



# Open Source Hardware

Hacking CNC for block printing



# Our Initial Design & Dye Inspiration

## THE PATTERN

### REPEATING PATTERNS



### NEGATIVE SPACES



## THE DYES



STAMPING ON REACTIVE  
NATURAL DIED FABRIC

[inspired by printing with bleach]

- \* Iron reactive dye
- \* pH reactive
- \* mordanted fabric with tannins
- stamped with iron = black



e.g. CORREOPSIS x IRON

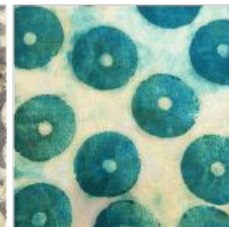


e.g. RED CABBAGE x LEMON JUICE [pink]  
or x BAKING SODA [blue]

### SOY RESIST WAX



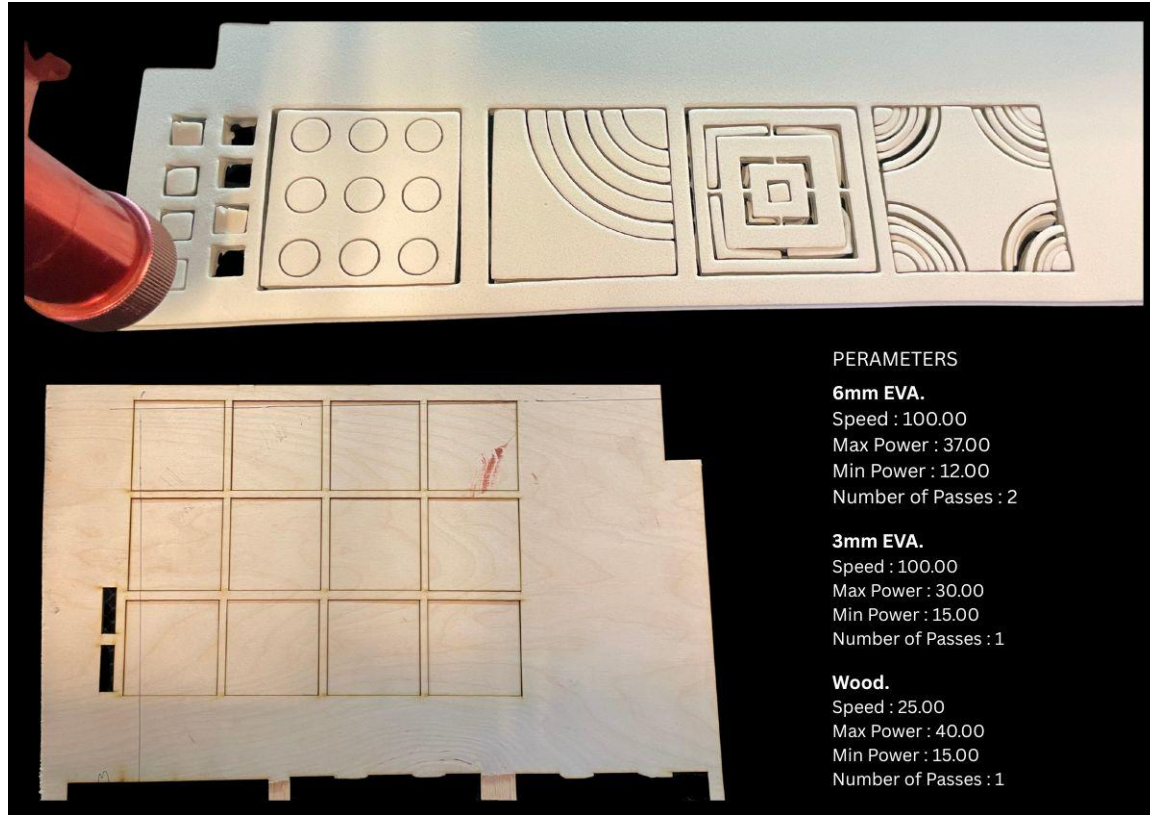
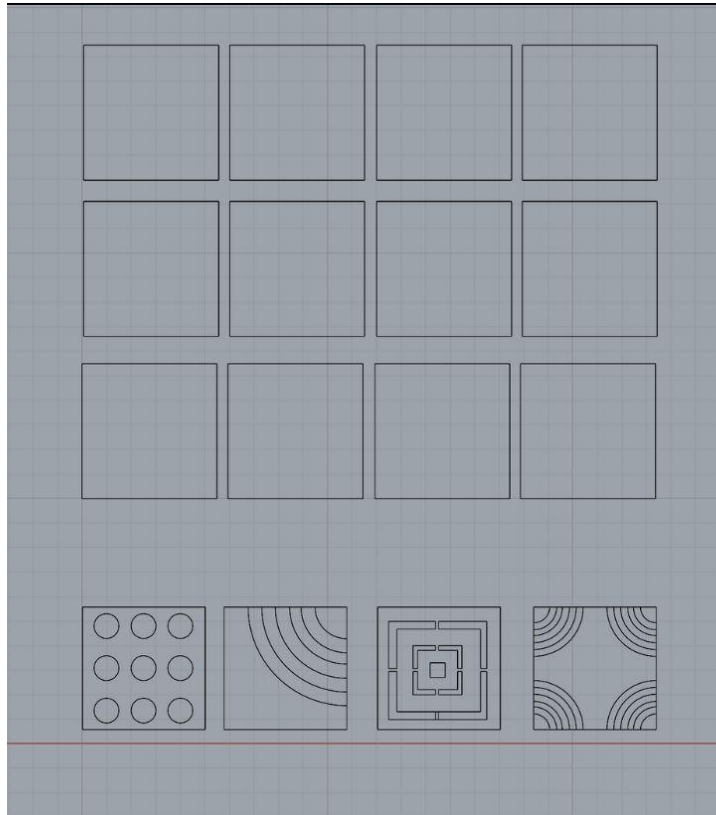
NATURAL DYE



NATURAL DYE x BLEACH

# Prototyping the Stamps

: Laser Cutting EVA and Wood  
in a range of patterns





# Prototyping the Stamps

: Experimenting with the different thicknesses and quality of stamps.



\*6 mm EVA worked best due to thickness\*



3mm EVA :  
3 consecutive stamps [no sponge] by hand



6mm EVA :  
1 stamp [with sponge] by hand

FIRST EXPERIMENTS  
: WELD, GUAR & VINEGAR

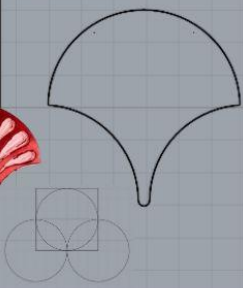


# Preparing the Final Stamp

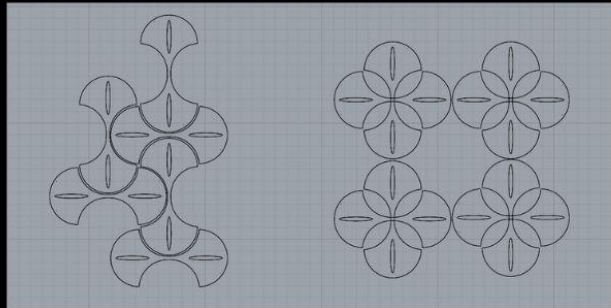
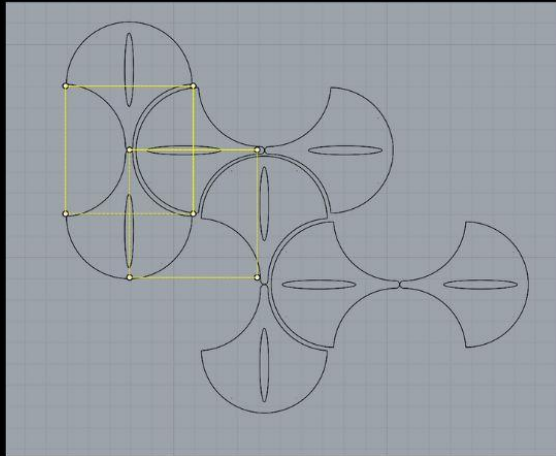
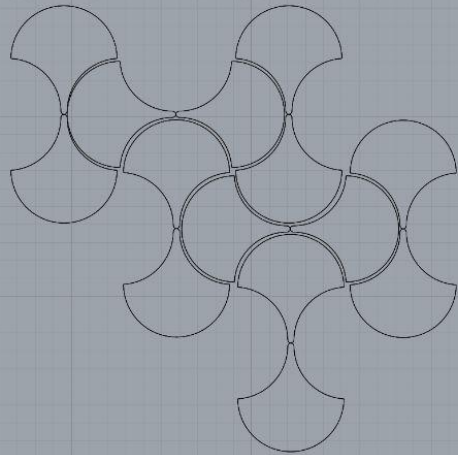
: working with Rhino to make a precise design & visualise layout.



INSPIRATION TRANSLATED  
INTO RHINO3D :



VISUALISING PATTERN & LAYOUT :



[double sided to reduce  
number of rotations needed]

THE FINAL STAMP :



: 6mm EVA on Wood

# Fabric preparation



## Preparing the fabric:

### 1. Scouring

### 2. Premordanting

12% oak galls of WOF  
simmer & stir for 2 hours

### 3. Mordanting

12% alum + 1,5% soda of WOF  
simmer & stir for 2 hours



# Ink Recipes

Coreopsis ink:  
- 200ml water  
- Handful of coreopsis  
- Allow heat for 40 minutes at 60-80 degrees.  
- Add a binder.



Making a coreopsis dye bath & condensed ink

Cochineal ink:  
- 50 grams of cochineal.  
- Grinned up  
- Add a hot drop of water  
- Sieve for smoothness.  
- Add a binder.



Weld ink:  
- 30 grams of weld extract.  
- Hot boiling water.  
- Add a binder.

# BoM of making the inks & dyes

Title	Quantity	Notes	Cost	Link
<b>Coreopsis</b>	Handful	In a concentrated bath (200ml), get a handful of coreopsis allow to cook for 40 minutes at 60 - 80 degrees.	€0, Foraged from previous years.	<a href="#">Tuinflora</a>
<b>Cochineal</b>	20 grams	Grind up cochineal using a pestle and mortar. Dilute with some boiling water (just a few drops). Sieve for smoothness.	€3.45/10g	<a href="#">Classicfabrics</a>
<b>Weld</b>	15 grams	To make a concentrated bath just add a little bit of boiling water.	€16/50g	<a href="#">Green ingredients</a>
<b>Alginate</b>	20 grams	12 grams of unique agar, 20 grams of glycerine, 400 ml water & blend.	€22	<a href="#">Unique Agar</a>
<b>Iron</b>	a pinch		€7/500g	<a href="#">Laboratorium discounter</a>
<b>Safflower</b>	8 grams (remaining lake pigment)			<a href="#">12 Taste</a>
<b>Soy beans</b>	2 handfuls (30 grams)	2 handfuls in boiling water for 2-3 hours. Blended up. Adding clean water, keeping it thick. Sieving through a cheesecloth.	€2/500g	<a href="#">Bionoot</a>
<b>Xanthan Gum</b>	2% per 100ml	Add to boiling water and mix well.	€5/100g	<a href="#">Holland &amp; Barret</a>
<b>Guar Gum</b>	2% per 100ml	Add to boiling water and mix well.	€5/50g	<a href="#">Hekserij</a>
<b>CMC</b>	2% per 100ml	Add to boiling water and mix well.	€12/500g	<a href="#">Labshop</a>
<b>Citric acid</b>	5-10 g	Add to boiling water and mix well.	€6/1kg	<a href="#">Natuur product</a>
<b>Alum</b>	12% alum	Add to boiling water and mix well.	€10/1kg	<a href="#">Deonlinedrogist</a>





# Soy mordant prints



Soy beans



Soak & boil



Blend



Through a cheese cloth



Different binder



Tests

## Soy mordant prints

2 handful of soy bean

Add boiling water, more than just covered and wait till swollen (usually 3-4 h with hot water; a day with cold)

—

Strain and throw away swelling water

Put the beans in a mixer / grinder and only add clean water little by little (maintain it thick as possible)

Put the mix through a cheese cloth - the liquid is the base of your paint.

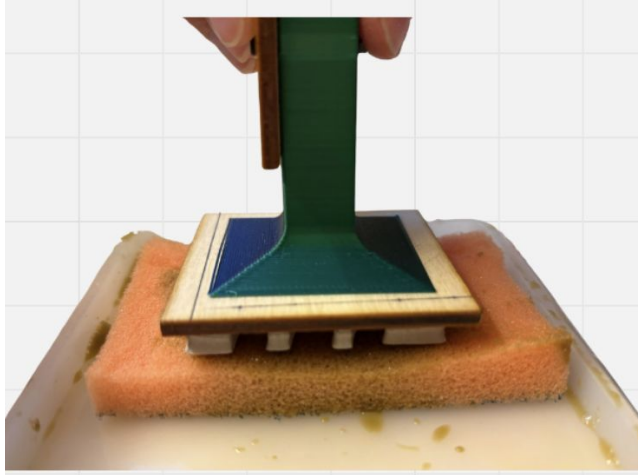
Repeat the process of adding little water and grinding the same beans a few times

Collect all the liquid from the various strained / grinded beans bath

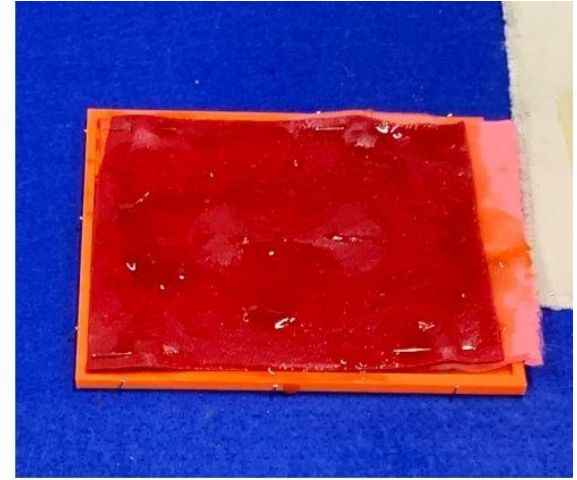
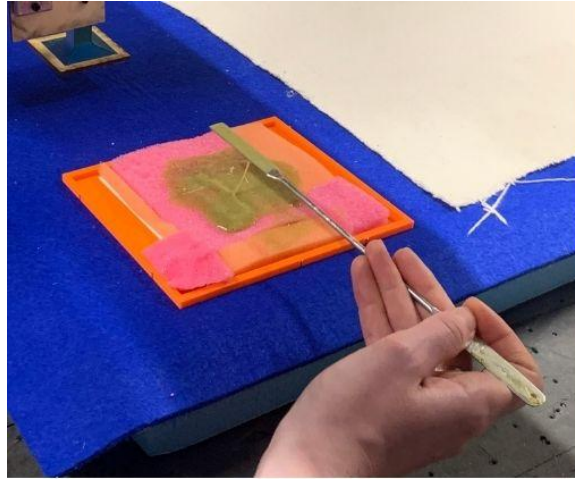
## Test: Thickener for Soy mordant ink

1. Pure soy milk
2. Soy milk + guar gum
3. Soy milk + xanthan gum
4. soy milk + alginate

# Ink Pad

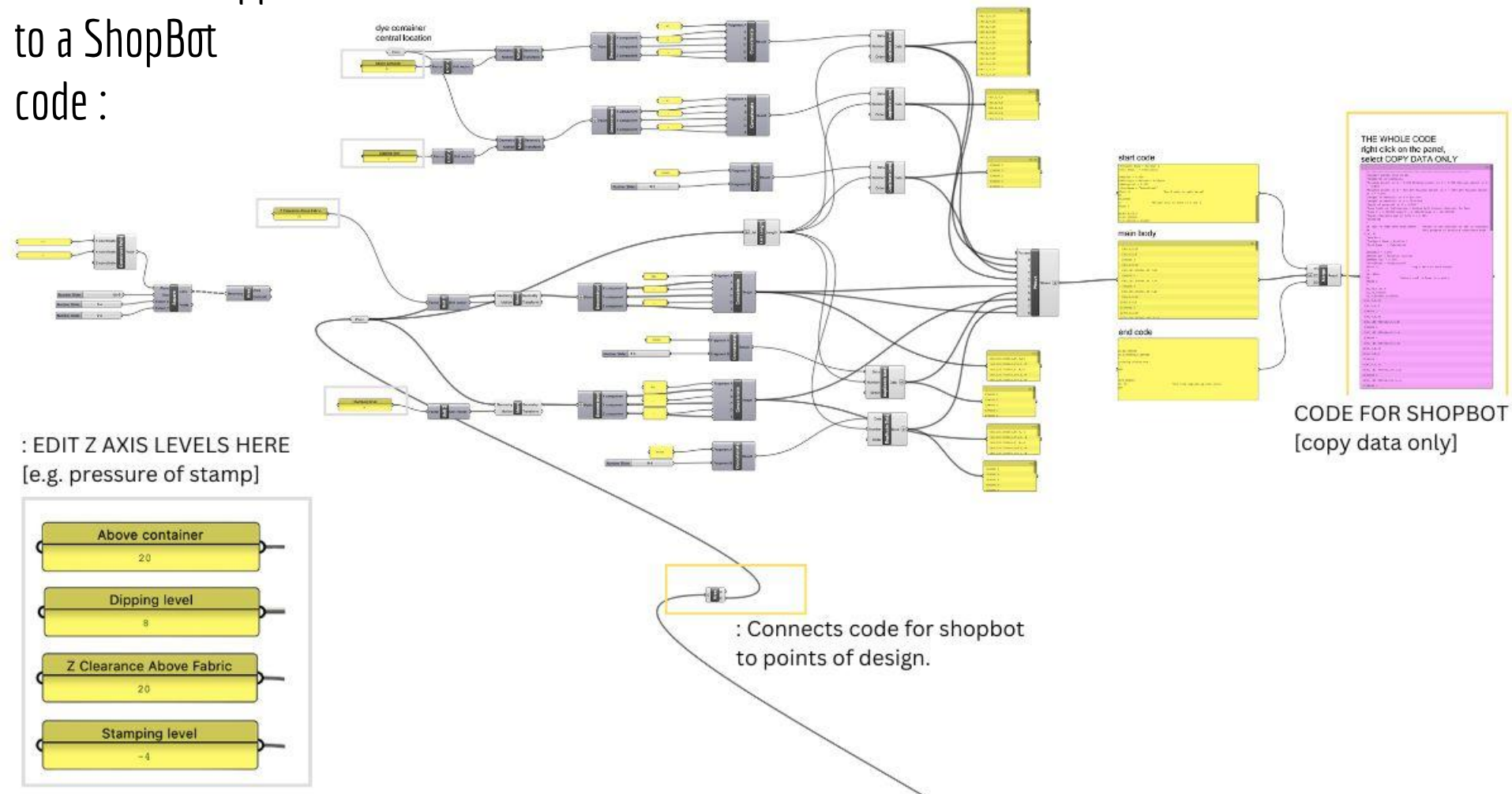


sponge



felt

# From Grasshopper to a ShopBot code :



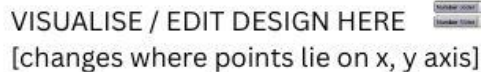
: EDIT Z AXIS LEVELS HERE  
[e.g. pressure of stamp]

: Connects code for shopbot  
to points of design.


CODE FOR SHOPBOT  
[copy data only]



A line graph with a single line sloping downwards from the top-left to the bottom-right. The line is straight and has a constant negative slope.

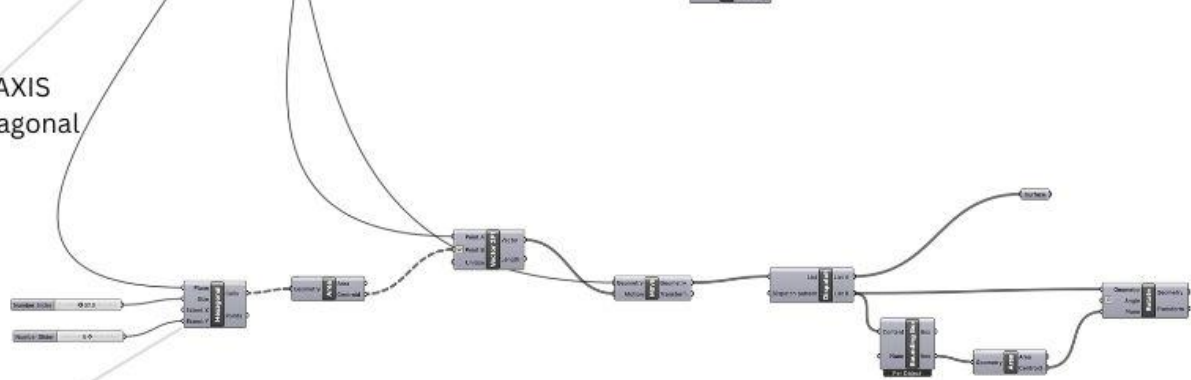


bake me!



: For points when stamp is VERTICAL, see image below]

The diagram illustrates the relationship between various geometric parameters for two types of shapes: Square and Hexagonal. It is organized into two rows, one for 'Square' and one for 'Hexagonal'. Each row contains a central column with the shape name, flanked by two columns of parameters. The 'Square' row lists 'Plane', 'Size', 'Extent X', and 'Extent Y' on the left, and 'Cells' and 'Points' on the right. The 'Hexagonal' row lists 'Plane', 'Size', 'Extent X', and 'Extent Y' on the left, and 'Cells' and 'Points' on the right. The central column is labeled 'Square' and 'Hexagonal' respectively.



\*visualising the stamp and points on the axis\*

# Getting the code from Grasshopper to ShopBot :

## STEP-BY-STEP.

[once the Shopbot code and design is ready]

- Connect the points within the design to the Area and therefore the rest of the code for the Shopbot.

[do this twice if there are different points for rotated designs e.g. pattern5A.sbp was horizontal and pattern5B.sbp was vertical]

- BAKE [the correct axis points]
- [take note of the setting you used or export high-res image of grasshopper]
- Then within large pink, panel, right click and 'Copy Data Only'
- Open TextEdit
- Paste...
- Format... 'Make Plain Text'
- Save.
- [Name] **.sbp**
- Untick ☐ If no extension is provided, use ".txt".
- Save onto USB.

The code within Grasshopper :

select COPY DATA ONLY

```
!SHOBBOT ROUTER FILE IN ROM
!GENERATED BY: HATCHWORK
!Minimum extent in X = 0.000 Minimum extent in Y = 0.000 Minimum extent in Z = -0.000
!Maximum extent in X = 830.000 Maximum extent in Y = 1550.000 Maximum extent in Z = 0.000
!Length of material in X = 830.000
!Length of material in Y = 1550.000
!Length of material in Z = 0.000
!Home Position Information = Bottom Left Corner, Material Surface
!Home X = 0.000000 Home Y = 0.000000 Home Z = 20.000000
!Rapid clearance gap or Safe Z = 4.000
!UNIT2:MM
!
IF $(!?) THEN GOTO UNIT_ERROR "Check to see software is set to standard
GOTO 90 "See program to absolute coordinate mode
!Toolpath Name = Profile 1
!Tool Name = FabricBrush
!WorkOffset = 0.000
!WorkOrigin = Material Surface
!WorkMaterial = 0.000
!WorkName = "FabricBrush" "Dog Z axis to safe height
!Tool -1
G7
TR,18000
CE "Return tool to home in x and y
PAUSE 2
M3,89,0,20,0
J2,20.000000,0.000000,0.000000
M3,0,0,20
ZM3,0,0,4
ZM3,0,0,4
ZPAUSE 3
M3,586.235572,130.840467,20
PAUSE 0
M3,586.235572,130.840467,-5
PAUSE 4
M3,586.235572,130.840467,20
PAUSE 4
M3,586.235572,130.840467,20
M3,0,0,20
M3,0,0,4
PAUSE 3
M3,0,0,20
M3,586.235572,211.040467,20
PAUSE 0
M3,586.235572,211.040467,-5
PAUSE 4
M3,586.235572,211.040467,20
J2,20.000000
J2,0.000000,0.000000,0.000000
!
!Turning router OFF
C7
END
!
```

The code within TextEdit :

```
!SHOBBOT ROUTER FILE IN ROM
!GENERATED BY: HATCHWORK
!Minimum extent in X = 0.000 Minimum extent in Y = 0.000 Minimum extent in Z = -0.000
!Maximum extent in X = 830.000 Maximum extent in Y = 1550.000 Maximum extent in Z = 0.000
!Length of material in X = 830.000
!Length of material in Y = 1550.000
!Length of material in Z = 0.000
!Home Position Information = Bottom Left Corner, Material Surface
!Home X = 0.000000 Home Y = 0.000000 Home Z = 20.000000
!Rapid clearance gap or Safe Z = 4.000
!UNIT2:MM
!
IF $(!?) THEN GOTO UNIT_ERROR "Check to see software is set to standard
GOTO 90 "See program to absolute coordinate mode
!Toolpath Name = Profile 1
!Tool Name = FabricBrush
!WorkOffset = 0.000
!WorkOrigin = Material Surface
!WorkMaterial = 0.000
!WorkName = "FabricBrush" "Dog Z axis to safe height
!Tool -1
G7
TR,18000
CE "Return tool to home in x and y
PAUSE 2
M3,89,0,20,0
J2,20.000000,0.000000,0.000000
M3,0,0,20
ZM3,0,0,4
ZM3,0,0,4
ZPAUSE 3
M3,586.235572,130.840467,20
PAUSE 0
M3,586.235572,130.840467,-5
PAUSE 4
M3,586.235572,130.840467,20
PAUSE 4
M3,586.235572,130.840467,20
M3,0,0,20
M3,0,0,4
PAUSE 3
M3,0,0,20
M3,586.235572,211.040467,20
PAUSE 0
M3,586.235572,211.040467,-5
PAUSE 4
M3,586.235572,211.040467,20
J2,20.000000
J2,0.000000,0.000000,0.000000
!
!Turning router OFF
C7
END
!
```

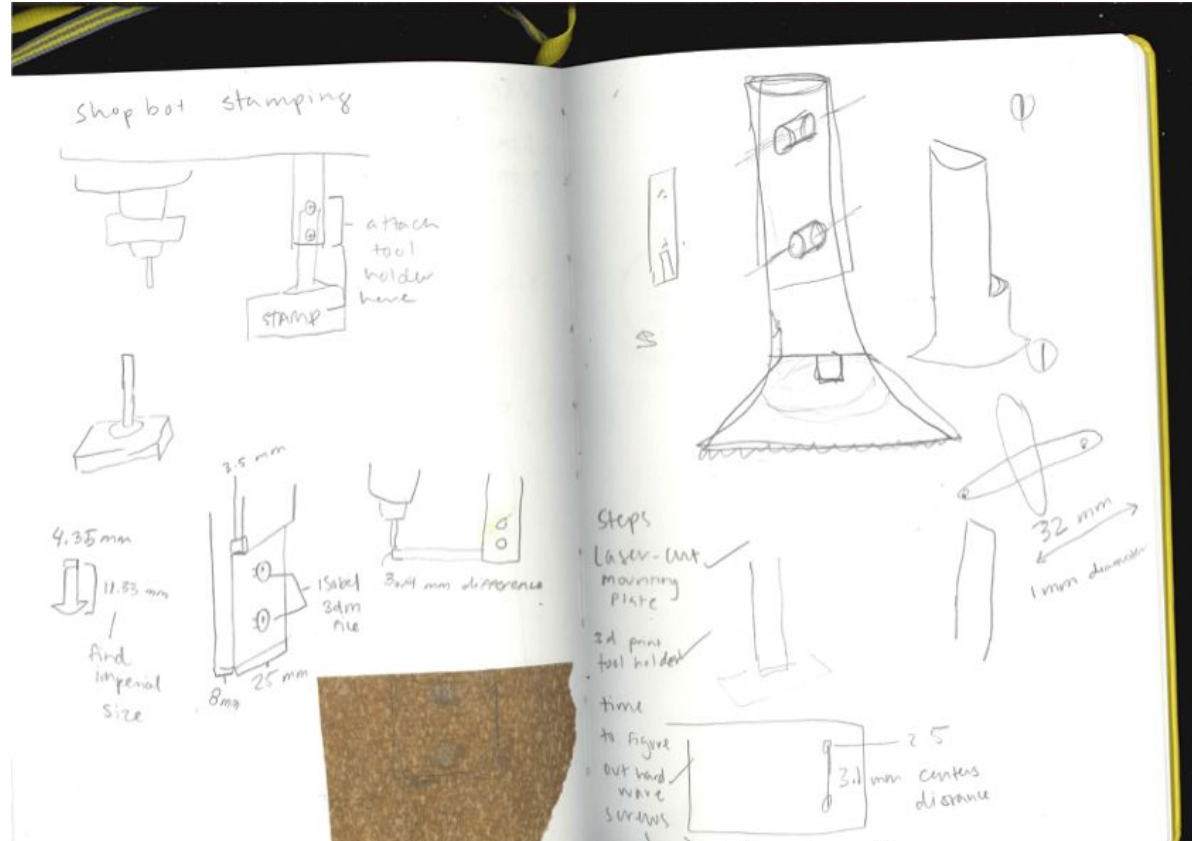
osave complete (160 seconds ago)

UNIT\_ERROR:  
CN, 91  
END

'Run file explaining unit error

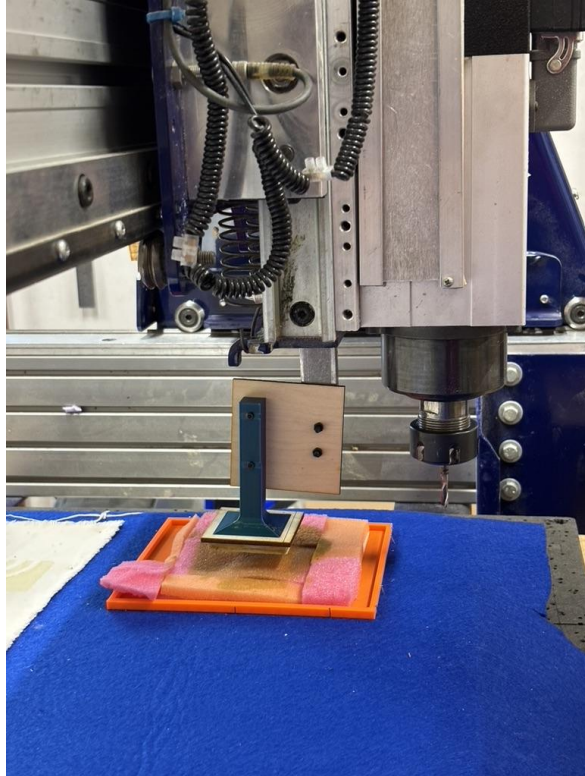
# 3D Printed Tool Parts: Sketch & Measure

- ❖ Attach the stamp tool where the "skirt" is usually attached
- ❖ Wide pyramid shape can distribute pressure evenly throughout the stamp
- ❖ Obtain measurements for the screw holes from Isobel & Carolina's 2024 documentation





# 3D Printed Tool Parts: First attempt & Ink pad



- ❖ Static, no electronics involved
- ❖ Not centered with the CNC tool
- ❖ Low infill and thin base led to breaking under pressure
- ❖ Unnecessary laser cut mounting plate
- ❖ Ink pad is large and low-profile to accommodate stamp sizes and clearance height

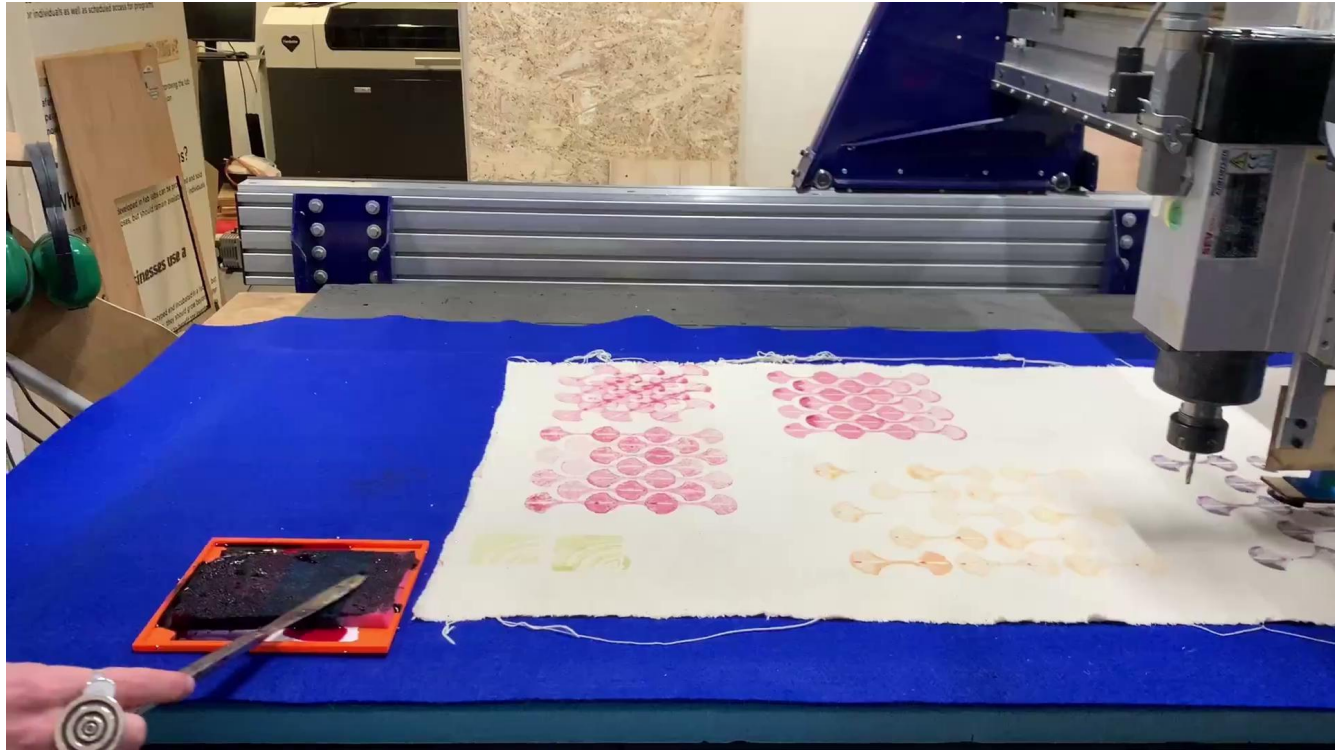
# 3D Printed Tool Parts: Second iteration



- ❖ Static, no electronics involved
- ❖ Mounts directly to machine
- ❖ Centered with the CNC tool
- ❖ Higher infill % and thicker base to prevent cracking

[Link](#)

# Stamping

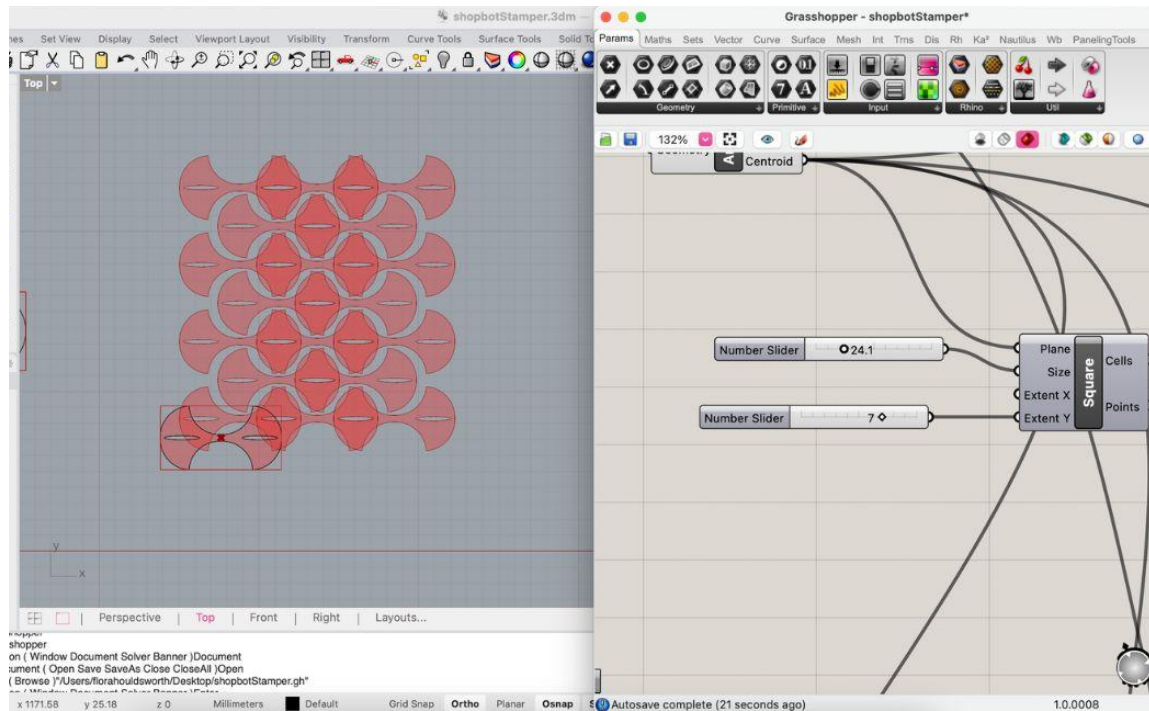


[Link](#) to Vimeo



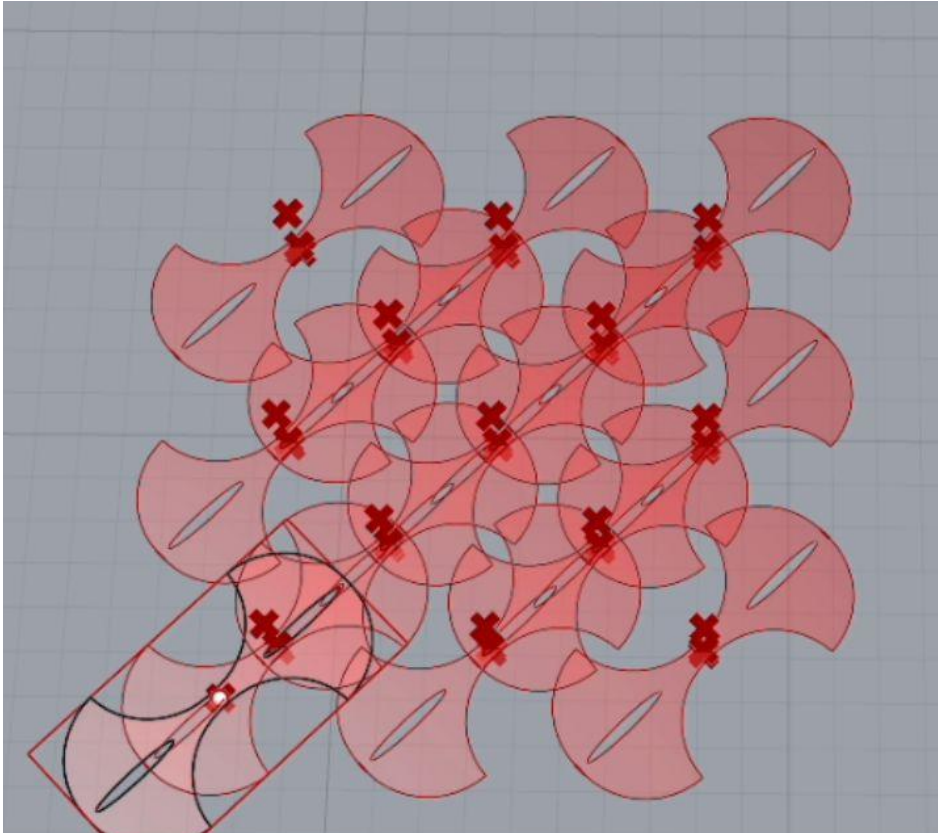
# Printing Experiments [from Grasshopper to Textile]

: Cochineal, Alginate & Water



: Cochineal, Alginate & Water

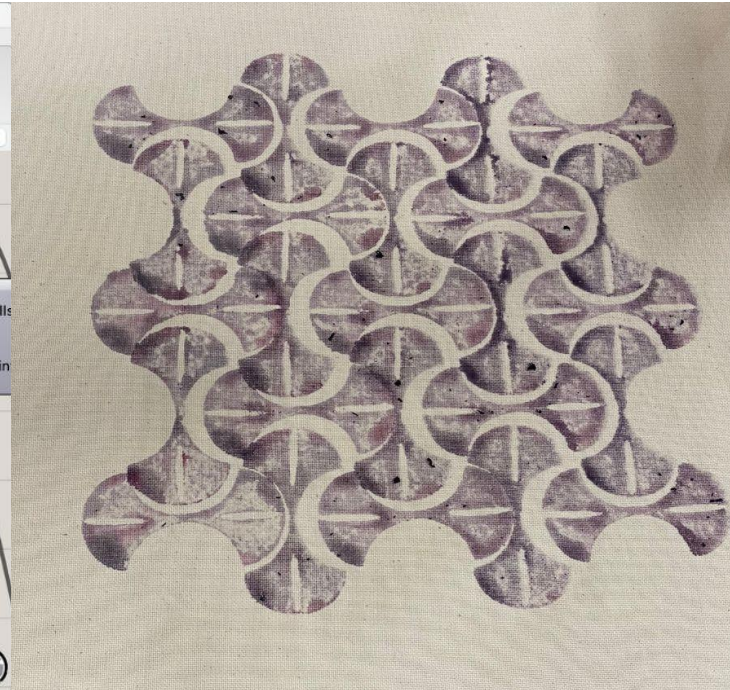
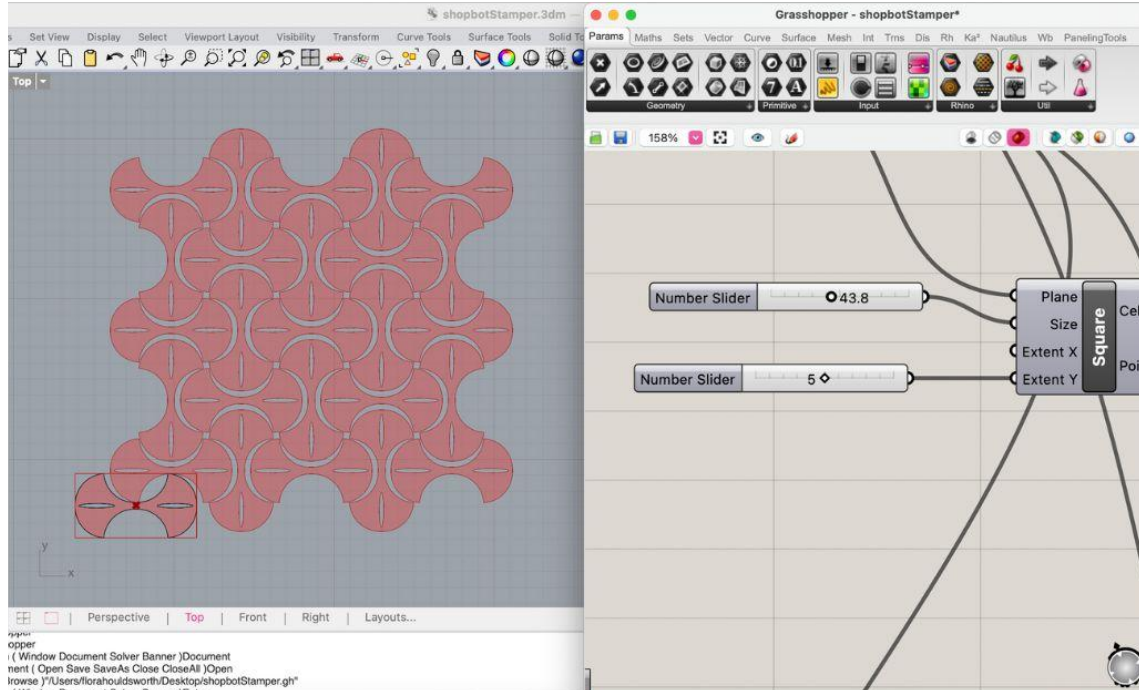
\*stamp manually turned 45° \*



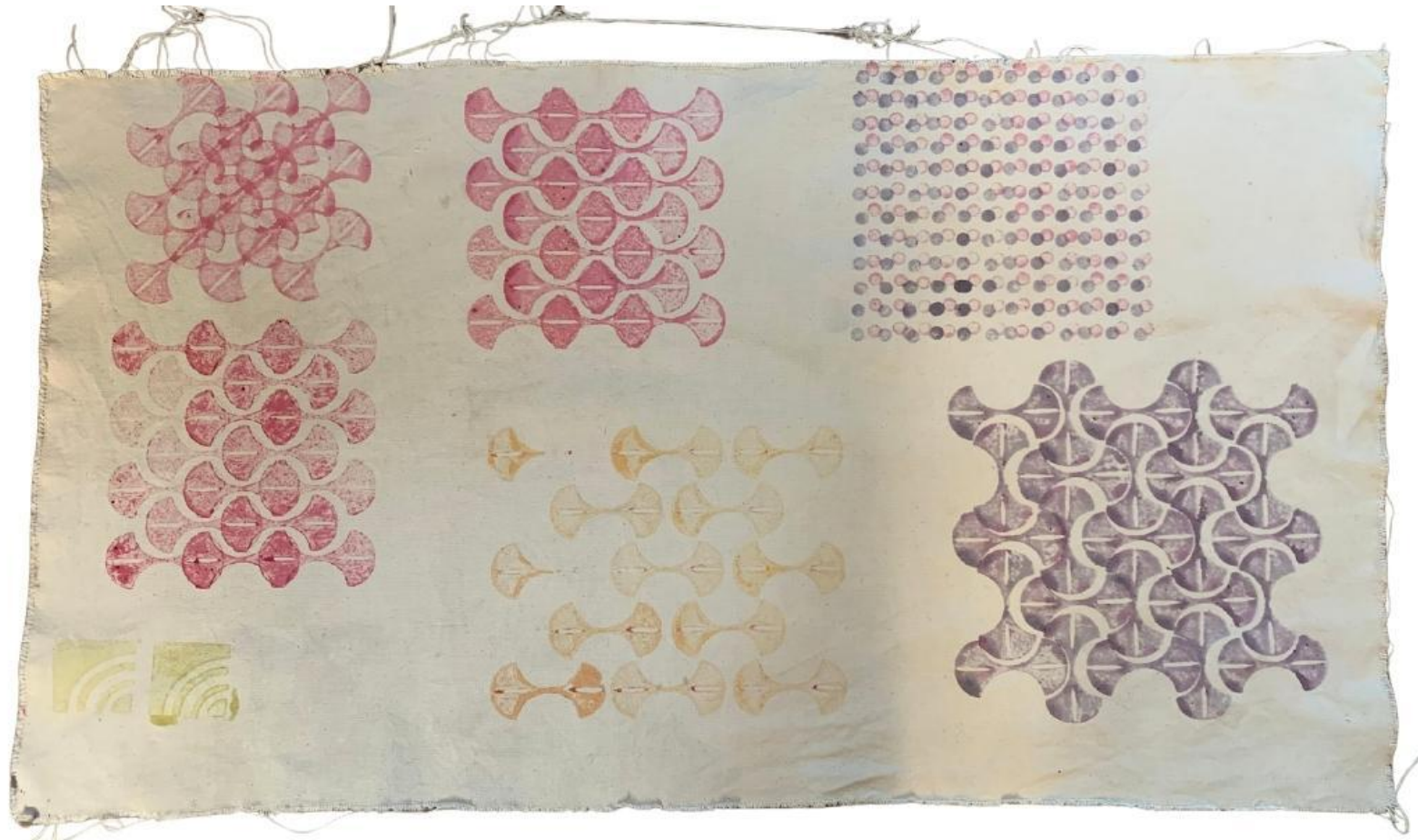


: IRON, Cochineal, Alginate & Water

\*stamp manually turned 90° \*

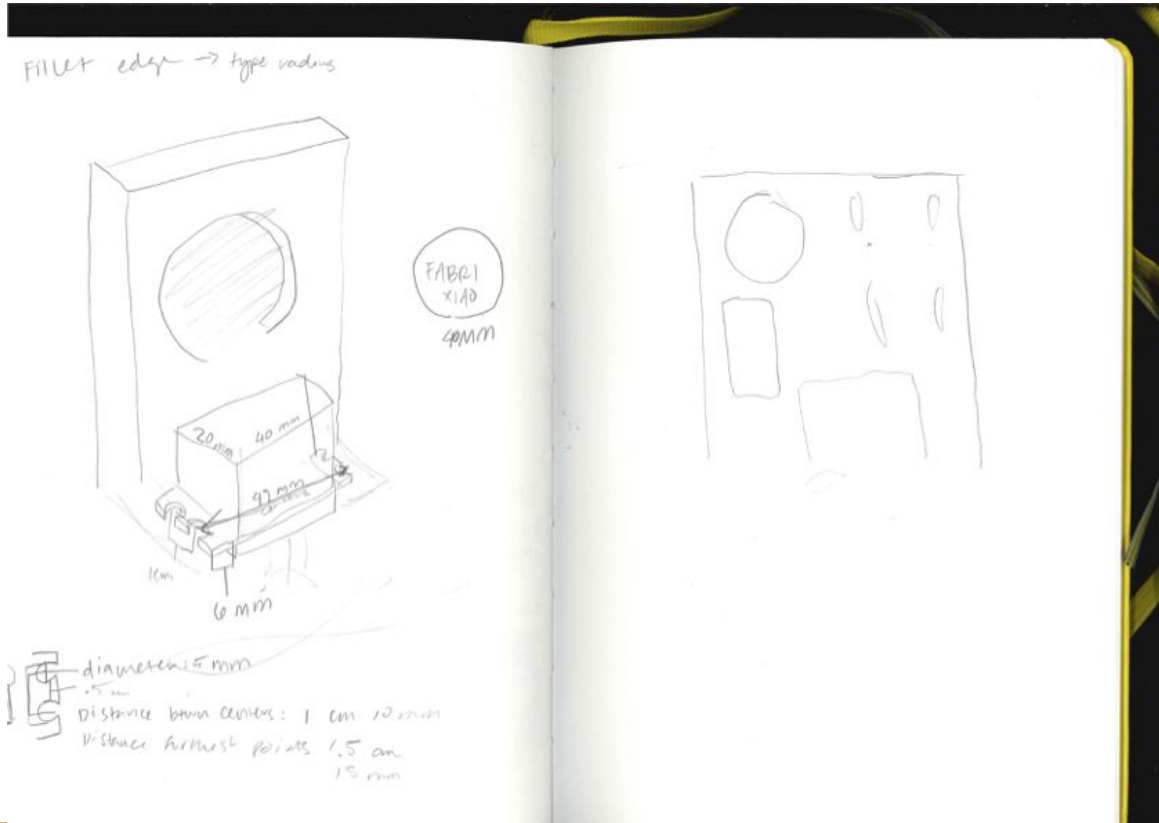


Result



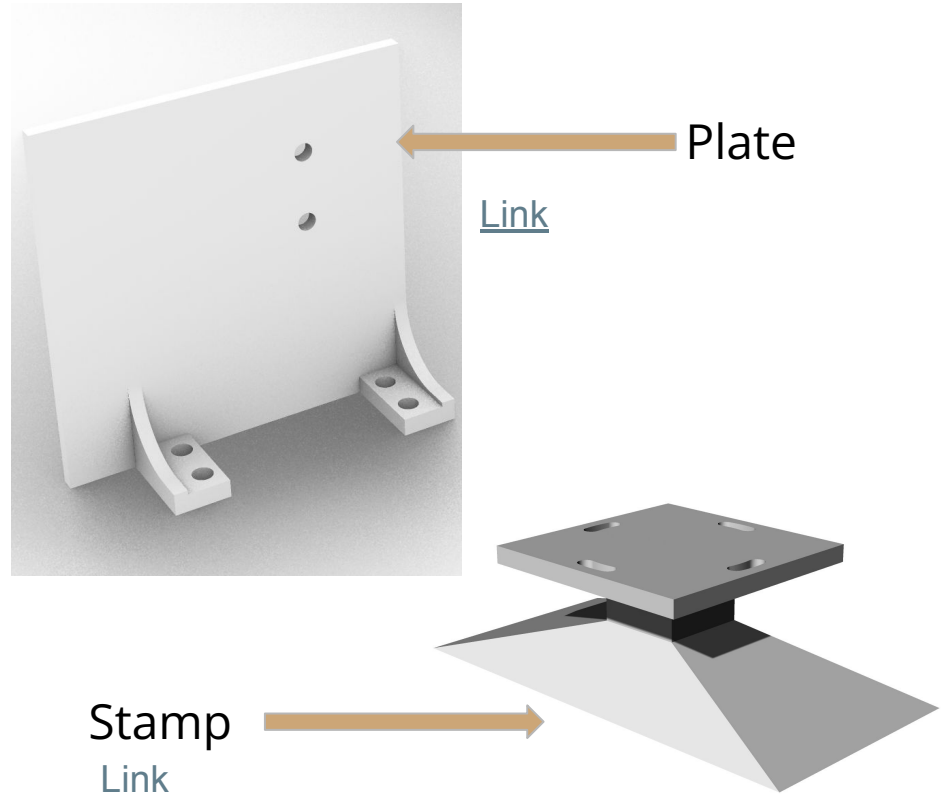
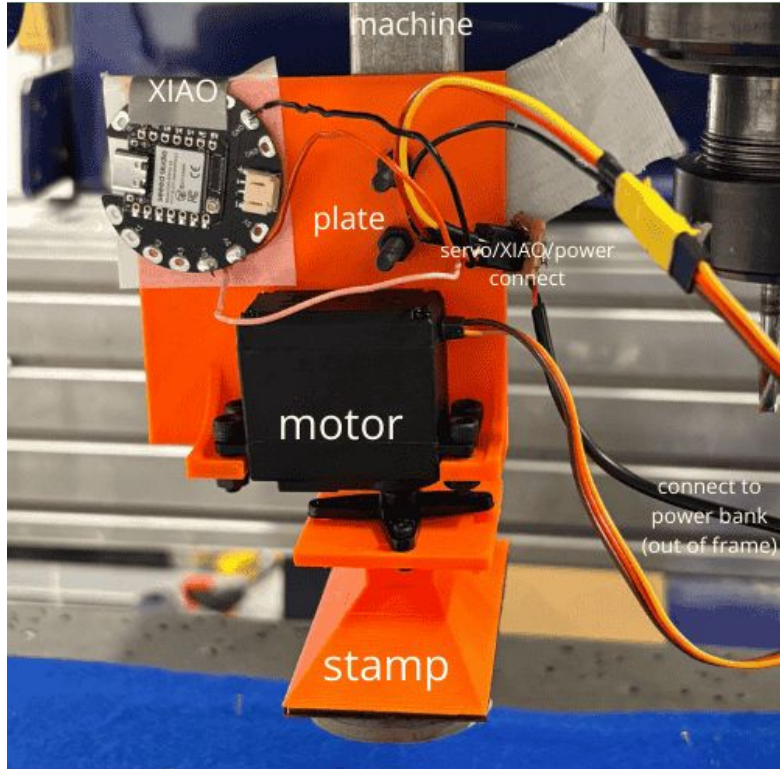


# 3D Printed Tool Parts: Rotating Stamper



- ❖ Must include space for servo motor and electronics
- ❖ 2 3D-printed parts: the motor holder and the stamp holder
- ❖ Center the motor attachment with machine mount

# 3D Printed Tool Parts: Rotating Stamper



# Rotating Stamper Coding

- ❖ Our idea was to create a pattern tool that could rotate. To achieve this, we planned to attach a servo motor to the stamper and use a button to change its rotation whenever needed.
- ❖ We used two ESP32 boards. One board, attached to the stamper, controlled the servo motor, while the second board was connected to the button input and sent a signal to the first board each time the button was pressed.

## Tools

- ❖ ESP32S3
- ❖ ESP32C3
- ❖ FabriXiao
- ❖ Wires
- ❖ Jumper wires
- ❖ Prototyping Board
- ❖ Power bank
- ❖ USB-C cable
- ❖ Servo Motor
- ❖ Button
- ❖ BreadBoard

# ESP NOW Protocol


**ESP-NOW** is a wireless communication protocol developed by Espressif, the company behind the ESP32 microcontroller MCU.  
[Arduino Tutorials](#)

## Characteristics


- ❖ **Wireless:** devices can send data to each other without the time-consuming process of pairing or joining a network.
- ❖ **Low Latency:** The data transmission is nearly instant.
- ❖ **Low Power Consumption:** since it is not connected to the internet, the ESP can go to sleep, wake up to send a message and go to sleep again
- ❖ **Range:** 220 m ca
- ❖ **Optional Encryption:** ESP-NOW supports AES-128 encryption, so your data isn't just flying around unprotected
- ❖ **Supports Many Peers:** You can communicate with up to 20 peers (unencrypted) or 10 peers (encrypted) from a single ESP device.
- ❖ **Can Work Alongside WiFi:** So, one device can collect data via ESP-NOW from a bunch of other ESPs, then upload it to the internet over Wi-Fi



## Architecture

- ❖ Any ESP can be a **client** or a **server**
- ❖ **Peer-to-peer Communication:** you can have one ESP32 sending data to another, one device sending data to multiple receivers or several ESPs all sending data to a central node
- ❖ **MAC address:** Every ESP board has a unique MAC address, and ESP-NOW uses these to route messages.
- ❖ Code to upload to find the MAC address 

Check the serial monitor, it will show something like:

80:B5:4E:C3:6A:64 

Rui Santos & Sara Santos - Random Nerd Tutorials

Complete project details at  
<https://RandomNerdTutorials.com/get-change-esp32-esp8266-mac-address-arduino/>

Permission is hereby granted , free of charge , to any person obtaining a copy of [this](#) software [and](#) associated documentation files .

The above copyright notice [and this](#) permission notice shall be included in all copies [or](#) substantial portions of the Software .

```
*/#include <WiFi.h>#include <esp_wifi.h>void readMacAddress () {uint8_t  
baseMac[6];esp_err_t ret = esp_wifi_get_mac (WIFI_IF_STA, baseMac);if (ret ==  
ESP_OK) {
```

```
Serial.printf ("%02x:%02x:%02x:%02x:%02x:%02x \n",
```

```
baseMac[0], baseMac[1], baseMac[2],
```

```
baseMac[3], baseMac[4], baseMac[5]);} else {
```

```
Serial.println ("Failed to read MAC address" );}void setup () {
```

```
Serial.begin (115200);
```

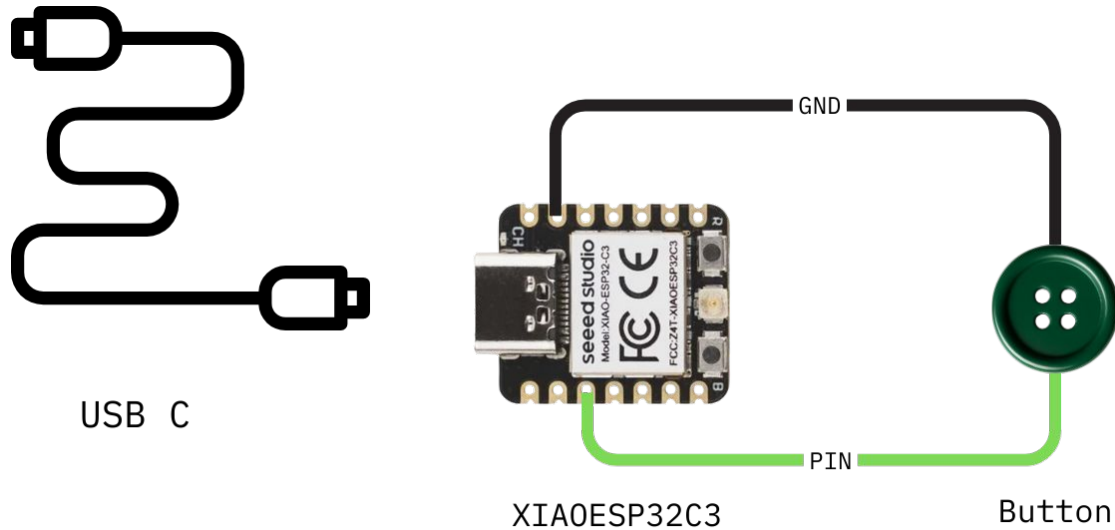
```
WiFi.mode (WIFI_STA);
```

```
WiFi.STA.begin ();
```

```
Serial.print ("[DEFAULT] ESP32 Board MAC Address: " );readMacAddress ();}void  
loop () {}
```

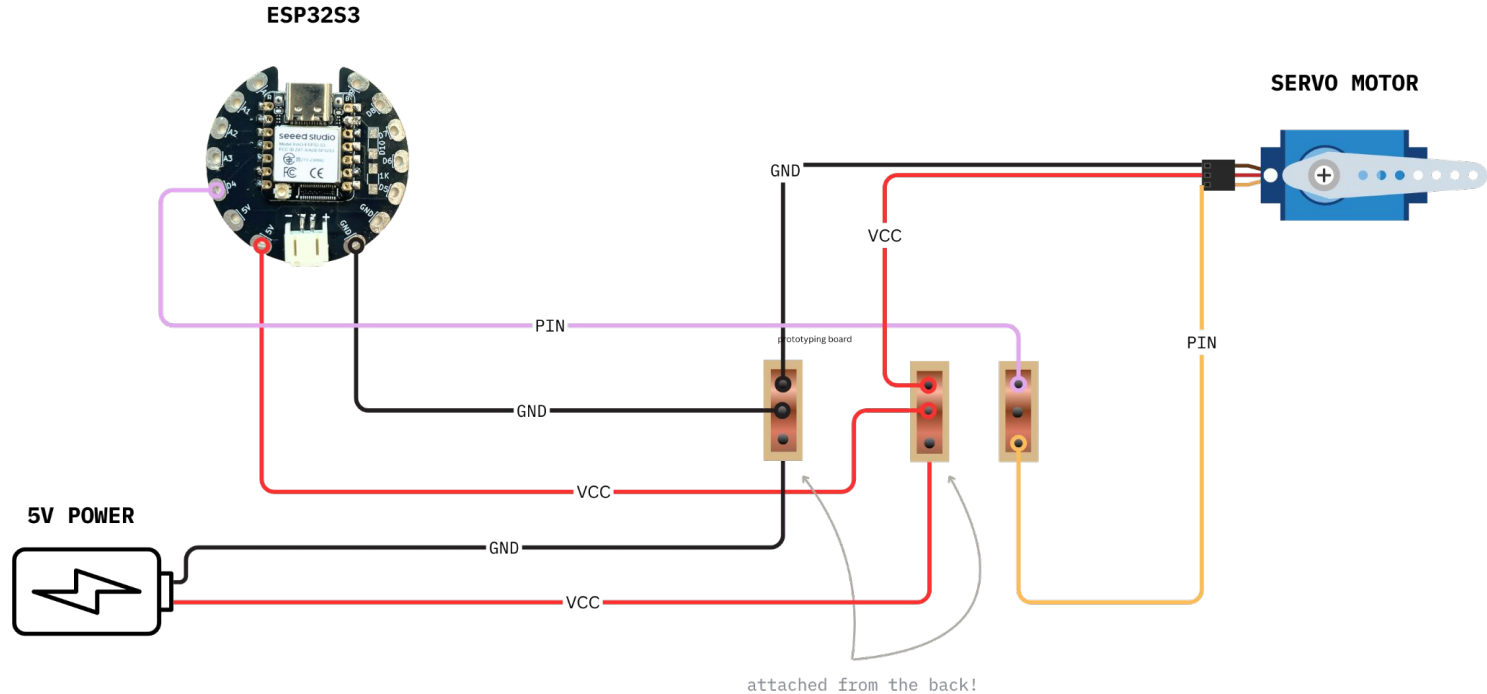
## Diagram Client

ESP32C3 (client) is connected to the button and give the input to the ESP32S3 (server)



# Diagram Server

ESP32S3 (server) is connected to servo motor and attached to the head tool. It receives input from the ESP32S3 (server)



# Client Code

```
#include <esp_now.h>
#include <WiFi.h>

#define BUTTON_PIN 20 // Change to the pin your button is connected to

uint8_t receiverMac[] = {0x10, 0x20, 0xba, 0x76, 0xe1, 0xec}; // Replace
with your S3 MAC

typedef struct {
    int angle;
} Message;

Message outgoingData;
bool lastBtn = HIGH;
int currentAngle = 0;

void onSent(const wifi_tx_info_t *info, esp_now_send_status_t status) {
    Serial.print("Send status: ");
    Serial.println(status == ESP_NOW_SEND_SUCCESS? "OK" : "FAIL");
}

void setup() {
    Serial.begin(115200);
    pinMode(BUTTON_PIN, INPUT_PULLUP); // use internal pull-up resistor

    WiFi.mode(WIFI_STA);

    if (esp_now_init() != ESP_OK) {
        Serial.println("ESP-NOW init failed");
        return;
    }

    esp_now_register_send_cb(onSent);
```

continues here

```
esp_now_peer_info_t peer= {};
memcpy(peer.peer_addr, receiverMac, 6);
peer.channel = 0;
peer.encrypt = false;

if (esp_now_add_peer(&peer) != ESP_OK) {
    Serial.println("Failed to add peer");
    return;
}

Serial.println("C3 Sender ready");
}

void loop() {
    bool btn = digitalRead(BUTTON_PIN);

    // Detect button press (falling edge)
    if (btn == LOW && lastBtn == HIGH) {
        delay(50); // simple debounce

        // Toggle angle
        currentAngle = (currentAngle == 0) ? 90 : 0;
        outgoingData.angle = currentAngle;

        Serial.print("Button pressed → sending angle: ");
        Serial.println(currentAngle);

        esp_now_send(receiverMac, (uint8_t*)&outgoingData,
            sizeof(outgoingData));

        delay(1000); // to avoid double measurements
    }

    lastBtn = btn;
}
```



If the **ESP32S3** is on you should see this in your **Serial Monitor**

```
17:47:33.630 -> Send status: OK
17:47:34.932 -> Button pressed → sending angle:
90
17:47:34.932 -> Send status: OK
17:47:46.188 -> Button pressed → sending angle: 0
17:47:46.188 -> Send status: OK
```



\*A3(GPIO5) - Uses ADC2, which may become inoperative due to false sampling signals. For reliable analog reads, use ADC1 instead. Refer to the ESP32-C3 datasheet.

In this part of the code where the button pin is defined, I used the actual GPIO number instead of `digitalPin 7`. The button wasn't responding when I set `BUTTON_PIN` to 7, but it started working correctly once I used the corresponding GPIO number for the ESP32-C3.

# Server Code

```
#include <esp_now.h>
#include <WiFi.h>
#include <ESP32Servo.h>
Servo servo;
int currentAngle = 0;

typedef struct {
    int angle;
} Message;
Message incomingData;

void onReceive(const esp_now_recv_info_t
*info, const uint8_t *data, int len) {
    memcpy(&incomingData, data,
sizeof(incomingData));
    Serial.print("Received angle: ");
    Serial.println(incomingData.angle);

    servo.write(incomingData.angle);
    currentAngle = incomingData.angle;
}
```

continues here

```
void setup() {
    Serial.begin(115200);
    servo.attach(5); // D5 pin
    servo.write(0);

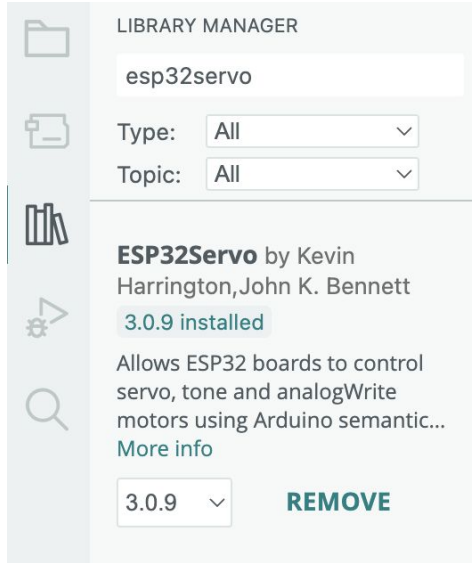
    WiFi.mode(WIFI_STA);

    if (esp_now_init() != ESP_OK) {
        Serial.println("ESP-NOW init failed");
        return;
    }

    esp_now_register_recv_cb(onReceive);

    Serial.println("S3 Receiver ready");
}

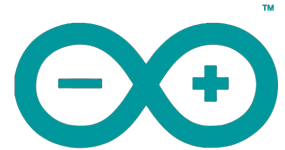
void loop() {}
```



You need to include the **ESP32Servo library**, which needs to be installed. You can do so by navigating:

Library Manager > Type "ESP32Servo" > Install

Arduino sketches: [Client](#) and [Server](#)

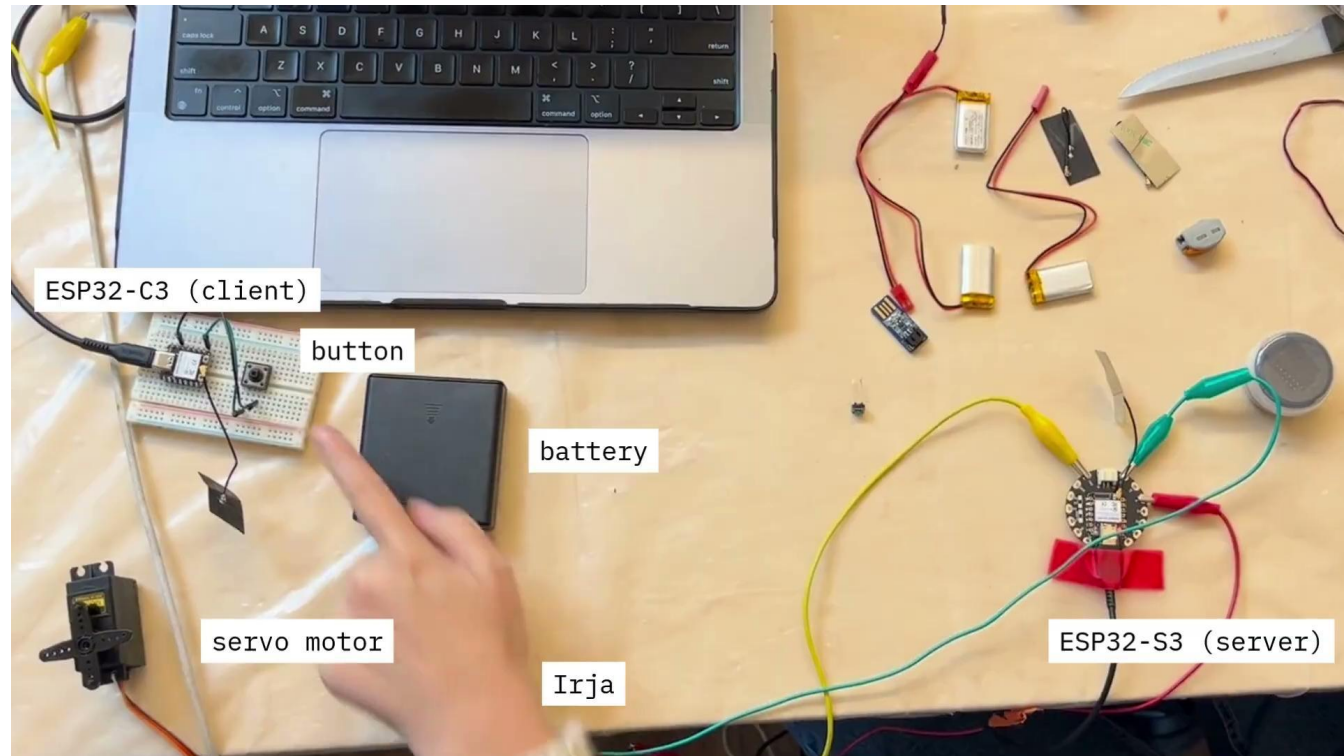


## BoM - Rotating Stamper

Title	Quantity	Notes	Cost	Link
ESP32S3	1		€8.82	<a href="#">kiwieletronics</a>
ESP32C3	1		€8.99	<a href="#">bit and parts</a>
Fabrixiao	1			
Servo MG995	1	10kg 180 graden	€6.99	<a href="#">benselectronics</a>
Wires		with different colours it is easier to distinguish the connections	€1.95	<a href="#">bits and parts</a>
Jumper wires			€0.50	<a href="#">tinytronics</a>
Switch	1		€0.15	<a href="#">tinytronics</a>
Power bank	1		€29.99	<a href="#">mediamarkt</a>
USB C cable	1		€6.79	<a href="#">alleskabels</a>



# Testing



Link [Vimeo](#)

# Powering

The ESP32S3 and the Servo MG995 need to be powered

## ESP32-S3

- Voltage: **5V** ( USB) or **3.3V** ( input)
- Current: ~150–500 mA

## MG995 Servo

- Voltage: **4.8V – 7.2V**
- Current:
  - **Idle:** ~10 mA
  - **Moving (no load):** ~500–900 mA
  - **Stall (under load): 1.5–2.5 A** spikes



This means that the servo has current spikes, making it quite unstable

# Troubleshooting

Ideally, we could have added a **capacitor** to store electrical energy and provide extra current during servo spikes.

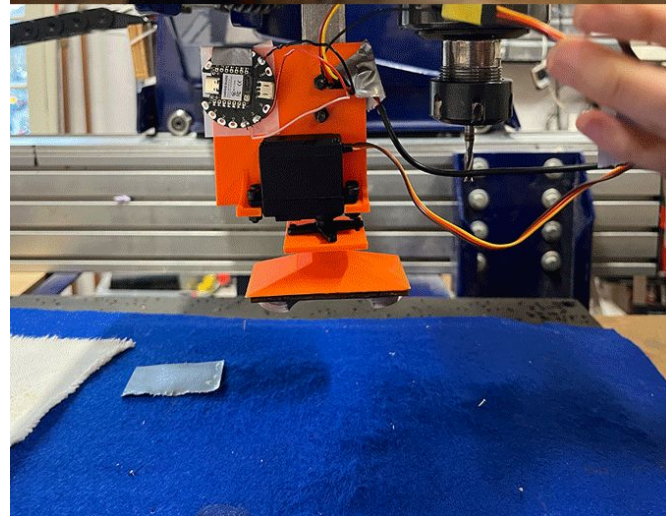
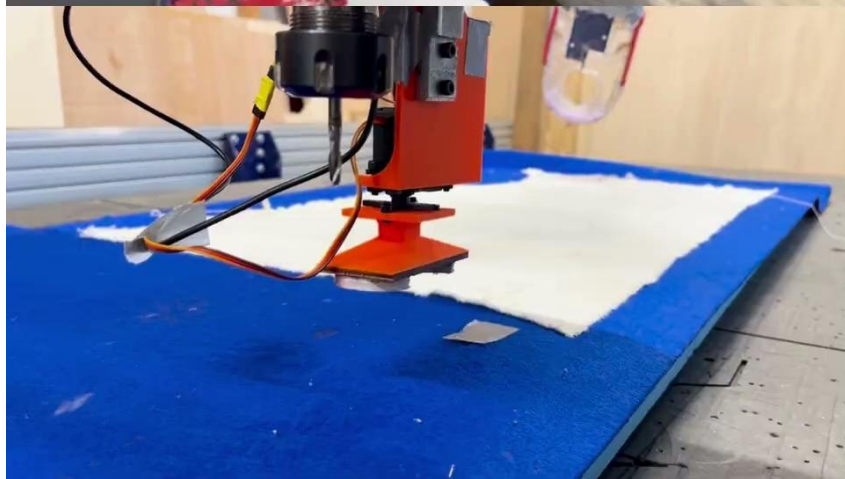
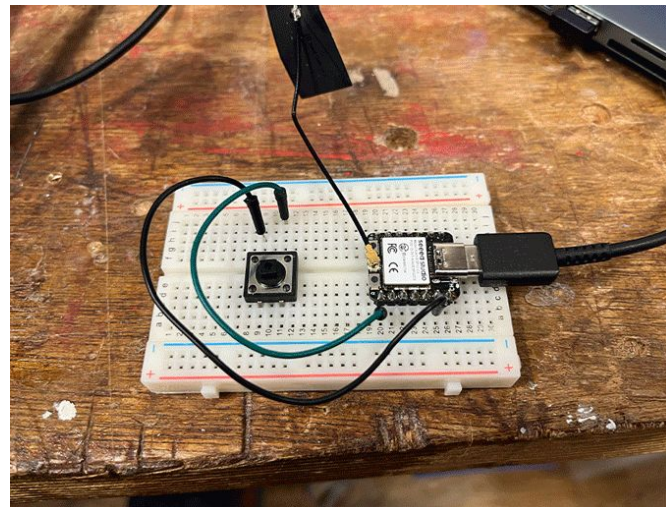
However, we couldn't find a capacitor in the lab that was suitable for our setup.

We tested a 6V power supply (for the servo) and a 3.7V battery (for the board) but the board was not turning on

We tested a 6V power supply connected to both the board and the servo, but we fried the board

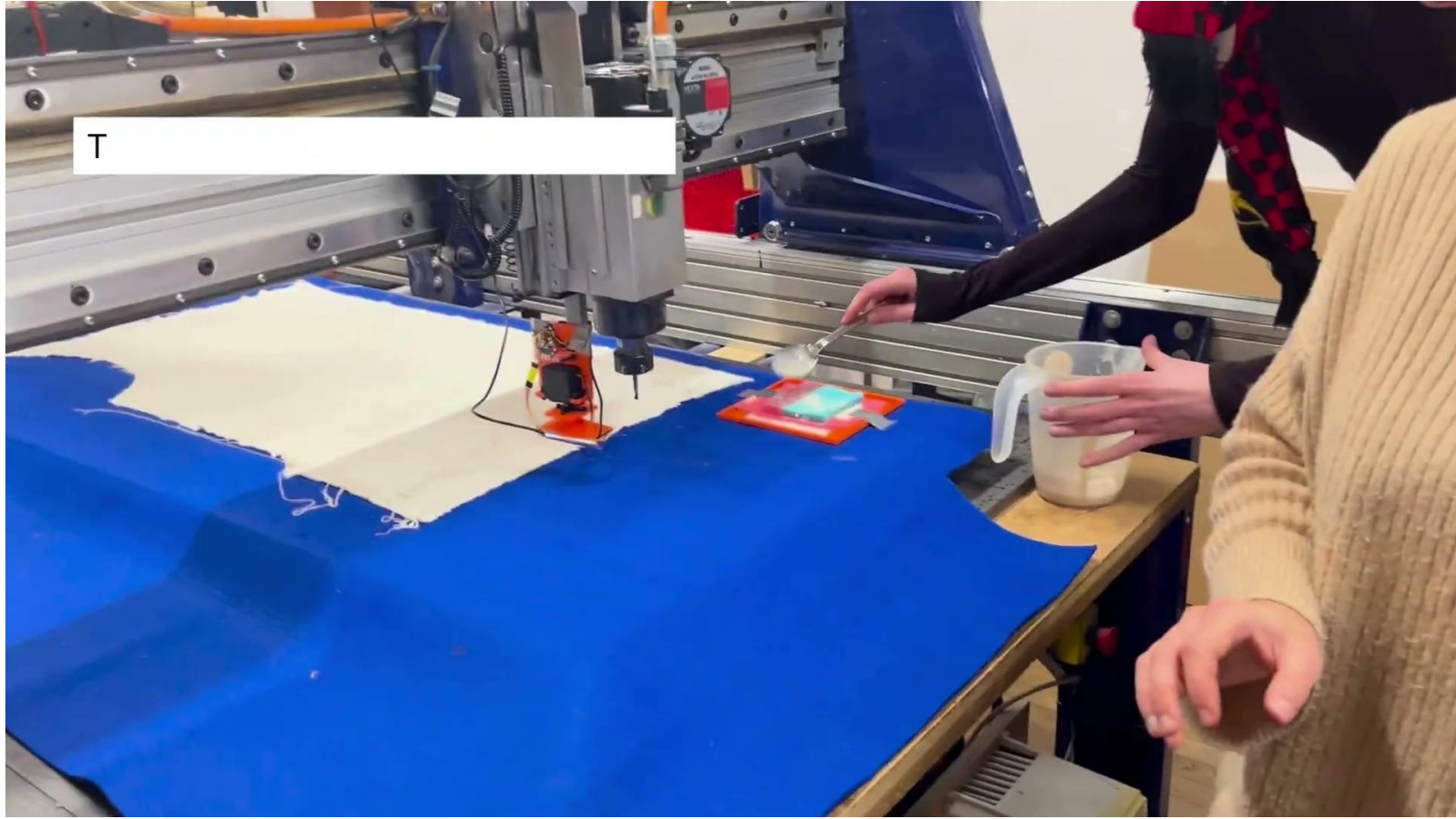
We tried using a LDO regulator with a 9V battery, but we noticed not enough current was flowing to the board

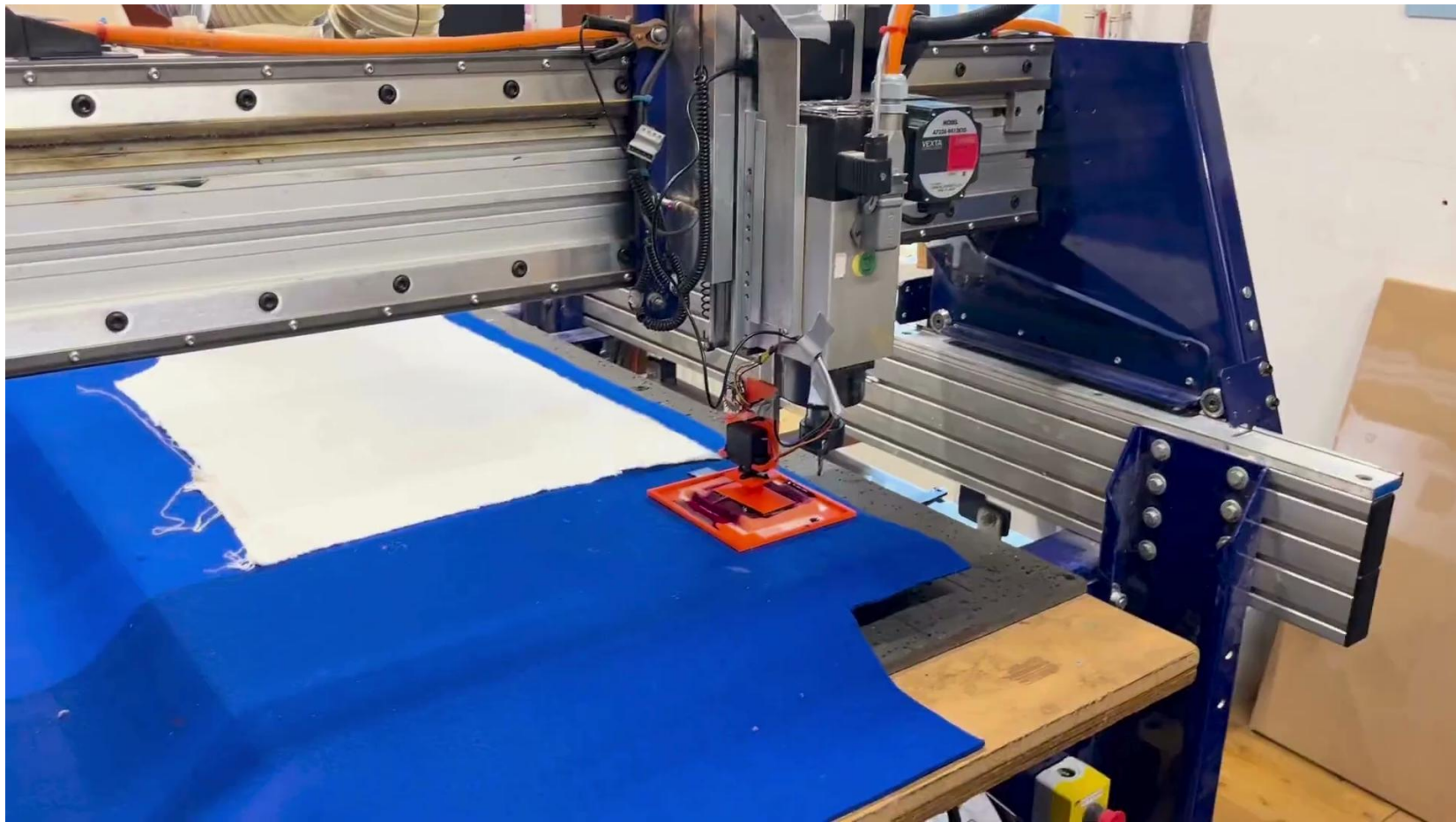
We ended up powering both the servo and battery with a power bank





# Testing the rotating stamper





Printing

[Link Video](#)



