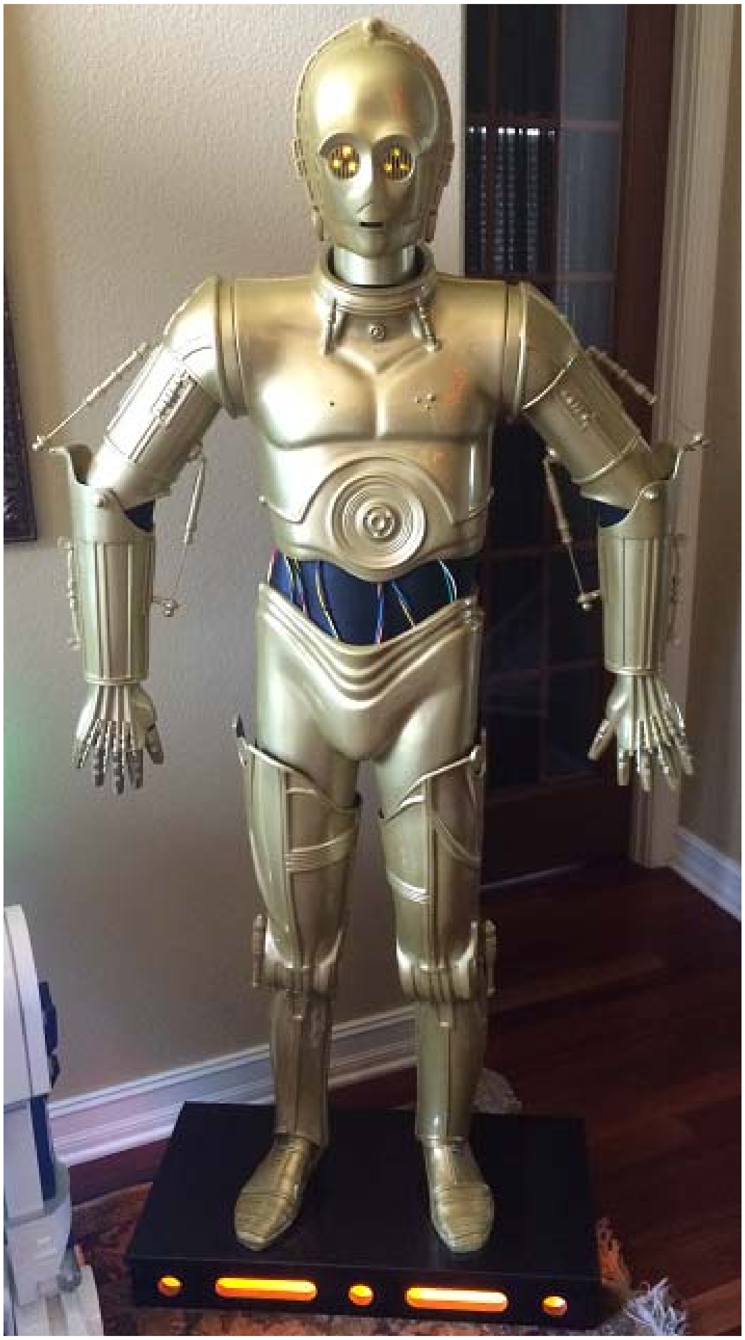
C3PO Droid Construction

September 2014



C3PO Droid Construction

1 DISCLAIMER 3

2 CONTROL 4

2.1 Wireless Remote 4
2.2 Arduino Mega Controller 5
2.3 Arduino Uno R3 Controller 6
2.4 Arduino WAV Shield 6
2.5 Pololu Maestro Servo Controller 8
2.6 Pololu RC Switch with Relay 9

3 ELECTRICAL ASSEMBLY 10

3.1 Component List 10
3.2 Electrical Component Mounting 11
3.3 Electrical Wiring 14
3.4 Elbow Actuators 15
3.5 Head Rotation Servo 16

4 ARMATURE 17

5 COSTUME COMPONENTS 21

5.1 Resin Head 22
5.2 Eyes 24
5.3 Ears 24
5.4 Resin Feet 26
5.5 Shorts 27
5.6 Arms 28
5.6.1 Decorative Item Attachment 28
5.6.2 Cylinders 29
5.6.3 Round Trim Ring 30
5.6.4 Upper Arm Attachment 31
5.7 Hands 32
5.8 Torso 35

6 RIBBING 36

7 PAINT 37

8 DROID BASE 40

8.1 Base Construction Detail 40
8.2 Base Backlighting 43

9 PROGRAM LISTING 45

9.1 Arduino Mega 2560 45
9.2 Arduino Uno R3 54
9.3 Pololu Maestro 60

1 DISCLAIMER

This document and the attached drawings are not to be used for construction.

You are hereby authorized to view, copy, print, and distribute the materials from this document or the attached drawings subject to the following conditions:

- The materials are for your personal informational use only and not for commercial purposes.
- No liability in respect of any of the contents of such publications or use thereof is accepted and no warranties expressed or implied are made in relation thereto.

Disclaimer

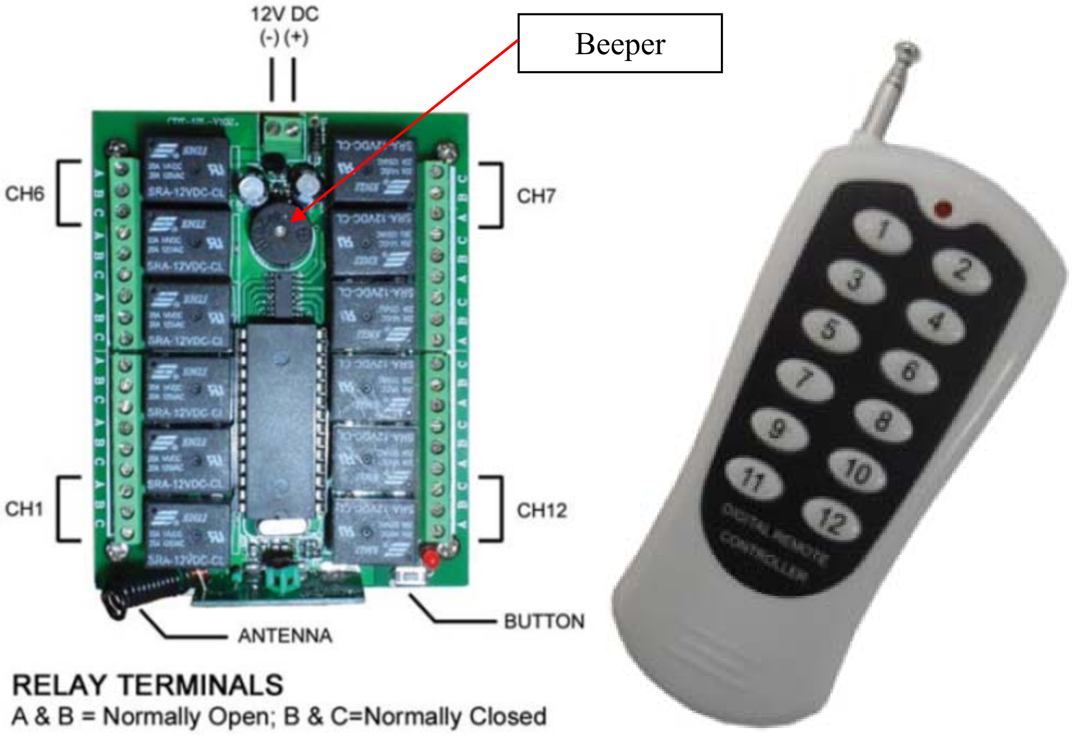
ANY INFORMATION CONTAINED IN THIS DOCUMENT OR THE ATTACHED DRAWINGS ARE PROVIDED [AS IS] FOR YOUR INFORMATIONAL PURPOSES ONLY, WITHOUT WARRANTY OF ANY KIND, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. THE AUTHOR OF THIS DOCUMENT AND ATTACHED DRAWINGS WILL IN NO EVENT BE LIABLE FOR ANY DIRECT, INDIRECT OR PUNITIVE DAMAGE OF ANY KIND THAT RESULTS FROM THE USE OF OR INABILITY TO USE THIS DOCUMENT OR THE ATTACHED DRAWINGS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, GOODWILL, OR BUSINESS INTERRUPTION. THE AUTHOR OFFERS NO ASSURANCES, WARRANTIES OR GUARANTEES AS TO THE VALIDITY OR COMPLETENESS OF THIS INFORMATION. INFORMATION WILL BE CHANGED, UPDATED AND DELETED WITHOUT NOTICE.

2 Control

The droid is controlled by two Arduino microcontrollers and a Pololu Maestro USB servo controller.

2.1 Wireless Remote

It all starts with a 12 channel wireless remote that controls 12 separate audio and motion sequences. The 12 function remote has twelve relays (7 amps max each) on screw down terminal blocks. The normally open contact for each of the twelve relays is connected to an Arduino Mega 2560 controller.



Note: The remote can be configured as toggle, latched, or momentary operating. Getting it to operate in momentary mode took some messing around at first but it has been bullet proof ever since I got the momentary operation working.

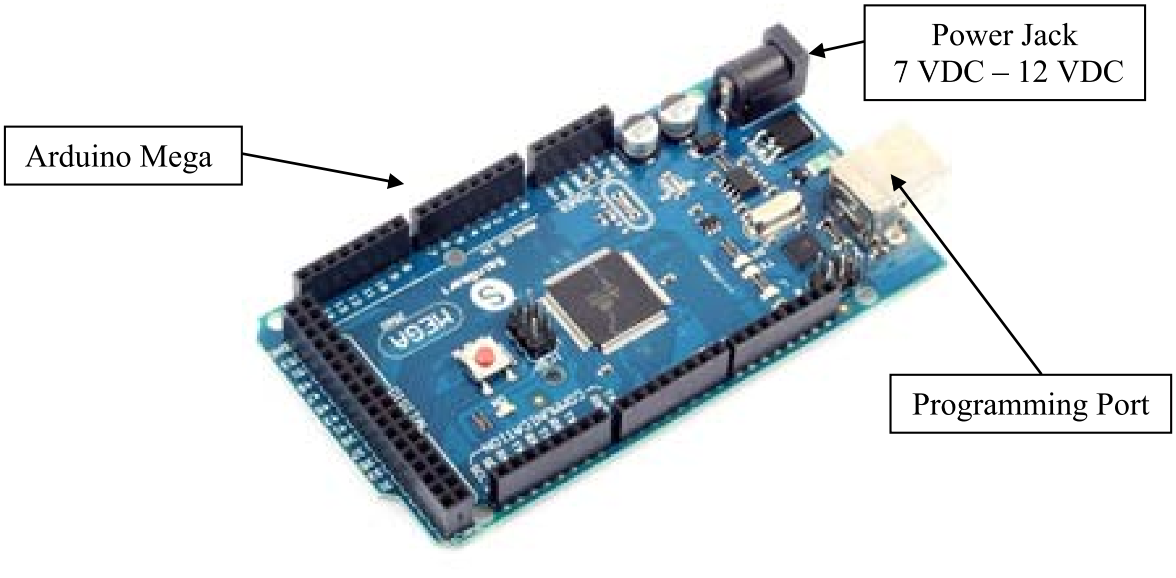
One thing I dislike about this remote is that a beeper on the board makes a chirp sound when a trigger is initiated. Placing tape over the beeper still did not quiet things down. I ended up getting rid of the whole piezo beeper by jamming a small screwdriver in it and making sure it never beeps again. Problem solved.

The specs for the remote can be found on the next page.

12 Channel Remote Specs
Operating Voltage: 12V
Operating Current: ≤10mA
Frequency: 310MHz, 315MHz, 418MHz, 433MHz
Transmitting Power: 10mW
Transmitting Distance: 100M
Encoding Type: PT2262, EV1527, HCS300 rolling code
Control Switch System:
Operating Voltage: DC12V
Quiescent Current: ≤6mA
Operating Frequency: 315MHz 433MHz
Receiver Sensitivity: -105dBm
Output Current: ≤10A
Size: 97 x 77 x 26mm / 3.81 x 3.03 x 1.02in(L x W x H)

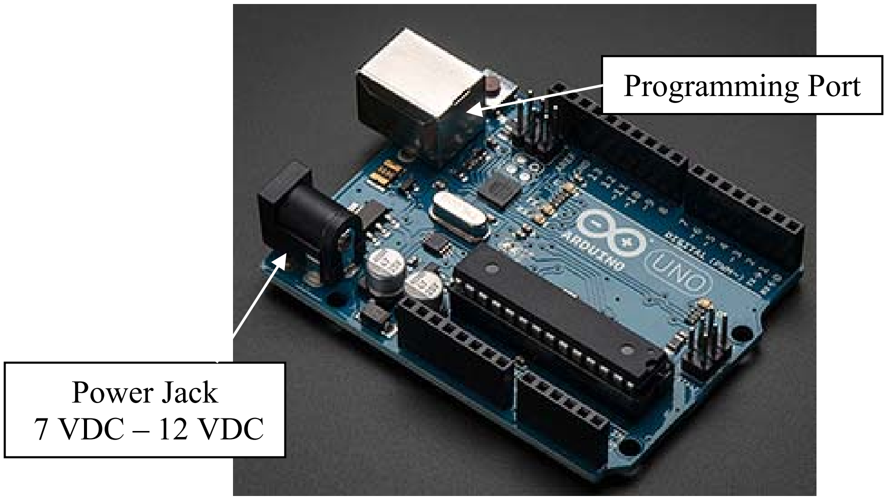
2.2 Arduino Mega Controller

The system utilizes an Arduino Mega 2560 and an Arduino Uno R3. The Arduino Mega is programmed to monitor the status of each remote control relay. Relays 1 to 12 equate to 12 separate show triggers. A show has specific audio and motion movements that are based upon the input that is received. The Arduino Mega also transmits a show playback request via a serial line to run a specific show subroutine in a Pololu Maestro servo controller.



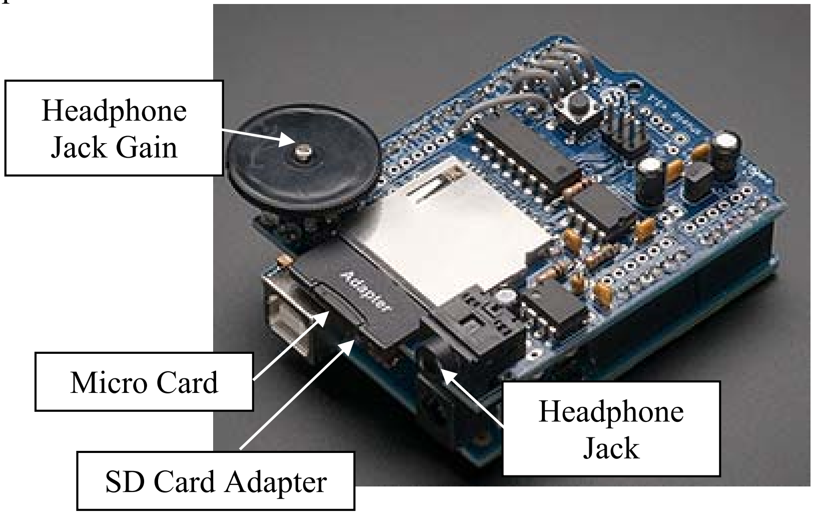
2.3 Arduino Uno R3 Controller

The Arduino Mega sends a binary code to an Arduino UNO R3 that represents the show number requested utilizing just 4 Arduino Mega 2560 outputs and 4 Arduino Uno R3 inputs. The Arduino Uno R3 shown below uses an ADAFruit Wav shield that plays back the show audio based upon the binary code received from the Arduino Mega 2560. The 4 bits are assigned values of 1, 2, 4, and 8. This technically gives you 15 combinations. I am not including all zeroes as a valid combination.



2.4 Arduino WAV Shield

The system utilizes an audio playback board for producing the audio. The audio board utilized for the C3PO droid is an Adafruit Wave Shield. The audio board sits on top of the Arduino Uno R3 controller and interfaces with the controller via the .1" male stacking pins. The Wave Shield has an SD card as shown below for storing the audio files. The SD card slot actually contains a micro SD card inserted into an SD card adapter. Full SD cards are getting difficult to find and micro SD cards with the adapter work well.



There are 12 audio files stored on the SD card. The SD card is formatted as Fat16 but Fat32 will also work. The audio payer will play any audio .wav file that is 22 kHz, 16 bit, mono, and of any length. The audio is routed through a standard headphone jack to a Pyle amplifier.

A small rotary potentiometer is located on the audio playback board to control the level of the audio signal sent to the headphone jack. In the picture above, this is defined as headphone gain. The black knob was removed and the gain adjusted high by utilizing a small #1 Phillips screwdriver.

Audio Tracks

I hired a voice actor to provide me with the following dialogue utilizing C3PO's voice.

Audio 1
"Happy Birthday Emily"

Audio 2
"Happy Birthday Kim"

Audio 3
"Happy birthday Megan"

Audio 4
"Hello, I am C3PO human cyborg relation. Welcome to the Guarnero household, it must be a great pleasure for you to make my acquaintance."

Audio 5
"Happy Halloween that is a most excellent costume you are wearing. Say trick or treat and select a carbon based snack of your choice."

Audio 6
"Happy Anniversary Kim"

Audio 7
"Yo mama so stupid she spent the whole day saying "am not" to Artoo! Yo mama so fat she has her own gravitational orbit! Yo mama so stupid she went to Bangkok to get a TIE fighter! I detect a tremor in the gravitational field. The last time I felt it was in the presence of yo mama."

Audio 8
"Kim must be what humans call a saint to allow Master John to have all of us standing in the foyer. Of course, since she is married to Master John, she must be the luckiest princess in the world. R2, my sensors indicate that Kim appears to be a very slim and attractive life form, what do you think?"

Audio 9
"Happy Thanksgiving"

Audio 10
"My sensors indicate that Kim's buns are superior to princess Leia's in every way"

Audio 11
"Master John is a most intelligent and handsome carbon based specimen. Please ignore Darth Vader and that ancient lost in space robot, I am the droid that will most interest you. After all, anyone can say danger, danger, will Robinson."

Audio 12
"Merry Christmas or are you one of those that prefer happy holidays?"

2.5 Pololu Maestro Servo Controller

The system utilizes a Pololu Maestro 18 channel servo controller. The Maestro servo controller allows control of the servos via a scripting language located on the controller. The Maestro also allows you to run programmed subroutines based on serial commands sent from an external controller. The subroutine program is simple and allows you to have statements that perform delays, move the servo to a specific position, and set the overall velocity and acceleration for each servo.

An Arduino Mega 2560 transmits a show playback request via a serial line to run a specific show subroutine in the Pololu Maestro servo controller. The subroutine requested to run in the Maestro is based on the remote control relay input that is activated on the Arduino Mega 2560 controller. I kept the Arduino audio trigger and Maestro subroutine number the same as the wireless key pressed so as to maintain consistency throughout the system. Show 1 is remote button press 1 which is wav file 1 and Maestro subroutine 1.

The Pololu Maestro I am using is an 18 channel device. I only use 5 of the 18 channels for the C3PO droid. Specifically, I am using the following channels.

Channel 0 – Head turn servo
Channel 2 – Head servo power on/off RC Switch
Chanel 6 – Left elbow linear actuator
Chanel 7 – Right elbow linear actuator
Chanel 11 – Eye LED dimmer

The Eye LED dimmer was created by clipping the motor leads of an old servo. The wires that went to the servo motor were routed to the eye LEDS through a 220 ohm resistor. By doing this I could control the eyes just as if they were any other axis and dim them as I wished.

2.6 Pololu RC Switch with Relay

The system utilizes a Pololu RC switch with relay to disconnect power from the head turn servo. Due to some mechanical imbalance that I cannot correct, the servo would hum when stopped in some positions. This switch was connected to one channel of the maestro and the head turn servo power wire was placed in series with the normally open contact of the relay. This way I can disable the servo after a move is complete and save wear and tear on the servo while eliminating noise from servo hum.

3 Electrical Assembly

The electrical components are mounted on two white steel sub-panels inside the torso. The materials used for the electrical assembly are listed in the following section.

3.1 Component List

The following list details all of the electrical components used in the system.

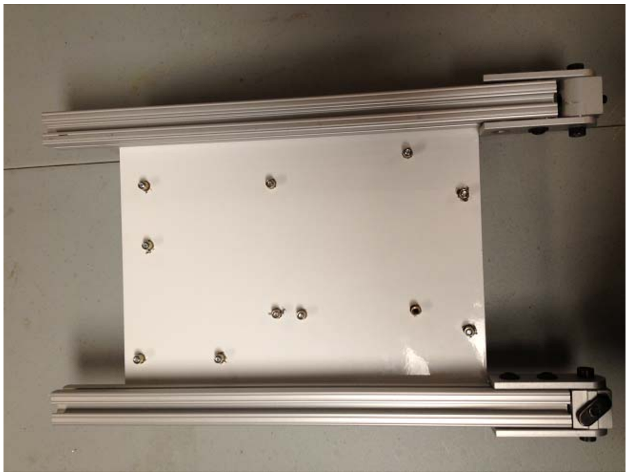
Description	Part Number	QTY	Purchased From
Sub Panel White Steel (12” x 10”)	HW-MP1210CS	2	Automationdirect.com
Din Rail (35mm x 10mm x 1 meter)	DN-R35SAL1-2	1	Automationdirect.com
Wire Duct (1” x 1.5”x 6.5’) Gray	T1E-1015G-1	1	Automationdirect.com
Power Supply 5VDC, 3A	PSE05-115	1	Automationdirect.com
Power Supply 12VDC, 1.25A	PSE12-115	1	Automationdirect.com
Power Supply 12VDC, 4.5A	PSE-BRKT-1	2	Automationdirect.com
Din Rail Mounting Kit For Power Supplies	PSC-12-060	2	Automationdirect.com
STTB2.5 Terminal Blocks	3031270	18	mouser.com
STTB2.5 End Covers	3030459	3	mouser.com
ST-4 Ground Block	3031380	1	mouser.com
End Block	0800886	4	mouser.com
STTB2.5 Plug In Bridge FBS 4-5	3030187	10	mouser.com
Fuse Block 6-DREHSILED 24	3025587	1	mouser.com
Fuse Block 6-DREHSILED 250	3025574	1	mouser.com
Wireless Trigger (12 Relay)	0990-12B	1	frightprops.com
Pyle Home PHA15 150 Watt Mono Audio Amplifier	PHA15	1	amazon.com
Arduino Uno R3	Prod ID: 50	1	Adafruit.com
Arduino Mega 2550 R3	Prod ID: 191	1	Adafruit.com
Wave Shield for Arduino Kit	Prod ID: 94	1	Adafruit.com
Pololu RC Switch With Relay	Item: 2804	1	Pololu.com
Mini Maestro 18 Channel USB Servo Controller	Item: 1354	1	Pololu.com
Hitec 37955S HS-7955TG Servo (Head)	HS-7955TG	1	amazon.com
Elbow Actuators & Board L16-P Linear Actuator 50 mm ,63:1,12 Volts	L16-50-63-12-P-LAC	2	firgelli.com
Extension Cable -S	Cable-S	2	firgelli.com
LAC Board Case	Case	2	firgelli.com

3.2 Electrical Component Mounting

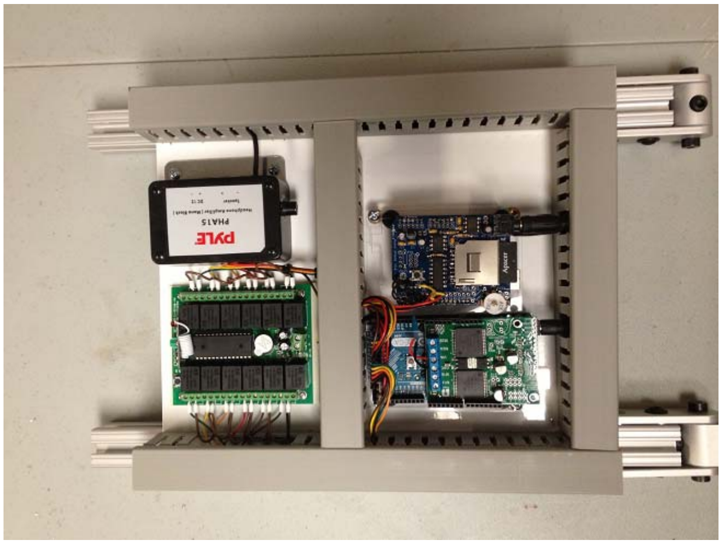
The electrical components are mounted to the sub-panels utilizing Din rail. Din rail and wire duct is mounted to the steel sub-panels using rivet nuts and screws. The rivet nut is a simple way to install a threaded hole in steel. Drill a hole, spin the rivet nut onto the threaded shaft of the tool, insert into the hole, squeeze the trigger, spin the knob on the end of the tool counter clockwise to remove the tool from the insert and you have a threaded insert. Much easier than tapping holes. I used a Marson 39302 poly nut thread setter kit I got on Amazon. The kit is shown below.

The kit comes with some rivet nuts but you an order additional ones on Amazon as well. I used the 8-32 and 10-32 rivet nuts exclusively. With the tools for fabrication in place, it is now time to fabricate. The attached drawings illustrate the layout of the control panels so you will want to refer to the drawings for details.

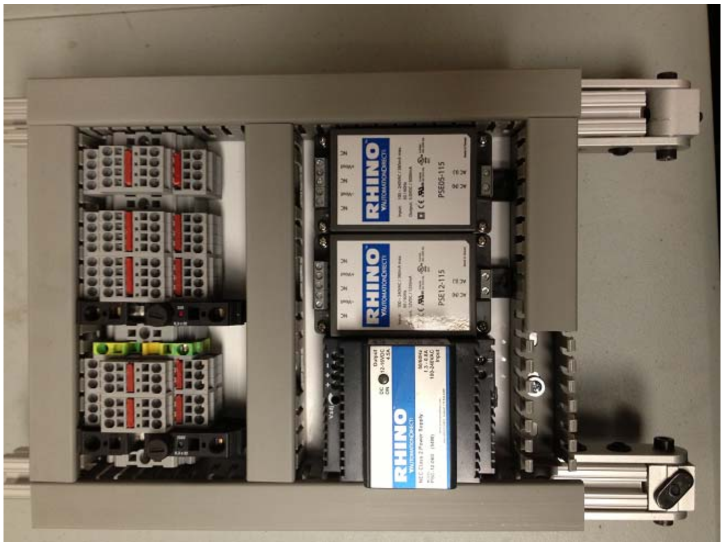
I mounted the sub-panels to 80/20 aluminum extrusion. The 80/20 is like erector set material for adults. The picture below shows one subpanel mounted to the backside of the extrusion. This picture also shows the rivet nuts sticking thought the back of the panel.



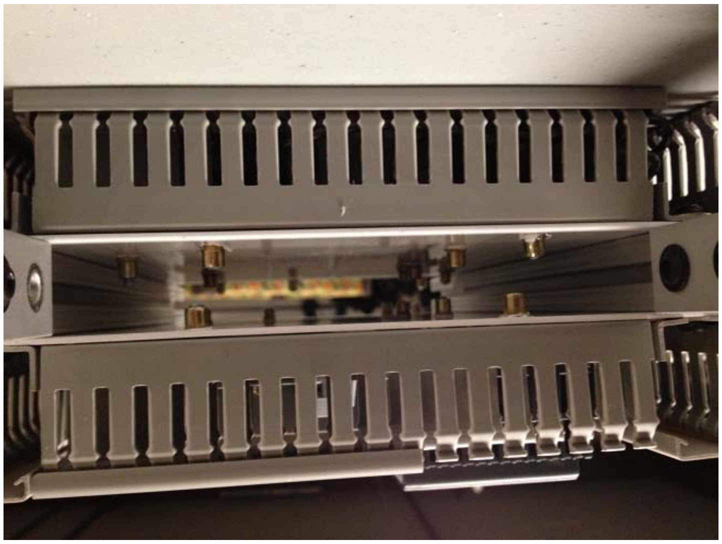
The other side of this panel looks like the following. This is the front of the torso. All of the components are not mounted on the front panel yet but it provides a good idea of where this is going.



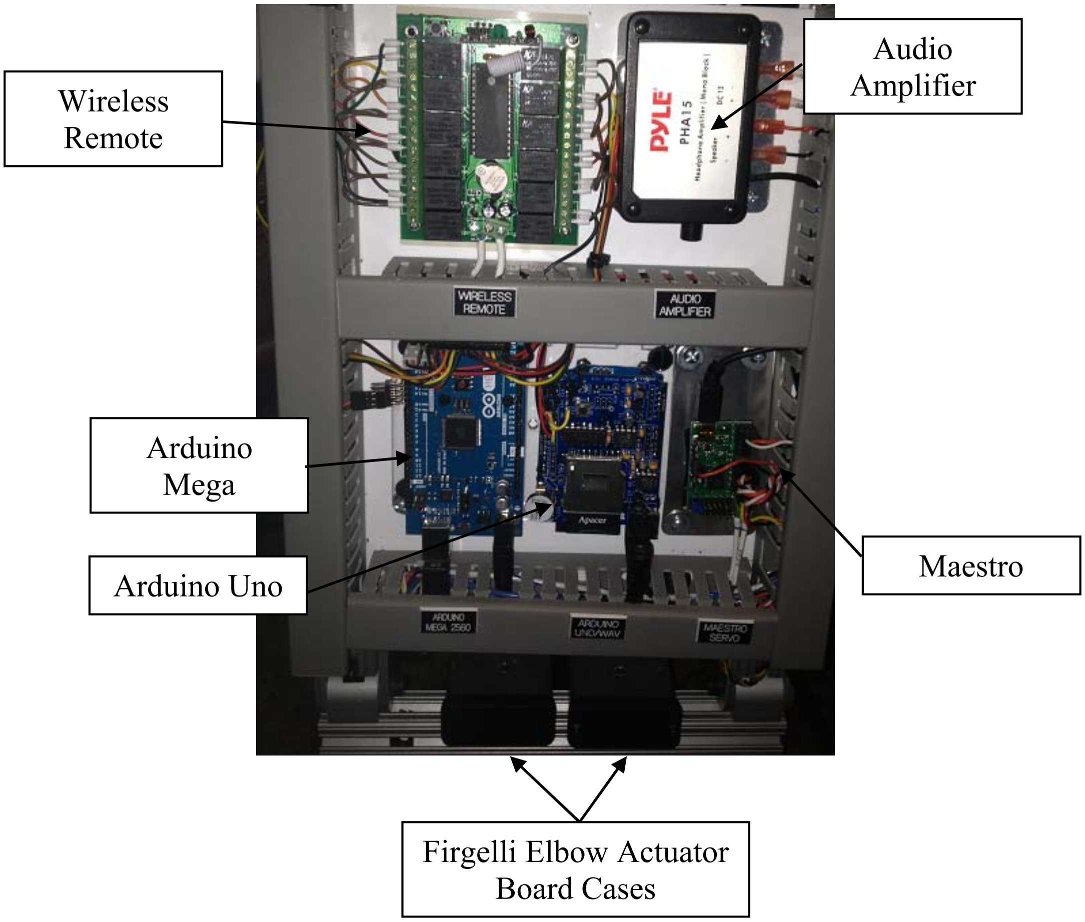
The following picture shows the second sub-panel that is mounted to the other side. This sub-panel contains the power distribution terminal blocks and power supplies. It is not shown in the pictures but a hole was drilled in the wire duct and back panels so the two panels have a hidden path for wires to pass between the front and the back of the torso.



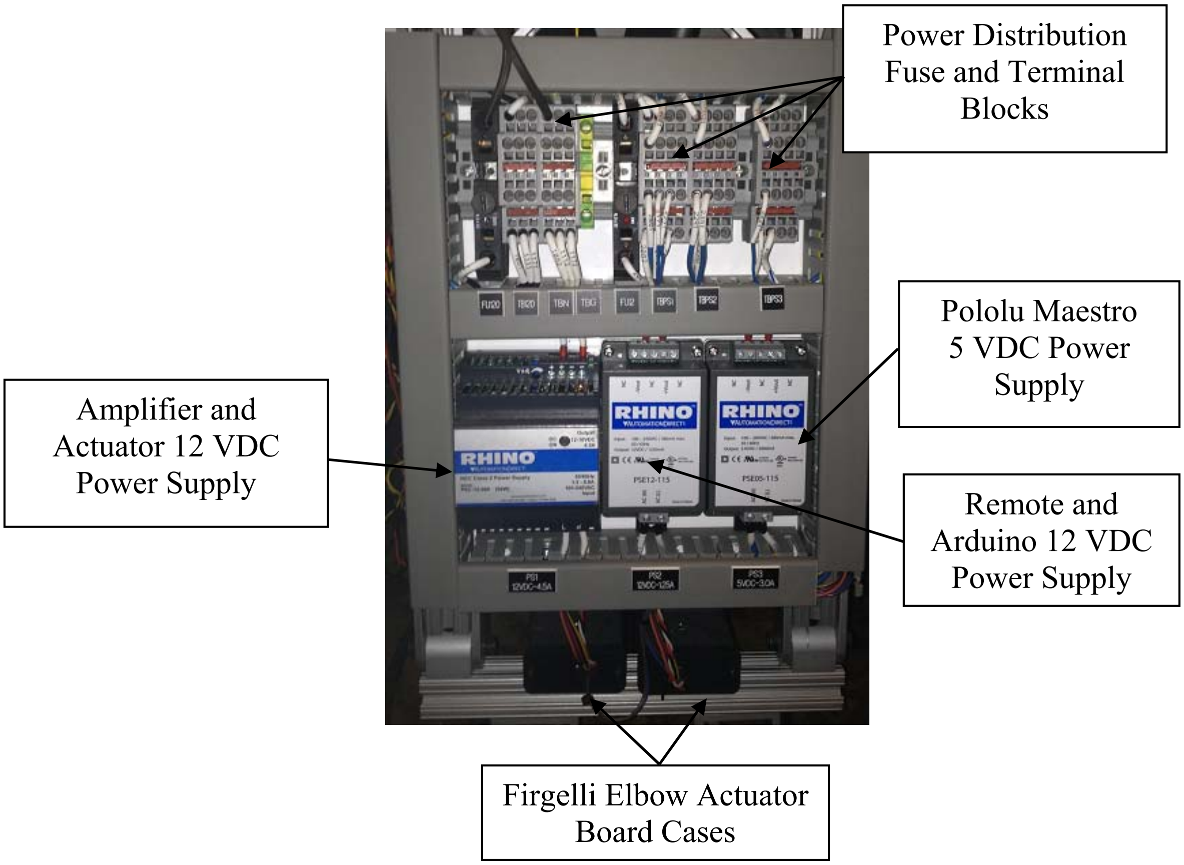
A side view of the sub-panel sandwich can be seen below.



Here is a view of the front panel with all components mounted and wired.

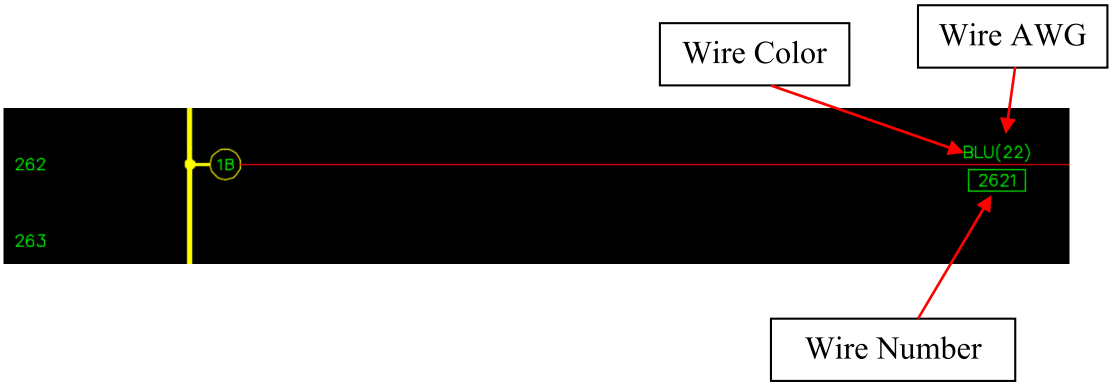


Here is a view of the rear panel with all components mounted and wired.


Power Distribution Fuse and Terminal Blocks
Pololu Maestro 5 VDC Power Supply
Amplifier and Actuator 12 VDC Power Supply
Remote and Arduino 12 VDC Power Supply
Firgelli Elbow Actuator Board Cases
RHINO
RHINO
RHINO

3.3 Electrical Wiring

As illustrated on the attached electrical drawings, a color code, wire awg, and wire number is associated with each wire. The drawing package will use the following terminology for a wire.



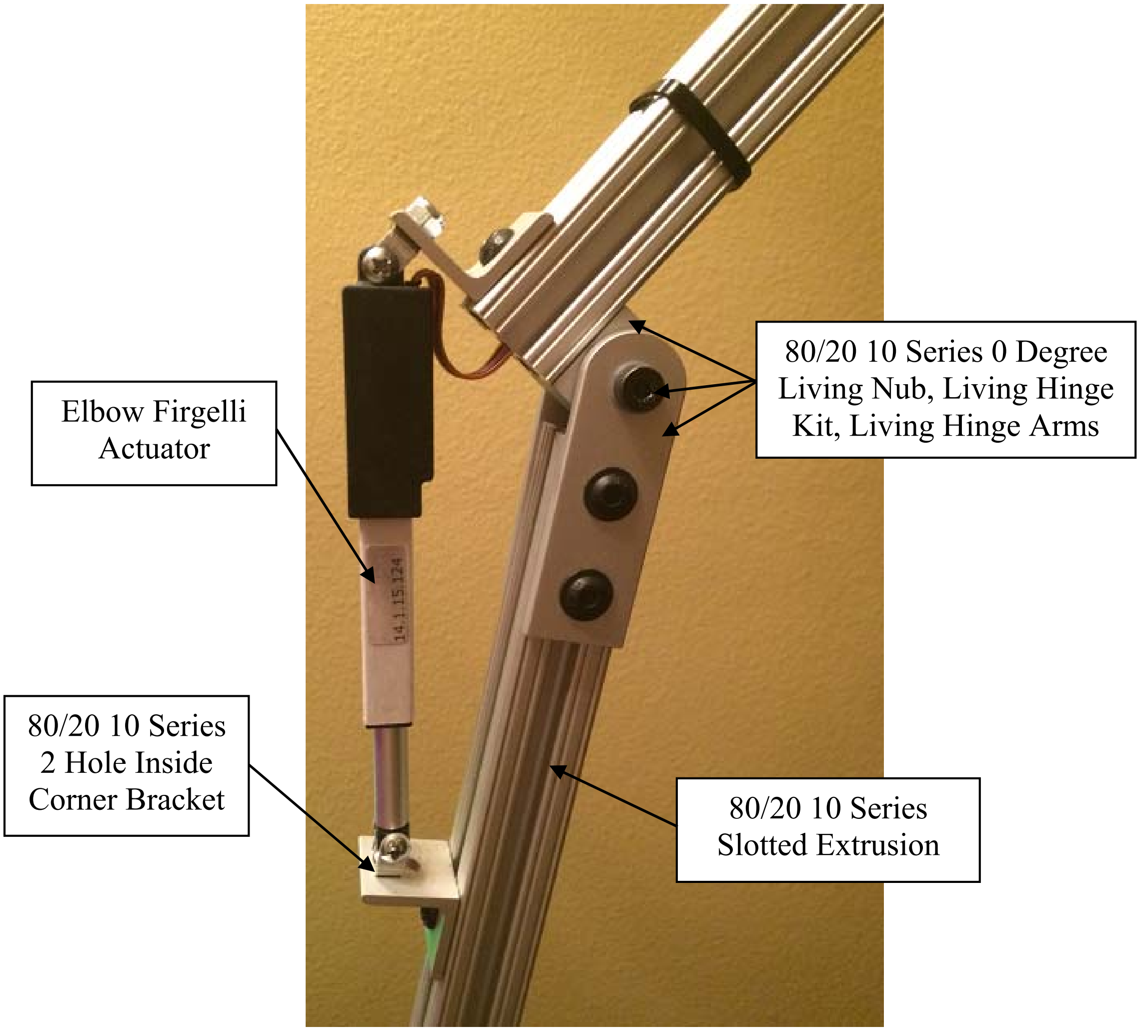
Each wire receives a ferrule. The wire ferrule is just a connector that is crimped on the wire which is used to terminate stranded wires. The wire used for the system is all MTW (machine tool wire) stranded and utilizes UL508A color codes where possible.

The ferrules, wire, and ferrule crimper can be purchased from automation direct or your favorite supplier.

The attached electrical drawings illustrate how the C3PO was wired. The drawings will show all point to point connections.

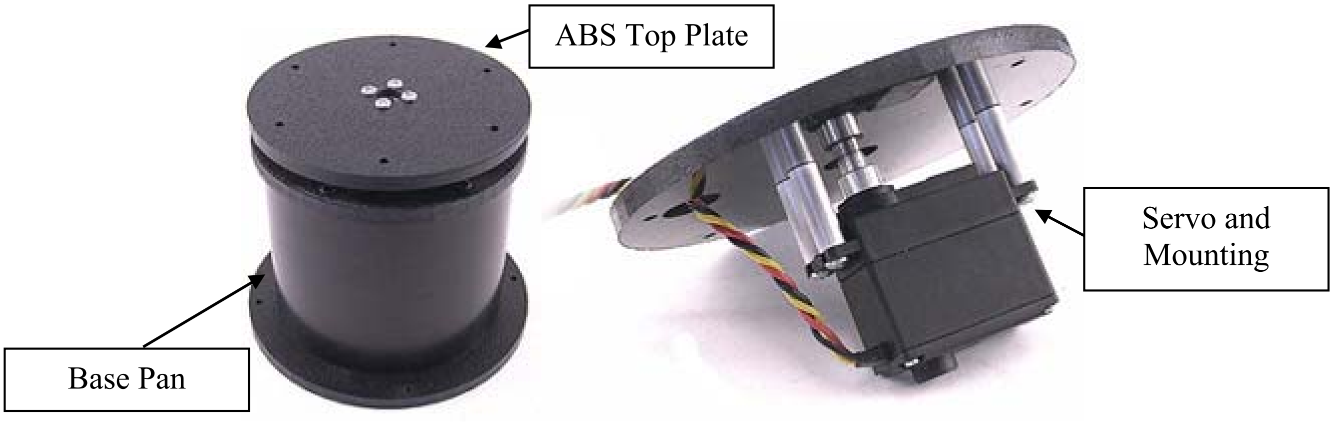
3.4 Elbow Actuators

Firgelli actuators are used to create elbow motion. The picture below shows an elbow actuator mounted on the 8020 frame.

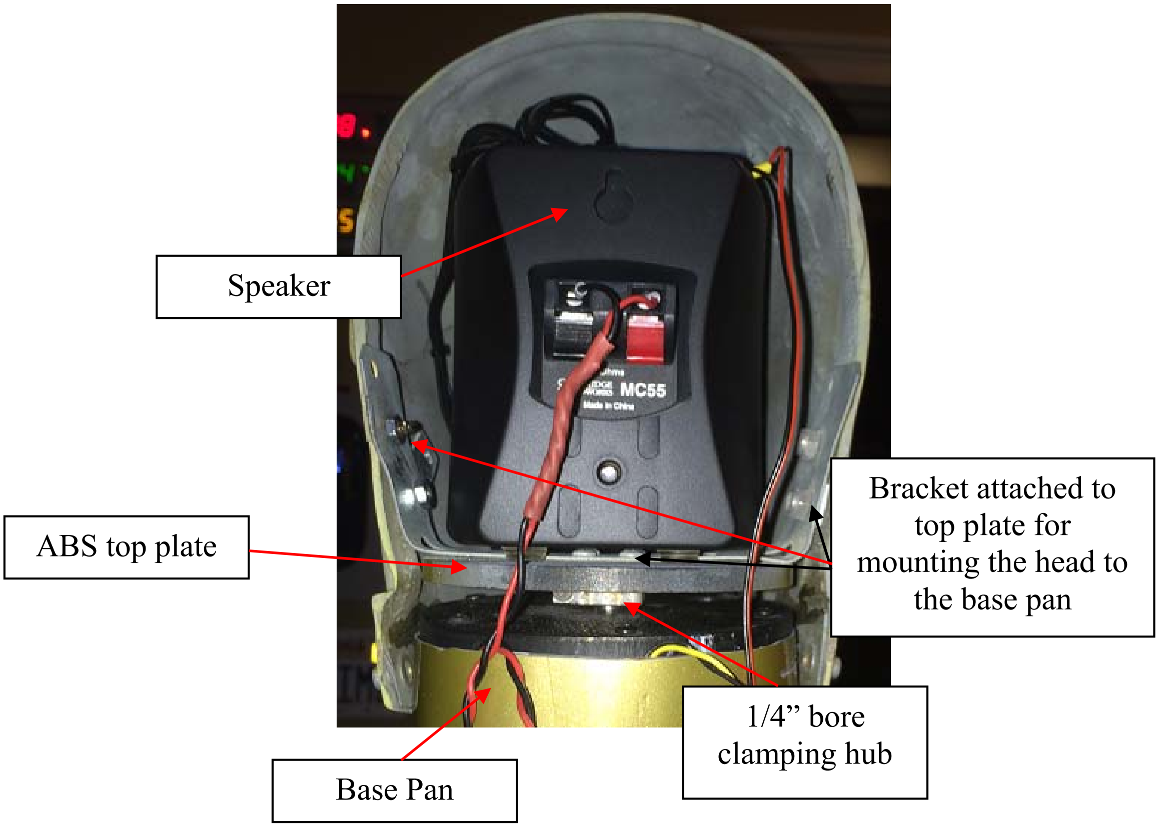

80/20 10 Series 0 Degree Living Nub, Living Hinge Kit, Living Hinge Arms
Elbow Firgelli Actuator
80/20 10 Series 2 Hole Inside Corner Bracket
80/20 10 Series Slotted Extrusion

3.5 Head Rotation Servo

The head is mounted to a Servo City DDP155 base pan. The base pan mounts the servo away from the ABS top plate via 4 aluminum standoffs. An external ball bearing in the ABS plate keeps the servo rigid side to side. The system includes a 1/4" bore clamping hub for attaching the head to the ball bearing supported shaft. The kit comes with a 4.25" diameter, 1/4" thick ABS top plate that was used to mount the head to.

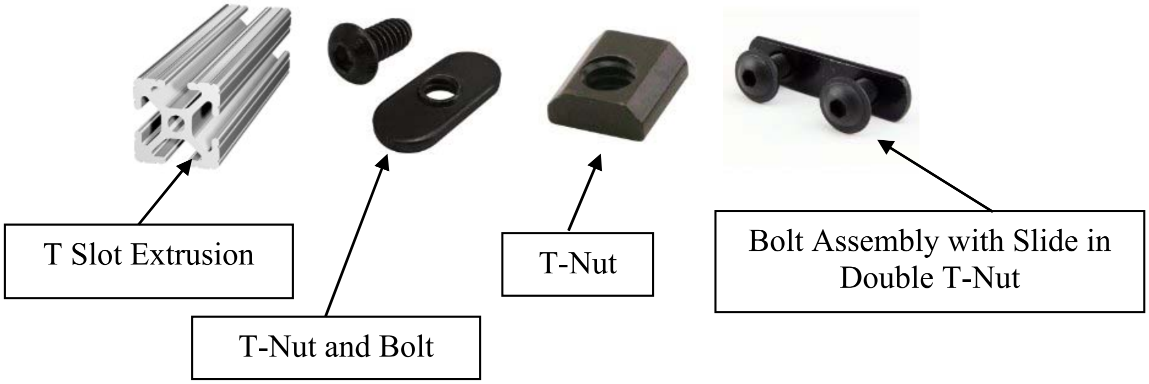


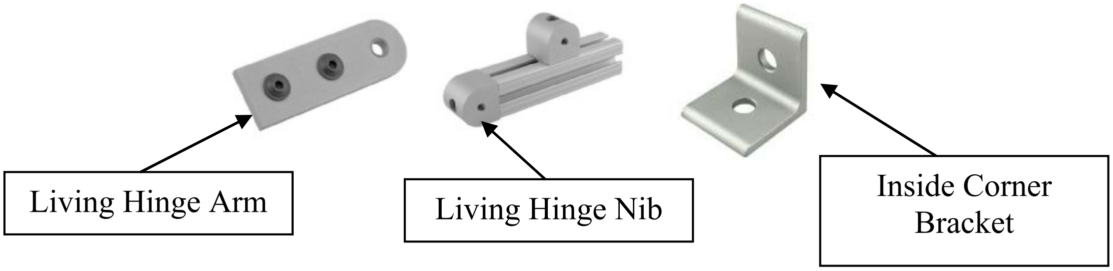
The picture below shows how I mounted the head to the base pan.


Speaker
ABS top plate
Bracket attached to top plate for mounting the head to the base pan
1/4" bore clamping hub
Base Pan

4 Armature

The droid is a display piece that stands on a lighted base. The costume and other components are mounted on an armature that makes up the skeleton of the droid. The armature needed to be stable and sturdy. I decided to use a material called 80/20. The 80/20 material consists of several aluminum extrusion pieces that can be used together to create things. I purchase all of my 80/20 supplies on Amazon. Some basic 80/20 supplies that I use are as follows.

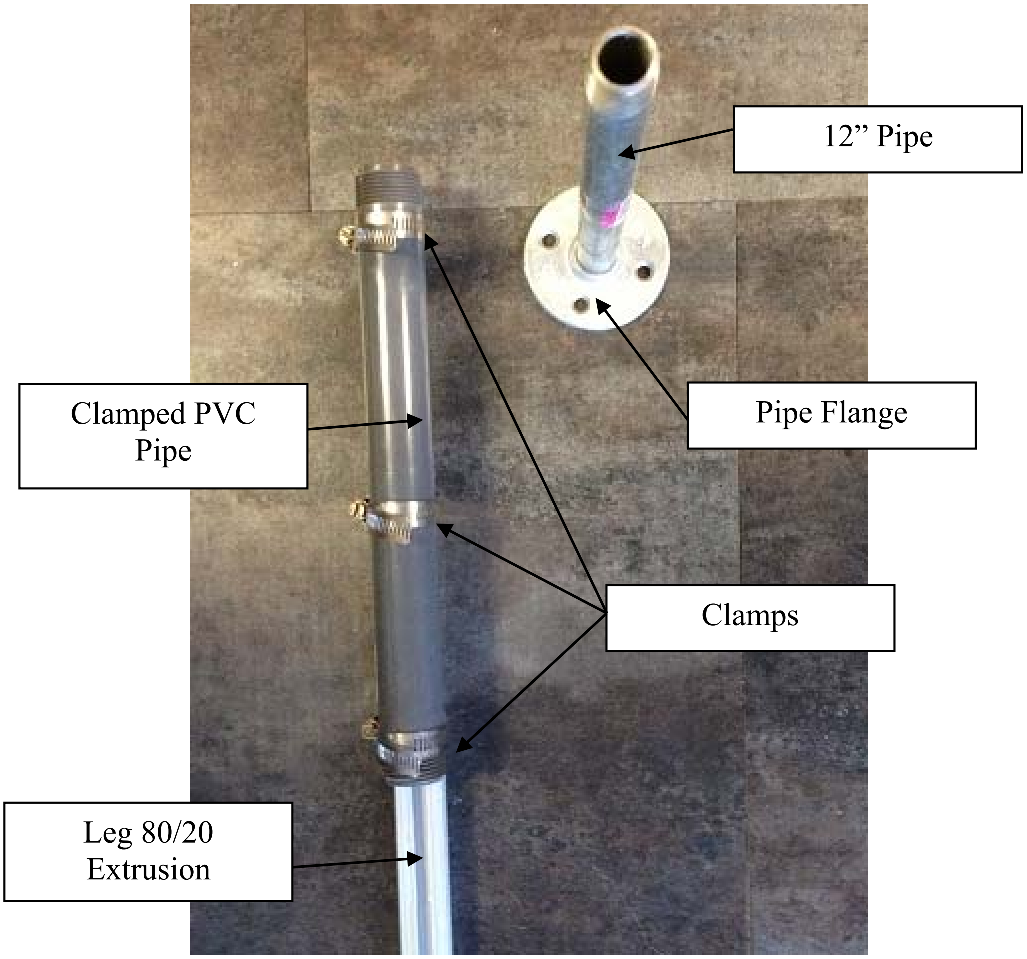




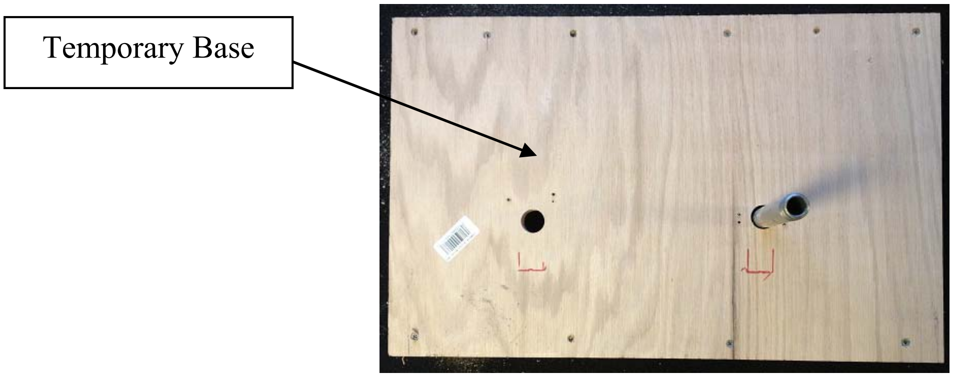
The frame is made of the T slot extrusion pieces. Each leg is made from a single 42" long piece of extruded T slot material. The T slot extrusion must be secured to the base so the droid is stable. The other requirement I placed on this project is that the base is easily removable form the droid in the event maintenance is required. My solution is actually very simple and sturdy. I made a wood temporary base to support the droid. Once the droid was complete, I made a permanent base. Details regarding base construction can be found later in section 8.0 of this document. The leg 80/20 extrusion was clamped to a piece of PVC pipe that would act as a receiver for a pipe that protrudes out of the wood base. With this method, the leg assembly can be slide down on top of the pipe protruding from the base and provide a solid method for holding the droid upright.

The PVC pipe attached to the extrusion simply slides on top of the flange/pipe assembly. I got the pipe and flange at Home Depot.

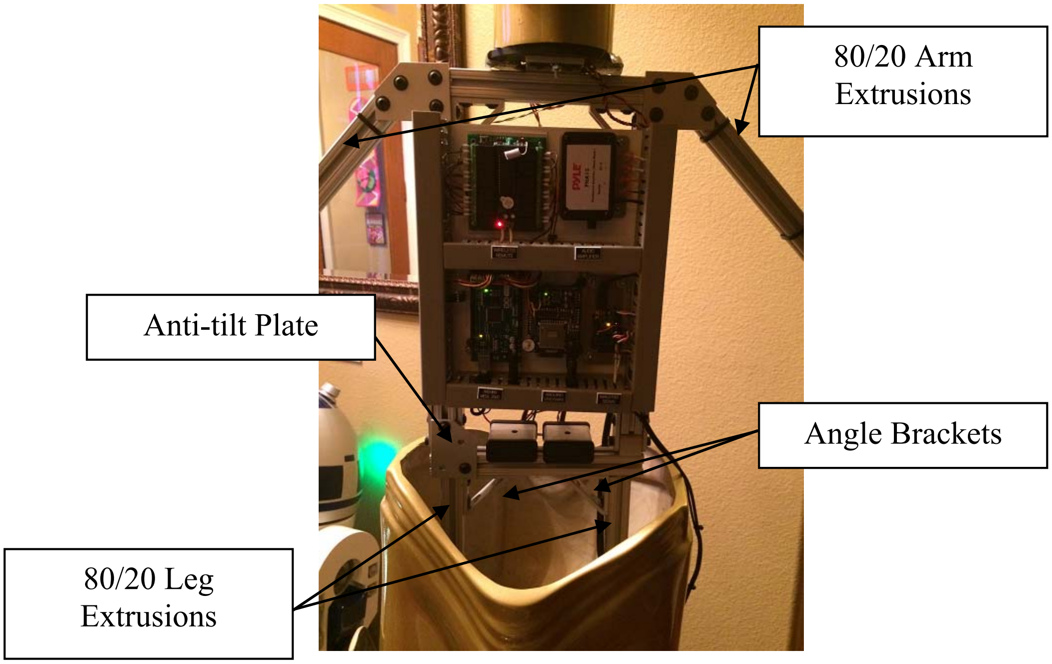
This picture below shows a closer view of the components.



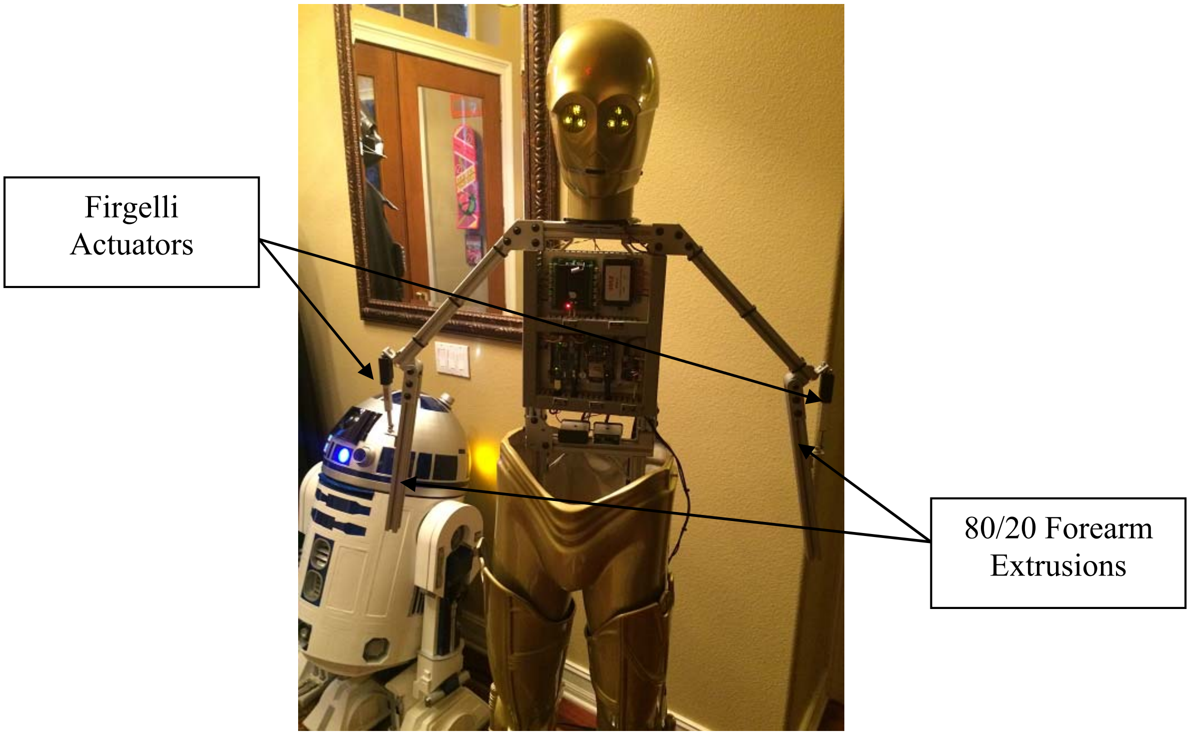
A temporary base made from wood looks like this with the pipe protruding through the top. The flange was mounted on the inside of the base.



The picture below shows more of the armature. The leg extrusions are connected with a cross member piece of extrusion with right angle brackets. I was going to have the C3PO droid tilt forward at the waist so I put living hinges between the leg portion and the torso frame. I decided against this so I used an 80/20 plate as shown to prevent it from tilting.



The elbow and forearm area is shown below. Note how the Firgelli actuators are attached to the arm.



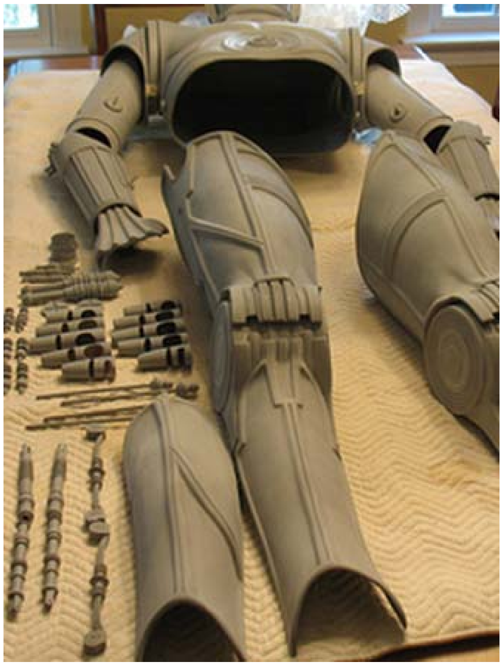
5 Costume Components

The most difficult part of the project was finding a source for costume components. I wish I could provide lots of details but I have not secured permission from the suppliers to do so. I will point you in the right direction and you will need to do a little research to figure out the best method for obtaining parts. Visit C3poBuilders.com since this link will provide a wealth of information and resources to guide you in your quest for parts.

The waiting list for the arms, legs, torso, and hands can be several months but once you receive them, the wait is worth it. This may also help you in your quest for parts.

Vendor 1 - Resin head and feet
Vendor 2 – Shorts
Vendor 3 – Brass head cylinder, hand rods, knuckles
Vendor 4 – Arms, legs, torso, hands

Here is a picture of some of the raw parts as delivered.

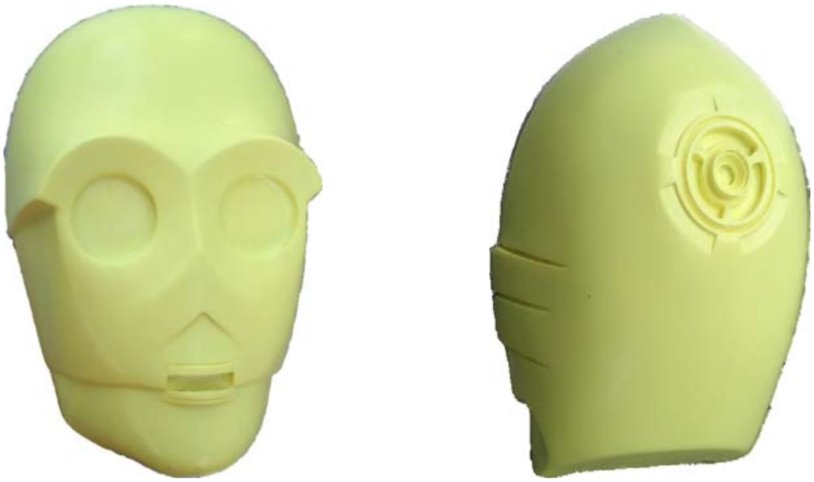


When I sent the parts out to be painted at Chrome Fish Customs in Florida, I created a parts list to give them so we both had a list of parts that I delivered. The list is shown below to give you an idea of all the components that will make up a C3PO droid.

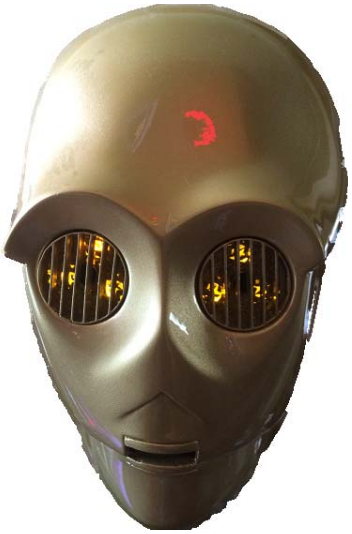
(1) C3PO head front
(1) C3PO head rear
(1) Brass head cylinder
(2) Eye pieces
(1) Neck cover (this is a round plastic piece made from a garage sale sign)
(10) Fingers
(6) Hand cylinders (these are brass)
(1) Left hand cover with brass knuckles attached
(1) Right hand cover with brass knuckles attached
(1) Shorts
(1) Left leg assembly with foot
(1) Right leg assembly with foot
(2) Leg cylinders (fiberglass cylinder pieces)
(1) Torso front
(1) Torso rear
(1) Left forearm cover
(1) Right forearm cover
(1) Left bicep assembly
(1) Right bicep assembly
(2) Forearm rods looking things with curly ends
(6) Small about 2" half cylinders
(4) Arm cylinders
(4) Arm cylinder rods (metal rod with end piece)

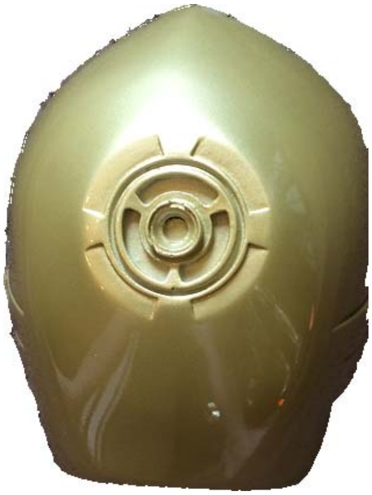
5.1 Resin Head

The resin head came in two parts. The head is molded so a Dremel tool is a must for cutting out the back of the molds. A Dremel tool was also used to cutout the mouth opening and a hole saw was used to cutout the eye opening. The raw head pieces are show below.

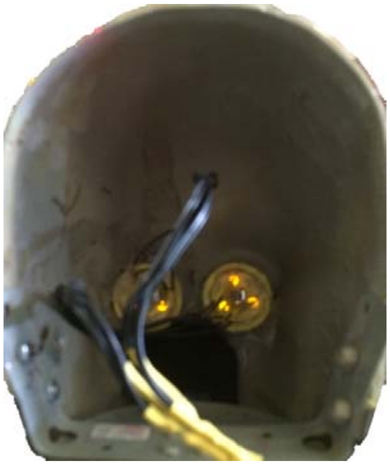


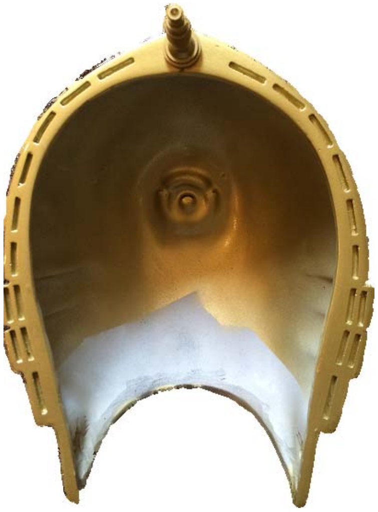
The completed head pieces are shown below.





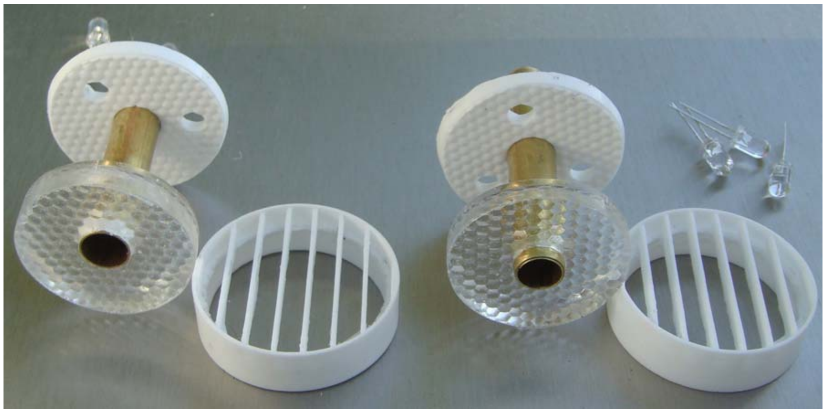
An inside view of the head pieces are shown below.





5.2 Eyes

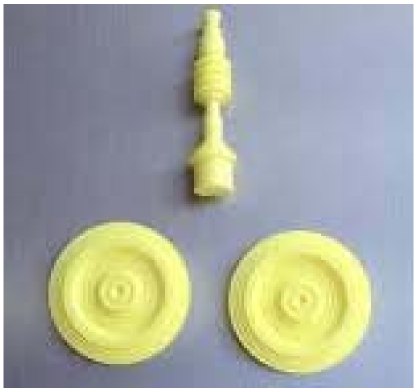
The eyes came with the resin head. The picture below shows the basic layout of the eye pieces that come with the head.



I used yellow 5 mm leds I purchased from radio shack since I lost the leds that came with the eyes. For details on wiring the eyes, refer to the attached electrical drawings.

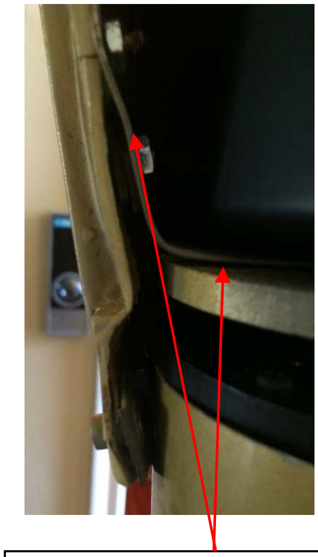
5.3 Ears

Resin ears and a resin head antenna came with the head as well. The raw parts look like the following.

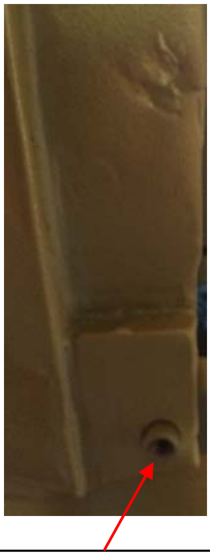


I used a brass antenna that I purchased from a different vendor.

The front of the head attaches to the servo city base via a bracket I made. I use a rivet nut to allow the back of the head to attach to the front of the head with a flat head Philips screw on each side of the head. The screw heads are covered by the ears. The arrangement is shown below.

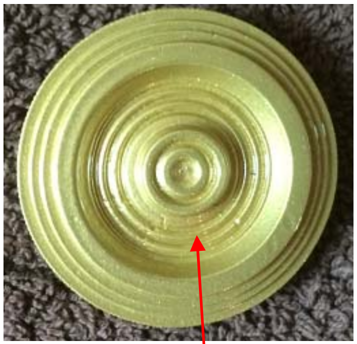


Bracket that attaches front of face to head turn base

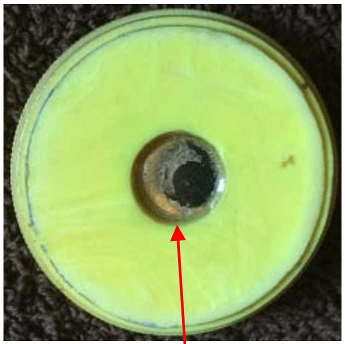


Rivet nut used to attach the back of head to the front of head

The ears need to cover the screw heads and stay in place but have the ability to be removed easily. My solution was to embed a magnet on the inside of each ear. The ears just snap over the screw heads via magnetic force. Pictures of the ears are shown below.



Outside Ear

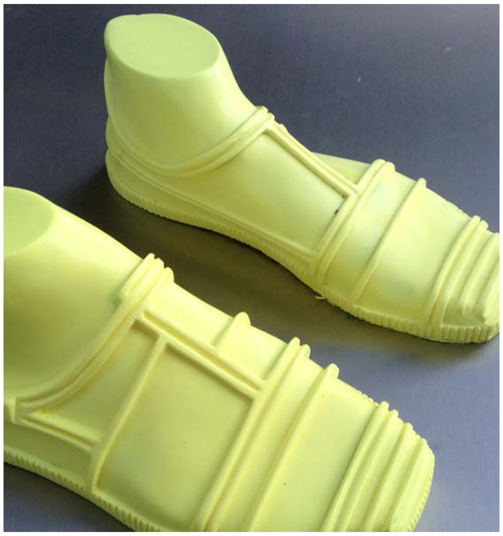


Inside Ear with magnet

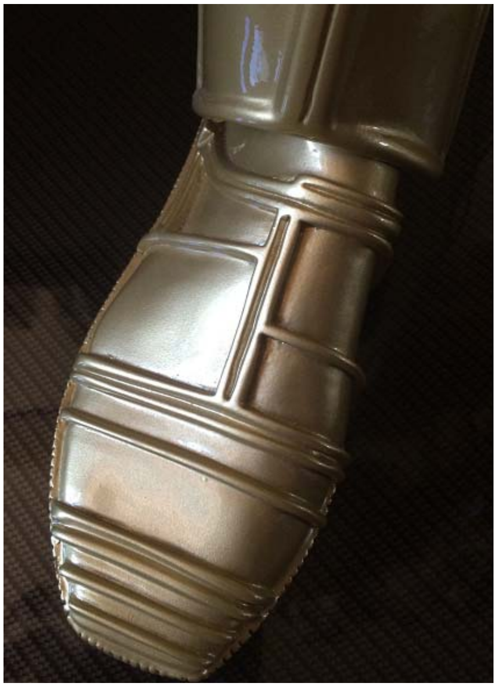
5.4 Resin Feet

I ordered the resin feet from the same vendor as the resin head. Since my C3PO is a display unit only, I needed to drill out the ankle area to allow an 80/20 extrusion to fit through the feet. The 80/20 is the armature for the droid.

The picture below represents the product in the way that you will receive it.

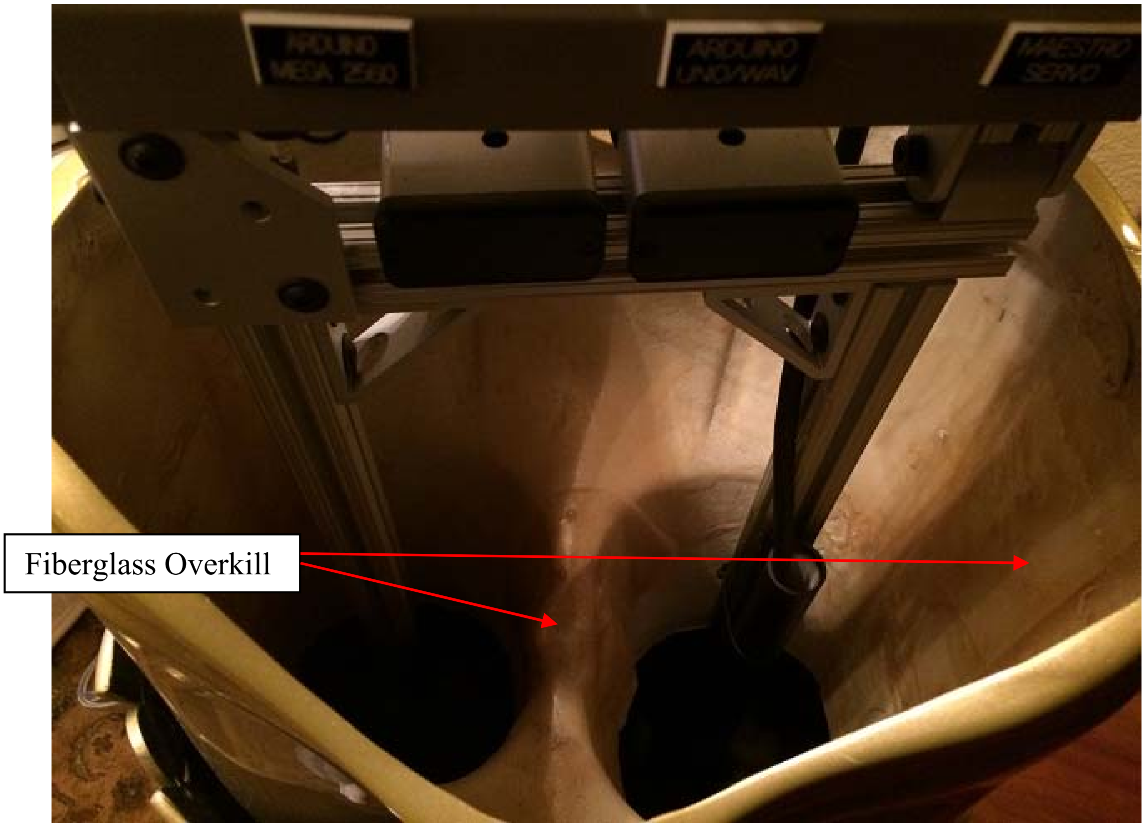


The picture below represents the product when it is finished.



5.5 Shorts

The shorts arrived as two vacuformed pieces of plastic. A plastic strip was included to act as a gluing surface for joining the two halves. I trimmed the plastic and used PVC glue. The type that is used for PVC pipe. The short material was a little thin in areas and the glue cause some deformation / melting in some areas. I wanted the shorts to be strong so I used automotive fiberglass to reinforce the inside of the shorts.

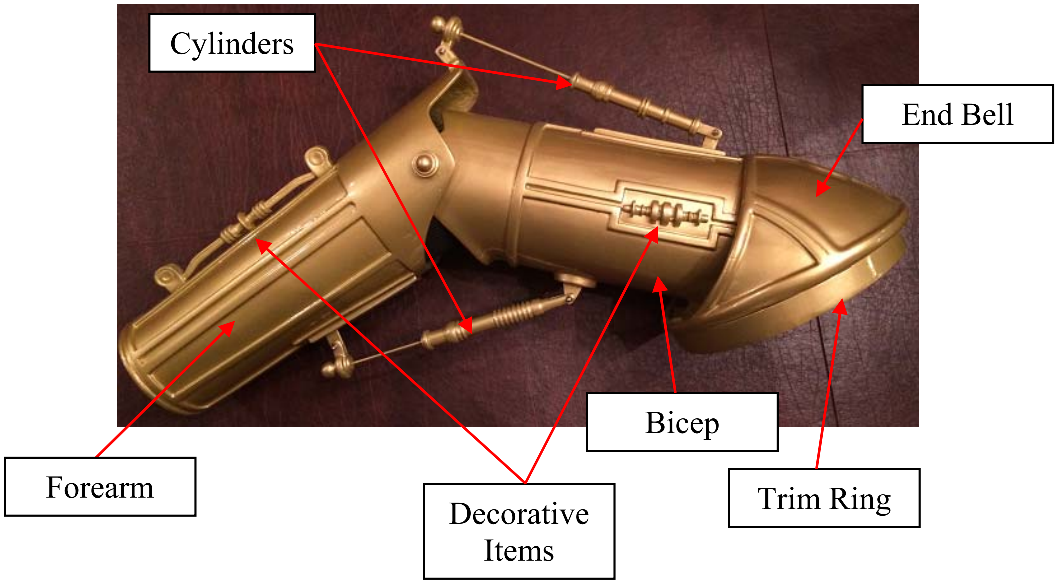


Fiberglass Overkill

Creating an even lip around the top of the shorts was time consuming. I used Bondo and a copious amount of sanding to make the outside of the shorts look good.

5.6 Arms

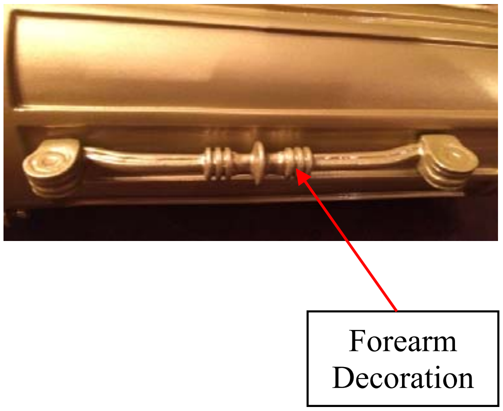
The arms are comprised of a forearm sleeve, bicep sleeve, two cylinders, upper arm end bell, end bell trim ring, and some decorative items.

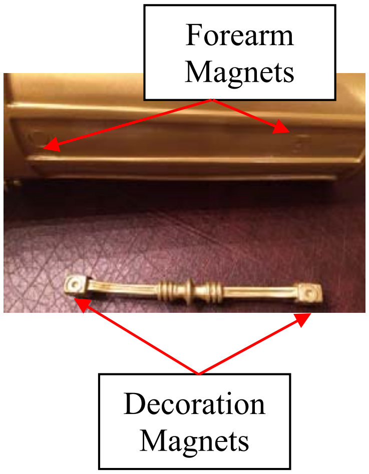


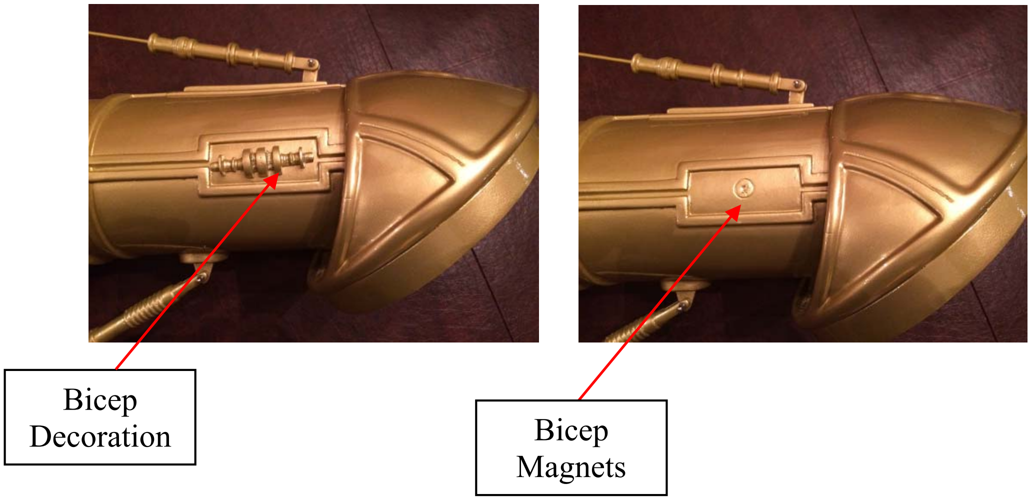
The forearm and the bicep pivot via the decorative cap screws shown. I fused the bicep parts as one solid piece using automotive fiberglass.

5.6.1 Decorative Item Attachment

The decorative items shown below are mounted on portions of the arm. I wanted the items removable so painting was an easier chore. To do this, I made everything attach magnetically. I embedded magnets in the arms and on the decorative items so they would stick on the arms easily and still be removable.



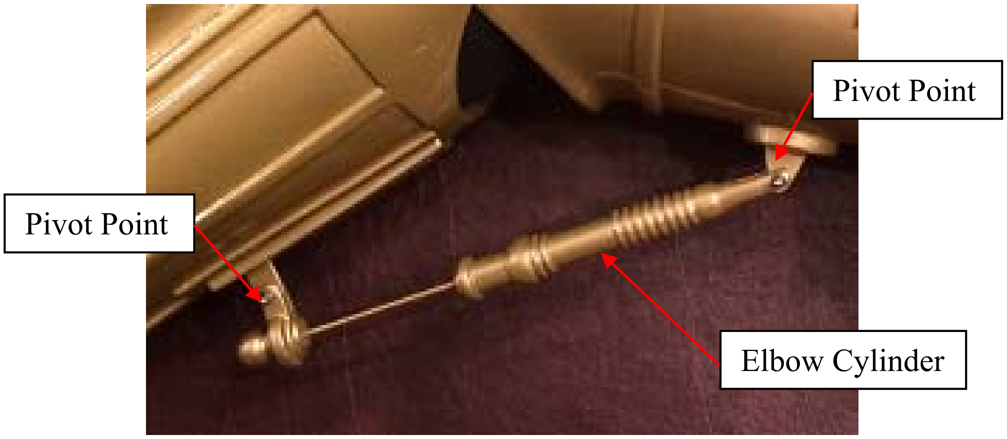


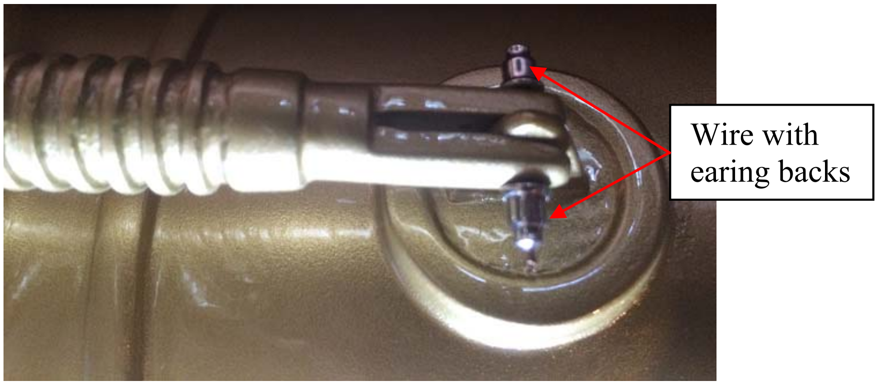


The bicep decorative item was already magnetic as delivered.

5.6.2 Cylinders

Each arm has two cylinders that give the illusion that the arm movement is caused by the cylinders. The cylinders just have a cap with a wire rod sticking out of it on one end and the other end is just a fiberglass cylinder with a hole in it to receive the wire rod. When the elbow extends and retracts, the cylinder looks like it is operating. The cylinder ends need to pivot about the cylinder mount. To do this, I used hobby wire and attached earring backs I got at Michaels hobby store.

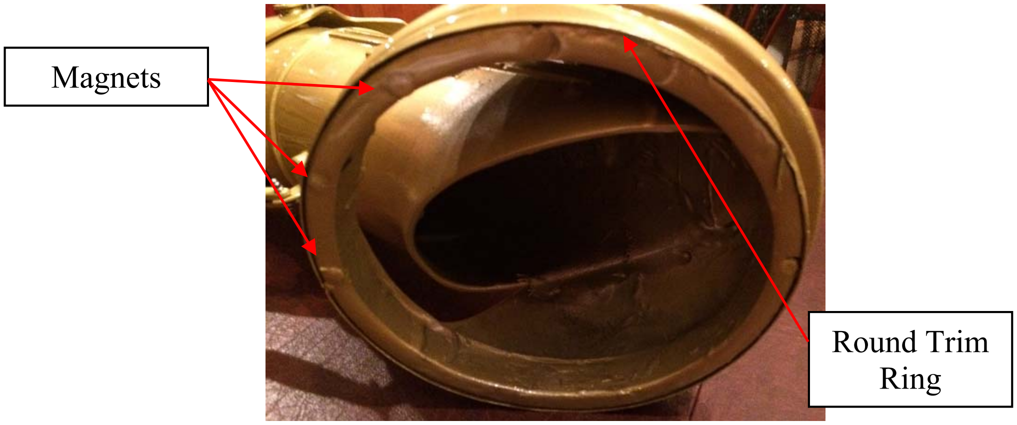


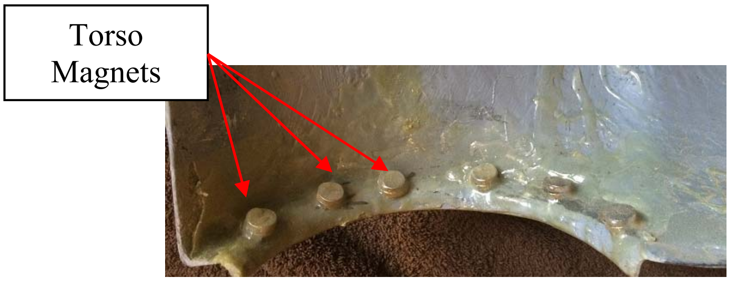


5.6.3 Round Trim Ring

The round trim ring had grooves engraved into it that were not uniform. I glued ¾" edging material used for plywood to the round trim rings. This provided a uniform and smooth surface for the outer edge of the trim ring.

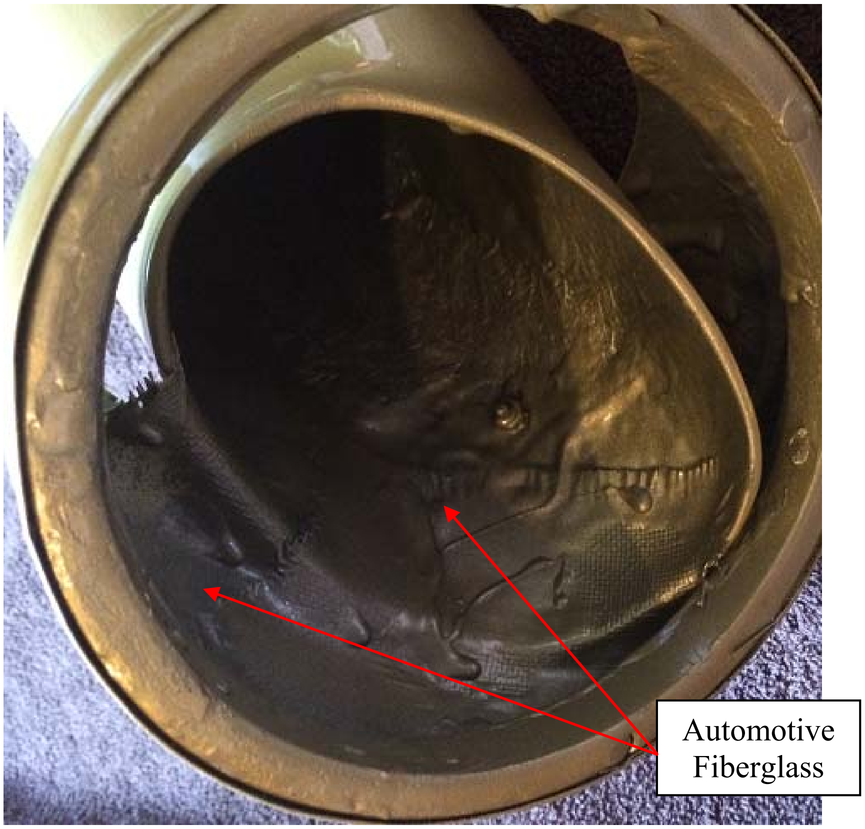
The trim ring fits over the torso. The arms needed to be removable so I embedded magnets in the trim rings and used epoxy on the inside of the torso to embed even more magnets where the rings slides over. I got the magnets at Home Depot.





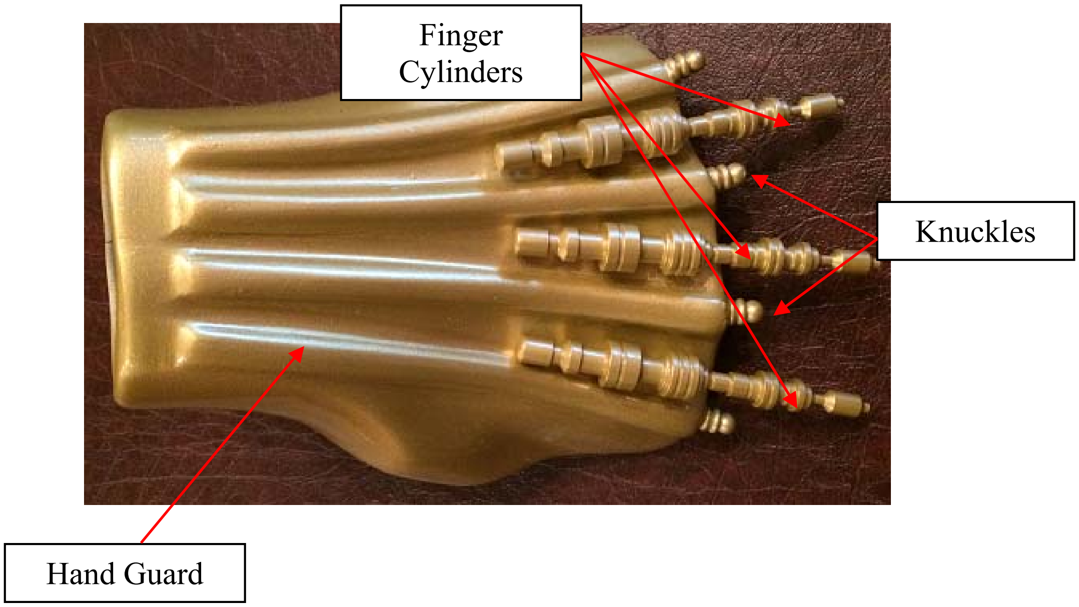
5.6.4 Upper Arm Attachment

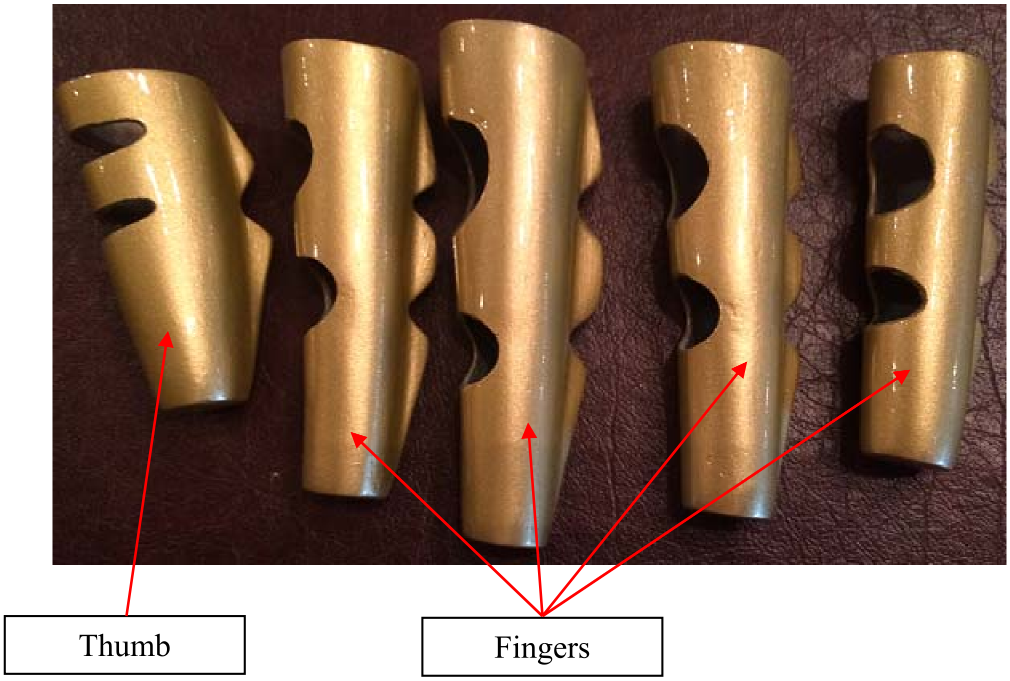
The upper bicep piece is attached to the end bell and round trim ring using automotive fiberglass. I joined these pieces together so they become one assembly. They will slide over the arm extrusion and snap on the torso via the magnets.



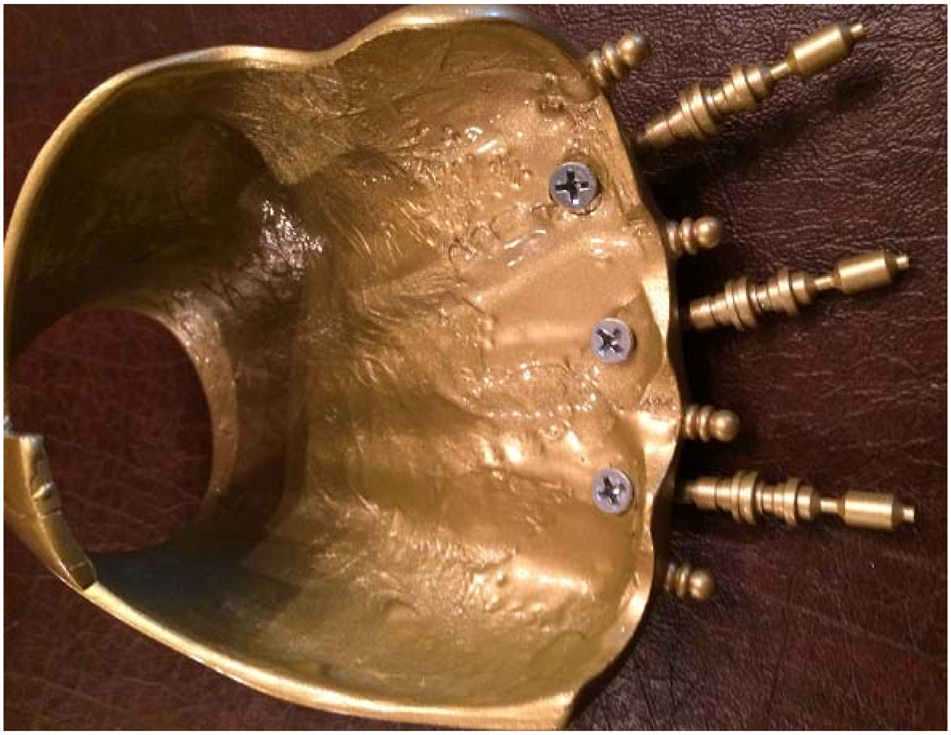
5.7 Hands

The hands consist of fingers, hand guard, knuckles, and finger cylinders. The fingers and hand guard came with the arm, leg, and torso parts. Since I am not using wrist rotation or finger movement, I could have gone with a fixed display hand. Since that is not what I chose initially, I had to make this work. The knuckles and finger cylinders are made from brass and I ordered these from the same vendor that supplied all of the other brass components.





The back side of the hand guard is shown. The finger cylinders just attach using screws and the knuckles have a threaded post on them. I tapped the hand to thread the knuckle in. I added some epoxy to the threads so they will stay put.

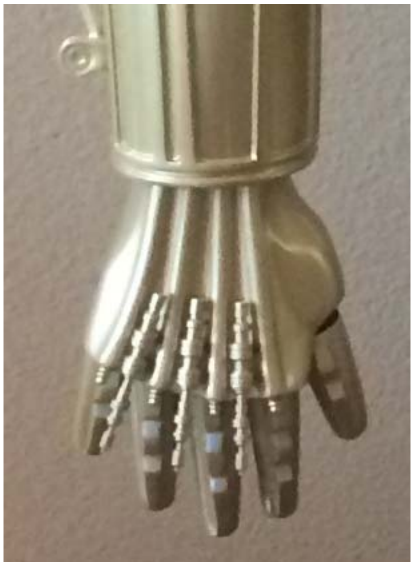


I used Halloween hands as the skeletal structure that I got at the dollar store. I cut the fingers to the proper length using a pair of scissors.

I used some flat aluminum stock that I purchased from Home Depot and affixed it to the back of the Halloween hand. The hand attaches to the forearm extrusion on the armature via the flat stock.

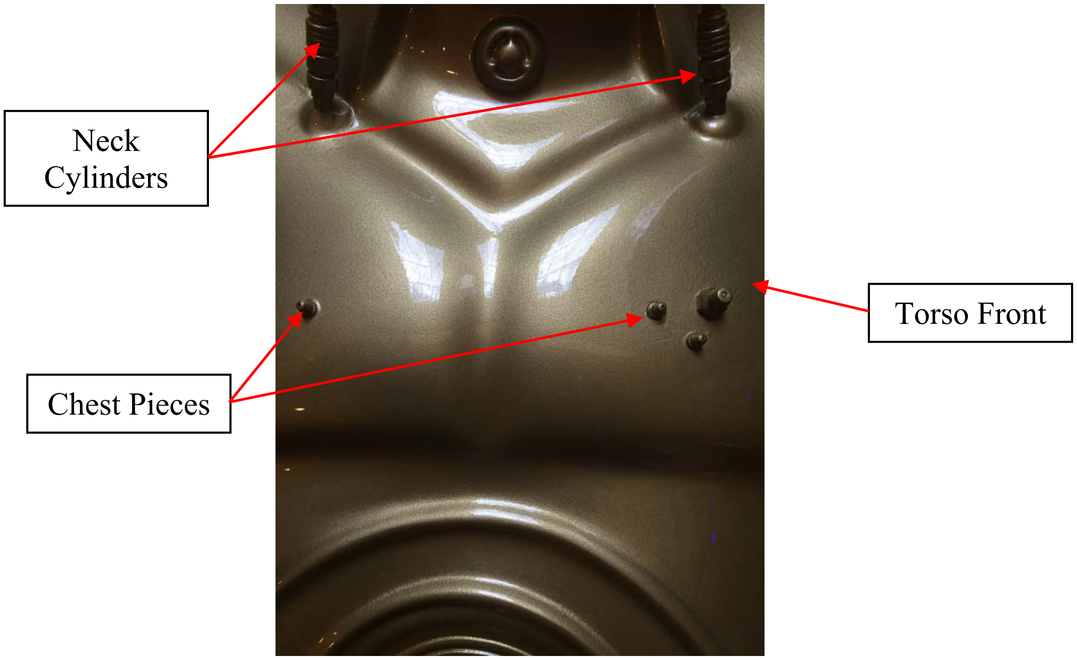
I covered the plastic hand assembly with an inexpensive glove purchased from Harbor Freight. A little Velcro for the hand guard and some quick set on the fingertips of the glove to hold the fiberglass fingers on and the hand is done.

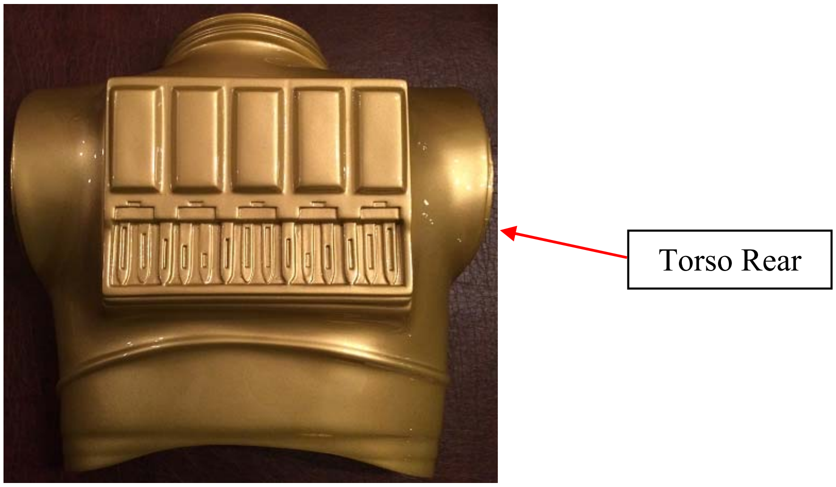
A completed hand is shown in the picture below.



5.8 Torso

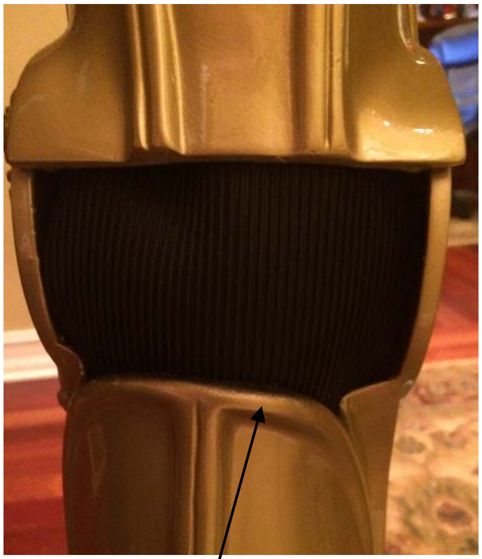
The torso consists of a front and back piece. The front piece requires that a few items were added to resemble the actual C3PO from the movies. The neck cylinders were purchased separately since I wanted to use brass cylinders. The Torso came with fiberglass cylinders but the brass ones are more detailed. The small chest pieces are earing backs and the larger piece is an oil fitting I got at AutoZone.



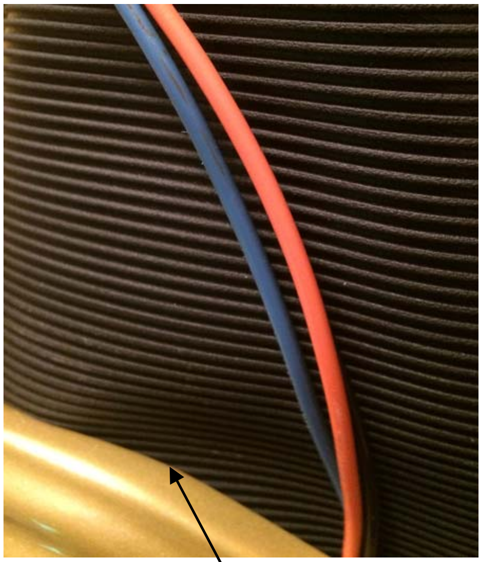


6 Ribbing

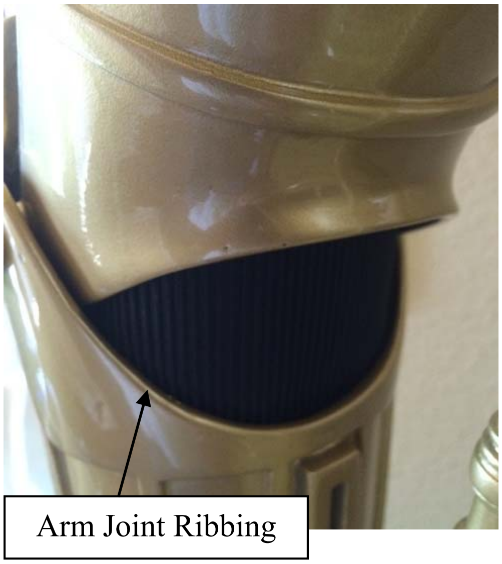
The stomach area, elbow joints, and knee joints use a black ribbing material. The material was purchased at Lowes in the flooring section where they have the carpet runners and such. The material is purchased by the linear foot. The material is flexible and can be cut with scissors. The pictures below show examples of how I used it at the knee joints and in the stomach area.



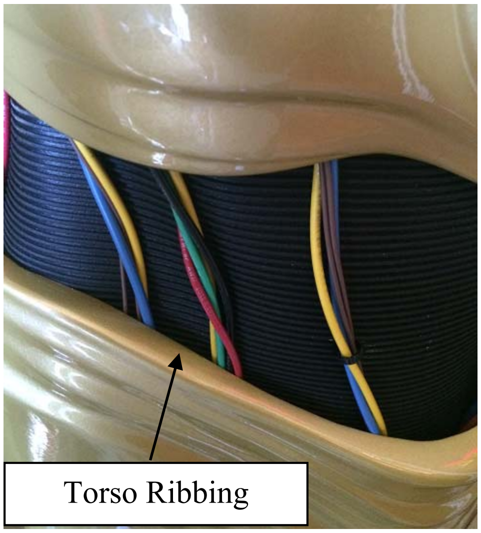
Knee Joint Ribbing



Torso Ribbing



Arm Joint Ribbing

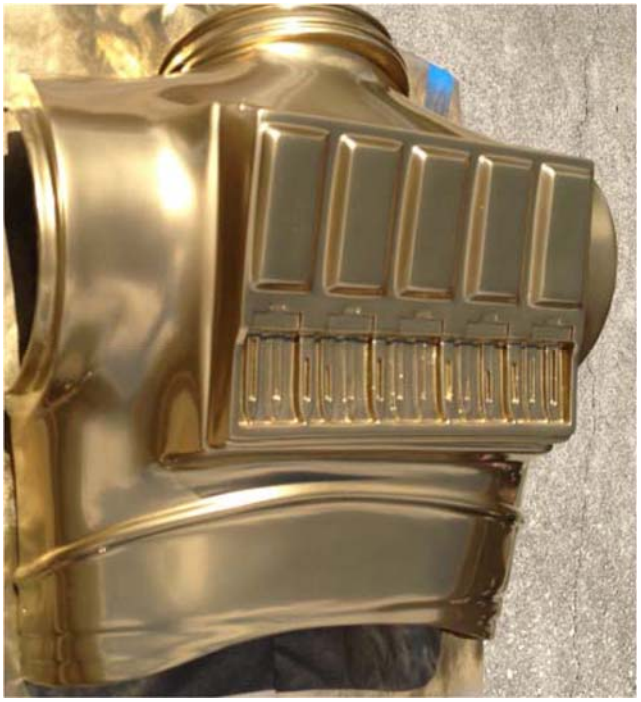


Torso Ribbing

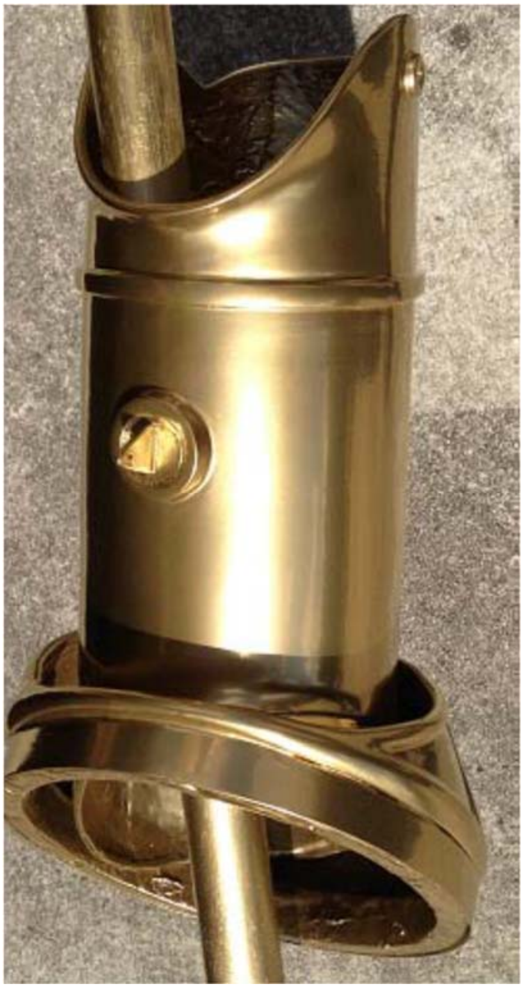
7 Paint

I spent some serious time deciding whether to paint the droid or get it vacuum metalized. My preference was the vacuum route but I could not find anyone in the state that does it. I found organizations across the country but I spent so much time, money, and effort getting the parts together, I was not going to take a chance on packing it up and shipping it. I originally painted the parts myself and it looked like gold chrome. The problem was that if you wiped it with even a dry rag, it would destroy the surface, terrible dulling, and black marks. I tried every paint except Alcad. Here are some pictures using Krylon Gold Leaf Foil Metallic

A black gloss coat was followed by the gold leaf metallic. The torso back and upper arm are shown in the following pictures.
.



Torso Back

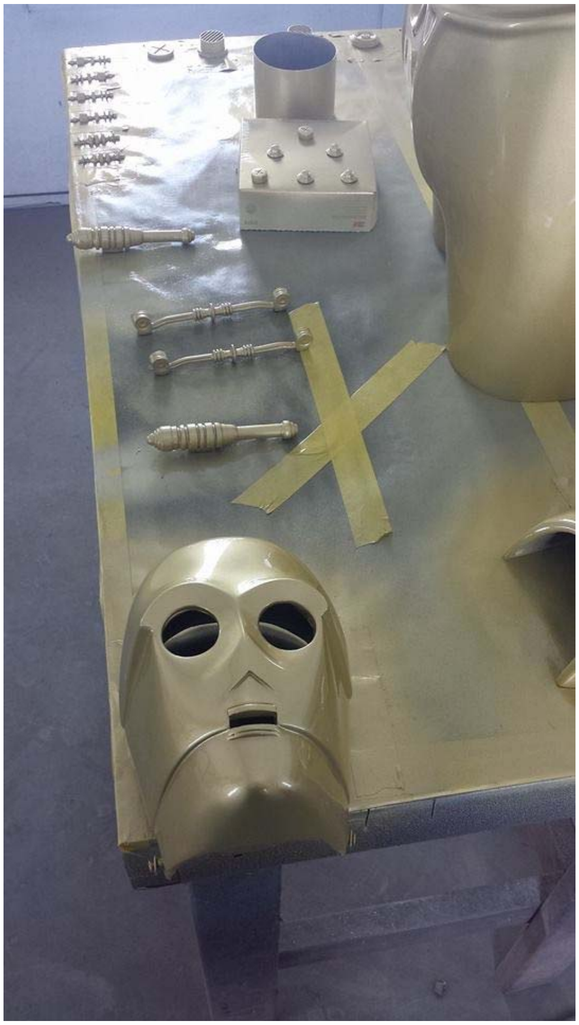


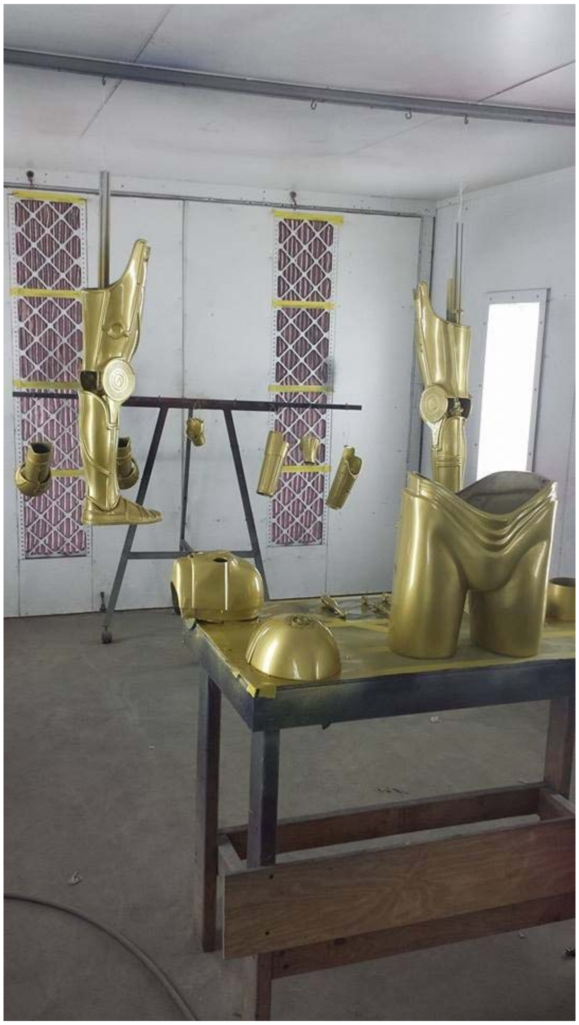
Upper Arm

I was very happy with the results and prefer this look over what I currently have. I tried several things such as letting it fully dry for two weeks, coating with water base poly, car wax, etc. Everything I tried destroyed the finish. I finally ended up taking it to Chrome Fish Customs and had them paint it. They do automotive painting and the Hydro Dipping shown in section 8 of this document.

The automotive paint looks good and is durable but it does not replicate the C3PO finish that we are accustomed to. If a paint ever comes out down the road where I can get a finish like that in the pictures above, I will redo it. I could not find anyone near Orlando that does the spray on gold or vacuum metalizing

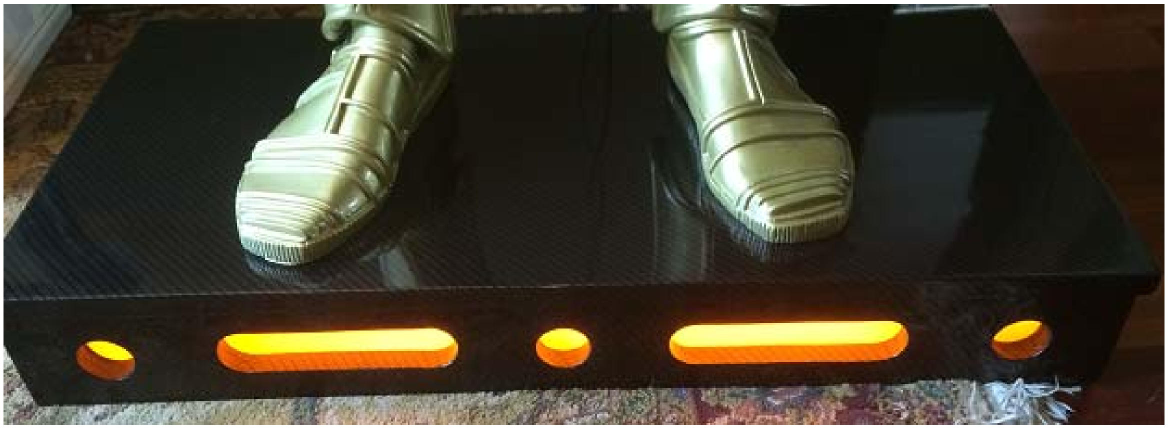
Here are some pictures of the parts in the paint booth at Chrome Fish Customs.





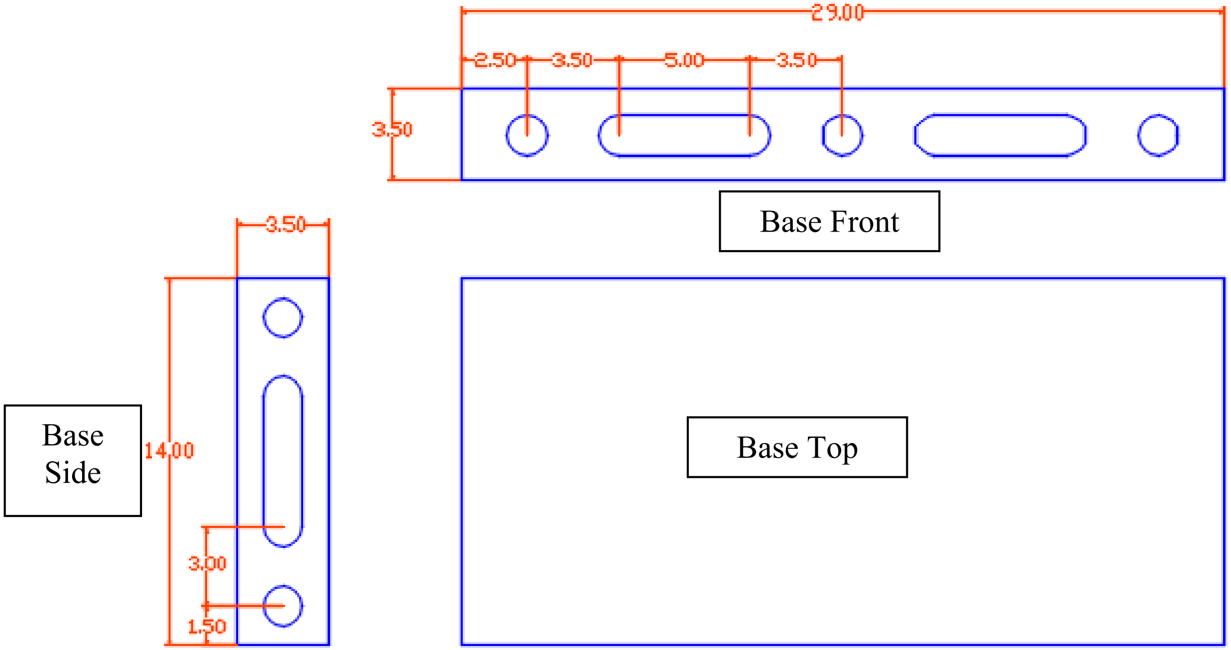
8 Droid Base

I wanted C3PO to stand upon an unusual back lit base. I made the base out of wood and once it was complete, I had it immersion printed using a Hydrographic system by Chrome Fish Customs so it looked like it was made from black carbon fiber. The base I created is shown below.



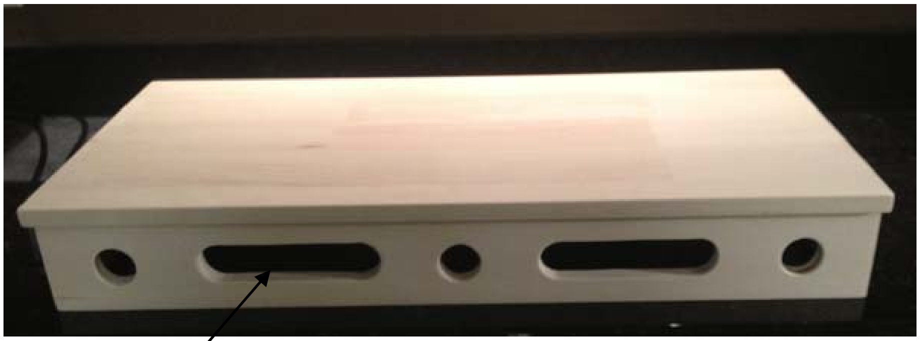
8.1 Base Construction Detail

I first made some plans regarding dimensions and cutouts required for the base. The drawing below illustrates dimensions for the base top and the 4 sides that were made. The wood is ¾ "poplar that was obtained from Home Depot. The circles and were drilled using a 1.5" hole saw and to get the oval shape, two 1.5 "holes were drilled and a jig saw was used to remove the wood that remained between the two holes.



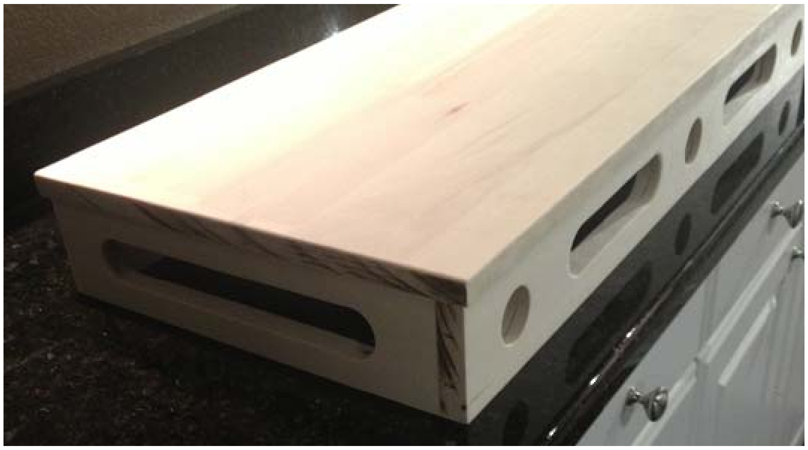
With the boards cut and the end pieces sanded down, the base will look like the following.

Here is a picture of the base from the front view.



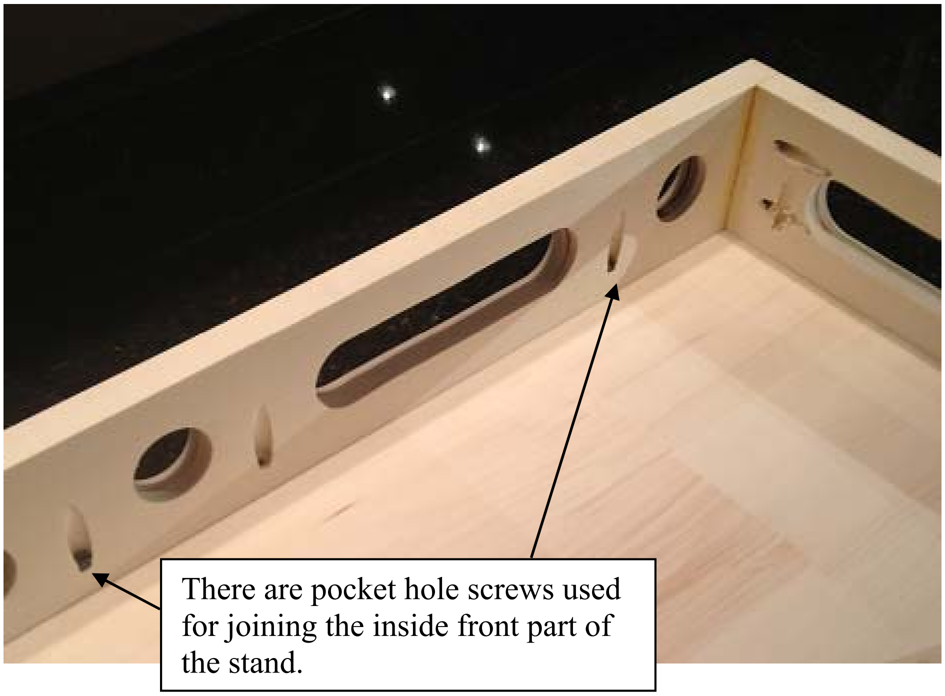
All boards are ¾" thick. The backlit openings have an opaque lens mounted to the back of each opening after the base was hydro dipped.

Here is a picture of the base from the side view. Both sides have the same pattern. As you can see, I went with one long oval instead of the two circles and oval shown in the layout drawing.



Here is a picture of the base showing the back view. The back is just solid.

The base is screwed and glued so it will not will not come apart and is very sturdy. Notice that I used pocket holes for the screws. I accomplish this with a Kreg jig. I use the Kreg jig for all wood working projects. It is easy to use and makes a rock slid hidden connection that is much easier than biscuit joining. You can get the Kreg jig, screws, square head bits, and everything you need at Lowes or Woodcraft.

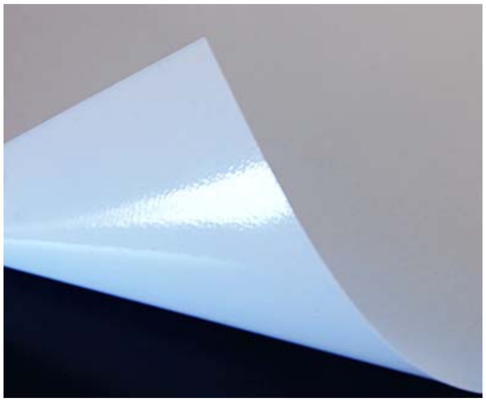


There are pocket hole screws used for joining the inside front part of the stand.

Now that the base is complete, it was off to Chrome Fish Customs to be Hydro Dipped. While I waited for the base to be finished, I got everything needed for the base backlight system.

8.2 Base Backlighting

The goal was to have light emitting from the cutout areas of the base. Once the base returned from Hydro Dipping, I affixed .020 "LED light panel flexible acrylic sheet material on the inside of the base to cover the cutout areas. The material provides a diffuse matte white area that the light can transmit through. The material shown below can be cut with scissors.



I ordered this material from www.tapplastics.com, it is sold as a roll that is 25" x 50" and it is not expensive.

The only thing left was to add light to the inside of the base. I decided to use an LED string since it would not get hot and should last a life time with no re-lamping issues. I ended up using a SUPERNIGHT 16.4 foot 12 volt SMD 5050 300 LED yellow waterproof flexible lamp light strip shown below.

Now that the LED light was decided upon, I need a way to power it. The Led light strip requires a 12 VDC source. I was not sure what else I might do so I decided to go with a very robust 12 VDC transformer. I could have used a smaller one but overkill is your friend. The item I used is shown below. It is a LEDwholesalers waterproof LED driver transformer rated at 60 watts and 12 volts. I ordered both of these items from Amazon.

I added four small wood blocks to the inside of the base arranged in a rectangle pattern and attached the LED strip to the blocks by wrapping it around the rectangle pattern. The Led strip connects to the transformer. I added a junction box inside the base that connected all of the 120 VAC stuff. The transformer needs 120 VAC power and I needed to feed 120 VAC up through the droid to the control panel in his torso for power distribution. Therefore, I added a hole to the back of the base that was finished with a nice cord grip and the main power cord fed through this hole and was connected to the junction box inside the base.

9 Program Listing

The code for the Arduino Mega, Arduino Uno, and the Maestro servo controller is shown in the following sections.

9.1 Arduino Mega 2560

The Arduino Mega 2560 reads the remote relay triggers and sends a 4 digit binary code to the Arduino Uno that represents the desired audio file to be played. The Arduino Mega 2560 also sends a subroutine number to the Maestro servo controller that represents the motion that should be played based on the remote trigger. Refer to the electrical drawings for hookup information.

```
//John Guarnero
//September 2014
//serial to Maestro config
#include <SoftwareSerial.h>
#define txPin 14
#define rxPin 15
#define sub0 0
SoftwareSerial mySerial(rxPin, txPin);

//Audio and Remote Setup
int Push1 = 0;
int Push2 = 0;
int Push3 = 0;
int Push4 = 0;
int Push5 = 0;
int Push6 = 0;
int Push7 = 0;
int Push8 = 0;
int Push9 = 0;
int Push10 = 0;
int Push11 = 0;
int Push12 = 0;
void setup()

{
//serial to Maestro config
mySerial.begin(9600);
delay(1000);

//define pins to use for binary code to Arduino Uno
pinMode(22, OUTPUT);
pinMode(24, OUTPUT);
pinMode(28, OUTPUT);
pinMode(30, OUTPUT);

//Make Binary Code = 0
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8

//Define all remote trigger pins as inputs with internal pullup resistor
pinMode(31, INPUT_PULLUP); //Relay 1
digitalWrite (31, HIGH);  //Enable Pullup
pinMode(33, INPUT_PULLUP); //Relay 2
digitalWrite (33, HIGH);  //Enable Pullup
pinMode(35, INPUT_PULLUP); //Relay 3
digitalWrite (35, HIGH);  //Enable Pullup
pinMode(37, INPUT_PULLUP); //Relay 4
digitalWrite (37, HIGH);  //Enable Pullup
pinMode(39, INPUT_PULLUP); //Relay 5
digitalWrite (39, HIGH);  //Enable Pullup
pinMode(41, INPUT_PULLUP); //Relay 6
digitalWrite (41, HIGH);  //Enable Pullup
pinMode(43, INPUT_PULLUP); //Relay 7
digitalWrite (33, HIGH);  //Enable Pullup
pinMode(45, INPUT_PULLUP); //Relay 8
digitalWrite (45, HIGH);  //Enable Pullup
pinMode(47, INPUT_PULLUP); //Relay 9
digitalWrite (47, HIGH);  //Enable Pullup
pinMode(49, INPUT_PULLUP); //Relay 10
digitalWrite (49, HIGH);  //Enable Pullup
pinMode(51, INPUT_PULLUP); //Relay 11
digitalWrite (51, HIGH);  //Enable Pullup
pinMode(53, INPUT_PULLUP); //Relay 12
digitalWrite (51, HIGH);  //Enable Pullup

//Serial.begin(9600);
Serial.println("John Guarnero 2014 C3PO Arduno Mega");

}
void loop()
{
```

```
//read the remote trigger inputs
Push1 = digitalRead(31);
Push2 = digitalRead(33);
Push3 = digitalRead(35);
Push4 = digitalRead(37);
Push5 = digitalRead(39);
Push6 = digitalRead(41);
Push7 = digitalRead(43);
Push8 = digitalRead(45);
Push9 = digitalRead(47);
Push10 = digitalRead(49);
Push11 = digitalRead(51);
Push12 = digitalRead(53);

//If show 1 triggered, initate sequence
if (Push1 == 0)
{
//Serial.println("Happy Birthday Emily");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run maestro subroutine
 mySerial.write((byte)0x00); //device id

//Send Value of Binary 1 To Uno to Play WAV File 1
digitalWrite (22, HIGH);  // 1 2 4 8
digitalWrite (24, LOW);   // 1 2 4 8
digitalWrite (28, LOW);   // 1 2 4 8
digitalWrite (30, LOW);   // 1 2 4 8
delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 2 triggered, initate sequence
if (Push2 == 0)
{
//Serial.println("Happy Birthday Kim");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x01); //device id

//Send Value of Binary 2 To Uno to Play WAV File 2
digitalWrite (22, LOW);   // 1 2 4 8
digitalWrite (24, HIGH);  // 1 2 4 8
digitalWrite (28, LOW);   // 1 2 4 8
digitalWrite (30, LOW);   // 1 2 4 8
delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 3 triggered, initate sequence
if (Push3 == 0)
{
//Serial.println("Happy Birthday Megan");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x02); //device id

//Send Value of Binary 3 To Uno to Play WAV File 3
digitalWrite (22, HIGH);  // 1 2 4 8
digitalWrite (24, HIGH);  // 1 2 4 8
digitalWrite (28, LOW);   // 1 2 4 8
digitalWrite (30, LOW);   // 1 2 4 8
delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 4 triggered, initate sequence
if (Push4 == 0)
{
//Serial.println("Guarnero House");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x03); //device id

//Send Value of Binary 4 To Uno to Play WAV File 4
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, HIGH); // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 5 triggered, initate sequence
if (Push5 == 0)
{
//Serial.println("Halloween");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x04); //device id

//Send Value of Binary 5 To Uno to Play WAV File 5
digitalWrite (22, HIGH);  // 1 2 4 8
digitalWrite (24, LOW);   // 1 2 4 8
digitalWrite (28, HIGH);  // 1 2 4 8
digitalWrite (30, LOW);   // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 6 triggered, initate sequence
if (Push6 == 0)
{
//Serial.println("Happy Anniversary");
```

```
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x05); //device id

//Send Value of Binary 6 To Uno to Play WAV File 6
digitalWrite (22, LOW);   // 1 2 4 8
digitalWrite (24, HIGH);  // 1 2 4 8
digitalWrite (28, HIGH);  // 1 2 4 8
digitalWrite (30, LOW);   // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 7 triggered, initate sequence
if (Push7 == 0)
{
//Serial.println("Jokes");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x06); //device id

//Send Value of Binary 7 To Uno to Play WAV File 7
digitalWrite (22, HIGH);  // 1 2 4 8
digitalWrite (24, HIGH);  // 1 2 4 8
digitalWrite (28, HIGH);  // 1 2 4 8
digitalWrite (30, LOW);   // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 8 triggered, initate sequence
```

```
if (Push8 == 0)
{
//Serial.println("Kim Saint & Kim R2D2");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x07); //device id

//Send Value of Binary 8 To Uno to Play WAV File 8
digitalWrite (22, LOW);   // 1 2 4 8
digitalWrite (24, LOW);   // 1 2 4 8
digitalWrite (28, LOW);   // 1 2 4 8
digitalWrite (30, HIGH);  // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 9 triggered, initate sequence
if (Push9 == 0)
{
//Serial.println("Happy Thanksgiving");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x08); //device id

//Send Value of Binary 9 To Uno to Play WAV File 9
digitalWrite (22, HIGH);  // 1 2 4 8
digitalWrite (24, LOW);   // 1 2 4 8
digitalWrite (28, LOW);   // 1 2 4 8
digitalWrite (30, HIGH);  // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
```

```
}

//If show 10 triggered, initate sequence
if (Push10 == 0)
{
//Serial.println("Kims Buns");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x09); //device id

//Send Value of Binary 10 To Uno to Play WAV File 10
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, HIGH); // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, HIGH); // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 11 triggered, initate sequence
if (Push11 == 0)
{
//Serial.println("Master John & Lost In Space");
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x0A); //device id

//Send Value of Binary 11 To Uno to Play WAV File 11
digitalWrite (22, HIGH);  // 1 2 4 8
digitalWrite (24, HIGH);  // 1 2 4 8
digitalWrite (28, LOW);   // 1 2 4 8
digitalWrite (30, HIGH);  // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}

//If show 12 triggered, initate sequence
if (Push12 == 0)
{
//Serial.println("Merry Christmas" );
 if (mySerial.available())
 Serial.write(mySerial.read());
 if (Serial.available())
 mySerial.write(Serial.read());
 mySerial.write(0xA7); //run subroutine
 mySerial.write((byte)0x0B); //device id

//Send Value of Binary 12 To Uno to Play WAV File 12
digitalWrite (22, LOW);   // 1 2 4 8
digitalWrite (24, LOW);   // 1 2 4 8
digitalWrite (28, HIGH);  // 1 2 4 8
digitalWrite (30, HIGH);  // 1 2 4 8

delay (500);
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}


if (Push1 == 1 && Push2 == 1 && Push3 == 1 && Push4 == 1 && Push5 == 1 &&
Push6 == 1 && Push7 == 1 && Push8 == 1 && Push9 == 1 && Push10 == 1 &&
Push11 == 1 && Push12 == 1)
{
digitalWrite (22, LOW);  // 1 2 4 8
digitalWrite (24, LOW);  // 1 2 4 8
digitalWrite (28, LOW);  // 1 2 4 8
digitalWrite (30, LOW);  // 1 2 4 8
}
}
```

9.2 Arduino Uno R3

The Arduino Uno R3 reads a 4 digit binary value from the Arduino Mega 2560 and plays an audio file associated with the binary value. This is the voice of the C3PO droid. Refer to the electrical drawings for hookup information.

```
//John Guarnero
//September 2014
#include <FatReader.h>
#include <SdReader.h>
#include <WaveHC.h>
#include <WaveUtil.h>

  int inp6 = 0;
  int inp7 = 0;
  int inp8 = 0;
  int inp9 = 0;

  int TC_Value = 8;
  int Last_TC = 0;
  int Demonstration_Value = 0;
  int Last_Demonstration_Value = 1;

SdReader card;     // This object holds the information for the card
FatVolume vol;    // This holds the information for the partition on the card
FatReader root;    // This holds the information for the filesystem on the card
FatReader f;        // This holds the information for the file we're play

WaveHC wave;      // This is the only wave (audio) object, since we will only play
one at a time

uint8_t dirLevel; // indent level for file/dir names
dir_t dirBuf;     // buffer for directory reads


/*
 * Define macro to put error messages in flash memory
 */
#define error(msg) error_P(PSTR(msg))

// Function definitions (we define them here, but the code is below)
void play(FatReader &dir);

// SETUP
void setup() {
  Serial.begin(9600);           // set up Serial library at 9600 bps for debugging
  putstring_nl("\nWave test!");  // say we woke up!

  putstring("Free RAM: ");       // This can help with debugging, running out of RAM
is bad
  Serial.println(FreeRam());

  //  if (!card.init(true)) { //play with 4 MHz spi if 8MHz isn't working for you
  if (!card.init()) {         //play with 8 MHz spi (default faster!)
    error("Card init. failed!");  // Something went wrong, lets print out why
  }

  // enable optimize read - some cards may timeout. Disable if you're having problems
  card.partialBlockRead(true);

  // Now we will look for a FAT partition!
  uint8_t part;
  for (part = 0; part < 5; part++) {   // we have up to 5 slots to look in
    if (vol.init(card, part))
      break;                           // we found one, so get out
  }
  if (part == 5) {                     // if we ended up not finding one  :(
    error("No valid FAT partition!");  // Say what went wrong
  }

  // Lets tell the user about what happened
  putstring("Using partition ");
  Serial.print(part, DEC);
  putstring(", type is FAT");
  Serial.println(vol.fatType(), DEC);     // FAT16 or FAT32?

  // Try to open the root directory
  if (!root.openRoot(vol)) {
    error("Can't open root dir!");      // Something went wrong,
  }

  //
  putstring_nl("Files found (* = fragmented):");

  // Print out all of the files in all the directories.
  root.ls(LS_R | LS_FLAG_FRAGMENTED);
}
```

```
// HELPERS
/*
 * print error message and halt
 */
void error_P(const char *str) {
  PgmPrint("Error: ");
  SerialPrint_P(str);
  sdErrorCheck();
  while(1);
}

void sdErrorCheck(void) {
  if (!card.errorCode()) return;
  PgmPrint("\r\nSD I/O error: ");
  Serial.print(card.errorCode(), HEX);
  PgmPrint(", ");
  Serial.println(card.errorData(), HEX);
  while(1);
}

//Start of Loop
void loop()
{

 byte i;

 inp6 = digitalRead(6); // Binary 1
 inp7 = digitalRead(7); // Binary 2
 inp8 = digitalRead(8); // Binary 4
 inp9 = digitalRead(9); // Binary 8

if (inp6 == HIGH && inp7 == LOW && inp8 == LOW && inp9 == LOW)// 1 2 4 8
{ //Happy Birthday Emily
Serial.println("1");
  TC_Value = 1;
     playcomplete("1.WAV");
   delay (1);
}
```

```
if (inp6 == LOW && inp7 == HIGH && inp8 == LOW && inp9 == LOW)// 1 2 4 8
{//Happy Birthday Kim
Serial.println("2");
  TC_Value = 2;
     playcomplete("2.WAV");
   delay (1);
}

if (inp6 == HIGH && inp7 == HIGH && inp8 == LOW && inp9 == LOW)// 1 2 4 8
{//Happy Birthday Megan
Serial.println("3");
  TC_Value = 3;
   playcomplete("3.WAV");
   delay (1);
}

if (inp6 == LOW && inp7 == LOW && inp8 == HIGH && inp9 == LOW)// 1 2 4 8
{//Welcome to the Guarnero House
Serial.println("4");
  TC_Value = 4;
   playcomplete("4.WAV");
   delay (1);
}

if (inp6 == HIGH && inp7 == LOW && inp8 == HIGH && inp9 == LOW)// 1 2 4 8
{//Happy Halloween
Serial.println("5");
  TC_Value = 5;
   playcomplete("5.WAV");
   delay (1);
}

if (inp6 == LOW && inp7 == HIGH && inp8 == HIGH && inp9 == LOW)// 1 2 4 8
{//Happy Anniversary
Serial.println("6");
  TC_Value = 6;
   playcomplete("6.WAV");
   delay (1);
}
```

```
if (inp6 == HIGH && inp7 == HIGH && inp8 == HIGH && inp9 == LOW)// 1 2 4
8
{//Say Some Jokes
Serial.println("7");
  TC_Value = 7;
   playcomplete("7.WAV");
   delay (1);
}

if (inp6 == LOW && inp7 == LOW && inp8 == LOW && inp9 == HIGH)// 1 2 4 8
{//Kim Saint dialog & Kim R2D2 dialog
Serial.println("8");
  TC_Value = 8;
   playcomplete("8.WAV");
   delay (1);
}

if (inp6 == HIGH && inp7 == LOW && inp8 == LOW && inp9 == HIGH)// 1 2 4 8
{//Happy Thanksgiving
Serial.println("9");
  TC_Value = 9;
   playcomplete("9.WAV");
   delay (1);
}

if (inp6 == LOW && inp7 == HIGH && inp8 == LOW && inp9 == HIGH)// 1 2 4 8
{//Kims Buns dialog
Serial.println("10");
  TC_Value = 10;
   playcomplete("10.WAV");
   delay (1);
}

if (inp6 == HIGH && inp7 == HIGH && inp8 == LOW && inp9 == HIGH)// 1 2 4
8
{//Master John & Lost In Space
Serial.println("11");
  TC_Value = 11;
  playcomplete("11.WAV");
   delay (1);
}
```

```
if (inp6 == LOW && inp7 == LOW && inp8 == HIGH && inp9 == HIGH)// 1 2 4 8
{//Merry Christmas
Serial.println("12");
  TC_Value = 12;
   playcomplete("12.WAV");
   delay (1);
}

Last_TC = TC_Value;
//Serial.println(Last_TC);
//

}

//  End of loop
// Plays a full file from beginning to end with no pause.
void playcomplete(char *name) {
  // call our helper to find and play this name
  playfile(name);
  while (wave.isplaying) {
  // do nothing while audio playing
  }
  // Audio is done playing
}

void playfile(char *name) {
  // see if the wave object is currently doing something
  if (wave.isplaying) {// audio already playing so stop it
    wave.stop(); // stop it
  }
  // look in the root directory and open the file
  if (!f.open(root, name)) {
    putstring("Couldn't open file "); Serial.print(name); return;
  }
  // OK read the file and turn it into a wave object
  if (!wave.create(f)) {
    putstring_nl("Not a valid WAV"); return;
  }

  // ok time to play! start playback
  wave.play();
}
```

9.3 Pololu Maestro

The Pololu Maestro controls the head, left elbow, and right elbow servos. The Pololu Maestro also controls the eye leds. To allow the eyes to fade up and down, a servo is attached to the Maestro with the motor removed and the leds are connected to where the motor leads of the servo went. The feedback pot was left in the servo circuit so when you give a servo command, the leds will brighten due to feedback error and more voltage being sent to the motor since the servo will try to get the feedback to match the command. The head servo would hum from time to time when the servo was stopped, this is due to some mechanical imbalance that I could not correct with the current setup. Therefore, I added a Pololu servo controlled relay that removes power from the servo when the head move is complete. The relay is just connected in series via a normally contact to the head servo power. This setup saves the servo from excessive wear and eliminates all noise when there is no motion. This has worked out great since I can move the head without the servo fighting me when I want to wipe things down for cleaning. Refer to the electrical drawings for hookup information.

```
<!--Pololu Maestro servo controller settings file,
http://www.pololu.com/catalog/product/1350-->
<UscSettings version="1">
  <NeverSuspend>false</NeverSuspend>
  <SerialMode>UART_FIXED_BAUD_RATE</SerialMode>
  <FixedBaudRate>9600</FixedBaudRate>
  <SerialTimeout>0</SerialTimeout>
  <EnableCrc>false</EnableCrc>
  <SerialDeviceNumber>12</SerialDeviceNumber>
  <SerialMiniSscOffset>0</SerialMiniSscOffset>
  <Channels MiniMaestroServoPeriod="80000" ServoMultiplier="1">
    <!--Period = 20 ms-->
    <!--Channel 0-->
    <Channel name="Head" mode="Servo" min="3968" max="8000"
homemode="Off" home="3968" speed="20" acceleration="0" neutral="6016"
range="1905" />
    <!--Channel 1-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off"
home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 2-->
    <Channel name="Head Switch" mode="Servo" min="3968" max="8000"
homemode="Off" home="3968" speed="256" acceleration="255" neutral="6000"
range="1905" />
    <!--Channel 3-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off"
home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 4-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="1" neutral="6000" range="1905" />
    <!--Channel 5-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 6-->
    <Channel name="Right Elbow" mode="Servo" min="3968" max="6976" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 7-->
    <Channel name="Left Elbow" mode="Servo" min="3968" max="7040" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 8-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 9-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 10-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 11-->
    <Channel name="Eyes" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="6" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 12-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 13-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 14-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 15-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 16-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
    <!--Channel 17-->
    <Channel name="" mode="Servo" min="3968" max="8000" homemode="Off" home="3968" speed="0" acceleration="0" neutral="6000" range="1905" />
  </Channels>
```

```
 <Sequences />
 <Script ScriptDone="true">#

#John Guarnero
#September 2014
#C3PO Servo Control for Firgelli cylinders,
#Hitec head servo, LED eyes, and Pololu RC Switch
#6000 center, 4000 min, 8000 max

sub Subroutine_1 #button 1
#Run the Happy Birthday Emily show
8000 11 servo #Set Eyes On
4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo
6000 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
7000 0 servo #Head Servo
4500 6 servo #Right Elbow Servo
500 delay
4500 7 servo #Left Elbow Servo
1000 delay
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
6000 0 servo #Head Servo
1000 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_2 #button 2
#Run the Happy Birthday Kim show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

6000 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
```

```
500 delay
7000 0 servo #Head Servo
4200 7      servo #Left Elbow Servo
700 delay
4500 6 servo #Right Elbow Servo
1100 delay
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
6000 0 servo #Head Servo
2000 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_3 #button 3
#Run the Happy Birthday Megan show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

6000 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
7000 0 servo #Head Servo
4200 7 servo #Left Elbow Servo
1000 delay
4500 6 servo #Right Elbow Servo
1200 delay
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
6000 0 servo #Head Servo
1500 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_4 #button 4
#Run the Guarnero House show
```

```
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

#Set Head and Arms
6000 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
700 delay
3500 0 servo #Head Servo
4200 6 servo #Right Elbow Servo
4500 7 servo #Left Elbow Servo
1000 delay
5500 0 servo #Head Servo
1100 delay
6500 0 servo #Head Servo
1100 delay
5000 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
7500 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
300 delay

5000 0 servo #Head Servo
1300 delay
6500 0 servo #Head Servo
1500 delay
7000 0 servo #Head Servo
700 delay
6000 0 servo #Head Servo
1000 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_5 #button 5
#Run the Halloween show
8000 11 servo #Set Eyes On
4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo
```

```
5100 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
100 delay
7300 0 servo #Head Servo
4200 6 servo #Right Elbow Servo
4200 7 servo #Left Elbow Servo
1000 delay
5000 0 servo #Head Servo
1000 delay
7000 0 servo #Head Servo
5000 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
6500 6 servo #Right Elbow Servo
500 delay
4500 0 servo #Head Servo
500 delay
6500 0 servo #Head Servo
1000 delay
7500 0 servo #Head Servo
300 delay
6800 0 servo #Head Servo
1500 delay
6000 0 servo #Head Servo
500 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_6 #button 6
#Run the Happy Anniversary
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

6000 0 servo #Head Servo
4200 6 servo #Right Elbow Servo
4200 7 servo #Left Elbow Servo
900 delay
7300 0 servo #Head Servo
900 delay
5000 0 servo #Head Servo
```

```
900 delay
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay
5500 0 servo #Head Servo
600 delay
6000 0 servo #Head Servo
300 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_7 #button 7
#Run the Jokes show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

5100 0 servo #Head Servo
100 delay
7300 0 servo #Head Servo
4500 6 servo #Right Elbow Servo
4500 7 servo #Left Elbow Servo
1200 delay
5000 0 servo #Head Servo
800 delay
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
1000 delay
6700 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
1000 delay
6000 0 servo #Head Servo
800 delay
7000 0 servo #Head Servo
800 delay
7500 0 servo #Head Servo
500 delay
4500 0 servo #Head Servo
900 delay
5000 0 servo #Head Servo
500 delay
```

```
6000 0 servo #Head Servo
900 delay
5100 0 servo #Head Servo
500 delay
7300 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
1200 delay
6700 0 servo #Head Servo
1200 delay
6000 0 servo #Head Servo
1000 delay
7000 0 servo #Head Servo
1000 delay
6000 0 servo #Head Servo
1000 delay
7000 0 servo #Head Servo
1000 delay
6000 0 servo #Head Servo
1000 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_8 #button 8
#Run the Kim Saint &amp; Kim R2D2 show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

5100 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
100 delay
7300 0 servo #Head Servo
4500 6 servo #Right Elbow Servo
4500 7 servo #Left Elbow Servo
1200 delay
5000 0 servo #Head Servo
800 delay
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
```

```
7000 0       servo #Head Servo
1500 delay
6000 0 servo #Head Servo
200 delay
4000 2 servo #Disable Head Servo
800 delay
4200 7 servo #Right Elbow Servo
1500 delay
6500 7 servo #Right Elbow Servo
6000 0 servo #Head Servo
4000 delay
8000 2 servo #Enable Head Servo
6000 0 servo #Head Servo
300 delay
7000 0 servo #Head Servo
4200 6 servo #Right Elbow Servo
1800 delay
6500 6 servo #Right Elbow Servo
1500 delay
6200 0 servo #Head Servo
1000 delay
7000 0 servo #Head Servo
1000 delay
6500 0 servo #Head Servo
800 delay
6000 0 servo #Head Servo
800 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_9 #button 9
#Run the Happy Thanksgiving show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

6000 0 servo #Head Servo
800 delay
7000 0 servo #Head Servo
500 delay
6000 0 servo #Head Servo
```

```
700 delay
4000 2 servo #Disable Head Servo
2000 delay
Eyes
quit


sub Subroutine_10 #button 10
#Run the Kims Buns show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
8000 2 servo #Enable Head Servo

6000 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
1000 delay
7000 0      servo #Head Servo
4200 6 servo #Right Elbow Servo
700 delay
4500 7 servo #Left Elbow Servo
1500 delay
6000 0 servo #Head Servo
5000 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
1500 delay
6700 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
700 delay
6000 0 servo #Head Servo
500 delay
4000 2 servo #Disable Head Servo
700 delay
Eyes
quit


sub Subroutine_11  #button 11
#Run the Master John &amp; Lost In Space show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
```

```
8000 2 servo #Enable Head Servo
500 delay

6000 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
500 delay

7300 0 servo #Head Servo
4200 6 servo #Right Elbow Servo
1200 delay
4000 2 servo #Disable Head Servo

4500 7 servo #Left Elbow Servo
4000 delay
8000 2 servo #Enable Head Servo
600 delay
5500 0 servo #Head Servo
1500 delay

6000 0 servo #Head Servo
5000 6 servo #Right Elbow Servo
6500 7 servo #Left Elbow Servo
2000 delay

6700 0 servo #Head Servo
6500 6 servo #Right Elbow Servo
1300 delay
6000 0 servo #Head Servo
1000 delay
6800 0 servo #Head Servo
1000 delay
6000 0 servo #Head Servo
1000 delay
4000 2 servo #Disable Head Servo
Eyes
quit


sub Subroutine_12  #button 12
#Run the Merry Christmas show
8000 11 servo #Set Eyes On

4000 2 servo #Disable Head Servo
100 delay
```

```
8000 2 servo #Enable Head Servo
200 delay
6000 0 servo #Head Servo
500 delay
6900 0 servo #Head Servo
600 delay
5500 0 servo #Head Servo
800 delay
6000 0 servo #Head Servo
800 delay
4000 2 servo #Disable Head Servo
1000 delay
Eyes
quit


sub Eyes

#Start the Eyes Fading
start1: #Pulse Eyes in a Loop
8000 11 servo
2600 delay
7450 11 servo
800 delay
goto start1
quit</Script>
</UscSettings>
```