Final Design Report
for
Project RANDOM

By Team Random:
Mark Cummings, Eric Gentry, Joey Machala,
Cameron Meredith, Collin Petersen, and
Nick Podgursky

A Senior Design Project at the University of Idaho
During the 2005–2006 Academic Year
Executive Summary

This paper is the final report for a project called Robotic Arm for Neuro-Digital Optimization of Movement (RANDOM). The project is a continuation of a project lead by Dr. Terence Soule of the University of Idaho. The goal of Dr. Soule’s project is to develop and refine a model of the human spino-neuromuscular system (SNMS). The spino-neuromuscular system is the collection of neurons that connect the brain to all of the muscles.

To accomplish their goal, Dr. Soule’s group created software to simulate the electrical connections between the neurons in the SNMS, a biceps brachii muscle, and the bones for a human arm. To see if the simulation of the neurons could drive a real arm instead of a simulation, Dr. Soule contracted us to build a robotic arm.

Our arm can be used to show, very clearly, how the changes in the spino-neuromuscular system affect movement. Although the arm and the software are a learning tool at this juncture, the things we can learn from this system could have practical applications in physical therapy, intelligent robotics, nonlinear control systems. In the distant future, this information could also help develop smart artificial appendages and neural interfaces with computers.

Economically, the project has been very efficient. We spent less than $3000 building the arm. All of the parts except the aluminum bone structure were off the shelf components. The aluminum bone structure was machined with help from the Mechanical Engineering Department at the University of Idaho.

Dr. Soule’s software and RANDOM’s arm can bring us all closer to understanding how the human body works, and show us how we can use that knowledge to our advantage.
## Contents

1 Background

2 Problem Definition

2.1 Product Requirements
2.1.1 Physical Components
2.1.2 Software
2.1.3 Documentation
2.1.4 General
2.2 Constraints

3 Concept Development

4 Product Description

4.1 Bone Structure
4.2 Muscle Fibers
4.3 Sensors
4.4 Electronics
4.5 Operation Overview

5 Product Evaluation

6 Economic Analysis

7 Conclusions, Recommendations, and Future Expansion

A Parts List

B Mechanical Drawing Package

C Design Failure Mode and Effect Analysis

D Electronics Schematics

E Microcontroller Code

F Protocol to Communicate with the Microcontroller
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Project RANDOM's Arm</td>
<td>ii</td>
</tr>
<tr>
<td>2</td>
<td>Block Diagram</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>The Arm</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>The Shoulder Components</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>The Elbow Components</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>The Circuit Boards</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>The Power Supply</td>
<td>10</td>
</tr>
</tbody>
</table>
1 Background

This project is named RANDOM which stands for Robotic Arm for Neuro-Digital Optimization of Movement. The artificial arm is an extension of a research project headed by Dr. Terence Soule of the University of Idaho. Dr. Soule’s project is called Evolutionary Training of a Biologically Realistic Spino-neuromuscular System.

Our team is made up of six seniors from the College of Engineering at the University of Idaho. Of the six of us, three are computer engineers, two are biological and agricultural engineers, and one is a mechanical engineer.

Dr. Soule’s project is documented on a web site. Its address is http://www.cs.uidaho.edu/~tsoule/website_with_hierarchy/index.html. As stated on the web site, the goal of the project is to develop and refine a model of the human spino-neuromuscular system (SNMS). The SNMS connects neurons in the brain to all the muscles in a human body and helps stabilize movement. An understanding of how the system works could help in a number of fields including injury rehabilitation and non-linear control. It could also help distribute the computational load of current control systems.

One of the products of Dr. Soule’s project is a piece of software. The software consists of a neural network that accurately models the SNMS and a simulated arm the neural network can control. The input to the program is a series of angles. Each angle represents a desired angle for the arm at a given time. The output of the program is a set of weights for the neural network that comes closest to providing the desired movement. The software uses a genetic algorithm to find the weights.

To increase interest in the project and explore further uses of the network, Dr. Soule contracted us to build a mechanical arm and interface. The arm will prove that the software can work on a noisy system and allow Dr. Soule’s group to see how the changes in the neurons affect movement.

2 Problem Definition

2.1 Product Requirements

The requirements of the project can be broken into physical components, software, documentation, and general.

2.1.1 Physical Components

- Build a one degree of freedom artificial bone structure that vaguely resembles a human arm. It must include a humerus, an elbow, and an ulna. This bone structure must withstand the process of training which may last for days. Training may also take place with up to a kilogram of weight at the wrist.

- Build an artificial muscle structure that vaguely resembles a human biceps brachii with six independently controlled muscle fibers. This artificial biceps brachii must also withstand training. The example of training we were given is a 145° change in ulna position every 20 minutes for several days.
• Incorporate three sensors. One to measure the angle of the ulna relative to the humerus. One to measure the change in length of the muscle fibers, and one more to measure then tension on the artificial tendon.

• Build an electronic interface between the computer and the muscles. This interface must be able to take commands from the computer that involves contracting and relaxing the muscle fibers. Also it must send back the values of the three sensors.

• Provide an appropriate energy source to power all of these components.

2.1.2 Software

• Modify the software, written in Java, to allow the neural network to control the arm. Also modify the simulation of the arm, so it more closely represents the physical arm. The more accurate simulation will be used to pretrain the arm in software.

• Write any software need for the electronic interface. This software should be generalized so it does not need to be modified very often. Parameters like the timing should be set by the software on the computer.

2.1.3 Documentation

• Write an operating manual if operating the software or the physical components is not trivial.

• Write a final report that includes a product analysis, an economic analysis, and a parts diagram/list.

• Write a reasonable amount of comments in the code so future people working on the software can understand our work.

2.1.4 General

• Make the physical components small enough so a single person can carry it on long trips.

• Make physical components reasonably aesthetically pleasing because the arm will be used as a demonstration tool.

• If the arm requires an external power source, it must be easy to use and commonly available. For example, if the arm requires electric power, it must use standard 110V AC power outlets, and not draw more current than is usually available.

2.2 Constraints

There were very few constraints impressed upon us. This is because Dr. Soule only wanted one arm and we planned on mostly using off the shelf products to build the arm. Also, the arm didn’t pose a environmental or ethical threat. Safety was of slight concern because we
knew that the arm was going to consume a fair amount of power to raise the arm. We agreed that we would put effort into testing for safety hazards and correct them. The only other constraint we had was our $4000 budget.

3 Concept Development

We started out development with a survey of previous solutions. Because Dr. Soule attended most our meetings, we were able to present these previous projects on a weekly basis and discuss the advantages and disadvantages of each one. After everyone felt confident we had a general grasp of the previous solutions, we created a block diagram (Figure 2) that listed out the components we would need.

![Team RANDOM Block Diagram](image)

**Figure 2: Block Diagram**

We found that most of the components could easily be purchased and integrated; however, off the self solutions for the bone structure were not appropriate. Since the rest of the components will be built on and around the bones, we started discussing how to build them first. Some of the options considered were:
• Plastic skeletons intended for science class rooms
• Real human bones
• Plastic bones with rubbery material around the elbow that looked like ligaments
• Machining our own metal bones

We considered the situation for a while. The plastic bones for science classes had a simple pin about a millimeter in diameter as a hinge. This would have never lasted up to the training. The real human bones did not come with any type of hinge, and we felt could be inappropriate for moral and ethical reasons. The plastic bones with rubber ligaments, seemed to pull apart easy and would have been hard to attach anything too without destroying strength of the bones. The only option left was metal bones.

The next topic we discussed was how close the metal bones should be to human bones. We agreed that the closer we made the bones to human bones, the longer and more expensive the solution was going to be. Dr. Soule did not have high expectations in this area, so we decided only the lengths of the bones should be human like. The shape of the bones, and how they came together at the joint, could be made in a way more appealing to machining time and ease of use.

Now that we had decided we could make the bones to fit the other components, we started to look at the muscle fibers. Two technologies, air muscles and muscle wire, seemed most feasible. We focused our time on those two options.

Air muscles are rubber tubes wrapped in a nylon mesh that stretch diameter wise when filled with air and shrink lengthwise. They had the advantage of looking like real muscles and contracting about the same as real muscles. The downside to air muscles is that they require compressed air. To supply the air for any length of time would require an air compressor. This is a large and noisy device which is not appropriate for demonstrations. Ultimately, the air compressor dissuaded us from using air muscles.

Muscle wire also acts very much like muscles. It is made out of a nickle-titanium alloy sometimes known as flexinol, and comes in strands about the diameter of a human hair. When enough current is passed through the wire to heat the material, the wire will shorten. It can “contract” about 5% of it’s length repeatedly. This was a much better solution for demonstrations because the wire can be powered by a wall outlet. The design was a little more challenging for us because the wire does not contract as much as real human muscles (up to 30%).

To compensate for the fact that muscle wire has a small contraction percentage, we investigated different methods of looping the wire back on itself. The first method we considered was running the wire over a pulley at the top of the humerus and connecting the wire to the bottom of the humerus. The second way was to put pulleys at the top and bottom of the humerus and run the wire up and down the pulleys eventually connecting at either the top or bottom of the pulleys. After a few calculations, we decided that our first method of connecting the muscle wire would provide the range of motion we were looking for.

After experimenting with the muscle wire a little bit, we found that it took a long time for the wire to relax. To help speed the cooling process, we decided to add some sort of airflow across the wire.
At this point we realized the other components could be bought and integrated with very little modification. The electric circuits could be soldered onto a development board and the bone structure could be custom built to incorporate the sensors. We spent a few weeks finding and discussing potential components. When we collected a significant number of options for each component, we developed criteria for picking the components. Then we assigned components to group members. Each group member went through all their assigned options for that component and ranked them based on predetermined criteria.

The criteria we used were:

- Availability
- Development Time
- Durability
- Portability
- Power Consumption
- Ease of Use
- Price
- Appearance

The items we ended up selecting are listed in Appendix A.

After selecting the components, we had two last topics to discuss before starting the design of the bones and the electric circuits. First, we wanted to make sure that some sort of protection scheme was in place. We didn’t want the muscle wire to over-contract and try to pull the arm apart. We decided that some sort of switch to disconnect the power from the wires was needed, but the switch would not need to be automated. Second, we wanted the arm to look good for demonstrations. We decided to buy a mannequin torso to mount the arm on.

4 Product Description

Figure 3 shows a wide view of the finished arm. In order to explain all of the components, we will break the arm into subsystems generally as shown in Figure 2. Throughout the descriptions, the different components will be referred to by their biological counterparts.
4 PRODUCT DESCRIPTION

4.1 Bone Structure

The bone structure was custom designed to support the muscle fibers, support the three sensors, and securely mount to a stand. A detailed CAD drawing package of the bone structure with dimensions is included in Appendix B. Almost all the parts of the bone structure is made of machined aluminum. To see the exact composition of each piece, look in appendix B.

Starting from the top of the bone structure, when mounted, you can see the shoulder components. Figure 4 shows two views of this area. The components include two plates that bolt through the humerus. The larger of the two plates allows the arm to be attached to a stand. The smaller of the two plates has holes to mount the length sensor.
Figure 4: The Shoulder Components

The humerus extends below the shoulder plates, and bolts to the elbow components. Near the top of the humerus, there is a round piece of plastic with six groves and a plate. The muscle fibers run through these groves, and the plate helps keep the fibers from popping out if the fibers go slack.

Figure 5: The Elbow Components

The elbow components connect the ulna and tension sensor to the humerus. You can see this area in Figure 5. At the top of the elbow, there are two plates. The top plate connects the tension sensor to the humerus, and the lower plate attaches the bearing assembly, also known as the joint, to the humerus. In the middle of the joint is a rod that can rotate freely due to the ball bearing around it. The housing of the angle sensor bolts onto the stationary part of the joint and the rotational part of the angle sensor fits inside the rod. A set screw locks the rotational part of the angle sensor to the rod. The ulna is attached to the joint by a block of aluminum. The ulna is bolted to the block and the block is bolted through the rod.

The ulna, like the humerus, is made from a tube. A wide slot was cut in the tube so the muscle fibers could apply torque anywhere along this slit. The muscle fibers attach to a
bolt. The bolt can be clamped down onto the ulna at any point along the slot. You can see this in Figure 5.

4.2 Muscle Fibers

The muscle fibers simulate a human biceps brachii muscle. They attach on one end to the ulna just like a biceps brachii, but due to the fact that our muscle wire only contracts 5% of its length, they do not attach at the other end to the top of the humerus. Instead, the muscle fibers continue over a Delrin pulley and down to the bottom of the humerus where the fibers attach to a connection block. You can see both connection points in Figure 5 and the plastic pulley in Figure 4.

The muscle fibers were designed to lift up to a kilogram at the end of the ulna and work with the software. In order to comply with the software, there are six fibers. Each of the fibers are made up of three strands of Flexinol. Flexinol contracts when heated past 90°C. We heat the Flexinol by running electric current through it.

We control the current through each of the wires individually. Each end of the group of three flexinol strands is crimped to a steal wire lead. The leads on the ulna side are all crimped around a ring and are grounded. The humerus side leads are held by a connection block made out of Delrin. Each of the hot wires are also held by the connection block.

4.3 Sensors

There are three sensors on our arm. One measures the angle between the humerus and ulna, another measures the tension on the muscle fibers, and the last one measures the change in length of the muscle fibers. All of these sensor’s outputs are connected to an analog to digital converter. Detailed information about these circuits can be found in Appendix D.

The angle sensor and length sensors are rotational potentiometers. The bar of the angle sensor rotates with the ulna part of the arm. The length sensor is a modified rotational potentiometer. The bar is turned by a cable. When there is no tension on the cable, the bar springs back to 0°. The length sensor is mounted up in the shoulder area of the arm and can be seen in Figure 4. The string attaches to the same place the muscle fibers do and can be seen in Figure 5. The angle sensor can also be seen in Figure 5.

The tension sensor is a strain gage. It is located at the bottom of the humerus between one of the plates and the muscle wire connection block. The output of the strain gage is in the millivolt range, so an amplifier and noise filtering circuit was added to boost the signal. The strain gage can be seen in Figure 5 and the amplifier circuit is the top right corner of Figure 6.
4.4 Electronics

The circuitry around the arm acts only as an interface with the computer. There is very little logic used. For ease, we used a Rabbit Semiconductor development kit. It is the large board at the bottom of Figure 6. The kit came with a microcontroller with flash memory, a board to power the processor and give us area to connect other chips, and the software to program the processor.

To the development board, we added on an analog to digital converter, a relay controller,
and a serial to USB converter. The analog to digital converter reads the outputs from the three sensors. The relay controller drives the six solid state relays with a higher voltage than the processor could supply. The serial to USB adapter was added to provide a RS-232 connector with proper voltages and a USB connector for improved speed and flexibility.

Two other boards were needed to control the arm. The relay board is pictured in the top left of Figure 6. The relays switch the current to the muscle wires. The third board was purchased to amplify and noise filter the signal from the tension sensor. It is pictured in the top right of Figure 6.

Almost all of the electronics are powered by the AC adapter that came with the development board. The muscle fibers are powered by a separate power supply. The power supply is pictured in Figure 7. Circuit schematics are in Appendix D.

![Figure 7: The Power Supply](image)

The software on the microcontroller is written in Dynamic C. The program is about 150 lines of code and is in Appendix E. The flash memory on the microcontroller stores the program when the power is not connected.

### 4.5 Operation Overview

To prepare the arm for operation, simply plug in the AC adapter for the microcontroller board and the power cord for the power supply.

The arm was designed around the software already in place, so the software changes are transparent to the user. The changes that occurred all deal with communicating with the arm. The details of the protocol are in Appendix F.
During use, there is a kill switch. It is a contact switch that will disconnect the muscle fibers from the power supply for as long as the switch is held in. As soon as the switch is let out, the muscle fibers will be under the control of the software again.

After operation, simply disconnect the RS-232 or USB cable and the two power connections.

5 Product Evaluation

There were only a few tests we felt were necessary. All of these test were electrical signal tests. Here are the tests we preformed.

1. Goal: Prove that each individual muscle wire package would fire.
   Test: Write ones to specific relay controller channels and monitor the arm.
   Outcome: Each wire fired correctly. The wire farthest from the body corresponds to the second bit on the relay controller, and the wire closest to the body corresponds to the seventh bit on the relay controller.

2. Goal: Prove that the muscle wires are durable enough for training.
   Test: Write a program that will constantly contract and relax the muscles and run the program for half a day.
   Outcome: The program ran successfully and the muscle wires showed the same behavior at the end of the session as at the beginning.

3. Goal: Characterize the angle sensor.
   Test: Capture the two byte output of the A2D converter at different angles and scale the values to floating point values from 0 to 2.5. This will show the voltage at that A2D channel.
   Outcome: The voltage read from the angle sensor is linearly proportional to the angle. The range of voltage read was from 0.2082 to 0.24762.

4. Goal: Verify that the tension sensor and length sensor has a linear relationship with the tension on the wire and the length of the wire respectively. Also to verify the ranges of those sensors are the full range of the A2D converter.
   Test: Pull on the muscle wires and watch the sensor readings.
   Outcome: Both sensors gave the full range of 0 to 2.5 volts.

6 Economic Analysis

We were told at the start of the project that there was only going to be one arm, so instead of focusing on manufacturability, we will simply list the expenses for this project.

Table shows our final expenses.
Our budget for the project was $4000, so we had money left over. We spent about 50% of our budget.

Labor is not included in the project’s budget for the simple reason that our labor was free and the services we used for fabrication were free. However, it may be beneficial to estimate how much more the project would have cost if everyone involved was paid. Table 2 shows our estimate of these costs.

Table 2: Salary and Fabrication Cost Estimate

<table>
<thead>
<tr>
<th>Description</th>
<th>Amount</th>
<th>Price</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering Design Time</td>
<td>300</td>
<td>50.00</td>
<td>15,000.00</td>
</tr>
<tr>
<td>Machining Cost and Operator Time</td>
<td>20</td>
<td>25.00</td>
<td>500.00</td>
</tr>
<tr>
<td>Electronic Assembly Cost and Operator Time</td>
<td>100</td>
<td>50.00</td>
<td>5,000.00</td>
</tr>
<tr>
<td>Programmer Time</td>
<td>100</td>
<td>50.00</td>
<td>5,000.00</td>
</tr>
<tr>
<td>Mentor Time</td>
<td>30</td>
<td>100.00</td>
<td>3,000.00</td>
</tr>
<tr>
<td>Faculty Time</td>
<td>35</td>
<td>150.00</td>
<td>5,250.00</td>
</tr>
<tr>
<td><strong>Total Cost</strong></td>
<td></td>
<td></td>
<td><strong>33,750.00</strong></td>
</tr>
</tbody>
</table>

7 Conclusions, Recommendations, and Future Expansion

The final version of the arm meets all the requirements specified in Section 2 except for the range of motion. We found our range of motion was about 90°. We tried to fix this problem, but were unsuccessful. The arm will not drop to 0° because there is not enough weight in the ulna part of the arm to stretch the muscle wire taught at that angle. The arm will not go up to 145° because the force required to do so is enormous at the current attachment point. The only way we can see to mitigate the problem is to make the muscle wires longer and move the attachment point on the ulna further away from the axis.

Dr. Soule knows about this problem and is still pleased with our results. We would
like to make the recommendation to add another pulley and lengthen the muscle wire if the opportunity arises.

Dr. Soule had several things in mind for future expansion. He would like to add a radius bone, and connect the biceps brachii to also pronate the forearm. Also he thought a triceps brachii muscle could be used to see if the computer could control competing muscles.

It should be reasonably easy to make both of these modifications in the future. The addition of the radius would entail adding a tube, a ball and socket joint, and some flexible connector between the radius and ulna at the wrist. It would also entail connecting some of the muscle wires to the radius instead of the ulna.

To add a triceps brachii, a slight extension would need to be made to the ulna so more muscle wires could be connected to it. Also, the pulley system would need to be modified for the additional muscle wires. Additional muscle wires would require additional relays. There are more channels on the relay controller, so no major modifications would need to be made to the development board. The software would need to change slightly and the communication protocol would change. We would guess that it would not take more than a day to change the software to handle the additional muscle fibers.
## Parts List

<table>
<thead>
<tr>
<th>Part Description</th>
<th>Part Name</th>
<th>Part Number</th>
<th>Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arm Bones</td>
<td>Aluminum Rod</td>
<td>N/A</td>
<td>15.00</td>
</tr>
<tr>
<td>Pulley</td>
<td>Delrin</td>
<td>N/A</td>
<td>From UI</td>
</tr>
<tr>
<td>Muscle Wire</td>
<td>Flexinol 250</td>
<td>Flexinol 250</td>
<td>416.23</td>
</tr>
<tr>
<td>Bearing</td>
<td>Generic</td>
<td>N/A</td>
<td>12.50</td>
</tr>
<tr>
<td>Tension Sensor</td>
<td>Thin Beam Sensor</td>
<td>TBS-40</td>
<td>114.98</td>
</tr>
<tr>
<td>Length Sensor</td>
<td>Series 150 Subminiature</td>
<td>150-0121-L1N</td>
<td>207.04</td>
</tr>
<tr>
<td>Angle Sensor</td>
<td>Rotational Potentiometer</td>
<td>N/A</td>
<td>From UI</td>
</tr>
<tr>
<td>Signal Amplifier and Conditioner</td>
<td>Transducer Techniques</td>
<td>TM0-1</td>
<td>294.7</td>
</tr>
<tr>
<td>MicroController Development Kit</td>
<td>Rabbit 3000</td>
<td>RCM3000</td>
<td>283.17</td>
</tr>
<tr>
<td>Power Supply</td>
<td>Pyramid Power supply</td>
<td>PS-36K</td>
<td>159.98</td>
</tr>
<tr>
<td>A2D</td>
<td>12 bit Analog to Digital</td>
<td>Max1231</td>
<td>5.43</td>
</tr>
<tr>
<td>Relay Controller</td>
<td>8-channel relay</td>
<td>Max4820</td>
<td>155.23</td>
</tr>
<tr>
<td>6 Relays</td>
<td>Crydon CMX60D10</td>
<td>CMX60D10</td>
<td>155.23</td>
</tr>
<tr>
<td>Serial To USB</td>
<td>Generic</td>
<td>N/A</td>
<td>29.00</td>
</tr>
</tbody>
</table>
B Mechanical Drawing Package

The drawings start on the next page.
2x Ø 0.26 THRU ALL
2x Ø 0.17 THRU ALL

Material: AL 6061-T6
Finish: 
Scale: 1:1

Comments: Part of Upper Arm Sub-assembly

Description: arm mount
Part #: 

SolidWorks Educational License
Instructional Use Only
Material: DELRIN
Finish: Insulating plate
Scale: 2:1

Comments: Part of the Upper Arm Subassembly

The information contained in this drawing is the sole property of the University of Idaho, ME Department. Any reproduction in part or as a whole without the written permission of the University of Idaho, ME Department is prohibited.

Proprietary and Confidential

Dimensions are in inches.
Tolerances: Two place decimal XX±.05
Three place decimal XXX±.005

Drawn: CWP 2/13/2006
Checked: 

UNIVERSITY OF IDAHO
ME DEPARTMENT

Description: Insulating plate

Part #: 
Description: L bracket
Material: Al 6061-T6
Finish:
Scale: 1.5:1

Comments: part of the upper arm sub-assembly

DIMENSIONS ARE IN INCHES
TOLERANCES:
TWO PLACE DECIMAL XX±.05
THREE PLACE DECIMAL XXX±.005

Material: Al 6061-T6
Finish:
Scale: 1.5:1
2x Ø 0.17 THRU ALL

2x Ø 0.09 THRU ALL

Description: pot mount

Material: Al 6061-T6

Finish:

Scale: 2:1

Comments: Part of the Upper Arm Sub-Assembly
Material: 1 in. delrin rod
Finish:
Scale: 3:1
Material: Steel
Finish: 
Part #: 
Description: housing sub-assembly
Scale: 3:1 
Comments: Part of bearing housing sub-assembly 

UNIVERSITY OF IDAHO ME DEPARTMENT 

Available for instructional use only
Material: AL 6061 T6
Finish:
Scale: 2:1

Description: ulna mount
Part #: 

Comments: part of Ulna sub-assembly

UNIVERSITY OF IDAHO ME DEPARTMENT

SolidWorks Educational License
Instructional Use Only
PRESS FIT BEARING INTO BEARING HOUSING AND ROD INTO BEARING

**ITEM NO.** | **PART NUMBER** | **QTY.**
--- | --- | ---
1 | bearing housing | 1
2 | bearing | 1
3 | rod | 1
<table>
<thead>
<tr>
<th>ITEM NO.</th>
<th>PART NUMBER</th>
<th>QTY.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ulna</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>ulna mount</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>HX-SHCS 0.164-32x0.4375x0.4375-N</td>
<td>2</td>
</tr>
<tr>
<td>ITEM NO.</td>
<td>PART NUMBER</td>
<td>QTY.</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>Bearing Assembly</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Humorous assembly</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>pulley</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>arm mount</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>pot mount</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>insulating plate</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>power plate</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>HX-SHCS 0.125-44x1.125x0.875-N</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>HX-SHCS 0.164-36x0.5x0.5-N</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>HX-SHCS 0.19-32x1.125x1.125-N</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>HX-SHCS 0.164-36x1.125x1.125-N</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>HX-SHCS 0.164-36x0.188x0.188-N</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>HX-SHCS 0.125-44x0.25x0.25-N</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>HX-SHCS 0.125-44x0.5x0.5-N</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>L bracket</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>load cell attachment</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>load cell</td>
<td>1</td>
</tr>
</tbody>
</table>
EXPLODED VIEW

DETAIL A

1

2

3

THREE PLACE DECIMAL XXX

PROPRIETARY AND CONFIDENTIAL

Material:
Finish:

Tolerances:
Two place decimal XX ± 0.05
Three place decimal XXX ± 0.005

UNIVERSITY OF IDAHO ME DEPARTMENT

Dimensions are in inches

Item # | Item
--- | ---
1 | Upper Arm Assembly
2 | Ulna Assembly
3 | HX-SHCS 6-32 x 1.25

SolidWorks Educational License
Instructional Use Only
Description: humorous
Part #:
Material: 3/4" 6061 T6 Al tubing
Finish: 
Scale: 1:2
C Design Failure Mode and Effect Analysis

The table starts on the next page.
<table>
<thead>
<tr>
<th>ITEM AND FUNCTION</th>
<th>POTENTIAL FAILURE MODE(S)</th>
<th>POTENTIAL EFFECT(S) OF FAILURE</th>
<th>SEV</th>
<th>POTENTIAL CAUSE(S) OF FAILURE</th>
<th>OCCUR</th>
<th>CURRENT DESIGN CONTROLS</th>
<th>DETECT</th>
<th>RPN</th>
<th>RECOMMENDED ACTIONS</th>
<th>ACTION TAKEN</th>
<th>SEV</th>
<th>DETECT</th>
<th>RPN</th>
<th>CLAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Muscle Wire</td>
<td>wire breaks</td>
<td>wire must be replaced</td>
<td>8</td>
<td>arm is overloaded</td>
<td>2</td>
<td>operator kill switch, tension feedback</td>
<td>3</td>
<td>48</td>
<td>have software warn against overload</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>wire wears out</td>
<td>&quot;</td>
<td>8</td>
<td>&quot;</td>
<td>2</td>
<td>&quot;</td>
<td>3</td>
<td>48</td>
<td>&quot;</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;</td>
<td>&quot;</td>
<td>&quot;</td>
<td>8</td>
<td>wires are over-powered</td>
<td>2</td>
<td>Voltage control knob, software phase modulation limits</td>
<td>5</td>
<td>80</td>
<td>design against or warn against putting too much current through wires</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sensors</td>
<td>connections fail</td>
<td>Neural net gets no feedback</td>
<td>7</td>
<td>corrosion, wires get pulled</td>
<td>1</td>
<td>wiring encased in mannequin where possible</td>
<td>2</td>
<td>14</td>
<td>heat shrink all connections, have backup sensors</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analog to digital converter</td>
<td>shorts out</td>
<td>Neural net gets no feedback</td>
<td>7</td>
<td>static discharge, over voltage input</td>
<td>3</td>
<td>microcontroller is contained inside manquin</td>
<td>2</td>
<td>42</td>
<td>encase the microcontroller board, limit voltages</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power supply fails</td>
<td>no output</td>
<td>no power is supplied to the system</td>
<td>8</td>
<td>power surge, bad parts</td>
<td>1</td>
<td>none</td>
<td>1</td>
<td>8</td>
<td>use a surge protector</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC/PC interface</td>
<td>communication failure</td>
<td>software becomes unsynched from the UC feedback</td>
<td>7</td>
<td>computer is rebooted while software is running</td>
<td>2</td>
<td>none</td>
<td>5</td>
<td>70</td>
<td>have microcontroller auto shut off if no communication</td>
<td>microcontroller reprogrammed to stop current if no communication</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>
D Electronics Schematics

The schematics and connections list start on the next page.
Electrical Component Interconnections:

**Computer:**
- RS232 (Serial) port → Rabbit 3100 RS232 Programming/Debug connector
- RS232 (Serial) port → Rabbit 3100 RS232 port
  OR
- USB port → Rabbit 3100 USB port

**M.W. (Muscle Wire) Power Supply:**
- Ground → Muscle Wire Ground Points 1-6
- + Voltage → Relays 1-6 Pin 1 (+DC Load)

**Microcontroller (Rabbit 3100):**
- RS232 Programming/Debug connector → Computer RS232 (Serial) port
- RS232 port → Computer RS232 (Serial) port
  OR
- USB port → Computer USB port
- Power Connector → uC Power Supply
- Tension sensor connector → Signal Conditioner 4-Pin Terminal Block Pins 3 and 4
- Angle Sensor Connector → Angle Sensor Signal Cable
- Length Sensor Connector → Length Sensor Signal Cable

**uC (Microcontroller) Power Supply:**
- 12 volt output → Rabbit 3100 Power connector
- 12 volt output (Via Vcc connector on the Rabbit 3100 board) → TMO-1 12 4-Pin Terminal Block Pins 1(+12 volt DC) and 2(Power Ground)
- 12 volt output (Via Vcc connector on the Rabbit 3100 board) → Fans 1-3

**Signal Conditioner (TMO-1):**
- 4-Pin Terminal Block:
  1 (+12 VDC) → (+) 12 volt output (Via Vcc connector on the Rabbit 3100 board)
  2 (Power Ground) → (-) 12 volt output (Via Vcc connector on the Rabbit 3100 board)
  3 (Analog Ground) → Tension sensor connector on the Rabbit board
  4 (Analog Output) → Tension sensor connector on the Rabbit board
- 5-Pin Terminal Block: See TMO-1 Operator Manual for Wire labeling specifics
  1 (+Excitation (Red)) → Tension Sensor cable Pin 2
  2 (-Signal (White)) → Tension Sensor cable Pin 5 (Red stripe)
  3 (+Signal (Green)) → Tension Sensor cable Pin 3
  4 (-Excitation (Black)) → Tension Sensor cable Pin 4
  5 (+Shield) → Tension Sensor cable Pin 1 (Blue stripe)

**Angle Sensor:**
- Signal Cable → Rabbit 3100 Angle Sensor Connector

**Length Sensor:**
- Signal Cable → Rabbit 3100 Length Sensor Connector
**Tension Sensor:**
- Signal Cable Pin 2 → Signal Conditioner 5-Pin Terminal Block Pin 1
- Signal Cable Pin 5 (Red stripe) → Signal Conditioner 5-Pin Terminal Block Pin 2
- Signal Cable Pin 3 → Signal Conditioner 5-Pin Terminal Block Pin 3
- Signal Cable Pin 4 → Signal Conditioner 5-Pin Terminal Block Pin 4
- Signal Cable Pin 1 (Blue Stripe) → Signal Conditioner 5-Pin Terminal Block Pin 5

**Relay Controller (MAX4820):**

**Dot Side (counting from the dot):**
1 (VCC) → Connects to the Rabbit (+3.3)
2 (SET’) → Connects to the Rabbit (+3.3)
3 (RESET’) → Connects to the Rabbit (SWITCH)
4 (CS) → Connects to the Rabbit (PB7)
5 (DIN) → Connects to the Rabbit (PD4)
6 (SCLK) → Connects to the Rabbit (PB0)
7 (DOUT) → Connects to the length sensor
8 (N/C) → DO NOT CONNECT
9 (GND) → GND
10 (OUT8) → Not connected

**Non-Dot Side (counting down from the dot):**
20 (OUT1) → Connects to the Relay controller
19 (OUT2) → Connects to the Relay controller
18 (PGND) →
17 (OUT3) → Connects to the Relay controller
16 (OUT4) → Connects to the Relay controller
15 (COM) →
14 (OUT5) → Connects to the Relay controller
13 (OUT6) → Connects to the Relay controller
12 (PGND) →
11 (OUT7) →

**A to D (MAX1031):**

**Dot Side (counting from the dot):**
1 (AIN0) → Connects to the tension sensor
2 (AIN1) → Connects to the length sensor
3 (AIN2) → Connects to the angle sensor
4-12 (AIN3-AIN11) → Not connected

**Non-Dot Side (counting down from the dot):**
24 (EOC’) → Connects to the Rabbit (PG0)
23 (DOUT) → Connects to the Rabbit (PD5)
22 (DIN) → Connects to the Rabbit (PD4)
21 (CS’) → Connects to the Rabbit (PB6)
20 (SCLK) → Connects to the Rabbit (PB0)
19 (VDD) → Connects to the Rabbit (+3.3)
18 (GND) → Connects to the Rabbit (GND)
17 (REF+) → Not connected
16 (CNVST’) → Connects to the Rabbit (PF5)
15 (AIN14) → Not connected
14 (AIN13) → Not connected
13 (AIN12) → Not connected

Fans 1-3:
- Fan Power Connectors → uC Power Supply 12 volt output (Via Vcc connector on the Rabbit 3100 board)

Relays (CMX60D10) 1-6:
- Pin 1 (+DC Load) → M.W. Power Supply +Voltage

M.W. (Muscle Wire – Flexinol 250) 1-6:
- Muscle Wire Ground Points → M.W. Power Supply Ground
- Muscle Wire + Voltage points 1-6 → M.W. Power Supply + Voltage
Microcontroller Code

// Team Random MicroController code
// Written by Nick Podgursky and Cameron Merridith
// University of Idaho
// Spring 2006
// Developed for Terry Soule’s Spino Neuro Muscular System
// Research Project
// Interfaces with...
// Rabbit Semiconductor RCM3100
// Max1031 ADC
// Max4820 Relay Driver

class auto
#define SPI_SER_B
#define SERB_USEPORTD
#define SPI_CLK_DIVISOR 100
#define SPI_CLOCK_MODE 1
#define CINBUFSIZE 15
#define COUTBUFSIZE 15
#define CS_ADC 6 // Chip Select for ADC
#define CNVST 5 // CNVST for ADC
#define CS_RELAY 7 // Chip Select for Relay Controller
#define BAUD_RATE 19200 // Baud rate for serial communication

// This is necessary for initializing RS232 functionality of LP35XX boards.
#if _BOARD_TYPE_ == 0x1200 || _BOARD_TYPE_ == 0x1201
brdInit();
#endif
use "spi.lib"

// Function Prototypes
void MAX1031_Init();
void MAX4820(unsigned char relayState);
void Start_Conversion();
void Read_ADC(double* tension, double* length, double* angle);

// Global Variables
char adc_reading[6]; // Array to store ADC readings

void main()
{
    unsigned char relayState;
    double tension;
    double length;
    double angle;
    char t1, t2, a1, a2, l1, l2;
    char top_reached;
}
```c
int flag;
relayState = 126;
top_reached = 0;
flag = 0;

serCopen(BAUD_RATE);
SPIinit();
MAX1031_Init();

// Main program loop
while(1)
{
   // Read the relayState from the computer, and pass to MAX4820
   costate
   {
      wfd{ relayState = cof_serCgetc();} // yields until successfully
      getting the state of the muscles
   }

   // To remove PWM, uncomment function call below
   // and comment out the PWM costate below
   //MAX4820(relayState);

   IntervalMs(10);

   wfd cof_serCputc(adc_reading[0]);
   wfd cof_serCputc(adc_reading[1]);
   wfd cof_serCputc(adc_reading[2]);
   wfd cof_serCputc(adc_reading[3]);
   wfd cof_serCputc(adc_reading[4]);
   wfd cof_serCputc(adc_reading[5]);

   /*
    * Code for closed loop control of the arm
    * if(flag == 0 && angle > 1.60)
    * {
    *     printf("Upper Angle Reached");
    *     flag = 1;
    *     relayState = 0;
    * }
    * else if(flag == 1 && angle < 1.2)
    * {
    *     printf("Lower Angle Reached");
    *     flag = 0;
    *     relayState = 126;
    * }
    */

   // Writing the relay controller (PWM)
   // To remove PWM, comment out this costate and uncomment the
   // costate
   {
      MAX4820(relayState);
   }
```
```c
    waitfor( IntervalMs(4));
    MAX4820(0x00);
    waitfor(IntervalMs(1));
}

  // Reading from the A2D
  costate
  {
    Start_Conversion();
    waitfor(BitRdPortI(PGDR,0)==0); // wait for the conversion to complete
    Read_ADC(&tension,&length,&angle);
    printf("%.4f %.4f %.4f\n", tension, length, angle); // display the readings
  }

void MAX4820(unsigned char relayState)
{
    BitWrPortI(PBDR,&PBDRShadow,0,CS_RELAY);// relay chip select low
    SPIWrite(&relayState,1);
    BitWrPortI(PBDR,&PBDRShadow,1,CS_RELAY);// relay chip select high
}

void MAX1031_Init()
{
    const static unsigned char ConversionRegister = 0x90; // 1001 0000
    const static unsigned char SetupRegister = 0x48; // 0100 1000
    const static unsigned int UDRegister = 0x00; // 0000 0000
    const static unsigned int BDRegister = 0x00; // 0000 0000
    const static unsigned char AveragingRegister = 0x3c; // 0010 1100
    const static unsigned char ResetRegister = 0x18; // 0001 1000

    // Configures port F to all output and Port G to all input.
    WrPortI(PGDDR,&PGDDRShadow,0x00); // 0000 0000
    WrPortI(PFDDR,&PFDDRShadow,0xFF); // 1111 1111

    BitWrPortI(PBDR,&PBDRShadow,1,CS_ADC); // AtoD chip select high
    BitWrPortI(PBDR,&PBDRShadow,0,CS_ADC); // AtoD chip select low
    SPIWrite(&ResetRegister,1); // Clear the FIFO
    BitWrPortI(PBDR,&PBDRShadow,1,CS_ADC); // AtoD chip select high
    BitWrPortI(PBDR,&PBDRShadow,0,CS_ADC); // AtoD chip select low
    SPIWrite(&SetupRegister,1); // Write the setup register
    BitWrPortI(PBDR,&PBDRShadow,1,CS_ADC); // AtoD chip select high
    BitWrPortI(PBDR,&PBDRShadow,0,CS_ADC); // AtoD chip select low
    SPIWrite(&ConversionRegister,1);
    BitWrPortI(PBDR,&PBDRShadow,1,CS_ADC); // AtoD chip select high
```
```c
void Start_Conversion()
{
    // Initiate a conversion
    BitWrPortI(PFDR, &PFDRShadow, 0, CNVST); // set CNVST low
    BitWrPortI(PFDR, &PFDRShadow, 1, CNVST); // set CNVST high
}

void Read_ADC(double *tension, double *length, double *angle)
{
    BitWrPortI(PBDR, &PGDRShadow, 0, CS_ADC); // AtoD chip select low
    SPIRead(&adc_reading, 6); // Read 6 bytes from the AtoD
    BitWrPortI(PBDR, &PGDRShadow, 1, CS_ADC); // AtoD chip select high

    // Combines the raw bytes
    *angle = (adc_reading[0] <<8) | (adc_reading[1]);
    *length = (adc_reading[2] <<8) | (adc_reading[3]);
    *tension = (adc_reading[4] <<8) | (adc_reading[5]);

    // Converts to voltage
    *tension = (double)*tension/1638;
    *length = (double)*length/1638;
    *angle = (double)*angle/1638;
}
F Protocol to Communicate with the Microcontroller

The input to the microcontroller from the computer is a single byte. Certain bit positions are assigned to different muscle fibers. The least significant bit and the most significant bit are don’t care bits. The bit next to the least significant bit controls the wire farthest from the torso and the bit next to the most significant bit controls the wire closest to the torso. The intermediate bits map to the intermediate wires in a linear fashion. A one in the bit position signifies current should be going through the muscle fiber, and a zero in the bit position signifies no current should be going through the fiber.

The output of the microcontroller is the sensor readings. These three values are read straight from the A2D converter and passed to the serial port. No conversion is done. Each of the values from the A2D converter are two bytes long. The two least significant bits and the four most significant bits are don’t care bits. This gives 10 bits to linearly represent voltages from 0 to 2.5 volts. The three sensor values are sent in order of tension, length, and then angle. The two byte values are sent in big endian order.

The microcontroller communicates with the computer a maximum of once every 10 milliseconds. It expects the computer to send the states of the muscle fibers first. Then it will wait up to 10 milliseconds, sample the A2D converter, and send the sensor data to the computer. After sending the sensor data, the microcontroller will loop and wait for the computer to send the states of the muscles again. The wait time will adapt to make sure the read/write cycle time is very close to 10 milliseconds. The computer program does not need to worry about timing issues if it is using blocking read and write commands.