the StellarisLM4F120 to source the USCIA and USCIB modules from the Sub main clock at 1 Mega Hertz.
2. Input correct the control register values in USCIA corresponding to 9600 baud and the appropriate modulation index based on the formula provided in the device data sheet. Likewise, input control register values in USCIB corresponding to I^2C communications, single master environment, and 7-bit addressing.
3. Initialize 3 circular buffers to store continuous gyroscope data.
4. Ask the user to input the “standing conditions” in order to compute the “trigger values”.

The circular buffers take advantage of the overflowing conditions for an unsigned data type. The range of an unsigned char type is 0 – 256, once the upper limit is reached should the user increase the value by one, it will result in the carry flag to be set and all bits of the char type will revert to 0 (the overflow and carry flags are ignored). This successfully creates a circular buffer with minimal maintenance of the index. Standing conditions refer to the user rest orientation, acceleration is assumed to be zero aside from that due to gravity. Triggers values are those that once encountered in a reading will trigger the recognition algorithm on the last X number of stored gyroscope values. Globally once the standing conditions are stored and the triggers calculated, the StellarisLM4F120 will go into Low power mode, starting back up only when data ready interrupts are sent by the MPU-6050. This cycle will continue indefinitely, eventually overwriting the oldest data in the buffer, until a trigger value (within a certain tolerance) is read over I^2C.

Figure 46: Circular buffer implementation

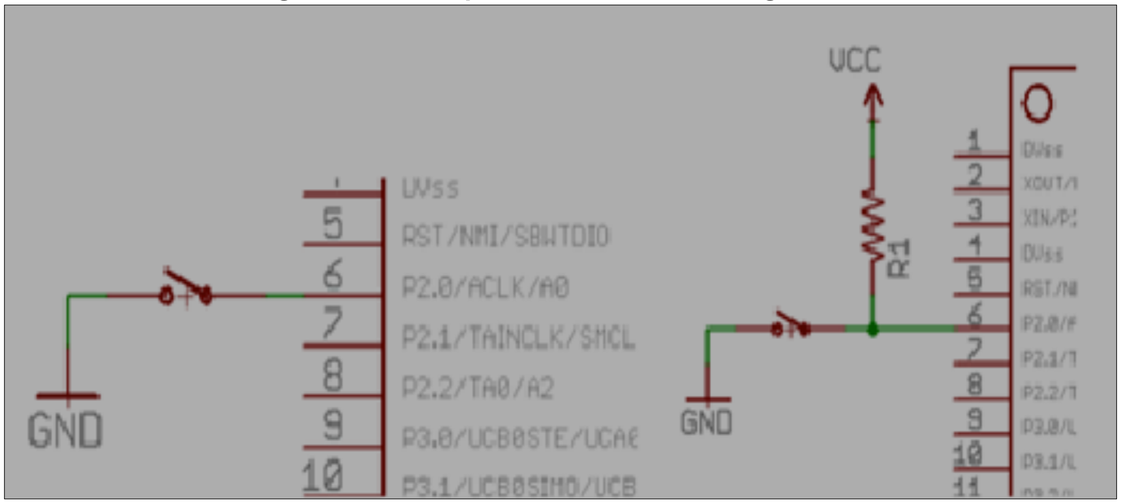


4.2.2. Power Supply Design

To begin the design portion for the power options first lets consider the chip configuration. The StellarisLM4F120 has two pins for power that correspond to the high and low states for the digital logic, which are the Vcc and Vss correspondingly. Typically the Vss is grounded and the Vcc is usually in the 1.5-3.6 V range, depending on the application the application that is being used. Note: on the value line devices, digital circuitries are labeled as DVCC and DVSS, with the D referring to the digital circuitry. Other devices have a separate AVCC and AVSS for analog signals and peripherals. For these devices, an A and D supply pins are tied to each other.

One thing that must be done in all designs regardless of the power source is every StellarisLM4F120 device has a RST/NMI pin, which allows the user to reset the chip externally by grounding that pin. For a button to behave as an input, the pin needs a definite default state which either it is grounded for active-high buttons or Vcc for active-low buttons. In order for a chip to be powered, the RST pin must be tied to Vcc. Note: that a direct wire is not a safe method of doing this; if a reset is triggered and the pin is grounded, it would short out the power supply. A pull-up resistor is required and was implemeted. Shown below in figure 48 is the circuit configuration for a pull-up resistor a resistor value of 47 kΩ is picked for the typical supply voltages one would use for any StellarisLM4F120 design:

Figure 47: Pull-Up Resistor Circuit Configuration



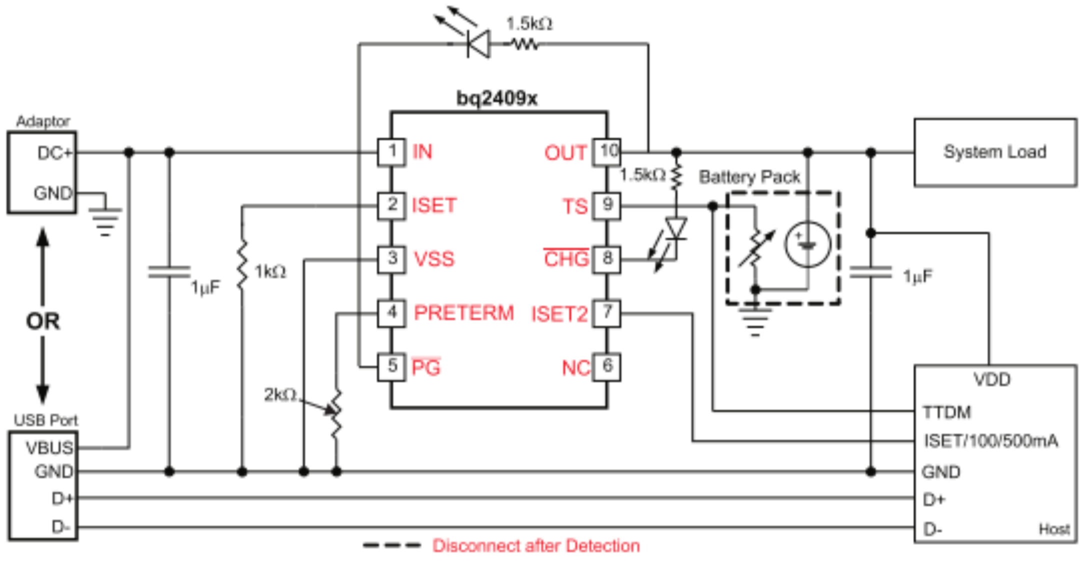
Another aspect that was considered for the schematic shown above is on the Vcc pin two capacitors can be added for filtering purposes. One of these is a 0.1 μF capacitor and the other a 10 μF capacitor. These capacitors are used to filter the power input, by which we mean keep any fluctuations in the power supply from affecting the value of Vcc. You will see below that many suggested designs for regulated power supplies use a 10 μF capacitor or similar on their output, and you may be able to get away with not using another on the StellarisLM4F120 if you use one of these power sources. In addition, connecting a small 1 nF capacitor to the RST to ground. Since the group did not see any fluctuations and sporadic resets then adding a capacitor was not necessary.

The 0.1 μF filtering capacitor will be included if we see changes in power. Digital circuits, especially when run at higher frequencies, can be vulnerable to noise from pins switching between high and low. This capacitor is used specifically to filter out that noise, and works best if it can be physically located close to the Vcc pin on the StellarisLM4F120. The information in this section was gathered from [2].

Since this device is a very low power consumer in comparison to other microcontroller. This would eliminate the need to run additional cables, or replace batteries whenever the circuit dies. All that would need to be changed is adding a circuit that ramps the input voltage supplied to the circuit, down to what the units use.

Given the fact that the accepted voltage for the StellarisLM4F120 is in the range from 1.8V to 3.6V. To keep the cost per unit down and still get good power output and battery life, lithium-ion meets the requirements of this project. With the voltage and cost restriction, we are limited to either using 3.6V or 1.2-volt cells connected in series batteries in series until the desired voltage was obtained. Figure 49 shows a rechargeable battery-powered system that uses TI's BQ24090 to save battery power.

Figure 48: Battery-Power System Block Diagram



4.2.2.1. LED Indicator

This reference design using two different color LEDs to indicate the stage of the charge process. If there is no battery insert, all LEDs are off. If the charging is in processing, the red LED will remain on until the battery is fully charged. A green LED will come on once an input voltage is plugged in which in this project it will be the mini USB on the circuit.

4.3. Microprocessor

4.3.1. Design Tools

4.3.1.1. Development Board

A development board is a printed circuit board with circuitry and hardware designed to facilitate experimentation with a certain microprocessor. It provides all the overhead circuitry such as power circuitry, programming interface, basic button inputs, output indicators (LEDs) and Input and output pins, etc.

Working with development boards is a great way for the designer to practice using the choice programming language as well as to test a certain project before jumping into his own board design. Using a development board helps the designer to determine if the design really works. It is mainly used for educational purposes. Moreover it is important that the designer determines the development board functions properly and that these functions will satisfy the project requirements.

Once the designer becomes comfortable programming the development board it will be up to the designer to develop his own printed circuit board or to simulate one from the development board used for practice. If a new board is recreated from the design of an existing development board a disclosure will be made to give credit to the manufacturer. In this project it is very likely for the designer to recreate a certain part of the development board used for practice. For the purposes of this project the designer chose the StellarisLM4F120 launchpad by Texas Instruments

4.3.1.2. Development Language

The development language used for the MSP430 and the StelarisLM4F120 was C-language. The group decided to use C because it is one of the most familiar languages as well as versatile for the project purposes.

4.3.2. MSP430 Design

4.3.2.1. Operation Procedure

At the start of the program the MSP430G2553 MCU initialized its Universal Serial Communication Module for I^2C communications on UCB0 and UART communications on UCA0. The MCU's Main clock was sourced from the internal Digitally Controlled Oscillator (DCO) at 16 MHz, The Sub-main clock (SMCLK, Also sourced from the DCO) passed through 1/4 divider resulting in a 4 MHz clock rate. UCB0, sourced from the SMLCK allows the usage of Low Power Mode 0 (CPUOFF) during interrupt driven transmission. Because UCA0 is sourced from the external 32 KHz crystal, during transmission to the Host computer, as well as idle delay resulting from the camera's maximum sample rate of 100 Hz, the MSP430 will remain in LPM3, effectively turning off the CPU as well as the DCO and consuming only 0.7 micro Amps. The Master (MSP430) will initiate a read operation on the I^2C Bus every 1/100 of a second, and proceed to decode the camera's internal registers. If the camera detected an object the MSP430 would send 4 bytes (2 bytes per coordinate axis at a 10 bit resolution) on the UART line corresponding to the X and Y coordinates of the object. Else, if a left or right click occurred the upper 6 bits of the first byte send would be used as command bits, signaling the host that a click has occurred or to wait for the second byte of coordinate information.

4.3.2.2. Camera PCB

After researching companies to provide the printed circuit boards the group chose Advanced Circuits. The decision was made on the price and numerous advantages provided by the company. Some of these advantages were gathered from [30] and are:

- Free PCB layout software, Eagle CAD
- Able to order multiple boards at a low price (the project requires 2)

- White Silk Screen
- Lead Time- 5 days
- 2-Layers
- 0.062" thickness
- 1 oz. cu. plate
- Up to 60 square in.
- Lead Free
- All holes plated

Moreover a figure of the final msp430 board design is provided below:

Figure 49: Camera Eagle CAD PCB

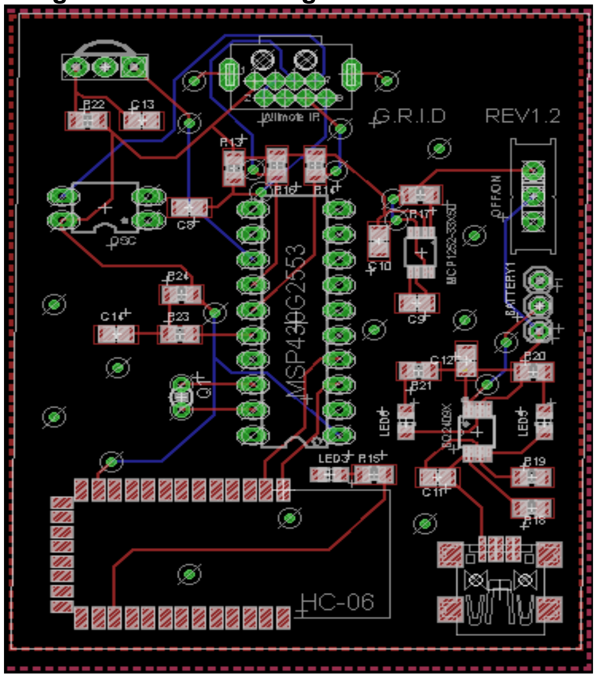
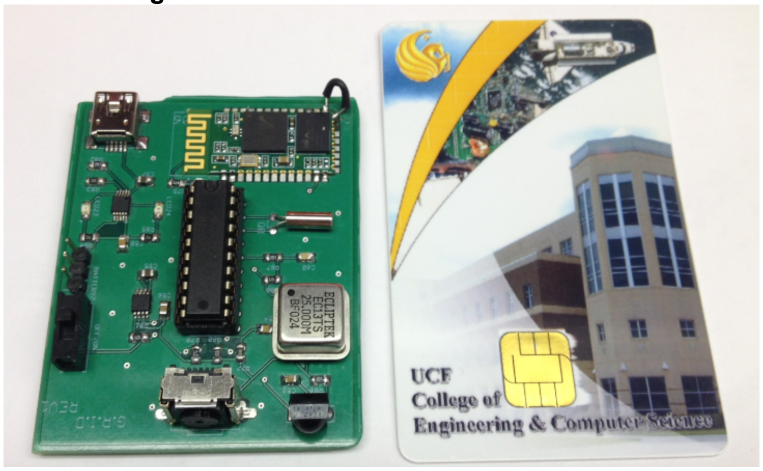


Figure 50: Camera Printed Circuit Board



4.3.3. Stellaris Design

4.3.3.1. Operation Procedure

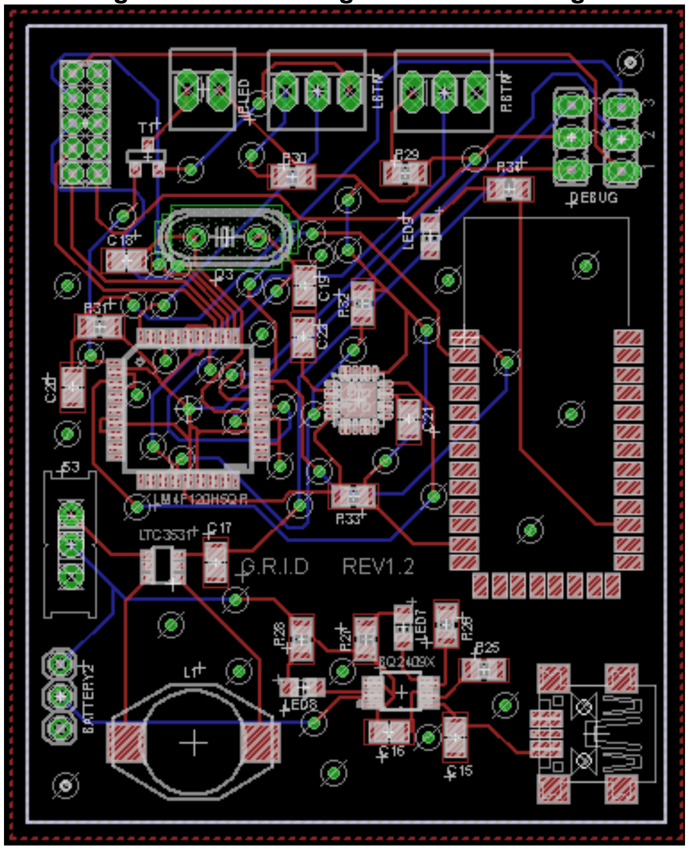
At the start of the program the Stellaris MCU would initialize its Clock system (80 MHz) as well as the various peripherals being used. Port B is used for UART1 and would transmit to the Bluetooth module. Port A will be used for I^2C communications to the mpu6050 with the interrupt pin connected to pin 7 on Port B with a falling edge trigger. TIMER0A is used to control the PWM output from pin 6 to the gate of the transistor at a 50% duty cycle with frequency of 1MHz. A second timer, TIMER1A, is used to control the duration of the 38 KHz pulses for the right and left click distinctions. The 38 KHz pulses are activated on the falling edge (Due to button press) of pin 5 and 3 on ports A and D respectively. The MPU is configured tosend interrupts when a full write cycle has been completed on its internal registers, as well as if high accelerometer date is detected on any axis. High Gs interrupts depend on the threshold force and the duration of the motion. For every sample, if it meets the threshold, a counter is increased; once the counter reached the duration value and interrupt is sent. The Stellaris will read accelerometer data from the mpu as dictated by the data ready interrupt and will store the values in a circular buffer of type int with an unsigned character type used as the index variable. Once the index reaches a value of 256, the index will overflow back to 0 on the next sample. Lastly the Stellaris will check the current sample against a “trigger value” which will signify a possible gesture has been performed. If the trigger flag is set, the stellaris will perform the DTW algorithm on the last 20 sampled values on the Z-axis and determine if it matches the template. Lastly, the MCU will transmit the corresponding command byte through the UART line to the host computer.

4.3.3.2. Glove PCB

The Stellaris PCB was ordered from the same place as the camera PCB which means that a lot of its features are very similar. This circuitry draws a bit more current but its 1 ounce copper traces are still good for its max current of 150mA.

Below there is a picture of the PCB that was designed by the group although it was not used in the final presentation.

Figure 51: Glove Eagle CAD PCB Design



Since the group had difficulties with placing the 64-pin Stellaris chip on the PCB due to some traces being wronged, they decided to use the Stellaris launchpad and a proto board for the final design. The proto board had all the components soldered in and a set of sockets that would hold the Stellaris launch pad on top. Below there is a picture of the launch pad and another one of the bottom view of the proto board.

Figure 52: Top View of Glove PCB

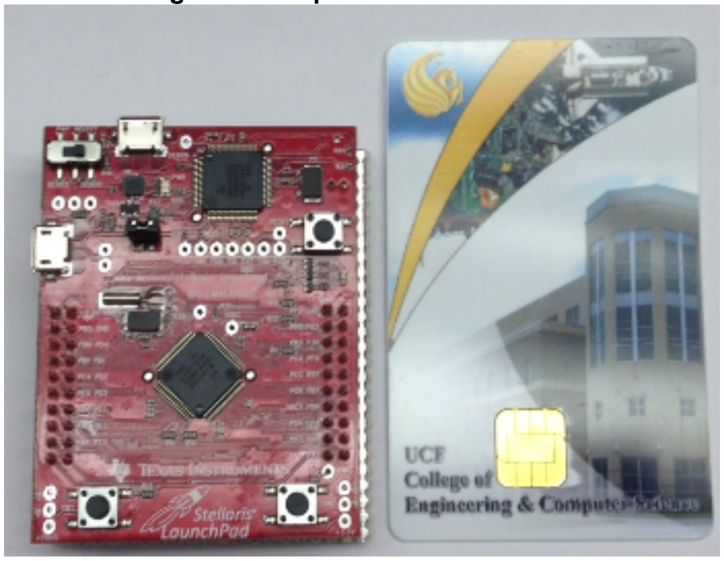
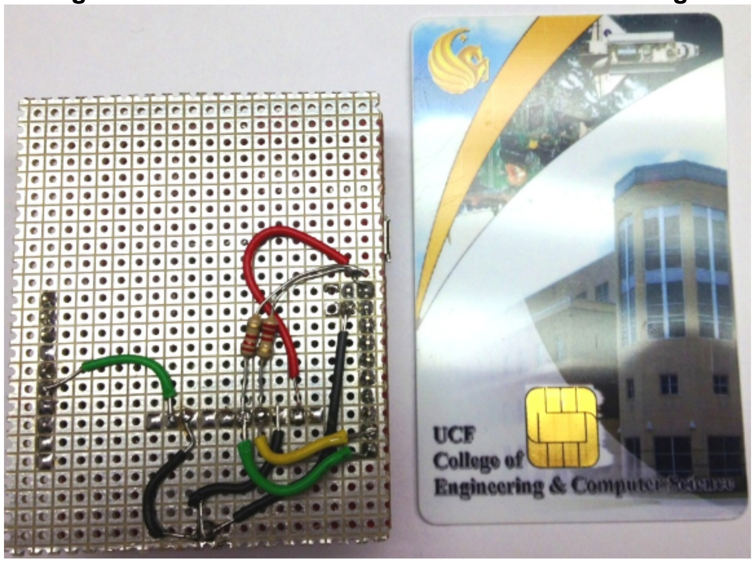


Figure 53:Bottom view of Glove Proto Board Design



4.5 Image Processing Algorithm

The Pixart IR sensor not only streams video but it also performs the image processing required for this project. This is done by capturing video capture and performing image processing. The image processing consists of algorithms such as the detection of the LEDS which were applied to the digitalized video signal before it is sent to the monitor.

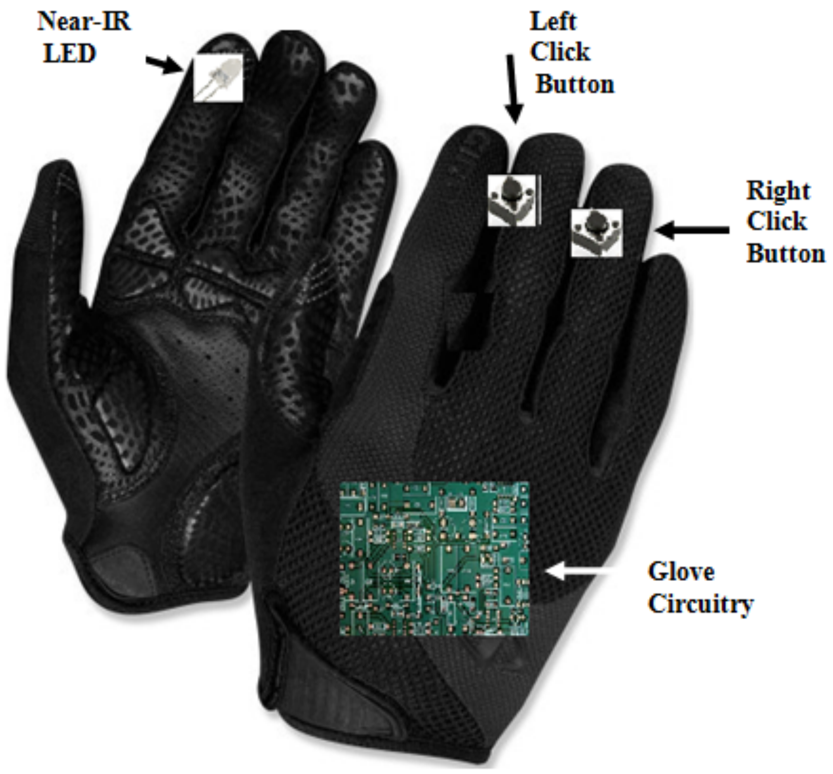
4.6 Aesthetics

4.6.1 Glove and Mount

The glove has a 2" X 2.5" circuitry on the back of the hand. As a result of the small size of this circuit it makes it simple to detach and can be placed on the hand of choice of the user to accommodate right and left handed users. The index finger has an LED attached on the palm side. The LED will be power via an extension cable running from the circuitry in the back of the hand for maximum comfort ability and flexibility. The wire extension will be sawn in so it is comfortable and permissive when the glove is on.

The push buttons used for clicking will be place in the middle finger. In the top of the middle finger will be the left click because that is a more natural movement to make and left clicks are used much more often. The right click button will be placed in the lower side of the middle finger because right clicks are less frequent than left clicks and that movement is comfortable for the user yet somewhat forced. The wire extension of these buttons will also be placed outside of the glove. Below there is a pictorial representation of the glove and its component locations

Figure 54: Master Hand



<u>Master Hand Features:</u>

- Gyroscope and Accelerometer: Arduino MPU 6050
- Microcontroller Launchpad: Stellaris LM4F120
- Near-IR LED (940nm & 60˚ viewing angle)

- Push buttons- clicking
- USB Interface

4.7 Software

4.7.1 Overview

GRID has several different sections when it comes to software. The most obvious is the computer GUI and integration driver, though the other software comes in on the StellarisLM4F120 microcontroller, Bluetooth module, and the communication protocols for everything to talk together. Each section required a different approach; first, the GUI and driver required a high-level language that can interact with the user and other components of GRID. The reason a high-level language is needed for this part as opposed to something like assembly, which could run a device driver just fine, was because there is specific input needed from the user to configure the device to the users desired arrangement.

There is one thing that was taken into consideration before the software is developed, that is whether to write the driver for use in the user space of the computer or the Kernel Space. In Kernel mode, the executing code has complete and unrestricted access to the underlying hardware. It can execute any CPU instruction and reference any memory address. Kernel mode is generally reserved for the lowest-level, most trusted functions of the operating system. Crashes in kernel mode are catastrophic; they will halt the entire PC. In User mode, the executing code has no ability to directly access hardware or reference memory. Code running in user mode must delegate to system APIs to access hardware or memory. Due to the protection afforded by this sort of isolation, crashes in user mode are always recoverable. Most of the code running on your computer will execute in user mode. Knowing that any problems with the code could cause a system crash if it is written for the kernel for the purpose of GRID the device driver was implemented for user space execution even though it has its limitations due to the fact of not having access to all the resources in the kernel.

With its object features, C++ is considered natural match for the semantics of Microsoft Windows Driver Model (WDM) and Windows Driver Foundation (WDF) drivers, and it is appealing for the added convenience and expressive power it provides. For the GUI and driver C++ is the best choice for a few reasons. First, C++ is a common language for writing a GUI and is equipped for the task, unlike C, which would require more work to produce the same results. Second, C++ also incorporates all the system libraries that will be required to write a device driver.

The other parts of GRID, such as the msp430, Stellaris and Bluetooth module, all will required a low-level language like assembly to tell it how to integrate with the rest of the system. This is simply because those sections did not need to interact directly with the user and there is less computation needed when it is in machine language. A few advantages machine language offers include: being faster and more efficient than high-level language, more compact and has direct access to resources needed.

4.7.2 Functional requirements

For the purpose of GRID there was essentially three sets of functional requirements, first for the GUI/Driver, second is the MSP430, and finally the FPGA. Each of these portions had their own set of requirements that were met in order for the project to be a success. In this case functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. The functional requirements are further supported by use case diagrams and backed by the non-functional requirements.

A use case defines a goal-oriented set of interactions between external actors and the system under consideration. Actors are parties outside the system that interact with the system. An actor may be a class of users, roles users can play, or other systems. There are distinctions between primary and secondary actors. A primary actor is one having a goal requiring the assistance of the system. A secondary actor is one from which the system needs assistance. A use case is initiated by a user with a particular goal in mind, and completes successfully when that goal is satisfied. It describes the sequence of interactions between actors and the system necessary to deliver the service that satisfies the goal. It also includes possible variants of this sequence, e.g., alternative sequences that may also satisfy the goal, as well as sequences that may lead to failure to complete the service because of exceptional behavior, error handling, etc.

The system was treated as a “black box”, and the interactions with system, including system responses, are as perceived from outside the system. Thus, use cases capture who (actor) does what (interaction) with the system, for what purpose (goal), without dealing with system internals. A completed set of use cases specifies all the different ways to use the system, and therefore defined all behavior required of the system, bounding the scope of the system. Generally, use case steps are written in an easy-to-understand structured narrative using the vocabulary of the domain. This was engaging for users who can easily follow and validate the use cases, and the accessibility encourages users to be actively involved in defining the requirements.

The following are the defined functional requirements for each section of the project:

GUI & Driver

- Intuitive design for user interaction with the device
- Took user input data and transmit back to Microcontroller
- Interpreted data from microcontroller via Bluetooth into defined computer actions

- Interpreted feed from FPGA into cursor movement
- Mapped the coordinates of the object to the appropriate locations on the display, solved for any difference in resolution
- Optimized code to use as little of the computer resources as possible

<u>MSP430</u>

- Tracked lighted objects
- Detected if an LED is present
- Calculated relative position (X and Y coordinates)
- Calculated size of blob
- Bluetooth Interface

<u>Microcontroller</u>

- Detected user motion
- Gathered data from gyroscope and accelerometer using I^2C
- Computed whether user completed a valid gesture
- Sent data flags to driver via Bluetooth using UART
- Optimized code with the use of interrupts in order to take advantage of low power modes in order to extend battery life.

Typically with the functional requirements there are a few use case diagrams that help to illustrate the use of the system and everything that goes on as a whole unit. The following figures are the use case diagrams for the GRID system. First figure is the use case diagram for the gloves and the host computer. The second is between the msp430 and host. The third is the interaction with the user and GUI.

Figure 55: Microcontroller Use Case diagram

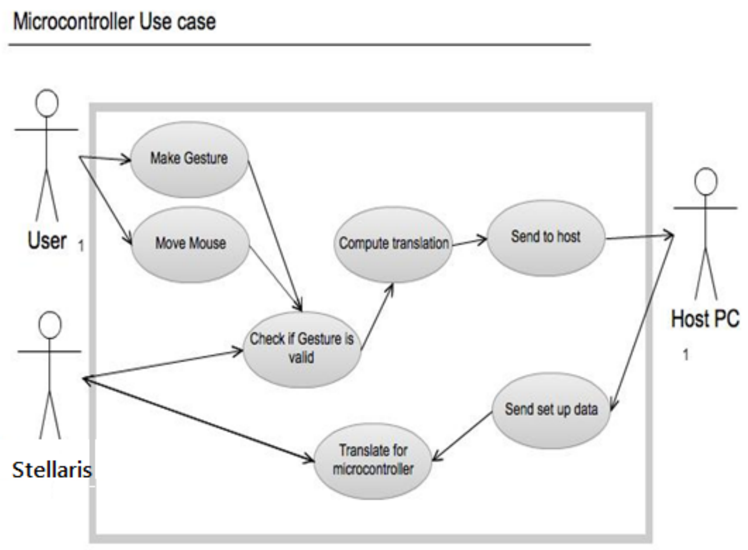


Figure 56: MSP430 Use Case diagram

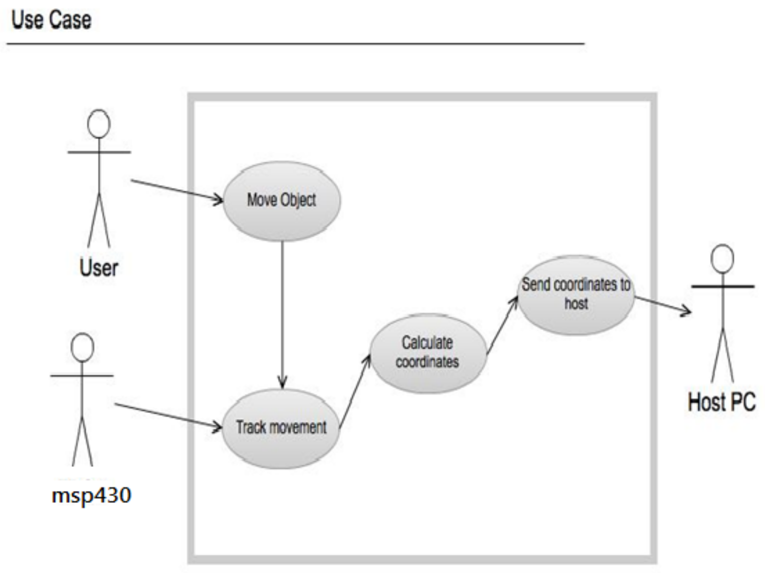
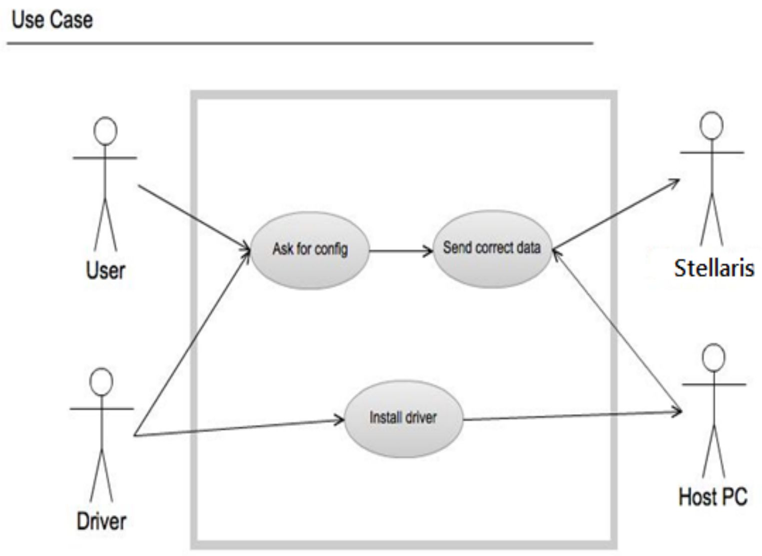


Figure 57: GUI/Driver Use Case diagram



4.7.3 Non-Functional Requirements

For the purpose of GRID there were essentially three sets of non-functional requirements, first for the GUI/Driver, second is the Stellaris, and finally the msp430. Each of these portions had their own set of requirements that were met in order for the project to be a success. In this case a non-functional requirement is a requirement that specifies criteria that wasused to judge the operation of a system, rather than specific behaviors. These are contrasted with functional requirements that define specific behavior or functions.

Non-functional requirements are more like quality attributes. These are things that don't actually do anything, but are important characteristics of the system. These attributes usually include things like performance, security, usability and compatibility. Below are listed the determined functional requirements for GRID:

GUI & Driver

- Coded in a way so as to be plug and play
- Intuitive design
- Quick in response
- Aesthetically pleasing

MSP430

- Fast in image processing
- Had efficient and well documented code
- Had accuracy within 16 pixel cluster
- Had operation range of 10-20 ft.
- Had real time image processing
- Had a battery life of up to 8 hours of continuous use
- Recharged in less than 2 hours

Stellaris LM4F120

- Communicated with other modules quickly and efficiently
- Weigh less than 3 pounds
- Responded to gestures less than 1 second
- Had operating range of 10-20 ft.

4.8 Device Driver

4.8.1 Overview

For the GRID driver it acts much like that of a typical windows mouse driver with some special changes for the implementation of the gestures. The following example gives a good idea on how the driver is to be implemented, it has been adapted from the original version which was created by Penny Orwick and Guy Smith in their book *Developing Drivers with the Windows Driver Foundation* [37]. The mouse driver locates itself in memory at boot time. It takes over both int 33h and int 10h. The driver is identified by an eight character sequence, in the case of the Microsoft mouse, it is the sequence MS$MOUSE. Before issuing any calls to the mouse driver, you should first establish its presence. There are two methods of accomplishing this. First, you can test to see if the driver was installed by checking for the device name, or use a mouse call to int 33h. Below is an overview of how initialization of the mouse is accomplished.

First, the program accessed the libraries that allow modification of resources located in the kernel. After access to the resources is accomplished then sets the AX register equal to zero and generates the proper interrupt for the initialization

of the driver, then it returns the value retrieved. In this case if the value is 1 the driver initialized properly, otherwise there was an error and it did not initialize.
This call also initialized the mouse system to the default parameters, if it is present.

Mouse Function Calls, this is a mouse call to INT 33h which will initialize the driver and set the communication protocol for the mouse. This version is for kernel based driver which will have direct access to all system resources.
First set AX equal to zero, which is the Mouse Installed Flag and RESET this Returns AX as a status byte, 0 = not present, -1 = present (and RESET). The default parameters for a RESET are, cursor position equals screen center, internal cursor flag = -1 (not displayed), graphics cursor = arrow (-1, -1), text cursor = inverting box, interrupt mask call = all 0 (no interrupts), light pen emulation = enabled, mouse/pixel ratio (H)= 8 to 8, mouse/pixel ratio (V)= 16 to 8, min/max cursor pos H = Depends upon card/mode and min/max cursor pos V = Depends upon card/mode. As AX increments the function call changes as follows.

Next if AX equals 1 Show Cursor. Then Increment the internal cursor flag, and if zero, displays the cursor on the screen. If the cursor flag is already zero, this function does nothing.
AX = 2 Hide Cursor decrements the internal cursor flag, and removes the cursor from the screen. AX = 3 Get Mouse Position and Button Status returns the state of the left and right mouse buttons, as well as the horizontal and vertical co-ordinates of the cursor. BX bit 0 is the left button (1=pressed, 0=released). BX bit 1 is the right button. CX is the cursor position, horizontal. DX is the cursor position, vertical.

AX = 4 Set Mouse Cursor Position. Upon entry, CX = new horizontal position and DX = new vertical position. AX = 5 Get Mouse Button Press Information. Upon entry, BX = which button to check for, (0=lft,1=rght). This returns the following information. AX = button status, bit 0 = left button, bit 1 = right button (1=pressed, 0=released). BX = count of button presses (0 to 32767, reset to 0 after this call). CX = cursor position, horizontal, at last press. DX = cursor position, vertical, at last press AX = 6 Get Button Release Information. Upon entry, BX = which button to check for, (0=lft,1=rght). This returns the following information:

AX = button status, bit 0 = left button
bit 1 = right button (1=pressed, 0=released)
BX = count of button releases (0 to 32767, reset to 0 after this call)
CX = cursor position, horizontal, at last release
DX = cursor position, vertical, at last release

AX = 7 Set Minimum and Maximum Horizontal Position. Upon entry, CX = minimum position. DX = maximum position.
AX = 8 Set Minimum and Maximum Vertical Position. Upon entry, CX = minimum position and DX = maximum position.

AX = 9 Set Graphics Cursor Block. Upon entry, BX = cursor hot spot (horizontal), CX = cursor hot spot (vertical) and DX = pointer to screen and cursor masks.
AX = 10 Set Text Cursor. Upon entry, BX = cursor select (0=software, 1=hardware), CX = screen mask or scan line start and DX = cursor mask or scan line end.
AX = 11 Read Mouse Motion Counters. This return the following values:

CX = horizontal count
DX = vertical count

AX = 12 Set User-Defined Subroutine Input Mask. Upon entry, CX = call mask, DX = address offset to subroutine and ES = address segment to subroutine.
Each bit of the call mask corresponds to:

0 = Cursor position change
1 = Left button pressed
2 = Left button released
3 = Right button pressed
4 = Right button released
5-15 = Typically Not used

To enable an interrupt, set the corresponding bit to a 1. When the event occurs, the mouse driver will call your subroutine.
AX = 13 Light Pen Emulation Mode ON. AX = 14 Light Pen Emulation Mode OFF
AX = 15 Set Mickey/Pixel Ratio. Upon entry, CX = horizontal ratio and DX = vertical ratio. The ratios specify the number of mickeys per 8 pixels. The values must be within the range 1 to 32767. The default horizontal ratio is 8:8, whilst the default ratio for the vertical is 16:8.
AX = 16 Conditional OFF. Upon entry, CX = upper x screen co-ordinate, DX = upper y screen co-ordinate, SI = lower x screen co-ordinate and DI = lower y screen co-ordinate. This function defines a region on the screen for updating. If the mouse moves to the defined region, it will be hidden while the region is updated. After calling this function, you must call function 1 again to show the cursor.

AX = 19 Set Double Speed Threshold. Upon entry, DX = threshold speed in mickeys/second. This function can be used to double the cursors motion on the screen. The default value is 64 mickeys/second.
After the driver is set and recognized by the system it can then be implemented. The following is an illustration of how the mouse can then be interfaced with the computer using C++, the following is just pseudo code.

```
Include the necessary system libraries to access the internal resources
    Set up a static 2D array of type integer
        Fill the array with the interrupt locations for all ports that are to be
        listened to.

    First function is void set up function
        Set default values in registers AH and AL
        Generate interrupt
    Check that mouse driver exists
```

```
        Set AX register value
        Generate interrupt and return register value
        BX will contain the number of buttons present on mouse
    Display cursor
        Set AX register to display mode and generate interrupt
    Define shape of the cursor
        Create buffer
        Set AX, BX, CX, DX and ES registers
        Generate interrupt
    In the main
        Call Check mouse, if present
            Print mouse present
            Exit
        Call set up function
        Call shape of mouse
        Show the mouse
```

These methods were implemented in GRID to help initialize the driver and set up the mouse functions. Other parts were custom made for the device on how it communicated to the Stellaris and msp430, as well as how it is to respond to the gestures that are to be implemented.

4.9 Gesture Library

4.9.1 Overview

The design of the gestures is to be intuitive and natural to make it easier for the user to learn how to use the glove and also to make it an enjoyable experience. The location of storage for the library was split by the nature of the intended gestures. What is meant by that statement is that all gestures that are detectable by the microcontroller via the instruments in direct connection will be stored in the memory that is directly accessible by the microcontroller, the gestures that are detected by the host computer will be stored on the host computer where the driver can do the necessary actions to carry out the desired gesture.

Below there is a list of the gestures that were implemented in the library. Gestures used in this project will be mostly application specific, though there are also general gestures to accommodate normal computer functions. This list is the final gesture implementation list:

- **Clicking**- to click the user has to press the buttons on the middle finger with the thumb on the main hand in and then release. The two buttons in the middle finger are for right or left click.

- **Zooming in and out**- the user has to move the hands closer to the monitor for zooming in or further from the monitor for zooming out. This gesture is dependent on the image processing to detect the position and direction of the motion. This gesture is also linked with the accelerometer data to determine if it passes the threshold for this gesture.

- **Rotation**–Rotation is the same motion as zooming, the difference is instead of moving the hands closer or further from the monitor the user simply will keep the hands straight and rotate their hands position about the centroid axis.

- **Swiping in any direction**-This is one of the application specific gestures, when using an application that has multiple pages the user waves the main hand past the boundaries, which were pre-set in the direction they wish to swipe. For example to move to the next page the user would wave their main hand to the left past the leftmost boundary.

- **Refresh**–The refresh gesture is a very unique gesture, which is one of the defining characteristics of our project. To execute this gesture the user will take their hand in the rest position and rotate their palm in a clockwise motion 180^o then return it to the rest position. This gesture makes good use of the gyroscope and accelerometer data to determine if the gesture was performed accurately.

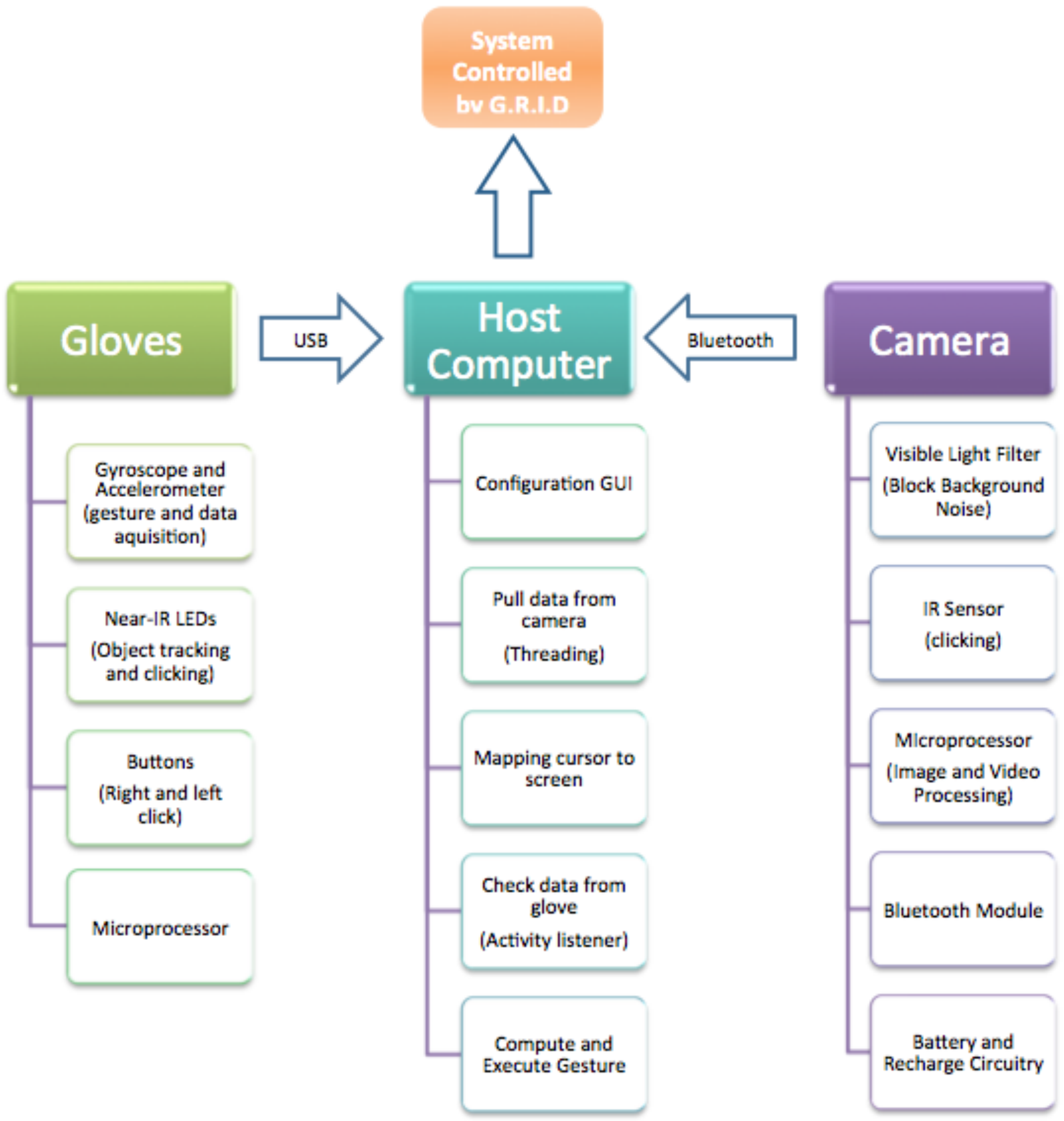
- **Multimedia Gestures**–These are all application specific gestures that can only be used within a media application. These actions will need to be distinct so they will be easily determined by the video image processing,

    - **Play**–To execute the play motion the user will take their hand at rest position and move it in a dropping motion. The data is then calculated by the gyroscope and accelerometer data to determine if it was indeed a proper gesture.
    - **Fast-forward**–To execute the fast-forward gesture the user will take their hand in the rest position and make a swiping motion to the right. Data is collected from the gyroscope and accelerometer to determine if it was a proper gesture.
    - **Rewind**–To execute the Rewind gesture the user will take their hand in the rest position and make a swiping motion to the left. Data is collected from the gyroscope and accelerometer to determine if it was a proper gesture.
    - **Pause**–To execute the pause gesture the user will take their hand at rest position and move it in a dropping motion. This gesture is the same as play; the reason for this is because in modern applications the command for play/pause is the same so we are keeping with modern convention. The data is collected from the gyroscope and accelerometer to determine if the motion preformed was valid.

5.0 Design Summary

GRID, gesture recognition interface device, is an interactive, as well as intuitive, way to control various computer functions by taking advantage of detectable and natural human mannerisms. In order to accomplish proper response to the various gestures were accounted for in the gesture library. GRID makes use of several sensors to gather diversified data about the current state and actions of the user. The sensors that make this possible are: a camera (tuned for the near-Infrared spectrum), an accelerometer (for swipe motion), and a gyroscope (for general orientation information) with room for upgradability by the addition of a digital magnetometer (providing additional user sensor data). Below figure 59 shows a flow chart of the GRID overall system:

Figure 58: GRID System Flow Chart


System Controlled bv G.R.I.D
Gloves
USB
Host Computer
Bluetooth
Camera
Gyroscope and Accelerometer (gesture and data aquisition)
Near-IR LEDs (Object tracking and clicking)
Buttons (Right and left click)
Microprocessor
Configuration GUI
Pull data from camera (Threading)
Mapping cursor to screen
Check data from glove (Activity listener)
Compute and Execute Gesture
Visible Light Filter (Block Background Noise)
IR Sensor (clicking)
Microprocessor (Image and Video Processing)
Bluetooth Module
Battery and Recharge Circuitry

In order to understand GRID's structure, the project was divided into smaller subsystems as seen in the flow chart above. Each of these subsystems has a very unique function that is critical to the overall functionality of the project. The subsystems that complement GRID are:

1. **Gloves Subsystem**

It consists of a regular set of gloves with a lightweight circuit mounted on it. The main purpose of the glove is to power the LEDs that the camera will be tracking for cursor location purposes, send fast IR light pulses to the IR receiver in the camera circuit for clicking purposes and gather hand movement information from the accelerometer via Bluetooth to the host computer which will be interpreted as gestures. In the glove hardware detail section below there will be more specifics about the individual components of the glove.

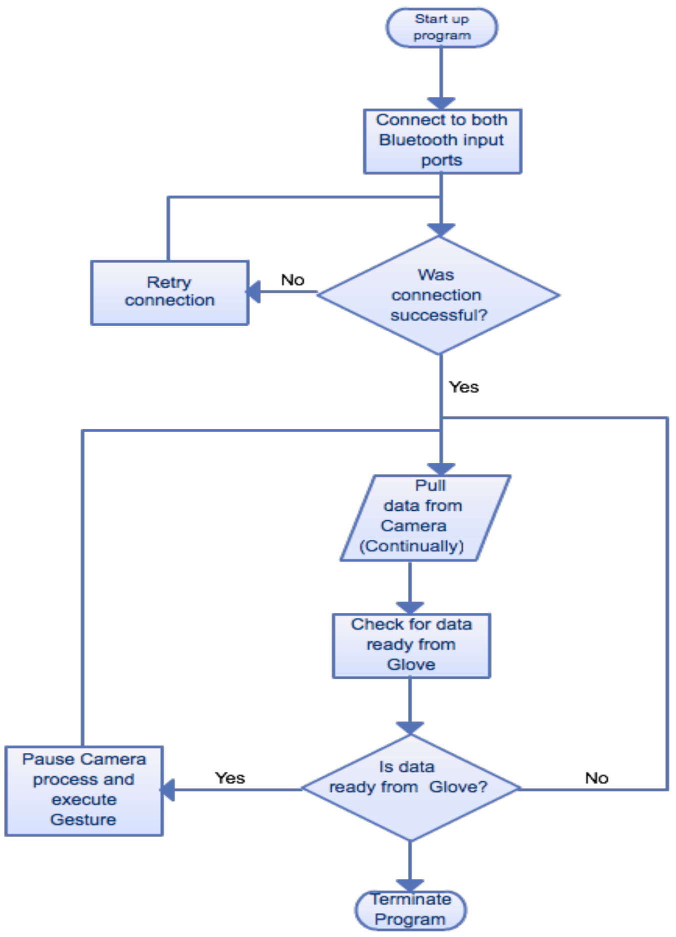
2. **Camera Subsystem**

The camera is the part of the project that tracks the position of the user's hands. The camera will be placed at an appropriate position that outlines the user's workspace. A glove will have a near IR-LED on the index fingers that will be tracked by the camera. The camera has an integrated processor that outputs the X and Y coordinate positions as well as the size of up to four bright IR points when interfaced with the microcontroller. This will allow the user to control the cursor of the computer, like a mouse. For further hardware details on the camera read the section called Camera Hardware Details, which explains the individual components of the camera.

3. **Driver**

The driver is basically the part of the project that will tie all components together by receiving information from the glove and the camera simultaneously via Bluetooth. When the driver receives user input data from the gloves it will interpret gestures being performed, and interpret data from microcontroller into defined computer actions. If the driver is receiving feed from the camera it will interpret it as a cursor movement, which involves mapping the coordinates of the object to the appropriate locations on the display, and solve for any difference in resolution. Clicking commands will also be received from the camera circuit board. Below there is a figure of the design flow of the driver.

Figure 59: Driver Design Flow



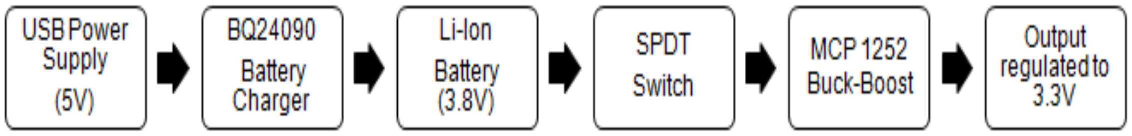
Start up program
Connect to both Bluetooth input ports
Was connection successful?
No
Retry connection
Yes
Pull data from Camera (Continually)
Check for data ready from Glove
Is data ready from Glove?
Yes
Pause Camera process and execute Gesture
No
Terminate Program

4. Power Supply

The power provided to the glove will come from a USB cable connected to the host computer. Since it is connected to the Stellaris launchpad the group[did not have to add any regulators to this portion of the circuit.

In the other hand the power supply for the camera was slightly more complicated. It consists of a battery charger and a buck-boost converter. The 5V input to the battery charger comes from a USB port, which is connected to the host computer when necessary and will only serve for this purpose, not for data transfer. There are two LEDs in this circuit, one will indicate when the battery is charging and when charging is complete.

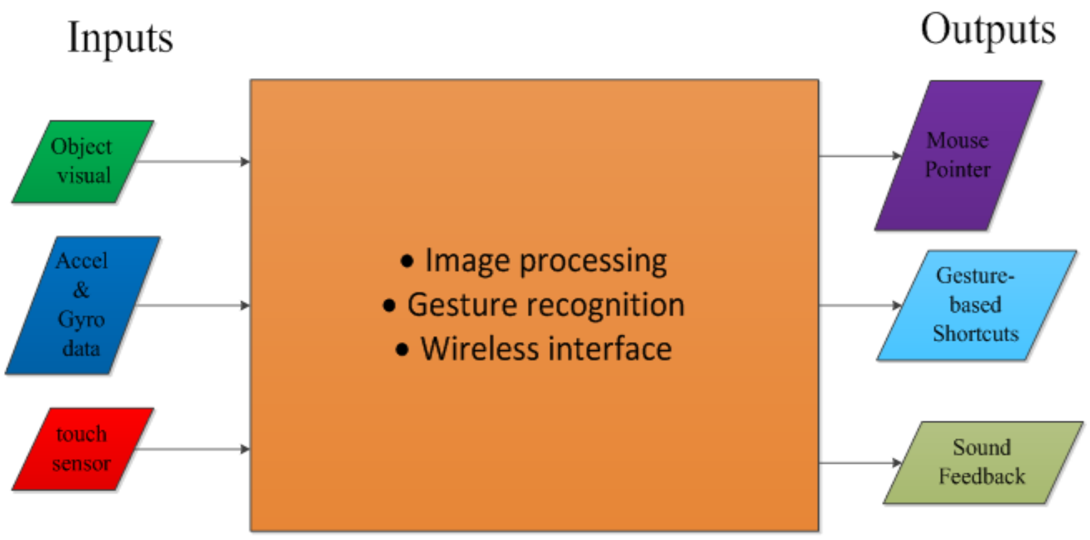
Figure 60: Power Supply Flowchart



5. System Interface

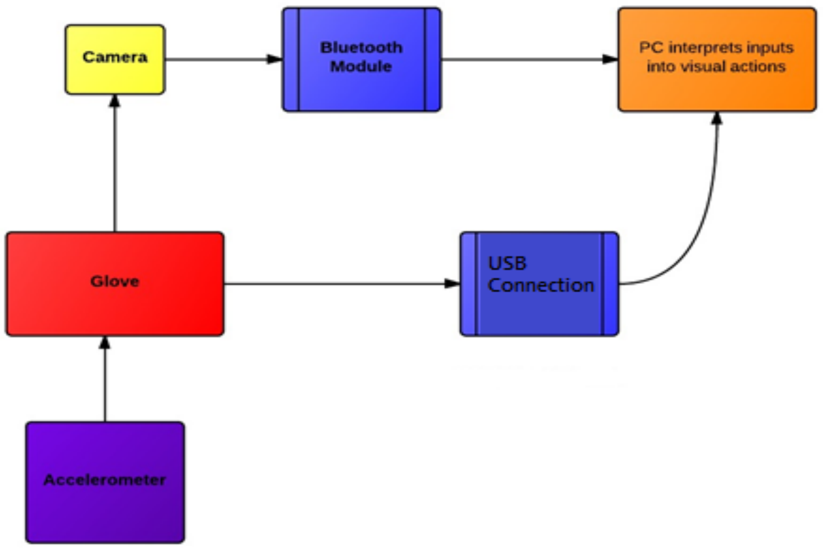
In GRID the systems will interface in different ways within each other. The diagram below shows the different inputs and outputs included in GRID:

Figure 61: System Interface



In the flowchart in figure 63 there is all the subsystems and how they communicate with each other. Some are wired connections and some are wireless. In the gloves there is a physical connections between the gyroscope and accelerometer, and microcontroller. The glove will communicate via USB to the host computer.

Figure 62: Interfacing Subsystems



In the other side of the computer we have the wired connections which will use USB, this is providing power as well as data transfer between devices. The msp430 to host computer connections will be all via Bluetooth. It is critical that all these connections are reliable as each subsystem is providing important information for the calculations of the cursor tracking and gestures.

6.0 Project Prototype Testing

6.1. Hardware Test Environment

6.1.1. Temperature

The majority of the hardware testing was performed indoors at "room temperature" (25˚C) unless specified otherwise. These are usual operating conditions of the system.

6.1.2. ESD Safety

One of the critical safety procedures that the group followed was to prevent ESD on the electronic components involved in this project. Electrostatic discharge is the sudden flow of electricity between two objects caused by contact, an electrical short, or dielectric breakdown. ESD can be caused by a buildup of static electricity and includes electric sparks, both, the ones an individual can and cannot see or hear. Even the unseen and unheard discharges can still be large enough to cause damage to sensitive electronic devices. For this reason the group will take the following measures:

- **Wear an ESD wrist strap-** This is one of the easiest and best ways to reduce ESD.
- **Grounding Mat-** This is the second best way to reduce ESD and certainly more comfortable than wearing a wrist strap
- **Zero Potential -** make sure the tester and the component are at zero potential by continuously touching an un-painted metal surface of the chassis or the computer power supply case.
- **Avoid Sitting Down -** It is also very important that the members were standing at all times when working with electronic devices. Setting on a chair can generate more electrostatic charge.
- **Unplug Electrical Cords -** The group made sure all power cords if any are removed.
- **Clothes -** Make sure not to wear any clothing that conducts a lot of electrical charge, such as wool.
- **Weather -** Lightning storms can increase the ESD risk; unless absolutely necessary. The group decided not to work on a electronic equipment during an electrical storm.
- **Accessories** - Another good practice that the group performed was to remove all jewelry before working with electronic equipment.

6.1.3. Lighting

Light settings are critical in some parts of the testing of this project. The light settings were normal unless the camera and filters were tested. When the camera and filters were tested the light settings of the room were low to moderate clarity to make it easier for the tester to see the glow in the IR card.

6.2. Hardware Specific Testing

6.2.1. Camera

The camera was interfaced with the msp430 in the PCB. This USB port not only will transfer data but it will also provide power to the camera. Below are two methods to make sure the camera is working and to make sure the camera is interfacing successfully with the microcontroller.

Test with the Controller- To make sure that the camera is not faulty the group tested it with the Wii system. Once the group confirmed that the Wii remote was working properly they came to the conclusion that the camera module was a good one.

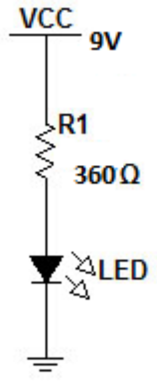
Test with the microcontroller-
To test if the camera is working properly, a small code was created in C and loaded into the msp430 to be able to stream from the camera. This was done with the development board. Once the physical connections were set, the group placed a moving object (near-IR LED) in front of the camera to make sure that the camera was interfacing properly with the msp430 and that the msp430 was properly interfacing with the host computer.

6.2.2. LEDs

A batch of near-infrared LEDS (940nm) were tested in the same circuit. Although the project requires only 4 LEDs the group will order extra ones in case of any malfunctions.

LED Testing Circuit: The set up was a simple series circuit to power them on and make sure they were working properly. The LED circuit was like the one shown in the figure below:

Figure 63: LED Test Circuit



To set up this circuit we needed to know the LED voltage drop and current drawn which is usually found in the LED data sheet. For this project we used an LED that will use 2V and the current will be 20mA. By using equations 1, 2 and 3 we can get the values needed to complete the LED test circuit. The source voltage was chosen to be 9V since a standard 9V battery will be used. Once we calculated the voltage drop across the resistor from equation 1 we plugged in all the known values into the Ohm's law, equation 3 to calculate the value of the resistor needed.

Equation 1: $V_{Resistor} = V_{source} - V_{LED}$= 9V-2V=$\underline{7V}$

Equation 2: $I_{LED} = I_{Resistor}$ (in series)=$\underline{20mA}$

Equation 3: $R = \frac{V_{resistor}}{I_{resistor}} = \frac{7V}{20mA}$ =$\underline{350\Omega}$

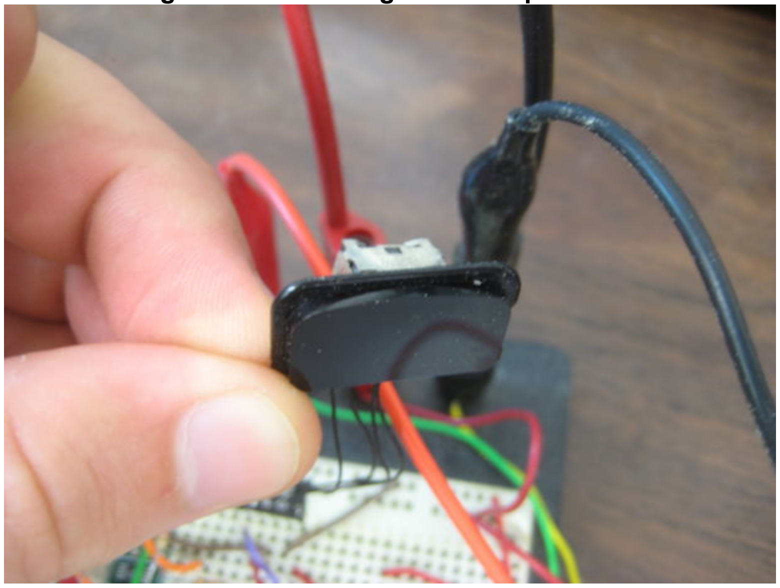
As seen above a 350Ω resistor is needed but we used the next available resistor value which is 360Ω.

Working LEDs: Once the circuit is set up we used an IR card to make sure the LEDs are working. This card will be placed close to the LED once it is placed in the circuit. If the IR card glows, the LED was working if the IR card does not react, most likely the LED was faulty. LEDs that are working properly will be set aside and the faulty ones will be discarded.

6.2.3. Visible Light Filter

Point the camera module at the lit up LEDs: The camera module was connected to a regular computer running Windows OS. The filters was placed in front of the module to block the visible light and attenuate the near-IR light. Once the camera was properly set up the group will point at the lit up LEDs in the circuit explained in section 6.2.2 and make sure that the camera was outputting a location for the LED and a blob size. Beloiw there is a picture of teh actual filter used.

Figure 64: Visible light filter implementation



6.2.4. Microcontroller

During the initialization procedure, several measures were taken to check for communications errors and correct them early on. For external visualization of errors, should a problem be detected during initialization, an LED would flash green when the tests are passed and red otherwise. During the clock system set up, the code references macros define by Texas Instruments which point to memory addresses where bit definitions for the various configurations were stored, should any of this definitions be erased, overwritten, or corrupted, the CPU would be “trapped” in a while loop and the LED will flash red.

The RS232 TTL Bluetooth module provided two modes of operation, one for transmission, and one for configuration of the module. After initialization of the Universal Serial Communication Interface module, the msp430g2553 would send a series of command terminated by “\r\n” to the RS232 TTL module and listen in for the expected response, indicating proper function. After ensuring functionality the msp430g2553 would proceed to send the signal to change the mode of operation to transmission and send the “UART read” string to be picked up by the host computer. This command was sent repeatedly until the host PC responded with an “OK”.

The MPU-6050 provided a self-test activated through code that permits the user to test the mechanical and electrical portions of the gyroscope and accelerometer independently. When the self-test was activated, the on-board electronics would actuate the appropriate sensor. The actuation will move the sensor’s proof masses over a distance equivalent to a pre-defined Coriolis force. The displacement results in a change in sensor output, which was observed the functionality of the device. The self-test response (STR) is defined as follows:

$$SelfTest\ Response = Gyroscope\ Output\ with\ SelfTest\ Enabled - Gyroscope\ Output\ with\ selfTest\ Disabled$$

This self-test response was used to determine whether the part has passed or failed the self-test by finding the change from factory trim on the self test response as follow:

$$Chage\ from\ Factory\ Trim\ of\ SelfTest\ Response(\%) = \frac{(STR - FT)}{FT}$$

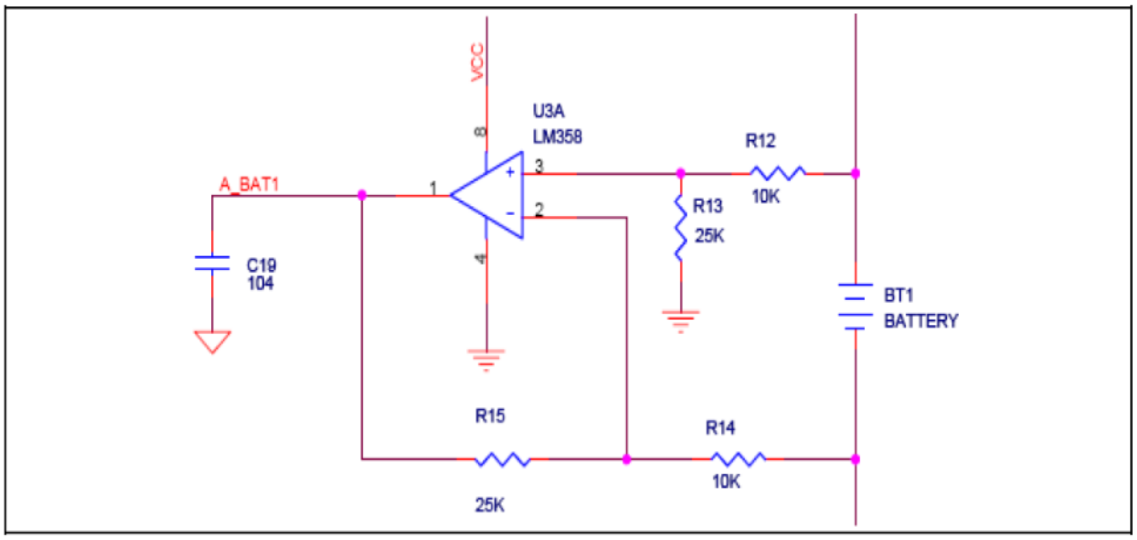
Where, FT was the factory trim value available via MotionApps software.

6.2.5. Power Circuit

6.2.5.1. Battery Voltage

To measure the charging voltage an operational amplifier was used to measure the voltage difference between the positive and negative pole of the battery. A difference operational amplifier was used to measure the battery's voltage. Figure 66 shown below shows a voltage op-amp circuit, which was implemented for the design of this project:

Figure 65: Voltage Op-Amp Circuit



The output voltage from the op-amp to microcontroller ($V_{Battery}$) is given by:

$$\boldsymbol{V_{Battery}} = \frac{\boldsymbol{R_{13}}}{\boldsymbol{R_{12}}}\boldsymbol{V_{+}} + \boldsymbol{V_{+}}$$

Where, $V_{Battery}$ is the output voltage from op-amp to microcontroller.

6.2.5.2. Temperature

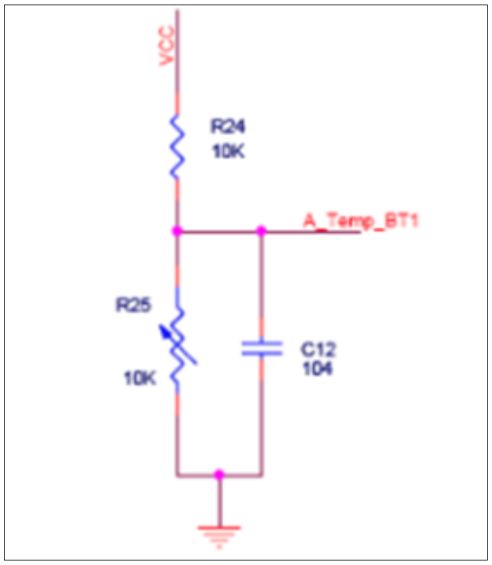
Temperature is an important parameter to test is the temperature of the lithium-ion batteries. Charging was terminated if the temperature rises above the operating temperature limit of the lithium-ion batteries. The group also used as backup method. Temperature was measured by a negative temperature coefficient resistor. The negative temperature coefficient is powered by the Vdd for the microcontroller and is part of the voltage divider. A detail circuit is shown is figure 67. The information in this section was gathered from [10].

The temperature is measured:

Vtemp = VDD × R25 /(R24 + R25)

Depending on the temperature (Vtemp) the resistor value would change. Therefore, Vtemp is changed accordingly, so, it can detect the temprature by check the voltage value of Vtemp by the analog to digital converter. Note: the temperature and resistor value do not have a linear relationship, which makes it difficult to calculate the temperature from the A/D converter. The information gathered in this section was gathered from [9].

Figure 66: Temperature Measurement Circuit



6.2.6. Glove and Mount

<u>**Heat Test**</u>- It was always a possibility that electric circuits dissipated heat. To make sure that the gloves and mount used on this project were the appropriate ones a heat test will be performed. The glove and mount, both items were placed inside an temperature chamber at 70˚C for one hour. The amount of heat tested was determined by a greater amount than the one that the electric circuitry of the gloves could dissipate. The gloves were carefully placed on a metal tray to avoid any possible accidents. Once ass determined that the gloves and mount can resist a small amount of heat it was fair to say that they could be used for this project.

<u>Weight Test</u>- In the project specifications the goal is for an individual glove is not to weight more than 3 pounds. A 4 pound weight system was be placed in each glove and the tester would make sudden movements with gloves for 5 minutes to make sure that each glove can support the weight of the circuitry which wless than the weight system used for testing.

6.3. Software Testing Environment

A parameter considered when testing the image recognition algorithm was the system environment. The purpose of the system environment was to work in an unstructured environment. Meaning an unstructured environment is one that has no artificial blue or green screen. This provides greater system flexibility and convenience but can make reliable segmentation extra difficult. As this environment required the need to distinguish the objects of interest from any other objects that may be present within the frame. This restraint may be overcome by limiting the target objects to saturated and distinctive colors to allow them to be distinguished from the unstructured background. Augmenting the unstructured environment with 10-structured color in this way is a compromise that enables a much simpler segmentation algorithm to be used.
Apart from the method to maintain the color distribution was to keep the background environment a constant. This method, only sees the target is in motion and the system is able to track its motion in a 2- D frame. For the purposes of this project we considered keeping the background constant, which allowed us to distinguish between the foreground (object of interest) and background.

To understand how the test environment is chosen, first it is important to look into what is meant and needed for a good testing environment. A testing environment is a setup of software and hardware on which the testing team is going to perform the testing of the newly built software product. This setup consists of the physical setup which includes hardware, and logical setup that includes Server Operating system, client operating system, database server (if applicable), front end running environment, browser (if web application), or any software components required to run this software product. This testing setup is to be built on both the ends – i.e. the server and client. In the case of GRID there was set of hardware and software being used in conjunction with a host computer.

To choose an effective test environment there were a few essential items to consider. The components used in the testing were closely mirror those of the average user – i.e. computer, OS, RAM, memory, Software etc. This was done so that all testing data can be relevant to the expected user and any errors can then be addressed before the system goes out to the users.

In addition, there were other factors that were taken into consideration. Customer's Environments: Understanding clearly the environment in which the customer is running the software. This had to be checked not only for server but also for the user's machines. The environments factors could be the hardware, OS, Database, Front end tools, browsers etc. Take care of all the versions of OS, browser the customer machines that are going to run this application. Test Server: Building the test environment as much as possible a replica of customer environment. This was applied to Server and client machines as well. Having a separate Test Server: Built the test environment on a separate server free from development and dedicated exclusively for testing purposes. This is so the conclusions that are drawn from the tests are not influenced by other applications, or items, on the computer. Understand business requirements well: The testers and test lead were very clear about the customer requirements based on which the test cases are to be built. More understanding, more coverage. Much clearer understanding, wider coverage. Finally, Documentation: Aesthetically document each and every test that is performed for a unit, module or integration testing of the product. For the purpose of GRID the testing was conducted on a few different environments to insure that the best testing coverage is achieved. It was over a series of consumer grade computers all with some version of the Microsoft windows operating system.

6.4. Software Testing

Testing was a critical part of any design project. This is especially true for the GRID project. Basically, every component of the system needed to be tested both individually and connected with other parts of the system. The goal of software testing is more than just finding bugs in the system and fixing them. Software testing is a process of discovering whether the system you built lives up to the standards of quality, efficiency, effectiveness and compliance to the system requirements.

There were many places where a failure may inter the project. It may be due to the programmer, analyst, or other individual during the lifecycle of the project. There could arise problems because of the complexity of the code, infrastructure, or how the issue to be solved was handled. Other issues may occur when integrating technologies or if there are many components in the system interacting with each other. These are all internal failures, though failures are not limited to just being internal. Outside of the system architecture failures could occur within the system due to environmental conditions or other program interactions. If the system has a failure there are repercussions for not only the user of the system but potentially everyone involved in the lifecycle of the system. If the system is well tested then it can reduce the possibility of any errors.

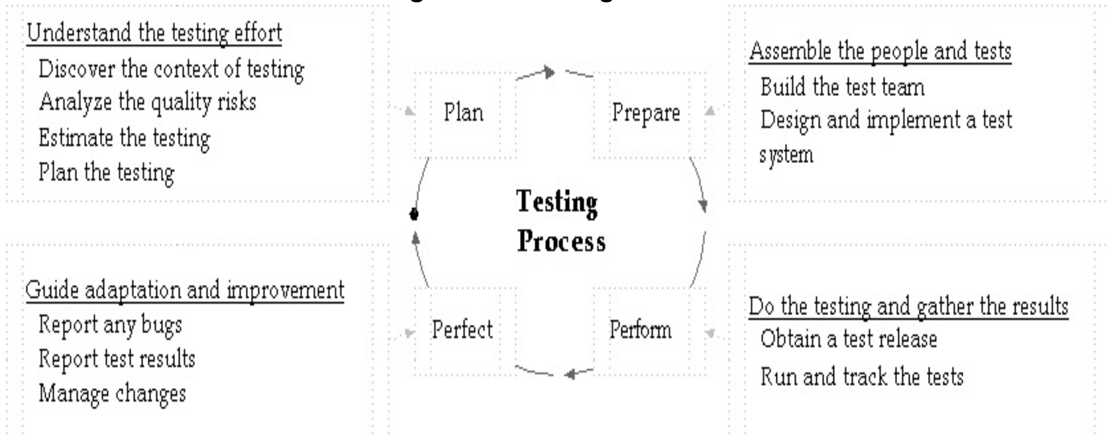
Typical testing objectives and ones that will be implemented for GRID include:

- Finding bugs and providing the information needed to fix the bugs
- Gain confidence about the level of quality of the system
- Prevent defects through early reviews and testing
- Provide information about the important aspects of the quality of the system

These objectives gave a clear idea into how the system works as well as assurances in the quality, effectiveness and efficiency of the system.

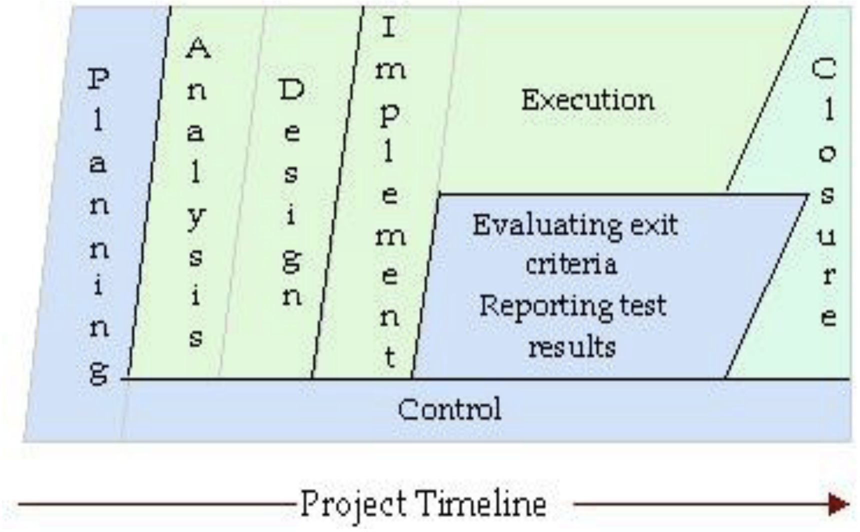
The test process can be summarized into three phases, Phase one: Planning and control, Phase two: Analysis and design and Phase three: Implementation and execution. In order to make sure that the tests cover the project in entirety the tests must be planned, prepared, performed and perfected. This cycle is a repetitive cycle that continues for the life of the system until it is ready for use.

Figure 67: Testing Process


Understand the testing effort
Discover the context of testing
Analyze the quality risks
Estimate the testing
Plan the testing
Plan
Prepare
Testing Process
Perfect
Perform
Assemble the people and tests
Build the test team
Design and implement a test system
Guide adaptation and improvement
Report any bugs
Report test results
Manage changes
Do the testing and gather the results
Obtain a test release
Run and track the tests

The three phases of the testing process were utilized for the proper testing of GRID. Phase one Planning and Control focuses on understanding the scope of the project. Phase two Analysis and Design reviews the system requirements to begin to design effective tests. Finally Phase three Implementation and Execution works on developing and executing the designed tests. These activities run throughout the life of the project and continue to evolve with the project.

Figure 68: Testing Phases



Each testing phase had its own specific goals that are exclusive to that phase of the testing process. Within Phase one there are the two sections of planning and control. The specific planning was:

- Determined testing scope, risks, objectives, and strategies
- Determined required test resources
- Implemented the test strategies
- Scheduled other test phases
- Determined the test exit criteria

Control which happened throughout the lifecycle of the project are as follows:

- Measured and analyze results
- Monitored progress, coverage and test exit criteria
- Initiated any needed corrective actions
- Made decisions concerning the system

For Phase two, first, the specific goals of the Analysis were:

- Reviewed the test basis (i.e. requirements or design specifications system architecture, etc.)
- Identified and prioritize test conditions, test requirements, or test objectives and required test, data based on analysis of test items
- Evaluated testability of the requirements and system

Second part of phase two Design:

- Designed and prioritize combinations of test data, actions, and expected results
- Identified the test data needed for test conditions and cases
- Designed the test environment
- Identified infrastructure, tools

For Phase three there is the implementation followed:

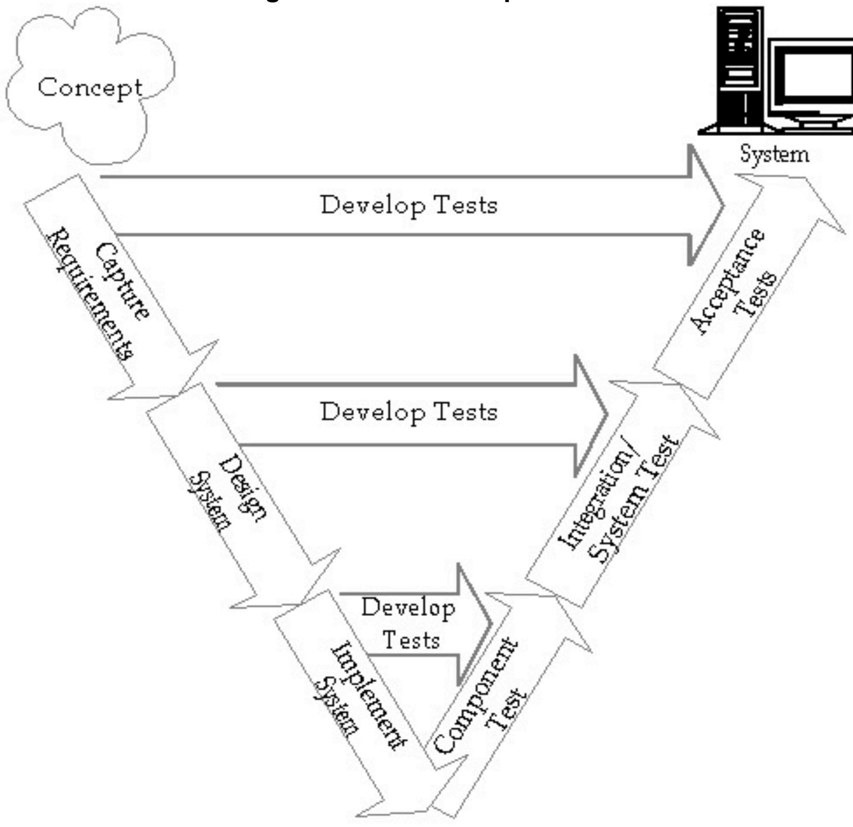
- Developed, implemented, and prioritized test cases, create data, write procedures
- Created test harnesses, scripts
- Organized test suites and sequences of test procedures
- Verified the test environment

And finally the second part to phase three Execution:

- Executed test cases (manual or automated)
- Logged test results, and other information about test and testware
- Compared actual and expected results
- Reported and analyze incidents
- Repeated corrected and/or updated tests
- Ran confirmation and/or regression tests

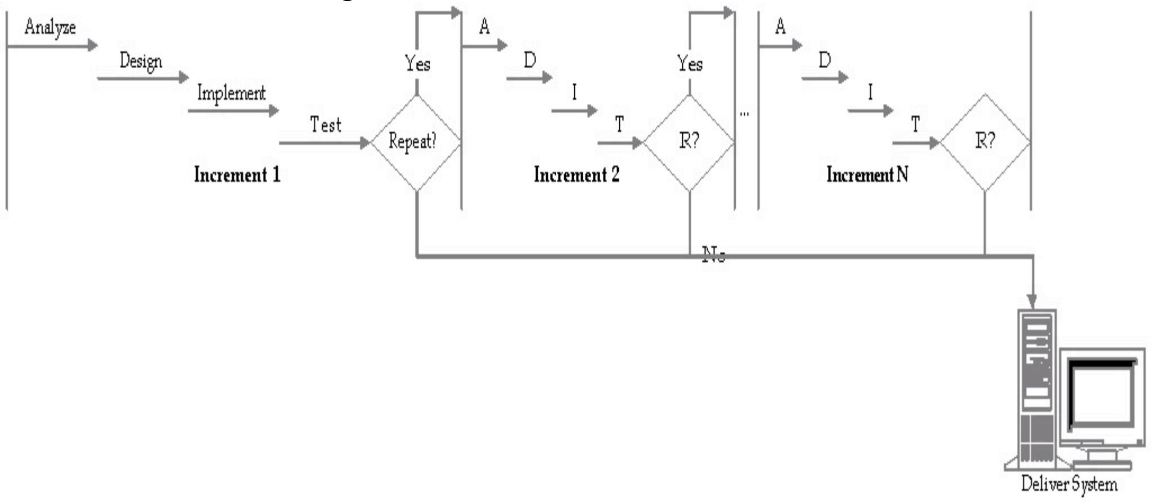
The specific goals of each phase help to organize and control the testing process in order to get the best possible results and the most efficient use of time testing. There is always needed a good balance between thoroughness and time efficiency. In order to maximize the use of time and testing there are a few different lifecycle models that streamline the entire process. The first of these models is known as the "V" or sequential model. In the figure below you can see the typical process of this model.

Figure 69: "V" or Sequential Model



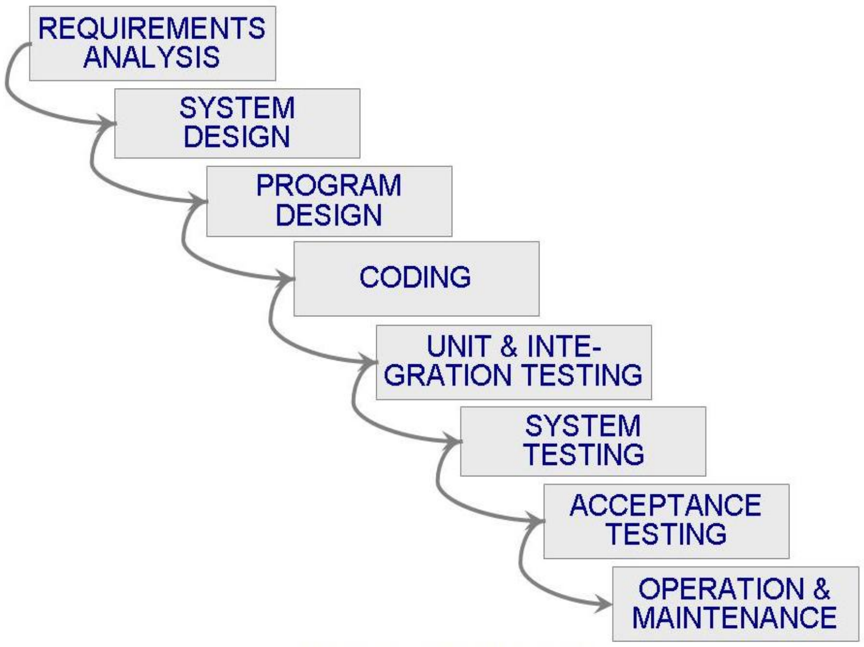
As you can see from the figure above this model is very intuitive. It combines the different phases of the testing process seamlessly with the development of the project and has goals set for each stage. As you continue into the design of the project you are designing the testing for the project right alongside and are starting the earliest stages of testing while the development is still in process. There are however a few drawbacks with this model the first being that it can be hard to plan for the testing that far in advance. The second is if the plans for the project fail or change, the end testing will suffer the effects. Depending on how big the changes are most to all of the tests will need to be redesigned.
The next lifecycle model to look at is the Iterative or Incremental model. The figure 64 below demonstrates the basic appearance of this model.

Figure 70: Iterative or Incremental Model



This model is very repetitive as shown above. Typically this model was chosen when the project is very Schedule risk driven to hit the market in a specific time window or delivery date. The focus here is to build the core functionality first and get it so there is something to ship, then once the core is running properly the feature sets are grown around it. It was designed that way so that something can be shipped out quickly. This model is becoming a popular approach because once the core is finished it gives the team more time to develop better features. The drawbacks to this model are that it is very tempting to ship a system that has very buggy features.
The next model that is typically the first approach most people take to coding or even solving problems is known as the Waterfall model. The figure 72 below illustrates what it looks like.

Figure 71: Waterfall Model

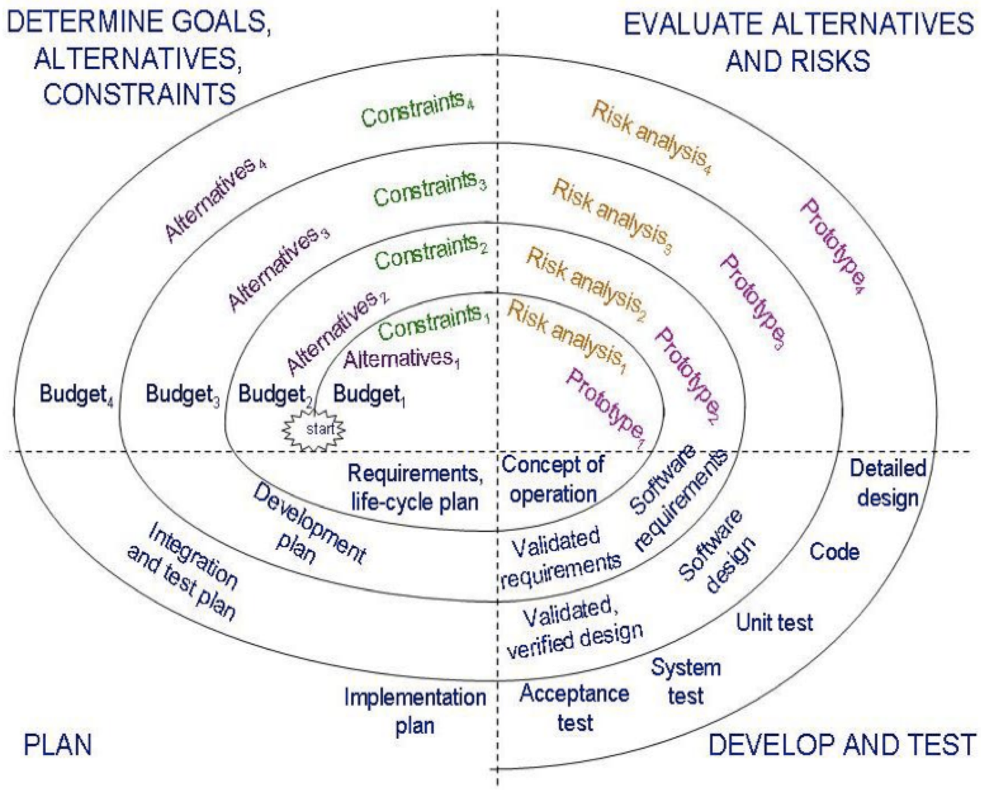


As you can see the Waterfall model is very straightforward and goes through each of the steps one at a time. It does address all the concerns and delivered a final product to the client in a timely manner.. The biggest problem with this model is that there is no iteration. If something happens to change the requirements or some part of the design is flawed there is no iteration to catch it, the whole process would need to be scraped and restarted. Even though this model is almost the first one most people turn to it is not the best solution to the problem.
The final model that was considered is known as the Spiral model. This particular model was suggested by Boehm (1988). It combines the development activities with risk management to minimize and control risks. The model is presented as a spiral in which each iteration is represented by a circuit around four major activities:

- Plan
- Determined goals, alternatives and constraints
- Evaluated alternatives and risks
- Developed and test
-

The figure 73 below shows what a typical Spiral model looks like.

Figure 72: Spiral Model



As you can see the amount of iteration and reanalysis that goes on in this model makes this model the most effective at getting a perfect system. Although it is great in the since that it rehashes everything until there are no problems with the system, this model requires a large amount of time and dedication that unfortunently will not be possible for every project that goes on.

For GRID there are two choices in the matter of lifecycle model that would be beneficial to use. The first would be the “V” model. This model is good for GRID because it allows for iteration and integrates the design of the software right alongside the design of the testing procedures. The one drawback for this model would be that if during the course of the project some fundamental design changes occur it could make the final end result suffer. The second model that would be fitting for GRID is the Incremental model. This model allows for quick development of the core system and then later focus on the development of the feature sets after there is a working core. This is an excellent use of time for this project as it has a deadline that must be met and also has specific requirements on the core system and the completion of feature sets is more of a bonus rather than requirement. For these reasons the Incremental model will be the model of choice for the lifecycle of GRID.

In conclusion of the software testing, in respect to this system under development, The lifecycle model being purposed for use is the Incremental model which provided the structure for the three phase testing plan. Keeping in mind that the testing process is iterative and continues throughout the life of the development. The goal is to build a core system that can then have feature set built up around the core system.

6.4.1. Image Recognition

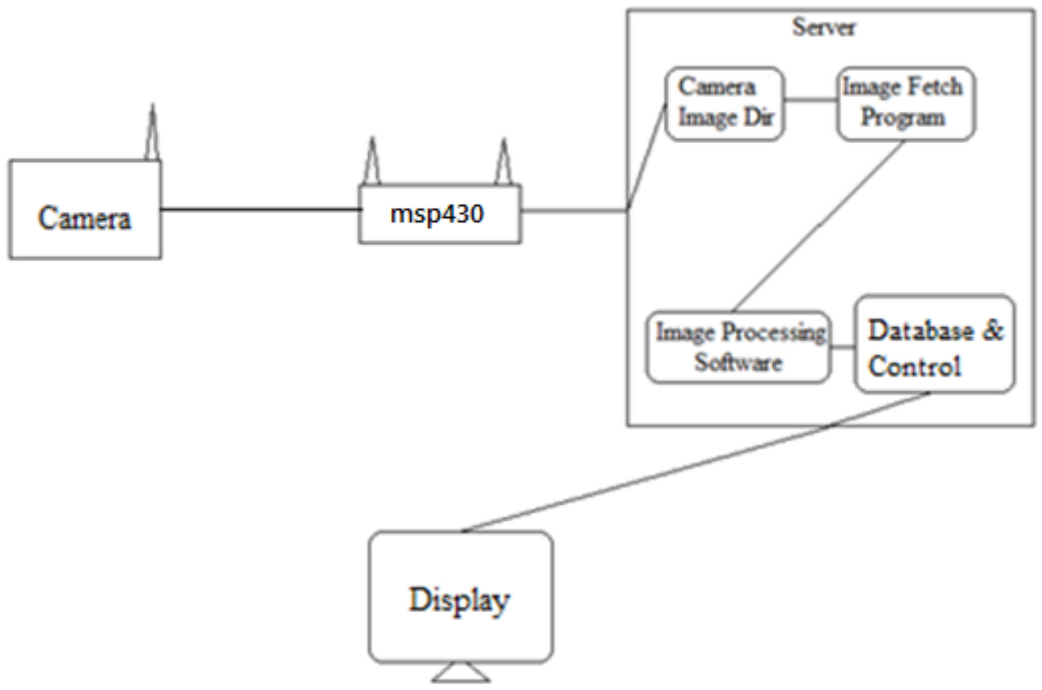
To begin with, a camera was mounted in a position with a good view of the hands of the user. This step was vital; the angle of which the camera views the gloves greatly determines how accurate our image processing will be. As for the vertical, ideally you’d like the camera to be pointing straight down on the users hands, so the closer the vector, which the lens draws with the ground, is to 90 degrees the better. The camera was communicating with the computer via Bluetooth and powered via recharge circuitry and battery. The server is the center of all the data, the images from each camera were saved in their own directory, or with unique file name structures, so the program knows what control image to compare it to. Once processed, the images were moved to another folder so the program does not accidentally analyze old images and update the system with this data. The output of the image processing will be transferred to one of 2 options to be determined once some test has been performed. First option, a data base that will be created with ID codes for each parking space, and therefore the program will have to be made to correspond and output accordingly, then this data will be used to update the display. Second option, integrate the database, display, and image processing software so that everything is processed in the same place and doesn’t need to exchange hands.

Following an Image at the Output of the msp430:

1. Camera took a snapshot
2. Image was sent to the msp430
3. Server acquired Image and stores in correct directory
4. Image fetch program detected a new image in the directory and stored it to memory
5. The image processing software determined what control image to use
6. Compared the 2 images
7. Outputted data sent to the database and Matlab software
8. Matlab software updates the display
    a. Note: the display may not be directly connected to the server in real implementations.

In Figure 74, shown below shows the Integration Block Diagram of the camera with the msp430 and server.

Figure 73: Integration Block Diagram



During our testing phase we chose to revise the details of our current system design, but the overall system shown in the block diagram in Figure 74 should remain true.

7.0 Administrative Content

7.1. Budget and Finance

The group was unable to receive funding from outside sources or sponsors. As a result, the team provided the funding for this project. In Table 13, a list of all the known parts for the project and their estimated costs are listed. Each team member donated around $100, giving the team a budget total of $400 to spend. The financial goal in this project is to make a low cost system under $400. A field was added onto the table called miscellaneous expenses, which will cover for extra expenses the group faced. The team also found it more cost effective to buy some devices in bulk. For instance, the Near-Infrared LED was found in mass at a cheaper rate. Not only does buying some parts in majority lower the price, but if the team ends up going through too many of certain parts, then an adequate supply will still remain for emergencies. The group also got some parts donated by Texas Instruments (TI) and by UCF faculty. Most of the parts were purchased through different vendors. For more information refer to the following parts list for further cost details:

Table 13: GRID Budget Breakdown

Part	**Price**
MPU 6050 x 2	**$21.00**
Bluetooth Module r2232 x5	**$59.80**
Stellaris Launchpad x2	**$26.00**
MPU 6050	**$12.86**
Near-IR LEDs	**$9.92**
Li-Ion Battery x2	**Donation**
IR Sensor	**Donation**
LTC1147-3.3	**$5.25**
EEPROMs	**$7.98**
SMT Resistors, Capacitors, and LEDs	**$15.00**
Battery Charger	**Sample**
Charge Controller	**Sample**
PCBs	**$100.00**
Other	**$100.00**
Total:	**$357.81**

7.2. Milestone Chart

This section contains the plan that our group established in order to keep track with all the responsibilities that needed to be accomplish during Senior Design I and II.

Table 14: Senior Design I Milestones

September 2012	
01	Started gathering project ideas
10	Started gesture library
15	Finalized project definition
16	Started researching in all subjects
28	Got project approved by Dr. Richie
29	Submitted proposal for sponsorship
30	Divided all parts in a fair amount for the group members
October 2012	
1	Researched
4	Became familiar with Java and C
11	Improved block diagram based on research
13	Started writing. Goal: 5 pages a week
15	Reviewed OpenCV Library
20	Determined function of each subsystem
27	Group meeting to discuss design improvements
November 2012	
2	Understood the process of the final project
6	Decided the actual parts that will be used for the project
7	Finalize design based on research
10	Group meeting to check page count
17	Merged the individual research of all group members
19	Came up with a test procedure to verify functionality of parts
30	Started ordering parts for testing as design was being done
December 2012	
1	Group meeting to check page count
2	Made final touches to our paper- Formatting
4	Improved testing procedures for all parts
6	Submitted final paper
12	Ordered parts as design was being done
15	Began writing object tracking code
30	Started prototyping

In the second portion of the class, Senior Design II, is when all the design will be implemented. For this semester the group prepared a milestone chart to follow. These charts are tentative and subject to change depending on the work flow. All group members have different schedules but they will make their best effort to follow the milestone schedule and to succeed in the prototype and design. Below table 15 shows the Senior Design II milestone:

Table 15: Senior Design II Milestones

January 2013	
5	Had all or most parts in hand Purchased at store miscellaneous hardware and electronic material Updated spending report with all parts purchased Set up a work station outside of school to be able to prototype from home Revisited milestones and made sure next months are realistic
15	Started testing each part separately
17	Tested code integration with individual parts if required
20	Ran simple codes on msp430 and Stellaris development board to get used to them
21	Started website to track project progress
23	Started putting together glove mount
30	Group meeting to review progress
February 2013	
1	Started testing of parts
3	Implemented visible light filter to camera
6	Started testing the glove
12	Tested battery lifetime and efficiency of charge circuitry
13	Started designing PCB with Eagle CAD
22	Decided on sewing the glove circuitry based on circuitry size
30	Group meeting to discuss progress
March 2013	
1	Started interfacing different parts of the project
2	Implemented object tracking code in hardware
3	Debugged and optimize code
4-9	Worked on project through Spring Break
15	Started working on PowerPoint presentation
20	Group meeting to show progress or design improvements
30	Ordered PCB
April 2013	
1	Ordered any parts if needed
3	Made sure most or all parts are interfacing with each other
5	Tested all gestures
7	Updated gesture library
8	Implemented PCB
10	Made final aesthetic touches
12	Made sure all parts are interfacing with each other
?	Presented Project

15	Started website for final project
15	Updated documentation according to new design
29	Turned in any final documentation if any
24-30	Wrapped up the semester

7.3. Work Distribution

- Martin- Gloves
- Landon- Code for computer controller
- Pamela and Evianis- Camera Circuitry

Gloves:

- Selected the optimum microcontroller to meet specifications.
- Selected gyroscope/accelerometer module.
- Selected near-IR LED as well as desired brightness for optimum power consumption as well as object detection.
- Designed circuitry attachment method
- Wrote all code relating to data acquisition from the gyroscope and accelerometer modules via I^2C (two wire interface).
- Wrote all code relating to data transfer.
- Optimized code with the use of interrupts in order to take advantage of low power modes in order to extend battery life.
- Selected battery and design recharge circuitry to be implemented.

Code for computer controller:

- Designed GUI for user interaction with device.
- Interpreted raw data from microcontroller into defined computer actions.
- Interpreted feed from msp430 into cursor movement.
- Mapped the coordinates of the object to the appropriate locations on the display to solve for any difference is resolution.
- Optimized code to use as little of the computer resources as possible.

Camera Circuitry

- Implemented Bluetooth Communication between PC and camera circuitry
- Selected most cost effective microprocessor in order to meet requirements.
- Selected most near-IR sensitive sensor
- IMplemented a visible light filter
- Researched of ways to implement object tracking code in hardware.
- Outputted the object(s) coordinates to the computer in real-time
- Outputted blob size in computer
- Designed printed circuit board
- Chose camera housing

8.0 Product Operation

8.1. Prerequisites

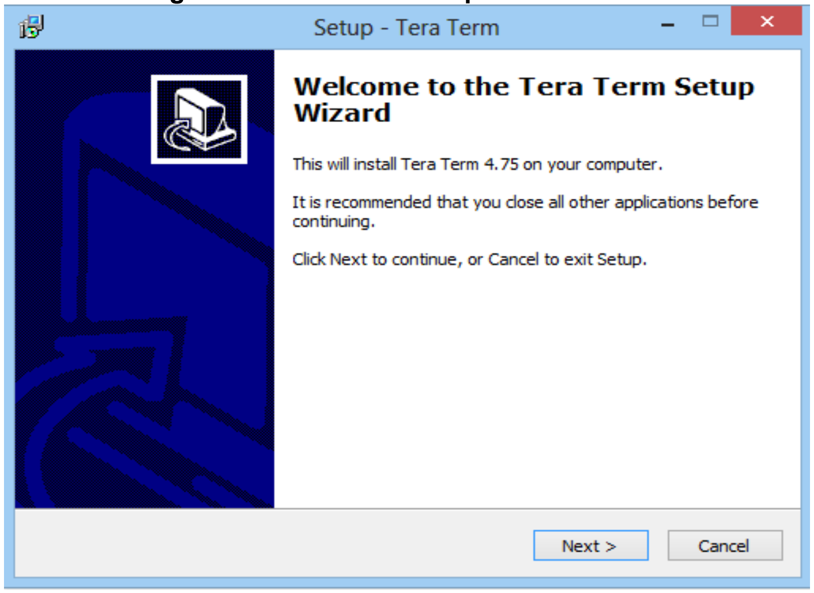
- Computer with Windows operating System.
- Java Run-time installed
- Terminal program installed

8.2. Installing a Terminal Program

Note: In order for the Driver program to detect the appropriate communication ports used by the camera and glove a terminal program is needed. Functionality has been tested with **“*Teraterm”,*** but any terminal program will suffice*.*

- Download the .exe file and save it to an easily accessible location in c:\. http://download.cnet.com/Tera-Term/3000-20432_4-75766675.html
- Double Left click, or right click and select “open” to initialize the installer.

Figure 74: Tera term setup wizard screen



Welcome to the Tera Term Setup Wizard

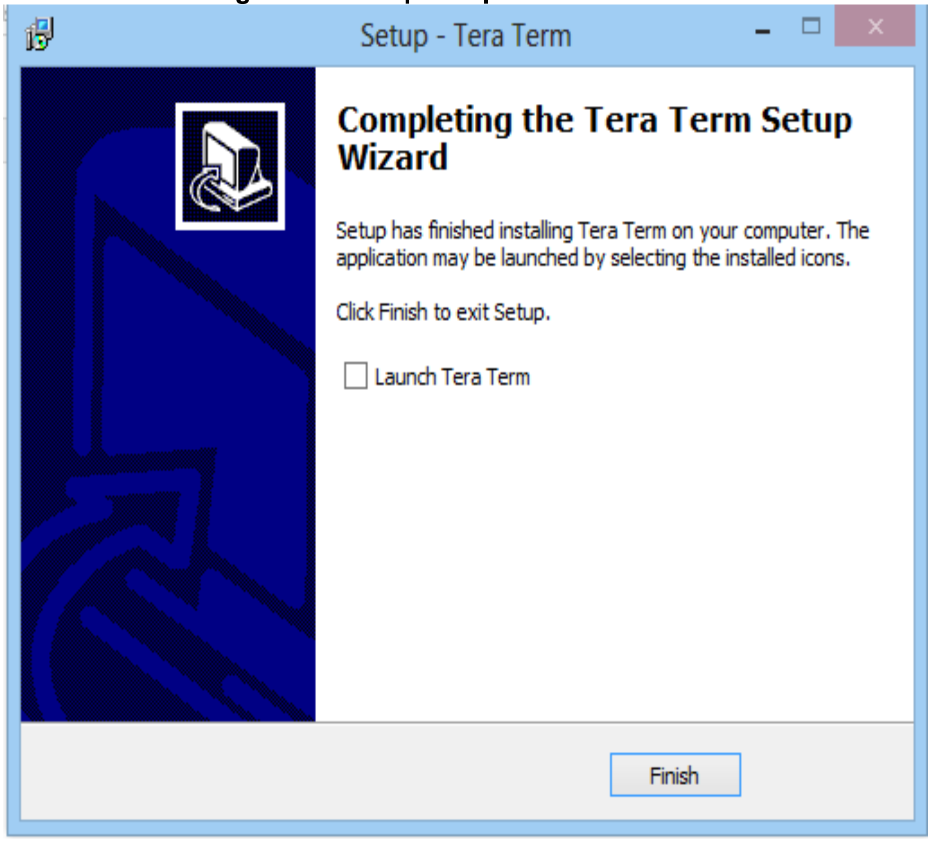
This will install Tera Term 4.75 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

- Follow the on-screen instructions and click finish.

Figure 75: Setup completed screen view



Setup - Tera Term

Completing the Tera Term Setup Wizard

Setup has finished installing Tera Term on your computer. The application may be launched by selecting the installed icons.

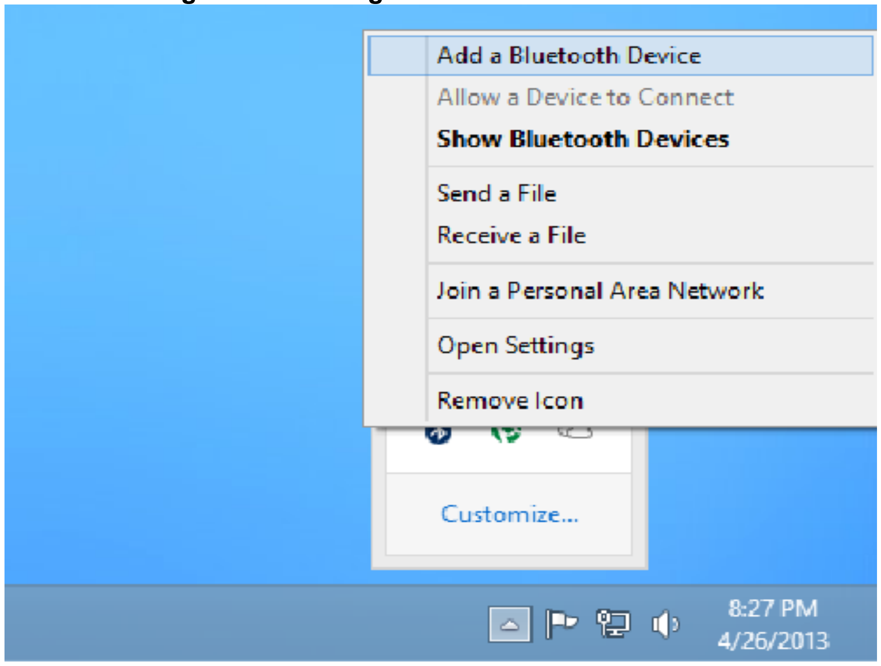
Click Finish to exit Setup.

☐ Launch Tera Term

8.3. Connecting to the Glove/Camera

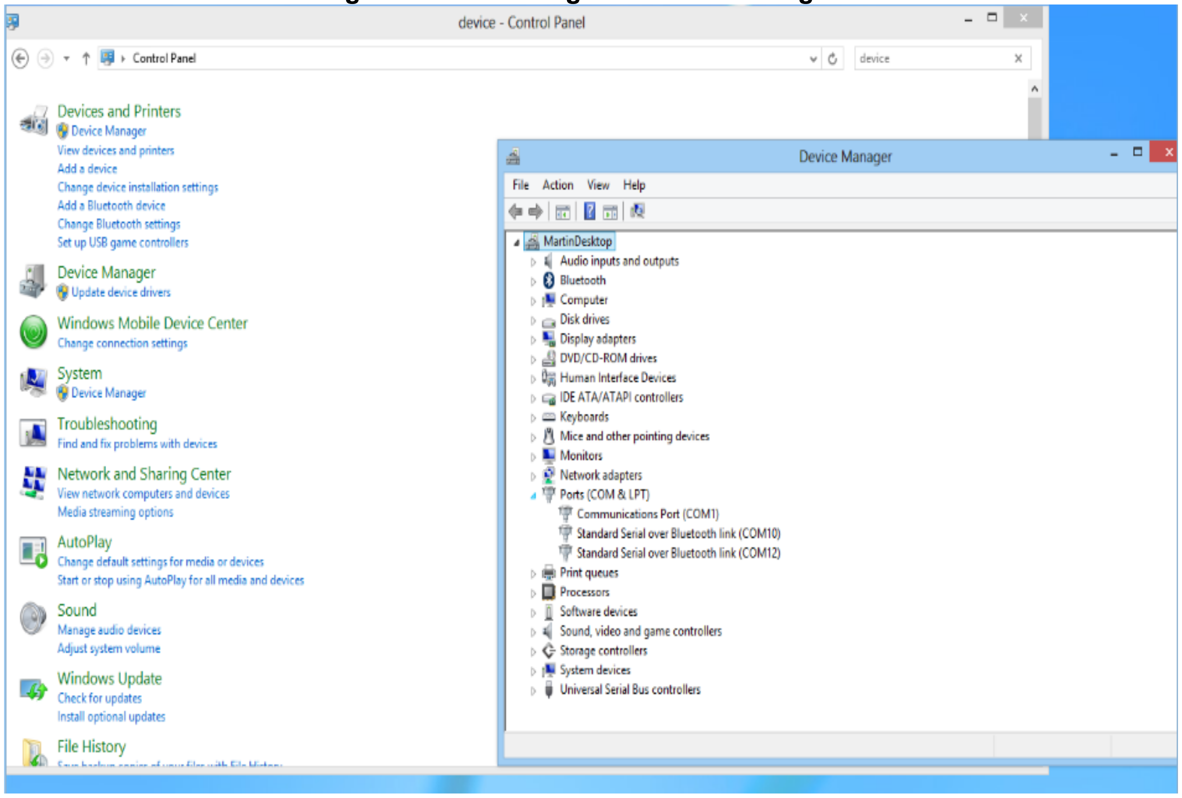
- To power on the device, move the two-state switch to the “on” position, as indicated in the PCB.
- If the batteries are charged and the voltage regulators are working, the BLUE L.E.D. mounted on the PCB should blink roughly twice per second. This indicates the device is Bluetooth discoverable.
- On the computer, in the toolbar left click the Bluetooth icon and select “add Bluetooth devices”.

Figure 76: Adding a Bluetooth device



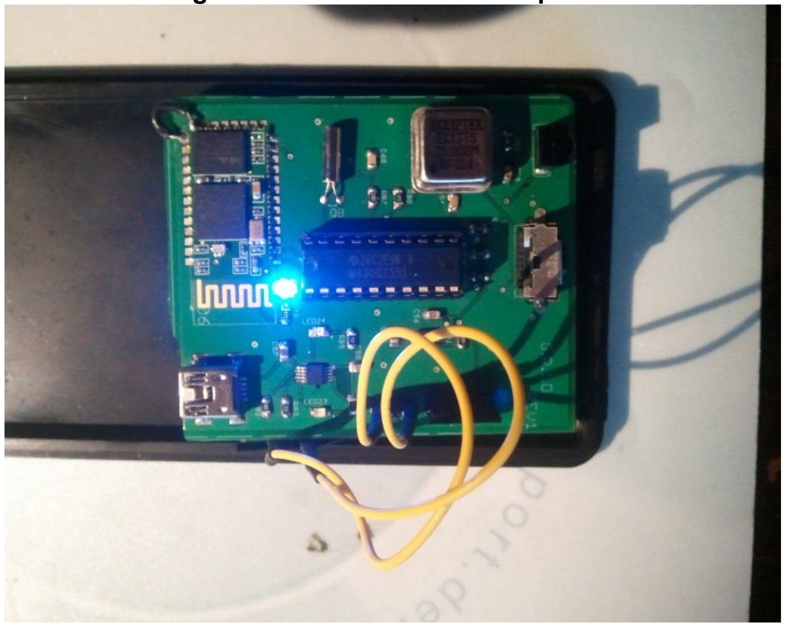
- The glove and camera will appear in the list of accessible devices.
- Once selected, the user will be prompted to insert a password. For both devices the password is: **“1234”**.
- Once the computer connects to the device, there will be a notification of the communication port assigned to each device. Alternatively, open control panel and in the search box (top right corner) type device manager. In the device manager window expand **“Ports (COM & LPT)”** to see the connected devices.

Figure 77:Accessing the Device Manager



- Once both devices have communication ports assigned, open *teraterm*. Select **Serial** and choose the port assigned to the camera and click **OK**.
- If pairing is successful the BLUE LED should stop blinking, and instead remain in the “on” state.
- Repeat the previous step to connect to the glove.

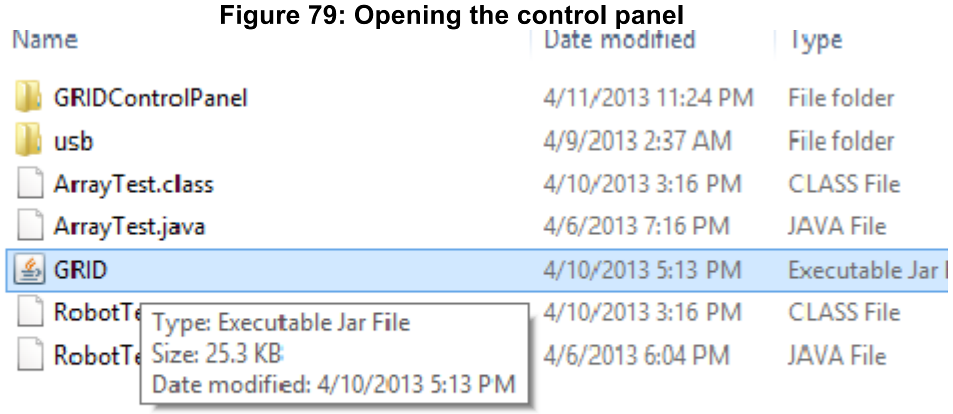
Figure 78: Bluetooth module paired



8.4. Running the Driver Program

Open the **"G.R.I.D."** folder and select **"*GRID.jar*"** This should open the control panel screen.

Figure 79: Opening the control panel



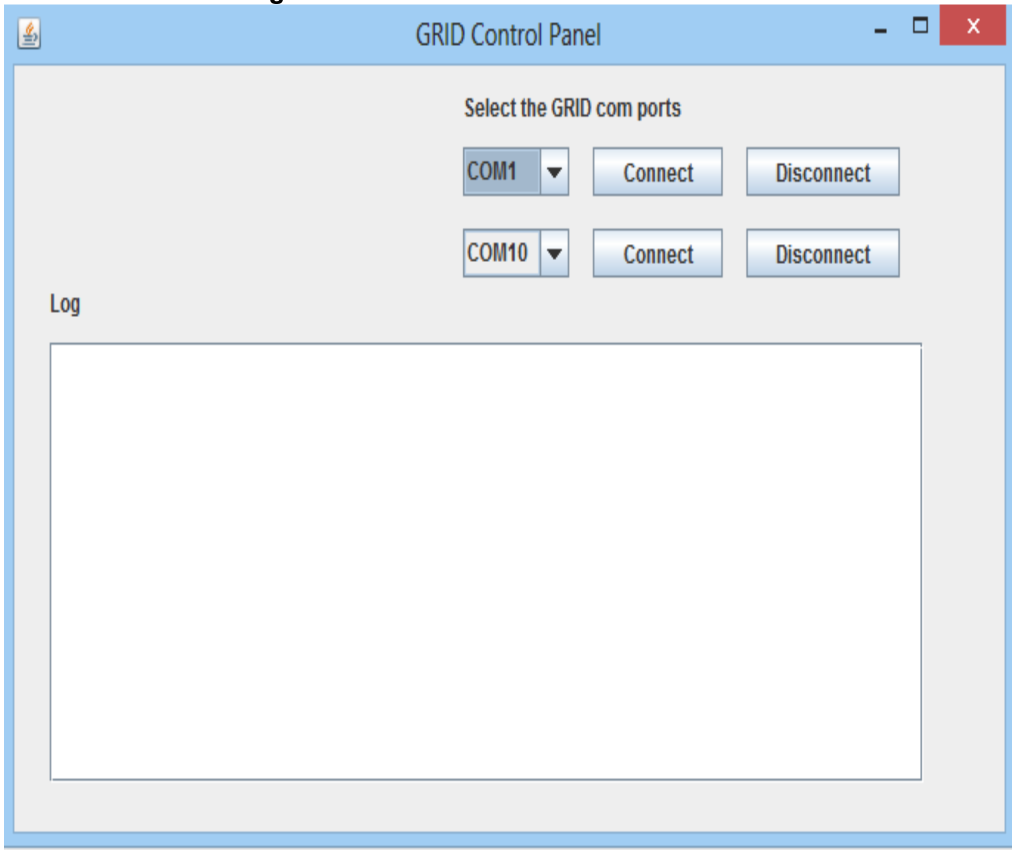
Name	Date modified	Type
GRIDControlPanel	4/11/2013 11:24 PM	File folder
usb	4/9/2013 2:37 AM	File folder
ArrayTest.class	4/10/2013 3:16 PM	CLASS File
ArrayTest.java	4/6/2013 7:16 PM	JAVA File
GRID	4/10/2013 5:13 PM	Executable Jar
RobotT	4/10/2013 3:16 PM	CLASS File
RobotT	4/6/2013 6:04 PM	JAVA File

Type: Executable Jar File
Size: 25.3 KB
Date modified: 4/10/2013 5:13 PM

On the drop down menus for the camera and glove select the appropriate communication port, but DO NOT click "camera" or "glove" to establish connection yet.
On ***"Teraterm",*** and on the camera window click file, disconnect. Alternatively, press **"ALT+I"** on the keyboard.
Likewise disconnect the glove from ***"Teraterm".***

At this time, proceed to click the **“Camera”** and **“Glove”** buttons in the driver program.

Figure 80: GRID Control Panel



If connection is successful the BLUE LED should remain “on” on both devices, and the driver is now decoding their transmissions.

Following pairing, introducing the IR LED (index finger) to the camera will result in the appropriate movement in the mouse. Right and Left clicks are located on the middle and ring fingers respectively.

8.5. Using the Gesture Library

G.R.I.D is able to distinguish 7 different gestures, and map them to computer functions.

1. Fast forward: while in windows media player quickly swipe in the direction of the user’s right.
2. Rewind: while in windows media player quickly swipe in the direction of the user’s left.
3. Full screen mode: while in windows media player quickly swipe in the up direction.

4. Play/Pause: while in windows media player quickly swipe in the down direction.
5. Zoom in: In any program quickly swipe towards the screen.
6. Zoom out: In any program quickly swipe away from the screen.
7. Refresh: while in Firefox or similar web browser make a circular motion to the right ending with the palm of the user's hand facing up.

8.6. Charging the Battery

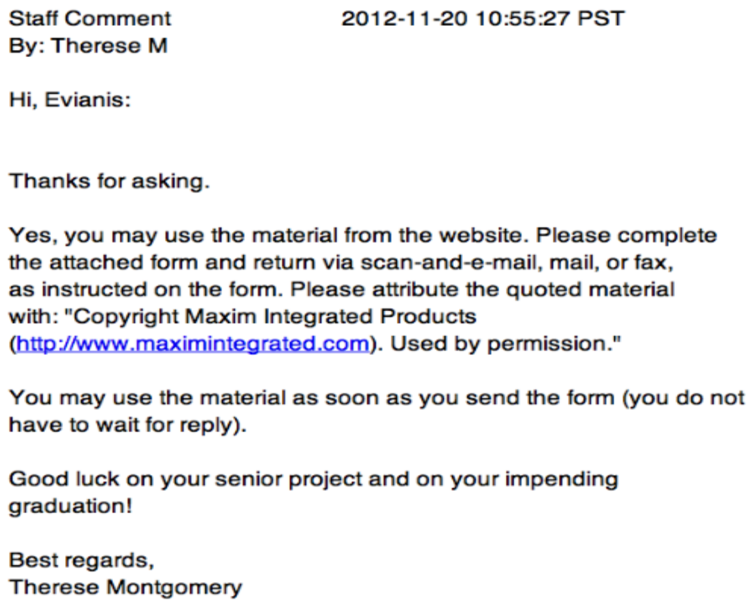
Note: the maximum rate of char is 500mA.

- In order to charge the batteries a "*Mini USB"* cable is required.
- Connect the cable to a power source (PC) and to the device.
- If the Power source is functioning properly and delivering the correct voltage, the GREEN LED will turn "on".
- If the batteries are sufficiently discharged the RED LED will turn "on" indicating the charging cycle has started.
- Once the battery is fully charged, the RED LED will turn "off", but the GREEN LED will remain "on" for as long as power is bring supplied by the source "PC".

Appendices

Appendix A – Permissions

1. Maxim Integrated Product Permission



Staff Comment 2012-11-20 10:55:27 PST
By: Therese M

Hi, Evianis:

Thanks for asking.

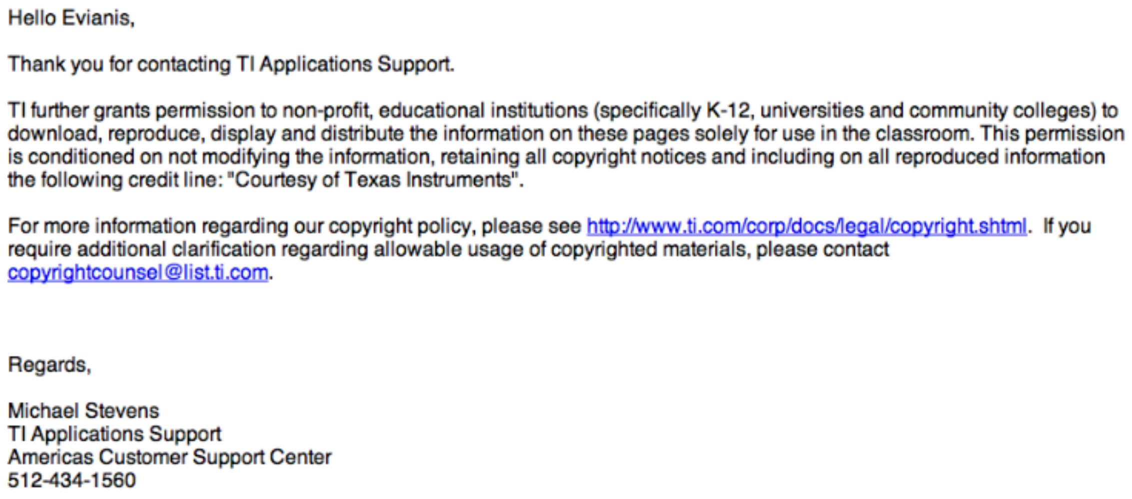
Yes, you may use the material from the website. Please complete the attached form and return via scan-and-e-mail, mail, or fax, as instructed on the form. Please attribute the quoted material with: "Copyright Maxim Integrated Products (http://www.maximintegrated.com). Used by permission."

You may use the material as soon as you send the form (you do not have to wait for reply).

Good luck on your senior project and on your impending graduation!

Best regards,
Therese Montgomery

2. Texas Instrument Permission



Hello Evianis,

Thank you for contacting TI Applications Support.

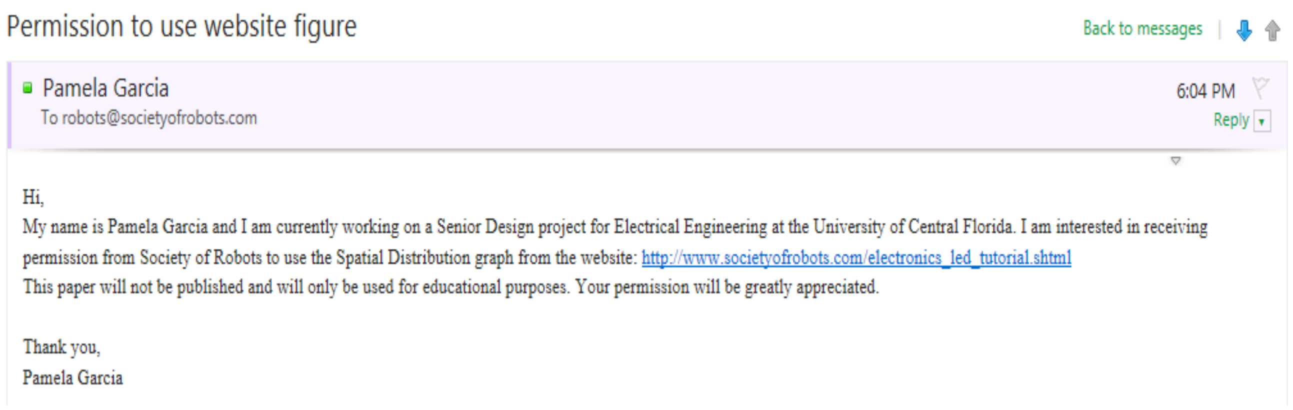
TI further grants permission to non-profit, educational institutions (specifically K-12, universities and community colleges) to download, reproduce, display and distribute the information on these pages solely for use in the classroom. This permission is conditioned on not modifying the information, retaining all copyright notices and including on all reproduced information the following credit line: "Courtesy of Texas Instruments".

For more information regarding our copyright policy, please see http://www.ti.com/corp/docs/legal/copyright.shtml. If you require additional clarification regarding allowable usage of copyrighted materials, please contact copyrightcounsel@list.ti.com.

Regards,

Michael Stevens
TI Applications Support
Americas Customer Support Center
512-434-1560

3. LED Spatial Distribution


Permission to use website figure

Back to messages

Pamela Garcia

To robots@societyofrobots.com

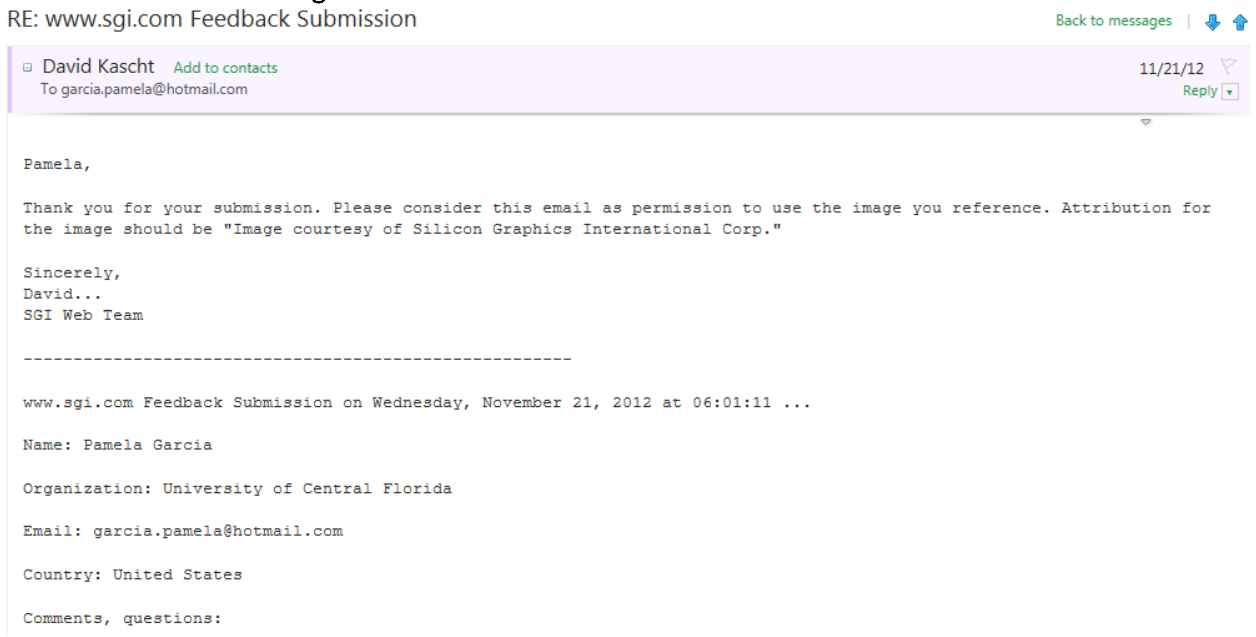
6:04 PM

Reply

Hi,
My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from Society of Robots to use the Spatial Distribution graph from the website: http://www.societyofrobots.com/electronics_led_tutorial.shtml
This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.

Thank you,
Pamela Garcia

4. Frame Rate Figure


RE: www.sgi.com Feedback Submission

Back to messages

David Kascht Add to contacts

To garcia.pamela@hotmail.com

11/21/12

Reply

```
Pamela,

Thank you for your submission. Please consider this email as permission to use the image you reference. Attribution for
the image should be "Image courtesy of Silicon Graphics International Corp."

Sincerely,
David...
SGI Web Team

-------------------------------------------------------

www.sgi.com Feedback Submission on Wednesday, November 21, 2012 at 06:01:11 ...

Name: Pamela Garcia

Organization: University of Central Florida

Email: garcia.pamela@hotmail.com

Country: United States

Comments, questions:
```

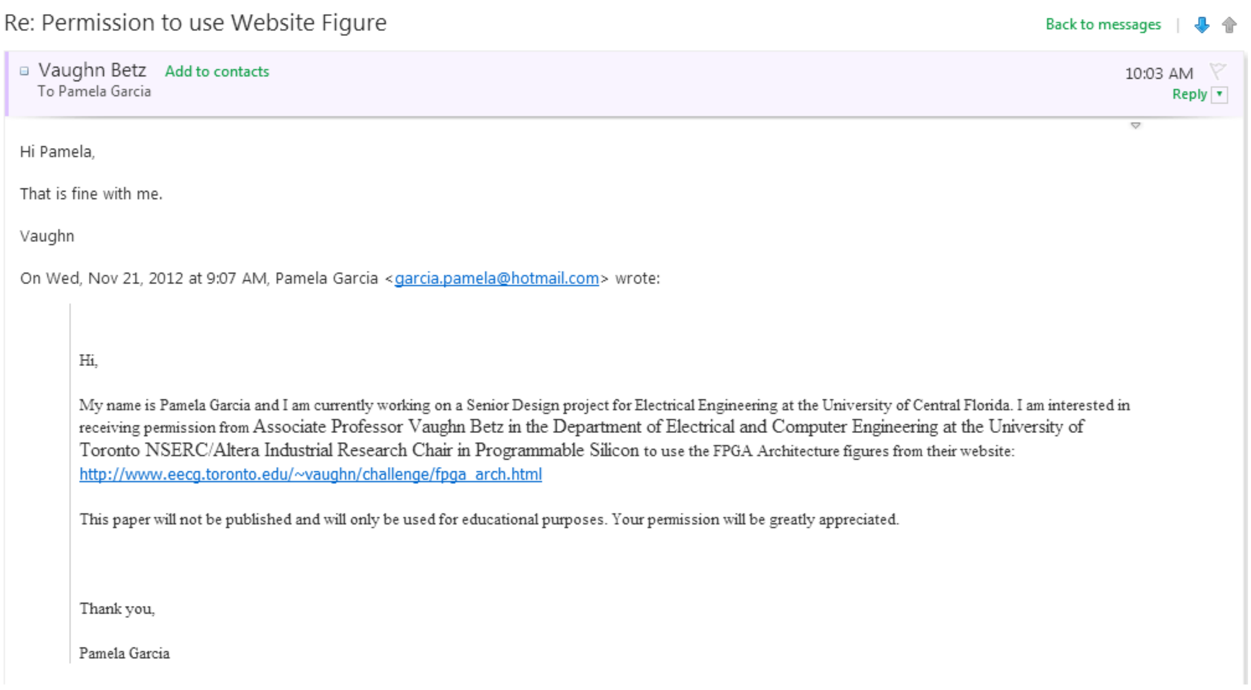
```
11/21/2012

Hi,
My name is Pamela Garcia and I am
currently working on a Senior Design
project for Electrical Engineering at
the University of Central Florida. I am
interested in receiving permission from
Silicon Graphics International Corp to
use the frame rate figure from their
website:
http://techpubs.sgi.com/library/tpl/cgi-
bin/getdoc.cgi/0650/bks/SGI_Developer/bo
oks/Perf_GetStarted/sgi_html/ch10.html#i
d5214752
This paper will not be published and
will only be used for educational
purposes. Your permission will be
greatly appreciated.

Thank you,
Pamela Garcia

-------------------------------------------------------
```

5. FPGA Architecture Figures


Re: Permission to use Website Figure

Back to messages

Vaughn Betz Add to contacts
To Pamela Garcia

10:03 AM
Reply

Hi Pamela,

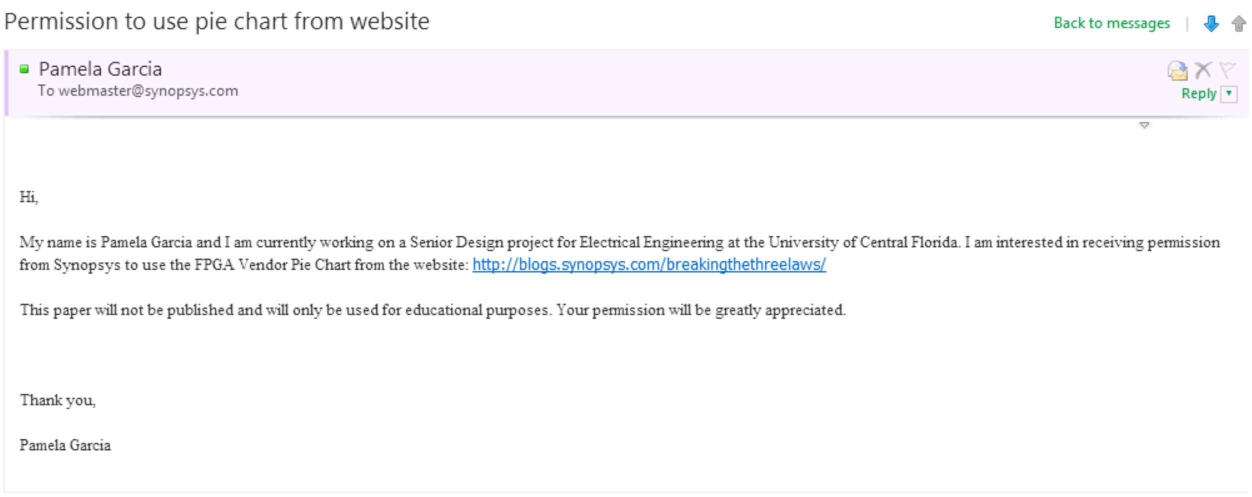
That is fine with me.

Vaughn

On Wed, Nov 21, 2012 at 9:07 AM, Pamela Garcia <garcia.pamela@hotmail.com> wrote:

> Hi,
>
> My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from Associate Professor Vaughn Betz in the Department of Electrical and Computer Engineering at the University of Toronto NSERC/Altera Industrial Research Chair in Programmable Silicon to use the FPGA Architecture figures from their website: http://www.eecg.toronto.edu/~vaughn/challenge/fpga_arch.html
>
> This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.
>
> Thank you,
>
> Pamela Garcia

6. FPGA Vendors Pie Chart


Permission to use pie chart from website

Back to messages

Pamela Garcia
To webmaster@synopsys.com

Reply

Hi,

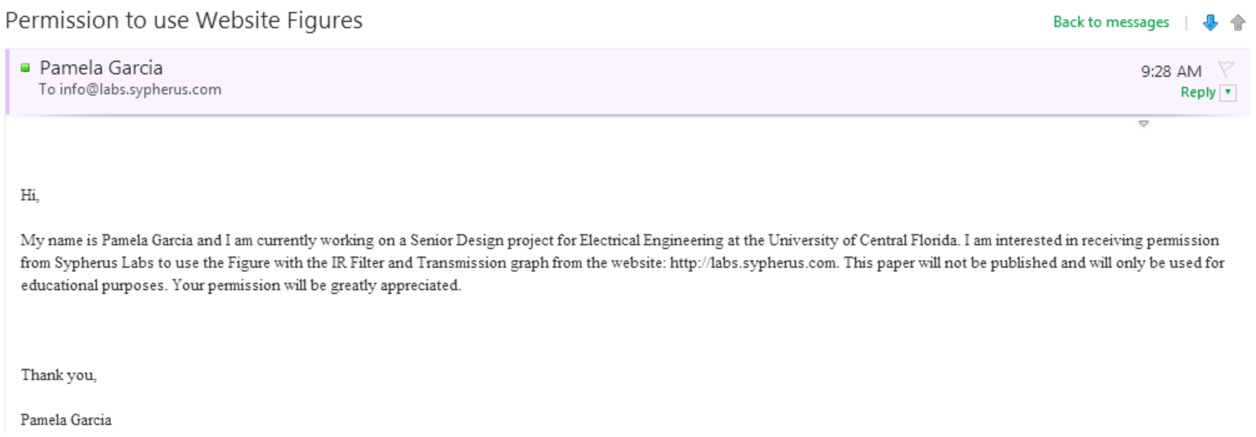
My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from Synopsys to use the FPGA Vendor Pie Chart from the website: http://blogs.synopsys.com/breakingthethreelaws/

This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.

Thank you,

Pamela Garcia

7. IR Filter Photo and Transmission Graph



Permission to use Website Figures

Pamela Garcia
To info@labs.sypherus.com
9:28 AM

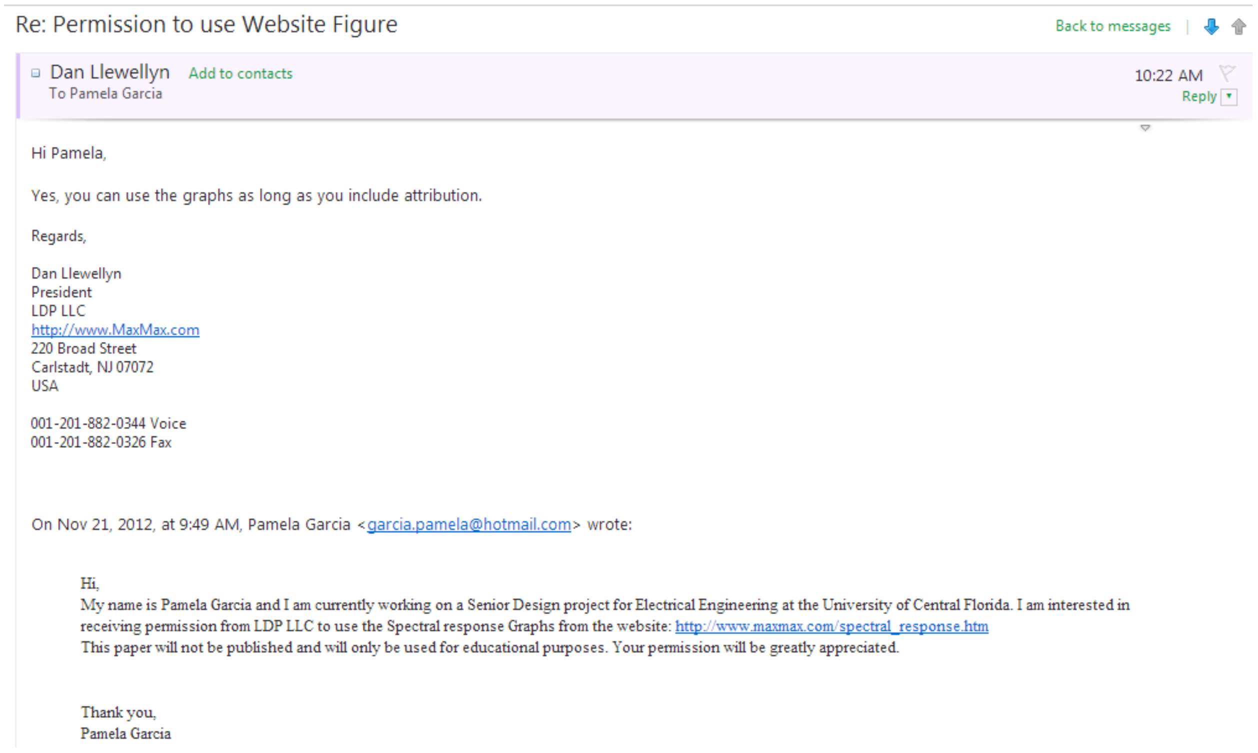
Hi,

My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from Sypherus Labs to use the Figure with the IR Filter and Transmission graph from the website: http://labs.sypherus.com. This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.

Thank you,

Pamela Garcia

8. Spectral Response Figure



Re: Permission to use Website Figure

Dan Llewellyn
To Pamela Garcia
10:22 AM

Hi Pamela,

Yes, you can use the graphs as long as you include attribution.

Regards,

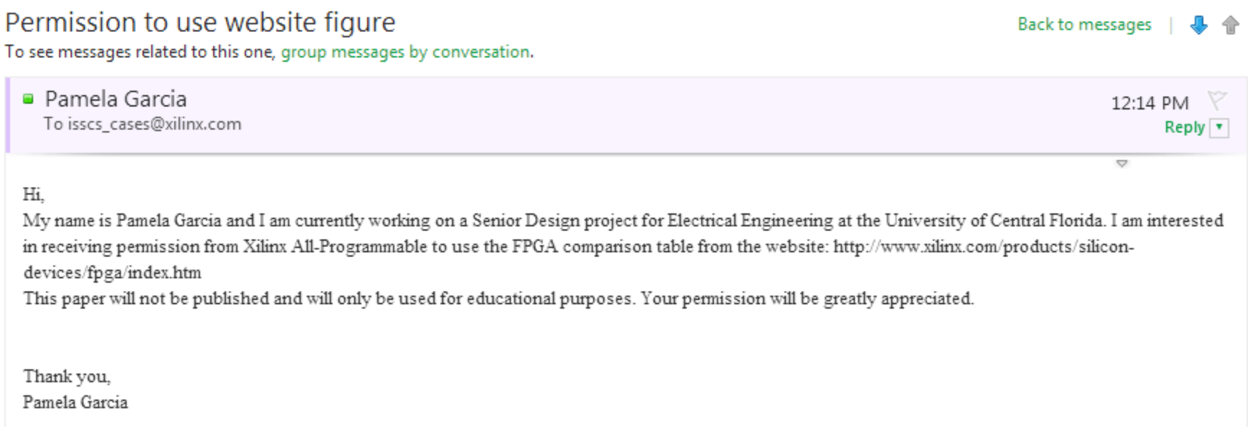
Dan Llewellyn
President
LDP LLC
http://www.MaxMax.com
220 Broad Street
Carlstadt, NJ 07072
USA

001-201-882-0344 Voice
001-201-882-0326 Fax

On Nov 21, 2012, at 9:49 AM, Pamela Garcia <garcia.pamela@hotmail.com> wrote:

> Hi,
> My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from LDP LLC to use the Spectral response Graphs from the website: http://www.maxmax.com/spectral_response.htm
> This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.
>
> Thank you,
> Pamela Garcia

9. Xilinx Table


Permission to use website figure

To see messages related to this one, group messages by conversation.

Back to messages

Pamela Garcia

To isscs_cases@xilinx.com

12:14 PM

Reply

Hi,

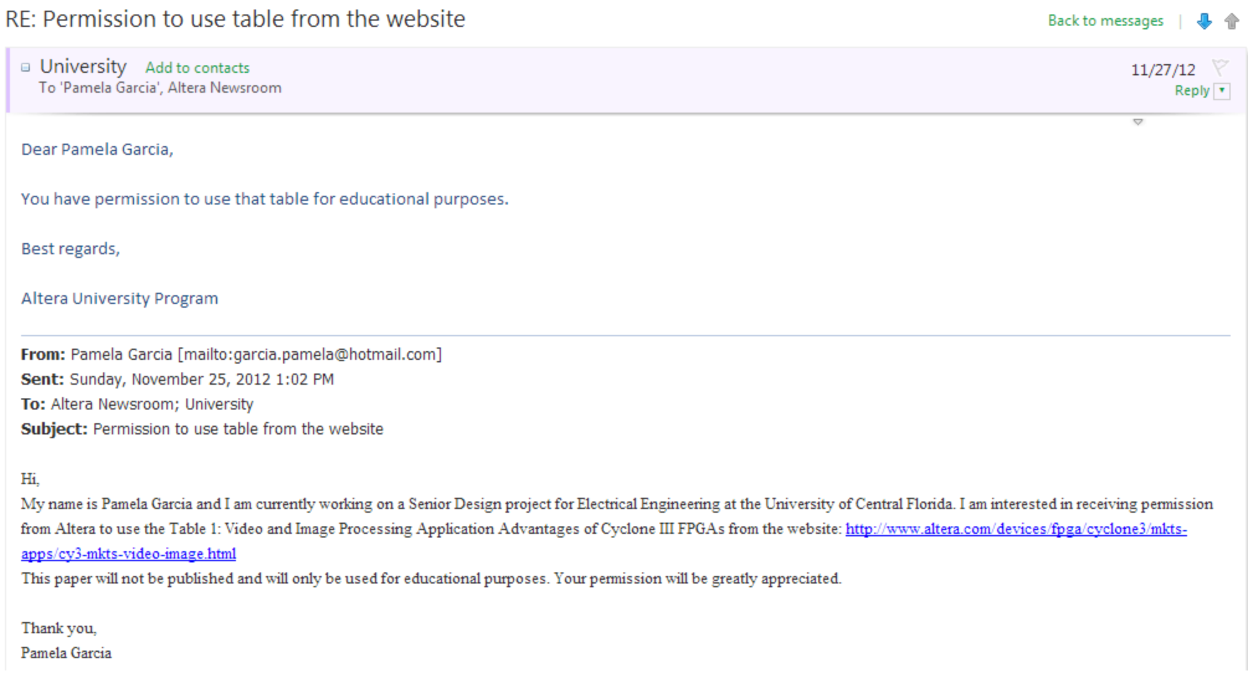
My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from Xilinx All-Programmable to use the FPGA comparison table from the website: http://www.xilinx.com/products/silicon-devices/fpga/index.htm

This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.

Thank you,

Pamela Garcia

10. Altera Table


RE: Permission to use table from the website

Back to messages

University Add to contacts

To 'Pamela Garcia', Altera Newsroom

11/27/12

Reply

Dear Pamela Garcia,

You have permission to use that table for educational purposes.

Best regards,

Altera University Program

From: Pamela Garcia [mailto:garcia.pamela@hotmail.com]
Sent: Sunday, November 25, 2012 1:02 PM
To: Altera Newsroom; University
Subject: Permission to use table from the website

Hi,

My name is Pamela Garcia and I am currently working on a Senior Design project for Electrical Engineering at the University of Central Florida. I am interested in receiving permission from Altera to use the Table 1: Video and Image Processing Application Advantages of Cyclone III FPGAs from the website: http://www.altera.com/devices/fpga/cyclone3/mkts-apps/cy3-mkts-video-image.html

This paper will not be published and will only be used for educational purposes. Your permission will be greatly appreciated.

Thank you,

Pamela Garcia

Appendix B - Works Cited

1. Kittur, Harshavardhan, and Chuanhai Bai. *FPGA BASED OBJECT TRACKING SYSTEM*. Thesis. Department of Computer Science, Lund University, n.d. N.p.: n.p., n.d. *FPGA BASED OBJECT TRACKING SYSTEM PROJECT REPORT*. Web. 2 Nov. 2012. <http://fileadmin.cs.lth.se/cs/Education/EDA385/HT11/student_doc/final_reports/objecttracking.pdf>.
2. MSP430F11x2, MSP430F12x2 Mixed Signal Microcontroller Datasheet. Texas Instruments, 22 Nov. 2012. Web. <http://www.ti.com/lit/ds/slas361d/slas361d.pdf>.

3. "Comparison Table of Secondary Batteries." Secondary (Rechargeable) Batteries â□□ Battery University. N.p., n.d. Web. 10 Nov. 2012. <http://batteryuniversity.com/learn/article/secondary_batteries>. àChemistry of Batteries
4. MSP430x1xx Family User's Guide. Texas Instrument, 22 Nov. 2012. Web. <http://www.ti.com/lit/ug/slau049f/slau049f.pdf>.
5. "Power Management." *Analog, Embedded Processing, Semiconductor Company, Texas Instruments*. N.p., n.d. Web. 20 Nov. 2012. <http://www.ti.com/lsds/ti/analog/powermanagement/power_portal.page>.
6. "Linear Regulator (LDO)." *Texas Instruments*. N.p., n.d. Web. 25 Nov. 2012. <http://www.ti.com/sc/device/TPS78001>.
7. 150mA, Low-Dropout Regulator, Ultralow-Power, IQ 500nA. Texas Instrument, Jan. 2007. Web. Nov. 22. <http://www.ti.com/product/tps78001>.
8. "Power Supply Design Considerations for Modern FPGAs." *Power Supply Design Considerations for Modern FPGAs*. N.p., n.d. Web. 7 Nov. 2012. <http://www.eetimes.com/design/programmable-logic/4015252/Power-Supply-Design-Considerations-for-Modern-FPGAs>.
9. "Choose the Right Power Supply for Your FPGA." *- Application Note*. N.p., n.d. Web. 10 Nov. 2012. <http://www.maximintegrated.com/app-notes/index.mvp/id/5447>.
10. "How Lithium-ion Batteries Work." *HowStuffWorks*. N.p., n.d. Web. 25 Nov. 2012. <http://electronics.howstuffworks.com/everyday-tech/lithium-ion-battery1.htm>.
11. "Controlling Frame Rate." Sillicon Graphics International Corp., 7 Dec. 2004. Web. 24 Nov. 2012. <http://techpubs.sgi.com/library/tpl/cgi-bin/getdoc.cgi/0650/bks/SGI_Developer/books/Perf_GetStarted/sgi_html/ch10.html>.
12. "Infrared." *Wikipedia*. Wikimedia Foundation, 20 Nov. 2011. Web. 24 Nov. 2012. <http://en.wikipedia.org/wiki/Infrared>.
13. Wheeler, Richard. "IR Webcams and Night Vision." *Calculated Images*. N.p., 28 June 2009. Web. 24 Nov. 2012. <http://www.google.com/imgres?um=1>.
14. "Coffee Table Multitouch Surface Computer." *Sypherus Labs*. N.p., n.d. Web. 13 Nov. 2012. <http://labs.sypherus.com/Coffee%20Table>.
15. Deimos. "Webcam Filter Removal." *FreeTrack Optical Head Tracking Software*. N.p., n.d. Web. 24 Nov. 2012. <http://www.free-track.net/english/hardware/filter_removal/labtec_webcam_pro.php>.
16. "LED Tutorial." *Society of Robots*. N.p., n.d. Web. 24 Nov. 2012. <http://www.societyofrobots.com/electronics_led_tutorial.shtml>.

17. *Laser/electro Optics Technology Series.* 2nd ed. Vol. 1. Waco, TX: Center for Occupational Research and Development, 1981. Print.
18. Niteesh. "VHDL Tutorial (Complete)." *AuthorStream*. N.p., n.d. Web. 21 Nov. 2012. <http://www.authorstream.com/Presentation/Niteesh-1347929-vhdl-tutorial/>.

19. Parker, Michael, Jr. "FPGA vs. DSP Design Reliability and Maintenance." *Altera White Paper* 1.1 (2007): 1-4. Altera Corporation, May 2007. Web. 24 Nov. 2012. <http://www.altera.co.jp/literature/wp/wp-01023.pdf>.

20. "Choosing FPGA or DSP for Your Application." *FPGA or DSP*. Hunt Engineering, Oct. 2012. Web. 24 Nov. 2012. <http://www.hunteng.co.uk/info/fpga-or-dsp.htm>.

21. Vikram. "FPGA or DSP." *HDfpga*. N.p., 18 Aug. 2011. Web. 24 Nov. 2012. <http://hdfpga.blogspot.com/2011/08/dsp-or-fpga-5-parameters-to-make-choice.html>.

22. Muhammad, Yasir. "Introduction to FPGA Technology." *Introduction to FPGA Technology*. FPGA Related, 12 May 2011. Web. 24 Nov. 2012. <http://www.fpgarelated.com/showarticle/17.php>.

23. "Generate Verilog and VHDL Code for FPGA and ASIC Designs." *MathWorks*. MathWorks, 2012. Web. <http://www.mathworks.com/products/hdl-coder/index.html;jsessionid=62f07e9387bd6e802d11f1a84839>.

24. "DSP Builder." *Altera News*. Altera Corporation, 2012. Web. 21 Nov. 2012. <http://www.altera.com/products/software/products/dsp/dsp-builder.html>.

25. Smith, Douglas J. "VHDL & Verilog Compared & Contrasted Plus Modeled Example Written in VHDL, Verilog and C." VeriBest Incorporated, n.d. Web. 21 Nov. 2012. <http://www.angelfire.com/in/rajesh52/verilogvhdl.html>.

26. Botros, Nazeih. *HDL Programming Fundamentals: VHDL and Verilog*. Boston, MA: Da Vinci Engineering, 2006. Print.

27. "What's New in the NI LabVIEW FPGA Module." *Developer Zone*. National Instruments Corporation, 2012. Web. 21 Nov. 2012. <http://www.ni.com/white-paper/12950/en>

28. "All Programmable FPGAs." Xilinx- All Programmable, n.d. Web. 25 Nov. 2012. <http://www.xilinx.com/products/silicon-devices/fpga/index.htm>.

29. "Low-Cost Cyclone FPGAs." *Altera News*. N.p., n.d. Web. 04 Dec. 2012. <http://www.altera.com/devices/fpga/cyclone/cyc-index.jsp>.

30. "Sunstone PCB Prototype Design | Sunstone.com." *Sunstone PCB Prototype Design | Sunstone.com*. N.p., n.d. Web. 04 Dec. 2012. <http://www.sunstone.com/pcb123/compare123.aspx>.

31. Nyasulu, Peter M. "Digital Systems Lab Manual." UCF Department of Electrical and Computer Engineering, Mar. 2010. Web. 04 Dec. 2012. Print.

32. "CCD vs. CMOS." *CCD vs. CMOS*. Teledyne Dalsa, n.d. Web. 25 Nov. 2012. <https://www.dalsa.com/corp/markets/CCD_vs_CMOS.aspx>.

33. "HP HD-3110 5.7 Megapixel Webcam, CMOS Sensor, USB 2.0 Interface #BK357AA." *EBay*. N.p., n.d. Web. 02 Dec. 2012. <http://www.ebay.com/itm/HP-HD-3110-5-7-Megapixel-Webcam-CMOS-Sensor-USB-2-0-Interface-BK357AA-/121005918631?pt=PCA_Video_Conferencing_Webcams>.

34. "USB Digital PC Web Camera with Microphone CMOS VGA Sensor." *EBay*. N.p., n.d. Web. 02 Dec. 2012. <http://www.ebay.com/itm/USB-Digital-PC-Web-Camera-with-Microphone-CMOS-VGA-sensor-/251144045621?pt=PCA_Video_Conferencing_Webcams>.
35. "Home | Product Categories | Batteries | PRT-10161." *USB LiPoly Charger*. Sparkfun, n.d. Web. 02 Dec. 2012. <https://www.sparkfun.com/products/10161>.
36. *Single-Chip, Multiple-Message Voice Record/Playback Device*. Nuvoton, 2 Dec. 2012. Web. <http://www.sparkfun.com/datasheets/BreakoutBoards/BOB-09579-ISD1900.pdf>.
37. Brown, Brian. "Advanced C." *Advanced C, Part 3 of 3*. N.p., 02 Jan. 1999. Web. 11 Oct. 2012. <http://gd.tuwien.ac.at/languages/c/programming-bbrown/advcw3.htm>.
38. "The Basics:Writing Windows Drivers." *The Basics:Getting Started Writing Windows Drivers*. Open System Resources, Inc., 04 Apr. 2004. Web. 25 Oct. 2012. <http://www.osronline.com/article.cfm?article=20>.
39. Khanduja, Jaideep. "Testing Environment." *Quality Assurance and Project Management What Is a Testing Environment for Software Testing Comments*. IT Knowledge, 12 Sept. 2008. Web. 10 Nov. 2012. <http://itknowledgeexchange.techtarget.com/quality-assurance/what-is-a-testing-environment-for-software-testing/>.
40. Orwick, Penny, and Guy Smith. *Developing Drivers with the Windows Driver Foundation*. Redmond, WA: Microsoft, 2007. Print.

Appendix C- Table of Tables

Appendix D - Table of Figures