HAND ACTIVATED NON-OBSTRUCTIVE SYSTEM (H.A.N.S.)

Richard Hathcoat / Isabel Carrillo / Kenneth Garmon / Tyler Kates

Dr. Ana Goulart

Electronic Systems Engineering Technology

Texas A&M University

EnginiumDesigns@Outlook.com

1 Introduction

The National Aeronautics and Space Administration (NASA) agency spends millions of dollars on various robotic platforms, and this drives the need for a cost efficient robotic platform. An example of a cost-efficient robotic platform is one that uses a non-traditional form of control, such as hand gestures. Furthermore, NASA is interested in having direct control of the robot's movements, as well as the ability to trigger preloaded autonomous commands. To help NASA evaluate this new technology, a prototype called Hand Activated Non-Obstructive System (H.A.N.S.) is being developed by a team of undergraduate students, as part of their Capstone Design course in the Electronic Systems Engineering Technology (ESET) program at Texas A&M University.

This paper describes the project management tools utilized and the design of the H.A.N.S. prototype that will use the iRobot Create platform as the target system. NASA seeks to evaluate and investigate this cost-efficient robotic platform that can perform similar tasks as a costly robotic platform and be taken on space missions. This prototype will only be used for demonstration and a proof of concept and will control the iRobot Create through the use of hand gestures in order to demonstrate the use of a nontraditional form of control and investigate its limitations. Additionally, the H.A.N.S. prototype will investigate any possible limitations to the hand gesture methodology.

1.1 Background

The H.A.N.S. project is scheduled to kick off on January 13, 2014 and the final prototype is scheduled for completion on May 3, 2014. The H.A.N.S. project is sponsored by the Electronic Systems Engineering Technology (ESET) program at Texas A&M University and this paper serves as an example for the types of capstone project proposals for undergraduate senior projects. The team, Enginium Designs, is comprised of four members with equally divided responsibilities. Richard Hathcoat is the Project Manager and Test Engineer, Isabel Carrillo is the Hardware Engineer, Tyler Kates is the Software Engineer, and Kenneth Garmon is the Systems Integration Engineer for the project. This capstone project is required to have a clearly designed and completely understood technical challenge. The challenge involves a requirement for hardware and software design, development, and test, as well as a requirement for system integration and system testing. The team is designing a prototype that will utilize an iRobot Create and the on-board technology available to integrate with an embedded controller. This system will communicate wirelessly to a PCB mounted on the hand that will interpret the gesture controls. These gesture controls can either trigger pre-programmed tasks on the iRobot Create or control its movements in real-time.

1.2 Proposal Structure

Enginium Design's proposal for the H.A.N.S. will be structured based on the following five major sections: 2 Project Scope, 3 Statement of Work, 4 Communication, and 5 Conclusion. The proposal will include descriptions of all technical aspects of the project as well as project management tools used to ensure the final prototype is on-time and on-budget upon completion.

2 Project Scope

The primary objective of this project is to build a fully functional prototype for NASA. This prototype will demonstrate the ability to control a remote robotic platform using hand gestures. The tasks that the robotic platform will perform include a number of preset demos as well as a setting that allows the user to directly control the platform's movements. The H.A.N.S. project is broken down into the following phases:

Research

This section is divided into three parts: Hardware, Software, and Communication. In the research phase, each team member will research the various components, software suites, and methodologies that are available for use in this project.

Design

The design phase is split into two parts: Hardware and Software. The hardware section includes choosing the components we will use for the prototype, three revisions of both the schematic and board layout, and three revisions of the enclosure. The software design for the robot controller consists of two revisions of the hierarchy chart and flow chart, a state machine diagram, software design review, and frame format for communications. The software design for the control device also includes two revisions of the hierarchy chart and flow chart, a state machine diagram, and a software design review. These designs will be used to produce a fully integrated prototype in the implementation phase.

Implementation

Implementation is split into two sections as well: Hardware and Software. The hardware consists of obtaining all the parts, populating the PCB, debugging the PCB, and building the wearable enclosure. The software implementation is broken down into two more sections: the robot controller, and the control device. Implementation for both of these sections includes three revisions of the code for both the robot controller and the control device which will include coding for the automated tasks and real-time control.

Testing

The test phase is split into three parts: Hardware, Software, and Communications. Hardware consists of the bulk of the testing which includes three rounds of testing to test the three revisions of the board. The hardware tests include testing power, sensor outputs, grounding, solder joints, and trace continuity. Software testing will test code coverage and functionality on both the embedded controller connected to the robotic platform and the microcontroller of the control device. The communications portion will test the wireless communications between the robotic platform and the control device. Wired communications between the controllers and their peripherals will also be tested.

Documentation

The documentation is split into two sections: Design Documentation and Sponsor Documentation. The design documentation includes the preparation and execution of CDR and TAT meetings, a bill of materials, and final documentation. The sponsor documentation includes a user guide and technical reference.

Close-Out

Finally, the close-out of project will include the final presentation and clean up. After all of these phases are complete, the entire project scope will have been covered.

2.1 Assumptions

Enginium Designs will create a prototype that will be used on a hard flat surface such as linoleum tile. There will be no obstruction between the control device and the robotic platform. Any monetary requirement for the project will be supplied by the sponsor. Enginium Designs is assuming that all of its members will be able to contribute to the design and development of the prototype.

2.2 Conceptual Design

H.A.N.S. uses the iRobot Create as the target system, which is powered by a rechargeable 14.4V DC battery. The iRobot has an onboard controller that will be integrated with an embedded controller, the BeagleBone Black (BBB). The embedded controller will receive control signals from the gesture control device. H.A.N.S. uses a gesture control device to wirelessly communicate to the embedded controller. Data transmission between the embedded controller and gesture control device will be achieved through Zigbee communication.

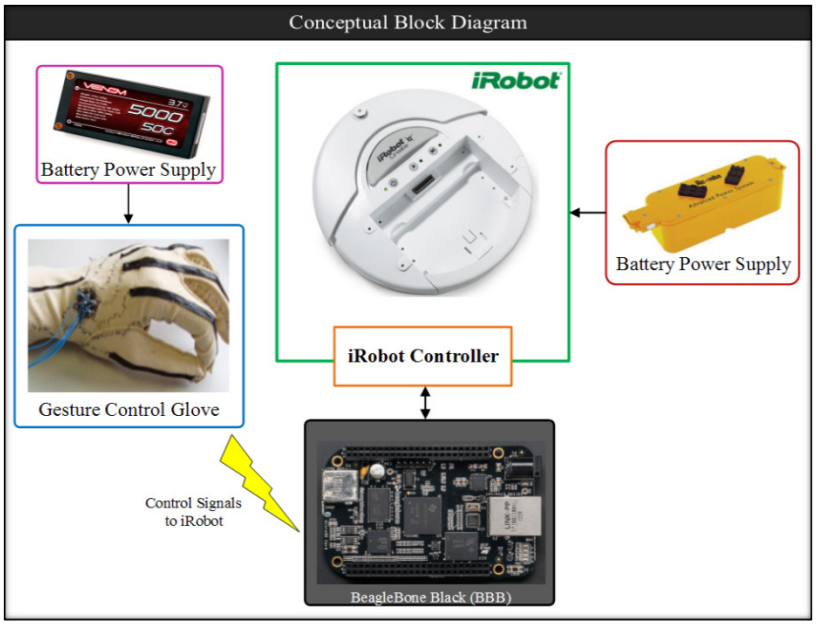

Figure 1. Conceptual Diagram

Zigbee was chosen because it has a range of more than 20 meters and has a reliable connection. The gesture control device is a wearable device that has sensors and a rechargeable battery power supply. The sensors are attached to the tips of each finger in the glove. The sensors will be used to trigger commands and directly control the iRobot. An accelerometer is attached to the gesture control device to allow for more commands and better control of the robots movements. Figure 1 shows the basic conceptual design for the H.A.N.S. project.

2.3 Test Matrix

The test matrix shown in Table 1 has been constructed to assist Enginium Designs in verifying that all functional requirements have been met by the final prototype. The test matrix shown in Table 1 has been constructed to assist Enginium Designs in verifying that all functional requirements have been met by the final prototype. The rows indicate each test that will be performed in accordance with the set functional requirements. The columns indicate the functional requirements agreed upon by the customer and Enginium Designs Team. An 'X' is placed in the grid to show which test corresponds to each functional requirement.
The functional requirements specified in the test matrix will develop into performance specifications that can be verified through a test plan. After a test plan is created, and the performance specifications are tested, a test report will be generated.

Table 1. Test Matrix

Test Names	Functional Requirements: Wireless Communication	Control Methodology	Trigger Automated Tasks	Real-Time Control	Operation Time	Failure Mode
Wireless Configuration Test	X					
Wireless Coverage Test	X					X
Low Power Mode Test					X	
Real Time Control Test				X		
Demo Tasks Test			X			
Sensor Test		X	X	X		
Gesture Control Operation Time					X	

Wireless Configuration Test

The communication between the control device and the robotic platform will be tested by sending data to and from each device. The test data will have to be programmed into the MSP430 microcontroller of the control device and observed through the output of the BeagleBone Black. A data rate of 100 kbps must be achieved. Also, a Linux Operating System based controller must be used to control the Hand Activated Non-Obstructive System (H.A.N.S.)

Wireless Coverage Test

The operating range between the control device and the robotic platform will be tested so that a range of 60 feet (~20 meters) is achievable. An LED indicator on the gesture control glove will indicate when the device is out of range. This failure mode will test the LEDs' responsiveness and accuracy of the communication signal.

Low Power Mode Test

A Low Power Mode test will be performed on the H.A.N.S. to understand how much energy is consumed in the low power modes and how it will affect the operation time. The low power mode must be tested to improve the efficiency of the circuit.

Real Time Control Test

Measure the time it takes for data and commands to be transferred through the entirety of the system. The response time should be less than or equal to 1 second. The H.A.N.S. will be tested to make sure we have direct control of the iRobot Create movements. The iRobot Create should be able to move forward, backward, clockwise, and counterclockwise. These movements will be triggered by the control device through our gesture sensing circuits.

Demo Tasks Test

Commands from the control device to the robotic platform should be able to trigger a number of preprogrammed automated tasks. The gesture control device will be able to initialize and carry out pre-programmed automated tasks for the iRobot Create.

Sensor Test

The sensor circuitry in the H.A.N.S. includes hall effect sensors, flex sensors, and an accelerometer. The sensors circuitry will need to be tested for sensitivity and responsiveness.

Each sensor's inputs will need to be observed and conditioned properly. The hall effect sensor's range should be small enough so that there is minimal interference with the other sensors. The sensors must operate in a non-traditional methodology to control the robotic platform.

Gesture Control Operation Time Test

The operation time of the control device will be tested by putting the H.A.N.S. through continuous use until the battery fully depletes. The gesture control device will need to be able to operate for at least 4 hours of use. The battery attached to the gesture control device must be rechargeable and enclosed in the glove.

3.0 Statement of Work

Figure 2 provides a high level functional overview of how the entire project is structured. There are two major sections. The first stems from the BeagleBone Black and connects to the iRobot Create. The second is based at the MSP430F247 Microcontroller. The MSP430F247 will connect to the sensing circuitry of the glove and process the gestures given by the user. The two sections are connected wirelessly through two Xbee modules attached to each controller. The arrows indicate whether each component is input, output, or has a bidirectional interface. The text within the arrows represents the communications protocol or interface type. Figure 3 illustrates two main subsystems that will interface together to form the H.A.N.S.

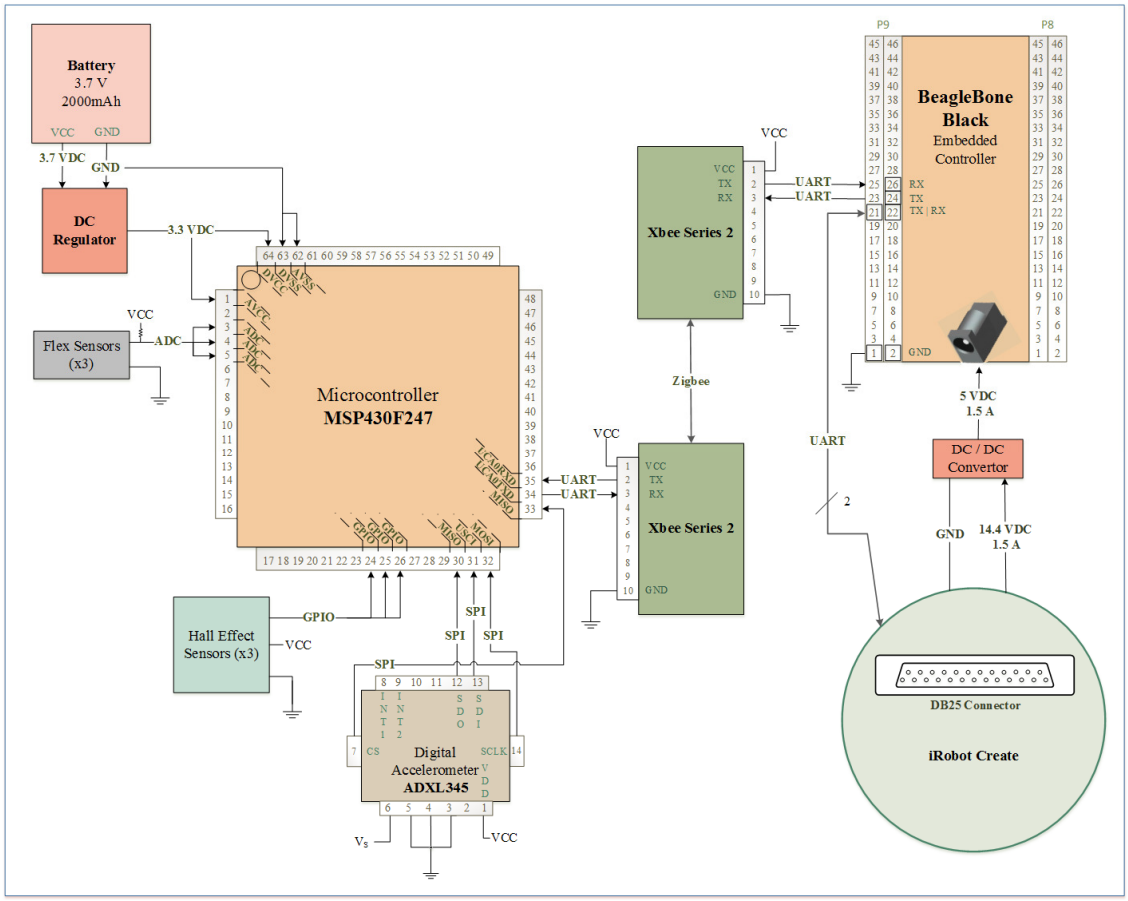

Figure 2. High Level Design

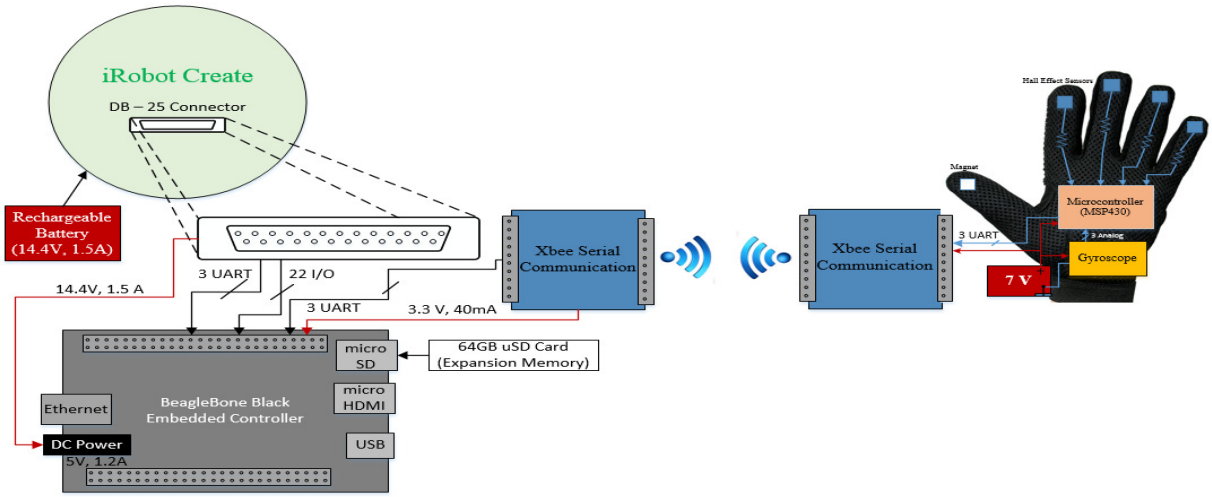

Figure 3. Functional Block Diagram

Microcontroller Power

The power for the control device will come from a 3.7V, 2000maH rechargeable battery. The 64-pin MSP430F247 (MSP430) microcontroller has one pin for analog power and one pin for digital power. Both of these pins will be connected to the positive output of the battery. The two grounds labeled DVSS are connected to the ground pin of the battery. The MSP430 operates with a power supply range of 1.8V to 3.6V. It has three power modes: active, standby, and off (RAM Retention). In active mode the MSP430 uses 230uA at 1MHz, 2.2V. In standby mode 0.5uA are required. In the RAM retention mode only 0.1uA are required. While the system is in use, the microcontroller will be mostly in active mode.

Accelerometer

The ADXL345 is a digital 3-axis accelerometer. It can measure accelerations up to ± 16g. It is a low power device that has a power supply range of 2V to 3.6V and a current draw of 40uA in measurement mode. The ADXL345 will be used to determine the orientation of the user's hand. Separate gestures will be accessed depending on whether palm of the hand is facing down or up. Figure 4 shows the connection of the ADXL345 to the MSP430. The ADXL345 uses either SPI or I2C to transfer digital data. The MSP430 is capable of both of these protocols, but SPI was decided upon for ease of use. The accelerometer will be powered by the battery shown. Pins 12 and 13 of the ADXL345 are the SPI input and output pins that connect to pins 30 and 31 of the MSP430.

Furthermore, there are sufficient pins on the microcontroller to also have the capability to utilize the ADXL335 which is also a 3-axis accelerometer but it is analog. Thus, requires three ADCs (analog to digital) pins on the microcontroller to read information about the movements in the x-axis, y-axis, and z-axis.

Sensing Circuitry

The main function of the MSP430 on the control device is to interpret the hand gestures of the user. This will be done with a combination of two types of sensors: the flex sensor, and the Hall Effect sensor. Flex sensors are flexible sensors that are made of a material that changes resistance when it is bent. The flex sensor will be placed along the length of each finger on the glove. Another resistor will be placed in series with the flex sensor to create a voltage divider

circuit. A voltage will be applied to the voltage dividing circuit from the battery. Therefore, the voltage across the flex sensor will change according to the degree the sensor is bent. This change in voltage will be measured through pins 3, 4, and 5 on the MSP430. Each one of these pins has an ADC which will allow the voltage to be seen digitally in the code. Hall Effect sensors essentially work as a transistor switch; however, the gate on the transistor is powered on by a magnetic field. Each one of the four Hall Effect sensors will be placed on the tip of each finger on the glove. A magnet will be placed on the thumb. Therefore, when the thumb comes in close vicinity with the one of the Hall Effect sensors attached to the fingers, the switch will go on. This gesture will be used for specific inputs. Each finger will have a separate command. The Hall Effect sensors will be connected to pins 24, 25, and 26 which are GPIO pins. Since the Hall Effect sensor is essentially a switch, a digital input will suffice.

Robotic Platform

The robotic platform is controlled through the BeagleBone Black embedded controller. The iRobot Create is powered by a 14.4V, 1.5A battery. This battery will power the entirety of the robotic platform. This includes the BeagleBone Black and the Xbee module. The BeagleBone Black has an onboard voltage regulator which will regulate the battery's 14.4V down to 5V. The iRobot Create has a DIB-25 interface port. This port contains all the pins that will interface with the BeagleBone Black. Pin 10 of the DIB-25 is the 14.4V power supply that will connect to the 5V DC barrel connector on the BeagleBone Black. Pins 1 and 2 on the DIB-25 port are the Serial UART pins that will connect to pins 21 and 22 on the BeagleBone Black. The ground pin of the DIB-25 is connected to pin 1 of the BeagleBone Black.

Xbee to BeagleBone Black

The BeagleBone Black receives information from the control device through an Xbee module. The Xbee communicates through Serial UART. Pins 2 and 3 of the Xbee are the output and input of the serial communication. These pins connect to pin 26 and 24 of the BeagleBone Black. Power and ground are supplied from pins 3 and 1 of the BeagleBone Black to pins 1 and 10.

MSP430 to Xbee

The control device communicates to the robotic platform through an Xbee module. The Xbee communicates through Serial UART. Pins 2 and 3 of the Xbee will be connected to pins 35 and 34 of the MSP430. Pins 1 and 10 of the Xbee module are connected to voltage and ground of the battery. The Xbee requires a voltage between 2.1V to 3.6V. The MSP430 will output data through the UART pin 2 and receive input from pin 1. The Xbee UART input is pin 3, and the UART output is pin 2.

3.1 Work to be Performed

The work required for the completion of the project was determined through the use of the Work Breakdown Structure (WBS) and the Responsibility Assignment Matrix (RAM) project management tools.

Work Breakdown Structure

The WBS is an effective tool for providing a good idea for the project scope and the work necessary to complete a project. In addition, the WBS is used as a graphical representation of the overall project flow and the different stages of the project. These stages can also be referred to as levels of a project that breakdown into the project, phases, activities, tasks, subtasks, and work

packages. Level 0 represents the project and is broken down into project phases. Phases identify high level stages and major components of the project that are then broken down into activities. The process continues by breaking down activities into tasks and then into subtasks with more detail at each level. Furthermore, a work package is always the lowest level block in its branch. These work packages are later utilized in the RAM to assign team members responsible for each and every work package as well as the man-hours required to complete each work package. The H.A.N.S. is broken down into six project phases shown in Figure 4.

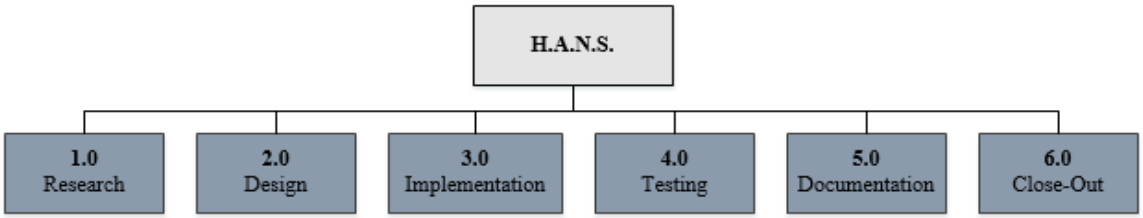

Figure 4. WBS – H.A.N.S. Phases

The first project phase is the research phase. This phase is necessary because it allows all team members to investigate different methodologies and technologies that can be used for the development of the H.A.N.S. Since all team members are not familiar with the project concepts this phase allows that various approaches and different solutions are explored for the project. Team members will take time to experiment with tutorials, new software, and the different communication protocols necessary to interface the H.A.N.S. subsystems. At the conclusion of this phase the team will select which technologies and methodologies are best fitting for the success of the H.A.N.S. project. The design phase is the second phase of the H.A.N.S. and definitely crucial to the success of the project. This phase contains two major activities, the hardware design and software design. The hardware design includes four subtasks: component selection, schematic design, PCB layout, and the design of the control glove. The subtasks are additional specific tasks within the hardware tasks such as the specific hardware components and the alpha, beta, and final versions for the schematic and PCB layout. The software design consists of two subtasks: the robot control software and the control device software. The subtasks for the software design activity similarly consist of alpha, beta, and final versions of code for the embedded controller and the control device microcontroller. The implementation phase is similar to the design phase as it is the implementation of what was designed in phase 2 so this phase also includes a hardware and software activity. The hardware tasks now consist of the procurement of hardware components, ordering and populating the PCB, creating the wearable control device, and debugging as needed. The software tasks consist of writing and debugging code for both the embedded controller and the microcontroller and revising it as needed. The embedded controller will control the iRobot Create and the microcontroller will process the information obtained from the hand gestures through the sensors. The fourth phase is the test phase and its primary purpose is to verify the design meets the performance requirements set by the customer. For this reason, most if not all work packages in the test phase correspond to a performance requirement. Testing will allow Enginium Designs to verify the H.A.N.S. functionality and performance meets or exceeds expectations. The fifth phase is the documentation phase and will be an ongoing process through the project. The documentation phase includes tasks such as preparing for weekly meetings, a mid-semester presentation, bill of materials, and final documentation. The last phase of the project is close-out. This phase will begin towards the end of the project timeline, after the functional prototype is fully complete and

documented. The activities within this phase include the final preparation and clean up. Final preparation includes preparation for the final presentation and the final presentation itself.

Responsibility Assignment Matrix

The Responsibility Assignment Matrix (RAM) created by Enginium Designs is used to delegate responsibility for individual work packages to certain team members. There are four primary roles that a team member can have within a work package. The person who is responsible for ensuring that the entire work package is completed on time is the leader. A participant will aid in the completion of a work package, but is not responsible for its completion. The input role is responsible for providing any theoretical or technical information that might aid in completion of the work package. Finally, the reviewer evaluates the work that was performed on the work package and determines whether it fulfills the requirements to complete the work package. Every work package in the RAM is assigned an estimated number of hours and duration in days. The hours represent the total work hours from every team member it takes to complete the work package. The number of days represents how many days the number of hours is spread between.

3.2 Precedence Diagram

Enginium Designs used a precedence diagram known as a Network Logic Diagram (NLD), in order to aid in organizing work packages onto a timeline to determine the total amount of time required to complete a project.

NLD General Description

The work packages in the Network Logic Diagram are organized and linked together based on the dependence they have with each other. If a work package needs to be completed before another work package can begin, the first work package is placed to the left of the dependent work package and then they are connected by a line.

A work package, as represented in the Network Logic Diagram, displays a duration value, in working days, that is used to calculate the earliest start, earliest finish, latest start, latest finish, and slack. The slack, also referred to as float, defines the amount of days that exist between the early start and late start days. The earliest start value refers to the first possible day that the work package can begin. This value is determined by the time required to complete the preceding work packages. The earliest finish value refers to the earliest possible day that the work package can be completed. This value is calculated by adding the earliest start date with the duration of the work package. The latest start value refers to the last day that the work package can be started without causing any delays in the project. If the work package is started after the latest start value, the only way to not cause a delay in the project is to finish the work package in less time than the duration value indicates. Latest finish refers to the last possible day that the work package can be completed without causing a delay in the project. The latest finish value is calculated by adding the duration value of the work package to the latest start value. To calculate these values, after they have been organized as described above, you place a zero in the early start block of the first work packages that do not need previous work packages to be complete before they begin. The early finish value is then calculated by adding the early start date with the duration value of the work package.

Finally, the final box to fill on each work package is the slack. To calculate the slack value, you can either subtract the late start from the early start, or subtract the late finish from the early finish. The calculated slack value can then be placed in the slack box. The slack indicates the

amount of days between the early start and the late start days. The slack days can be used to reallocate manpower efforts during the project if it is needed. A person that has slack can be used to help with another work package that may be falling behind without increasing the overall project duration. When a Network Logic Diagram is completed, a path that strings from the start of the project to the end of the project can be found with zero slack days. This path is referenced as the critical path and each work package on this path must be completed on time or the project may be delayed. The critical path for the H.A.N.S. project reveals that the duration of the project is 95 work days and mainly consists of the hardware portion of the project (schematic generation, PCB layout, ordering, debugging of the three board revisions), the testing phases, and finally the final documentation and presentation.

Critical Path

The critical path for the H.A.N.S. project reveals that the duration of the project is 95 work days. The critical path for this project consists mostly of hardware, testing of hardware, and documentation. This gives a sizeable amount of slack to the software and communication paths.

3.3 Task Schedule

Gantt charts are useful at graphically presenting the project tasks schedule on a timeline. The overall project schedule for the six main stages of the project is shown in Figure 5. The Gantt chart makes it possible to determine the duration of each and every stage of the H.A.N.S. along with the start and end dates. It is worth noting that the dates used for the start and end dates were obtained from the NLD which obtained its contents from the WBS. The six phases include: Research, Design, Implementation, Test, Documentation, and Close-Out. A green diamond indicates each specific phase has started and a red star indicates each specific phase has ended. Each of the six project phases are broken down into individual tasks, illustrating each start and finish date. The early start and early finish dates used for each task were taken from the Network Logic Diagram. Also, each task represents the corresponding work package from the Network Logic Diagram. The NLD obtained its contents from the WBS. A green diamond indicates each specific task has started and a red star indicates each specific task has ended.

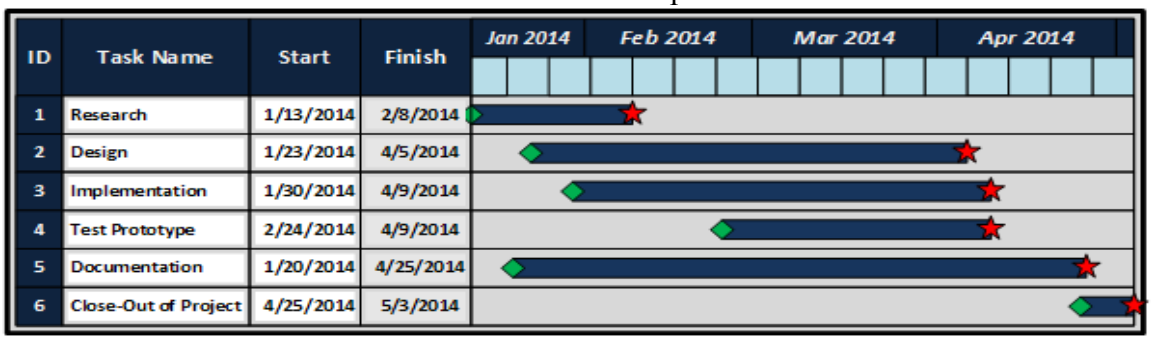

Figure 5. Project Overview Gantt Chart

3.4 Deliverables

Deliverables are defined as a tangible item that will be delivered from Enginium Designs to our client to ensure the H.A.N.S meets the criteria for the functional requirements. The deliverables will show the progress and development of the H.A.N.S. Figure 6 starts from the project start date of January 13, 2014, to the end date of May 3, 2014. This timeline graphically represents each deliverable with its specific delivery date, separating each deliverable into categories. Each category is distinguished with a color which can be followed using the color code map.

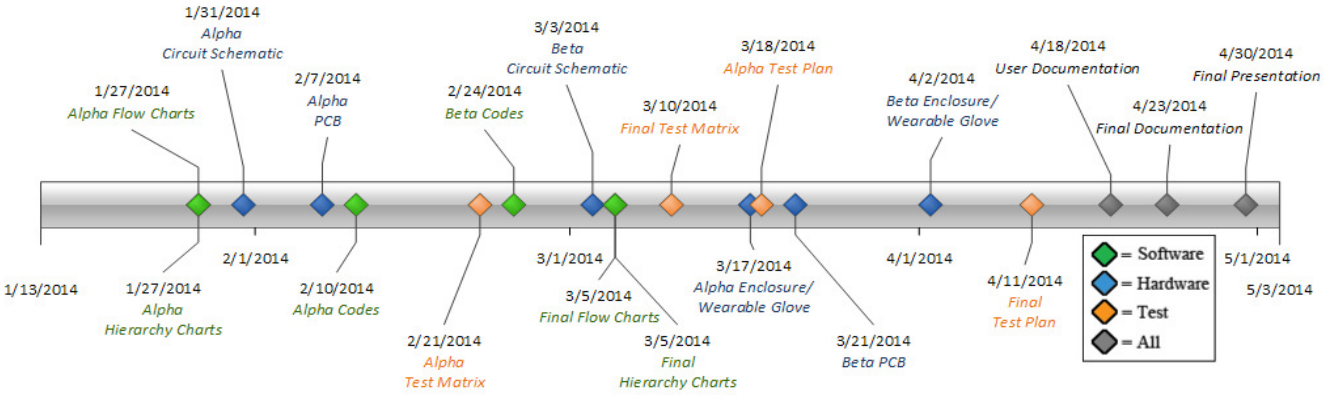

Figure 6. Deliverables Timeline

3.5 Milestones

Milestones are major events that demonstrate significant progress to the customer such as the completion of sections, hardware, or software, and allows for the advancement of the project. The milestones in this timeline are organized using colors to separate the hardware, software, and documentation. The timeline of the major milestones for this project are shown in Figure 7.

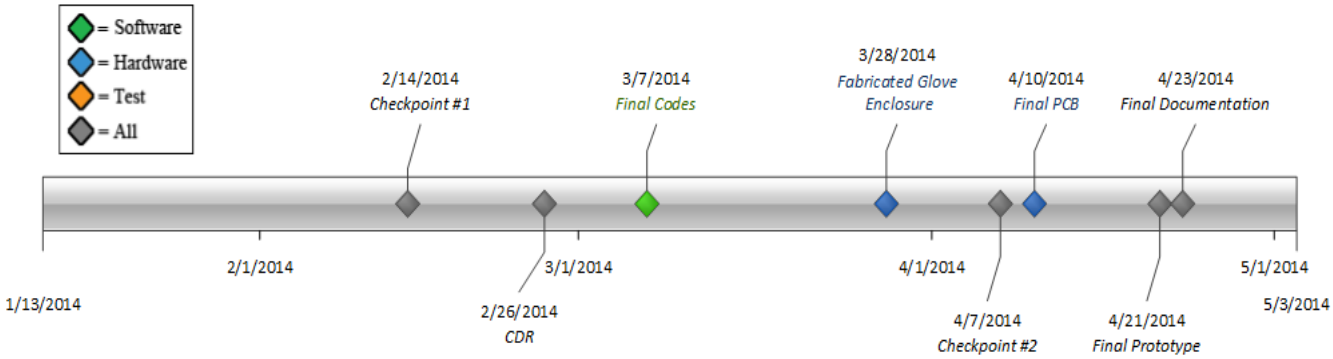

Figure 7. Milestone Timeline

4 Communication

4.1 Communication Packets

Figure 8 shows the two different packet types used in communication between the two controllers. The first is the demo packet type. Sending a mode byte containing the character ‘d’ puts the system into the demo state. A value of 1 through 6 can be sent in the number byte. This value selects which demo will be performed. The second packet type is for real time control. Sending a mode byte of ‘r’ puts the system into the real time control state. The direction byte designates forward (‘f’), back (‘b’), or standstill (‘s’). The speed byte denotes the speed based on a 9 variable speed system. The angle byte denotes the angle on a similar 9 angle system.

Mode	Number
d	1
d	2
d	3
d	4
d	5
d	6

Mode	Direction	Speed	Angle
r	f/b/s	1-9	1-9

Figure 8. Communication Packets

4.2 Communication Timing

There are three different types of communication predicted to occur during operation of the system. The first, shown in figure 9, is successful communication. This is where the glove sends the packet and the platform responds with an acknowledgement before 2ms after the packet was sent. The platform then performs the demo specified in the packet after sending the acknowledgement.

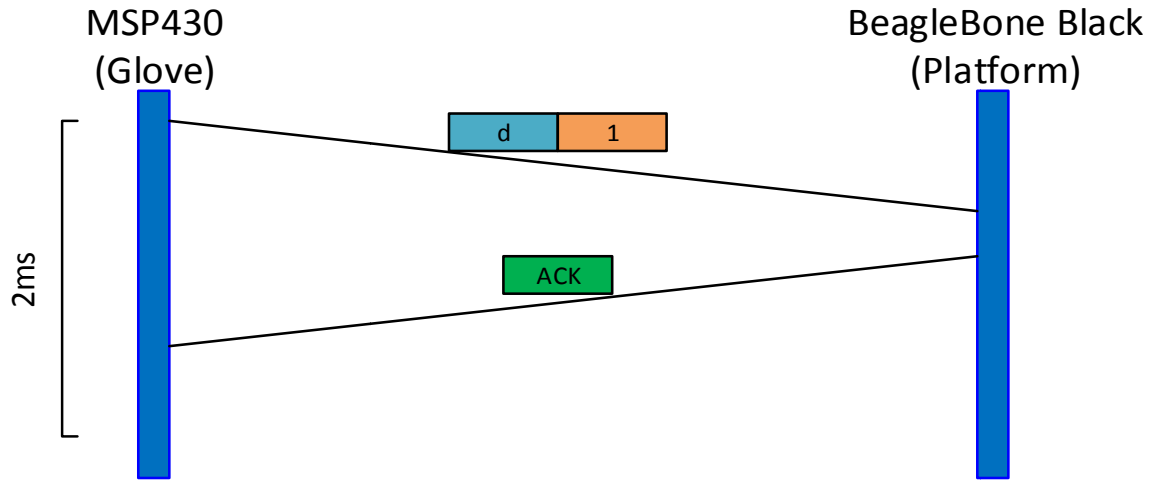

Figure 9. Successful Communication

If communication is initially successful within the 2ms window, there are two more states that communication can be reestablished. The first is "late acknowledgement." This is where the glove receives no acknowledgement after 2ms of sending the packet. The glove proceeds to resend the packet 5 times with intervals of 1ms. The platform receives the packet and sends the acknowledgement and proceeds to perform the demo specified in the packet. This can be seen in Figure 10.

The last state is leads to failure mode. This is where the glove sends the packet, but no acknowledgement is received after 2ms. The packet is then resent 5 times with 1ms intervals. After 5ms the glove is put into failure mode. Failure mode consists of the glove sending a verification byte every 2ms until communication is reestablished. If communication is reestablished, the platform will not receive the original packet and will begin polling for a new command. When communication is reestablished, the glove will exit failure mode and wait for a new command to send to the platform.

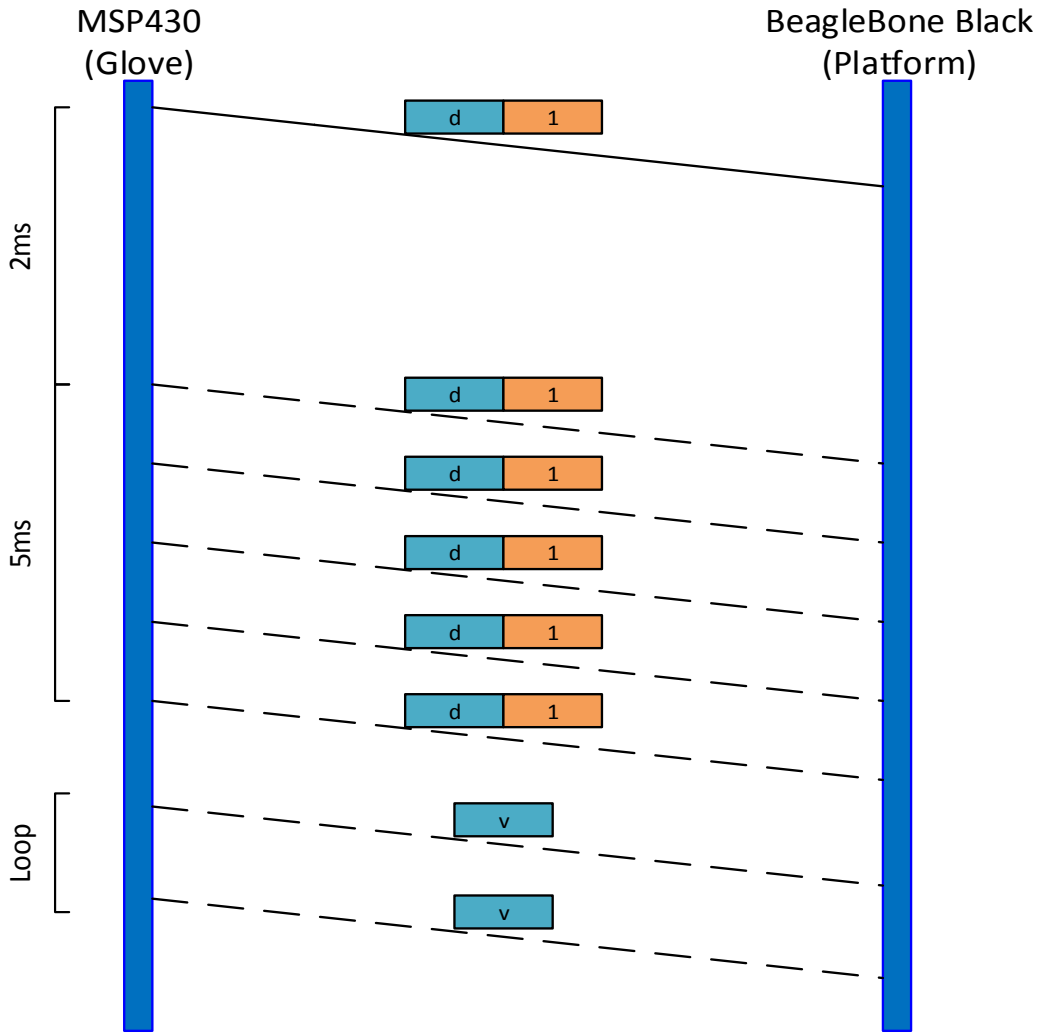

Figure 10. Late Acknowledgement and Failure Mode

5 Conclusion

As of today, February 24, 2014, the H.A.N.S. project has shown significant progress with being on schedule and under budget. In agreement with the set schedule in the deliverable timeline, Enginium Designs have completed the Alpha Flow Charts, Alpha Hierarchy Charts, Alpha Circuit Schematic, Alpha PCB Layout, and Alpha Test Matrix. Thus, completing all the Alpha deliverables in the project design and moving forward to the Beta design. At this time, the H.A.N.S. software has been successful with communication from the gesture control glove to the iRobot. Three demonstrations have been performed and tested successfully with the use of the flex and hall effect sensors operating in conjunction with each other. Further testing and coding is needed to implement the operation of the accelerometer. The accelerometer will enable multiple directions for the glove to have three more demonstrations with the use of real time control. For the status of the H.A.N.S. hardware, Enginium Designs is currently populating and testing the Alpha PCB. The Alpha PCB Layout can be seen in Figure 11. Once the board is fully tested against the functional requirements, Enginium Designs will correct any problems for the future Beta PCB. Enginium Designs plans to have a fully functional prototype and required documentation by May 3, 2014.

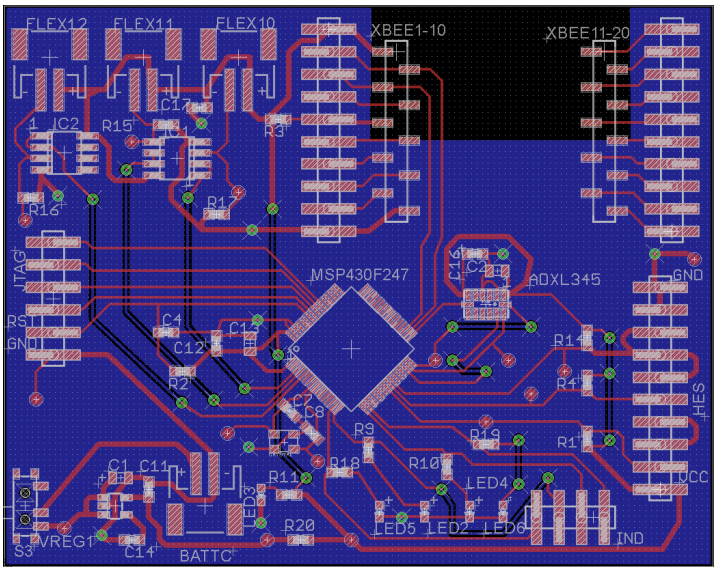

Figure 11. PCB Layout