



Simon Fraser University,
8888 University Dr.
Burnaby, BC Canada
Email: telemedix-440@sfu.ca

February 16, 2011

Dr. Andrew Rawicz
School of Engineering Science
8888 University Drive
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 440 design specification for a remotely controlled home care robot

Dear Dr. Rawicz

Enclosed is our Design Specification for a Controlled Home Assistive Device, which describes a remotely controlled home care robot. Telemedix's goal is to design a robot that will be controlled over Wi-Fi, to aid the elderly and/or people with disabilities, and those who live in remote locations.

This design specification describes in detail how we plan to build our product. It also discusses the testing methodology that will be used. This document will later be used as a guide for the design and development of our device.

Telemedix is comprised of six senior engineering students from the systems, electrical and computing concentrations, with a broad range of skills and specialties. If you have any questions or concerns regarding our functional specification or product, please contact us by email at telemedix-440@sfu.ca.

Sincerely,

A handwritten signature in blue ink that reads "Robin Mahony". The signature is written over a horizontal line.

Robin Mahony
CEO
Telemedix

Enclosure: Design Specification for a Controlled Home Assistive Device



Design Specification
for a Controlled Home Assistive Device



Design Specification for a Controlled Home Assistive Device

| | | | | | |
|---------------------|----------------------|-----------------------|------------------------|---------------------|---------------------|
| Robin Mahony | Adrian Mocanu | Graham Holland | Vojta Prochazka | Rohit Thomas | Ravi Panchal |
|---------------------|----------------------|-----------------------|------------------------|---------------------|---------------------|

| | | | | | |
|-----|-----|-----|-----------|-------------------|-----------------|
| CEO | CFO | CTO | Secretary | Managing Director | Director of R&D |
|-----|-----|-----|-----------|-------------------|-----------------|

Contact Person:

Robin Mahony
telemedix-440@sfu.ca

Submitted To :

Dr. Andrew Rawicz - ENSC 440
Mike Sjoerdsma - ENSC 305
School of Engineering Science
Simon Fraser University

Issued Date:

March 13, 2011

Executive Summary

This design document will only focus on those aspects of our device that we rated as critical priority (P1) in the Functional Specification. This is due to the fact that these are the only requirements that we intend to implement. This design specification will discuss all the different components of our system, and how we plan to implement these components. The major components of our system can be grouped according to the hardware they relate to. We have chosen this hardware with a mind to balance performance, compatibility ease of use, while still remaining within our budget.

For the motors, wheels and base of our robotic device, we opted to use an electric wheelchair. This will allow us to modify an already working mechanical system that can support the weight requirements. For the mainboard that will be the center of control, we chose the BeagleBoard. This board has the necessary hardware, adequate processing power, and is relatively inexpensive. It will run software written in C/C++, that will handle its many duties including: capturing the audio/video feed and posting it to a URL, receiving the controlling data from the controlling software and sending the necessary commands to the motor controller and receiving and playing the audio feed sent from the controlling software. For the audio/video capabilities we required of our robotic device, we will use the Logitech Quickcam Pro 9000. This will allow us to record high quality video and audio. For playing the audio that is received from our user, we will use ordinary, battery powered computer speakers as they are inexpensive and the batteries will allow us to reduce the strain on our other main power sources. For the pan and tilt functionalities of our camera, we chose a Lynxmotion pan and tilt kit. This was chosen because the kit will greatly simplify the work to be done, and it is very affordable. For interfacing from the BeagleBoard to the motors (wheelchair and pan tilt), we will use the Arduino Mega 2560. It will receive instructions from the BeagleBoard, and then through the use of an external circuit, will control the wheelchair and pan tilt motors. It was chosen mainly for its flexibility, ease of use, and because of its extensive online support. Lastly, an iPhone will be used as the controller of our robotic device. Using an application written in Objective C, it will provide the user with an interface that will be capable of controlling the movement of the camera and robotic device, will send audio from the user to the robotic device, and will stream the webcam's audio/video feed live to the user.

All the different components of our system will be tested thoroughly, by adhering to the unit and integration test plans we have discussed in this document. The design and implementation of these features and procedures will be accomplished and integrated into our product with an expected date of completion of April 10.

Table of Contents

| | |
|---------------------------------------|-----|
| Executive Summary | 1 |
| Table of Contents | i |
| List of Figures | iv |
| List of Tables | vi |
| Glossary | vii |
| 1 Introduction | 1 |
| 1.1. Scope..... | 1 |
| 1.2. Intended Audience | 1 |
| 2 System Overview..... | 2 |
| 3 Hardware..... | 5 |
| 3.1. iPhone..... | 5 |
| 3.2. BeagleBoard-xM Rev A3 | 5 |
| 3.3. USB Wi-Fi Dongle..... | 7 |
| 3.4. Arduino Mega | 7 |
| 3.5. Camera and Pan-Tilt System | 9 |
| 3.6. Speakers..... | 10 |
| 3.7. Batteries..... | 10 |
| 3.8. Wheelchair | 12 |
| 3.9. Lynxmotion Pan-Tilt Kit..... | 14 |

| | | |
|--------|--|----|
| 4 | Mechanical..... | 15 |
| 4.1. | Mechanical System Overview..... | 15 |
| 4.2. | Medical Station..... | 16 |
| 4.2.1. | Load Analysis..... | 17 |
| 4.3. | Wheelchair..... | 18 |
| 4.4. | Pan-Tilt Camera System..... | 20 |
| 4.5. | Electrical Housing..... | 22 |
| 5 | iPhone Software..... | 25 |
| 5.1. | General Requirements..... | 25 |
| 5.2. | iPhone Software System Overview..... | 25 |
| 5.2.1. | User Interface..... | 25 |
| 5.2.2. | Receiving Audio/Video..... | 26 |
| 5.2.3. | Sending Audio and Other Data..... | 26 |
| 5.2.4. | Recording Audio..... | 26 |
| 5.3. | System Diagrams..... | 27 |
| 5.3.1. | Flowchart-High Level Overview..... | 27 |
| 5.3.2. | Dataflow Diagram-Sending Data to BeagleBoard..... | 28 |
| 5.3.3. | Dataflow Diagram-Sending Audio to BeagleBoard..... | 28 |
| 5.3.4. | Dataflow Diagram-Receiving Audio/Video from BeagleBoard..... | 28 |
| 5.3.5. | Class Diagram..... | 29 |
| 6 | BeagleBoard Software..... | 30 |
| 6.1. | General Design Guidelines..... | 30 |
| 6.2. | Beagle Board Software System Overview..... | 30 |
| 6.2.1. | Configuration..... | 31 |
| 6.2.2. | Webcam Capture..... | 31 |
| 6.2.3. | Receiving and Playing Audio..... | 31 |
| 6.2.4. | Forward Movement Data..... | 32 |
| 6.2.5. | Software Shutdown..... | 32 |
| 6.3. | High Level Software Flowchart..... | 32 |

| | | |
|--------|--|----|
| 7 | Arduino Software..... | 34 |
| 8 | Joystick Voltage Control Circuit..... | 38 |
| 9 | Test Plan..... | 40 |
| 9.1. | iPhone Software Test Plan | 40 |
| 9.1.1. | Unit Test | 40 |
| 9.1.2. | Integration and Requirements Testing | 42 |
| 9.2. | BeagleBoard Software Test Plan | 43 |
| 9.3. | Arduino Software Test Plan and Electronics Test Plan | 46 |
| 9.4. | System Integration Test Plan | 47 |
| 10 | Conclusion..... | 48 |
| 11 | References | 49 |

List of Figures

| | |
|---|----|
| System Overview | 2 |
| CHAD | 4 |
| Overview of the BeagleBoard-xM..... | 6 |
| Belkin F5D7050 Wireless USB Dongle | 7 |
| Arduino Mega 2560 | 8 |
| Logitech Quickcam Pro 9000 Webcam | 9 |
| Ultra Thin Speakers | 10 |
| Beagle Juice | 11 |
| Mega Backpack..... | 12 |
| Lancer 2000 Power Wheelchair..... | 13 |
| Simplified wheelchair wiring diagram | 13 |
| Lynxmotion pan-tilt servo diagram..... | 14 |
| CHAD mechanical drawing | 16 |
| Medical Station mechanical drawing | 17 |
| 24 kg load simulation results..... | 18 |
| Wheelchair mechanical drawing | 19 |
| Camera System mechanical drawing | 20 |
| Tilt angle range of the servo motor | 21 |
| Pan angle range of the servo motor | 21 |
| Electronic component shelving | 22 |
| BeagleBoard-xM enclosure mechanical drawing..... | 23 |
| Arduino Mega 2560 enclosure mechanical drawing | 24 |
| Example of the Active View that will be displayed to the user | 26 |

High Level Overview 27

Dataflow for sending data to the BeagleBoard 28

Dataflow for recording/sending audio to the BeagleBoard 28

Dataflow for Audio/Video from BeagleBoard..... 28

Class Diagram for CHAD Application 29

High level flowchart for BeagleBoard Software 33

8 Bit Word Map 34

Servo motor position vs duty cycle 35

Motion Control Flowchart 36

Arduino Software Flowchart 37

Arduino to Joystick Controller Circuitry 39

List of Tables

| | |
|---|----|
| Arduino Mega 2560 Specifications | 8 |
| Axis to pin voltage mapping for motor control | 14 |
| Summary of BeagleBoard Software Processes | 31 |
| Arduino output to motor control circuitry | 35 |
| Joystick controller voltages | 38 |

Glossary

| | |
|-----------------------|--|
| CHAD | Controlled Home Assistive Device |
| GUI | Graphical User Interface |
| iOS | iPhone Operating System |
| USB | Universal Serial Bus |
| Wi-Fi | Wireless Fidelity, used in reference to WLAN (IEEE 802.11) technology. In simple terms, it is wireless Ethernet or internet. |
| Client | The person operating the controlling software |
| Patient | The person directly interacting with the robotic device |
| SCM | Source control management |
| Socket | The endpoint of a bidirectional inter-process communication flow across an Internet Protocol-based computer network, such as the Internet. [5] |
| Socket Address | combination of an IP address (the location of the computer) and a port (which is mapped to the application program process) into a single identity [5] |
| D-Pad | Directional pad. A cross shaped device in which each point represents a direction (top means up or forward, left means left, etc) |
| Packet | a formatted unit of data carried by a packet mode computer network [6] |
| IP Address | Internet protocol address |
| Port Number | an application-specific or process-specific software construct serving as a communications endpoint [7] |
| MDF | Medium-density fiberboard |
| TCP | Transmission Control Protocol |
| PWM | Pulse Width Modulation |
| PCB | Printed Circuit Board |
| LED | Light Emitting Diode |
| LPDDR | Low Power Double Data Rate. |
| SBC | Single Board Computer. |
| UVC | USB Video Class. |
| I²C | Inter-Integrated Circuit |
| DVI | Digital Visual Interface; a video interface standard |
| GPIO | General Purpose Input/Output |
| SRAM | Static random-access memory |
| EEPROM | Electrically Erasable Programmable Read-Only Memory |
| UART | Universal asynchronous receiver/transmitter |



1 Introduction

The Controlled Home Assistive Device is a device for people who require a minor amount of supervision for their medical needs, but either do not wish to pay for the costs of a full time nurse, or live in a secluded community where nurses or home care specialists are not readily available. Our objective is to provide these individuals with a product that would allow visual and audible communication with a nurse, physician, or home care specialist at a remote location. While the patient administered some simple tests to check vital states - such as blood pressure or blood sugar levels - the nurse would be able to simply monitor the patient to ensure tests results are in a healthy range. This would allow them to maintain their independence, while at the same time ensuring their well being. The proof-of-concept requirements for the robotic device are outlined in the following document.

1.1. Scope

This document outlines how we aim to achieve our functional specifications listed in Functional Specifications for a Controlled Home Assistive Device. This document applies to only the proof-of-concept model and not the functional model, therefore only specifications of priority P1 and some of priority P2 will be adhered to.

1.2. Intended Audience

This document is intended for use by the entire Telemedix team. It will act as a guideline to ensure all proof-of-concept design requirements are met. This document will also outline test cases that ensure all such requirements are fulfilled.

2 System Overview

An overview diagram outlining the various components and their interconnections for the Telemedix home assistive system is shown in Figure 2.1 below. A more detailed explanation of the individual hardware and software components is given in the subsequent sections.

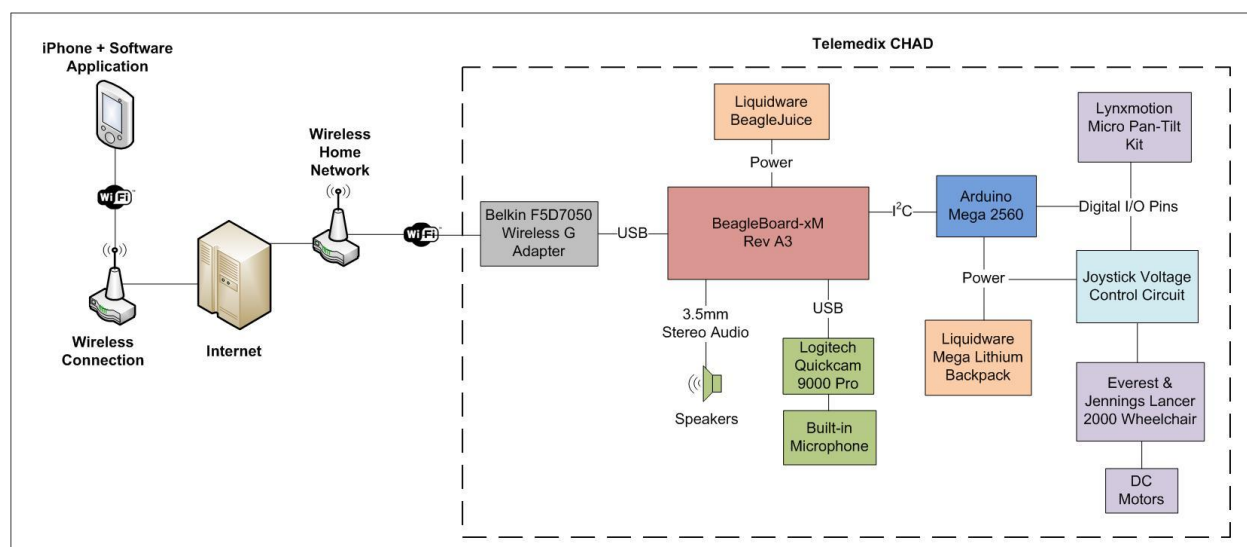


Figure 2.1: System Overview

As shown in the above figure, the system comprises of two main components, the iPhone running the control application and CHAD, the home assistive robotic device, which communicate to each other over the Internet via Wi-Fi connections on each end.

The iPhone application is to be operated by the user, likely a nurse or other home care professional, to control the movement of CHAD and also to control the position of the camera mounted on the device. The software will display a live video feed from the device-mounted camera and will play the audio captured by the built in microphone. Additionally, the iPhone software will stream audio to CHAD allowing voice communication between the user and the patient.

The BeagleBoard-xM single board computer will be mounted within CHAD and will provide high level control of the device subsystems. CHAD will connect to a local Wi-Fi connection via the

Belkin F5D7050 Wireless G Adapter connected by USB to the BeagleBoard, providing a communication channel to the iPhone application. Additionally, software running on the BeagleBoard will capture the audio and video feed from the Logitech Quickcam Pro 9000 webcam, connected via USB, and stream this data to a URL where it can be received by the iPhone application. The BeagleBoard will also receive the audio feed from the iPhone application and play this audio through stereo speakers connected via the 3.5mm audio output jack. Lastly the BeagleBoard will receive device and camera mount movement commands from the iPhone application via a direct TCP/IP socket connection using the same Wi-Fi internet connection. These commands will be passed to the Arduino Mega 2560 using the I2C protocol.

The Arduino Mega will be responsible for controlling the Lynxmotion Micro Pan-Tilt camera mount and interfacing to the joystick circuitry of the Everest & Jennings Lancer 2000 wheelchair. Software running on the Arduino will interpret movement commands received from the BeagleBoard over I2C and drive the appropriate digital I/O pins to properly control the pan-tilt servos and the joystick controller of the wheelchair. Some custom circuitry will be used to adjust voltages from the digital I/O pins of the Arduino Mega to interface to the joystick controller. This circuit is discussed in more detail in a subsequent section.

Power is supplied to the BeagleBoard, the Arduino Mega and the joystick voltage control circuit by two battery packs from Liquidware Open Source Electronics: the BeagleJuice and the Mega Lithium Backpack.

A developed 3D model of CHAD is shown below in Figure 2.2. The physical characteristics of the device will be presented and discussed in detail in the Mechanical section of this document.



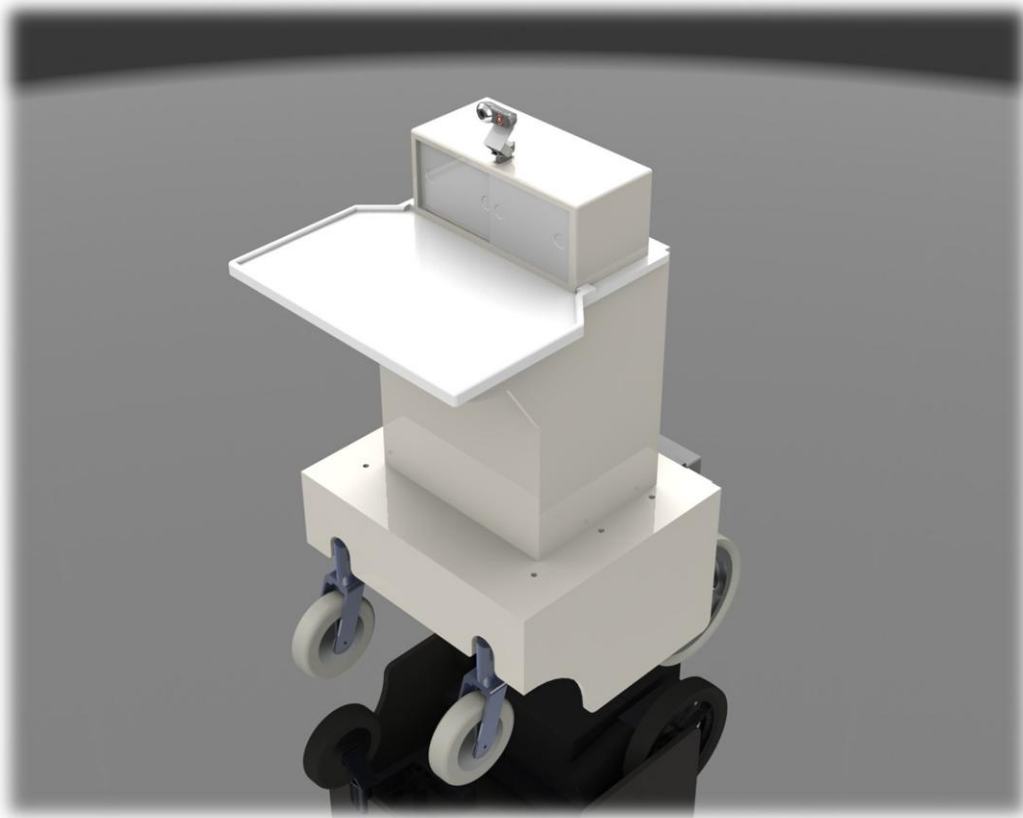


Figure 2.2: CHAD

3 Hardware

This section of the document outlines the major hardware components used in the Telemedix CHAD system. Various features and specifications are given for each component along with a rationale for why these specific components were selected for the design.

3.1. iPhone

In order for use to control our device from remote locations, it is necessary for our remote control software to satisfy a list of requirements. The iPhone was chosen as our controller hardware because it satisfied all these requirements as well as providing some other advantages. First, it has all the necessary capabilities that we require. It has the ability to connect to Wi-Fi networks, it has a microphone for recording audio, a front facing camera (which would be useful for implementing two way video if we were going to), a high resolution screen for displaying video that is easily viewable (it also allows for adjusting the brightness if the user so desires), and speakers loud enough for the audio feed to be easily heard. Another benefit is that the iPhone has a touch screen, which will allow us a lot of flexibility in choosing our controlling scheme. Also, a mobile device was chosen for the controlling hardware because we wish to demonstrate that our device can be controlled from anywhere, allowing for operation of our device to be very convenient. On top of all this, one of our group members has experience developing applications for the iPhone and also has one, saving us the cost of purchasing one and allowing for the development of the controlling software to be more efficient.

3.2. BeagleBoard-xM Rev A3

The BeagleBoard-xM is a low-cost single board computer capable of delivering laptop like performance with support for a large number of peripheral devices through various expansion headers and ports. A system overview highlighting some of the key features is shown in Figure 3.2.1 below.



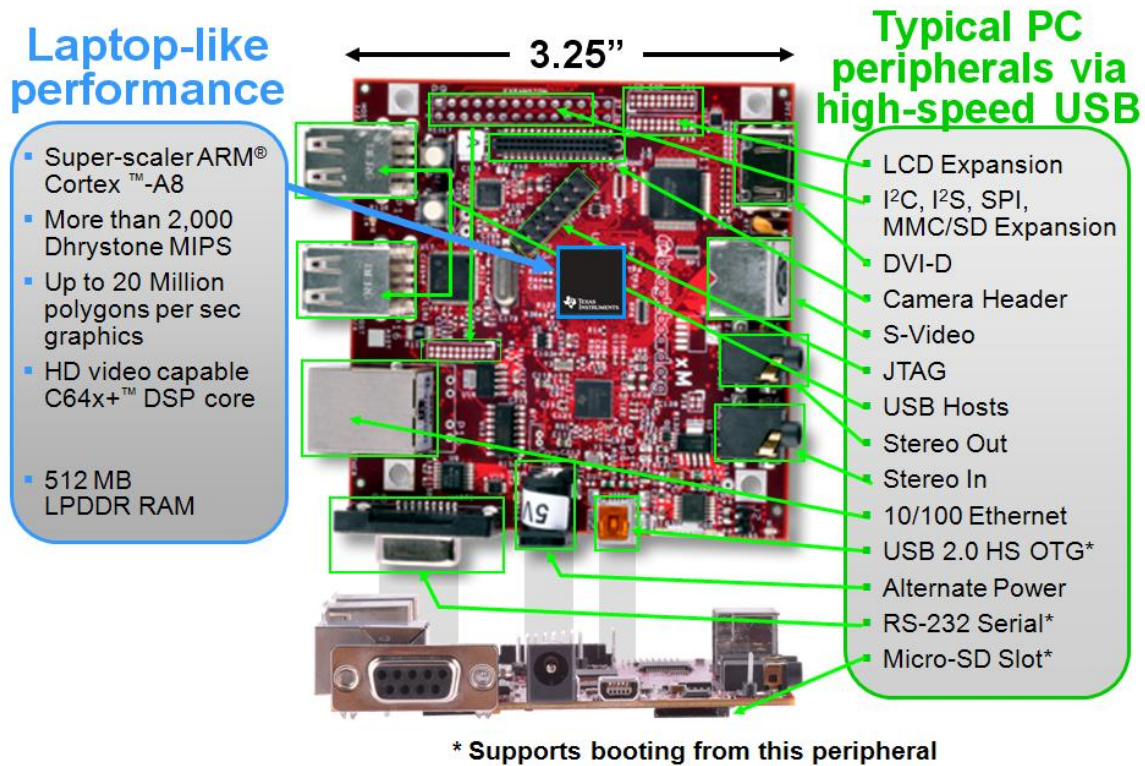


Figure 3.2.1: Overview of the BeagleBoard-xM [9]

The BeagleBoard features an OMAP3 series processor from TI, with an ARM Cortex A8 core running at 1 GHz and a C64x+ DSP core clocked at 800 Mhz. [3.1.2] The board comes with 512 MB of LPDDR RAM and uses a microSD card for storage. A 4 GB card ships with the board preloaded with a demo image of the Angstrom Linux distribution.

There are several features of this SBC that make it suitable for our robotics application. The BeagleBoard-xM is priced at \$149 USD giving a very good feature to cost ratio. The board is capable of running full Linux desktop distributions including Ubuntu 10.04 Lucid Lynx which will be used due to its ease of use, compatibility with a wide number of peripherals and large community support base. The BeagleBoard also includes 4 USB host ports which will be used to interface to the webcam and the USB Wi-Fi dongle which will provide wireless internet access for CHAD. A 3.5 mm stereo output jack is also included on the BeagleBoard, which will be used to send stereo audio received from the iPhone to the speaker system. The board also has support for the I²C interface through one of the many expansion headers. This interface will be used to connect the BeagleBoard to the Arduino Mega 2560.

The BeagleBoard is a low power system, designed with mobile applications in mind. [10] We will be using the BeagleJuice lithium battery module, made available from Liquidware Open Source Electronics to supply the required 5 V to the BeagleBoard. This battery is discussed in further detail in Section 3.7.

There are a number of features that will ease software development on this device including DVI-D connection for computer monitors, 4 USB ports with support for low-speed, full-speed and high-speed devices allowing developers to use a standard USB mouse and keyboard. The 10/100 Ethernet connection allows the device to be connected to the internet, without the need for a wireless USB dongle which is useful for development purposes. The board can also be powered from a 5 V AC/DC adapter eliminating the need to use a battery pack during the development stages. Lastly, the device also includes an RS-232 serial port which will be useful for debugging purposes.

3.3. USB Wi-Fi Dongle

Since CHAD is to be remotely controlled over a Wi-Fi Internet connection, a USB Wi-Fi dongle will be used to provide Wi-Fi Internet to the BeagleBoard. We chose to use the Belkin F5D7050 G Wireless USB Network Adapter. This adapter is shown below in Figure 3.3.1.



Figure 3.3.1: Belkin F5D7050 Wireless USB Dongle [11]

This adapter was specifically chosen for its compatibility with the BeagleBoard as well as the community support for configuring the device. [12]

The device supports the IEEE 802.11g standard, which will provide sufficient bandwidth for our video and data streaming application.

3.4. Arduino Mega

Arduino is an open-source electronics prototyping platform which has flexibility and easy to use hardware and software. It is intended for various applications, especially for interacting/prototyping with different devices and circuitry. Arduino is based on an ATmega microcontroller, which is programmed using Arduino programming language and Arduino IDE. Its projects can be stand-alone or can be connected to a computer. The Arduino programming language is based on C and has many software libraries to provide flexibility and ease of use. One such library is the servo library, which features an automatic refresh of the motor signal every 50 μ s to hold the motors' position. Additionally, this platform is widely documented and has a large online community.

The purpose of using an Arduino microcontroller platform is to control the motion of the robotic device. Acting as a slave, the Arduino will communicate with the BeagleBoard and will execute received instructions to control the chassis motors and the pan tilt camera system.

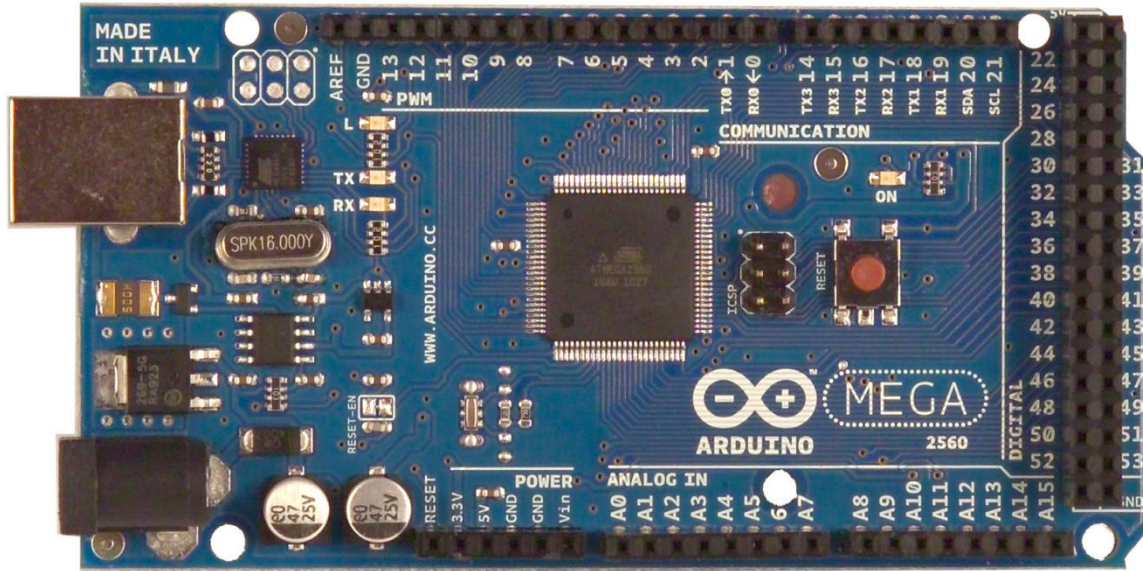


Figure 3.4.1: Arduino Mega 2560 [13]

The robot will have an Arduino Mega 2560 as shown above, which is based on the ATmega1280. The following table is a summary of the Arduino Mega 2560 specifications and features.

Table 3.4.1: Arduino Mega 2560 Specifications [13]

| | |
|-------------------------------|---|
| Digital i/o Pins | 54 (14 PWM Pins) |
| Analog Pins | 16 |
| UARTS | 4 |
| Clock Speed | 16MHz |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limits) | 6-20V |
| DC Current per I/O Pin | 40 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 256 KB of which 8 KB used by bootloader |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Serial Communication Pins | 10 (supports I ² C) |
| Additional Accessories | USB Connection |
| | 5V Power Jack |
| | ICSP Header |
| | Reset Button |
| | Overvoltage Protection |

The major reason for choosing this platform was due to the large flexibility and the ease of use of both hardware and software. There is a large community and wiki for code samples, and support. Lastly, for all these, features and benefits, the Arduino cost is also cost effective.

3.5. Camera

The camera used is the Logitech Quickcam Pro 9000 USB webcam. Figure 3.5.1 below shows the webcam.



Figure 3.5.1: Logitech Quickcam Pro 9000 Webcam [14]

This specific webcam was chosen due to the following features: Hi-Speed USB 2.0 interface, built-in microphone, 2 megapixel HD sensor, autofocus, supported resolutions of up to 1600 x 1200, reasonable cost of \$79.99 USD, Linux support through UVC drivers [15] [16]. The webcam will interface to the BeagleBoard via USB and driver support will be provided by the Linux UVC driver, which is included in kernels from Linux 2.6.26 and onward. [17]

3.6. Speakers

Below is a figure of the speakers that will be used in the design.



Figure 3.6.1: Ultra Thin Speakers [18]

To avoid the use of any more additional power sources, portable speakers will be mounted on CHAD to enable playback of the audio received from the iPhone. The NXT 4W Folding Speaker will be used due to the slim design, and the fact that it operates from AAA batteries.

3.7. Batteries

The robot is powered by using 4 batteries. Two 12V batteries are connected to the motor controller in order to power up the motors. The additional two batteries are to power the BeagleBoard and the Arduino Mega 2560 respectively.

The BeagleBoard will be powered using a battery shield called the Beaglejuice as shown below.

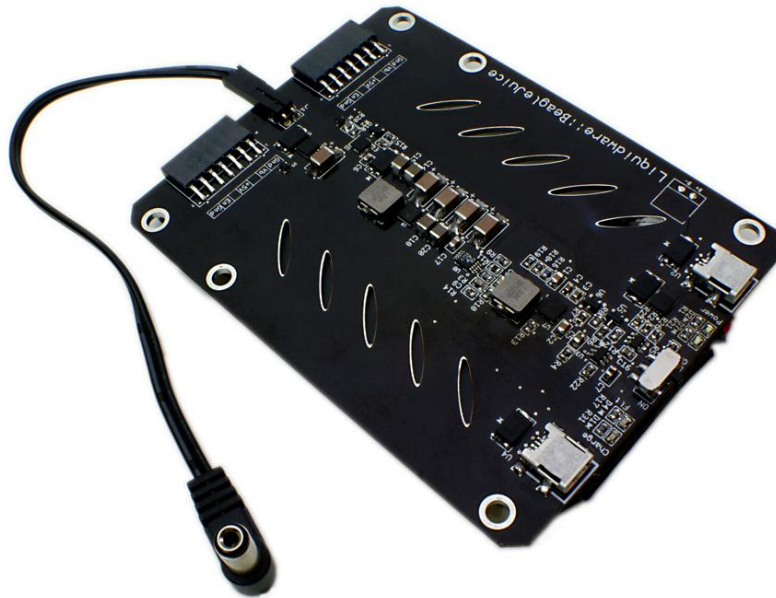


Figure 3.7.1: Beagle Juice [19]

The Beaglejuice is a battery module that provides a 5V power supply directly to the BeagleBoard. The shield's battery is a 4500 mAh Li-ion battery and connects to the BeagleBoard via 2-pin barrel jack. The battery can be charged through a Type B-mini USB port and a second USB port is available for accelerated charging. There is also an ON/OFF switch to make booting and shutting down simple and safe. Since it is a shield, it can be mounted on the backside of the board using standoffs, which keeps the expansion ports easily accessible.

Below are a summary of the battery specifications :

- 4500 mAh battery
- 5V output
- 1.5A current delivery
- powers a BeagleBoard for at least 6.5 hrs
- on/off switch
- single or dual USB charging via Type B-mini ports
- battery charge status LEDs
- pin-headers to provide direct power supply for any application
- 4 mounting standoffs included

The Arduino Mega 2560 is powered using another shield called the Mega Backpack as shown below.

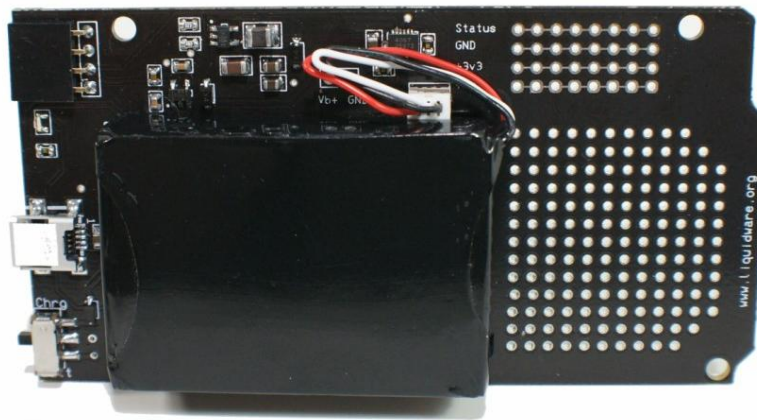


Figure 3.7.2: Mega Backpack [20]

This is a Li-ion battery that delivers 29 hours of standby time for the Arduino. It is mounted on a PCB that has the same dimensions as the Arduino Mega 2560, and fits exactly onto the backside of Arduino using standoffs. The board comes with additional prototyping space for placing external circuitry. The battery is charged via mini USB.

Below is a summary of the battery specifications:

- Bare battery PCB matches the size of the Arduino
- Extended easy-soldering prototyping space
- High Capacity 2200mAh Lithium Ion Battery (29 Standby Arduino Hours)
- Rechargeable via Arduino USB
- Rechargeable via USB Type-B Mini Cable
- Supplies regulated 5V and 3.3V
- Monitor battery voltage status
- Dimensions:
 - L x W = 4.00×2.10 inches (same as the Arduino Mega 2560)
 - Battery height = 0.498 inches

The reason we have chosen these battery backs is because they have been proven to work with these boards and provide regulated voltage. Additionally, these can supply power for long periods of time, and can be mounted below the boards using standoffs.

The ground pins from the backpacks will be connected to the ground of the joystick controller to provide a common ground.

3.8. Wheelchair

The Everest & Jennings Lancer 2000 wheelchair will be utilized as the underlying support structure and drive train function. The wheelchair includes two independently controlled Fracmo DC-motors and accompanying gearboxes [1] [2]. The wheelchair is powered by two 12V batteries made by

MK Battery [3]. Similar wheelchairs by this manufacturer are capable of supporting 136 kg (300lbs); this provides high flexibility in designing CHAD when weight may become an issue [4]. The mechanics and implementation of the wheelchair will be discussed in the following sections.



Figure 3.8.1: Lancer 2000 Power Wheelchair [25]

A wiring diagram of the wheelchair's electronic system is provided below in Figure 3.8.2.

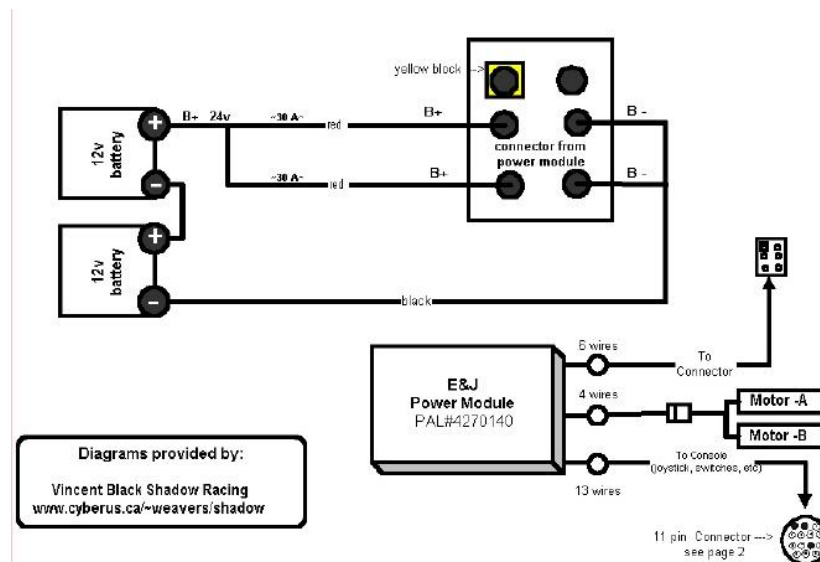


Figure 3.8.2: Simplified wheelchair wiring diagram [26]

The wheelchair's motors are controlled through the E&J Power Module, previously shown in Figure 3.8.2. The control mechanism to this power module is a multi-directional joystick which allows for accurate maneuverability. The joystick works off the basis of two control pins, namely X-axis and Y-axis. By varying the voltage on each pin across a certain range, the power module supplies a



signal to the required wheelchair motors. The X-axis controls the wheelchair’s left and right rotation whereas the Y-axis controls the forward and reverse direction. A combination of these axes will move the wheelchair in any desired direction

For CHAD’s application, only a single axis will be manipulated at once. This means it will rotate on its center point and move forward or reverse on separate instances. The axis-voltage mapping is shown below in Table 3.5.1

Table 3.5.1: Axis to pin voltage mapping for motor control

| Direction | X-axis Voltage (V) | Y-axis Voltage (V) |
|-----------|--------------------|--------------------|
| Forward | 0 | 0.77 |
| Reverse | 0 | -1.07 |
| Left | -1.20 | 0 |
| Right | 0.85 | 0 |

3.9. Lynxmotion Pan-Tilt Kit

The Lynxmotion pan and tilt kit provides multi-directional movement through the use of two servo motors mounted together with custom brackets which also come with the kit. This kit is an ideal choice for mounting the required camera due to its small footprint. The kit is shown below in Figure 3.9.1.



Figure 3.9.1: Lynxmotion pan-tilt servo system [29]

The kit will be discussed in further detail in terms of its mechanical and electrical functionality in the appropriate sections.

4 Mechanical

4.1. Mechanical System Overview

CHAD consists of 3 major mechanical subsystems; the medical station, wheelchair chassis, and the pan-tilt camera system. Each subsystem contributes to an integral part of CHAD and will be discussed in detail in its own subsection. A mechanical outline of CHAD, who's 3D-model is previously shown in Figure 2.2, is shown below with dimensions focusing on the overall structure and size.



| ITEM NO. | PART NUMBER | DESCRIPTION | QTY. |
|----------|------------------------|---------------------|------|
| 1 | Wheelchair_Final-Assy | WHEELCHAIR CHASSIS | 1 |
| 2 | MedStat_Final-Assy | MED. STATION & BODY | 1 |
| 3 | Wheelchair_Control-box | MOTOR CONTROL BOX | 1 |
| 4 | CamSys_Final-Assy | CAMERA SYSTEM | 1 |
| 5 | Bb_Enclosure-Assy | BEAGLEBOARD | 1 |
| 6 | Arduino_Enclosure-Assy | ARDUINO | 1 |

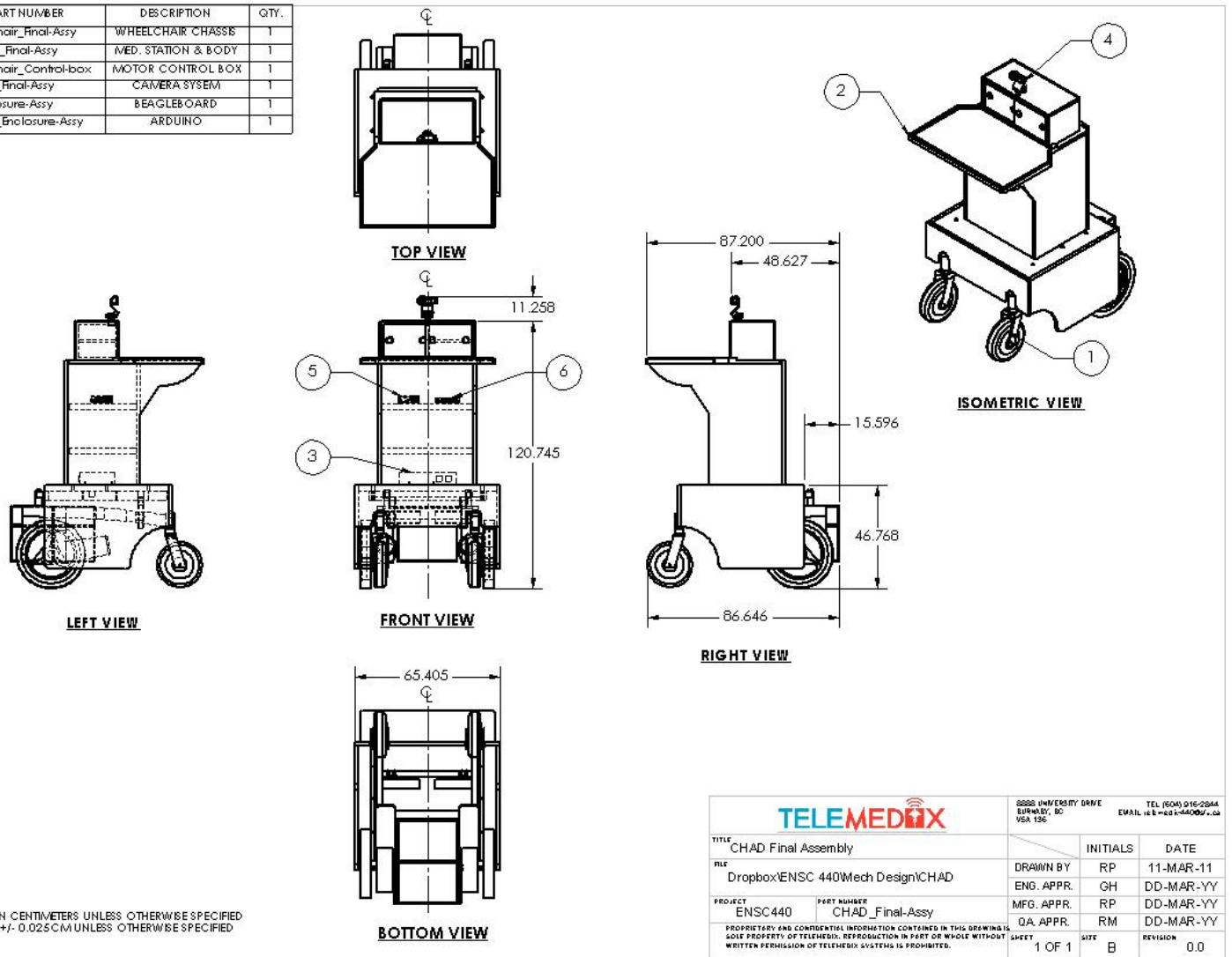


Figure 4.1.1: CHAD mechanical drawing

4.2. Medical Station

The medical station will be comprised of a table area, cabinet, and a center support structure built from individual panels. The majority of this subsystem will be built from 16mm MDF panel that will be a strong and relatively light building material. The base will have 6 mounting holes (see Bottom View), which will align with holes on the wheelchair chassis and will be secured with the combination of flat washers and bolts. The side skirt panels will be used to hide several of the wheelchair's components from the observer. Cutouts made on the skirts will allow the wheelchair's caster wheel to rotate freely without any collisions. The cabinet will be built with two identical sliding doors to allow easy access to items stored inside. The cabinet will have 3 compartments giving a total of 0.0805 cubic meters. Mounting holes (see Detail A) on top of the cabinet will be used to secure the camera system. The table will provide 220 square centimeters of surface area, which

should be more than ample for its purpose. Shelving will be provided inside the hollow structure to allow the electronic components to be securely mounted and stowed away. The overall weight will be approximately 36 kg (MDF density=745 kg/m³ [27]).

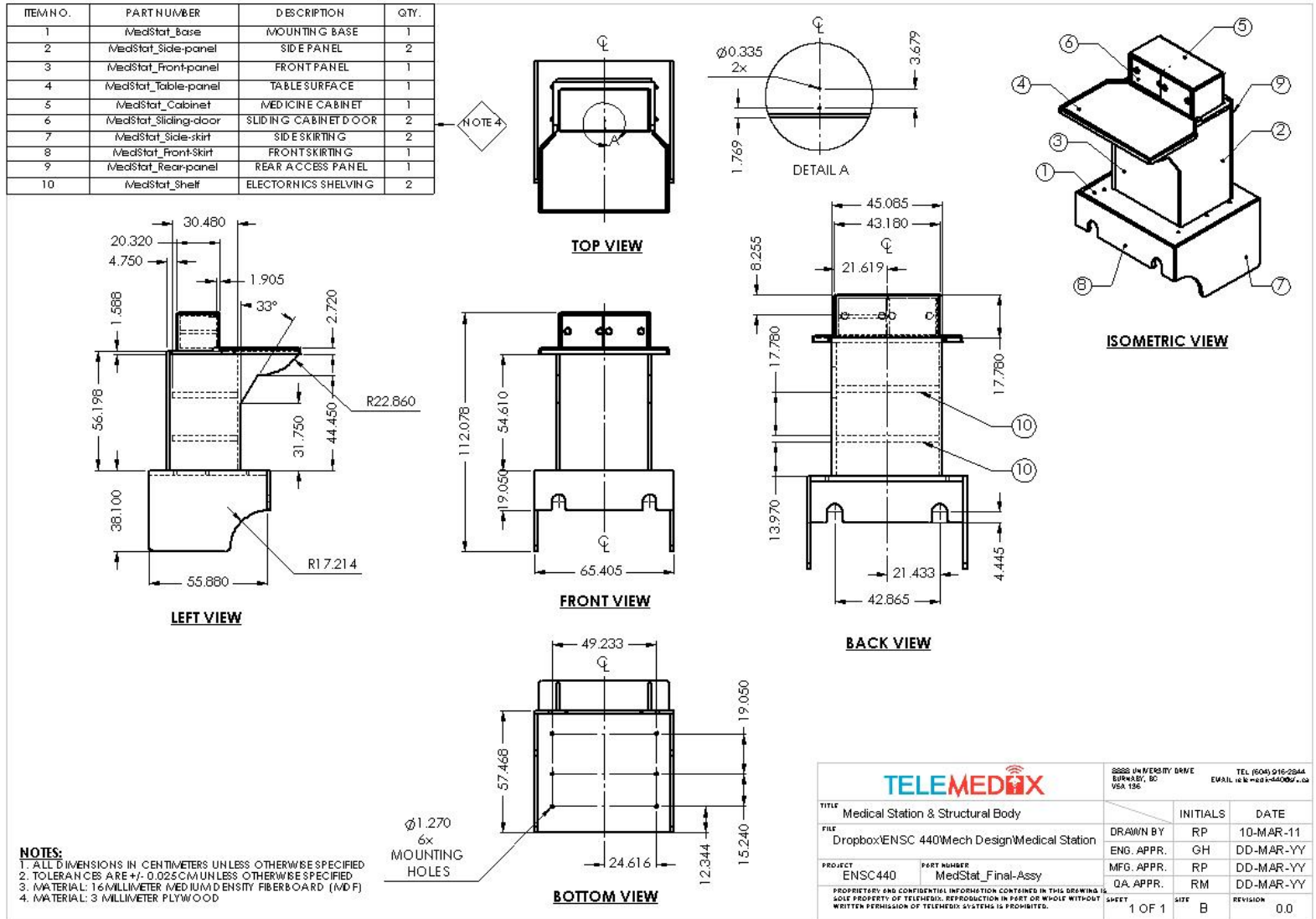


Figure 4.2.1: Medical Station mechanical drawing

4.2.1. Load Analysis

Since the table must be capable of supporting 12 kg, a simulation has been completed which tests the medical station under twice the required load, 24 kg. The results, seen below, show maximum displacements on the scale of 10⁻⁵ mm. This is absolutely negligible and passes the required supporting weight. MDF is a strong and versatile material for this application.

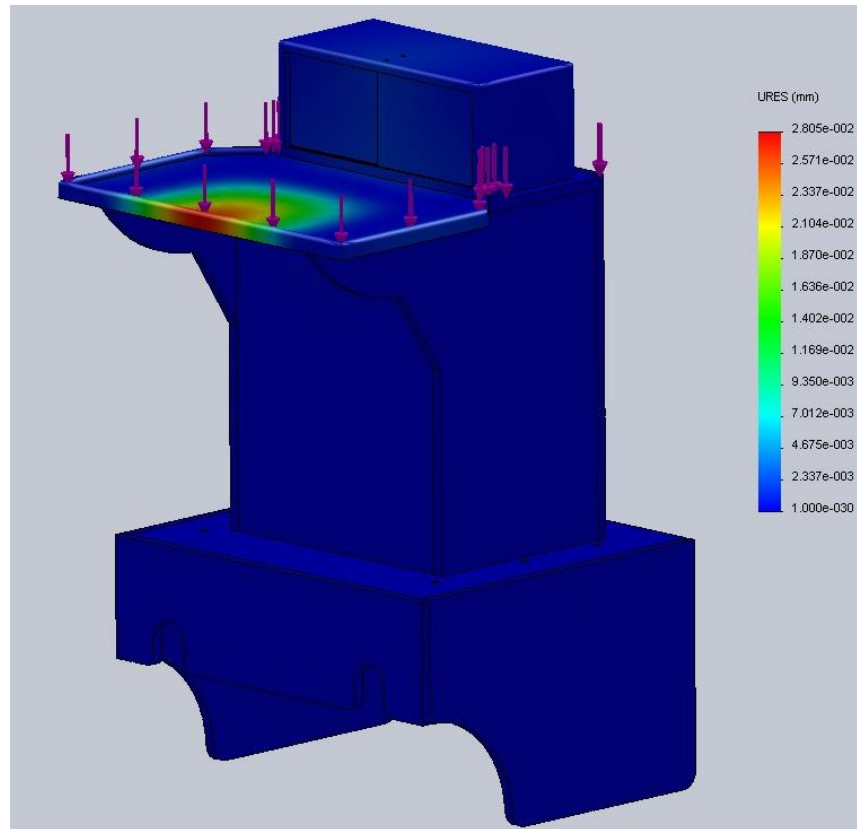


Figure 4.2.2: 24 kg load simulation results

4.3. Wheelchair

By removing unnecessary parts of the Lancer 2000 Power Wheelchair such as the seat and arm/leg rests, the chassis and electromechanical components have been utilized. The chassis is capable of supporting approximately 138 kg [24]. Since the medical station and maximum table load account for only 48 kg, the chassis is an excellent support structure. A mechanical drawing of the wheelchair is shown below in Figure 4.3.1.

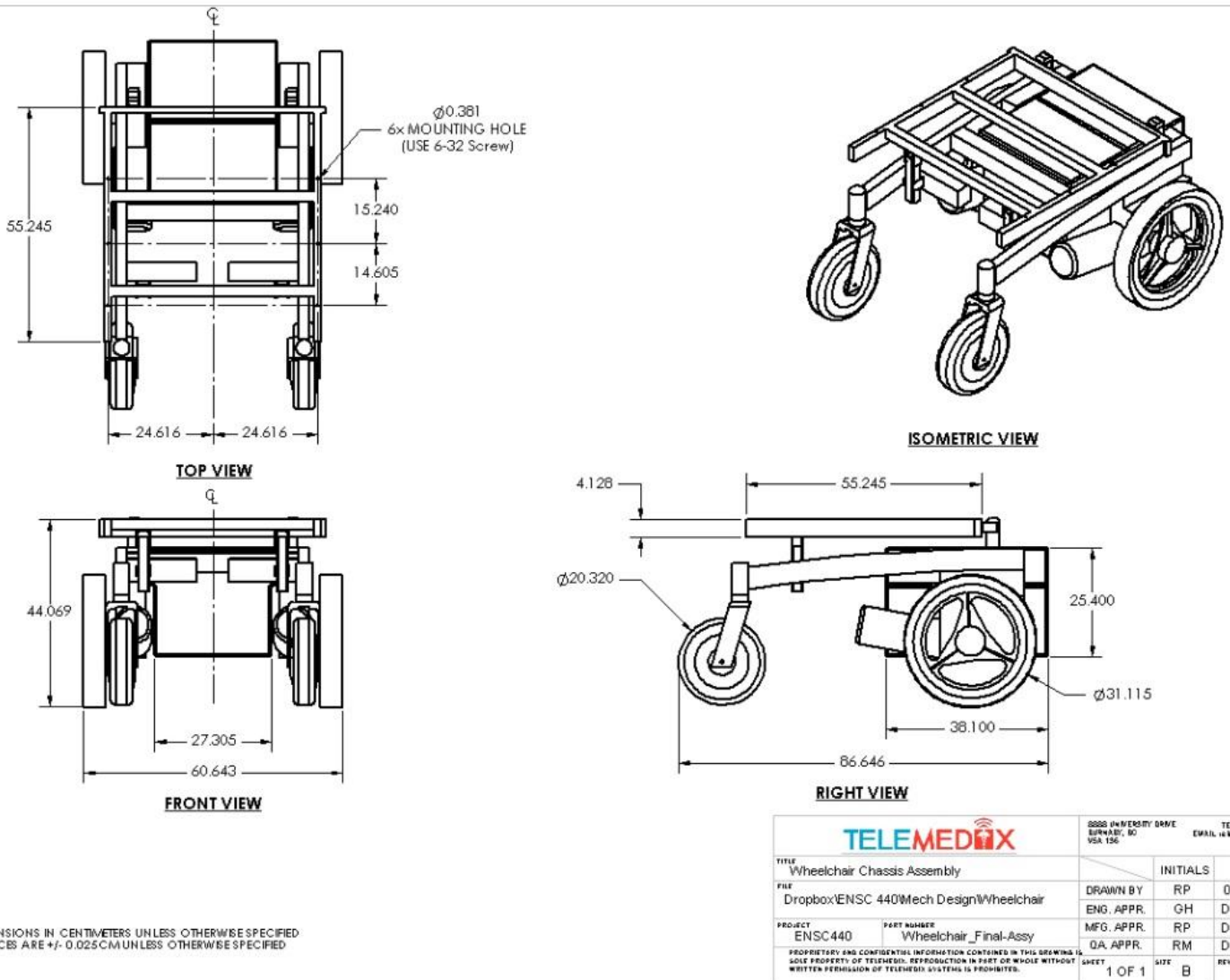
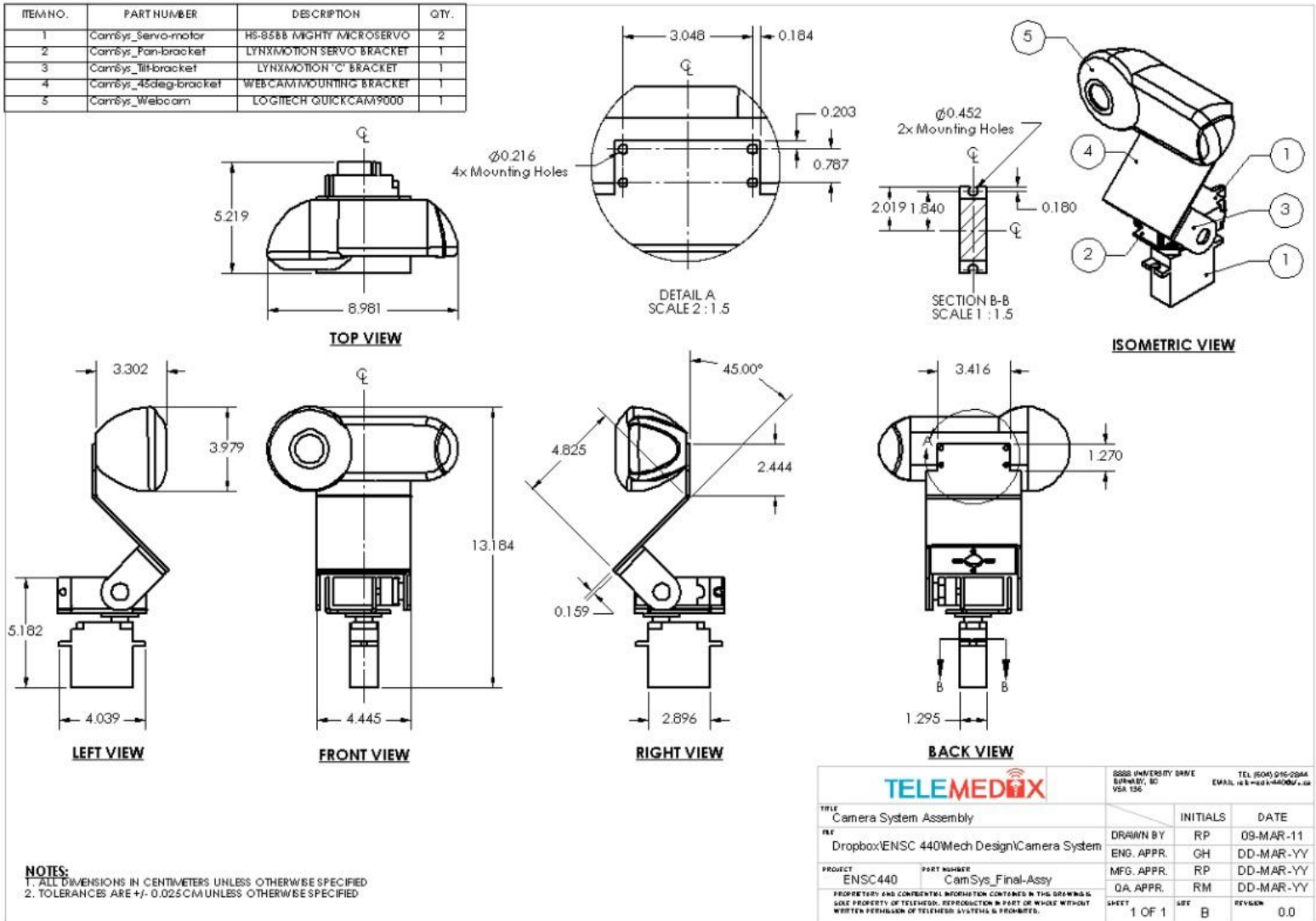


Figure 4.3.1: Wheelchair mechanical drawing

The DC motors, manufactured by Fracmo, run off the 12V battery supply and have a speed range of 1500-4000 RPM with an output power of 75-500 W [21]. The motors are accompanied by Fracmo gearboxes, which convert the high RPM motors to a maximum output torque of 40 N•m [22]. The gearbox also has a mechanical override lever that allows for the wheelchair to be moved manually.

4.4. Pan-Tilt Camera System

The Lynxmotion pan-tilt kit is supplied with two HS-85BB Might Micro servos. Each servo has a torque of 0.3 N-m and a variable angle of 180 degrees [28]. Metal servo mounting brackets have also been supplied by the Lynxmotion kit. The Logitech Quickcam 9000 Pro will be secured by a



custom sheet-metal bracket (ITEM NO. 4). The mechanical diagram of the camera system is shown below in Figure 4.4.1.

Figure 4.4.1: Camera system mechanical drawing

The range of the camera's pan and tilt angle is demonstrated below in Figure 4.4.2 and 4.4.3. These angles cover a sufficient viewing angle and are also capable of looking down at the table area.

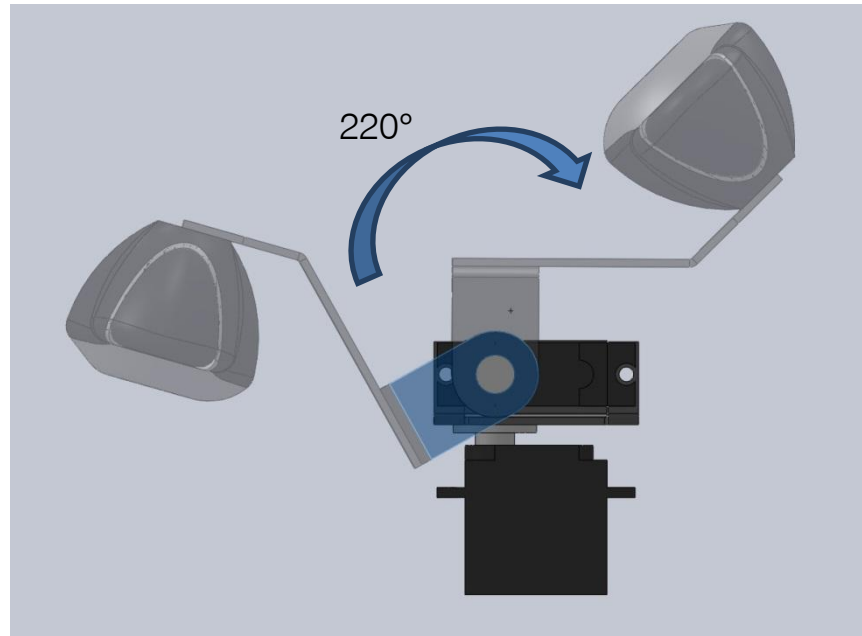


Figure 4.4.2: Tilt angle range of the servo motor

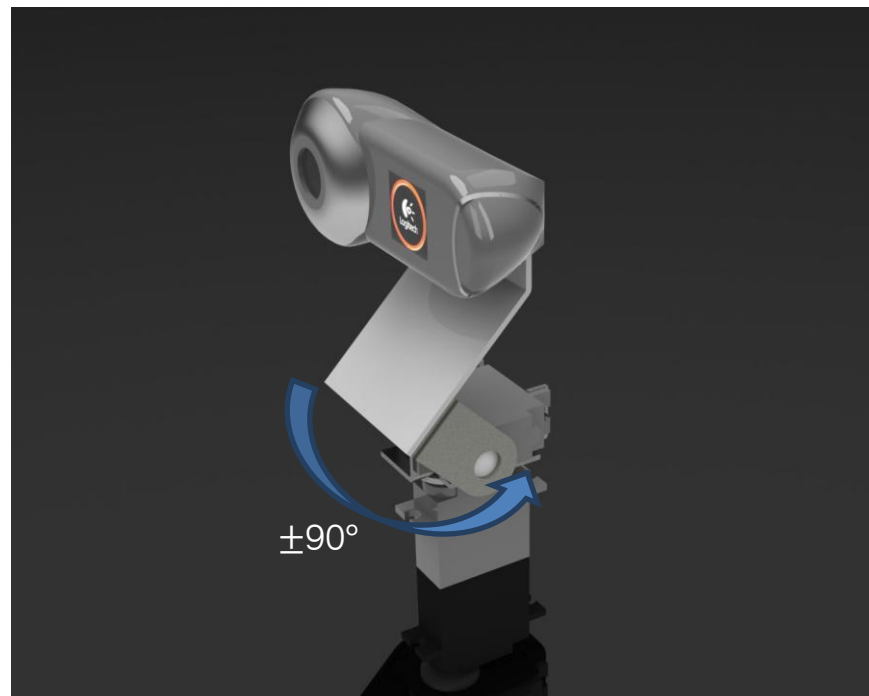


Figure 4.4.3: Pan angle range of the servo motor, current location @ 0°

4.5. Electrical Housing

The two main development boards, BeagleBoard-xM and Arduino Mega 2560, will be enclosed individually in sheet metal boxes and stored within the device's inner shelving. This will protect the electronics from physical damage and will allow them to be easily mounted and removed as required. Figure 4.5.1 shows the components placed inside the device's mounting area. Other components such as power circuitry and wiring harnesses will be stowed away inside the same shelving unit. This will provide a clean exterior look to the device without any visible wires and connectors.

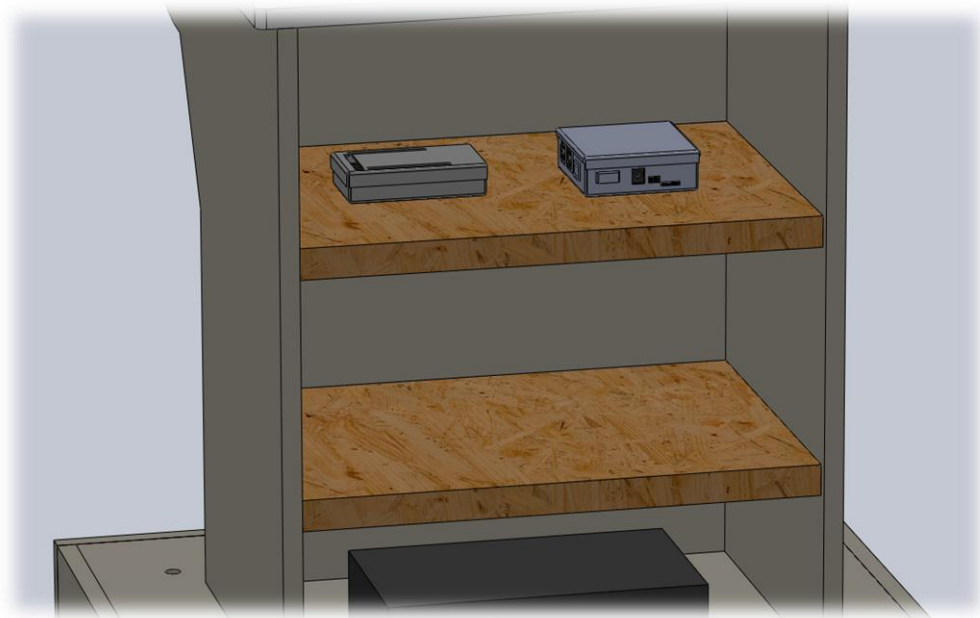


Figure 4.5.1: Electronic component shelving

Figure 4.5.2 and Figure 4.5.3 show the mechanical drawings for both the BeagleBoard and Arduino enclosures respectively. Each enclosure will have specific cut-outs for the various connectors attached to the board.

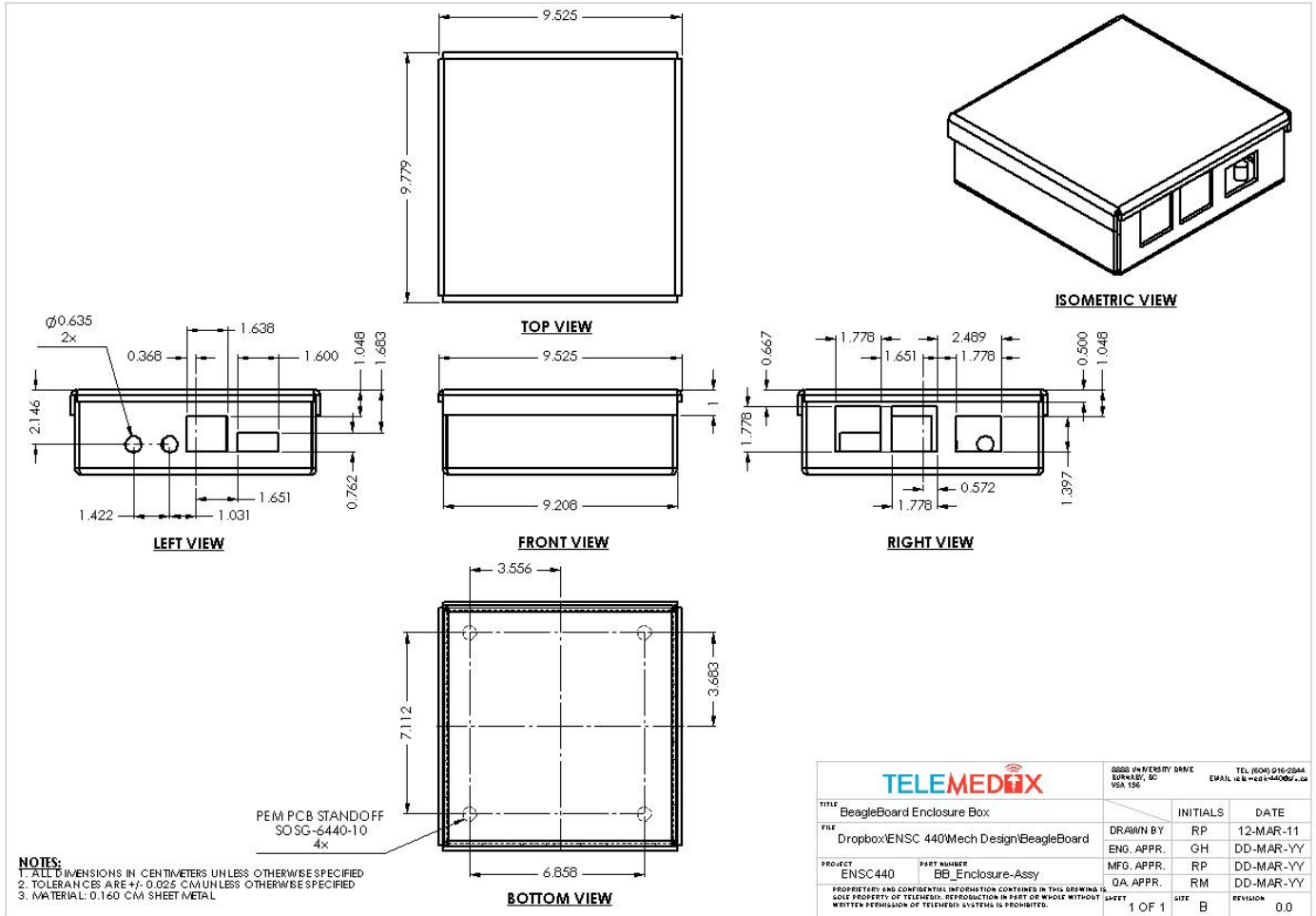


Figure 4.5.2: BeagleBoard-xM enclosure mechanical drawing

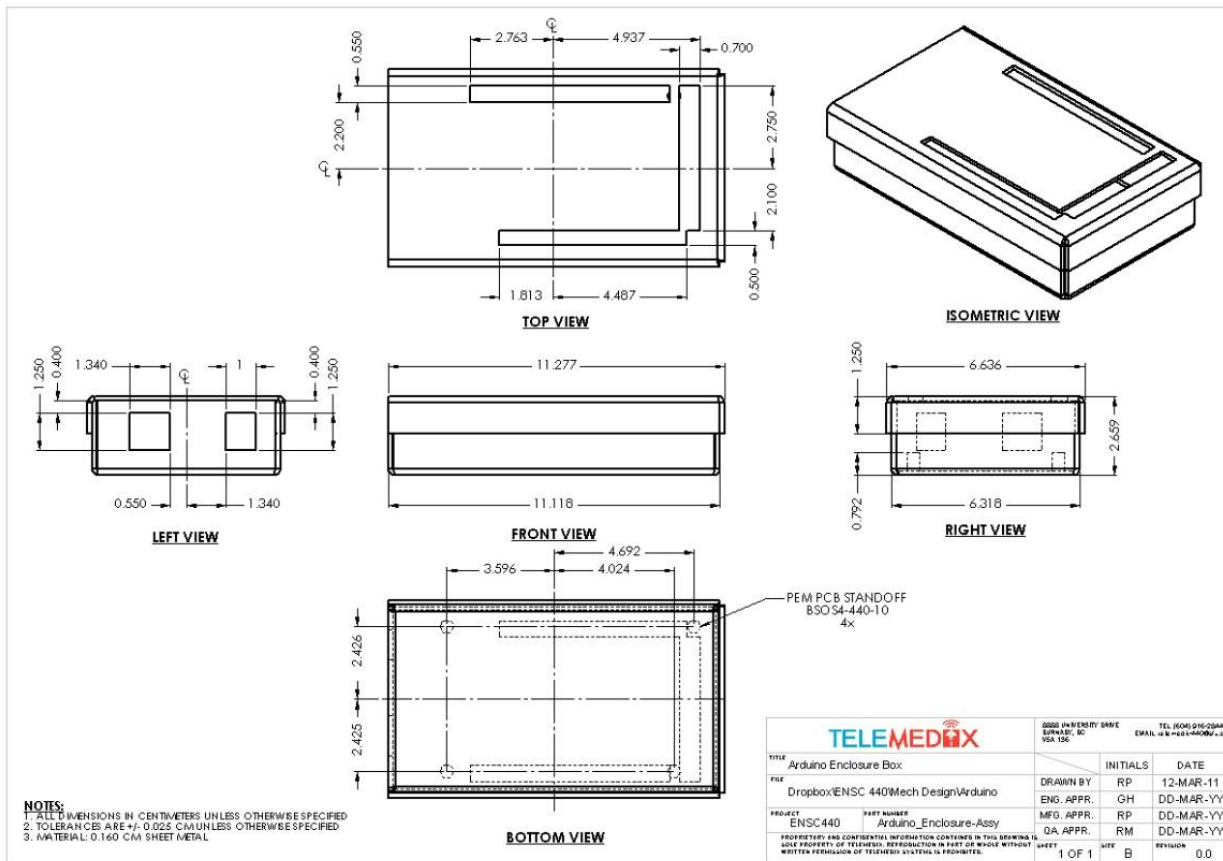


Figure 4.5.3: Arduino Mega 2560 enclosure mechanical drawing

5 iPhone Software

5.1. General Requirements

- Will be using Mercurial Distributed SCM (version 1.7.5+20110201) for managing all source code with a repository located on Google code
- Will be developed using the Objective C language in XCode 3.2.5
- Will use MediaPlayer framework for playing live audio/video feed sent from BeagleBoard
- Will use Core Audio and AVFoundation frameworks for audio processing
- Will use Core Foundation framework for socket communication

5.2. iPhone Software System Overview

The iPhone software for our robotic device will be operated by the user, allowing them to control the movement of the camera and the robotic device itself. It will play a live video/audio feed (from the camera mounted to the robotic device) and will allow the user to easily move the robotic device and the camera independent of each other. It will also record the audio spoken by the user, and send this to the robotic device itself, to be played on its speakers.

5.2.1. User Interface

The iPhone control software will consist of three main screens; the main menu, a help screen and the screen from which the robotic device will be controlled (called the “Active View”). The main menu will consist of a start button and a help button. The help button will lead the user to a help screen, which will list some of the basic controls and what they do. The start button, when pressed will first check for an internet connection. If none is found, it will prompt the user to connect to the internet and the software will remain at the main screen. Once a connection has been established, pressing start will cause the software to remotely connect to the robotic device (described in more detail in the following section). After this, the audio/video feed from the robot will be fetched and played/displayed to the user. Overlaid on this video feed, the controls for the movement of the robotic device itself and the camera will be overlaid (Figure 5.2.1).





Figure 5.2.1: Example of the Active View that will be displayed to the user

The left D-Pad will control the camera movement, while the right D-Pad will control the movement of the robotic device. There will also be a button that the user can press to exit the program (no picture available as not yet implemented) that will safely shut down the software.

5.2.2. Receiving Audio/Video

In order to receive the live audio/video feed discussed above, the iPhone will be accessing the data from a URL and will play this feed live on the iPhone. The video/audio feed will be posted to a URL by the BeagleBoard software (discussed in the corresponding section), and will be fetched using some Objective C audio/video frameworks (MoviePlayerController class from MediaPlayer framework). One of the methods in this framework takes a URL to an audio/video feed, and does the work of fetching the data. This “movie player” is then loaded into the Active View (the view that the user controls the robot/camera from) and is displayed.

5.2.3. Sending Audio and Other Data

In order to send/receive data between the iPhone and the BeagleBoard, a socket will be used, which will allow for data to be sent and received directly between the two. This socket connection will be used to send the control strings related to the movement of the camera and robotic device, and will also be used to send the audio feed recorded by the iPhone to the BeagleBoard. Using the Core Foundation framework, we will establish a TCP socket connection by supplying the IP address and port number of the server (ie. the BeagleBoard). Then using methods from this framework, we can write to a stream that can be read by the server.

5.2.4. Recording Audio

To record audio, we will use the AVFoundation and Core Audio frameworks. We will also use these frameworks to process the audio into small files (buffers) to be sent to the BeagleBoard using socket communication. A function (“recordAudio”) will handle the task of recording and

processing the audio into formats suitable for being sent via socket, and then will send the audio. This function will run once “Start” is pressed from the main menu.

5.3. System Diagrams

5.3.1. Flowchart-High Level Overview

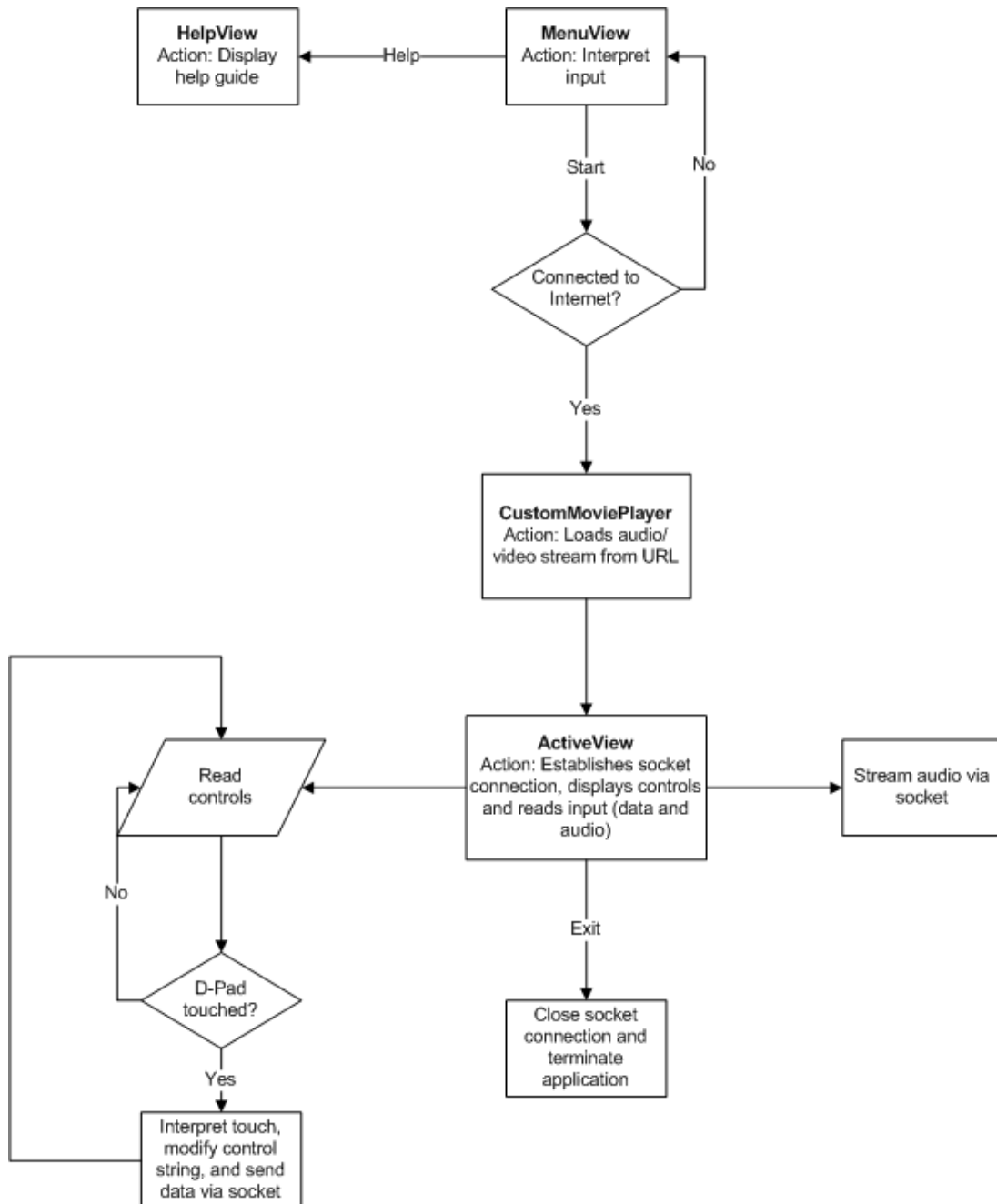


Figure 5.3.1: High Level Overview

5.3.2. Dataflow Diagram-Sending Data to BeagleBoard

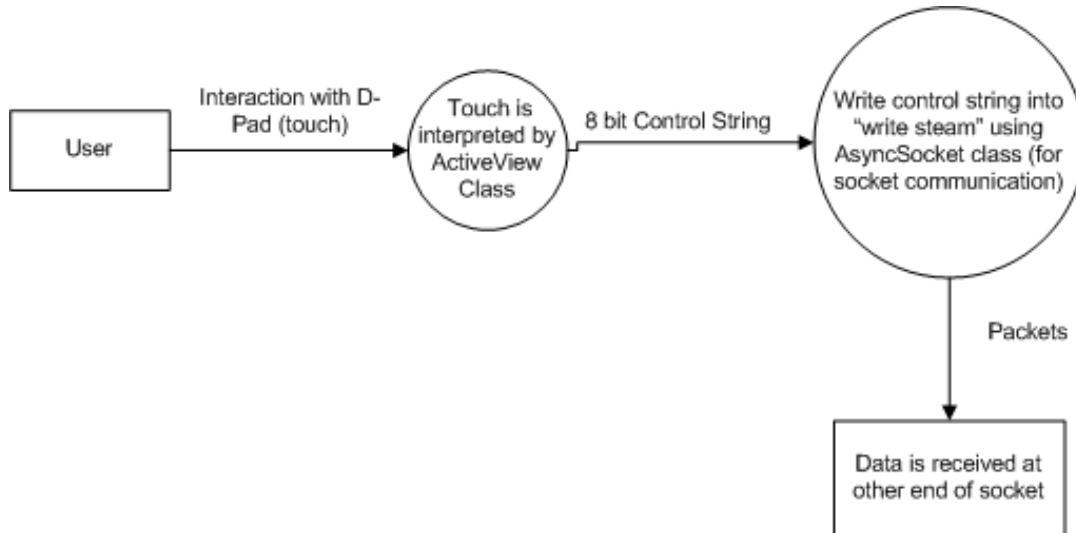


Figure 5.3.2: Dataflow for sending data to BeagleBoard

5.3.3. Dataflow Diagram-Sending Audio to BeagleBoard

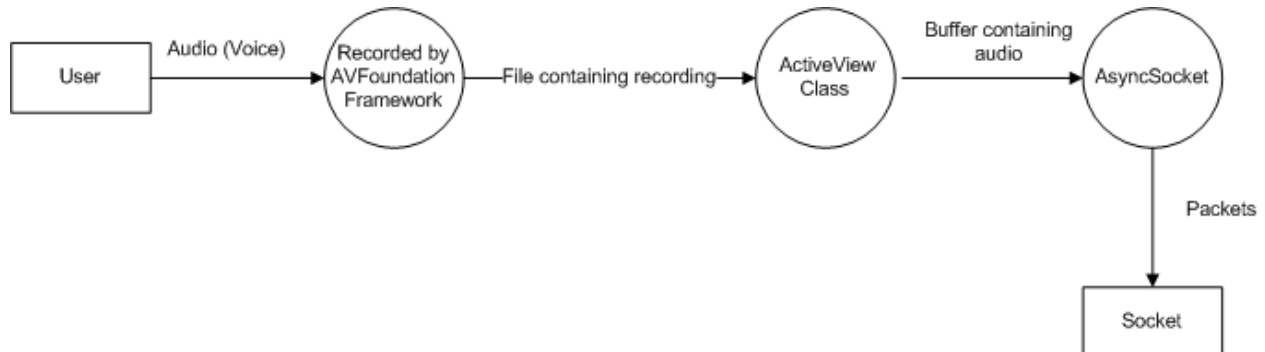


Figure 5.3.3: Dataflow for recording/sending audio to BeagleBoard

5.3.4. Dataflow Diagram-Receiving Audio/Video from BeagleBoard

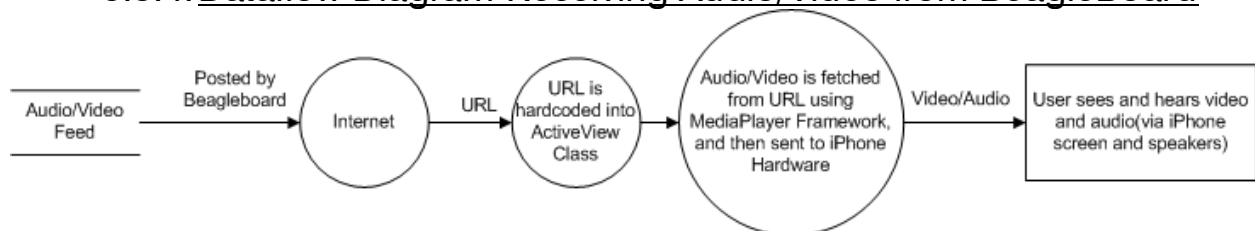


Figure 5.3.4: Dataflow for Audio/Video from BeagleBoard

Note: Please note that that URL will be hard coded into the software.

5.3.5. Class Diagram

The following is a high-level class diagram illustrating all the major classes and their respective functions. The attributes and operators of the parent classes have been omitted as these classes are provided for use as part of the Objective C language. The proper documentation for these classes can be found online at <http://developer.apple.com>

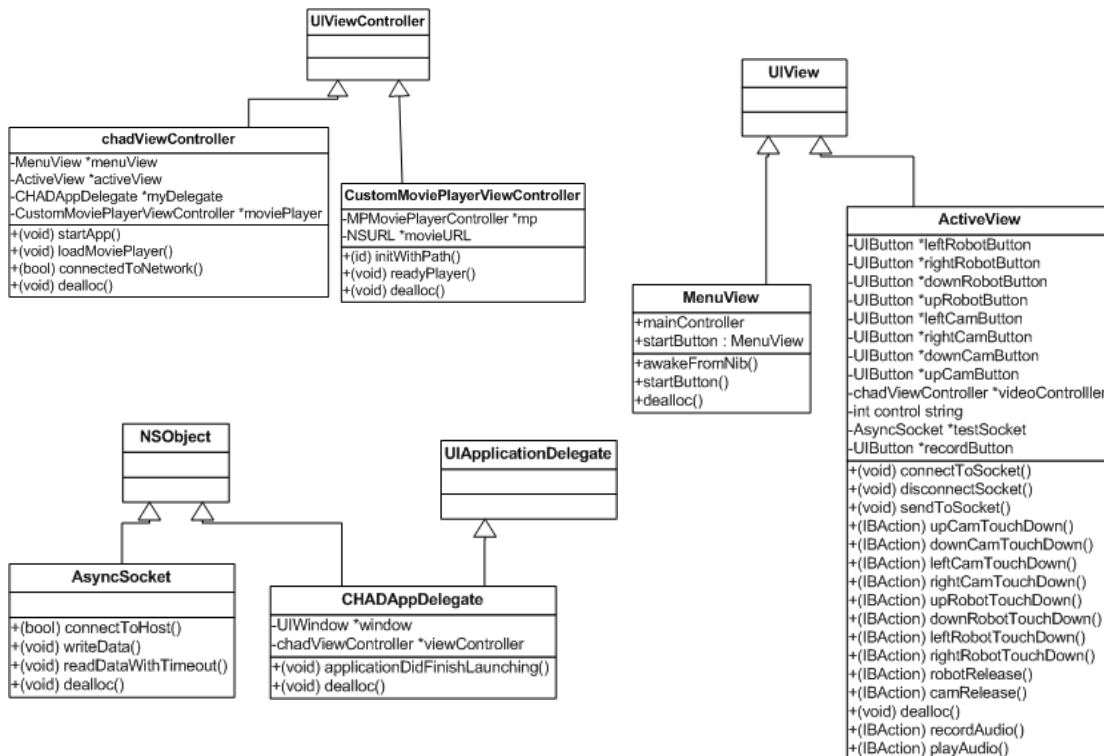


Figure 5.3.5: Class Diagram for CHAD Application

6 BeagleBoard Software

6.1. General Design Guidelines

- Will be using Mercurial Distributed SCM (version 1.4.3) for managing all source code with a repository located on Google code
- Will use Ubuntu 10.04.2 Lucid Lynx operating system with custom Linux 2.6.37 kernel from rcn-ee.net
- Will use Xfce 4.6 as the graphical desktop environment
- Will be developed using the C/C++ programming languages
- Will use gcc 4.4.3 to compile code for the BeagleBoard
- Will use VideoLAN VLC to capture video and audio from webcam and post to URL, as well as to play the audio stream from the iPhone
- Will use Linux socket libraries including functions within socket.h header file for socket communication
- Will use functions from i2c-tools software package for I2C communication
- Will monitor the USER button and safely shutdown when pressed

6.2. Beagle Board Software System Overview

The BeagleBoard software will provide high-level control over the various subsystems of the robotic device, allowing the user to control the functions of the robot through the iPhone application discussed previously. It will capture the audio and video feed from the camera system and post this feed to a URL from where the iPhone application will access it. The movement commands for the camera mount and the robotic device itself, sent from the iPhone over a socket connection, will be forwarded to the Arduino Mega over I2C. The software will play the audio stream sent from the iPhone application as well. Lastly the software will monitor the USER button on the BeagleBoard and shutdown the system when it is pressed.

These four primary functions will be split into four processes summarized in Table 6.2.1 below.



Table 6.2.1: Summary of BeagleBoard Software Processes

| Process | Purpose |
|--------------|---|
| capture | use VLC to capture video and audio feed from the webcam and post this feed to a URL |
| audio | use VLC to receive audio stream and play through the device speakers |
| arduino | receive camera and device movement commands from iPhone software, send movement commands to the Arduino Mega over I ² C protocol |
| safeshutdown | monitor the USER button on the BeagleBoard and issue a shutdown command when pressed to safely shutdown the system |

These processes are discussed in more detail in the following sections.

6.2.1. Configuration

Since the BeagleBoard software will run on the robotic device itself it must do so without any human interaction after appropriate setup. Upon power up of the device, the Ubuntu operating system will load and the default user will be configured to log in automatically. Prior to the device being used on a particular Wi-Fi network, the software will need to be configured to automatically connect to that network by appropriately editing the `/etc/network/interfaces` file. The system will also be configured to start the processes mentioned above once the default user has been automatically logged in.

6.2.2. Webcam Capture

The capture process will first check to make sure that the UVC driver has initialized the webcam properly and has an entry for it in the `/dev` directory. It will then spawn a VLC process with the appropriate parameters to capture the audio/video stream from the webcam and post this stream to a URL via the Wi-Fi connection. The iPhone software will use this URL to obtain the audio/video feed as mentioned previously. This process will listen for the SIGTERM signal and will end the stream and close any connection to the webcam when it is received.

6.2.3. Receiving and Playing Audio

When started, the audio process will first check that the audio output device has been configured correctly by appropriate kernel modules. The process will then setup a socket for receiving the audio data from the iPhone application. Once the data has been received, the audio process will pipe the audio data to a VLC process which will output the audio to the speaker system through the 3.5 mm stereo output jack. This process will listen for the SIGTERM signal, and when received will end the audio playback and close the socket connection.

6.2.4. Forward Movement Data

The arduino process will setup a socket connection and act as the server for the iPhone application to connect to. Upon starting, this process will also initialize the connection over I²C to the Arduino Mega using functions from the i2c-tools software package. The process will remain running and receive movement commands for both the camera mount and the device itself from the iPhone software. These commands will then be translated for sending to the Arduino software over the previously initialized I²C connection. This process will also listen for the SIGTERM signal and will promptly close both the socket connection and the I²C connection.

6.2.5. Software Shutdown

The safeshutdown process when launched will monitor the status of the USER button on the BeagleBoard. When the USER button is pressed this process will issue a `shutdown -h now` command to safely shutdown the system and halt the processor. After this, power from the board may be removed. Rather than constantly poll the USER button status, the safeshutdown process will read the contents of `/dev/input/event0` to be notified of button press events when they occur. This will allow the CPU to continue to execute code for other running processes.

6.3. High Level Software Flowchart

Figure 6.3 below is a high level flowchart for the BeagleBoard software, showing the steps taken from when the BeagleBoard is powered on and the actions taken by each process described above.



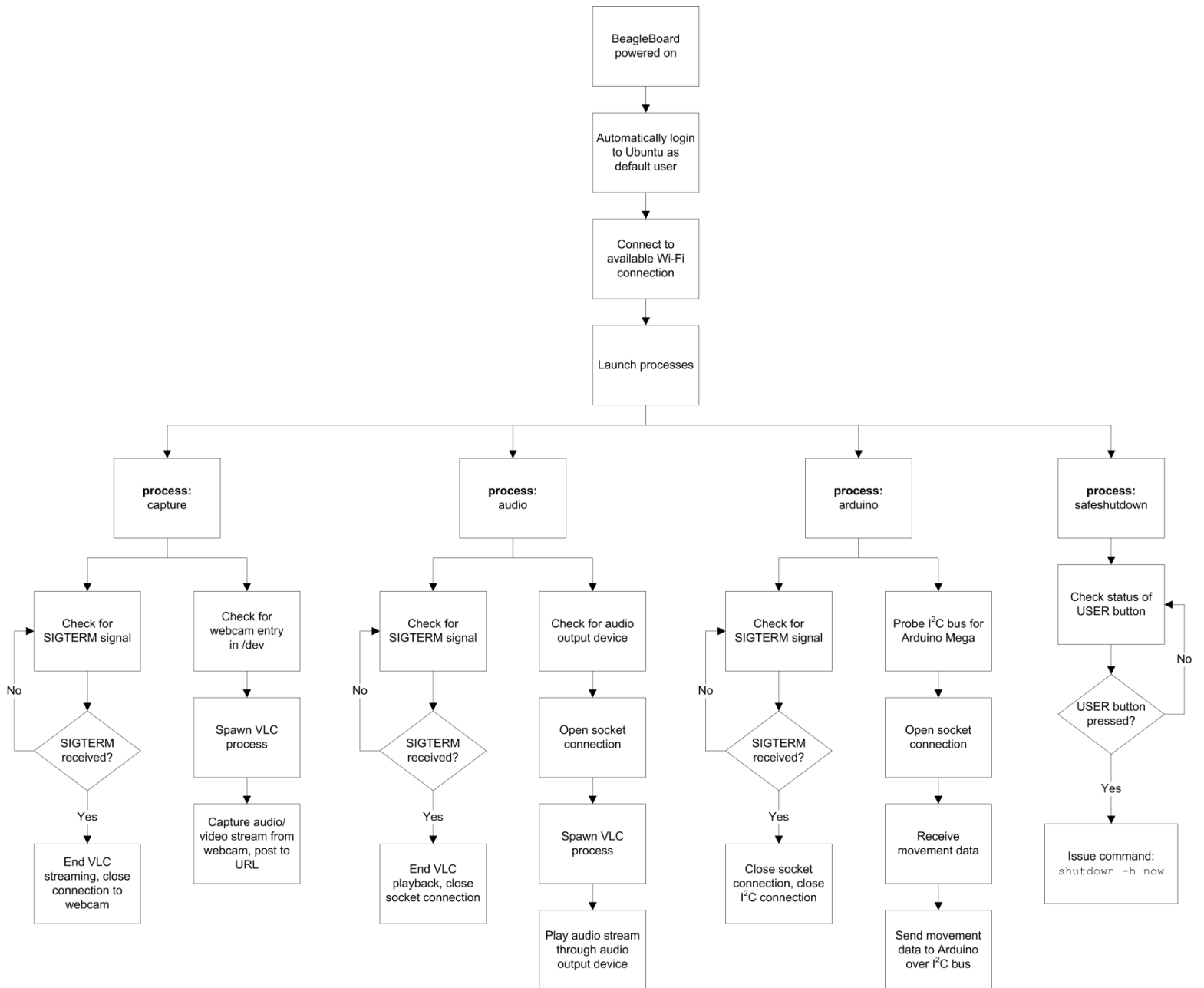


Figure 6.3.1: High level flowchart for BeagleBoard software

7 Arduino Software

The Arduino Mega 2560 is used to control the motion of the robotic device. Acting as a slave, it will communicate with the BeagleBoard to receive commands from the iPhone user, and output signals to the chassis motors and pan tilt system to engage the appropriate motion.

The pins of the Arduino can be seen by looking at Figure 3.4.1.

The Arduino communicates with the BeagleBoard using I²C protocol over a serial connection using pins 20 and 21. Two wires from pins 20 and 21 are connected to 2 GPIO headers on the BeagleBoard.

Once the BeagleBoard receives a command from the iPhone, it will be sent to the Arduino using a 8 bit word. The 8 bits are control instructions for both the chassis motors and the pan tilt camera system. The first 4 bits will control the motion of the robotic device itself, telling CHAD to move left, right, forward, or backward. Similarly, the last 4 bits control the pan tilt camera system by telling it to look left, right, up, or down. The figure below demonstrates how the 8 bits will be interpreted.

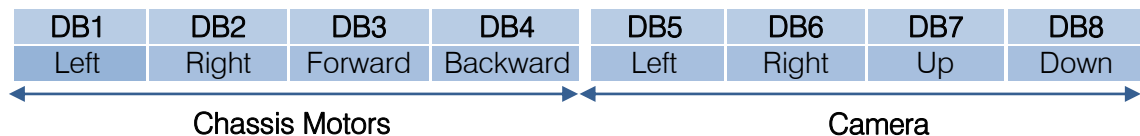


Figure 7.1: 8 Bit Word Map

When DB1 –DB4 are all zeros, the motors will stop, while when DB5-DB8 are all zeros, the camera pan tilt motors will stop moving as well and hold its current position. When DB5-DB8 are all ones, then the camera will move to the center position. The robot can only move in one direction at a time since the iPhone uses a D-pad interface, therefore, of the 4 motor bits (DB1 –DB4), only one can be active at a time. The same applies for the pan tilt camera system, which means that of DB5 –DB8, only one bit can be active at a time. Both D-pads on the iPhone interface can be used at the same time, which means that the camera and the robot can be operated simultaneously.

With having received the iPhone commands, the Arduino must then send output signals to the chassis motors using the digital I/O pins and external circuitry. This is achieved by sending analog voltages to a joystick controller of the wheelchair, to mimic the joystick position. The joystick

controller interprets these voltages (x-y coordinates of a joystick position) and sends this information to the motor controller which converts this data into the appropriate motor speeds and directions. By mimicking the joystick position, we do not have to deal with the complex control of the motors. The chassis motors are controlled by 4 digital 5V pins on the Arduino, pins 22-25. The first 2 pins determine the x and y voltages, and the last 2 pins determine the x and y directions. The table below better illustrates the how the 4 digital pins output the desired x-y coordinates.

Table 7.1: Arduino output to motor circuitry

| 22 | 23 | 24 | 25 |
|----|----|-------|-------|
| x | y | x-dir | y-dir |

Since the device is being controlled by a D-pad, it can only move along one axis direction at a time. This means pin 23 cannot be active while pin 22 is active, and vice versa.

The pan tilt camera system is controlled by 2 servo motors, one of which makes the camera pan, and the other makes it tilt. These servo motors are controlled using 2 PWM signals from the Arduino, pins 2 and 3. The length of the PWM duty cycle determines what location the motor need to move to. Below is a diagram to better illustrate this functionality

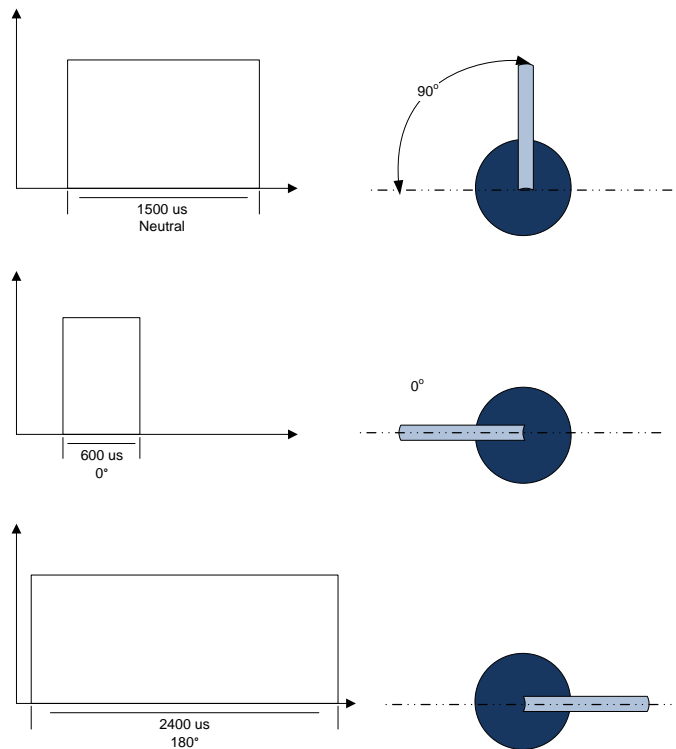


Figure 7.2: Servo motor position vs duty cycle

Below is a summary of the interaction with the beagle, the chassis motors and the pan tilt camera system.

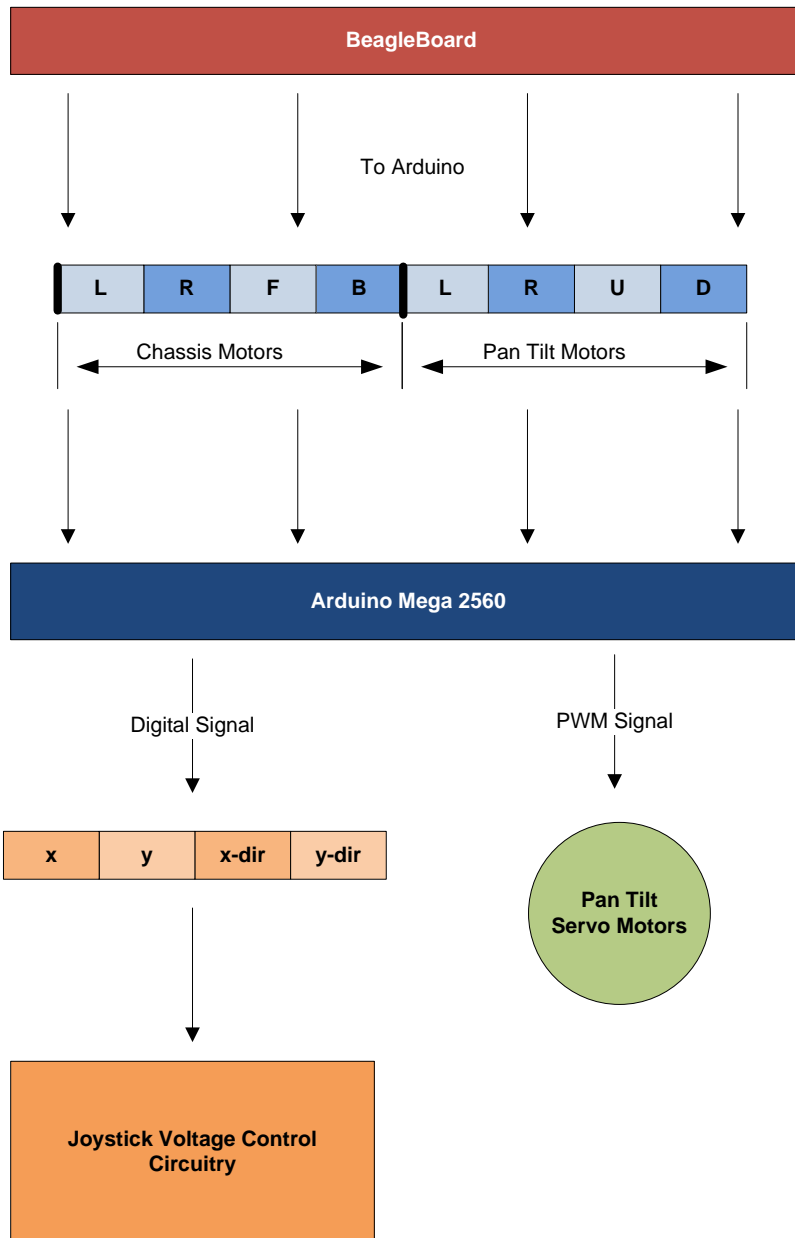


Figure 7.3: Motion Control Flow Chart

The flowchart below shows the complete motor control system software using the Arduino board.

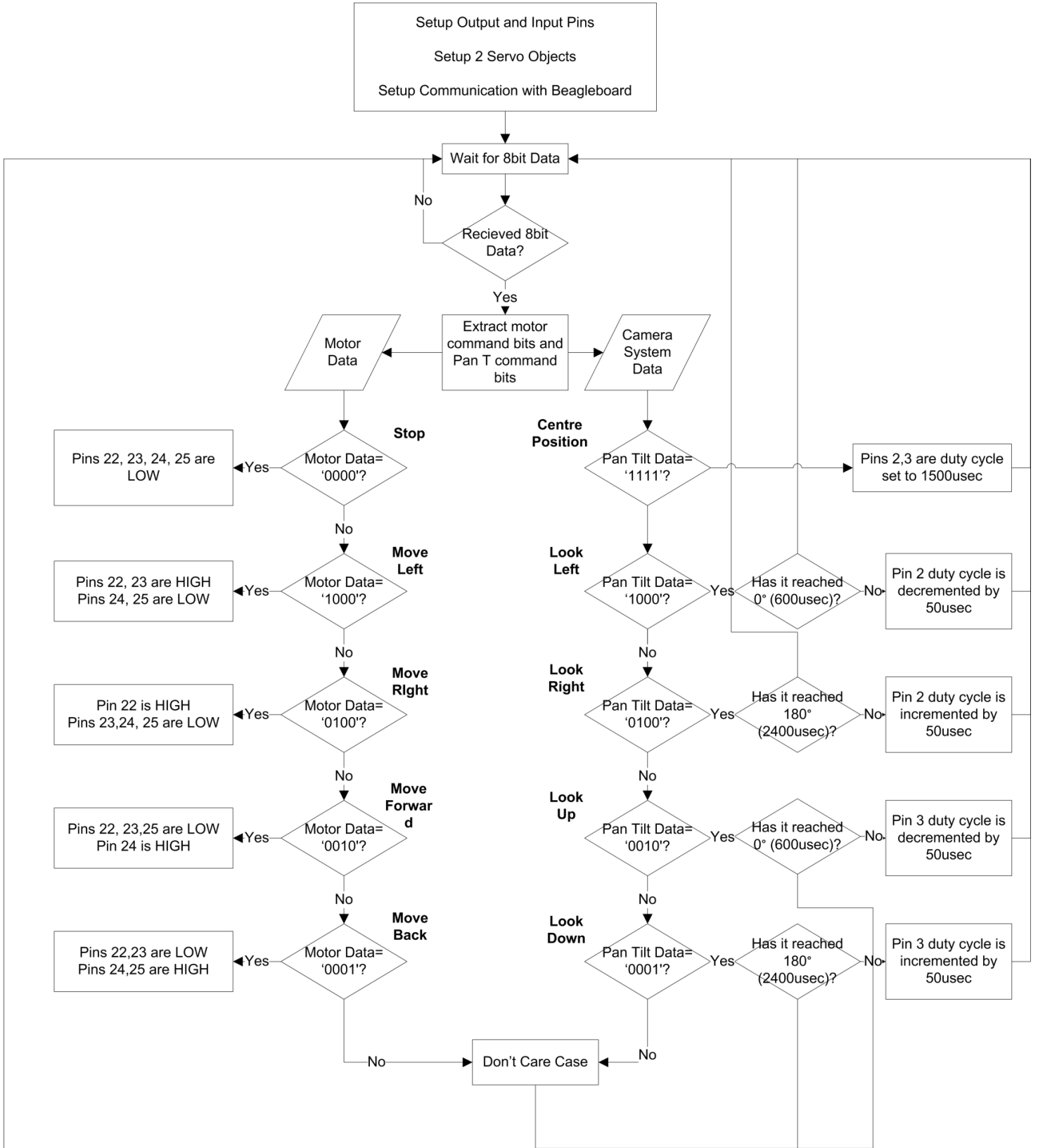


Figure 7.4: Arduino software flowchart

8 Joystick Voltage Control Circuit

As described in the Arduino section, the control of the motor is done by emulating joystick voltages which correspond to the coordinates of the joystick relative to the centre position. The voltage for the center position is approximately 5.76V relative to ground. The table below shows the voltages of the x and y components relative to the centre tap wire (the origin of the x-y plane).

Table 8.1: Joystick Controller Voltages

| | X-voltage | Y-voltage |
|---------------|-----------|-----------|
| Move Left | -1.2 V | 0 V |
| Move Right | 0.85 V | 0 V |
| Move Forward | 0 V | 0.77 V |
| Move Backward | 0 V | -1.07 V |

Since the Arduino board only outputs +5 V from the digital I/O pins, additional circuitry is required. Below is a circuit diagram for the x-voltage output.

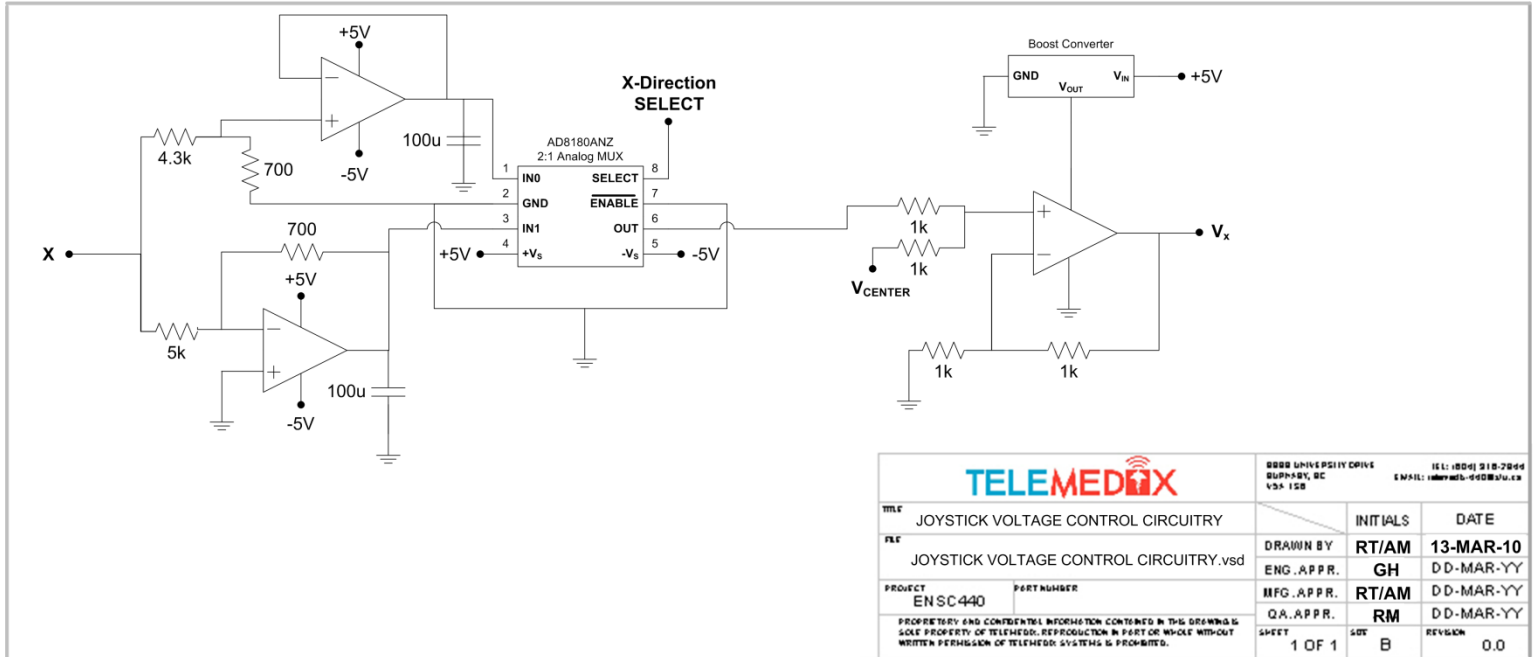


Figure 8.1: Arduino to Joystick Controller Circuitry

As shown from the above circuit diagram, this done by splitting the 5 V signal into two. One signal is inverted and scaled down using a basic inverting op amp, and the other is scaled down using a voltage divider. The signals are scaled down by a factor of 1/5. The resulting signals are then sent into a 2:1 analog multiplexer, where the SELECT voltage comes from the digital I/O pin which determines the sign. A HIGH select signal makes the output of the circuit negative, and a LOW makes the output a positive voltage. The output of the analog MUX is then summed up with voltage of the center position using a summing amplifier, which produces the x-voltage. The circuit for y component is identical to the circuit which outputs the x component of the joystick. In order to operate the motors safely, capacitors are added to the outputs of the op amp so that the output signal ramps smoothly to the appropriate voltage.

Lastly, since we are limited to 5V supply rails from the batteries, the last stage will require power supply rails larger than 5V so that the op amps do not clip the signal. In order to solve this problem, we will use a DC/DC boost converter. This boost converter is an IC that will provided regulated 10V, however the output current will be lower than the input current in order to conserve power.

9 Test Plan

9.1. iPhone Software Test Plan

9.1.1. Unit Test

In order to ensure software that is error free, we will first test each individual component of the iPhone software, to ensure it performs its tasks correctly. The major components of the iPhone software are listed below, as well as all each components major tasks and the steps we will take to test them.

Component: chadViewController

Task: Test Internet connection

Test Steps:

1. Connect to internet
2. Run “connectedToNetwork” function
Expected Result: prints “connected to network” to log
3. Disconnect from internet
4. Run “connectedToNetwork” function
Expected Result: prints “not connected to network” to log

Task: Loads each view (MenuView, ActiveView, HelpView)

Test Steps:

1. Start application
Expected Result: MenuView should be loaded
2. Click “Start” button
Expected Result: ActiveView loads and controls are shown
3. Restart application and touch “Help” button
Expected Result: HelpView loads

Component: CustomMoviePlayerController

Task: loads movie from URL

Test Steps:

1. Run “loadMoviePlayer” function



Expected Result: Audio/Video feed located at given URL should be displayed and start playing

Component: MenuView

Task: displays “Start” and “Help” options

Test Steps:

1. Start application

Expected Result: “Start” and “Help” buttons are displayed

Component: ActiveView

Task: Process D-Pad touches

Test Steps:

1. Start application
2. Press “Start”
3. Touch each separate direction (left, up, right, down) for each D-Pad

Expected Result: Log displays appropriate message for each touch (corresponding to the correct control string)

4. Touch all possible combinations of D-Pad controls, one from each D-Pad. (ie. left camera and right robot, etc)

Expected Result: Correct control string corresponding to the combination of directions being touched is displayed to log. Note that if more than one direction is touched on the same D-Pad, the second direction touched will override the first

Task: Quit button

Test Steps:

1. Load ActiveView
2. Press “Quit”

Expected Result: Application exits to MenuView

Task: Connecting to socket and sending data via socket

Test Steps:

1. Load ActiveView

Expected Result: Server receives connection from ActiveView

2. Press any D-Pad direction

Expected Result: Log shows that write stream was written to and server receives correct data

Task: Recording and sending audio

Test Steps:

1. Load ActiveView

Expected Result: Audio stream is established, iPhone records voice, and server receives voice packets



9.1.2. Integration and Requirements Testing

After all the components have been tested and found to be working correctly, we will then integrate them and test the features that were listed in our requirement document. The integration of the ChadViewController and its corresponding views will have been done during the unit testing, as it was necessary for loading the views in order to test them. The following tests will be done to ensure the integration has brought all the components together and that the integrated system is now able to satisfy all the requirements necessary.

Requirements: Must be able to connect to the internet and software must be able to detect if there is an internet connection

Expected Result: iPhone recognizes Wi-Fi network and connects to it . Software can determine whether or not iPhone is connected.

Test Steps:

1. Start iPhone
2. Connect to Wi-Fi network
3. Start application
4. Press "Start"
5. Verify software recognized that there was an active connection
6. Disconnect from Wi-Fi network
7. Restart application and press "Start" again
8. Software recognizes there is no active connection and prompts user to connect

Requirements: Ability to send information used to control the movement of the camera system and the robotic device itself .

Expected Result: After pressing "Start" from main menu, application will establish connection with our server (using a socket). Then pressing any of the directions on the D-Pad will result in the control string being correctly modified and sent via socket to the server.

Test Steps:

1. Start application
2. Press "Start"
3. Verify connection with server is established
4. Press all possible combinations of D-Pad controls
5. Verify correct strings corresponding to the touch combinations are generated
6. Verify server receives these strings in their entirety

Requirements: Ability to receive live streaming audio and video from the robotic device

Expected Result: After pressing "Start", live stream will be displayed to user automatically

Test Steps:

1. Start application
2. Press “Start”
3. Verify video/audio stream appears to user.
4. Verify video is easily viewable under multiple lighting conditions (based on opinions of people with varying sight, glasses and no glasses, etc)
5. Verify audio is of a quality and that the audio can be easily understood (based on opinions of multiple users, with varying hearing abilities)

Requirements: Ability to record and send live audio to the robotic device

Expected Result: After pressing “Start”, live audio stream will be established and audio data will be sent automatically and continuously to robotic device

Test Steps:

1. Start application
2. Press “Start”
3. Verify audio stream is established
4. Verify audio data is being sent to robotic device
5. Verify audio data is being received by robotic device and that data is not corrupted

Requirements: Must be able to terminate software without damaging hardware of robotic device or that of iPhone

Expected Result: After pressing “Quit”, application will terminate. The iPhone will still be in perfect working order, and the robotic device will as well.

Test Steps:

1. Start application
2. Press “Start”
3. Press “Quit”
4. Verify iPhone is still in proper working order
5. Verify robotic device is still in proper working order, and that no components have been damaged

9.2. BeagleBoard Software Test Plan

Requirements: Must be able to load Ubuntu upon power up and automatically log in to a default user.

Expected Result: BeagleBoard loads Ubuntu and automatically logs in to the default user.

Test Steps:

1. Power up BeagleBoard
2. Verify that Ubuntu loads up properly and that system logs in automatically as default user

Requirements: Must be able to detect presence of Wi-Fi network and automatically connect to appropriate network.



Expected Result: BeagleBoard detects presence of Wi-Fi network and connects to it.

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Once successful log in occurs, verify Ubuntu detects presence of Wi-Fi network and automatically connects to appropriate network.

Requirements: Must be able to automatically initiate capture, audio, arduino and safeshutdown processes and ensure that all four can run simultaneously without conflict.

Expected Result: BeagleBoard initiates capture, audio, arduino and safeshutdown processes and runs all four simultaneously without conflict.

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Verify correct initiation of all four processes
5. Verify all four processes running simultaneously without conflicting with each other

Requirements: Must be able to receive video captured by webcam and post it to URL

Expected Result: BeagleBoard receives video captured by webcam and posts it to URL

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Verify video data sent by webcam is properly received
6. Verify received data is posted to URL

Requirements: Must be able to receive audio captured by webcam and post it to URL

Expected Result: BeagleBoard receives audio captured by webcam and posts it to URL

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Verify audio data sent by webcam is properly received
6. Verify received data is posted to URL

Requirements: Must be able to get iPhone audio stream from socket connection and output audio to speakers



Expected Result: BeagleBoard receives iPhone audio from socket connection and outputs to speakers

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Verify iPhone audio data from socket is received properly
6. Verify received audio data is output properly through speakers

Requirements: Must be able to establish and maintain connection to Arduino over I²C

Expected Result: BeagleBoard establishes and maintains connection to Arduino board

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Verify connection to Arduino is properly established
6. Verify established connection is maintained indefinitely

Requirements: Must be able to send data to Arduino through established connection

Expected Result: BeagleBoard sends data to Arduino over I²C connection

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Wait for connection to Arduino to be established
6. Verify data sent to Arduino through I²C connection is correct

Requirements: Must be able to establish and maintain socket connection to iPhone

Expected Result: BeagleBoard establishes and maintains socket connection to iPhone

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Verify socket connection to iPhone is properly established
6. Verify socket connection to iPhone is maintained indefinitely

Requirements: Must be able to receive movement data from iPhone through socket connection



Expected Result: BeagleBoard receives movement data from iPhone through socket connection

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Wait for socket connection to iPhone to be established
6. Verify movement data sent by iPhone through socket connection is received correctly

Requirements: Must be able to terminate software without damaging hardware of robotic device or that of iPhone

Expected Result: After pressing "USER" button on the BeagleBoard, Ubuntu will shutdown and BeagleBoard can be safely turned off.

Test Steps:

1. Power up BeagleBoard
2. Wait for Ubuntu to load and automatically log in
3. Wait for successful connection to appropriate network
4. Wait for all four processes to be running simultaneously
5. Wait for socket connection to iPhone to be established
6. Wait for connection to Arduino to be established
7. Press "USER" button on the BeagleBoard
8. Verify Ubuntu shuts down properly
9. Verify BeagleBoard powers off properly without causing damage to any associated hardware

9.3. Arduino Software Test Plan and Electronics Test Plan

To test correct operation of the motion control system, we will test the system from the top down. First, we will ensure that the BeagleBoard is sending the correct data. This will be done by making the Arduino feed through the input to 8 output pins without performing any operation. These outputs will be connected to an array of LED's so that the data can easily be read and compared to the intended command. A software test cycle will be developed to loop the data through every possible combination.

Working down the flowchart, the Arduino interpretation of the data will be tested. Since the drive motors operate in an active/inactive manner, 4 LED's can be used to easily see the signal being sent to the motor. The designed circuit will initially be tested individually using the oscilloscopes and a function generator, which will produce a mock input signal. The circuit will be tested again using the signals sent from the Arduino itself.



The pan tilt servo system will first be tested using oscilloscopes to look at the PWM signals being sent to the two servo motors. At this point, we will verify that the drive motor signals and camera mount signals are also operating simultaneously. Once everything is working as expected, the servos will be connected directly to the Arduino. They will be run through a test cycle using the software described earlier.

Next, the circuit will be connected to the joystick controller, and the drive motors will be tested using the same software test cycle. Finally, the complete system will be cycled through the software test plan to ensure that the two subsystems operate properly and simultaneously.

9.4. System Integration Test Plan

In order to integrate the separate components of our device efficiently and effectively, we will integrate components in a specific manner. Since the testing will involve having to integrate certain components, we will integrate them as necessary to perform the tests. This will be done as follows.

The BeagleBoard and its software will be integrated with the iPhone and its software. This will be done by first establishing communication between the two, then by running through a series of test scenarios that will test all possible combinations of data and data types being sent between these devices. This will involve testing things such as all possible combinations of data strings (for controlling the robot and camera movement). While communication between the iPhone and BeagleBoard is being established and debugged, we will in parallel work on integrating communication between the BeagleBoard and Arduino controller. The integration between the joystick controlling circuit and the wheelchair motors themselves will also be done at the same time as its development, as in order to properly develop and test this circuit, the wheelchair motors will have to be involved throughout the whole process. The pan tilt system will be integrated with the Arduino after we have established communications between the Arduino and the wheelchair motors, as these are a higher priority. Once that is established, the Arduino and pan tilt system will be integrated, performing simple movements at first, and working up to more complex ones. Once we have established communication between the iPhone to BeagleBoard and the BeagleBoard to Arduino, we will then integrate them all together, once again performing the same test scenarios that we performed for the previous integrations. Lastly, the motor system (wheelchair and pan tilt) communication will be integrated with the iPhone, BeagleBoard and Arduino, and once again all previous test scenarios will be executed. The other mechanical components will be developed and built independently, since there is no functional relationship between them and the rest of the project. The final integration of these components (including the medical cabinet, table, etc) will be the last part of our project to be fully integrated and completed.



10 Conclusion

This document specifies the design solutions to the Telemedix CHAD system. Details of the hardware components are also included, as well as the reasoning for choosing these components. A working prototype of the robotic device is expected to be available April 10th. In addition to design solutions, a test plan is also outlined to ensure that Telemedix's product meets all the functional specifications.

11 References

- [1] Fracmo Fractional DC Electric Motor Range. *PM62 Fractional DC Motor* [Online]. Available FTP: <http://www.fracmo.com/english/motors/pm62.htm>
- [2] Fracmo Gearbox Range for Fractional DC Motors. *G34 Gearbox for Fractional DC Motors* [Online]. Available FTP: <http://www.fracmo.com/english/gearboxes/g34.htm>
- [3] MK Battery. *8G24* [Online]. Available FTP: <http://www.mkbattery.com/images/8G24.pdf>
- [4] Everest & Jennings, A Graham-Field Brand. *Metro Power III* [Online]. Available FTP: <http://www.grahamfield.com/nosync/productimagesV2/ProductAdditionalInfoItemOriginal2689.PDF>
- [5] Felsen Medical inc. *Products > Motorized Vehicle, Everest & Jennings Lancer 2000* [Online]. Available FTP: http://www.patientmedical.com/images/mo/ej_lancer_2000.jpg
- [6] Vincent Black Shadow Racing. *The Wiring Diagram and Pin Chart for Everest & Jennings Wheelchair Speed Controller* [Online]. Available FTP: <http://www.cyberus.ca/~weavers/shadow/eandj.htm>
- [7] Ranger Board inc. *Premium MDF Specifications* [Online]. Available FTP: http://www.lumberproducts.com/products/PDF_Lit/ranger_specs.pdf
- [8] SERVOCITY. *HS-85BB+ Might Micro* [Online]. Available FTP: http://www.servocity.com/html/hs-85bb__mighty_micro.html
- [9] "BeagleBoard-xM Product Details," *BeagleBoard.org - hardware-xM*, Sept 14, 2010 [Online] Available: http://farm5.static.flickr.com/4068/4644432552_4d3d5d4d29_o_d.png [Accessed: Mar 13, 2011]
- [10] G. Coley, "BeagleBoard-xM Rev A3 System Reference Manual," *BeagleBoard.org*, Oct 18, 2010 [Online] Available: http://beagle.s3.amazonaws.com/design/xM-A3/Beagle_SRM_xM_A3_0.pdf [Accessed: Mar 13, 2011]



- [11] "Wireless G USB Adapter," belkin.com, 2011 [Online] Available:
http://www.belkin.com/IWCatProductPage.process?Product_Id=179211# [Accessed: Mar 13, 2011]
- [12] G. Cohler, "BeagleBoard Shopping List" code.google.com/p/BeagleBoard, Oct 9, 2010 [Online] Available: <http://code.google.com/p/beagleboard/wiki/BeagleBoardShoppingList> [Accessed: Mar 13, 2011]
- [13] "Arduino Mega 2560". Arduino.cc, 2011 [Online] Available:
<http://arduino.cc/en/Main/ArduinoBoardMega2560> [Accessed: Mar 13, 2011]
- [14] "Logitech Quickcam Pro 9000 - Web camera - color - audio - Hi-Speed USB," LifeCam Review: Logitech Quickcam Pro 9000, Mar 2010 [Online] Available: <http://ecx.images-amazon.com/images/I/41cv3v-MB%2BL.jpg> [Accessed: Mar 13, 2011]
- [15] "Logitech Webcam Pro 9000," logitech.com, 2011 [Online] Available:
<http://www.logitech.com/en-us/webcam-communications/webcams/devices/6333> [Accessed: Mar 13, 2011]
- [16] "Logitech USB Video Class (UVC) Device Feature List," quickcamteam.net, Aug 26, 2009 [Online] Available:
http://www.quickcamteam.net/devices/logitech_uvc_device_feature_list_by_device.pdf [Accessed: Mar 13, 2011]
- [17] Wikipedia contributors, "USB video device class," en.wikipedia.org, Mar 9, 2011 [Online] Available:
http://en.wikipedia.org/w/index.php?title=USB_video_device_class&oldid=417994294 [Accessed: Mar 13, 2011]
- [18] "Ultra Thin Speakers", brookstone.com, 2011 [Online] Available:
<http://www.brookstone.com/idesign-Ultra-Thin-Speaker.html> [Accessed: Mar 13, 2011]
- [19] "BeagleJuice", liquidware.com, 2011 [Online] Available:
<http://www.liquidware.com/shop/show/BB-BJC/BeagleJuice> [Accessed: Mar 13, 2011]
- [20] "Mega Backpack", liquidware.com, 2011 [Online] Available:
<http://www.liquidware.com/shop/show/BPM/> [Accessed: Mar 13, 2011]
- [21] Fracmo Fractional DC Electric Motor Range. *PM62 Fractional DC Motor* [Online]. Available FTP: <http://www.fracmo.com/english/motors/pm62.htm> [Accessed: Mar 13, 2011]
- [22] Fracmo Gearbox Range for Fractional DC Motors. *G34 Gearbox for Fractional DC Motors* [Online]. Available FTP: <http://www.fracmo.com/english/gearboxes/g34.htm> [Accessed: Mar 13, 2011]



- [23] MK Battery. *8G24* [Online]. Available FTP: <http://www.mkbattery.com/images/8G24.pdf> [Accessed: Mar 13, 2011]
- [24] Everest & Jennings, A Graham-Field Brand. *Metro Power III* [Online]. Available FTP: <http://www.grahamfield.com/nosync/productimagesV2/ProductAdditionalInfoItemOriginal2689.PDF> [Accessed: Mar 13, 2011]
- [25] Felsen Medical inc. *Products > Motorized Vehicle, Everest & Jennings Lancer 2000* [Online]. Available FTP: http://www.patientmedical.com/images/mo/ej_lancer_2000.jpg [Accessed: Mar 13, 2011]
- [26] Vincent Black Shadow Racing. *The Wiring Diagram and Pin Chart for Everest & Jennings Wheelchair Speed Controller* [Online]. Available FTP: <http://www.cyberus.ca/~weavers/shadow/eandj.htm> [Accessed: Mar 13, 2011]
- [27] Ranger Board inc. *Premium MDF Specifications* [Online]. Available FTP: http://www.lumberproducts.com/products/PDF_Lit/ranger_specs.pdf [Accessed: Mar 13, 2011]
- [28] SERVOCITY. *HS-85BB+ Might Micro* [Online]. Available FTP: http://www.servocity.com/html/hs-85bb__mighty_micro.html [Accessed: Mar 13, 2011]
- [29] Lynxmotion Lynx B-Pan and Tilt Kit [Online] Available: <http://www.lynxmotion.com/p-287-lynx-b-pan-and-tilt-kit-black-anodized.aspx>
- [Cover Page] stock.xchng (July 30, 2007). "838151.jpg," at stock.xchng [Online] Available: <http://www.sxc.hu/photo/838151> [March 09, 2011].

TELEMEDIX

TELEMEDIX
Simon Fraser University,
8888 University Dr.
Burnaby, BC Canada
Email: telemedix-440@sfu.ca

