“Asspire” Guide

j00d3 on GitHub

Table of Contents

1 Disclaimer and Statements

I wanted to start this off by saying: I am not an electrical or software engineer! I'm sure that the way I have been doing things for this project has not been as efficient, effective, or sustainable as it could be. I am open to suggestions! Please email me at judebiss02@gmail.com for any inquiries or suggestions.

The "Asspire" has not been tested for safety. I am not responsible for any physical harm that may be caused by the device, nor am I responsible for any mistakes made in the process of building your own device. However, I can help troubleshoot problems for your own device.

The "Asspire" was inspired by "NoGasm," an open source DIY sex toy you can find here. I believe the newer version of this project is now called the Edge-o-Matic 3000 and can be found here.

Now that all of that is out of the way, let's talk about what the "Asspire" project is about! Basically I wanted to find a way to make generative art using your butt. This includes visual art, music, you name it! Once you get some sensor readings from Arduino, you can use those numbers to do pretty much whatever you want. I'm sure I'll get some questions as to *why* I wanted to make generative art with my butt, so here's my answer in parts:

1) Butts are incredibly universal. Nearly everyone has a butt and can use the "Asspire" if they would like to.
2) The experience of sexual and bodily agency and autonomy. This device allows you to use your body and sexual expression to create things in unconventional ways, which I think is really powerful. While I can't necessarily control how people want to use this technology, this is my intention.
3) I wanted to make people think more about how they can use under-utilized parts of their bodies, in the context of art-making, to make art!
4) I think it's really conceptually interesting to create art with one of the first things your cells make as an embryo.
5) I think it's more interesting to use this device than to just slather your butt in paint and slap it on a canvas. That's just my personal taste, though.

So now that you know what this project is about, I'll be talking more about how to make and use the "Asspire!"

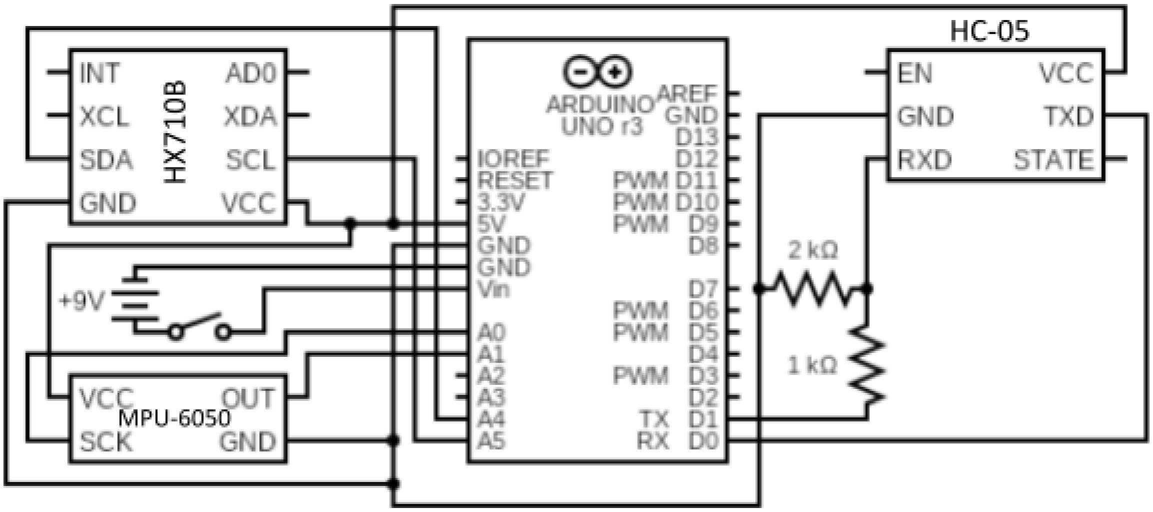
2 Materials

Bill of Necessary Materials*

Material	Quantity	Link to what I used
Arduino (I used Arduino Uno)	1	https://www.amazon.com/Arduino-A000066-ARDUINO-UNO-R3/dp/B008GRTSV6
Inflatable Buttplug	1	https://loveplugs.co/products/inflatable-2
Fanny Pack	1	https://www.amazon.com/dp/B0B9R9HKQ3?psc=1&ref=ppx_yo2ov_dt_b_product_details
Silicone Tubing (If you're using the same pressure sensor, I would recommend tubing with 3mm ID or slightly larger)	> 1ft	https://www.amazon.com/dp/B08L7YSMSZ?psc=1&ref=ppx_yo2ov_dt_b_product_details
Bluetooth Module (I used HC-05; **Note: you need capabilities for master-mode**)	1	https://www.amazon.com/dp/B074GMQ6G3?psc=1&ref=ppx_yo2ov_dt_b_product_details
Pressure Sensor (I used MPU-6050)	1	https://www.amazon.com/dp/B09XV6XHG3?psc=1&ref=ppx_yo2ov_dt_b_product_details
IMU (I used HX710B)	1	https://www.amazon.com/dp/B07V67DQ5N?psc=1&ref=ppx_yo2ov_dt_b_product_details
Silicone Caulk	1	https://www.amazon.com/dp/B0C9VYYCPB?psc=1&ref=ppx_yo2ov_dt_b_product_details
Perfboard (I used 3cm x 7cm)	1	

*There are additional materials that you can use/fabricate to make the final product a bit nicer-looking and secure. I will talk more about the optional materials in the **Other** section. Listed are the necessary components.

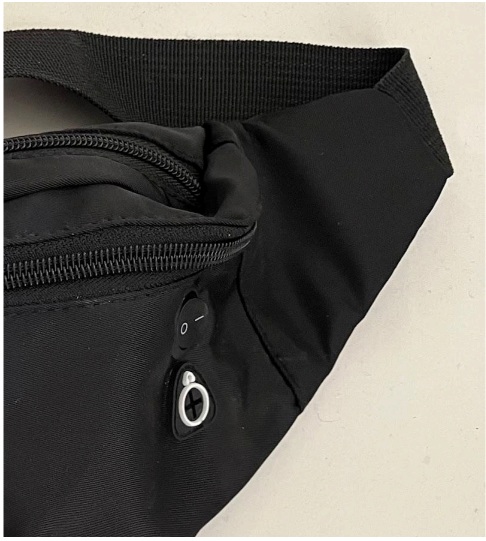
3 Electronics Diagram



*Apologies for the messiness! I don't usually make wiring diagrams.

**The HX710B is the IMU, the MPU-6050 is the pressure sensor, and the HC-05 is the bluetooth module.

I placed all of these electronics inside the fanny pack. I will talk more about my electronics mounting setup in **Other**. If you're doing something similar, I would recommend placing your switch somewhere on the outside of the fanny pack for ease of use. I cut a hole in the outside of my fanny pack that was a bit smaller than my switch so I could mount the switch externally. Here's where I placed my switch on the fanny pack:

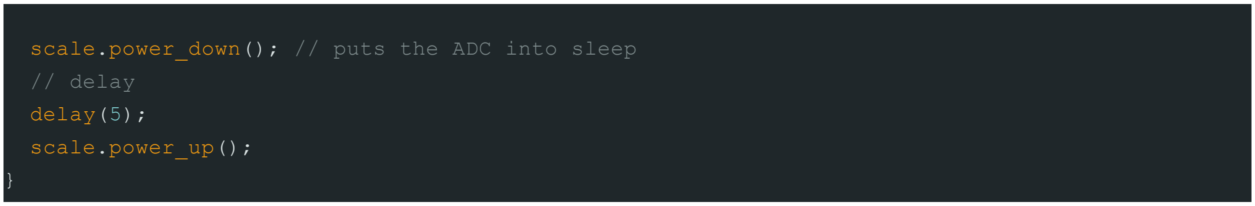


4 Arduino Code

The arduino code is in the GitHub repo. However, you can also copy the code written here:

```
// Adapted from:
//IMU example code (GY-521 module): Michael Schoeffler 2017, http://www.mschoeffler.de
//Pressure Sensor (MPS20N0040D): FILE: test_modulo_MPS20N0040D.ino; VERSION: 1.0
Description: Program for testing; pressure sensor module type MPS20N0040D; Created on
10/14/2018 by Adriano Gandolfo; Website http://www.adrirobot.it   Blog
http://it.emcelettronica.com/author/adrirobot; Facebook page
https://www.facebook.com/Adrirobot-318949048122955; Instagram
https://www.instagram.com/adrirobot/

#include "Wire.h" // This library allows you to communicate with I2C devices.

#include  "HX711.h"  // analog-to-digital converter management library

const int MPU_ADDR = 0x68; // I2C address of the MPU-6050. If AD0 pin is set to HIGH, the I2C
address will be 0x69.

int16_t accelerometer_x, accelerometer_y, accelerometer_z; // variables for accelerometer raw
data
int16_t gyro_x, gyro_y, gyro_z; // variables for gyro raw data
int16_t temperature; // variables for temperature data

char tmp_str[7]; // temporary variable used in convert function

HX711 scale;

double tareValue;

char* convert_int16_to_str(int16_t i) { // converts int16 to string. Moreover, resulting
strings will have the same length in the debug monitor.
  sprintf(tmp_str, "%6d", i);
  return tmp_str;
}

void setup() {
  Serial.begin(9600);

  // setup for IMU
  Wire.begin();
  Wire.beginTransmission(MPU_ADDR); // Begins a transmission to the I2C slave (GY-521 board)
  Wire.write(0x6B); // PWR_MGMT_1 register
  Wire.write(0); // set to zero (wakes up the MPU-6050)
  Wire.endTransmission(true);

  // setup for pressure sensor
  scale.begin (A1 ,  A0);
  scale.set_scale(2280.f); // this value is obtained by calibrating the scale with known
weights
  tareValue = scale.get_units();
```

```
}
void loop() {
  Wire.beginTransmission(MPU_ADDR);
  Wire.write(0x3B); // starting with register 0x3B (ACCEL_XOUT_H) [MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2, p.40]
  Wire.endTransmission(false); // the parameter indicates that the Arduino will send a restart. As a result, the connection is kept active.
  Wire.requestFrom(MPU_ADDR, 7*2, true); // request a total of 7*2=14 registers

  // "Wire.read()<<8 | Wire.read();" means two registers are read and stored in the same variable
  accelerometer_x = Wire.read()<<8 | Wire.read(); // reading registers: 0x3B (ACCEL_XOUT_H) and 0x3C (ACCEL_XOUT_L)
  accelerometer_y = Wire.read()<<8 | Wire.read(); // reading registers: 0x3D (ACCEL_YOUT_H) and 0x3E (ACCEL_YOUT_L)
  accelerometer_z = Wire.read()<<8 | Wire.read(); // reading registers: 0x3F (ACCEL_ZOUT_H) and 0x40 (ACCEL_ZOUT_L)
  temperature = Wire.read()<<8 | Wire.read(); // reading registers: 0x41 (TEMP_OUT_H) and 0x42 (TEMP_OUT_L)
  gyro_x = Wire.read()<<8 | Wire.read(); // reading registers: 0x43 (GYRO_XOUT_H) and 0x44 (GYRO_XOUT_L)
  gyro_y = Wire.read()<<8 | Wire.read(); // reading registers: 0x45 (GYRO_YOUT_H) and 0x46 (GYRO_YOUT_L)
  gyro_z = Wire.read()<<8 | Wire.read(); // reading registers: 0x47 (GYRO_ZOUT_H) and 0x48 (GYRO_ZOUT_L)

  // print out data from IMU
  Serial.print("aX ");
  Serial.print(convert_int16_to_str(accelerometer_x));
  Serial.print("/aY ");
  Serial.print(convert_int16_to_str(accelerometer_y));
  Serial.print("/aZ ");
  Serial.print(convert_int16_to_str(accelerometer_z));
  // the following equation was taken from the documentation [MPU-6000/MPU-6050 Register Map and Description, p.30]
  Serial.print("/temp ");
  Serial.print(temperature/340.00+36.53);
  Serial.print("/gX ");
  Serial.print(convert_int16_to_str(gyro_x));
  Serial.print("/gY ");
  Serial.print(convert_int16_to_str(gyro_y));
  Serial.print("/gZ ");
  Serial.print(convert_int16_to_str(gyro_z));

  // print out data from pressure sensor (tares for the first second the device is on)
  if (millis() < 1000){
    tareValue = scale.get_units();
  }
  Serial.print("/pre ");
  Serial.print(scale.get_units() - tareValue, 1);
  Serial.println();
```



```
  scale.power_down(); // puts the ADC into sleep
  // delay
  delay(5);
  scale.power_up();
}
```

***Important note**: when you flash the code to your arduino, you will want to remove the **RX** and **TX** pins from the arduino board. This is so that the bluetooth module isn't interfering with the USB port. After the code has been uploaded, you can reconnect the RX and TX pins to use bluetooth functionality. Be sure to check your bluetooth ports in your device's bluetooth settings. The bluetooth port you will be using will be listed under the **HC-05** (or whatever other bluetooth module you are using), and you will be using the **Incoming** port. This is so we can receive the data we need from the sensors over our serial port.

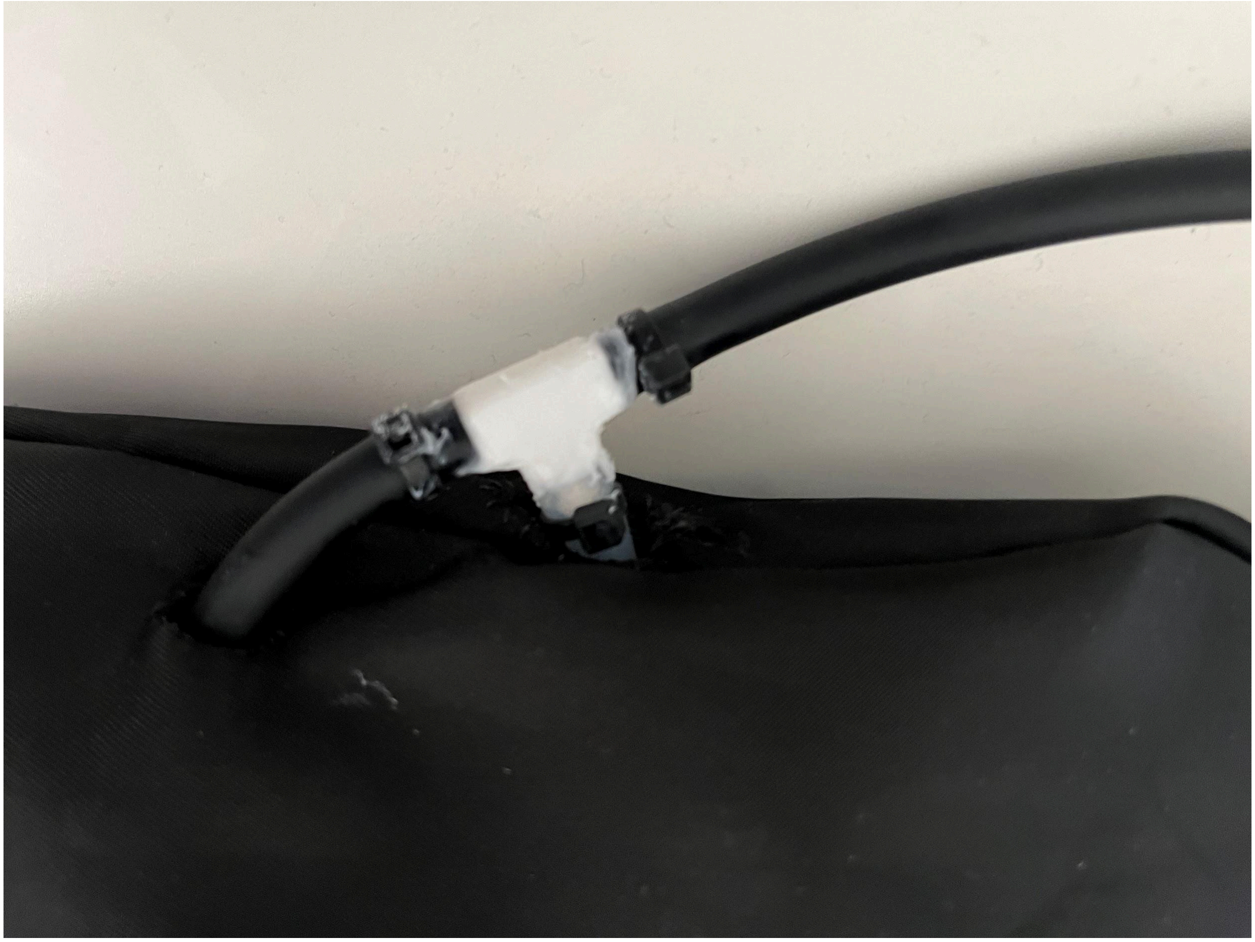
5 Tubing

In order to get pressure readings from the inflatable buttplug, we will need to split the tubing. My setup was fairly simple: I cut the tube of the inflatable buttplug in half, then 3D-printed a tee connector to install a new, third silicone tube that connected to the pressure sensor. I will talk about how to make your own tee connector in the **Other** section.

I would also recommend modifying your fanny pack for the tubing to fit. I cut two holes in my fanny pack for the tubes: one for hooking the third tube up to the pressure sensor, and the other to hold the pump.

Then, I used zip ties to secure the tubes to the tee connector and pressure sensor. Then, I used silicone caulk as directed on the packaging to ensure an air-tight fit to the tee connector and pressure sensor. **I would recommend using a tube with the same inner diameter as the outer diameter of your pressure sensor connection. This will make your life easier.**

After curing, here's what my setup ended up looking like:

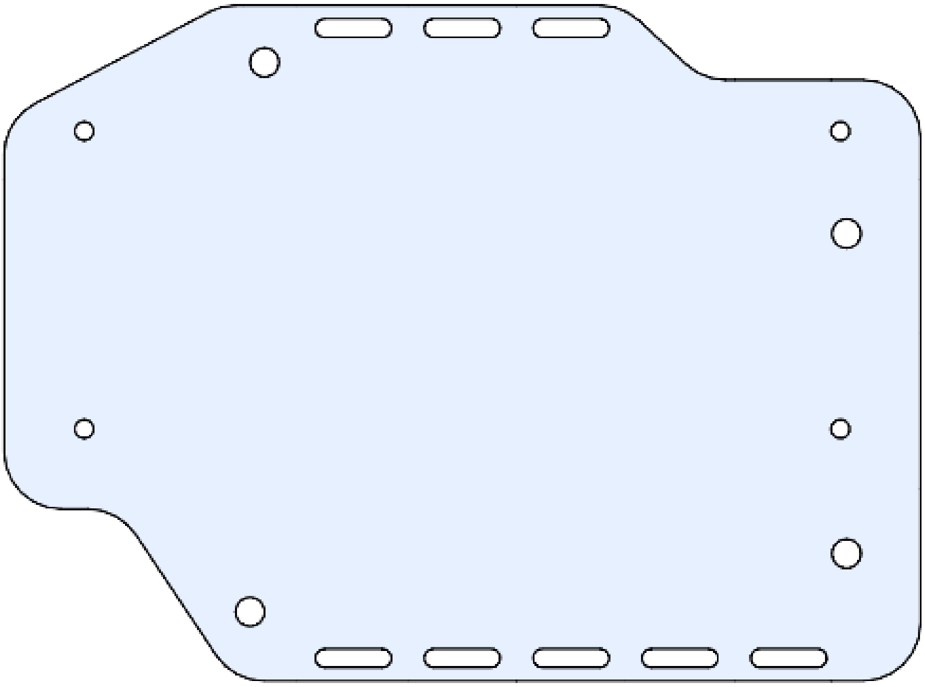


The zip ties can be cut after the silicone caulk cures if you'd like. I personally wanted to keep mine on in case there were any other issues I had to fix with the silicone caulk.

6 Other

6.1 Electronics Mounting

To mount my electronics, I made a custom laser cut piece of ⅛" acrylic sheet. The DXF file I used can be found in the GitHub repo. I would recommend some sort of mounting setup to install the electronics into the fanny pack, including the 9V battery. Here's what my DXF file looks like:

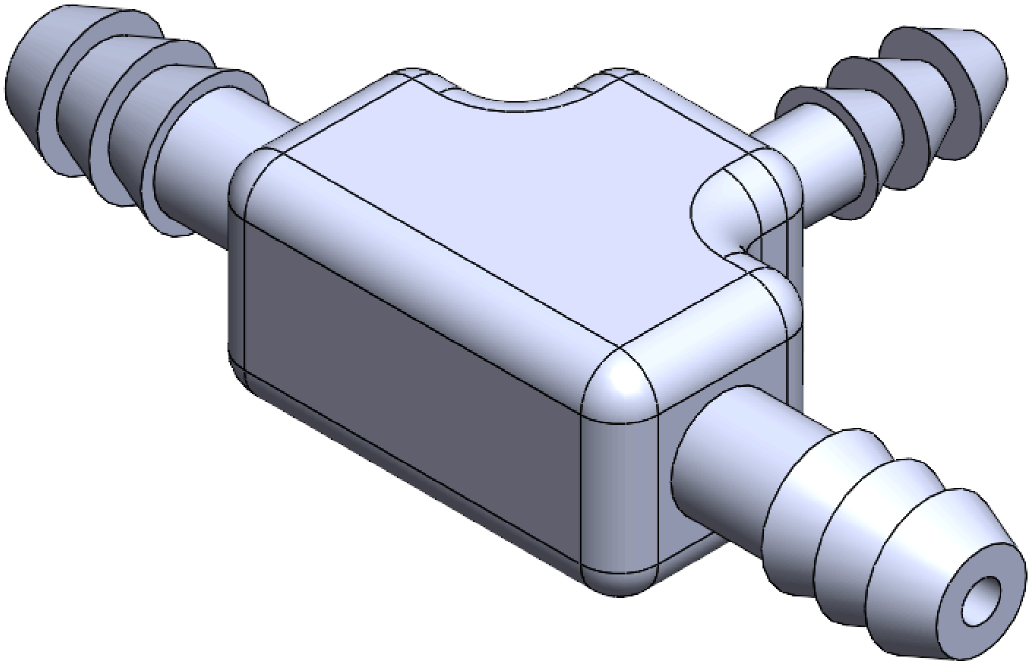


If your arduino is not on your perfboard, it is important to make sure there is ample space for your wires to connect between the arduino and perfboard if you're mounting them vertically like I did.

Regardless of the mounting configuration, I would recommend putting holes in your mounting platform that can attach to the walls of the fanny pack with thread and/or zip ties. I used zip ties for my setup because the thread I had access to was too thin for the fanny pack fabric.

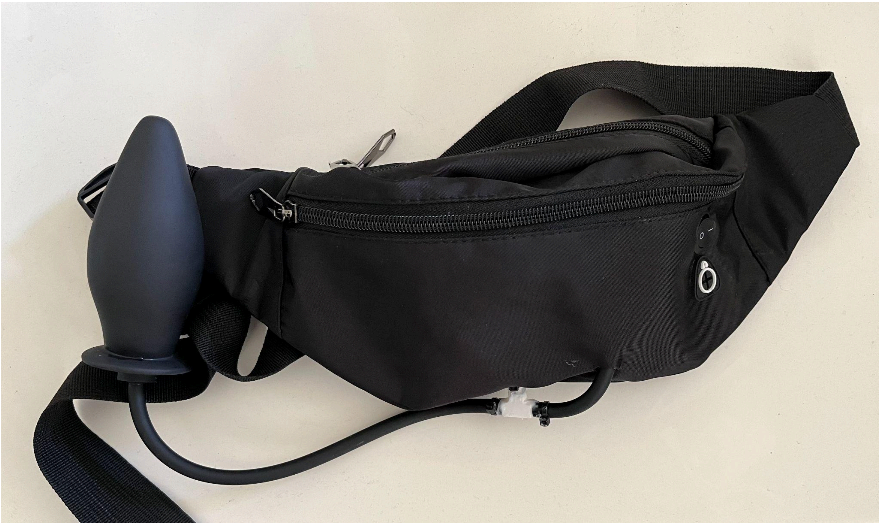
6.2 Custom Tee Connectors

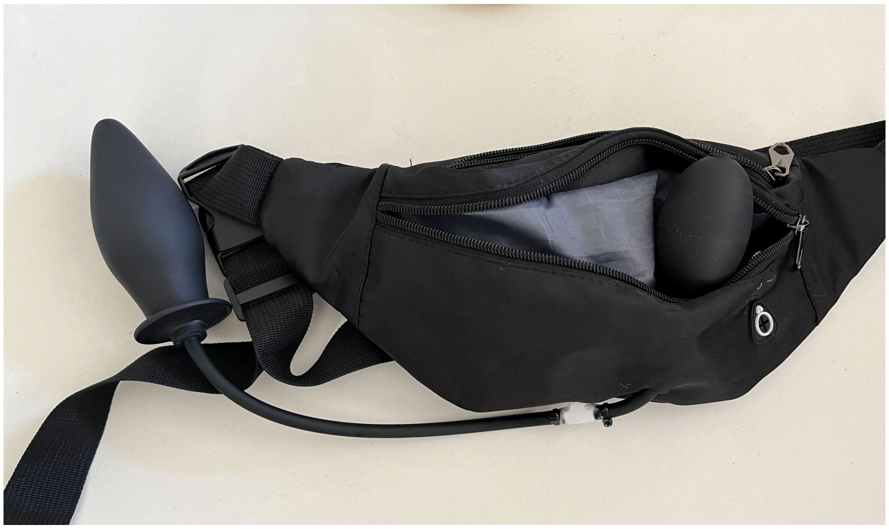
For the tubing, I made a custom 3D-printed part using a Bambuu 3D printer. Because the inner and outer diameter of the inflatable buttplug tubes were non-standard, I couldn't use off-the-shelf hardware. If you buy the same inflatable buttplug as I did, the inner diameter is 4.5mm. If you're designing your own part, I would recommend making the ridges of the insertable parts of the tee connector slightly larger than the inner diameter of your desired tube sizes. You can find the SolidWorks, STL, and STEP files for the part I designed in the GitHub repo. Note that the tubing used was 2 x 4.5mm ID and 1 x 3mm ID. Here is what the part looks like:

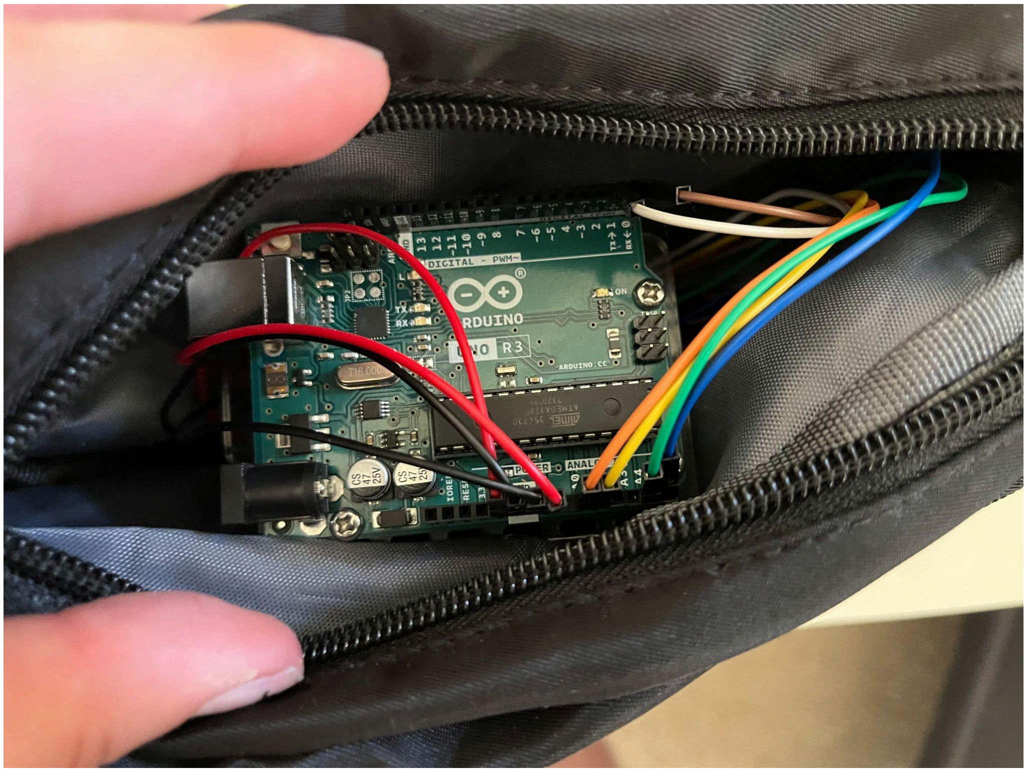


7 User Guide

Here’s what my final product looked like:







Now that all of the software and hardware is properly installed into your device, here’s the fun part: *you get to do nearly whatever you want with the data*. I used Processing and Max 8 (my code for which can be found in the GitHub repo) to make generative art and music with, but you can use whatever you’d like! For only one application, communicating over serial will work fine. However, if you are using more than one application at a time, I would recommend using OSC from one application to the others to do so. There will be a bit of delay, though.

Important note: when turning on the “Asspire,” I currently have the arduino code set to approximately zero the pressure sensor for a second. You will want to make sure your pressure is at the correct frame of reference before use. This means you should pump the inflatable buttplug as much as you are comfortable with, and to relax the buttplug before and a second after you turn the device on for best results.

Have fun!

8 Acknowledgements

This project was made possible by the Frank-Ratchye STUDIO for Creative Inquiry, Spring 2024 FRFF Micro-grant. Special thanks to Nica Ross and Harrison Apple at the STUDIO for helping me get this project funded and figuring out some legal issues regarding the project.

Thanks to my advisors for this project at the Carnegie Mellon University School of Art for helping this idea come to life: Golan Levin, Isla Hansen, Clayton Merrell, Ranee Henderson, and our TA Seamus Gallagher.

Thanks to the Miller ICA for presenting my work at Resonance, the 2024 exhibition for Carnegie Mellon University School of Art undergraduate senior show. Special thanks to Margaret Cox for her help and guidance during the exhibition process.

Thanks to all of the maker spaces I used at Carnegie Mellon University to make this project possible, including TechSpark, Robotics Club, and the School of Art digital fabrication and physical computing labs.