Art-Net Controlled Winch

by WoutDemeyere

Hello everybody, in this instructable i'll be explaining how i created my art-net controlled winch. "Your what?" i hear you ask, well let me explain very quickly. A few years ago we threw a party with the local youth house, and as stage design we had the idea of making a moving roof.

With 9 winches (takels) we lifted up a giant white cloth and through the protocol dmx we controlled them. But at the time the winches we used where 3phase controlled. So to make it go up and down we had to create a system of relais controlled by a central Arduino mega that was controlled using dmx. The relais where switching voltages of 230V and 12 amps.

To put it in more understandable terms we had to create a giant mess of wires and relais that was super unstable, unreliable and pretty dangerous actually.

So we thought to ourselves there has to be a better way to do this. We started thinking and came to the conclusion that the best way to do this would be through a modular system of winches, controlled over the Art-Net protocol so we would only need a voltage source and an ethernet cable.

So thats exactly what i created and i will try to explain to you how i did it. The whole setup relies on a raspberry pi that controls the setup process. Ones its setup the control is giving over to a light computer (Chamsys, etc..) that will control the winches from there on.

I am asuming that you have basic knowledge of python, arduino and raspberry pi aswel as understanding something about the art-net protocol and controllers.

Supplies:

What you'll need:

- Raspberry pi
- Arduino
- Arduino Ethernet shield
- Arduino motor shield
- Any OLED screen
- ethernet cable
- ethernet switch
- DC Motor with encoder build on

Step 1: Setting Up the Raspberry Pi

Ok so the way this will be working is the raspberry pi will be running an Apache and a MariaDB server. The apache server is to host the website, the MariaDB is to keep a database where we will store the data of the winches.

I won't walk you through the entire processes of setting up the pi with ssh, if you're unfamiliar here's a good tutorial.

So first of we'll make sure the raspberry pi is all setup, in your terminal run:

To install apache

```
sudo apt install apache2 -y
```

To install mariaDB

```
sudo apt-get install mariadb-server
```

These are all the python extensions we'll need

```
pip3 install mysql-connector-python
pip3 install flask-socketio
pip3 install flask-cors
pip3 install gevent
pip3 install gevent-websocket
pip3 install netifaces
```

For the oled display we need a bit more difficult setup process that can be found here.

Ok that's the pi all done!

Step 2: Understanding How the Setup Will Work

So how you want to setup the winch is by giving it a start position and an end position. It will be assigned a certain channel and with this channel you'll only be able to move between the chosen stat and end position.

To select these position you will need to move the winch to them, once you're there you will turn a certain channel to the value of 56. When this exact channel reaches that value it will know that this is his end start / end position, if it needs to move up or down or if it needs to change its artnet values. Moving the winch also is done by setting a certain channel to 56. "And why 56" i hear you wonder, well i had to chose something :).

The position is calculated through the encoder thats on the DC motor.

Step 3: Backend of the Pi

The backend for the system can be found on my github. I've written my own library for the art-net protocol so feel free to use it. I wont walk you through everything line by line but i will give you the big picture idee of it all.

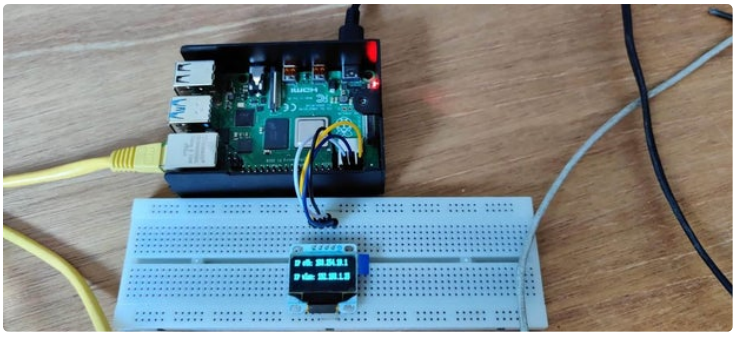
The code runs a flask server which communicates with the apache server running on the pi. It uses the flask-socketio module to send & receive data to the frontend. The art-net lib uses the socket module from python to send UDP packets to and from the arduino.

Every method that begins with a @socketio.on('F2B_***') is waiting for an F2B call from the frontend. Once it's received it will execute the according action. Be that sending an artnet commando or getting data from the DB and sending it back to the front.

The oled_show_info() method is used to display the ip of the ip (Both on the wlan and ethernet interfaces).

so just run the code with

```
python3 app.py
```



Step 4: Frontend of the Pi

To be able to edit the frontend of the pi you'll first need to give yourself acces to the directory /var/www/html/. This is where apache gets it's files to display on its website. To get acces execute:

```
sudo chmod 777 /var/www/html/
```

Now thats all done get the code for the frontend of my github and place it in the /var/www/html/ directory.

The frontend uses the same principle as the backend but now it sends the F2B_*** commands when a button is pressed or slider is moved.

And that's the frontend done!

Step 5: The Arduino

For using the arduino you'll need to use the ethernet shield and the motor shield. Just gently push them onto the arduino. Make sure you don't push the motor shield to far onto the ethernet shield or you'll short the 2 motor pins on the ethernet connection!

The code for the arduino can also be found on my github. Upload the artnet_winch.ino file and all should be good.

Make sure you define the right pins to the correct pins of your motor. The motorshield pins are chosen with the header pins on top of the shield. These pins are chosen under the //--- motor config. Aswel as the encoder pins that need to be connected to the right pins of the arduino.

Also make sure your MAC address of the shield is correct. This can be found on a sticker underneath the shield and edited in the MAC variable. The IP you use for the pi needs to be in the same range as the PI, this can normally be found on the oled screen.

The code looks a bit much but is not that hard to understand. Basically the ethernet shield constantly reads udp packets coming in. If that packet is an Artnet packet it decodes it and gets the info it needs from it. The different artnet packets are nicely described on the Art-Net website, so if your interested you can read what they all do.

If it receives an ArtPoll packet it will response with an ArtPollReply. this is used in the callout function in the backend to find which devices are on the network.

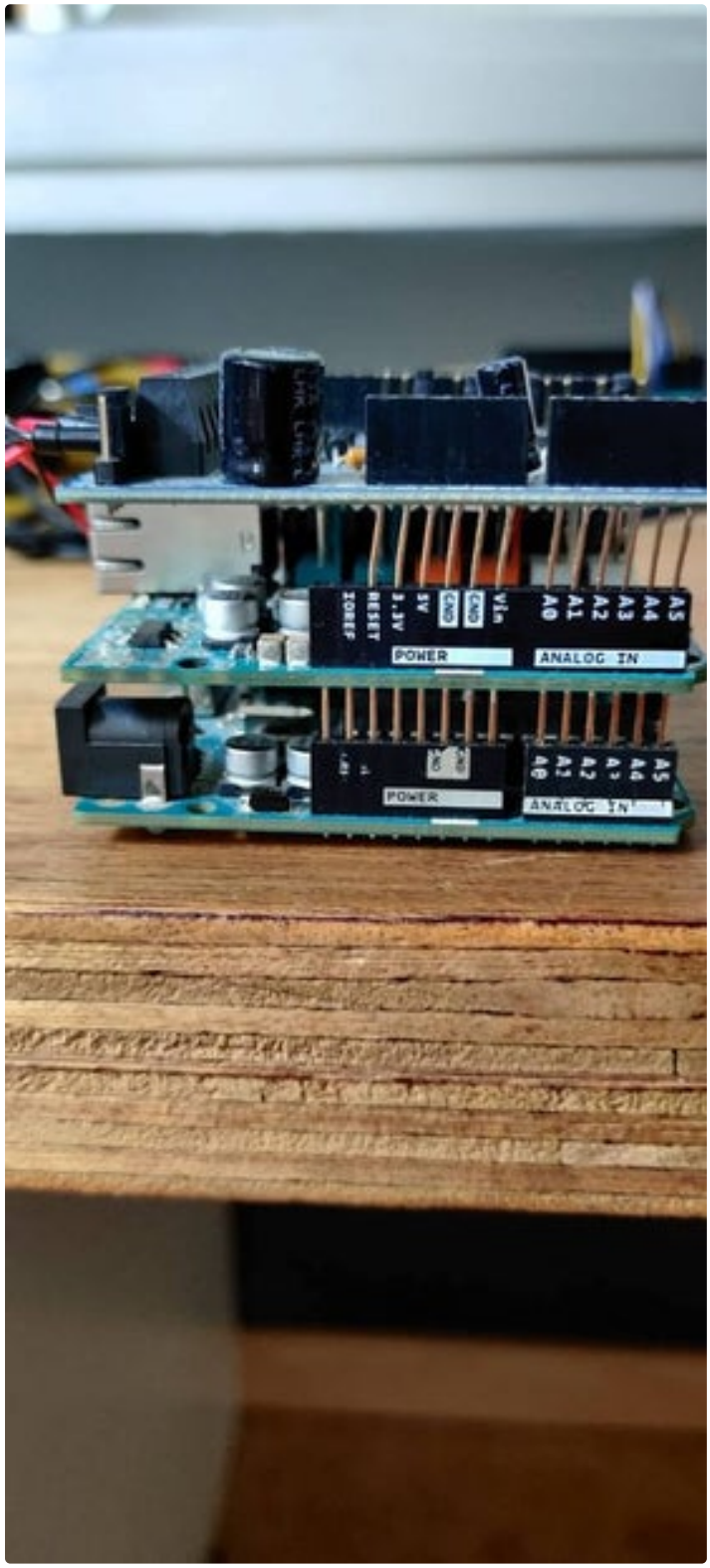
If it receives and ArtDMX packet it will decode the packet and use the given data to execute certain setup commands or move the winch in position.

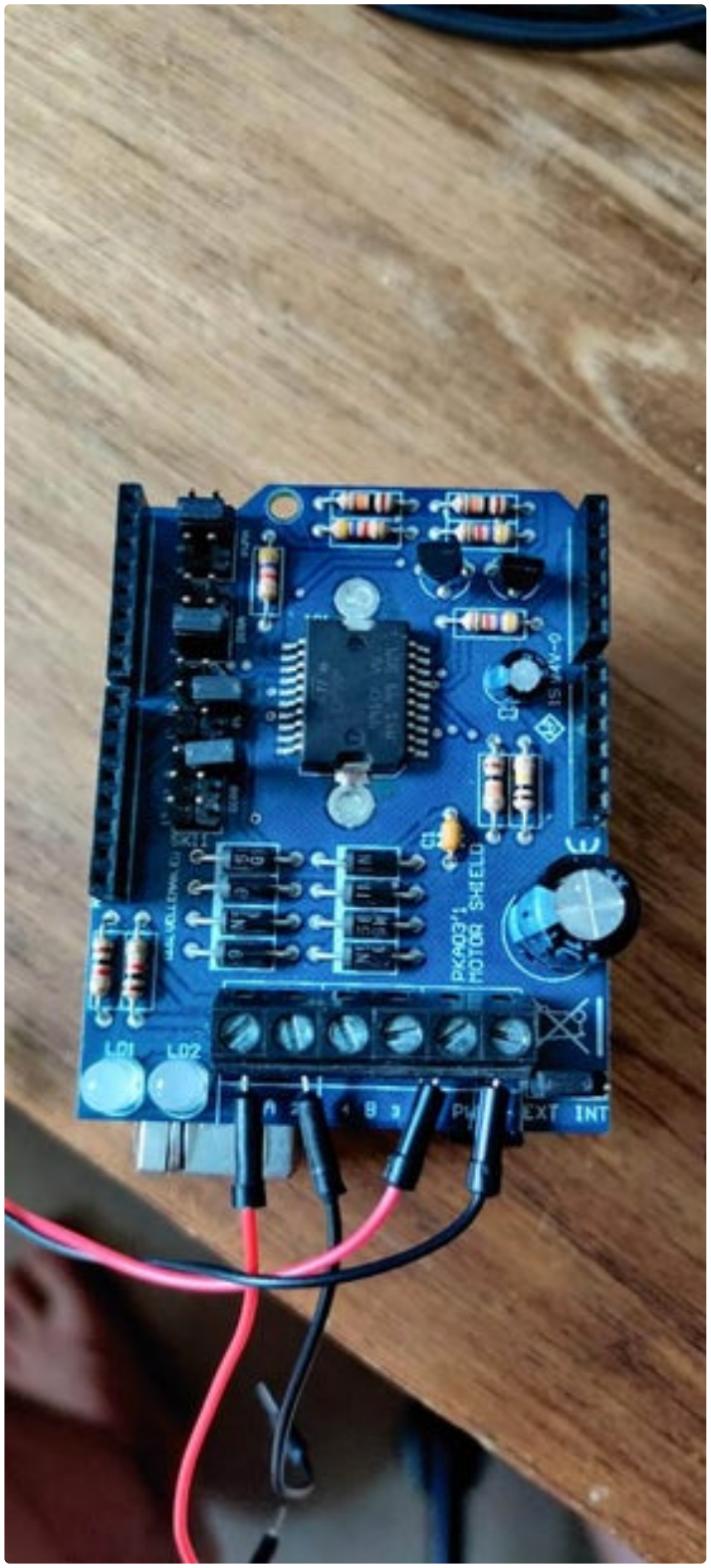
The move_takel function transforms the given DMX value (between 0-255) to a position between the start and end position (0 being the end and 255 the start). If the position of the encoder does not equal the transformed value the winch will move up/down depending on where you are.

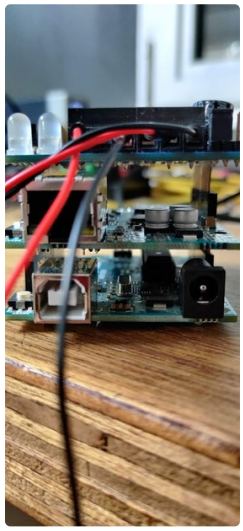
I was working on a feedback loop between the PI and arduino so that it could keep up with its position but well my arduino ran out of memory to store the program :).

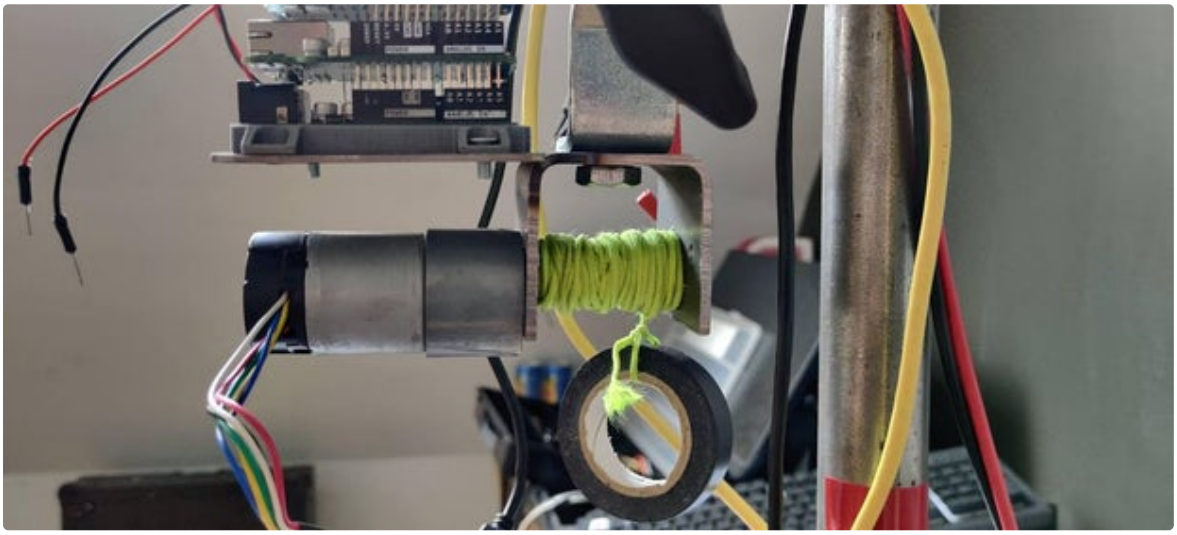
Mounting it all

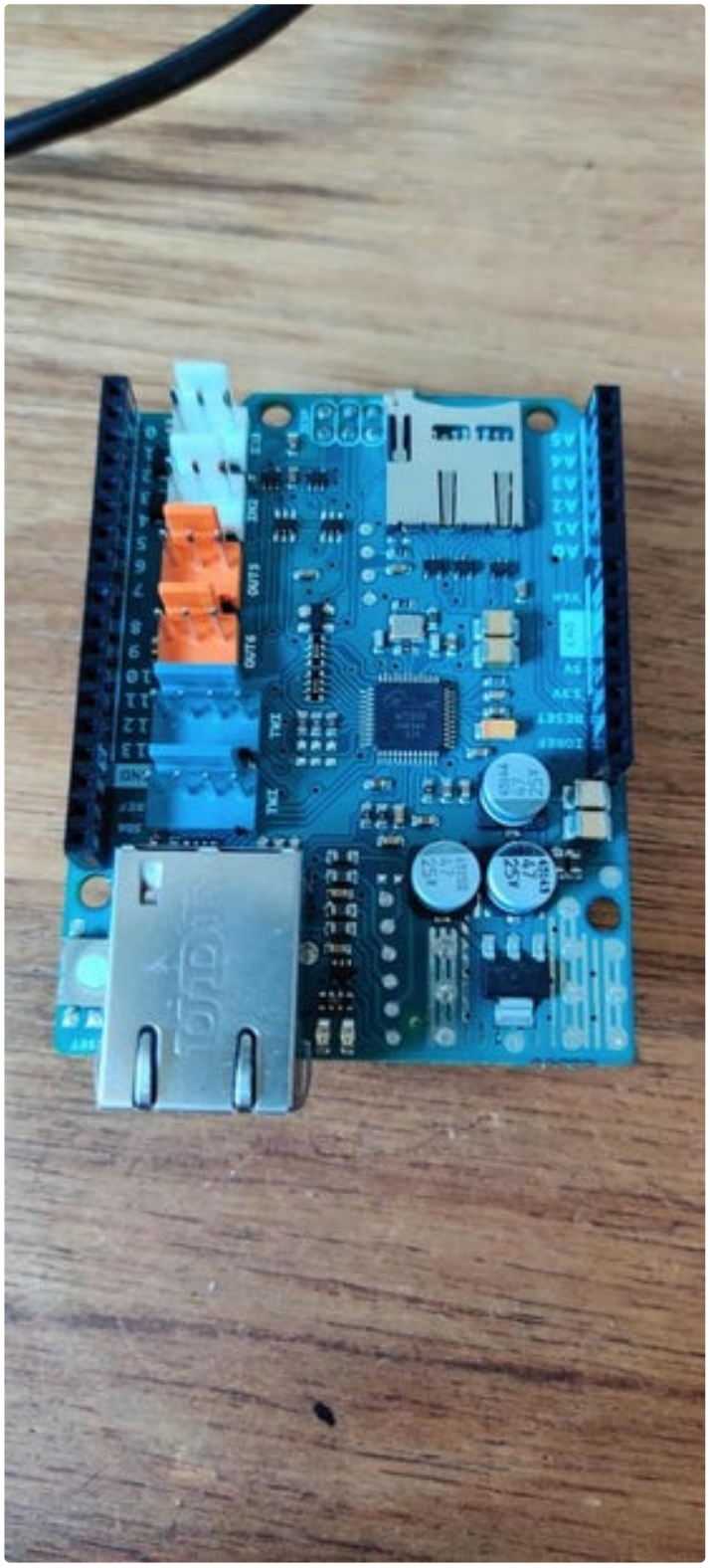
For its mounting i attached the motor to a metal holder and put a sort of tube on top of it. Then just attached a cord on the tube and used a roll of tape as a weight. This is very basic and you can get very creative in the way you want to mount it.


POWER
ANALOG IN
POWER
ANALOG IN


PXA03' MOTOR SHIELD
LD1
LD2
EXT
INT







Step 6: Connecting It All

Once the arduino and the pi are all setup simply plug in both ethernet cables in your switch and that should be it!

You can test it by surfing to the wlan ip given on the oled screen and you should see the site. Press find devices to see if you can find the device. If it shows nothing then your arduino is not well connected or the IP you've chosen is not in the same range as that of the PI.

Once you see the device just press setup. In the setup menu you can move the winch by using the arrows and test if it's start and end positions are correct with the slider.

Make sure your controller is also plugged into the shield and it's IP is in range of that of the PI and arduino.

That's it!