

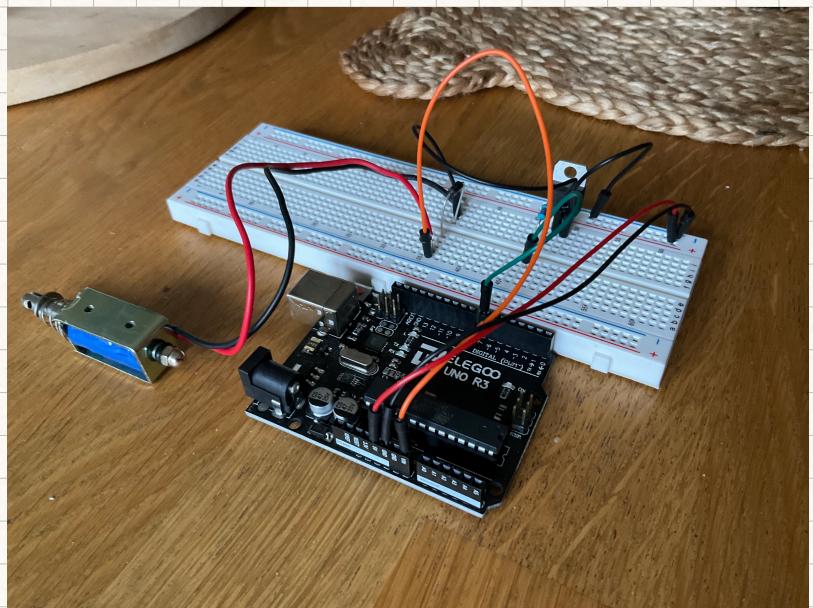
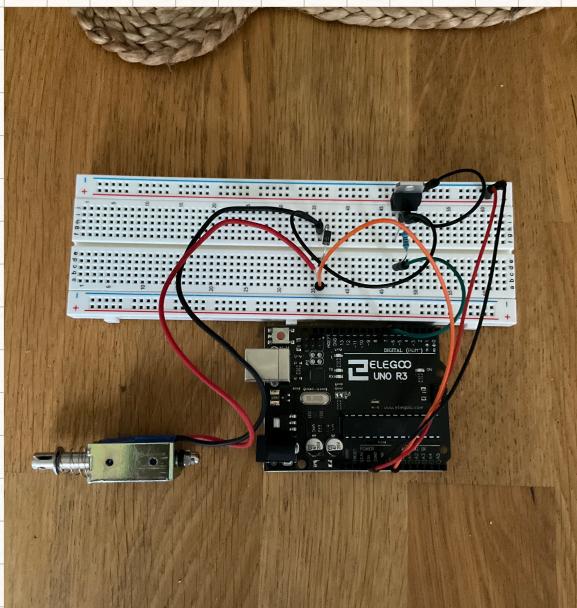
Electrical Assembly of AW actuation device - button actuation

10 NOV 2021

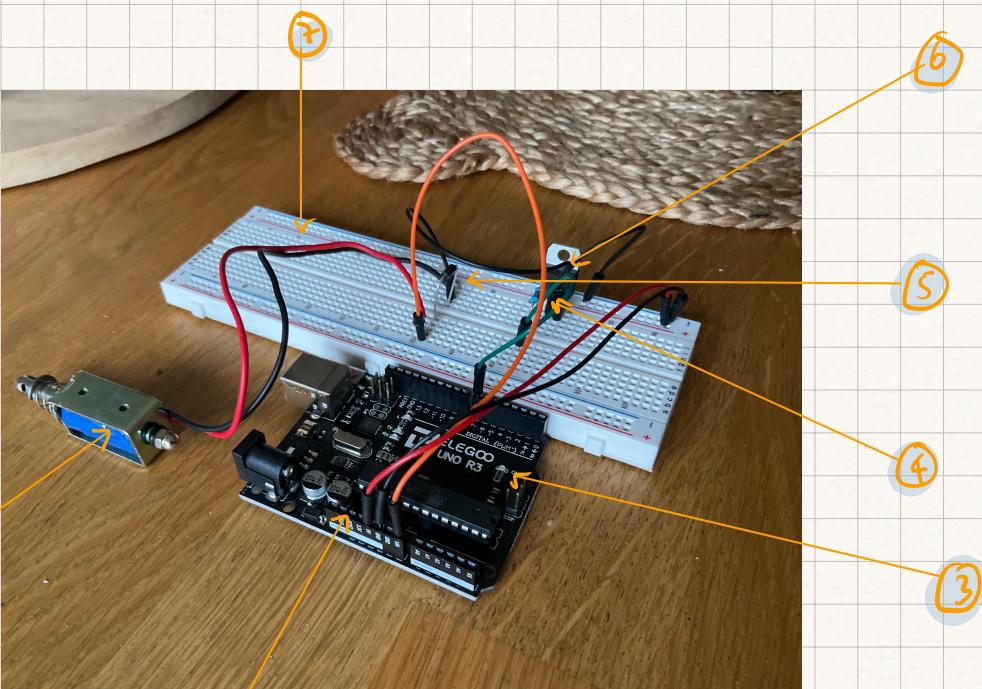
Initial Build Design

Due to the constraints found when reviewing the Model A assembly, the team decided to switch to solenoid with a linear actuation project. This helped as it replicates AW guitar open source hardware. We decided to use this mechanism and hook it up to a breadboard and get it running. Below shows a photo of the results and details of what the parts are.

Photos of electrical build with Arduino of linear solenoid Actuation



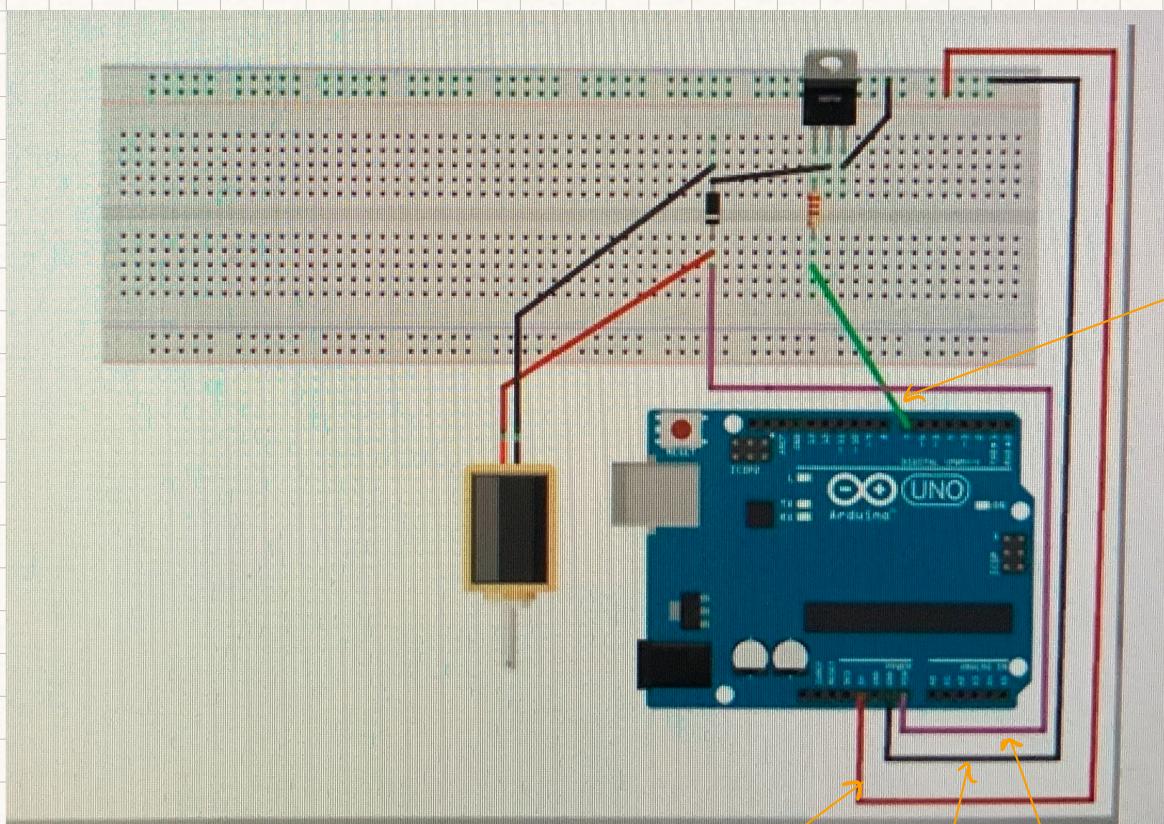
Part List



(2)

- 1 Heschen solenoid Electromagnet HS-093B DC6V 5N 10mm stroke push pull type open frame - £5.29 each
- 2 Jumper wires - £6.99 pack of 120
- 3 Elegoo Arduino UNO R3 board - £9.99 each
- 4 1k Ω resistor - £4.99 pack of 200
- 5 IN4001 diode, 1A 50V - £2.20 pack of 100
- 6 TIP120 transistor NPN 5A 60V silicon darlington transistor - £8.99 pack of 20
- 7 Haljim solderless plug in breadboard, 820 ties, 2 power lanes, 200 PTS, 16.5x5.4x0.85 cm - £4.99 each

Board layout / circuit diagram



Pin 7

* you can chose
any pin you
want as long
as it's in the
program

Credit: Arduino lesson 4
- solenoid control
using a transistor
- DIY Tech Guy

Ground

+5V to
board

Vin - to all more the
5V to go to the
power or the circuit we
use the plug in here
to raw power the arduino

Code to run the program

Define pin variable

The screenshot shows the Arduino IDE interface with a sketch titled "solenoid_pulse". The code defines a pin variable and sets up the pin mode as an output. It then enters a loop where it alternates between HIGH and LOW states with a one-second delay between each state change.

```
solenoid_pulse
int solenoidPin = 7;

void setup() {
  // put your setup code here, to run once:
  pinMode(solenoidPin, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(solenoidPin, HIGH);
  delay(1000);
  digitalWrite(solenoidPin, LOW);
  delay(1000);
}
```

Arduino Uno on /dev/ttyACM0

delay
before
high / low

write the
value of the output
to the pin

setup the pin to
be an output