Indiana University-Purdue University Fort Wayne

Department of Engineering

ENGR 410 - ENGR 411

Capstone Senior Design Project

Report #2

Project Title: Self-Assisted Transfer Apparatus (SATA)

Team Members: Mr. Phuong Le (EE)

Mr. Mengwei Li (EE)

Mr. Andrew Ottenweller (ME)

Mr. Dennis Waters (ME)

Faculty Advisors: Dr. Josue Njock Libii (ME)

Dr. Guoping Wang (EE)

Date: April 28, 2008
# Table of Contents

Acknowledgements........................................................................................................Page 1  

Abstract............................................................................................................................. Page 2  

Section I: Conceptual Design.......................................................................................... Page 3  

Section II: Construction.................................................................................................. Page 18  

Section III: Testing............................................................................................................ Page 27  

Section IV: Evaluation and Recommendations............................................................... Page 35  

Conclusion.......................................................................................................................... Page 37  

References.......................................................................................................................... Page 38  

Appendicies......................................................................................................................... Page 39
Acknowledgements
Team SATAlite would like to extend our gratitude to all of the professors of Mechanical and Electrical engineering for all of the technical knowledge gained and used on this project. In addition, we would like to add a special thanks to our advisors, Dr. Josue Njock Libii and Dr. Gauping Wang for their support throughout this project. We would also like to thank Dr. Jiaxin Zhao for his added support with four-bar linkage analysis and MATLAB code and strain gage application and reading. We would like to thank Dr. Donald Mueller for his added advice and guidance. Additional thanks goes to John Ladd at Studio M for his extensive help on the 3-D drawings, and Ken Farlow from Custom Welding for welding of the frame. We would like to extend special gratitude to John Mitchel at the Engineering Machine shop for his extensive help and work on designing, building and implementation of the structural parts of the final prototype. We would like to think Bob Tilbury for building and troubleshooting our PCB boards. The IPFW nursing department provided a medical geriatric dummy for our durability test and deserve our gratitude. Finally, we want to thank Tim O’Connel at Quality Tool for his sponsorship, vision and guidance on this project as well as the construction and implementation of the final prototype.
Abstract
Through a sponsorship from Tim O’Connell, IPFW has commissioned a redesign of Mr. O’Connell’s original patented design for an in-home self-assisted transfer apparatus to aid disabled patients in getting out of bed in their own homes. Millions of people around the world have debilitating diseases such as Parkinson’s or spinal injuries that make exiting their beds very difficult. Our task was to design a machine that could feasibly function within the confines of a normal room and bed, which would effectively lift a large person out of their bed without causing any further injury. This device must function repeatedly, safely and efficiently. It must also not bend, twist or otherwise become compromised during operation and actuate safely with very little noise. It must use standard home electrical outlets and should not require special assistance to be installed on a standard adult bed.

This report contains the results of the second stage of the design process which includes building, testing and evaluation of the final design. At the conclusion of the design process we discussed our results with our sponsor and determined that engineering the original design with modifications was a better course to take. Thus we went back to the original design which was mounted at the foot of the bed with a ten degree decline and lifted the user through linear actuation of telescoping tubes and springs with a DC electric motor. Problematic areas were chosen to be changed to provide safer actuation and more strength and stability.

The first stage of the building process was to identify problem areas and provide solutions. Areas of interest that were modified from the original design include: a new DC electric motor, the cable and attachment used, rerouting the cable, additional cross support members, a closed motor and electronic section, shortening of upper section, improved actuation of inner sliding tubes, and changing the grab-bar section. These areas would be the main focus for the structure of the final prototype to improve on the original design and will be discussed in depth further in the report.

The next stage was the construction of the prototype from the newly designed parts. The only original parts of the prototype are the frame material and geometry and the actuation process. The original prototype was only used to evaluate areas of improvement and any modifications were done to an extra frame provided my Mr. O’Connel. We designed and constructed new cross members, a motor mount plate, electrical mount plate, bushings, a grab-bar, a motor spindle, pulley brackets, and internal sliding tubes. We then had the cross members welded and assembled all the parts.

The final stages are testing and evaluation of the prototype. After completing the build, we tested the prototype based upon the parameters given in the problem statement. We designed and executed an overall durability test, stress test, and max torque, amp and voltage test under loaded and no load conditions throughout the entire actuation. We then evaluated these results and compared them to our original predictions.
Section I: Conceptual Design
Figure I-1: Diagram of changes to original design.

Figure II-2: Full Solidedge model of conceptual design.
Our conceptual design used for the prototype was based on Mr. O’Connel’s original design and used the exact same frame with modifications. The actuation is linear with a $10^\circ$ decline from the horizontal on the upper frame. The SATA frame will be mounted at the foot of the bed and the user reaches up and grabs a bar. A button is pressed and the motor spins, retracting the cables routed through both upper tubes and into the ends of the inner tubes by the grab-bar, drawing them up the incline until the user releases the button in the sitting position. The user then grabs a stand-assist bar provided to come to a standing position and exit the bed. The grab-bar remains in the up position until the user needs to use it again. A wired remote is supplied that has up, down and a reset function that automatically lowers the grab-bar into place with one push. The up and down buttons are momentary and will only function as long as they are pressed.

This design was chosen to lift a maximum 300 lb, 6ft person into a sitting position in a time of less than 15 seconds. Stress analysis was performed using ANSYS and maximum loading conditions from laying to sitting and sitting to standing for a 300 lb, 6ft user. The maximum stress areas for sitting operation were at the bottom bend and the maximum stress area when using the stand-assist bar is at the joint of the diagonal support member. Kinematic analysis was performed to ensure proper motor torque and accelerations and forces at the hand and shoulder joint do not cause harm. The accelerations were very small due to the slow velocity of the actuation. The force on the hands is 8 lbs and the force on the shoulder is 24 lbs as can be seen in the figures.

![Figure I-3. A plot of the torque required with respect to the angle of rotation.](image-url)
A motor cowling will be implemented to reduce noise during actuation and to protect the exposed moving parts and electrical equipment. This cowling will contain the motor, spindle,
cables and pulleys as well as the backup power unit and all electronic equipment such as control boards, transducer and roller stop sensor. The cowling area will be covered to prevent cuts, burns and collisions from harming the user. The entire metal frame will also be covered with foam as much as possible to reduce harm from shocks, burns, pinches and collisions.

The original design by Mr. O'Connel worked very well but was not analyzed or optimized. We have produced some changes that will make the SATA safer, more reliable and more visually appealing. These changes will also ensure that the SATA can handle larger users and will provide smoother actuation. The frame is 3’ by 4’ by 5’ and attaches to beds as small as a twin and as large as a king. The frame is constructed of AISI 1020 cold-rolled steel with yield strength of 50800 Psi, tensile strength of 60900 Psi, ID of 1.032” and OD 1.125”. We moved the cross members to provide for motor mounting and increase the cowling area to allow for electrical and mechanical components to fit. The entire upper corner will be surrounded by a plastic cowling that will keep moving components safely hidden and reduce noise during operation.

**Motor and Spindle**

The kinematic analysis provided the necessary forces and torque to choose an appropriate motor. The maximum torque required for full motion for a 300 lb user is 301 in-lb. A Bison 348 Series PMDC 24V ¼ hp parallel shaft gearmotor was chosen as it provides 310 in-lb of torque. The motor will be attached to a 1/8" steel mounting plate which will secure it in an upright position to the two supports along the vertical members. The motor will be centered inside of the cowling and will have a 2" aluminum spindle attached to the shaft with a key. The spindle will wind up the cable from both sides of the frame to provide actuation.

*Figure I-6: Exploded View of Bison Motor, Spindle, and Motor Bracket.*
Pulleys and Cable

The original design used twine to pull the two sides of the sliding mechanism. The twine came straight out from the motor spindle through the open area between the frame supports. We decided it would be safer and more appealing if the cable was rerouted through the steel tubing frame. Slots will be machined on both inside upper sections of the frame and attach pulleys that will guide the cable from the motor and spindle through the frame tubes to the End Cap. From an upper limit of tension in the cable of about 200 lb, the cable was chosen as 7x19 strand core wire rope with a breaking strength of 2000 lb. The breaking strength is based on a recommended factor of safety of 10, while the wire rope composition is maximized for flexibility and durability. From the cable specifications, a 2" plastic pulley was chosen and purchased from McMaster-Carr, based upon a design procedure suggested by a pulley manufacturer, Carl Stahl SAVA Industries incorporated.

![Figure I-7: Assembled Pulley and Pulley Bracket.](image)

Spring

To provide a restoring force to the grab-bar mechanism, we decided to keep the same concept as the original and use a spring inside of the upper members of the frame to return the grab-bar to the extended position. The selected spring has an unloaded length of 48”, an outside diameter of 0.65”, and a wire diameter of 0.062”. These parameters result in a spring constant of 125 lb/ft. This spring length was chosen to ensure adequate preload force existed when the grab bar was at full extension. The diameter of the spring was chosen to accommodate the cable running through the spring, the inside diameter of the inner tubing, and the required force to extend the grab-bar.

Grab-Bar

To improve safety and simplicity the swinging grab-bar design was eliminated. The new grab-bar will be a straight piece of steel tubing, attached to the inner sliding tubes using the end cap assemblies. The tube is small enough in diameter to accommodate smaller user's hands and can be outfitted with grips that can rotate to provide a secure handhold.
End Cap Assembly

The next design change was how the inner sliding tube, grab-bar, cable, and spring were attached to the frame. The chosen solution is the End Cap Assembly (ECA). The ECA consists of two pieces: the End Cap Coupler and the End Cap Connector. The Connector acts as a spring stop and secures the cable via the cable stud fitting; it is threaded into the inner sliding tube. The Coupler is a two piece welded element that couples the grab-bar to the inner sliding tube. The coupler is attached to the inner sliding tube via a jam nut.

![End Cap Assembly Diagram]

Stand-Assist Bar

After coming to a stop in the sitting position, the user may require assistance exiting the bed. A Stand-Assist Bar (SAB) has been designed that rotates from resting against the upper frame member through a 120 degree arc, parallel to the floor. The original design consisted of a SAB with a forked end fitting which was mated to a tongue that extended from the frame. To eliminate pinch points and provide a larger stopping area, the location of the fork and tongue components of the joint were switched. The SAB is 12 inches long and will provide enough support for the user to pull them out of bed and into the standing position. Figure I-7 contains both the SAB Fork and SAB Tongue Fittings.

![Stand-Assist Bar Diagram]
**PIC (Peripheral Interface Controller) Microcontroller**

The PIC microcontroller is the brain of the circuit and controls all actions. Microcontrollers contain at least two primary components, random access memory (RAM), and an instruction set. RAM is a type of internal logic unit that stores information temporarily and is cleared after the power is turned off. The instruction set is a list of all commands and the corresponding functions. During operation, the microcontroller steps through the program after receiving valid instructions to do a specific job [6].

The power of the microcontroller is directly related to two parameters, clock speed and the instruction set. The controller operates synchronously with the clock to control the speed and direction of the DC motor. The architecture is a high-performance RISC (Reduce Instruction Set Computer) CPU with 35 single word instructions. A RISC controller has a smaller instruction set for quicker speed and faster execution. The PIC16F877 can operate with a 4, 8, or 20 MHz clock input. Each instruction cycle takes four operating clock cycles and each instruction takes 0.2 µs when the 20 MHz oscillator is used [1]. The block diagram in Figure 4.21-1 shows how the motor will operate.

![Block Diagram of Motor Control](image)

**Figure I-10: Block Diagram of Motor Control**

A reset is used for putting the microcontroller into a known condition. The microcontroller can behave rather inaccurately under certain undesirable conditions. In order to continue its proper functions it has to be reset, meaning all registers would be placed into a starting position. The reset can also be used when trying out a device as an interrupt during the program execution or to get a microcontroller ready when loading a program. The most important reset sources are reset during power on (Power-On Reset) and bring logic zero to MCLR, the microcontroller’s pin. The Power-On Reset occurs each time a power supply is brought to the microcontroller and serves to bring all registers to a starting position initial state. Forcing logical zero to the MCLR pin during normal operation of the microcontroller is often used in program development.
### Specifications
- 16-bit Timers: 1
- 8-bit Timers: 2
- ADC 8-bit Channels: 8
- Core Architecture: PIC16F
- Data Bus Size: 8 bits
- I/O Pins: 33
- Low Voltage Detect: N
- Max Clock Speed: 20 MHz
- Operating Temp: -40 to 85 °C
- Package Details: PLCC
- Pin Count: 44
- Power On Reset: Yes
- Program Memory Size: 14 KBytes
- Program Memory Type: 1
- PWM (9-16bits) Channels: 1
- RAM Size: 368 Bytes
- VDD Max: 5.5 V
- VDD Min: 2 V
- Watchdog Timers: Yes
- Capture/Compare/PWM Modules: 2

### Figure I-11: PIC Microcontroller

**Motor Driver (H-bridge)**

An H-bridge is a circuit that allows DC electric motor to run forward or reverse. The H-bridge has four MOSFETs elements within the bridge, two on each side. If the two switches are turned on at the same time, J1 and J4 are open so that current flows and the motor will rotate in the positive direction. If J3 and J2 are open the current will flow in the opposite direction, reversing the rotation of the shaft. The high power H-bridge from Signal Consulting, LCC operates with a power supply of 9V to 50V at high load current maximum of 20 A.
• PWM from 0-20 kHz for Bi-directional Speed Control.
• Four HexMOSFETs with Heat Sink for 9V to 50V DC Motors (or any Load).
• Max. Continuous Load (or Motor) Current: +/- 20 A at 60 Hz PWM, and +/- 2 A at 20 kHz PWM.
• Max. Surge Load (or Motor) Current for 2 seconds: +/- 80 A at 60Hz PWM, +/- 15 A at 20 kHz PWM.
• The PWM Control Lines are Optically Isolated from the Motor-Power Circuits.
• 100% Solid-State Components (no relays)
• 0 to 5 V (TTL) for Control-Voltage Inputs, with 0 to 100% duty-cycle variation for PWM.

The speed of a DC motor can be controlled by changing the average voltage applied to the input to the H-bridge. A pulse width modulation (PWM) signal is created by switching the output on and off at certain duty cycle. There are two parameters that affect the performance of the PWM, frequency of the pulses and length of pulses (duty cycle). The speed of motor is directly proportional to the DC voltage applied across its terminals. By changing the duty cycle of the PWM signal the average voltage seen by the motor can be varied thus varying the speed. Pulse-width modulation PWM can be used to control the motor with the H-bridge. The motor has a maximum speed of 90 RPM, but the required output speed is 15 RPM, so duty cycle should be at %17 [5].

The PIC16F877 has two Capture/Compare/PWM (CCP) channels modules CCP1 and CCP2, pins 16 and 17. Timer0, Timer1, and Timer2 are counters that increment based on the clock cycle and the timer prescaler. Timer0 is an 8-bit counter but Timer1 and Timer2 are 16-bit.

**Figure I-13: H-bridge Si20HPB4-50V-20A**

**Table I-1: Control Input Truth Table**

<table>
<thead>
<tr>
<th>VHGH</th>
<th>VCG</th>
<th>Motor Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW = 0V</td>
<td>LOW</td>
<td>STOP</td>
</tr>
<tr>
<td>HIGH = 5V</td>
<td>LOW</td>
<td>FORWARD</td>
</tr>
<tr>
<td>LOW = 0V</td>
<td>HIGH</td>
<td>REVERSE</td>
</tr>
<tr>
<td>HIGH = 5V</td>
<td>HIGH</td>
<td>X</td>
</tr>
</tbody>
</table>

**Figure I-14: PWM output**

The PIC16F877 has two Capture/Compare/PWM (CCP) channels modules CCP1 and CCP2, pins 16 and 17. Timer0, Timer1, and Timer2 are counters that increment based on the clock cycle and the timer prescaler. Timer0 is an 8-bit counter but Timer1 and Timer2 are 16-bit...
counters. Timer2 is usually used for PWM or capture functions. When the counter reaches the limit of 255 for the 8-bit and 65535 for 16-bit it will reset back to 0. The Timer2 prescale takes the frequency and reduces by a certain factor, the value can be 1, 4, or 16. TOSC is the time it takes for the clock to oscillate one time. Timer2 is used by CCP1 and CCP2 modules, to configure the pulse frequency by calculating the pulse period formula:

$$PWM_{period} = (PR2 + 1) \times 4 \times TOSC \times (\text{prescaler})$$

$$f_{pulse} = \frac{1}{T}$$

If the PR2 is at 255 and the PIC is running at 20 MHz, then the pulse frequency is 19.53 KHz with a TMR2 prescaler of one [1].

### Motor Control

The simplest method to control the rotation speed of a DC motor is to control its driving voltage. The higher the voltage, the higher speed the motor tries to reach. In the basic PWM method, the operating power to the motors is turned on and off to modulate the current to the motor. The reason is that a motor is mainly a large inductor. It is not capable of passing high frequency energy, and hence will not perform well using high frequencies. Reasonably low frequencies are required, and then PWM techniques will work. Lower frequencies are generally better than higher frequencies, but PWM stops being effective at too low a frequency. The idea that a lower frequency PWM works well simply reflects that the "on" cycle needs to be wide before the motor will draw any current because of motor inductance. To find the applicable frequency, the motor was tested with a square duty cycle using a variable frequency, and then observe the drop in torque as the frequency is increased. This technique can help determine the roll off point as far as power efficiency is concerned.

The speed of the motor is proportional to the voltage supplied to the motor. By varying the voltage across the motor, the speed of the can be controlled. The average voltage of the motor is determined by the PWM generated by the microcontroller. The average motor voltage is the power supply and duty cycle.

$$V_{average} = Duty\ Cycle \times V_{supply}$$

### Sensor and Pushbutton

Sensors can detect and quantify movement, speed, light, and distance. All sensors can be categories, a digital logic level and output an analog result. The design uses a switch to set a limit of travel for the grab bar. The switch is placed on top of the apparatus, to stop the motor when it retracts.
The user can operate the grab bar by pressing the button located on the handle or by using a switch directly connected to the motor. If the motor is not in use for a preset duration, the motor will actuate. After using the handle, it will automatically retract up into the structure.

![Flowchart of Motor Control Program](image)

The push button control is a device that provides control of the motor by pressing the button that opens or closes contacts. This control contains three pushbuttons to actuate the motor forward, reverse, and reset. The sets of momentary contacts are used with push buttons so that when the button is pressed the motor will do a certain task. The button with the up symbol is for moving the handle toward or away from the user. After two minutes, the grab bar will retract to the top of the apparatus for clearance. The pushbutton located on the grab bar can also be used for pulling the user to a desired height remotely.

The pushbutton on the grab bar can be operated by the user to swing the handle. In order to move the handle forward the button must be pressed until the desired height is desired. Once the
pushbutton is release, the motor will stop. This operation gives the user more control over the distance of travel. The pushbutton on the handle is the transmitter that controls the motor by sending a pulse to the microcontroller through the receiver.

**IR Remote Control Transmitter and Receiver**

In order to make an infrared control, this would require a transmitter, encoder, receiver, decoder and a protocol of communication. The transmitter uses an infrared LED to send pulses at a certain frequency. The receiver uses a photo-diode to detect the infrared light traveling through the air. The encoder modulated the signal and the decoder demodulates it from the transmitter.

![Flowchart of remote control](image)

*Figure I-16: Flowchart of remote control*

For the IR remote control transmitter, we have one momentary push button connected to the encoder. The encoder will modulated the signal and later on sent to the receiver by the IR LED.
For the encoder, we are using the Microchip © PIC10F206 microcontroller. It has low-power, high-speed Flash technology, wide temperature range (Industrial: -40°C to +85°C), and wide operating voltage range from 2.0V to 5.5V. Therefore it is durable and can be powered by two AA or AAA which is easy to get and replace.

The IR-D15A is a pre-programmed microcontroller that decodes Sony Corporation infrared protocol. The decoder has 14 momentary outputs and one latching output for remote power control.
The infrared (IR) receiver module is used to read IR information sent by the remote controller. The Panasonic PNA4602M receiver is a 38 kHz modulated source that does not require any other components. The module has a 40 KHz demodulation circuit inside to make signal stand out above noise. The IR light source blinks in a particular frequency and the receiver is tuned to that frequency, so it can disregard everything else. The integrated circuit inside the chip is sensitive to a specific frequency in the 32-40 kHz range. The output of the module is high when there is no IR signal and low when there is a signal.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Rating</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collector supply voltage</td>
<td>$V_{CC}$</td>
<td>-0.5 to +7</td>
<td>V</td>
</tr>
<tr>
<td>Power dissipation</td>
<td>$P_D$</td>
<td>200</td>
<td>mW</td>
</tr>
<tr>
<td>Operating ambient temperature</td>
<td>$T_{opr}$</td>
<td>-20 to +75</td>
<td>ºC</td>
</tr>
<tr>
<td>Storage temperature</td>
<td>$T_{sg}$</td>
<td>-40 to +100</td>
<td>ºC</td>
</tr>
</tbody>
</table>

We are implementing the Sony protocol, Sony SIRC, which is used in Sony devices. The Sony protocol uses pulse-width modulation for a longer bit period bit period. A logic of “1” will modulate the carrier for 1.2 ms and “0” is half as long for 0.6 ms. The carrier frequency is at 40 kHz with the duty cycle of about 25%. While the button on the transmitter is held down it will send a packet every 45 ms and 2.4 ms followed by a 7-bit command and a 5-bit address. This allows the receiver to adjust its gain for varying signal levels.
Power Supply and Back-up Power

A power supply converts alternating current to a direct voltage of 24V to power the electrical components and motor. The Mean Well AC/DC switching power supply has a single output of 24V and a maximum current of 3.2A.

- Input Voltage: 88-264VAC
- Output Voltage: 24V
- Minimum Current: 0 A
- Maximum Current: 3.2 A
- Power: 76.8 W

Figure I-21: Mean Well Power Supply

An uninterruptible power supply (UPS) is a battery backup device that maintains a continuous power supply of electric power to the motor when the utility power is not available. The UPS stays in idle mode until there is power failure, at which time it switches power from the outlet to its own battery source. UPS are inexpensive way to have a continuous power source during a power outage.
Section II: Construction
During the construction of the SATA, some aspects went smoothly while other aspects required re-engineering. This was to be expected and attempts were made to limit the impact the new changes had on the design.

Frame

The frame is the most critical component of the improved SATA. The frame provides structural support when lifting the user and provides a means of actuation. The proposed changes to the frame, machining the pulley slot and shortening the upper tubing, went very smoothly. Shortening the upper frame optimized the actuation length allowing the user to sit up further.
Motor and Spindle

The next modification made was choosing a new motor based upon kinematic calculations of the frame as a four-bar slider-crank linkage. With a loading of 180lbs (from anthropometric measurements the upper body is approximately 3/5 of the total weight) the maximum torque required to rotate into a sitting position was 301 in-lbs, for which we chose a Bison 24V DC gearmotor that supplies 310 in-lbs. The motor is 12 inches long and 4 inches wide, weighs 14 pounds, and has a gear ratio of 215.6:1.

The Motor and Spindle are two of the key components in the actuation of the SATA. The purpose of these two components is to provide power from the motor to the cables as well as a place to store the wound cable. To secure the motor to the frame, it was attached to a 1/8” steel mounting plate. The motor plate itself was secured to the two rearmost supports along the vertical members via 4 UNF ¼ bolts. The motor plate was originally intended to be installed in the center of the frame; however, to fit the electrical components of the SATA, it was repositioned 3.25 inches to the right. Another aspect of the plate that needed modified was the strength. During the heavy load of operation, the plate experienced an undesired twisting flex. A rib was placed vertically on the plate in an effort to reduce the flexing.

Pulleys and Cable

The pulley and cable system was designed to route the cable through the frame members. This was done to improve safety as well for better aesthetics. During construction, the pulley and cable system we designed required very little modifications to be implemented. The pulley pins, pin clips, pulley bracket, and pulley all fit well and operated smoothly. The only enhancement made was the addition of several plastic washers placed between the lower surface of the pulley and the pulley bracket. These were added to center the pulley in the pulley slot and to decrease the friction between the pulley and the bracket. Specific details regarding the size of each of the created components can be seen in the appropriate appendix.

Figure II-2: Motor Mounting Plate with Rib Modification.

Figure II-3: Assembled Pulley and Pulley Bracket.
Compression Spring

The springs were another component critical for proper telescoping action. The cable can only pull the grab-bar in, so a restoring force will be provided by the compression springs for complete actuation. The spring originally selected was a continuous length spring. The selected spring had an unloaded length of 48”, an outside diameter of 0.65”, and a wire diameter of 0.062”. These parameters result in a spring constant of 125 lb/ft. However, a set of custom made continuous springs were not a cost effective restoring force solution. To remedy this problem, a solution was devised that would attach two springs using a spring coupler. The spring coupler needed to allow the cable to smoothly slide through the center while maintaining enough strength and surface area for the spring. The spring and coupler solution worked effectively and showed proof of concept. The specific dimensional details regarding the Spring Coupler can be seen in the proper appendix.

Grab-Bar

The new grab-bar was a straight piece of steel tubing, attached to the inner sliding tubes using the end cap assemblies. The tube is small enough in diameter to accommodate a smaller user's hands and can be outfitted with grips that can rotate to provide a secure handhold. The manufacturing of the grab-bar went without a hitch; the grab-bar freely rotates while fitting snugly against the End Cap assemblies.

End Cap Assembly

The End Cap Assembly (ECA) is a component that was designed to perform several tasks. The ECA is required to connect the Grab Bar to the telescoping tubes as well as provide an anchor point for the cables and provide a stop for the end of the internal spring. To fit these needs, the ECA went through several design iterations before an effective, robust and manufacturable prototype could be built. The original ECA was a two piece assembly consisting of an End Cap Connector and an End Cap Coupler. The End Cap Assembly was designed to be a two piece assembly in order for the grab-bar to be properly attached. A one piece end cap would provide a
cable anchor but not provide the flexibly needed to secure the Grab Bar. By creating a two piece ECA, the End Cap Connector can be screwed tight to the telescoping tube and the End Cap Couplers, with Grab Bar, can then be slid into place and fastened.

![Figure II-5: Original End Cap Assembly](image)

The created End Cap Connector varies little from the final design. The only modification made during the creation of this piece was to alter the flange angle. The original flange angle was designed to be $45^\circ$ but, due to machine restrictions, was changed to $30^\circ$.

![Figure II-6: Final End Cap Assembly](image)

The End Cap Coupler went through more changes than its sister piece, the Connector. The first major obstacle was the compound curves that were designed for stress concentration reduction and aesthetics. The machining capabilities of the IPFW Machine Shop do not include any tooling that would be used to create a filleted surface. To accommodate, all fillets were changed to be chamfers. The second issue was with the complex geometry of a one piece Coupler. Under the suggestion of John Mitchel, we divided the Coupler into two easily machined parts that could be welded together. The final obstacle the Coupler faced was the location where the grab-bar was going to slide onto the Coupler. Due to the two previous modifications, the outer lip of the grab-bar feature could not be flush with the outer surface of the End Cap. This grab-bar feature was moved away from the End Cap to accommodate a chamfer as well as a bead of weld.
**Stand-Assist Bar**

The Stand-Assist Bar (SAB) was modified under the direction of Tim O’Connell. The two components designed by the SATA team were changed primarily for easy manufacturing. Our design called for a smooth radius on the tongue fitting, however, the actual prototype built has a flared section. This was done to stop the rotation of the SAB at a fixed angle. The major change to the fork fitting of the SAB was to remove the rotation stop. The modifications made by Mr. O’Connell essentially moved the material on fork fitting that would cause the SAB rotation to stop at 120° to the tongue fitting.

![Figure II-7: SAB Fork and SAB Tongue Fitting](image)

**Bushing Machining**

The bushings required to allow the telescoping tube to operate smoothly were not available. To remedy this, two sets of oversized oilite bushings were purchased. Two flanged bushings were required to fit snugly with the inner diameter of the outer tubing and allow free movement of the inner tubes. Two non-flanged bushings were needed to fit snugly around the outer diameter of the inner tubing. The outer surfaces of the non-flanged bushing were sized to allow smooth movement of the inner tube. The bushings were machined with the aid of John Mitchel to fit the requirements above. In addition, the left non-flanged bushing was chamfered to accommodate the actuation of the snap action switch.

**Remote Control Housing**

The remote control housing was built to hold the circuit components and to attach them to the grab-bar. The housing had to be designed to allow comfortable operation of the SATA as well. An off the shelf plastic box was selected as the basis for the remote. Modifications to the box were needed before the components could fit and it could be mounted to the grab-bar. To mount the momentary push button to the box, a 5/16” diameter hole was drilled in the center of a face and then notched. A 9/64” hole was drilled in the opposite face so the infrared LED could be attached and send the actuation signal. Finally, a 7/8” diameter hole was machined through the sides of the box so it could be fastened to the grab-bar.

![Figure II-8: Remote Control Housing](image)
Remote and Receiver

Like the motor control circuit, the remote and receiver were first build on the breadboard for test and later on transferred to a prototype board and a PCB. To program the remote, we used MPLAB IDE and PICSTART Plus. To test the remote and the receiver, one LED is connected to the output pin of IR-D15A. Once the receiver module receives the signal, the IR-D15A will demodulate the signal and output a low signal to the output pin so the LED lights up.

After the testing, the remote control circuit will be mounted on the grab-bar and the receiver will be connected to one of the input pin on motor control circuit to drive the motor forward.
Motor Control

The motor control circuit was constructed on a breadboard, a prototype board, and printed circuit board (PCB) shown in Appendix A, Figure A3. The code was tested on the breadboard using LEDs connected to pins CCP1 and CCP2. Other components were also tested on the breadboard. The limit switch is a J-7-V2 Omron ultra subminiature snap action switch with three pins for normally open/close and ground. When the bushings make contact with the limit switch it stops the motor from operating in the forward direction. All of the pushbuttons were used to debug the program.

When the switch closes or opens the contact causes many small voltage spikes. The spikes cause the microcontroller to detect many button events. These spikes are generally present for about 10 milliseconds. In the de-bounce routine, it waits for a button to be pressed and provides a delay after the button is depressed. This prevents the microcontroller from responding to multiple button presses for an individual one.

The program controls the motor speed with a built-in function to output the PWM and duty cycle. MPLAB ICD 2 is a coding environment that can integrate most C language compilers and program the microcontroller. The Custom Computer Services (CCS) C compiler has built-in drivers for the real-time clocks, timers and LCD. The CCS C compiler has a function to activate the PWM at different frequencies by dividing the timer2 by 1, 4 and 16. The duty cycle can be varied by changing the period from 0 to 255 which determines when the clock value is reset.

To prevent multiple buttons from being pressed simultaneously, the microcontroller will only accept one button at a time. If one button is already pressed none of the other buttons cannot become active. When none of the buttons are in use the RTCC, Real Time Counter / Clock, counts to 120 seconds and retracts the handle until it makes contact with the limit switch. After the code was tested, the PIC16F877 was transferred to the PCB.
Printed Circuit Board (PCB)

Both of the remote and motor control boards were fabricated in the Electrical and Computer Engineering Technology Department.

The major steps in creating the PCB design and fabrication process includes:

- Design and test the prototype circuit
- Generated the circuit’s schematic in Multisim
- Perform the physical layout of circuit in Ultiboard
- Fabricate, populate and test the PCB

The circuit design of the motor controller and remote were prototyped and tested to verify that the design operates correctly. Then, the schematic is created in Multisim and transferred into Ultiboard. Using the software, components are placed and routed in the physical layout of the PCB. The Gerber files are created for use in a prototyping system to mill, drill, and cut the PCB substrate. All of the components are placed and soldered to the substrate.

The motor control PCB dimensions are 2.5 x 2.5 inches and mounting holes on the four corners. The board has a 40 and 18 pin through-hole socket for the PIC16F877 and IR-D15A, respectively. The 7805 voltage regulator is also a through-hole component and had to be soldered on the top power plane because of loose connections. The resistors and capacitors are all surface mounts components that are soldered on with flux paste. There are also ports for the IR receiver, limit switch, H-bridge, and pushbuttons. Header pins and crimped wires are used as connectors to the ports.

When testing the PCB, the board has solder bridges between adjacent tracks making a connection where there should be none. The power terminal through hole did not make a connection on the top plane. Before placing the microcontrollers on the PCB, a continuity test was carried out to determine if there are any open connections across any components. There were broken traces on the board from leaving the soldering iron on the board for a long period of time.

When testing the header pins and pushbuttons, the crimped wires did not make secure contact with the header pins. The header pins where removed from the PCB and the wires where wired directly to the board. The microcontroller was then placed on the PCB and CCP1 and CCP2 pins where connected to the oscilloscope to check the PWM output. The traces did not always make contact with the through holes pins of the socket. Also, some of the traces rubbed off from the soldering iron, so soldering bridges were required to make connect. After the PCB was functioning properly it was connect to the 24V power supply, H-bridge, and limit switch. All of the pins were checked with a voltmeter before placing the components onto the board. The pushbuttons were tested to check the response of the motor.
Uninterruptible Power Supply (UPS)

The UPS was disassembled to be mounted on the apparatus. The space is limited, in order to mount the UPS on to the frame; the cover has to be taking off. Once the cover is gone, we need to rewire the wires to all the components that we have. This must be done very carefully making sure there is no wire touching with each other.

Figure II-14: The Uninterruptible Power Supply
Section III: Testing
Durability Testing

Our major test consists of a simple durability test for the entire system of our working prototype. We originally planned for and obtained a geriatric dummy to provide realistic testing but when faced with the task of loading the dummy with 180 pounds, we decided not to risk damaging it. We then built a test setup out of wood that would rotate about two points for the hips and shoulders. We cut and assembled 2X4s to accommodate the 7, 25 lb circular weights (180 lbs with wood and friction effects) and drilled holes for the hands to attach to the grab-bar. We then built a platform that would be placed directly onto the lower part of the SATA frame. We chose to make the height of the shoulder high enough to simulate laying on the bed without actually needing a mattress to make testing easier. To secure the test frame to the SATA and keep it from moving we added a wedge of 2X4 to hold it in place.

We originally planned for durability testing to last for 7300 cycles which is the equivalent of 10 years of use twice a day. Upon further discussion with our sponsor we decided that a 5 year test would be adequate, lowering the cycle count to 3650. One complete cycle lasts approximately 30 seconds from laying to sitting to lying down again. The total test time was approximately 30 hours but was not done continuously. There were a few problems with the code that we used which was timed to lift the test apparatus. We attached a snap-action sensor through a drilled hole near the pulley of the SATA that would detect when the internal tube moved too far and would reverse the motor to automatically let the apparatus back down. Programming this caused a problem with the PIC controller so we reverted to controlling the actuation using precise timing. An LCD screen was attached to the pushbutton sensor which counted every time it was pushed keeping accurate count of the cycles.

Stress measurement

For the stress test we attached strain gages to two areas identified to be high stress areas. The first strain gage was attached to the lower bend area and subjected to the loading of raising the user from laying down to sitting up. The maximum strain was recorded throughout the entire actuation in increments. The first reading we obtained was without attaching the test apparatus to the SATA frame and was -13 µε. After attaching the setup and moving through one cycle, the strain was positive 9 µε. At regular intervals we measured the strain throughout and entire cycle and recorded the highest value displayed. The initial value or 9 µε was subtracted from the values obtained to find the change in strain. This value was then multiplied by the modulus of elasticity (29000 ksi) using Hooke’s law for uniaxial strain to give the final values of stress as plotted per cycle in figure III-1.
The second strain gage was attached to the diagonal brace point which was identified to be a high stress area during use of the stand-assist bar. This gage was read in the same manner as the previous one while using the stand assist bar but was not be durability tested. The test was performed by suspending 180 lb down on the stand-assist bar and reading the maximum value on the strain gage reader. The strain and stress results were assumed to be uniaxial but in reality are not due to the welds causing a moment. The resultant stress was calculated the same as the lower bend area. The strain reading was 1078µε which gave a stress of 31257 psi. This value is much lower than our predicted value of 42367 psi which as we said is due to the fact that the loading was not uniaxial.
Duty Cycle Test

We are using a PWM controller for changing the direction and speed of the motor. Duty cycle at 50% has no net current flow and the motor doesn't move. When testing the motor with a low frequency, the motor outputs an acoustic noise and is sluggish to changes in duty cycles. The waveform at a 10 kHz to 20 kHz operates at a range high enough that the audible noise is attenuated and the switching losses present in the MOSFETs are reduced.

When the PWM is used the motor can generate a whining sound. The motor made a whine when driven frequencies are within the audible frequency range of 20 Hz to 4 kHz. To eliminate the noise the motor must run greater than 4 kHz because the mechanism of the motor winding will attenuate the motor noise to about 4 kHz. The motor operates at a frequency of 19.53 kHz to reduce noise.

\[ F_{PWM} = \frac{F_{osc}}{4(PR2 + 1)prescaler} \]

The microcontroller was connected to a liquid crystal display (LCD) to display the number of cycles. The testing program runs the motor automatically and stops when it reaches 3650 cycles.
When the handle bushing makes contact with the limit switch it will cause the motor to rotate counter clockwise to extend the handle.

We used the durability setup to measure the changes in voltage, current, and speed. The PWM frequency is at 19.53 kHz and the duty cycle is varied to change the speed of the motor. In table III-1, it shows the changes in duty cycle respect to the voltage, current, and speed with no load on the handle. The torque, power, and speed are calculated from the measurements of voltage across the motor. The voltage, current, and speed was measured first with no load on the handles besides the spring force. Different duty cycles were set to change the average voltage and speed of the motor. The speed and torque remain constant throughout the test when the motor reaches 70%.

\[
Torque = \frac{P}{\omega}
\]
\[
P = \text{Power}
\]
\[
\omega = \text{angular velocity}
\]

The speed of the motor was measured by the time it takes the motor shaft to make one revolution, which is about 7.2 seconds. The equation below is used to find the rpm with angular speed.

\[
rpm = \frac{1 \text{ rev}}{7.2 \text{ sec}} \times 2\pi \frac{\text{rad}}{\text{rev}} \times \frac{60 \text{ sec}}{1 \text{ min}}
\]
Table III-1: Duty Cycle with No Load

<table>
<thead>
<tr>
<th>Duty cycle (%)</th>
<th>Voltage (V)</th>
<th>current (A)</th>
<th>RPM (Theoretical)</th>
<th>RPM (Measured)</th>
<th>Power (W)</th>
<th>Torque (no load)</th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>1.27</td>
<td>0.83</td>
<td>4.565</td>
<td>4.5455454545</td>
<td>1.0541</td>
<td>0.231902</td>
</tr>
<tr>
<td>60</td>
<td>13.7</td>
<td>1.06</td>
<td>4.98</td>
<td>5</td>
<td>14.522</td>
<td>2.9044</td>
</tr>
<tr>
<td>65</td>
<td>21.4</td>
<td>1.09</td>
<td>5.395</td>
<td>5</td>
<td>23.326</td>
<td>4.6652</td>
</tr>
<tr>
<td>70</td>
<td>23.16</td>
<td>1.09</td>
<td>5.81</td>
<td>5.5555555556</td>
<td>25.2444</td>
<td>4.543992</td>
</tr>
<tr>
<td>75</td>
<td>23.7</td>
<td>1.09</td>
<td>6.225</td>
<td>8.3333333333</td>
<td>25.833</td>
<td>3.09996</td>
</tr>
<tr>
<td>80</td>
<td>23.8</td>
<td>1.09</td>
<td>6.64</td>
<td>8.3333333333</td>
<td>25.942</td>
<td>3.11304</td>
</tr>
<tr>
<td>85</td>
<td>23.8</td>
<td>1.09</td>
<td>7.055</td>
<td>8.3333333333</td>
<td>25.942</td>
<td>3.11304</td>
</tr>
<tr>
<td>90</td>
<td>23.8</td>
<td>1.09</td>
<td>7.47</td>
<td>8.3333333333</td>
<td>25.942</td>
<td>3.11304</td>
</tr>
<tr>
<td>95</td>
<td>23.9</td>
<td>1.09</td>
<td>7.885</td>
<td>8.3333333333</td>
<td>26.051</td>
<td>3.12612</td>
</tr>
<tr>
<td>100</td>
<td>23.9</td>
<td>1.09</td>
<td>8.3</td>
<td>8.3333333333</td>
<td>26.051</td>
<td>3.12612</td>
</tr>
</tbody>
</table>

The plots are the voltage and speed changes depending on the duty cycle. When the duty cycle reaches 75% the speed and voltage is constant. Since, the handle does not have a load, the current stays constant.

Next, we used the durability setup to measure the changes in voltage, current, and speed. As the duty cycle increases the current through the motor decreases. Similar to the test with no load the speed stayed constant at about 70%. As the duty cycle increased the power and torque increased, shown in figure . The current was recorded once the load has been lifted by the handle. The voltage across the motor was measured with an oscilloscope. The maximum speed is reached quickly at about 14.7V at 70% duty cycle. The recommended duty cycle to lift a person to a sitting position is between 70 to 100%.
### Table III-2: Duty Cycle with Load

<table>
<thead>
<tr>
<th>Duty cycle (%)</th>
<th>Current (A)</th>
<th>Radians/Sec</th>
<th>RPM (Load)</th>
<th>Voltage (V)</th>
<th>Power (W)</th>
<th>Torque (in-lbs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>1.48</td>
<td>0.261799388</td>
<td>2.50</td>
<td>1.8</td>
<td>2.67462</td>
<td>90.4142183</td>
</tr>
<tr>
<td>65</td>
<td>1.46</td>
<td>0.523598776</td>
<td>5.00</td>
<td>7.02</td>
<td>10.25271</td>
<td>173.2939184</td>
</tr>
<tr>
<td>70</td>
<td>1.36</td>
<td>0.872664626</td>
<td>8.33</td>
<td>14.7</td>
<td>20.069175</td>
<td>203.5285876</td>
</tr>
<tr>
<td>75</td>
<td>1.36</td>
<td>0.872664626</td>
<td>8.33</td>
<td>17.2</td>
<td>23.550563</td>
<td>238.834567</td>
</tr>
<tr>
<td>80</td>
<td>1.32</td>
<td>0.872664626</td>
<td>8.33</td>
<td>18.8</td>
<td>24.83104</td>
<td>251.8203413</td>
</tr>
<tr>
<td>85</td>
<td>1.28</td>
<td>0.872664626</td>
<td>8.33</td>
<td>20.0</td>
<td>25.730962</td>
<td>260.946768</td>
</tr>
<tr>
<td>90</td>
<td>1.28</td>
<td>0.872664626</td>
<td>8.33</td>
<td>21.3</td>
<td>27.495437</td>
<td>278.8409267</td>
</tr>
<tr>
<td>95</td>
<td>1.28</td>
<td>0.872664626</td>
<td>8.33</td>
<td>22.5</td>
<td>28.879991</td>
<td>292.8821775</td>
</tr>
<tr>
<td>100</td>
<td>1.20</td>
<td>0.872664626</td>
<td>8.33</td>
<td>23.8</td>
<td>28.7147</td>
<td>291.2059082</td>
</tr>
</tbody>
</table>

*Figure III-6: Volts and Duty Cycle with Load*
Figure III-7: Power and Torque with Load

Figure III-8: Duty Cycle and Torque with Load
Remote control Test
For the remote control test, we tested the distance and angle to verify the range of the receiver. The maximum range for the remote control is when the handle is fully extended with a distance between the remote and receiver of 80 cm. The available angle that the receiver can receive a signal is 30 degrees above the horizontal and 20 degrees below (Show in figure III-8).
Section IV: Evaluation and Recommendations
**Durability Test**

The durability test was designed as a pass/fail type of test. After 3650 cycles the SATA experienced very little in the way of wear and tear and still operates which indicates a pass of this test. At the beginning of every cycle there is a deflection of about 2 inches as predicted by our ANCYS loading that does not appear to have caused any permanent deflections. The cables got caught on the pulleys during initial testing when tension was not maintained causing some of the insulation to be ripped off but otherwise causing no damage. From the durability testing it appears that the SATA is very reliable up to five years and appears to have promising results beyond.

**Stress Test**

The predicted stress from ANCYS analysis was 48 ksi for the lower bend area. The results of the stress test for the lower bend area during the durability test show a maximum stress of 47038 psi at the 5 year mark. The stress has a small but steady increase in this area during its lifetime but is below predicted values and also below maximum stress of 50.8 psi. This stress is also the maximum limit recommended so 95% of users will be below this making it a very viable solution.

The predicted stress for the diagonal brace area was 42367 psi and maximum measurements show a stress of only 31257 psi during testing. This was measured as uniaxial but in reality because of the welds it is not. We have estimated that most of the stress in this area will be from bending so we believe this is a good estimator of stresses the SATA can withstand.

**Cost Evaluation**

Table IV-1 is a summary of the cost of creating the SATA prototype. The cost associated with all of the mechanical components was $646.30, with the majority of it in the 24VDC Bison Motor. Overall, the mechanical components were 77.9 % of the total cost of the SATA. The SATA electrical components cost $183.20; this represents 22.1 % of the total cost of the SATA. The total SATA cost calculated did not include manufacturing cost of the components. All items that were manufactured, including pipe bending and cutting, part machining, welding, and the PCB boards, were donated to team SATA.
Table IV-1: Total SATA Cost.

<table>
<thead>
<tr>
<th>Item</th>
<th>Item Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATA Mechanical Components</td>
<td>646.30</td>
</tr>
<tr>
<td>SATA Electrical Components</td>
<td>183.29</td>
</tr>
<tr>
<td>SATA Manufacturing</td>
<td>-</td>
</tr>
<tr>
<td><strong>Total Cost($)</strong></td>
<td><strong>829.59</strong></td>
</tr>
<tr>
<td><strong>Budget ($)</strong></td>
<td>1500.00</td>
</tr>
<tr>
<td><strong>Over/Under ($)</strong></td>
<td><strong>670.41</strong></td>
</tr>
<tr>
<td><strong>Percent Over/Under</strong></td>
<td>44.69</td>
</tr>
</tbody>
</table>

From the problem statement, the total budget for the project is $1500. As seen in table IV-1, the total cost of the SATA was $829.59. Accordingly, the project came in $670.89 under budget. The final cost of the prototype placed the project 44.7% under budget.

The prototype cost figures should be approached cautiously, however. The prototype was produced at the cost given above, but significant savings were made with donated services. A production run would require manufacturing costs to be included; however as the quantity of SATAs produced increases, the unit cost would decrease.

Full cost evaluation break down can be seen in the cost evaluation section of the Appendix. Complete lists of components purchased for the mechanical and electrical aspects of the prototype are present. In addition, a table of the projected costs of the components manufactured is included. Finally, a tally of total costs for the SATA prototype with projected cost of manufacturing is included.
Conclusions
In conclusion the SATA design that was built and tested is a viable prototype. All areas identified as major problems were dealt with and solved using knowledge garnered from both mechanical and electrical classes and education. The solutions appear to have held during testing even though problems arose and were dealt with. The pulley cables were rerouted resulting in safer actuation. The internal sliding tubes providing linear actuation were made more efficient with bronze oilite bushings and strong springs with internal couplers. The cable and springs were adequately attached to the grab-bar in a very efficient fitting that is also manufacturable. The motor has enough torque to lift a 300lb person into a sitting or near sitting position and the stand-assist bar will support the weight of said user into a standing position. The SATA mounts to the end of any bed and will accommodate users in the 95 percentile of height and weight. The total cost of the prototype is 45% under budget with expected manufacturing costs predicted to be under $1000 with mass production.

The electrical design meets all of the design tasks outlined in the problem statement. All buttons to operate the apparatus are placed at a comfortable location for ease of use. The PIC generates a pulse with given parameters of period and duty cycle to vary the speed of the motor. When testing the PCB for the motor control and remote control, the boards did not operate constantly. The prototype board, in Appendix A, was used in the final design because of time constraints from debugging the PCB. A remote pushbutton on the handle remotely controls the motor to lift the user. Also, a pushbutton station that has a forward, reverse, and reset button is available. The power supply is from a UPS surge protector that can switch to a built-in battery supply during a power outage and will last over 4 complete cycles and can be effectively used in case of emergencies.

Several recommendations would be made to improve upon the final prototype we created. Even though the oilite bushing system we used was effective, a better alternative may be ball bearing bushings. Using this type of busing would create quieter and smoother operation. Another aspect that could be improved would be the finish of the telescoping tube and the frame tubes. The tubing used had a relatively rough finish, especially the frame tube; the rough finish of the tubes created significant friction problems. By creating a grinding, polishing, or buffing the metals the friction present would be reduced. Several frame changes would be made as well. A redundant cross tube could be eliminated that would make the SATA lighter, but remain a rigid structure. Also, strengthening the lower bend of the SATA frame would decrease flexing of the corner. These are the biggest areas of improvement.

The SATA was originally designed by Mr. O’Connel to help his father maintain his self-reliance in the face of a disease that eliminates the ability to even get out of bed. Through engineering and analysis, we feel confident that this prototype has improved upon that design and with further improvements and testing, will be a viable and helpful machine to many. The SATA has been tested and proven that it can be used by a 300lb person to safely get into and out of bed requiring no other assistance from others. Many disabled and handicapped people will greatly benefit from a device such as this which does not exist in the market today.
References


Appendices
Appendix A: Schematic of Motor Control

Figure A1: Schematic of Motor Control
Figure A2: Layout of motor control PCB

Figure A3: Prototype board of Motor Control Schematic
**Appendix B: PIC Code for Motor Control**

```c
#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h>  //C:\Program Files\PICC\Devices

#define ALL_OUT 0x00 //Constant to set data direction register to output
#define ALL_IN  0xff   //Constant to set data direction register to input
#define ints_per_second 76     //(20000000/(4*256*256))

BYTE seconds;
BYTE int_count;

void forward();
void reverse();
void wait();

int rtcc
clock_isr()
{
    if(--int_count ==0)
    {
        ++seconds;
        int_count=ints_per_second;
    }
}

void main()
{
    int_count=ints_per_second;
    set_timer0(0);
    setup_counters( RTCC_INTERNAL, RTCC_DIV_256 | RTCC_8_BIT);
    setup_timer_2(T2_DIV_BY_1,255,1);  //setup TMR2 T2_DIV_BY_1 = 20Mhz/1
    enable_interrupts(INT_RTCC);            // Start RTC
    enable_interrupts(INT_TIMER1);
    enable_interrupts(INT_TIMER2);
    enable_interrupts(global); //set GIE bit
    seconds = 0;
    while(1)
    {
        while(input(PIN_C4))      //top sensor
        {
```
if(seconds >= 5 && seconds <= 8)
{
    output_high(PIN_B4);
    reverse();
}
else if(!input(PIN_C4))
{
    setup_ccp2(CCP_OFF);
    // seconds = 0;
    break;
}
else
{
    break;
}
}
while(!input(PIN_C5)) //top sensor
{
    if (!input(PIN_A0) ) //wait for forward button
    {
        setup_ccp1(CCP_OFF);
    } else
    {
        break;
    }
}
if (!input(PIN_A0) ) //wait for forward button
{
    seconds = 0;
    forward();
    delay_ms(250);
}
else if (input(PIN_B5))
{
    seconds = 0;
    forward();
    delay_ms(250);
} else
{
    setup_ccp1(CCP_OFF);
}
if (!input(PIN_A1)) // wait for reverse button
{
    seconds = 0;
    reverse();
    delay_ms(250);
} else
{
    setup_ccp2(CCP_OFF);
}

void forward()
{
  setup_ccp1(CCP_PWM);
  set_pwm1_duty(254);  // 127.5 / 255 = %50 duty cycle
                   // period = Fosc/(Fpwm*4*T2DIV)-1 //
                   // setup_timer_2(T2_DIV_BY_X,period,1); //
                   //
                   //
                    //
                   // value = (duty_cycle% * Fosc)/(Fpwm*4*T2DIV) //
                   // >>> set_pwm1_duty(value); //

  // delay_ms(300);
  while( (!input(PIN_A0)) && (!input(PIN_A1)) )  //
  {
    setup_ccp2(CCP_OFF);
  }
}

void reverse()
{
  //
  delay_ms(250);
  setup_ccp2(CCP_PWM);
  set_pwm2_duty(254);  // 127.5 / 255 = %50 duty cycle
  //
  delay_ms(300);
  while(!input(PIN_A0) && !input(PIN_A1)) //|| input(PIN_B5)
  {
    setup_ccp1(CCP_OFF);
  }
}
Appendix C: Testing Code

/**SATA Team**************************************************************************
//*Developer: Phuong Le               *
//*Group: SATA Team                  *
//*Purpose: This program is for the durability test, the motor runs in reverse until it hits the limit switch. Then the motor will run forward for 12 seconds and then back in reverse. *
/********************* ****************************************************************************/

#include<16F877A.H> //C:\Program Files\PICC\Devices
#fuses HS,NOWDT,NOPROTECT,NOBROWNOUT,NOPUT,NOLVP //Configuration Fuses
#use delay(clock=20000000) //20Mhz Clock
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include <STDLIB.H>

void forward();
void reverse();
#define ALL_OUT 0x00 //Constant to set data direction register to output
#define ALL_IN 0xff   //Constant to set data direction register to input
#define ints_per_second 76     //(20000000/(4*256*256))
BYTE seconds;
byte int_count;
#define on2 0
int icount; //Note _must_ be int16, since you want more than 8bits...

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

void forward();
void reverse();
#define ALL_OUT 0x00 //Constant to set data direction register to output
#define ALL_IN 0xff   //Constant to set data direction register to input
#define ints_per_second 76     //(20000000/(4*256*256))
BYTE seconds;
byte int_count;
#define on2 0
int icount; //Note _must_ be int16, since you want more than 8bits...

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

void forward();
void reverse();
#define ALL_OUT 0x00 //Constant to set data direction register to output
#define ALL_IN 0xff   //Constant to set data direction register to input
#define ints_per_second 76     //(20000000/(4*256*256))
BYTE seconds;
byte int_count;
#define on2 0
int icount; //Note _must_ be int16, since you want more than 8bits...

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

void forward();
void reverse();
#define ALL_OUT 0x00 //Constant to set data direction register to output
#define ALL_IN 0xff   //Constant to set data direction register to input
#define ints_per_second 76     //(20000000/(4*256*256))
BYTE seconds;
byte int_count;
#define on2 0
int icount; //Note _must_ be int16, since you want more than 8bits...

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>

#include<16F877A.H> //C:\Program Files\PICC\Devices
#include<MATH.h> //C:\Program Files\PICC\Devices
#include<LCD.C>
#include<STDCIO.h>
#include<STDLIB.H>
enable_interrupts(global); //set GIE bit

while(1)
{
    if(!input(PIN_C5)) // wait for limit switch
    {
        if (icount<3650)
        {
            icount = icount + 1;
            delay_ms(250);
        }
        forward();
        delay_ms(12000);
        lcd_gotoxy(1,1);
        printf( lcd_putc," %d ", icount);
    }
    else
    {
        reverse();
        delay_ms(250);
    }
}

void forward()
{
    setup_ccp1(CCP_PWM);
    set_pwm1_duty(250);  //127.5 / 255 = %50 duty cycle
    while(!input(PIN_A0) && !input(PIN_A1) && !input(PIN_B5))
    {
        setup_ccp2(CCP_OFF);
    }
}

void reverse()
{
    setup_ccp2(CCP_PWM);
    set_pwm2_duty(250);  //127.5 / 255 = %50 duty cycle
    while(!input(PIN_A0) && !input(PIN_A1) && !input(PIN_B5))
    {
        setup_ccp1(CCP_OFF);
    }
}
Appendix D: Remote Control Code

FILE: ir_tx_SONY.asm
AUTHOR: Tom Perme
COMPANY: Microchip Technology, Inc.
DEVICE: 10F206
CREATED: 10/08/2006
UPDATED: mm/dd/yyyy

DESCRIP: Application Note example file to illustrate SIRC protocol being transmitted over an infrared LED.

Software License Agreement:

The software supplied herewith by Microchip Technology Incorporated (the "Company") for its PICmicro® Microcontroller is intended and supplied to you, the Company's customer, for use solely and exclusively on Microchip PICmicro Microcontroller products. The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

---

errorlevel -227 ; Ignore substituting pseudo-code "return" for real asm "retlw 0"
#include <p10f206.inc>

; Config Bits
__config _MCLRE_OFF & _WDT_OFF ; MCLROFF = set GP3 as digital input
; WDT_OFF = disable watchdog timer

; System Inputs
#define BTN_CHAN_UP GPIO, GP0 ; Define the two buttons which input the commands to send over IR.
#define BTN_CHAN_DOWN GPIO, GP1

; System Outputs
#define OUTPUT_LED GPIO, GP2 ; Define the OUTPUT LED PIN

; SONY CONSTANTS
#define ADDR_TV d'1' ; Device Address
#define ADDR_CD_PLAYER d'17' ; Device Address
#define CMD_POWER d'21' ; Sony Command
#define CMD_CHAN_UP d'16' ; Sony Command
#define CMD_CHAN_DOWN d'17' ; Sony Command
#define CMD_VOL_UP d'18' ; Sony Command
#define CMD_VOL_DOWN d'19' ; Sony Command

; Register Assignments
#define Delay_Count 0x10 ; Define registers for delay routines (0x10,0x11 for 10F20x)
#define Delay_Count2 0x11
#define DataByte 0x12 ; Define a byte to use for RC5 Data
#define AddrByte 0x13 ; Define a byte to use for RC5 address
#define BitCounter 0x15 ; Variable to hold #bits sent during part of transmission

; PROGRAM CODE
;------------------------------------------------------------------
org 0               ; Processor Reset vector
movwf OSCCAL       ; Load factory calibrated value from reset into osccal
; 10F20x have only 1 reset to org 0
; but this goto Main is for portability
goto Main          ; redirect power on reset to Main label

Main:

Init:

; Initialize port functions and directions
;
bcf CMCON0, CMCONP ; Disable comparator
movlw b'00001000'  ; Enable wake-up on change, pullups, and
option           ; set prescalar to WDT to disable tmr0
movlw b'00000011' ; Define GP0, GP1 as outputs.
tris GPIO        ; Load W into TRIS
bcf OUTPUT_LED   ; Init output low

MainLoop:

; CHANNEL_UP BUTTON
; Check if button is pushed down
btfsc BTN_CHAN_UP  ; button is up, skip stop code
goto skip_chanup   

call DebounceDelay ; Short debounce delay
btfsc BTN_CHAN_UP   ; check if button is up (false indicator)
goto skip_chanup   ; btn is up, skip action code

; Detected the button as pressed. Send keydown code.
;
Button's Action Code
;
; Repeatedly send SIRC transmission while button is held down
chanup_action:

movlw ADDR_TV      ; Load Device Address
movwf AddrByte
movlw CMD_CHAN_UP  ; Load Data byte with command
movwf DataByte
call SendSONY
btfss BTN_CHAN_UP  ; Keep sending code while btn down
goto chanup_action

skip_chanup:

; CHANNEL_DOWN BUTTON
; Check if button is pushed down
btfsc BTN_CHAN_DOWN ; button is up, skip play code
goto skip_chandown 

call DebounceDelay ; Short debounce delay
btfsc BTN_CHAN_DOWN ; check if button is still down
goto skip_chandown ; btn is up, skip action code (false indicator)

; Detected the button as pressed. Send keydown code.
; Button’s Action Code
; 
; Repeatedly send SIRC transmission while button is held down
chandown_action:
movlw ADDR_TV
movwf AddrByte ; Load Device Address
movlw CMD_CHAN_DOWN
movwf DataByte ; Load Data byte with command
call SendSONY
btfss BTTN_CHAN_DOWN
goto chandown_action ; Keep sending code while btn down
skip_chandown:

goto MainLoop ; loop forever

; SUB-ROUTINES

;*************************************************************
; SendSONY
;*************************************************************
; Send the Sony SIRC protocol. Designed to be called over
; and over in a loop while a button is held down.
; Inputs:
; DataByte = 7 bits of data to send in LSBs (data = command)
; AddrByte = 5 bits of device address to send in LSBs
;*************************************************************
SendSONY:
movlw 0
movwf BitCounter ; Clear bit counter to count ones

; SEND PREAMBLE
; Must pulse carrier for 2.4ms 2400us/25us = 96 pulses
movlw d'96' ; -1 These two instr
movwf Delay_Count2 ; -1 are lost overhead time.

CarrierLoopSynPulse:
bsf OUTPUT_LED ; -1 (BEGIN ON TIME = 7us)
goto $+1 ; -2us

goto $+1 ; -2us

goto $+1 ; -2us delayed 7us
bcf OUTPUT_LED ; -1 (BEGIN OFF TIME = 18us)
movlw d'4' ; -1 (Load to finish time accurately)
movwf Delay_Count2 ; -1

decfsz Delay_count, F ; -1

goto $-1 ; -2

; 1 + 1 + 3*N-1 = 18-3 --> x = 4.33
; Choose N=4, and one nop (1 nop = 0.33 of 3cycle loop)
nop
decfsz Delay_Count2, F ; -1 3us tacked on each pulse xcept last one
goto CarrierLoopSynPulse ; -2 TAKE OFF OF ABOVE CALC

; SEND BITS IN SPEED EFFICIENT MANNER (unlooped)

; SEND DATA
; Shift Out DataByte from LSB to MSB
; bit 0
rrf DataByte, F ; Shift out LSB .. C = LSB
btfs STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; NOTE!!
call SendOne ; bit is 1, send a one
btfsc STATUS, C ; NOTE!!
in f BitCounter, F
; NOTE!! testing carry after subroutine
; works ONLY if subroutine does NOT contain
; any add/subtract/rotate commands. These commands augment C flag.

; bit 1
rrf DataByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
inc f BitCounter, F

; bit 2
rrf DataByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
inc f BitCounter, F

; bit 3
rrf DataByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
inc f BitCounter, F

; bit 4
rrf DataByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
inc f BitCounter, F

; bit 5
rrf DataByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
inc f BitCounter, F

; bit 6
rrf DataByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
inc f BitCounter, F

; SEND ADDRESS
; Begin shifting out address
; bit 0
rrf AddrByte, F ; Shift out LSB.. C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C ; bit is 1, send a one
call SendOne ; bit is 1, send a one
btfsc STATUS, C
```assembly
incf BitCounter, F
; bit 1
rrf AddrByte, F ; Shift out LSB, C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C
    call SendOne ; bit is 1, send a one
btfsc STATUS, C
    incf BitCounter, F ; bit 2
rrf AddrByte, F ; Shift out LSB, C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C
    call SendOne ; bit is 1, send a one
btfsc STATUS, C
    incf BitCounter, F ; bit 3
rrf AddrByte, F ; Shift out LSB, C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C
    call SendOne ; bit is 1, send a one
btfsc STATUS, C
    incf BitCounter, F ; bit 4
rrf AddrByte, F ; Shift out LSB, C = LSB
btfss STATUS, C ; if bit is 1, skip next instr.
call SendZero ; bit is 0, send a zero
btfsc STATUS, C
    call SendOne ; bit is 1, send a one
btfsc STATUS, C
    incf BitCounter, F
; Delay remaining time so that repetitive calls of SendSONY
; occur at 45ms intervals (as per SONY spec)
; Quickly clear the output
bcf OUTPUT_LED ; Set output low for off time

; Decide how many ones were sent, and compute time to delay
; Delay time remaining = 45 -2.4 -12*1.2 - N*0.6 where N=#ones
; [every time] [Tx dependant]
; 45 - 16.8ms - N*0.6ms
; Td = 28.2 - N*0.6ms = (47-N)*600us
; Determine number of times to do loop = 47-N
comf BitCounter, F ; Perform "47 - BitCounter"
movlw d'47' ; Result will overflow, but 8-bit
addwf BitCounter, F ; result will be valid.

; Perform variable delay (more 1’s reduces num loops)
movfw BitCounter ; Load #times to loop from above
movwf Delay_Count2 ; into outer loop delay counter
call delay_600us
decfsz Delay_Count2, F ; Go through loop
    goto $-2 ; if count not 0, keep looping

; This delay is reasonably precise because the overhead for
; setting up the loop only adds a few microseconds to the loop
; time each time.  However compared to a 20-28ms delay between
; packets, the few microseconds are negligible.

return
```

------------------------------------------------------------------
SendOne:

; LOW PORTION (600us = 600 instr cycles)
bcf OUTPUT_LED ; Turn off LED
 ; Time Start
movlw d'199' ; +1 us (value = N)
movwf Delay_Count ; +1
decfsz Delay_Count, F ; Loop Eq. = 3*N-1 us
goto $-1 ; 1 + 1 + 3*N-1 = 600
nop ; +1us (Accts for 0.33)

; Time stop = 600us

; HIGH PORTION (1.2ms = 1200 instr)
; Toggle 7us on, 18us off, for: fc = 40kHz, DC = 28%
; These two clock cycles contribute to LOW TIME
movlw d'48' ; -1 (2 addit'l low cycles on low time)
movwf Delay_Count2 ; -1 num pulses counter

CarrierLoopOne:

bsf OUTPUT_LED ; -1 (BEGIN ON TIME = 7us)
goto $+1 ; -2us
goto $+1 ; -2us
goto $+1 ; -2us delayed 7us
bcf OUTPUT_LED ; -1 (BEGIN OFF TIME = 18us)
movlw d'4' ; -1 (Load to finish time accurately)
movwf Delay_Count ; -1
decfsz Delay_Count, F ; -1

goto $-1 ; 1 + 1 + 3*N-1 = 18-3 --> x = 4.33
; Choose N=4, and one nop (1 nop = 0.33 of 3cycle loop)
nop
decfsz Delay_Count2, F ; -1 3us tacked on each pulse except last one
goto CarrierLoopOne ; -2 TAKE OFF OF ABOVE CALC

; DONE Sending a one
return ; -2 return from subroutine

SendOne:

; LOW PORTION (600us = 600 instr cycles)
bcf OUTPUT_LED ; Turn off LED
 ; Time Start
movlw d'199' ; +1 us (value = N)
movwf Delay_Count ; +1
decfsz Delay_Count, F ; Loop Eq. = 3*N-1 us

SendZero:
goto $-1 ; 1 + 1 + 3*N-1 = 600 
     ; N=199.33  --> N=199 + one nop
     ; +1us (Accts for 0.33) 
nop ; Time stop = 600us 

; HIGH PORTION (1.2ms = 1200 instructions) 
; Toggle 7us on, 18us off, for:  fc = 40kHz, DC = 28% 
; These two clock cycles contribute to LOW TIME 
movlw d'24' ; -1 (2 add'l low cycles on low time) 
movwf Delay_Count2 ; -1 num pulses counter 

CarrierLoopZero: 
    bsf OUTPUT_LED ; -1 (BEGIN ON TIME = 7us) 
goto $+1 ; -2us 
goto $+1 ; -2us 
goto $+1 ; -2us delayed 7us 
bcf OUTPUT_LED ; -1 (BEGIN OFF TIME = 18us) 
movlw d'4' ; -1 (Load to finish time accurately) 
movwf Delay_Count ; -1 
decfsz Delay_Count, F ; -1 
    ; 1 + 1 + 1 + 3*N-1 = 18-3  --> x = 4.33 
    ; Choose N=4, and one nop (1 nop = 0.33 of 3cycle loop) 
nop 
decfsz Delay_Count2, F ; -1 3us tacked on each pulse xcept last one 
goto CarrierLoopZero ; -2 TAKE OFF OF ABOVE CALC

; DONE Sending a one 
return ; -2 return from subroutine 

;------------------------------------------------------------------
;  delay_600us
; Precise delay.  Delays 600us including call into and the 
; return out of the call.  
;------------------------------------------------------------------
delay_600us: 
    ; +2 us to enter subroutine on CALL 
    ; Time Start = 0 us 
    movlw d'198' ; +1 us (value = N) 
movwf Delay_Count ; +1 
decfsz Delay_Count, F ; Loop Eq. = 3*N-1 us 
goto $-1 ; +2 + 1 + 1 + 3*N-1 + 2(ret) = 600 
     ; N=198.33  --> N=198 + one nop
     ; +1us (Accts for 0.33) 
nop ; Time stop = 596us
return ; +2 Return program flow 

;------------------------------------------------------------------
;  DebounceDelay
; Quick and simple delay to provide time for bouncing of 
; a button to settle.  The buttons under test have very little 
; bouncing, but it's still good to provide a few usec of 
; debounce time anyway.  
;------------------------------------------------------------------
DebounceDelay: 
    movlw d'3' ; Move 0xFF into w (count, N) 
movwf Delay_Count ; Move w --> Delay_Count 
decfsz Delay_Count, F ; Decrement F, skip if result = 0 
goto $-1 ; Go back 1, keep decrementing until 0 
     ; Loop delay = 3*N-1
return ; Return program flow

; TOTAL DELAY ~12us

;........................................................................

end
### Appendix E: Back-up Power Specification

#### Input

<table>
<thead>
<tr>
<th>Nominal Input Voltage</th>
<th>120V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Frequency</td>
<td>50/60 Hz</td>
</tr>
<tr>
<td>Input Connections</td>
<td>NEMA 5-15P</td>
</tr>
<tr>
<td>Cord Length</td>
<td>6 feet (1.83 meters)</td>
</tr>
<tr>
<td>Input voltage range for main operations</td>
<td>88 - 139V</td>
</tr>
</tbody>
</table>

#### Output

<table>
<thead>
<tr>
<th>Output Power Capacity</th>
<th>200 Watts / 350 VA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Configurable Power</td>
<td>200 Watts / 350 VA</td>
</tr>
<tr>
<td>Nominal Output Voltage</td>
<td>120V</td>
</tr>
<tr>
<td>Waveform Type</td>
<td>Stepped approximation to a sinewave</td>
</tr>
</tbody>
</table>

#### Environmental

<table>
<thead>
<tr>
<th>Operating Environment</th>
<th>32 - 104 °F (0 - 40 °C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Relative Humidity</td>
<td>5 - 95%</td>
</tr>
<tr>
<td>Operating Elevation</td>
<td>0-10000 feet (0-3000 meters)</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>5 - 113 °F (-15 - 45 °C)</td>
</tr>
<tr>
<td>Storage Relative Humidity</td>
<td>5 - 95%</td>
</tr>
<tr>
<td>Storage Elevation</td>
<td>0-50000 feet (0-15000 meters)</td>
</tr>
<tr>
<td>Audible noise at 1 meter from surface</td>
<td>45.00 dBA</td>
</tr>
<tr>
<td>of unit</td>
<td>14.00 BTU/hr</td>
</tr>
</tbody>
</table>

#### Table II-1: Back-up Power runtime chart

<table>
<thead>
<tr>
<th>Watts</th>
<th>VA</th>
<th>50</th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>400</th>
<th>Full</th>
<th>Half</th>
</tr>
</thead>
<tbody>
<tr>
<td>BE350R</td>
<td></td>
<td>24 min</td>
<td>10 min</td>
<td>2 min</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>BE550R</td>
<td></td>
<td>1 h 5 min</td>
<td>28 min</td>
<td>10 min</td>
<td>4 min</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>
# Appendix F: Cost Report

## SATA Mechanical Component Cost

<table>
<thead>
<tr>
<th>Item</th>
<th>Part Number</th>
<th>Vendor</th>
<th>Quantity</th>
<th>Unit Cost ($)</th>
<th>Item Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spring-tempered Steel Compression Spring, 36&quot; Length, .562&quot; Od, .063&quot; WD</td>
<td>9662K43</td>
<td>McMaster Carr</td>
<td>5</td>
<td>-</td>
<td>14.72</td>
</tr>
<tr>
<td>18-8 Stainless Steel Machine Screw Nut, 1/4&quot;-28 Screw Size</td>
<td>91841A215</td>
<td>McMaster Carr</td>
<td>100</td>
<td>-</td>
<td>10.32</td>
</tr>
<tr>
<td>Positive-grip Wire Rope Stud End Fitting, For 1/8&quot; Diameter, Plain Steel</td>
<td>3475T53</td>
<td>McMaster Carr</td>
<td>2</td>
<td>24.60</td>
<td>49.20</td>
</tr>
<tr>
<td>Bowed E-style Retaining Ring, For 1/4&quot; Shaft Diameter</td>
<td>98398A120</td>
<td>McMaster Carr</td>
<td>100</td>
<td>-</td>
<td>7.00</td>
</tr>
<tr>
<td>Clear Plastic Flat Washer, 1/4&quot; Screw Size, .257&quot; Id,.1/2&quot; Od,.056&quot;-.068&quot; Thk</td>
<td>90940A013</td>
<td>McMaster Carr</td>
<td>25</td>
<td>-</td>
<td>11.50</td>
</tr>
<tr>
<td>Sae 841 Bronze Sleeve Bearing, For 3/4&quot; Shaft, 1-1/8&quot; Od</td>
<td>6391K449</td>
<td>McMaster Carr</td>
<td>2</td>
<td>5.74</td>
<td>11.48</td>
</tr>
<tr>
<td>Sae 841 Bronze Flanged-sleeve Bearing, For 7/8&quot; Shaft, 1-1/8&quot; Od</td>
<td>6338K481</td>
<td>McMaster Carr</td>
<td>2</td>
<td>4.21</td>
<td>8.42</td>
</tr>
<tr>
<td>Vinyl-coated Steel Wire Rope, Galv, 7x19, 1/8&quot;-3/16&quot;D</td>
<td>8912T631</td>
<td>McMaster Carr</td>
<td>50 ft</td>
<td>0.78/ft</td>
<td>39.00</td>
</tr>
<tr>
<td>High-strength Nylon Pulley For Wire Rope, For 3/16&quot; Rope Dia, 2-3/4&quot; Od</td>
<td>9466T72</td>
<td>McMaster Carr</td>
<td>2</td>
<td>6.67</td>
<td>13.34</td>
</tr>
<tr>
<td>Bison 348 Series 12 V DC Parallel Shaft Gearmotor</td>
<td>011-348-3200</td>
<td>Bison Motors</td>
<td>1</td>
<td>340.25</td>
<td>340.25</td>
</tr>
<tr>
<td>1 -1/8&quot; Od, 12 Gage, 1020 Steel Tubing</td>
<td>-</td>
<td>Fort Wayne Metals</td>
<td>20 ft</td>
<td>4.98/ft</td>
<td>99.60</td>
</tr>
<tr>
<td>7/8&quot; Od, 18 Gage, 1020 Steel Tubing</td>
<td>-</td>
<td>Fort Wayne Metals</td>
<td>6 ft</td>
<td>3.78/ft</td>
<td>22.68</td>
</tr>
<tr>
<td>1/4&quot; x 1&quot; Hex Bolt Coarse Thread, Zinc Plated</td>
<td>31906</td>
<td>Home Depot</td>
<td>4</td>
<td>0.1</td>
<td>0.40</td>
</tr>
<tr>
<td>6&quot; x 18&quot; Sheet Metal 16 Gauge Plain Steel</td>
<td>47250</td>
<td>Home Depot</td>
<td>2</td>
<td>5.97</td>
<td>11.94</td>
</tr>
<tr>
<td>1/4&quot; x 36&quot; Threaded Rod Coarse Thread Zinc Plated</td>
<td>17320</td>
<td>Home Depot</td>
<td>1</td>
<td>1.37</td>
<td>1.37</td>
</tr>
<tr>
<td>1/4&quot; Nut Hex Coarse Thread Stainless Steel</td>
<td>31906</td>
<td>Home Depot</td>
<td>20</td>
<td>0.07</td>
<td>1.40</td>
</tr>
<tr>
<td>#8-18 x 1/4&quot; Self-tapping Screw, Hex Head</td>
<td>44674</td>
<td>Home Depot</td>
<td>4</td>
<td>0.08</td>
<td>0.32</td>
</tr>
<tr>
<td>3/8&quot; x 1.5&quot; Hex Bolt Coarse Thread, Zinc Plated</td>
<td>33451</td>
<td>Home Depot</td>
<td>4</td>
<td>0.12</td>
<td>0.48</td>
</tr>
<tr>
<td>3/8&quot; Nut Hex Coarse Thread Stainless Steel</td>
<td>16774</td>
<td>Home Depot</td>
<td>4</td>
<td>0.10</td>
<td>0.40</td>
</tr>
<tr>
<td>Thermwell 36&quot; x 3/4&quot; Pipe Insulation</td>
<td>65442</td>
<td>Home Depot</td>
<td>1</td>
<td>2.48</td>
<td>2.48</td>
</tr>
</tbody>
</table>

Total Cost ($) 646.30
<table>
<thead>
<tr>
<th>Item</th>
<th>Part Number</th>
<th>Vendor</th>
<th>Quantity</th>
<th>Unit Cost ($)</th>
<th>Item Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terminal Blocks 5.0MM ECONOMY 2P</td>
<td>538-39890-0302</td>
<td>Mouser</td>
<td>1</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>DIP Sockets 40P DUAL WIPE DIPSKT</td>
<td>517-4840-6004-CP</td>
<td>Mouser</td>
<td>2</td>
<td>0.32</td>
<td>0.64</td>
</tr>
<tr>
<td>.100&quot; Board Mount Connectors 6P 2ROW</td>
<td>517-852-01-06</td>
<td>Mouser</td>
<td>2</td>
<td>1.46</td>
<td>2.92</td>
</tr>
<tr>
<td>Resistors, 47kOhm.5%</td>
<td>71-CRCW0805-47K-E3</td>
<td>Mouser</td>
<td>7</td>
<td>0.05</td>
<td>0.35</td>
</tr>
<tr>
<td>Capacitor, 1.0uF</td>
<td>80-C0805C105M4V</td>
<td>Mouser</td>
<td>3</td>
<td>0.11</td>
<td>0.33</td>
</tr>
<tr>
<td>Switches, PB_SPST</td>
<td>612-TL110SEF100Q</td>
<td>Mouser</td>
<td>6</td>
<td>0.15</td>
<td>0.90</td>
</tr>
<tr>
<td>Resistors, 470Ohm.5%</td>
<td>71-CRCW0805-470-E3</td>
<td>Mouser</td>
<td>3</td>
<td>0.05</td>
<td>0.15</td>
</tr>
<tr>
<td>LED, SML-LX15IC-TR</td>
<td>696-SML-LX15IC</td>
<td>Mouser</td>
<td>1</td>
<td>0.52</td>
<td>0.52</td>
</tr>
<tr>
<td>Capacitor, 2.2uF 25V</td>
<td>581-21063G225Z</td>
<td>Mouser</td>
<td>1</td>
<td>0.32</td>
<td>0.32</td>
</tr>
<tr>
<td>Capacitor, 22 pF</td>
<td>80-C0805C220J1GAC</td>
<td>Mouser</td>
<td>2</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>2 PIN SIL HORIZONTAL GOLD PIN HEADER 2mm</td>
<td>855-M22-2530205</td>
<td>Mouser</td>
<td>2</td>
<td>0.14</td>
<td>0.28</td>
</tr>
<tr>
<td>.100&quot; Board Mount Connectors 4P 1ROW</td>
<td>517-974-01-04-RK</td>
<td>Mouser</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Resistors, 100Ohm.5%</td>
<td>71-CRCW0805-100</td>
<td>Mouser</td>
<td>2</td>
<td>0.04</td>
<td>0.08</td>
</tr>
<tr>
<td>Resistors, 10kOhm.5%</td>
<td>CRCW080510K0FKEA</td>
<td>Mouser</td>
<td>2</td>
<td>0.05</td>
<td>0.10</td>
</tr>
<tr>
<td>Capacitor, 10uF 16 V</td>
<td>80-T4918106K016</td>
<td>Mouser</td>
<td>2</td>
<td>0.23</td>
<td>0.46</td>
</tr>
<tr>
<td>DIP Sockets 18P DUAL WIPE DIPSKT</td>
<td>517-4818-3000-CP</td>
<td>Mouser</td>
<td>2</td>
<td>0.19</td>
<td>0.38</td>
</tr>
<tr>
<td>Small Signal Transistors NPN EPITAXIAL SILICON</td>
<td>512-MMBT3904K</td>
<td>Mouser</td>
<td>2</td>
<td>0.03</td>
<td>0.06</td>
</tr>
<tr>
<td>Transistors Bipolar NPN</td>
<td>863-2N3904G</td>
<td>Mouser</td>
<td>1</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Resistors 1/10watt 22ohms 1%</td>
<td>71-CRCW0603-22-E3</td>
<td>Mouser</td>
<td>2</td>
<td>0.06</td>
<td>0.12</td>
</tr>
<tr>
<td>Resistors 1/10WATT 510HMS 5%</td>
<td>260-51-RC</td>
<td>Mouser</td>
<td>2</td>
<td>0.04</td>
<td>0.08</td>
</tr>
<tr>
<td>2mm M22 Crimp Connectors</td>
<td>855-M22-3050042</td>
<td>Mouser</td>
<td>25</td>
<td>0.13</td>
<td>3.25</td>
</tr>
<tr>
<td>3 PIN SIL FEMALE HOUSING</td>
<td>855-M22-3010300</td>
<td>Mouser</td>
<td>4</td>
<td>0.21</td>
<td>0.84</td>
</tr>
<tr>
<td>Connectors 4 PIN SIL FEMALE HOUSING</td>
<td>855-M22-3010400</td>
<td>Mouser</td>
<td>2</td>
<td>0.30</td>
<td>0.60</td>
</tr>
<tr>
<td>6+6 PIN DIL VERTICAL GOLD PIN HEADER</td>
<td>855-M22-2520605</td>
<td>Mouser</td>
<td>2</td>
<td>0.68</td>
<td>1.36</td>
</tr>
<tr>
<td>Capacitor, 1.0uF</td>
<td>80-C0805C105M4V</td>
<td>Mouser</td>
<td>2</td>
<td>0.11</td>
<td>0.22</td>
</tr>
<tr>
<td>PicMicrocontroller</td>
<td>PIC16F877A-I/A</td>
<td>Microchip</td>
<td>4</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>H-Bridge</td>
<td>SI20HPB4-50V-20A</td>
<td>Signalcc</td>
<td>1</td>
<td>79.99</td>
<td>79.99</td>
</tr>
<tr>
<td>IR-D15A Infrared Decoder IC</td>
<td>IR-D15A</td>
<td>Reynolds Electronics</td>
<td>2</td>
<td>8.00</td>
<td>16.00</td>
</tr>
<tr>
<td>Back-up Power</td>
<td>BE350R</td>
<td>APC</td>
<td>1</td>
<td>10.99</td>
<td>10.99</td>
</tr>
<tr>
<td>Snap-action Sensor</td>
<td>653-J-7-V2</td>
<td>Omron</td>
<td>2</td>
<td>13.27</td>
<td>26.54</td>
</tr>
<tr>
<td>2mm M22 Crimp Connectors</td>
<td>855-M22-3050042</td>
<td>Mouser</td>
<td>2</td>
<td>0.13</td>
<td>0.26</td>
</tr>
<tr>
<td>3 PIN SIL FEMALE HOUSING</td>
<td>855-M22-3010300</td>
<td>Mouser</td>
<td>2</td>
<td>0.21</td>
<td>0.42</td>
</tr>
<tr>
<td>Connectors 4 PIN SIL FEMALE HOUSING</td>
<td>855-M22-3010400</td>
<td>Mouser</td>
<td>2</td>
<td>0.30</td>
<td>0.60</td>
</tr>
<tr>
<td>SMT Connectors 6+6 DIL VERT HEADER GOLD</td>
<td>855-M22-5320605</td>
<td>Mouser</td>
<td>2</td>
<td>0.90</td>
<td>1.80</td>
</tr>
<tr>
<td>SWITCHING POWER SUPPLY</td>
<td>RS-75-24</td>
<td>TRC Electronics, Inc.</td>
<td>1</td>
<td>30.98</td>
<td>30.98</td>
</tr>
</tbody>
</table>

Total Cost ($) 183.29
<table>
<thead>
<tr>
<th>Item</th>
<th>Part Number</th>
<th>Vendor</th>
<th>Quantity</th>
<th>Unit Cost ($)</th>
<th>Item Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame Member</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>2</td>
<td>70.00</td>
<td>140.00</td>
</tr>
<tr>
<td>Cross Member</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>4</td>
<td>20.00</td>
<td>80.00</td>
</tr>
<tr>
<td>Diagonal Member</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>2</td>
<td>20.00</td>
<td>40.00</td>
</tr>
<tr>
<td>Pulley Bracket</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>2</td>
<td>35.00</td>
<td>70.00</td>
</tr>
<tr>
<td>End Cap Coupler 1</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>2</td>
<td>25.00</td>
<td>50.00</td>
</tr>
<tr>
<td>End Cap Coupler 2</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>2</td>
<td>25.00</td>
<td>50.00</td>
</tr>
<tr>
<td>End Cap Connector</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>2</td>
<td>25.00</td>
<td>50.00</td>
</tr>
<tr>
<td>Bushing Machining</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>4</td>
<td>15.00</td>
<td>60.00</td>
</tr>
<tr>
<td>Motor Bracket</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>1</td>
<td>50.00</td>
<td>50.00</td>
</tr>
<tr>
<td>Motor Rib</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>1</td>
<td>15.00</td>
<td>15.00</td>
</tr>
<tr>
<td>Welding</td>
<td>-</td>
<td>Custom Welding</td>
<td>-</td>
<td>-</td>
<td>100.00</td>
</tr>
<tr>
<td>Stand Assist Bar Male Coupler</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>1</td>
<td>70.00</td>
<td>70.00</td>
</tr>
<tr>
<td>Stand Assist Bar Female Coupler</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>1</td>
<td>70.00</td>
<td>70.00</td>
</tr>
<tr>
<td>Stand Assist Bar Cross Member</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>1</td>
<td>50.00</td>
<td>50.00</td>
</tr>
<tr>
<td>Bottom Frame Rails</td>
<td>-</td>
<td>Quality Tool Co.</td>
<td>2</td>
<td>35.00</td>
<td>70.00</td>
</tr>
<tr>
<td>Spring Stops</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>2</td>
<td>25.00</td>
<td>50.00</td>
</tr>
<tr>
<td>Toggle Switch Bracket</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>1</td>
<td>15.00</td>
<td>15.00</td>
</tr>
<tr>
<td>PCB</td>
<td>-</td>
<td>IPFW Electrical Shop</td>
<td>1</td>
<td>180.00</td>
<td>180.00</td>
</tr>
<tr>
<td>Pully Pin</td>
<td>-</td>
<td>IPFW Machine Shop</td>
<td>2</td>
<td>25.00</td>
<td>50.00</td>
</tr>
</tbody>
</table>

Total Cost($) 1260.00
### Total SATA Cost (Projected)

<table>
<thead>
<tr>
<th>Item</th>
<th>Item Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATA Mechanical Components</td>
<td>646.30</td>
</tr>
<tr>
<td>SATA Electrical Components</td>
<td>183.29</td>
</tr>
<tr>
<td>SATA Manufacturing</td>
<td>1260.00</td>
</tr>
<tr>
<td><strong>Total Cost($)</strong></td>
<td><strong>2089.59</strong></td>
</tr>
</tbody>
</table>

### Total SATA Cost

<table>
<thead>
<tr>
<th>Item</th>
<th>Item Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATA Mechanical Components</td>
<td>646.30</td>
</tr>
<tr>
<td>SATA Electrical Components</td>
<td>183.29</td>
</tr>
<tr>
<td>SATA Manufacturing</td>
<td>-</td>
</tr>
<tr>
<td><strong>Total Cost($)</strong></td>
<td><strong>829.59</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Budget ($)</th>
<th>1500.00</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Over/Under ($)</strong></td>
<td><strong>670.41</strong></td>
</tr>
<tr>
<td>Percent Over/Under</td>
<td>44.69</td>
</tr>
</tbody>
</table>
### Appendix G: Stress Readings and Calculations

Modulus in psi

<table>
<thead>
<tr>
<th></th>
<th>29000000</th>
</tr>
</thead>
</table>

**Lower Bend**

<table>
<thead>
<tr>
<th>Cycle</th>
<th>Strain</th>
<th>Stress (psi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td>0.000009</td>
<td>261</td>
</tr>
<tr>
<td>1</td>
<td>0.001557</td>
<td>45153</td>
</tr>
<tr>
<td>100</td>
<td>0.001566</td>
<td>45414</td>
</tr>
<tr>
<td>500</td>
<td>0.001578</td>
<td>45762</td>
</tr>
<tr>
<td>1000</td>
<td>0.001586</td>
<td>45994</td>
</tr>
<tr>
<td>1500</td>
<td>0.001591</td>
<td>46139</td>
</tr>
<tr>
<td>2000</td>
<td>0.001597</td>
<td>46313</td>
</tr>
<tr>
<td>2500</td>
<td>0.001602</td>
<td>46458</td>
</tr>
<tr>
<td>3000</td>
<td>0.00161</td>
<td>46690</td>
</tr>
<tr>
<td>3650</td>
<td>0.001622</td>
<td>47038</td>
</tr>
</tbody>
</table>

**Diagonal Brace**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.001078</td>
</tr>
<tr>
<td></td>
<td>31257</td>
</tr>
</tbody>
</table>
Appendix H: Part Prints

FRAME (RIGHT)

Dimensions:
- 25.606
- 22.135
- R 3.75
- R 2.250
- 80°
- 90°
- R 1.875
- 50.625
- 27.000
- 33.324

Other notes:
- SOLID EDGE
DIAGONAL SUPPORT TUBE
TELESCOPING TUBE
END CAP CONNECTOR

- Ø .715
- Ø .313
- Ø .881
- R .375

Dimensions:
- 625 mm
- 680 mm
- 780 mm
- 979 mm
END CAP COUPLER 1

ø 997
ø .312
ø .750
R .124
1.127
.709
.397
END CAP CONNECTOR ASSEMBLY

1.37
MOTOR MOUNT BRACKET, BENT
SPRING STOP

Ø 1032

Ø 870

Ø 450

250

500