CREATING AN OCTO-BANGER CONTROLLER, START TO FINISH

Purpose: This document will illustrate the steps required to build an OctoBanger controller from scratch. Please bear in mind that this is not a recipe, and we are not baking a cake here. If you wanted to merely copy every single step in this document, you could successfully mimic the results very easily. It is my hope; however, that people reading these instructions actually think about what they are doing, and realize that these are just suggestions and examples- not hard and fast rules. The controller in this example **may or may** not fit your prop's requirements.

Safety Disclaimer: I try to avoid switching mains voltages (120vac in the US, 220vac in most other countries) if I can help it. I do realize that bigger, more extravagant props use things like fog machines, AC motors, etc, so sometimes we must. If you are unfamiliar with electrical work of this nature, this document is absolutely no substitute for the training required for handling high voltage safely. If this is your first project, I implore you to stick with 12vdc based lights, motors, and solenoids for your props. Messy wiring and ignorance can add up to fire, death and disaster very quickly!

OK, Let's Make a Prop Controller

But First, we need to ask:

- How many channels do we really need?
- Are we going to want sound?
- Are we switching AC, DC, or both?
- How do we want to trigger the prop (Switch, PIR, etc.)?

For this fictitious/example prop, let's say we need to control:

- (2) 12vdc LED's, come on at the same time (only needs one channel)
- (2) 12vdc solenoid valves, independently switched
- (1) 12vdc LED spot light
- (1) Shiatzu massage motor

So looking at the above list, my first thought would be to use 4 mechanical relays (either in the form of a 4 relay module or shield), and a single SSR (solid state relay) for control of the AC motor.

I also plan to use sound, so I will need an MP3 module and amplified speakers of some sort.

Now that I have a general idea of what I am going to build, let's gather some parts:

1. The Arduino. This is the brain of the controller. I am going to choose an Arduino UNO for this project (actually a clone from China). They are super cheap and cost under $5 delivered. Please note that this must be a 328P based Arduino.

SPONSORED

NEW UNO R3 ATmega328P CH340 Mini USB Board for Compatible-Arduino

$4.52 From China

Was: ~~$5.65~~

Buy It Now

Free shipping

2. I need 4 relays, so I am going to use a 4 channel relay shield. This will snap right to the top of the UNO, and the relays will switch moderate DC loads no problem.

4 Channel 5V Relay Module Board Shield For PIC AVR DSP ARM MCU Arduino F5

$3.40 From Malaysia

Buy It Now

Free shipping

77 sold

See more like this

3. I want to trigger this controller with a PIR when people walk by, so I am going to buy a PIR

High Quality HC-SR501 Infrared PIR Motion Sensor Module For Arduino Raspber

$0.99 From China

Buy It Now Top Rated Plus

Free shipping

427 sold

4. I want audio, so I am going to use a Catalex serial controlled MP3 player. Note, these use a micro SD card, so if you don't have one laying around, you will need one of those too. I buy <1GB cards used & in bulk, so that makes them much cheaper. You don't need a big honking card for just a small MP3 file or two.

YX5300 UART Control Serial MP3 Music Player Module For Arduino/AVR/ARM/PIC GM

$3.41 From China

Buy It Now

Free shipping

180 sold

5. These jumper wires are handy for connecting the MP3 module and the PIR. In this example I am using the kind with **male to female** ends.

1x/40Roots 20cm Male to Female Dupont Cable Jumper Wire for Arduino Breadboard

$0.99 From Hong Kong

Buy It Now

Free shipping

46 sold

6. Although not necessary, this type of cable makes a nice extended PIR cable.

10 Pcs 3 Pin Male to Female RC Servo Extension Cord Cable 50cm Length HP

$2.89 From Hong Kong

Buy It Now

Free shipping

See more like this

7. To Switch the AC motor, I plan on using an SSR like this one. They can switch heavier/noisier AC loads much more safely and reliably than a mechanical relay can.

SSR-25 DA 25A DC/AC 24V-380V Solid State Relay for PID Temperature Controller

$2.89 From China

Buy It Now

Free shipping

82 sold

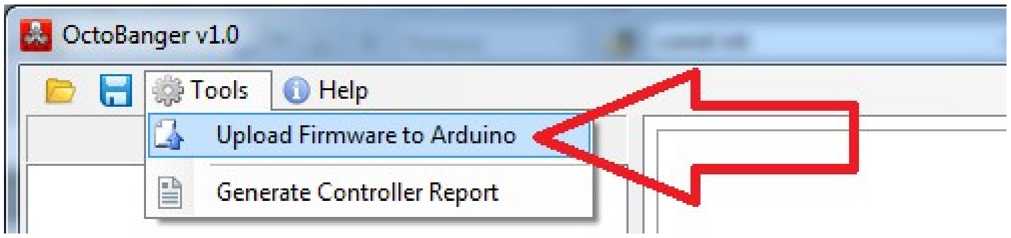
Other misc supplies I will use:

- an AC extension cord
- computer speakers from the thrift store
- USB cable
- regulated 12vdc adapter capable of supplying 1 amp for the Arduino, plus whatever current is going to be required by the loads you are running. If you want to use a separate power supply to run your DC loads, great- that is up to you and will vary on a case by case basis. For that reason, I am not going to beat to death the topic of power here, and concentrate on the controller itself.

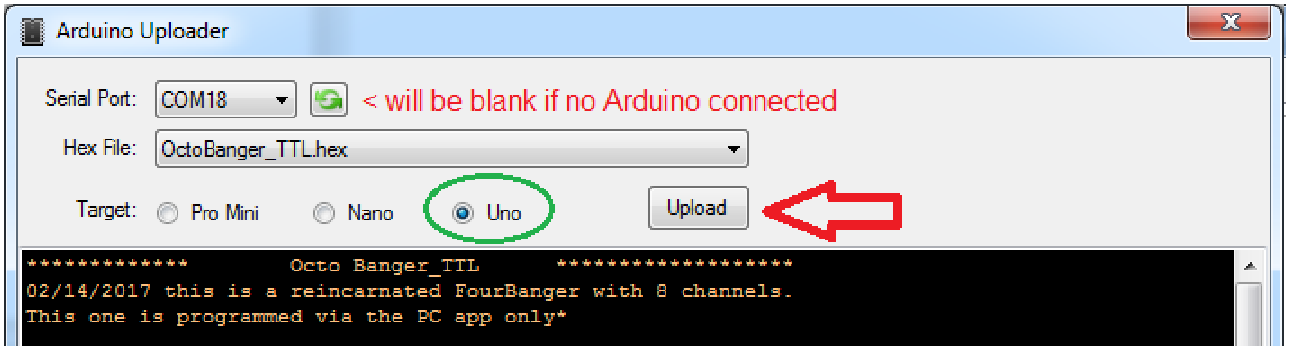
Step 1: Upload firmware to the Arduino.

Before performing any wiring on our Arduino, we should first upload the firmware.

- Connect BARE Arduino (no wires connected yet or shields installed) into the PC via USB
- Launch the OctoBanger PC app. Once in the main screen, open the firmware upload utility from the Tools menu:



- You should be able to leave all the settings as-is and just click the Upload button



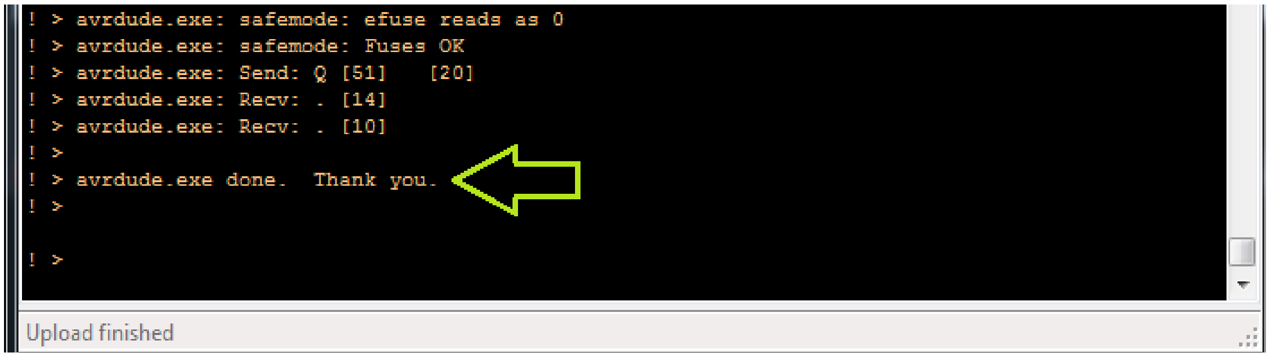
Serial Port: COM18 < will be blank if no Arduino connected

Hex File: OctoBanger_TTL.hex

Target: Pro Mini Nano [✓ Uno]

```
**************          Octo Banger_TTL       ********************
02/14/2017 this is a reincarnated FourBanger with 8 channels.
This one is programmed via the PC app only*
```

- The upload takes about 5 seconds. Once complete, a successful result will end with a message like this:

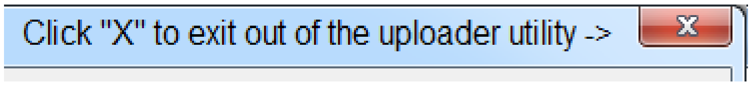


```
! > avrdude.exe: safemode: efuse reads as 0
! > avrdude.exe: safemode: Fuses OK
! > avrdude.exe: Send: Q [51]   [20]
! > avrdude.exe: Recv: . [14]
! > avrdude.exe: Recv: . [10]
! >
! > avrdude.exe done.  Thank you.
! >

! >
```

Upload finished

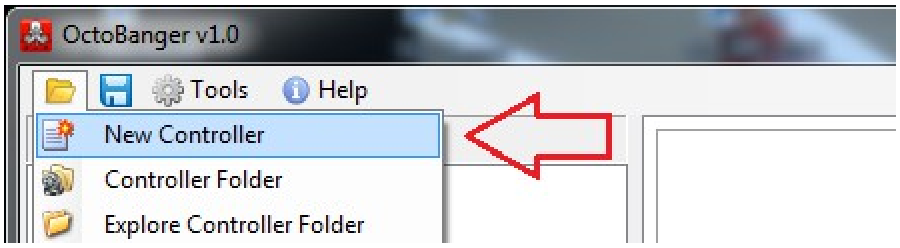
That is it for the firmware upload.



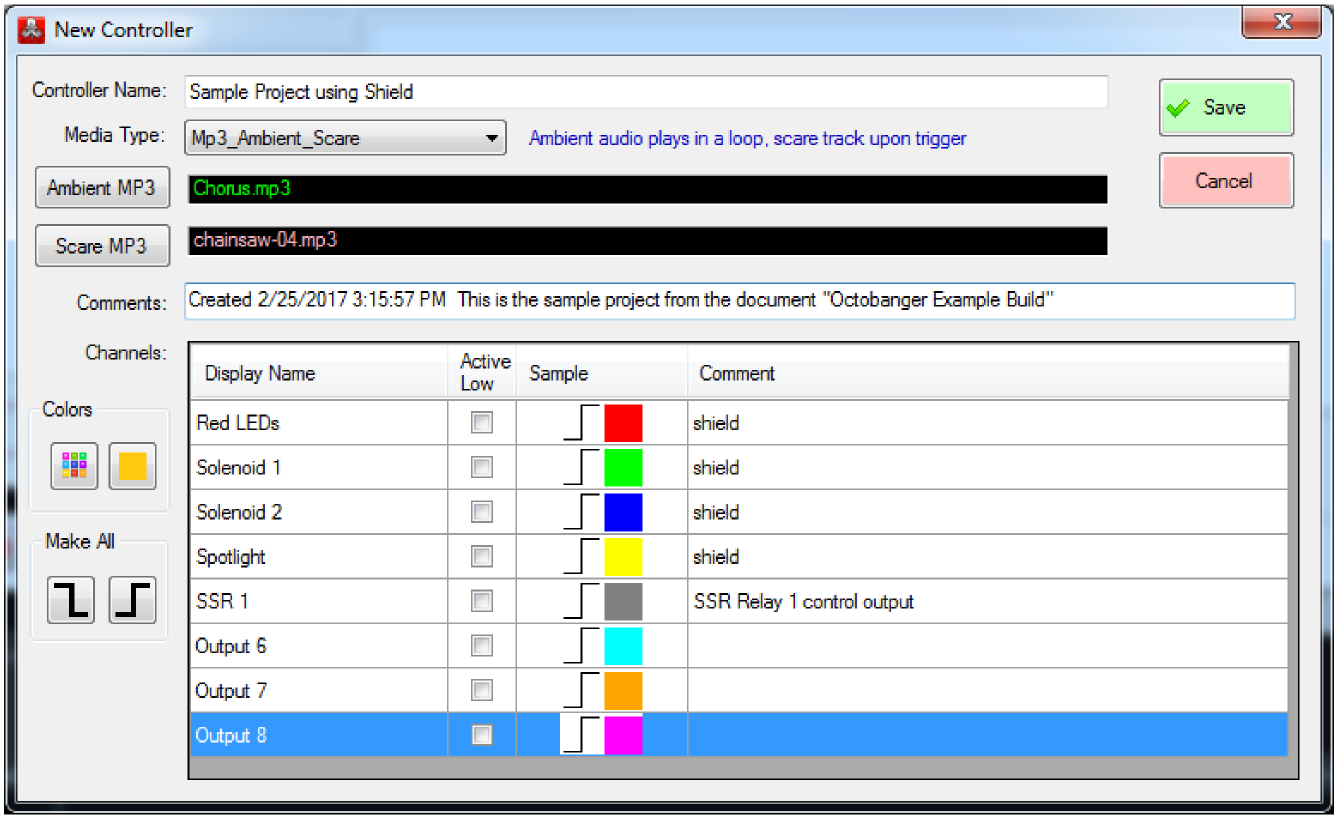
Click "X" to exit out of the uploader utility ->

Step 2: Create a new controller project.

From the File menu, select “New Controller”:



Here is how I filled out the options for this sample project:



Controller Name: Sample Project using Shield

Media Type: Mp3_Ambient_Scare — Ambient audio plays in a loop, scare track upon trigger

Ambient MP3: Chorus.mp3

Scare MP3: chainsaw-04.mp3

Comments: Created 2/25/2017 3:15:57 PM This is the sample project from the document "Octobanger Example Build"

Channels:

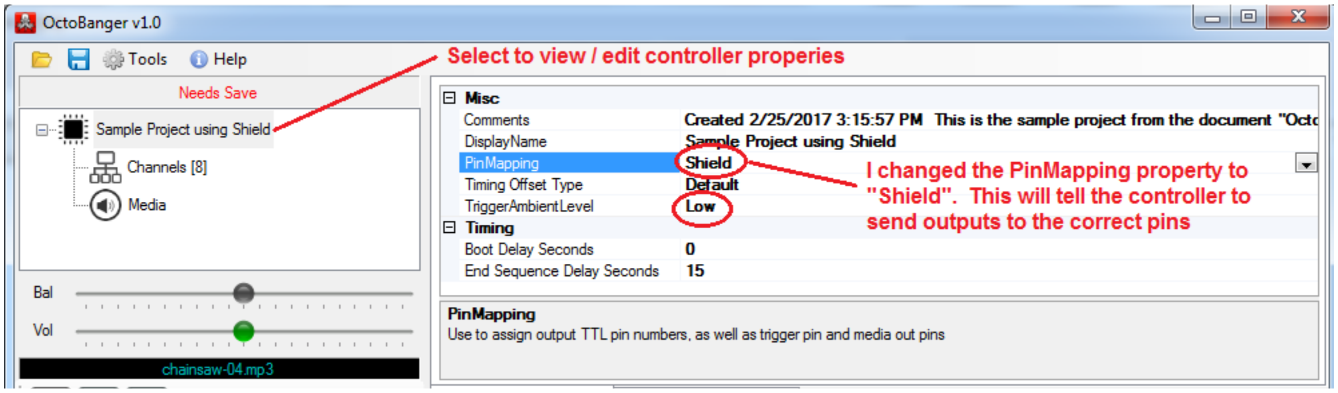
Display Name	Active Low	Sample	Comment
Red LEDs	☐	🟥	shield
Solenoid 1	☐	🟩	shield
Solenoid 2	☐	🟦	shield
Spotlight	☐	🟨	shield
SSR 1	☐	⬛	SSR Relay 1 control output
Output 6	☐	🟦	
Output 7	☐	🟧	
Output 8	☐	🟪	

I updated the info on each of the channels I plan to use.

I also changed the Media Type to “Mp3_Ambient_Scare”, because I plan to use an ambient track as well as a scare track. I clicked the 2 MP3 buttons and browsed to the location on my PC where they were stored. If you don't have audio files in mind at this point, you can just skip this part and spoint to your audio files later.

Finally, I clicked the green “Save” button in the upper right.

Since this project is using a relay shield, I am going to first set the PinMapping property to Shield:



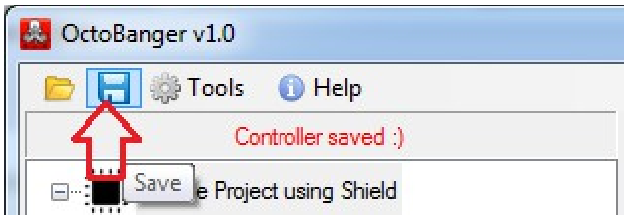
Misc
Comments: Created 2/25/2017 3:15:57 PM This is the sample project from the document "Octo
DisplayName: Sample Project using Shield
PinMapping: Shield
Timing Offset Type: Default
TriggerAmbientLevel: Low
Timing
Boot Delay Seconds: 0
End Sequence Delay Seconds: 15

I changed the PinMapping property to "Shield". This will tell the controller to send outputs to the correct pins

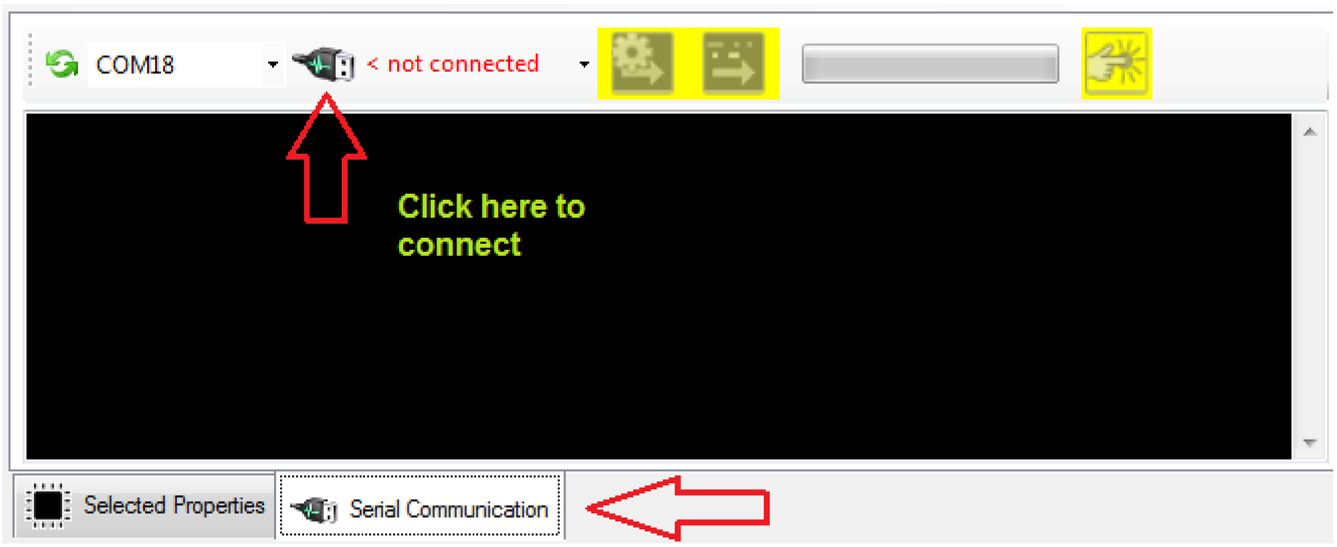
PinMapping
Use to assign output TTL pin numbers, as well as trigger pin and media out pins

Since I also plan to use a PIR, I will ensure that the TriggerAmbientLevel property is set to Low. This is because the PIR's we typically use normally output a LOW TTL level when inactive, then go TTL HIGH when they "see someone". There will be further explanation below when we wire the PIR.

With those 2 important properties now set, I will click the save button to save things:

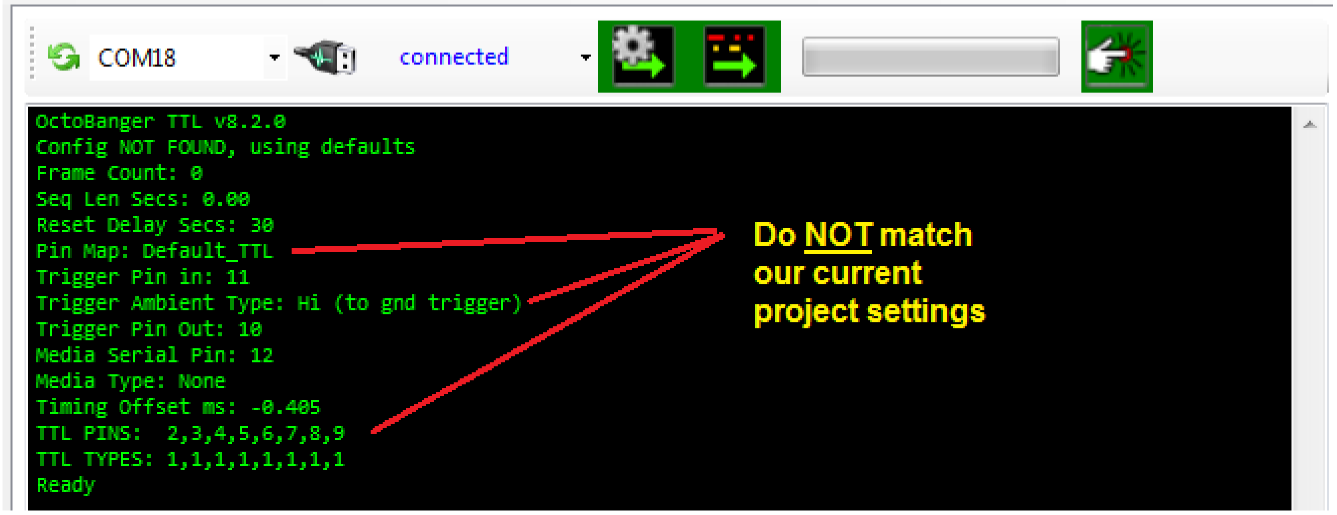


Then I will flip over to the Serial Communication tab:



Click the USB icon to connect to the Arduino.

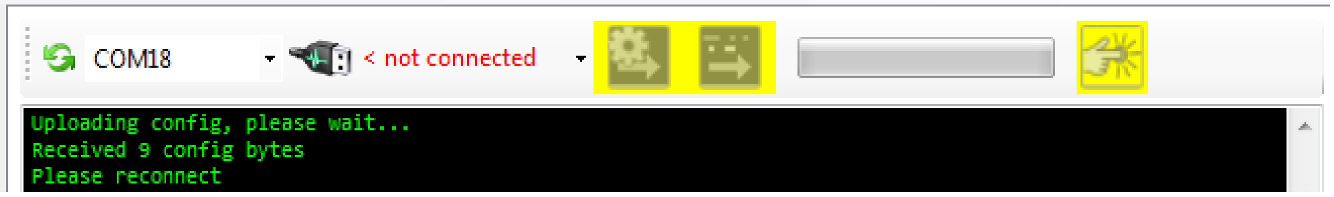
This is what a typical, out-of-the-box message will look like once a new OctoBanger is connected to:



```
OctoBanger TTL v8.2.0
Config NOT FOUND, using defaults
Frame Count: 0
Seq Len Secs: 0.00
Reset Delay Secs: 30
Pin Map: Default_TTL
Trigger Pin in: 11
Trigger Ambient Type: Hi (to gnd trigger)
Trigger Pin Out: 10
Media Serial Pin: 12
Media Type: None
Timing Offset ms: -0.405
TTL PINS:  2,3,4,5,6,7,8,9
TTL TYPES: 1,1,1,1,1,1,1,1
Ready
```

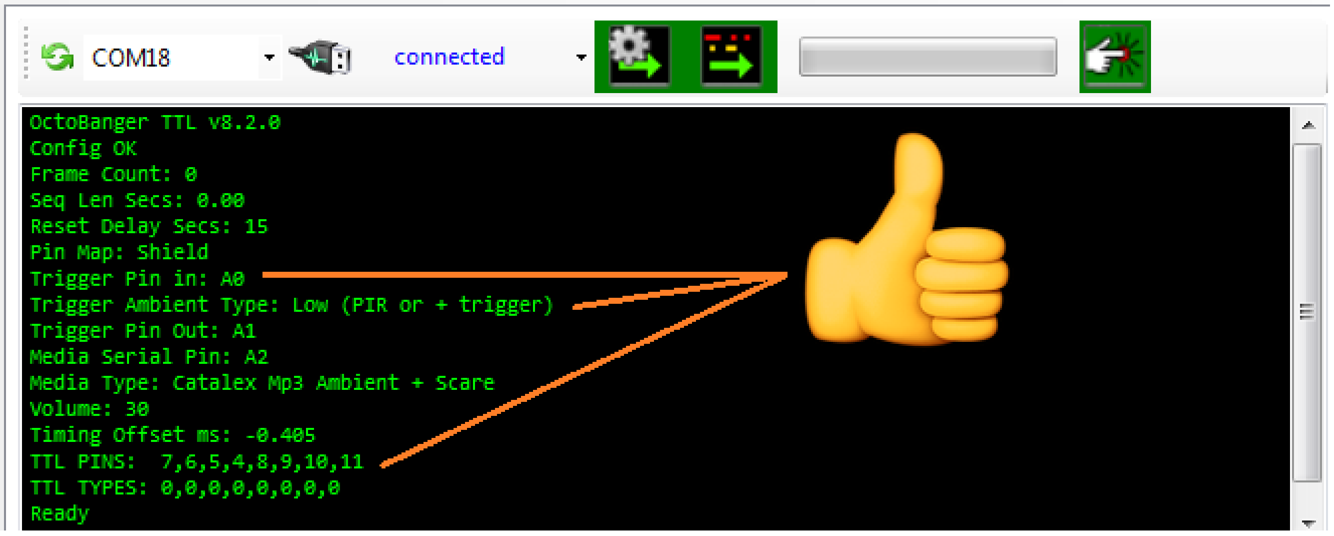
We will click this icon to upload our controller project's settings to the Arduino:

The config upload automatically disconnects us. The reason for this is to essentially "reboot" the Arduino. Since we may be changing pin numbers and their usages, this is a requirement.



```
Uploading config, please wait...
Received 9 config bytes
Please reconnect
```

After clicking the USB connect icon again, we see that the controller reports back our desired settings:

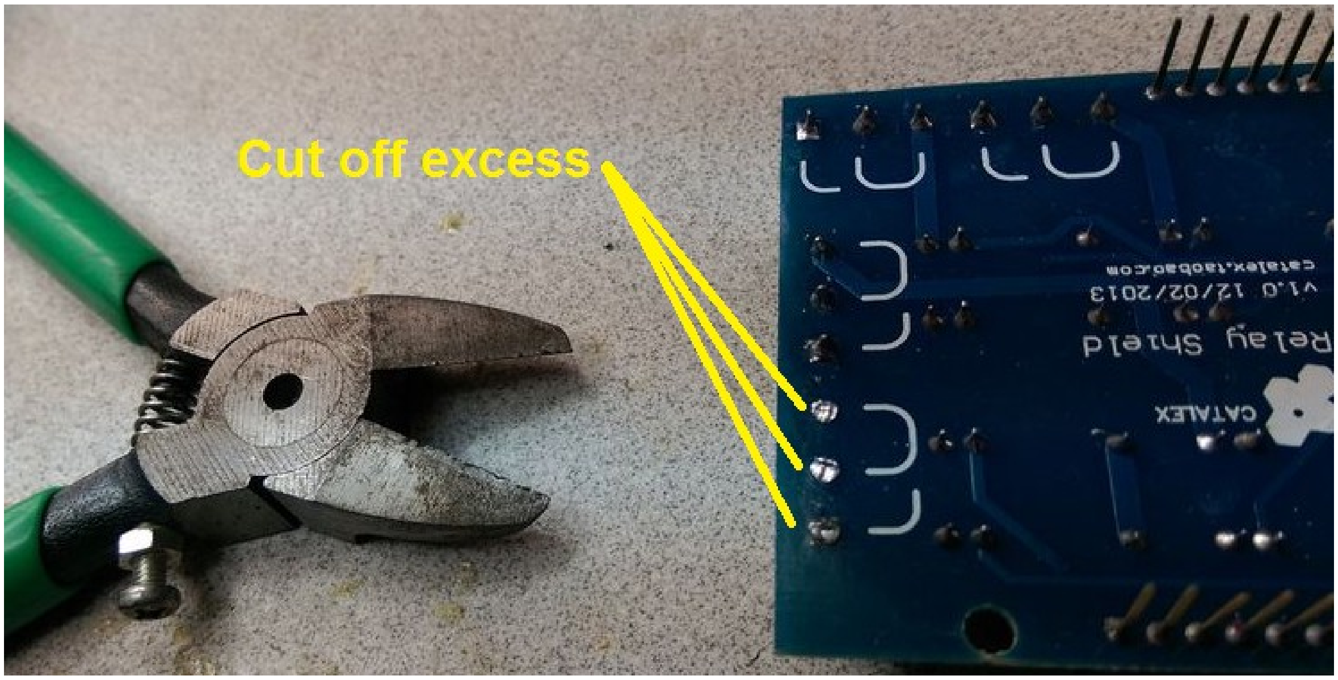


```
OctoBanger TTL v8.2.0
Config OK
Frame Count: 0
Seq Len Secs: 0.00
Reset Delay Secs: 15
Pin Map: Shield
Trigger Pin in: A0
Trigger Ambient Type: Low (PIR or + trigger)
Trigger Pin Out: A1
Media Serial Pin: A2
Media Type: Catalex Mp3 Ambient + Scare
Volume: 30
Timing Offset ms: -0.405
TTL PINS:  7,6,5,4,8,9,10,11
TTL TYPES: 0,0,0,0,0,0,0,0
Ready
```

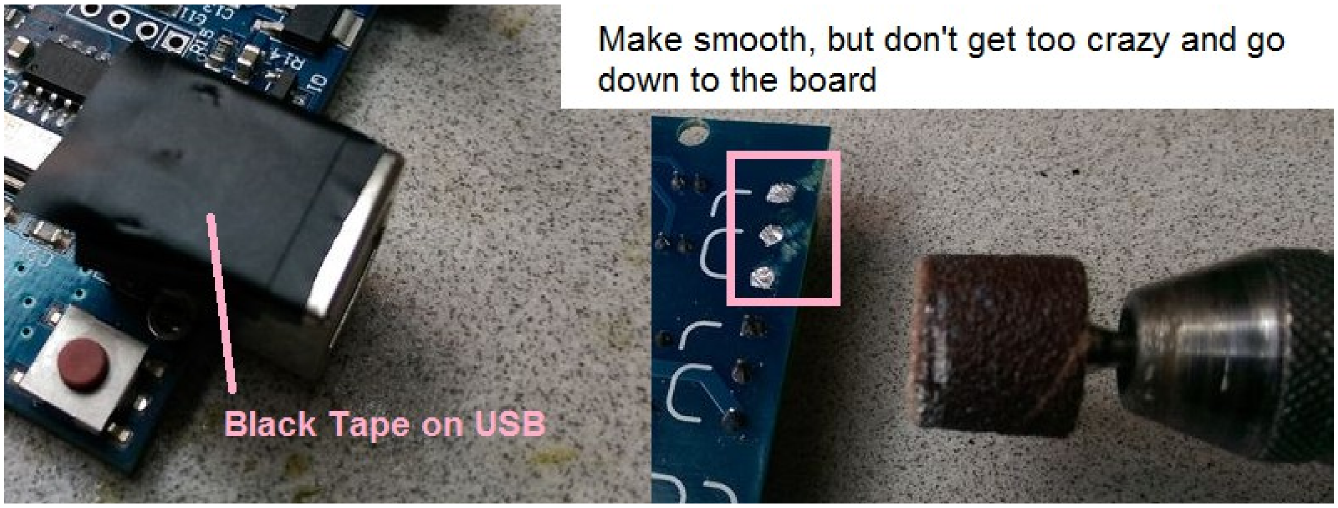
Click the USB icon to disconnect, then PHYSICALLY DISCONNECT the Arduino.
We can also make note of the Trigger Pin in (A0) and Media Serial Pin (A2) for use in the next step.

Step 3: Connect our shield and some wiring.

Now that the firmware is uploaded and has been configured for the proper pinout, we can start wiring the controller up. First off, I had to address a FLAW in the shield I was using. You need to trim the pins on the #1 relay so they will not **SHORT OUT** on the Arduino USB block! Important!!!



I sanded mine down as well, and added a few layers of black tape to the USB connector.


Make smooth, but don't get too crazy and go down to the board
Black Tape on USB

That should keep the relay pins and the USB / Ground isolated. This is another reason why I just stick to 12v switching with these things and use an SSR when I want to switch 120vac.

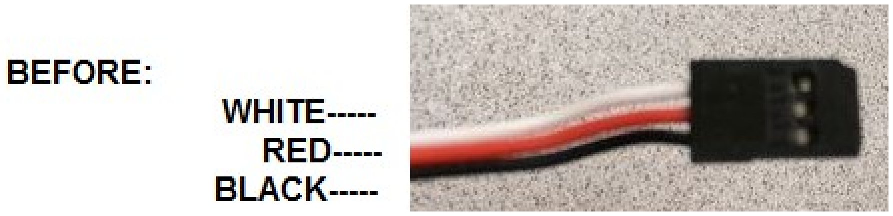
If you feel that this modification is more trouble than it is worth, I certainly don't blame you! Remember that you could easily substitute a typical four-channel relay module (like used with the fourbanger project) in place of the relay shield.

Wiring, continued...

I am going to “hardwire” i.e. solder my PIR, MP3, and SSR relay wires onto the Arduino. You don't have to do this, but it is how I like to do mine.. If you want to just use Dupont jumper wires on yours, go ahead- this is **how I make mine**. I solder the connections just to make them extra stable and permanent.

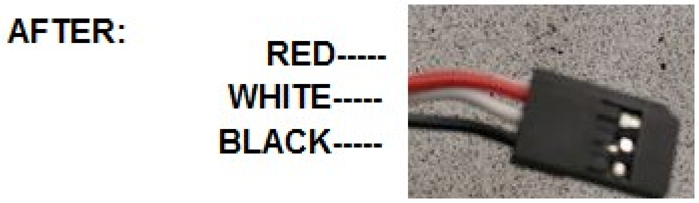
I will start with my PIR connector. I am using an RC servo cable for a little extra length. The red will be 5v, black will be GND, and signal is going to be A0. Before using this cable, I need to swap pins to get them in the correct order for my PIR:

BEFORE:

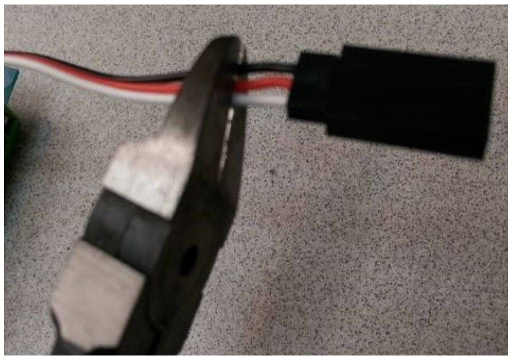


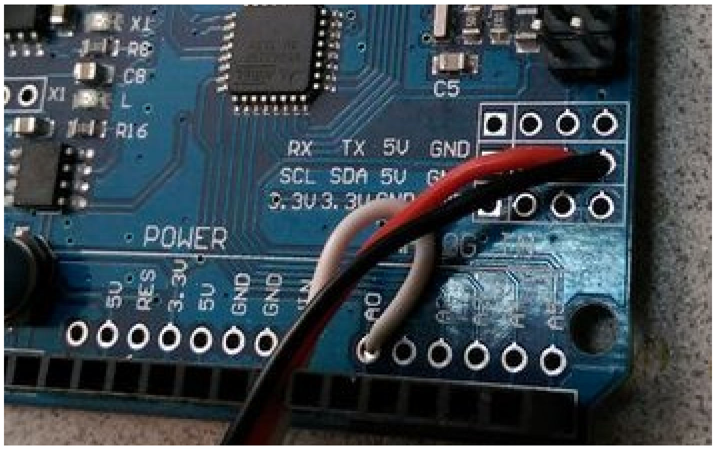
I used an X-Acto knife to free the pins by pulling back the holding tabs & slid them out

AFTER:



I cut off the other end of the cable, stripped the wires, and soldered them into the board like this:

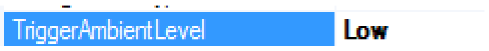


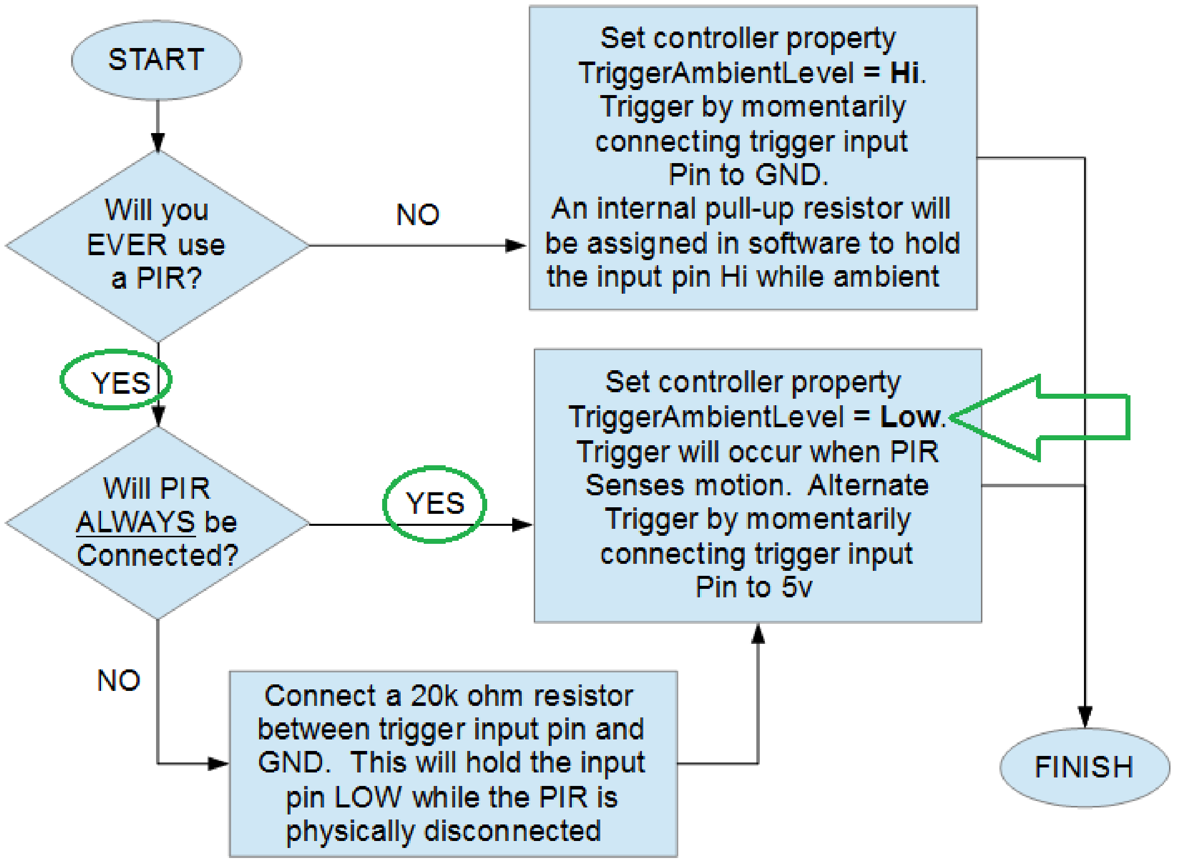


One of the things I really like about the Chinese clone UNO boards is the convenient 5v and GND solder terminals located by the analog pins.

TRIGGERING LOGIC

Below is a decision chart that shows the factors taken into consideration when we set this property in the PC application:


TriggerAmbientLevel Low



START
Will you EVER use a PIR?
NO
Set controller property TriggerAmbientLevel = **Hi**. Trigger by momentarily connecting trigger input Pin to GND. An internal pull-up resistor will be assigned in software to hold the input pin Hi while ambient
YES
Will PIR ALWAYS be Connected?
YES
Set controller property TriggerAmbientLevel = **Low**. Trigger will occur when PIR Senses motion. Alternate Trigger by momentarily connecting trigger input Pin to 5v
NO
Connect a 20k ohm resistor between trigger input pin and GND. This will hold the input pin LOW while the PIR is physically disconnected
FINISH

With this controller, if I ever decide that I want to use a button to trigger instead of the PIR, I will merely cover the PIR up rather than physically disconnect it. I would then be able to trigger the controller by momentarily connecting the trigger input pin (A0 in this case) to 5v.

If the PIR were merely disconnected, what would happen is that the trigger input pin would be left “floating”. This is an undefined logical condition that is neither hi or low, and unpredictable behavior.

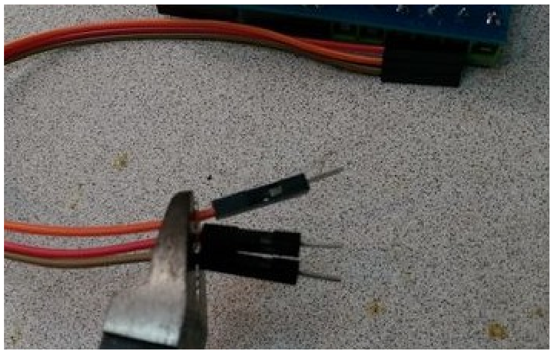
With a physically disconnected PIR, you have 2 choices:

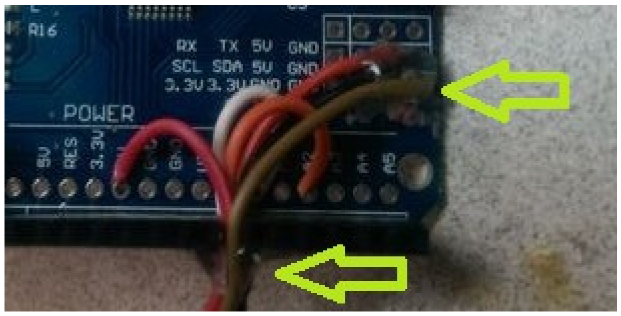
1) Change the TriggerAmbientLevel property to Hi. Then trigger with a momentary switch to GND.
2) Connect 20k ohm resistor from trigger input to GND. trigger with a momentary switch to 5v.

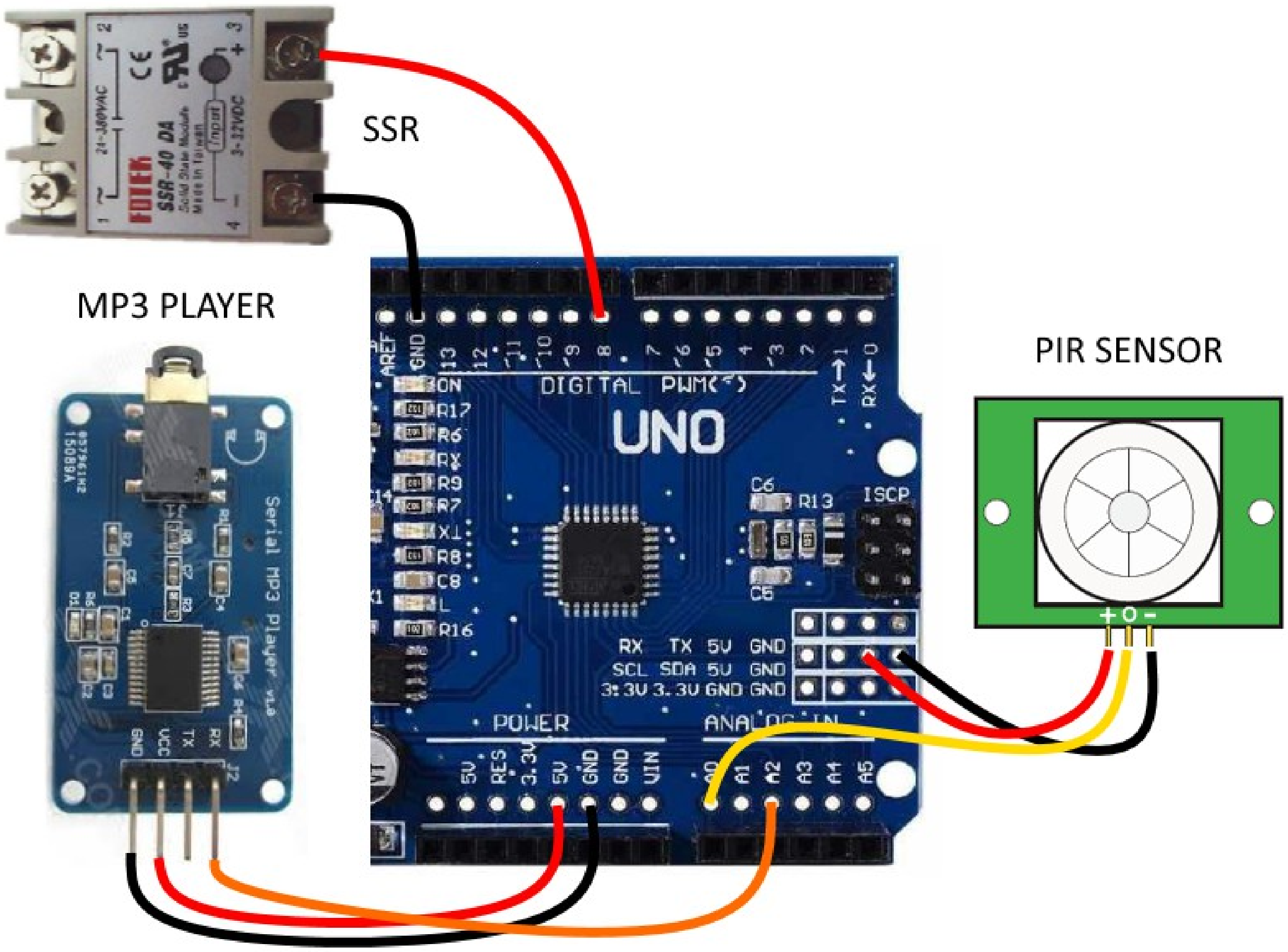
If your controller starts triggering upon power up, your settings may be incorrect. Follow the diagram above and correct your settings and wiring accordingly. Also, please remember that you may configure the controller to run without a PIR while building, then add one & reconfigure when you are finished building your prop. I often do this, since PIR's are sensitive and can be a pain to trigger while testing. Finally, please remember that you can trigger the controller from the PC app with one click!

Wiring, continued...

Next I wired up 3 wires for the Catalex serial MP3 player. For these I just used regular Dupont jumper wire, and cut one end off (leaving the female end). I stripped them and soldered the red to 5v, brown to GND, and the orange to A2. I then put a couple gobs of hot glue on there to keep things in place.



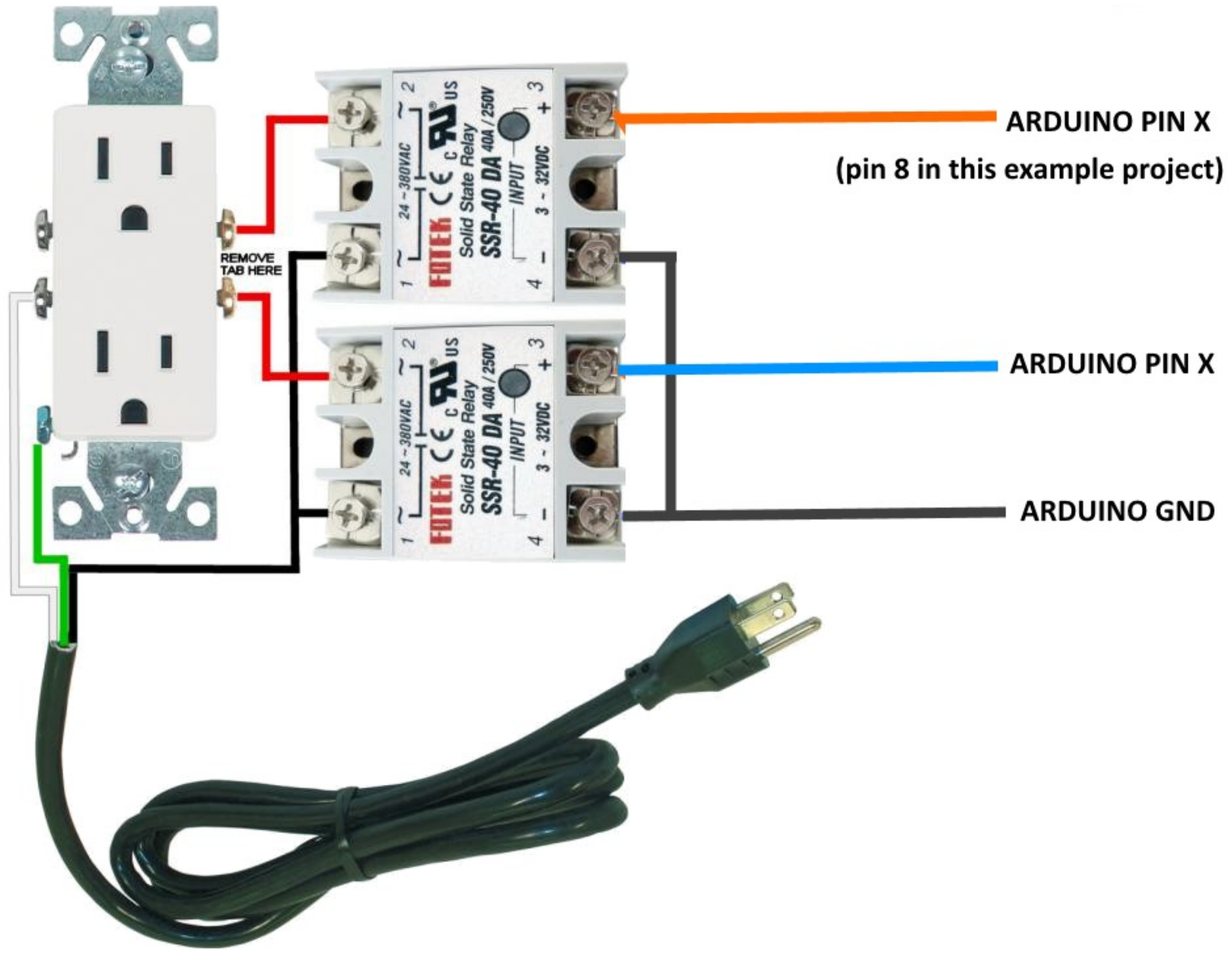




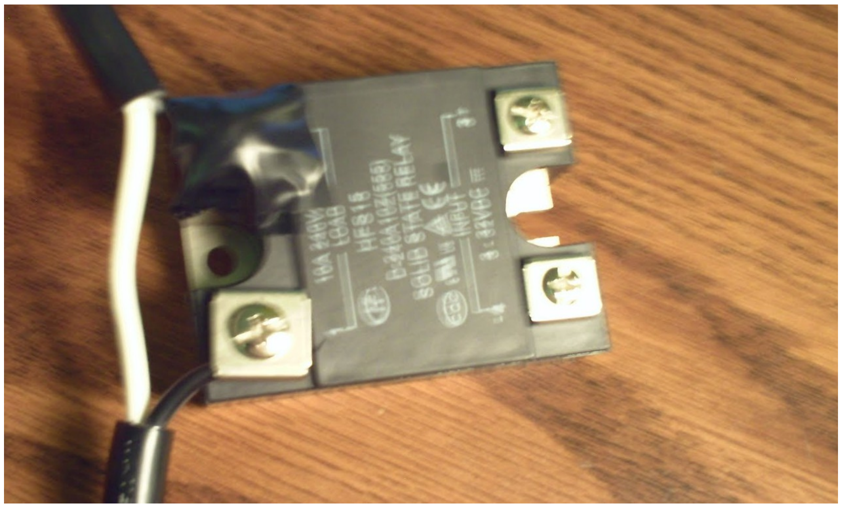
This is the essential wiring diagram (not including the wires that will be hooked up to the shield relays).

Although this document is more about the controller and less about the fictitious items I may or may not be switching on and off with it, I thought I would take a few minutes to go out to the web and heist someone 's great diagram.

I just used one SSR in this project, but if I were to use two, I would certainly hook them up as shown below.


REMOVE TAB HERE
FOTEK Solid State Relay SSR-40 DA 40A / 250V
24 ~ 380VAC
INPUT 3 - 32VDC
FOTEK Solid State Relay SSR-40 DA 40A / 250V
24 ~ 380VAC
INPUT 3 - 32VDC
ARDUINO PIN X
(pin 8 in this example project)
ARDUINO PIN X
ARDUINO GND

For a single SSR, a very simple method to switch power is to use a cut extension cord like this:

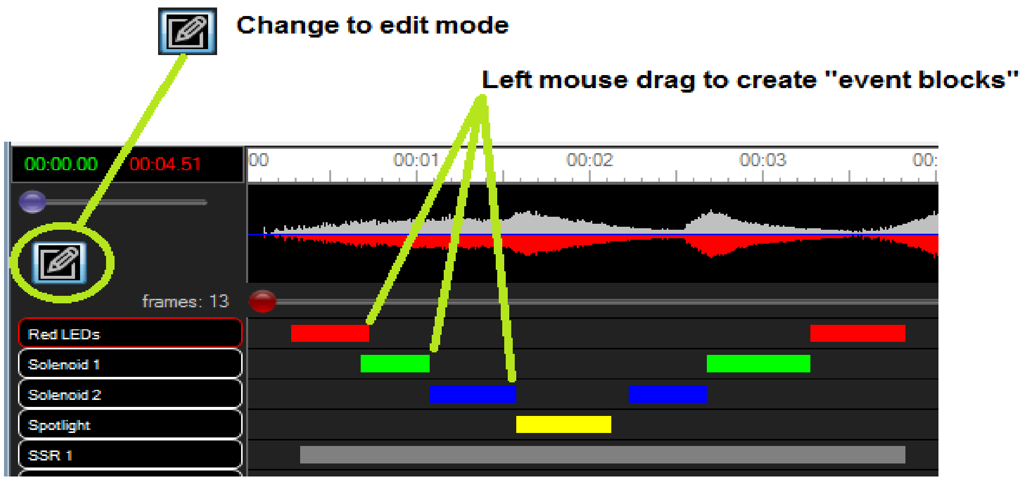


Make sure to switch the BLACK/HOT wire, and cover those live contacts with black tape.

Step 4: Upload a Test Sequence

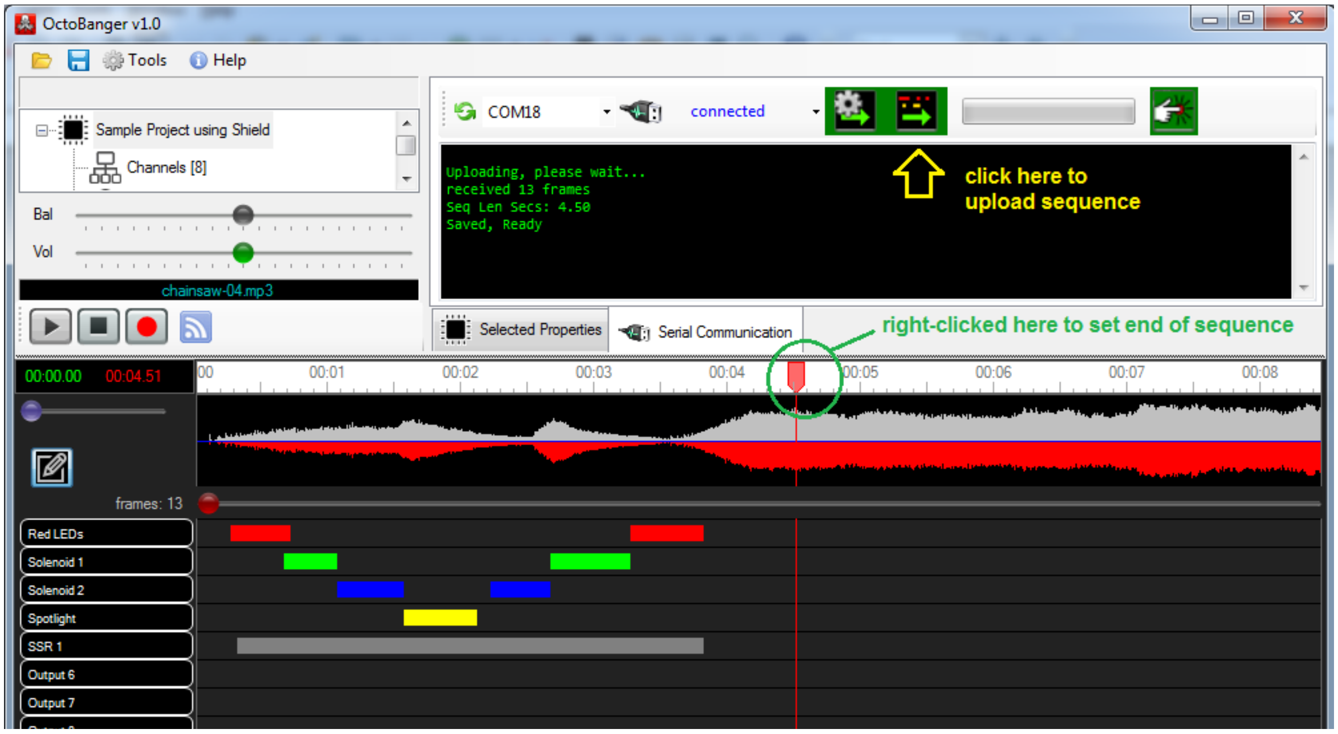
Connect the Arduino USB to the PC and open the OctoBanger application. I will now make a few test events on the timeline in order to test the controller.

I created a quick set of events on the timeline in order to test the channels I was using:



I right-clicked on the timeline at about 04:30 to set the end of the sequence. The ultimate sequence for a real prop will likely be longer- I sometimes like to make a simple little test sequence like this at first just to ensure that all my relays are wired up correctly. Once I run a quick test to make sure that all channels are functional, I will go back and tweak the event data to be "just right".

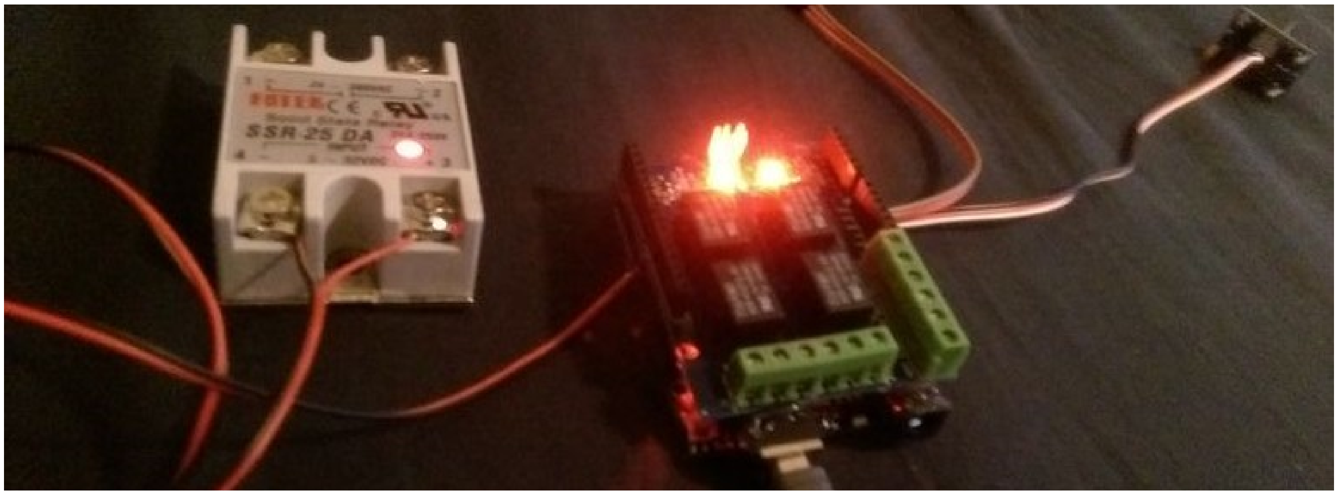
I then clicked this button to upload the sequence data to the controller:

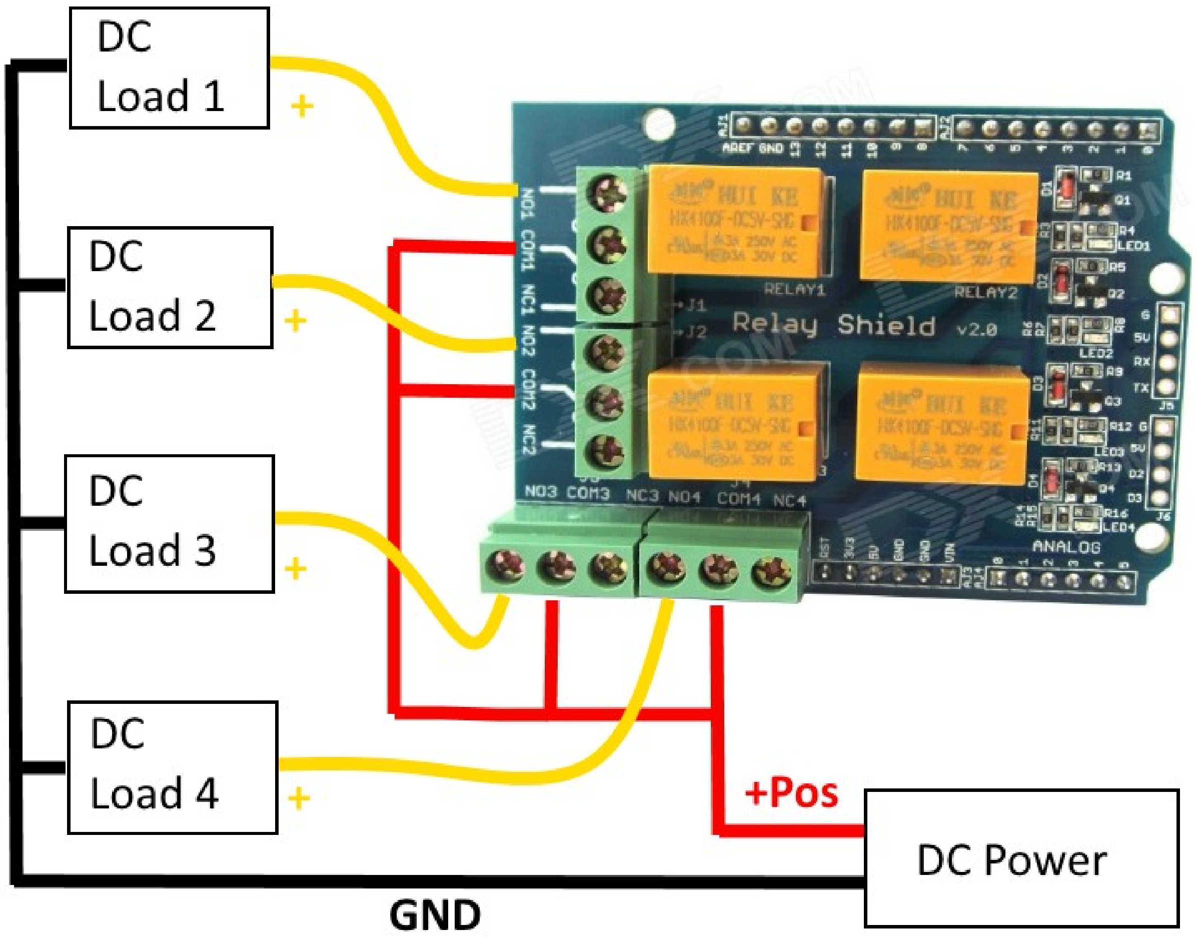


```
Uploading, please wait...
received 13 frames
Seq Len Secs: 4.50
Saved, Ready
```

From the PC application, you can use the trigger button to trigger the sequence:

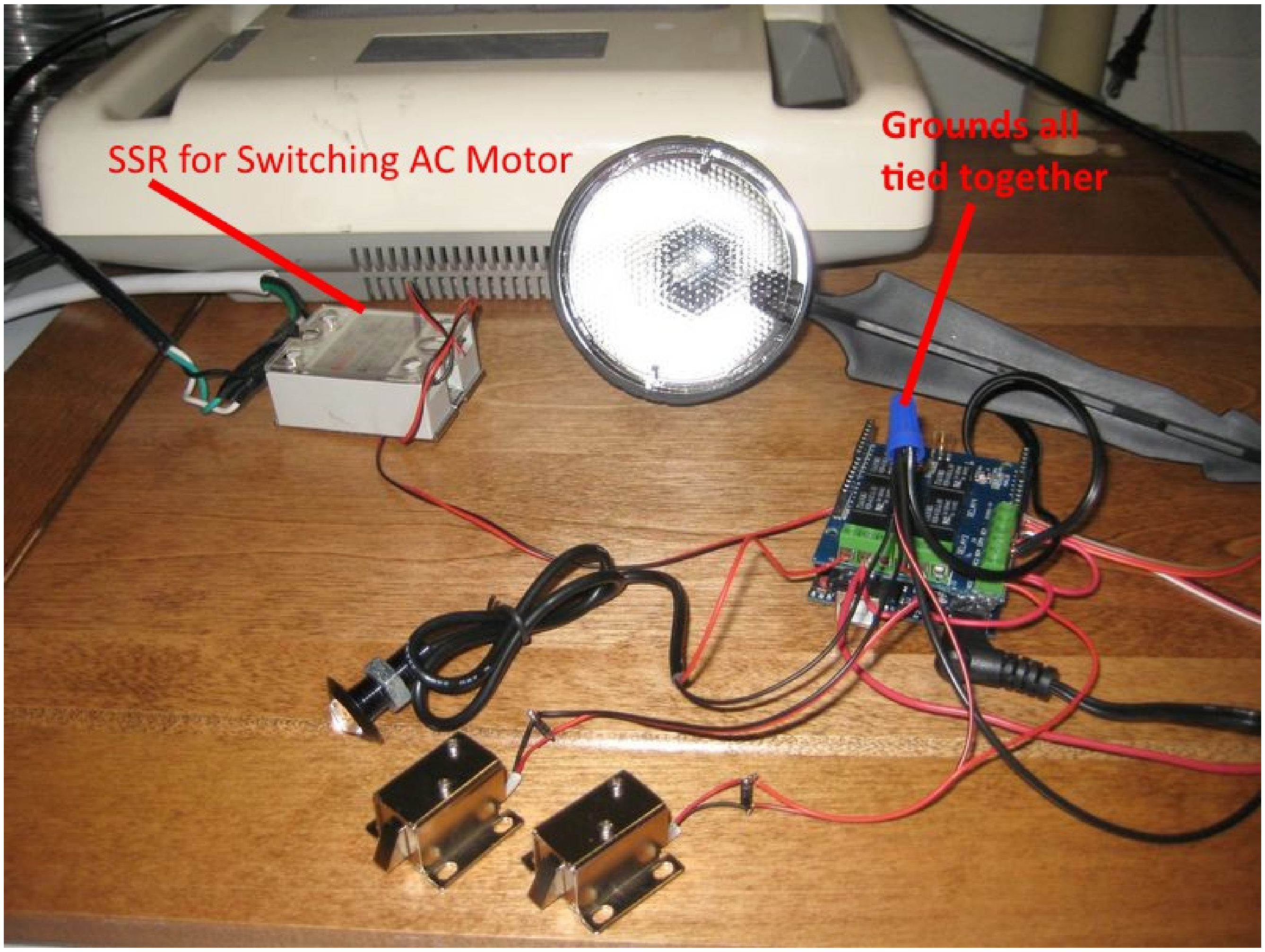
Test 1, lights flashing and relays clicking. A nice thing about the SSR I am using is that it has a red display LED that turns on to let you know the relay is active.





This diagram shows how DC loads can be wired, as long as all of the loads share the same voltage rating.

I was reminded that it is always a good idea to switch the positive side of DC voltages and to tie all the grounds together. While it will work if you want to switch the grounds, it is a better habit to switch the positives.


SSR for Switching AC Motor
Grounds all tied together