

Sci-Fi 3D Printed Mask With Transparent Display



by Mukesh_Sankhla

This Mask is a 3D Printed Winter Soldier mask from Capital America Civil War, I have modified the mask by adding two Transparent OLED Displays as the eyes of the Mask. This are 128x64 pixels OLED see through displays from DFRobot.com which I am using for showing few animations. Since the display are transparent I can see through them even when the pixels are on.

The transparent display are two way display, means the image is visible on the both side but inverted at one side.

The aim of this project is to make others see the animating eyes, even the image is visible on both side the person wearing the mask cannot see the display content since the displays is too close to the eyes to focus.

To switch between different animations and texts I have used Blynk app.

Thank You NextPCB:

This project is successfully completed because of the help and support from NextPCB. NextPCB is one of the most experienced PCB manufacturers in Global, has specialized in the PCB and assembly industry for over 15 years. Not only could NextPCB provide the most innovative printed circuit boards and assembly technologies in the highest quality standards, the fastest delivery turnaround as fast as 24 hours.

Guys if you have a PCB project, please visit their website and get exciting discounts and coupons.

Only 0\$ for 5-10pcs PCB Prototypes Nextpcb.com/pcbprototype

4-layer PCB price reduction up to 40%: Nextpcb.com/4-layer-pcb

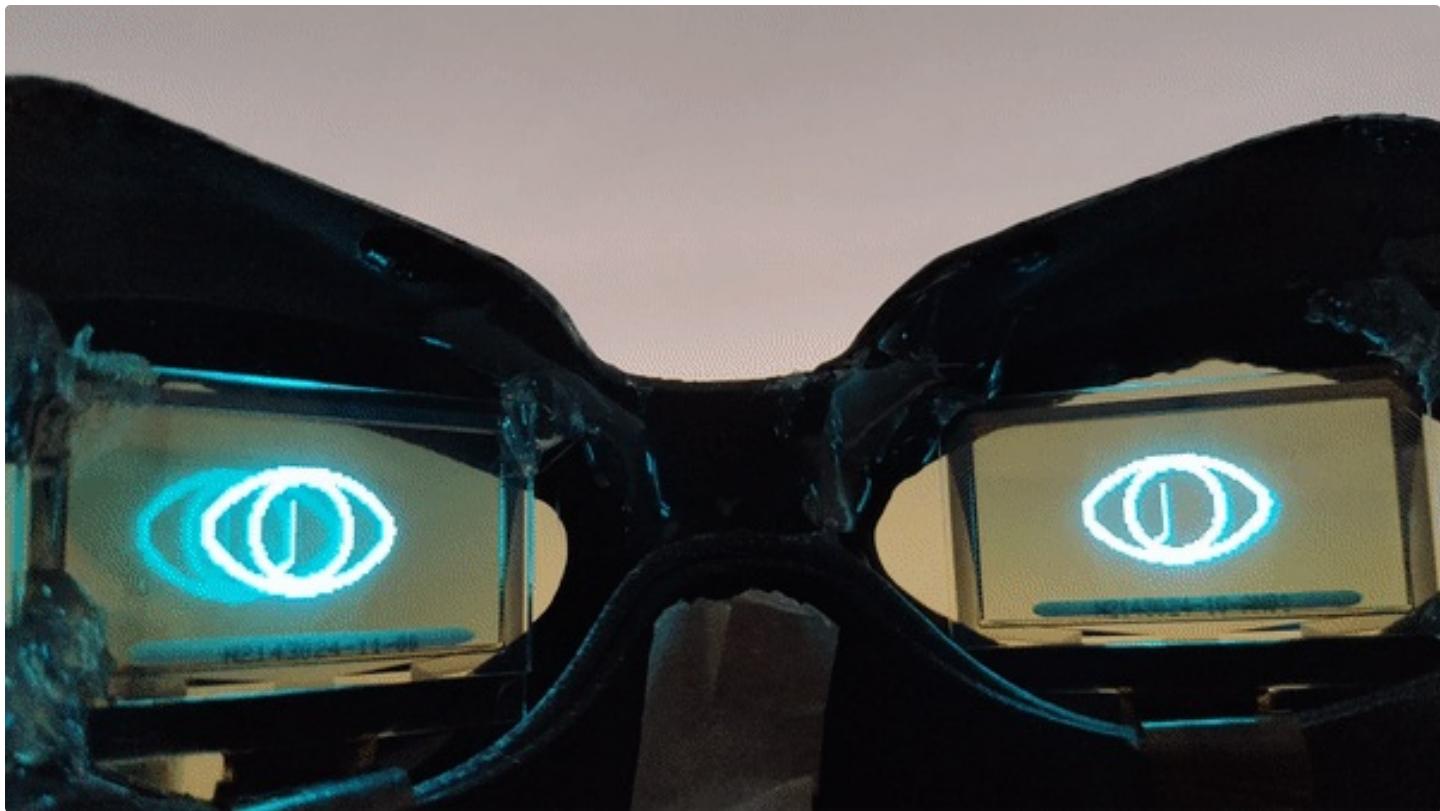
Register and get \$100 from NextPCB: Nextpcb.com/coupon

Supplies:

- 2x Transparent Display
- 1x FireBeetle ESP32
- 1x 3.7 V Battery
- 1x Push Switch
- Spray Paint(Silver, Black)
- 3D Printed Parts

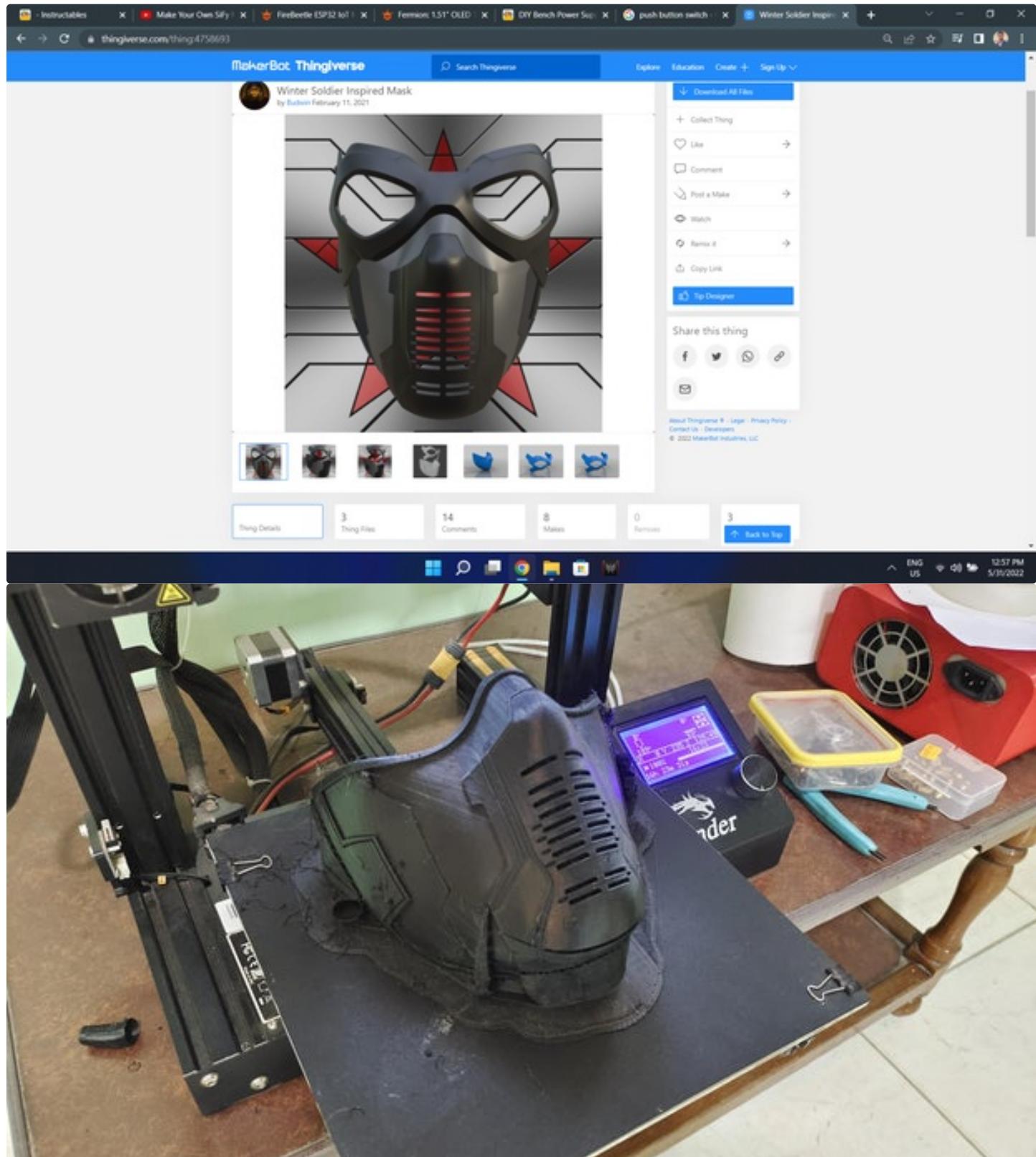


<https://youtu.be/Ej0FG-7REmM>



Step 1: 3D Printing

- Download the [Winter Soldier Mask stl Files](#).
- Open the files in slicing software.
- I have printed the objects with 0.2mm layer height, 20% infill and with tree support enabled.
- After slicing 3D print the objects.
- Remove the support material once prints are complete.



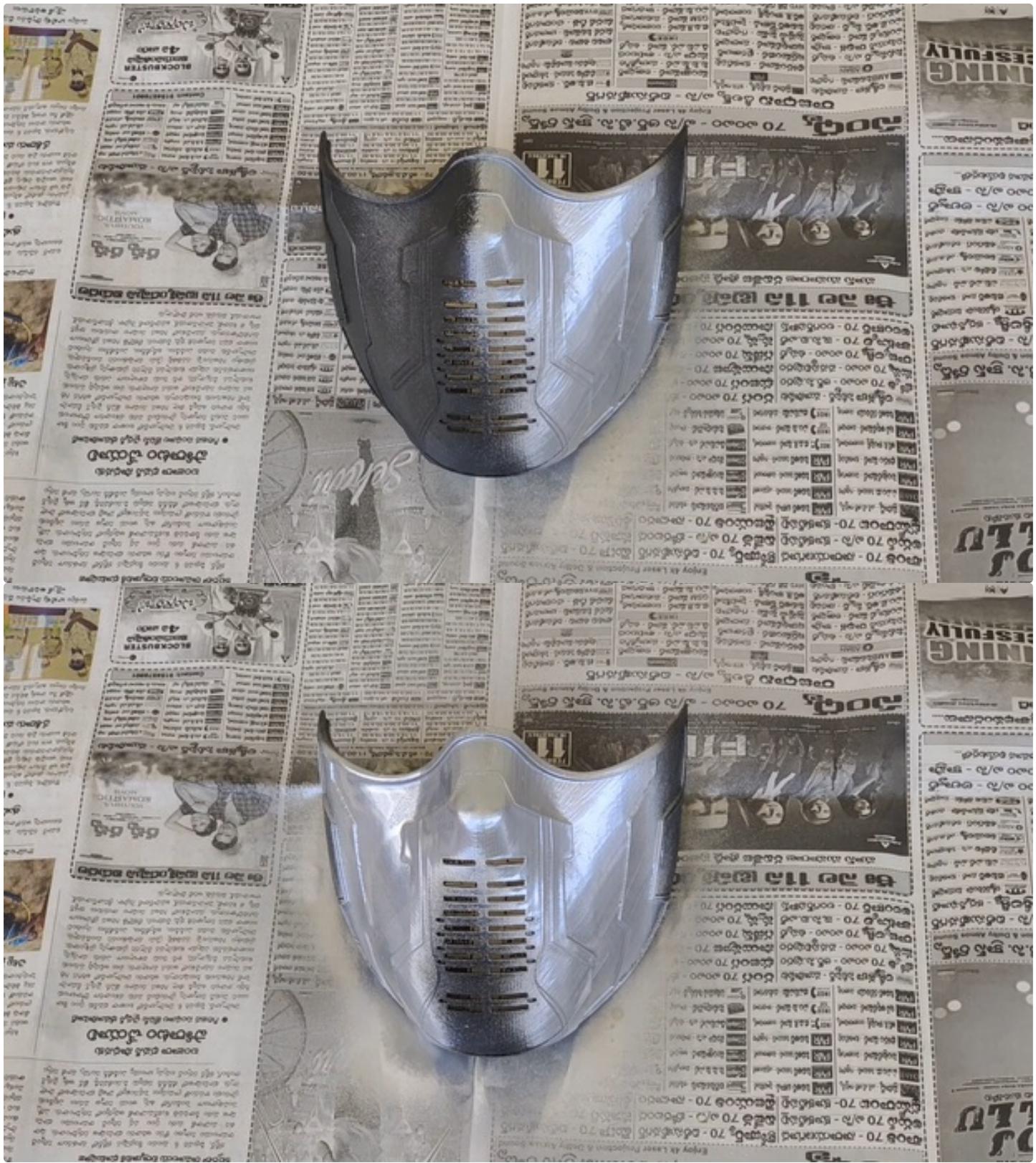
Sci-Fi 3D Printed Mask With Transparent Display: Page 4

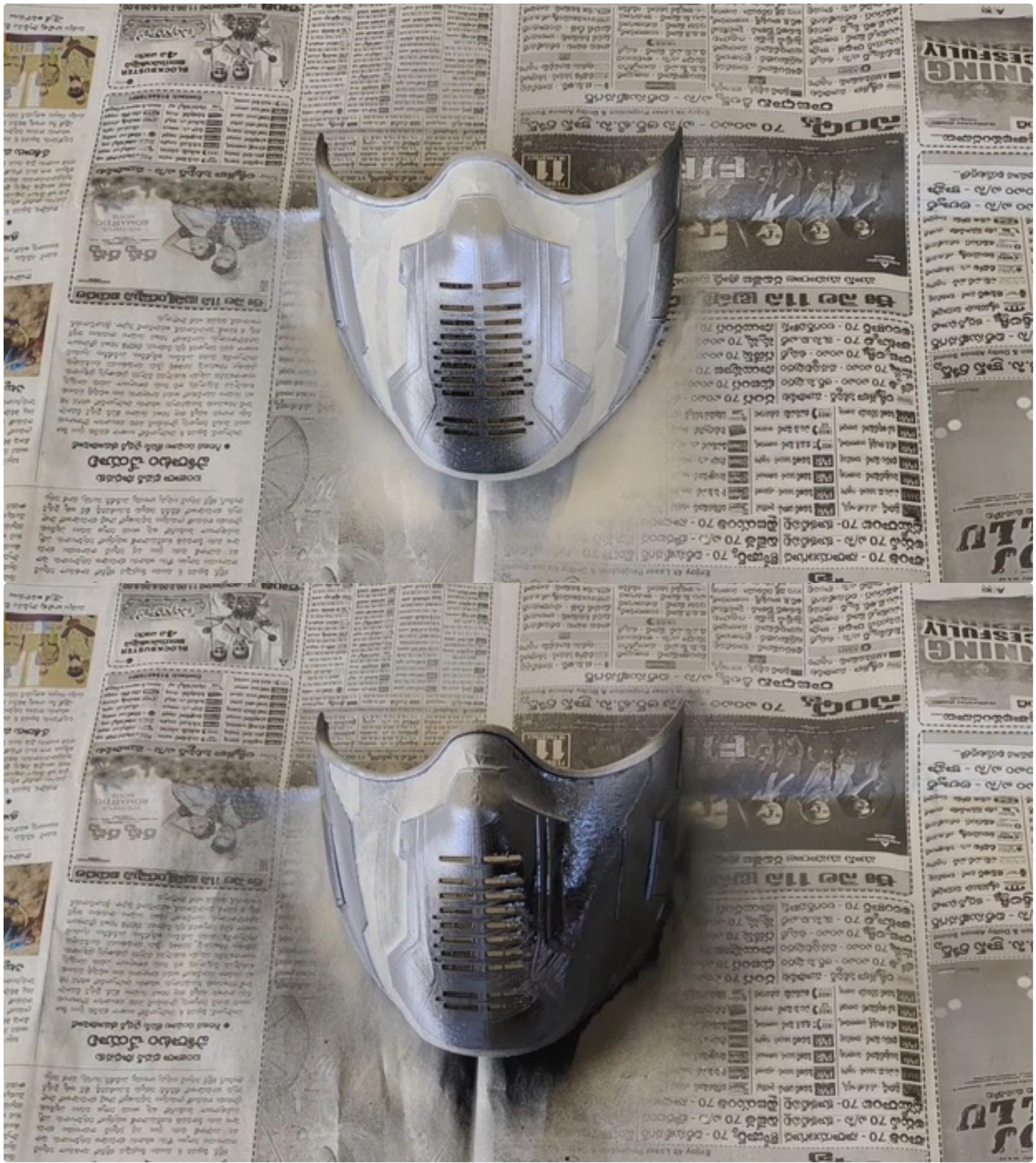


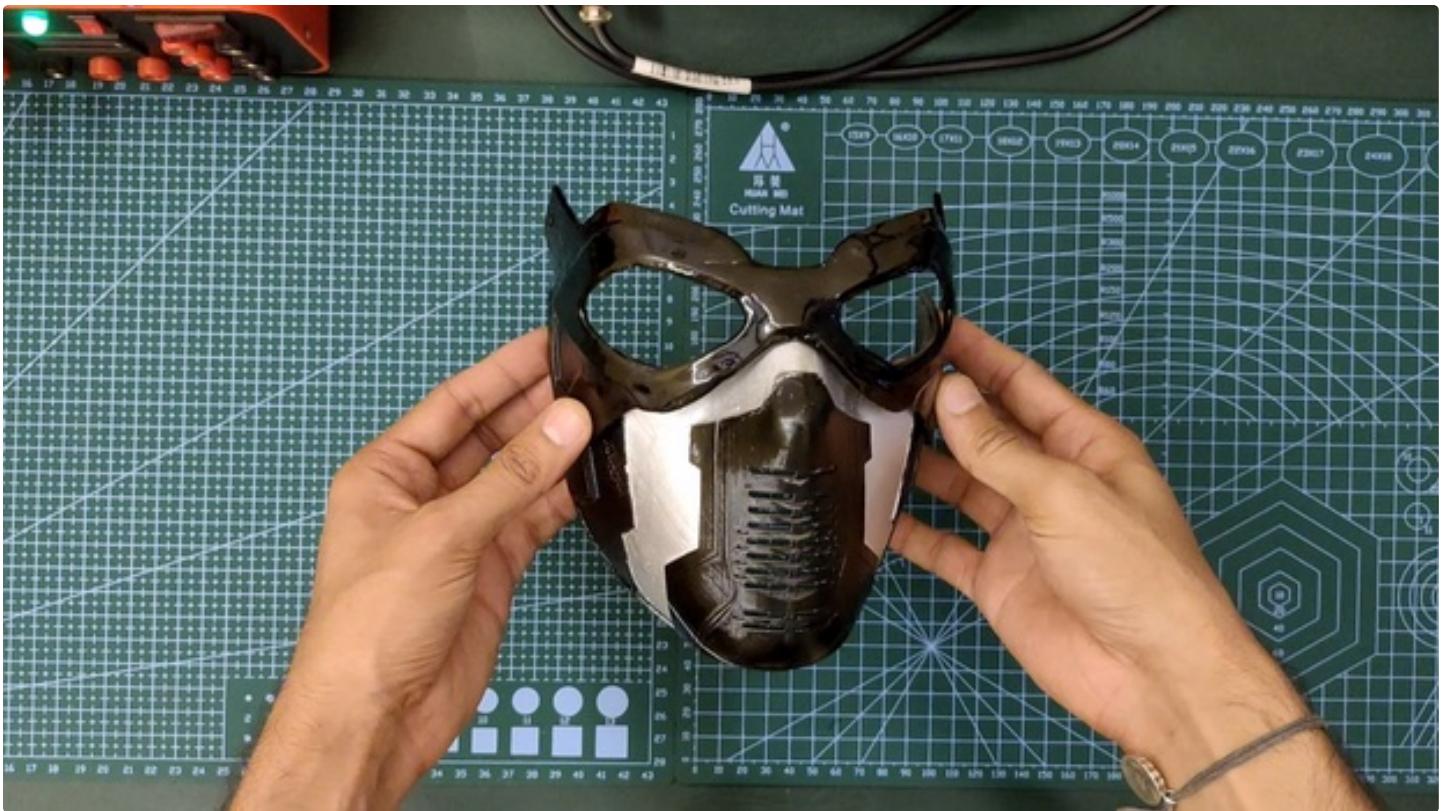


Step 2: Painting

- For painting the mask I am using Silver and Black colored Paint Sprays.
- First I painted the silver part.
- After the silver part is dry I masked it and painted the black.
- I am not using any sanding and smoothening techniques, but if you want then please go for it.
- After the paint is dry glue both the mask objects as one single piece.







Step 3: Eyes

- For the eyes I am using a pair of sunglasses.
- Removed their lenses and glued them to the mask.

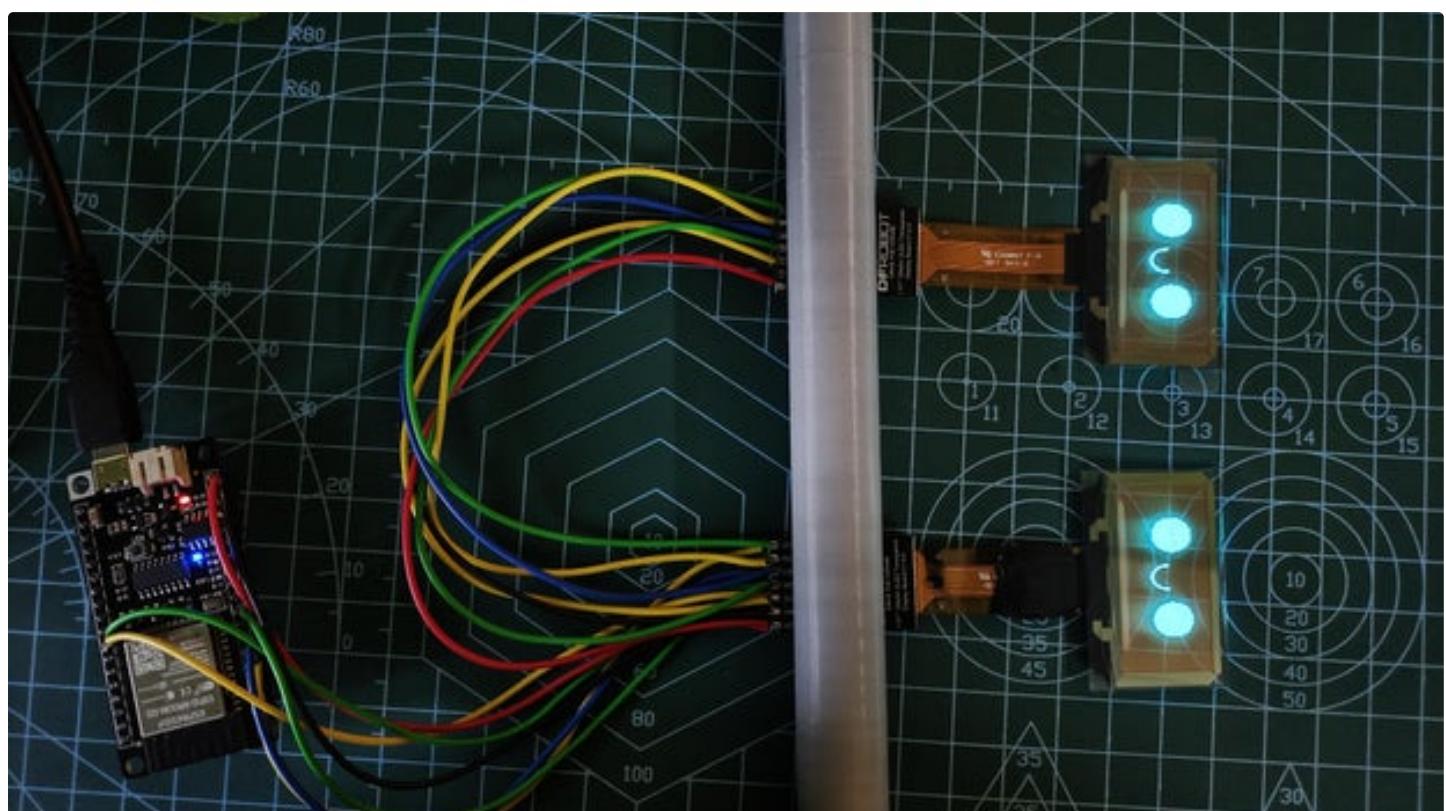
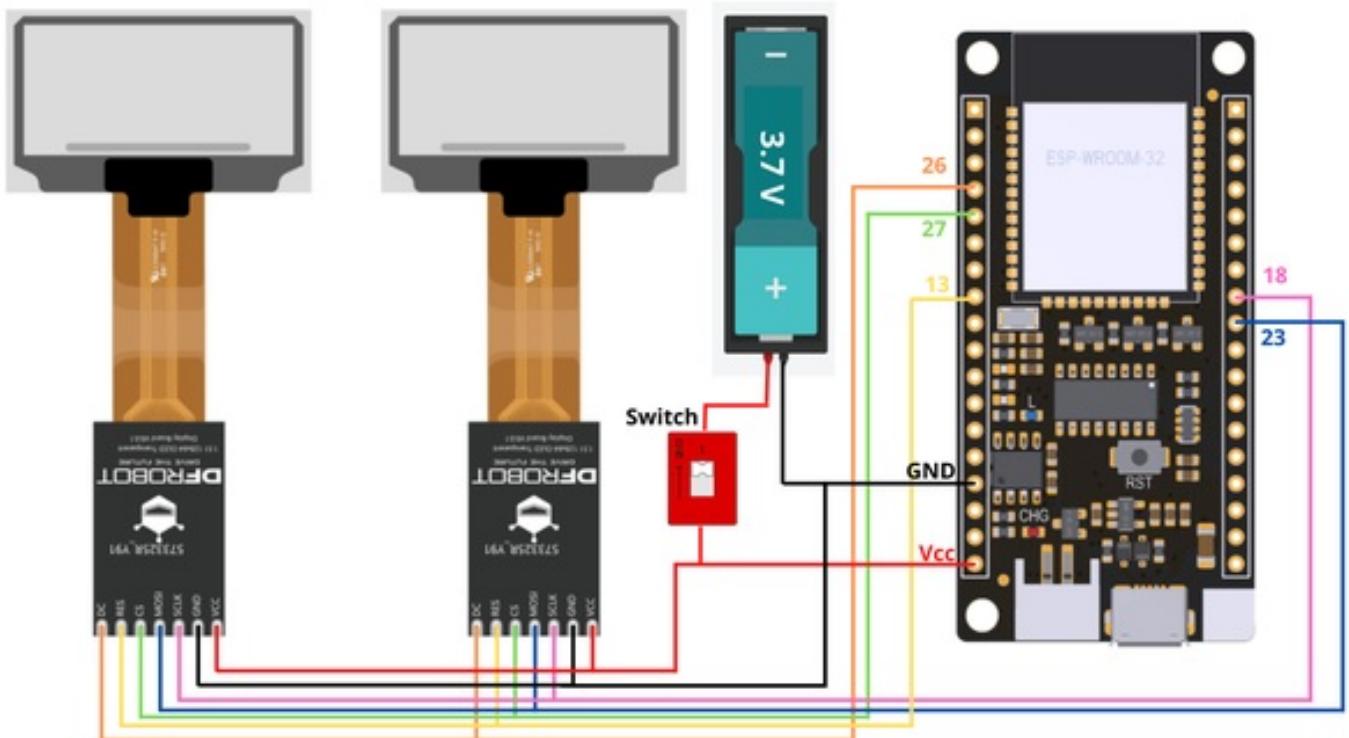


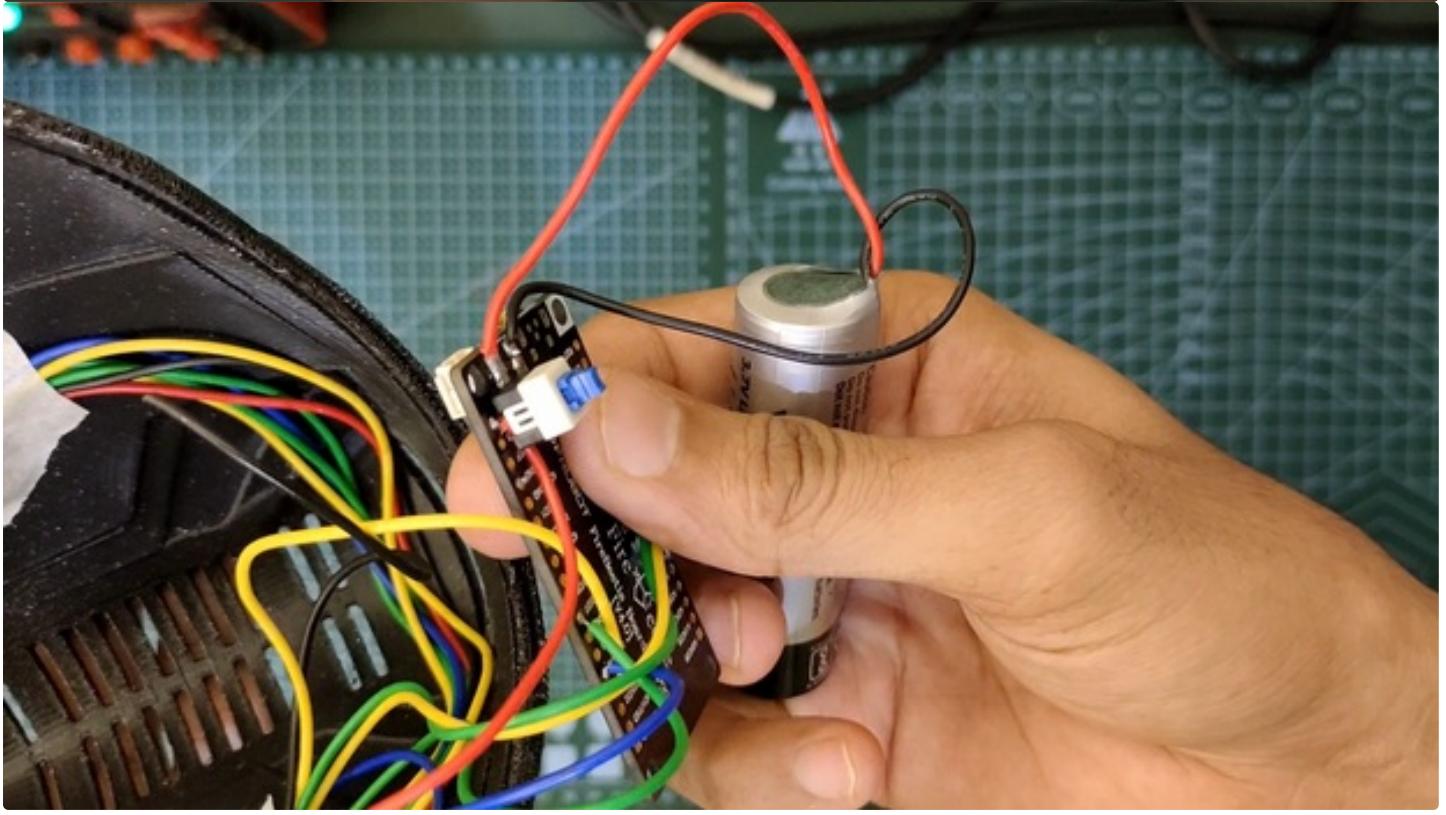
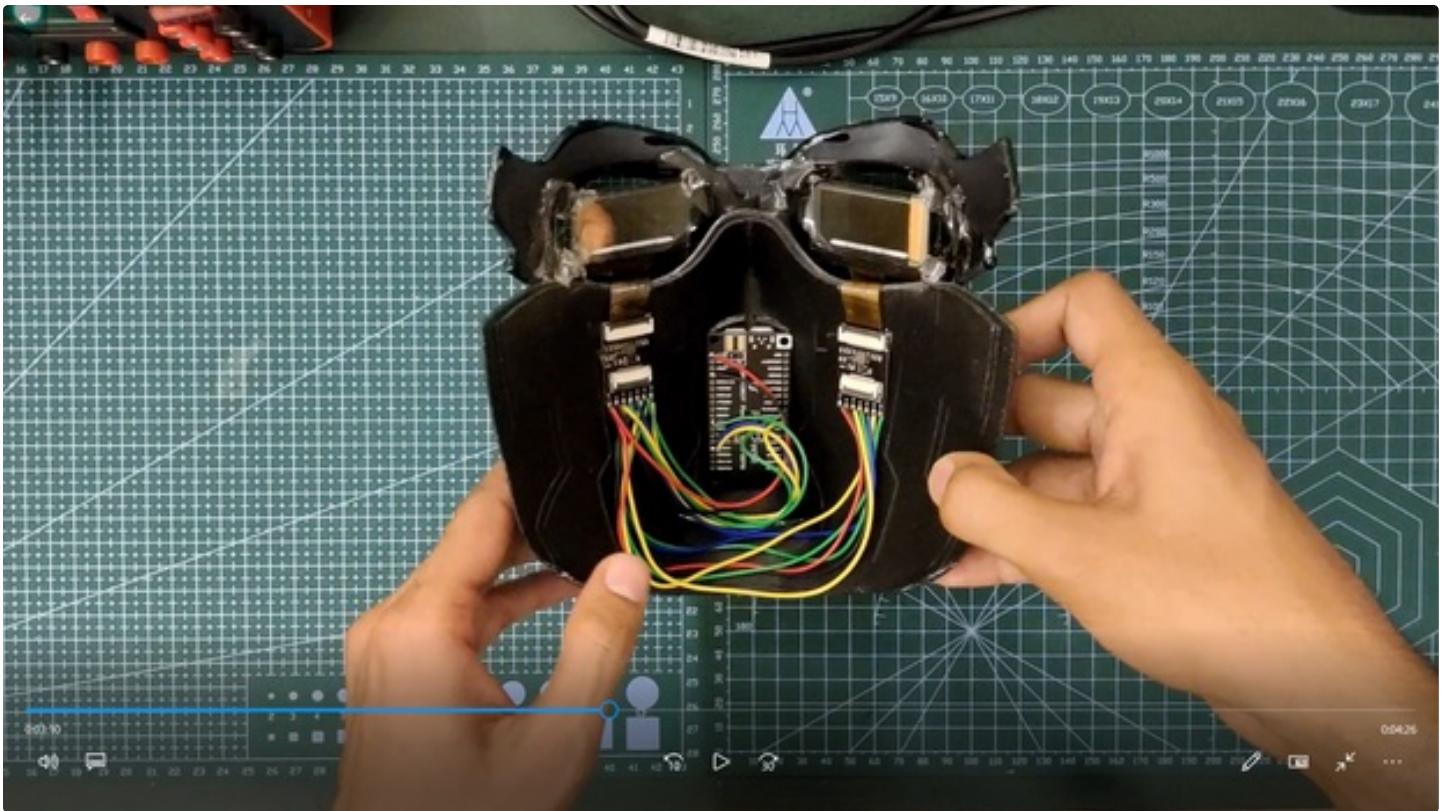
Sci-Fi 3D Printed Mask With Transparent Display: Page 9

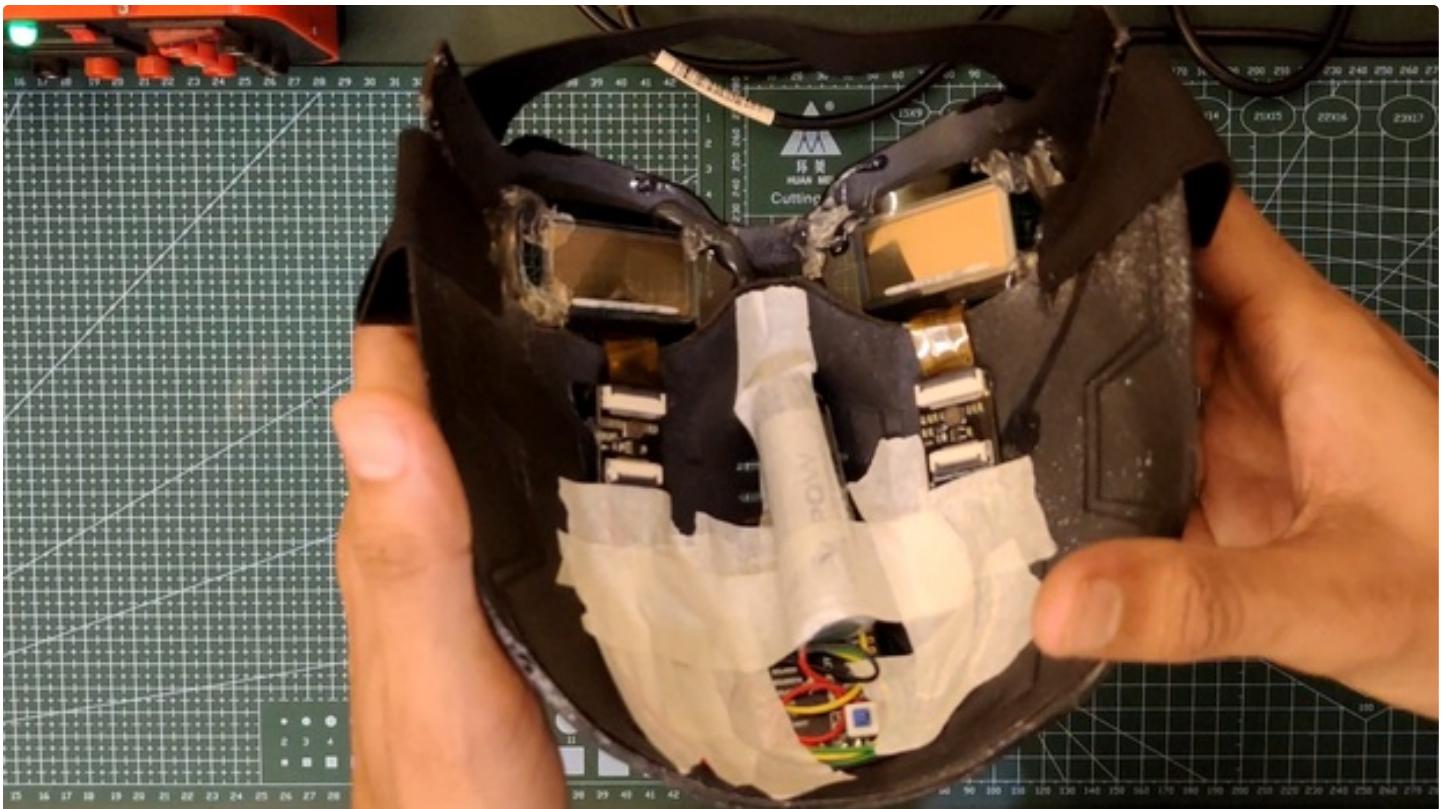


Step 4: Circuit

- Make all the connections as shown in circuit diagram.
- After the connections are made glue and stick all the electronics in the mask.
- I have glued the battery and ESP32 Board inside the mask since I didn't want any wiring coming out of it.

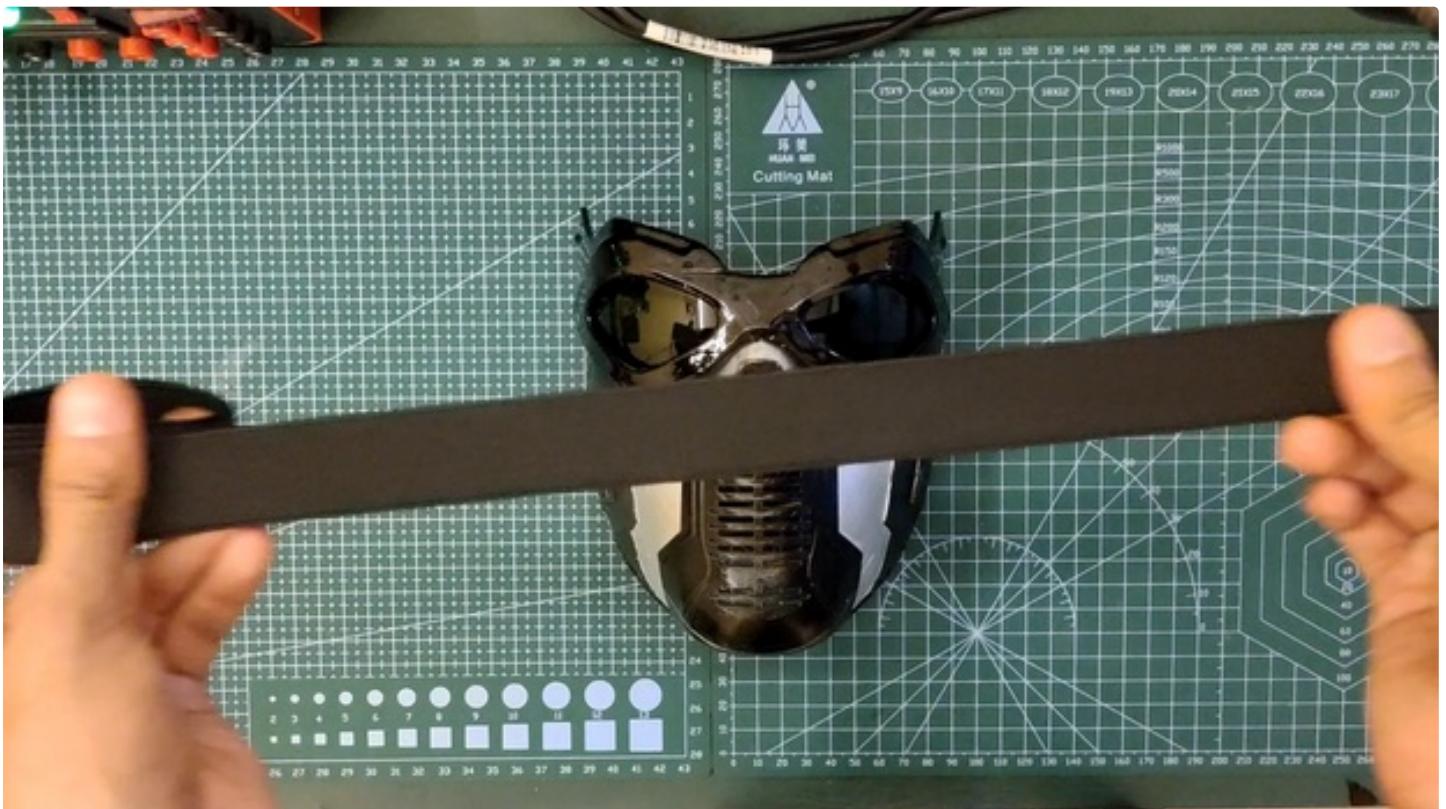


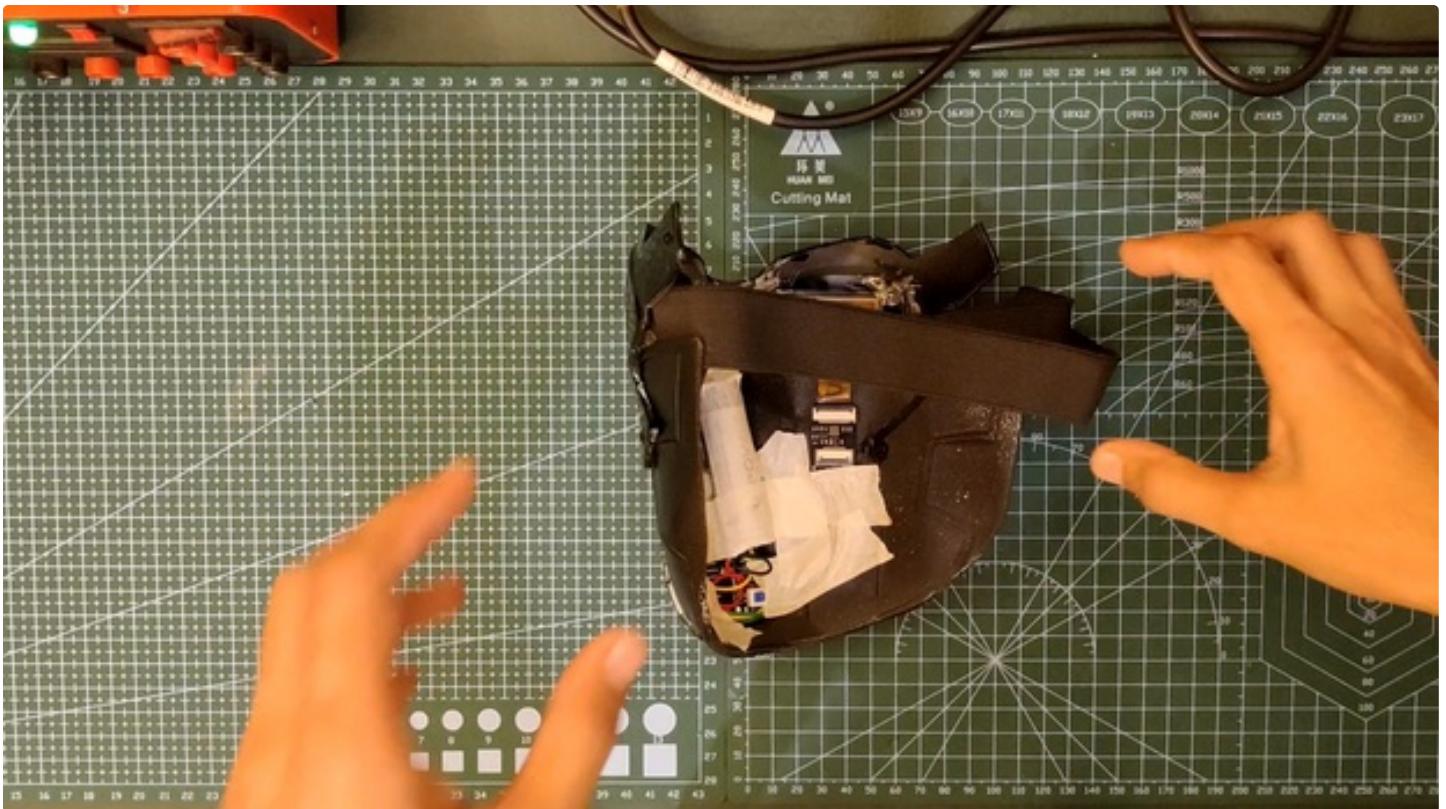




Step 5: Back Support

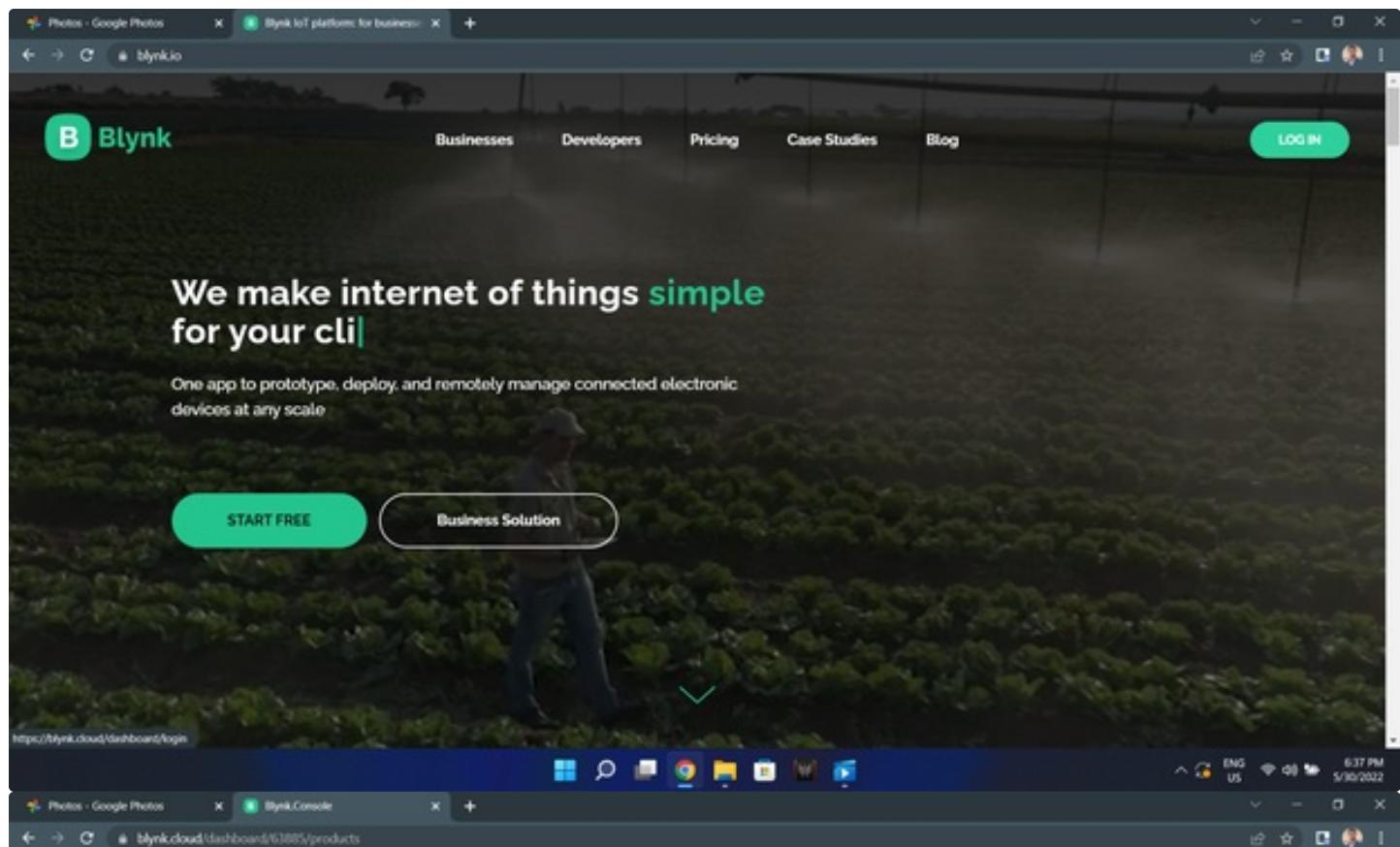
- I am using a 1 inch elastic band for the back support.





Step 6: Blynk Setup

- Go to Blynk.io
- Create a account and logion.
- Go to the templates and create a new template.
- After the template is created copy the Template ID and Name.
- Go to DataStream's and create 3 new data streams(for Animation Number, Text Number, Text Size) as shown in images.
- After creating the DataStream's click on save.



This screenshot shows the Blynk Console Templates page. On the left, there's a sidebar with icons for Home, Templates, Devices, and Help. The main area is titled "Templates" and contains a search bar with placeholder text "Search Templates". Below the search bar are two template cards: "Mask" (1 Device) and "Smart Sense" (1 Device). Each card has a small icon showing a circuit diagram. To the right of the templates is a "New Template" button. At the bottom right, there's a "Region: b1r1 Privacy Policy" link. The URL in the browser address bar is https://blynk.cloud/dashboard/61005/products.

Screenshot of the Blynk Console interface showing the creation and management of device templates.

Create New Template

- NAME:** Mask
- HARDWARE:** ESP32
- CONNECTION TYPE:** WiFi
- DESCRIPTION:** This is my template

Mask (1 Device)

Smart Sense (1 Device)

Info Metadata Datastreams Events Automations Web Dashboard Mobile Dashboard

HARDWARE: ESP32 **CONNECTION TYPE:** WiFi

MANUFACTURER: Smart Sense

OFFLINE IGNORE PERIOD: 0 hrs 0 mins 0 secs

TEMPLATE ID: TMPLCdownCYJE

DESCRIPTION: This is my template

FIRMWARE CONFIGURATION:

```
#define BLYNK_TEMPLATE_ID "TMPLCdownCYJE"
#define BLYNK_DEVICE_NAME "Mask"
```

Template ID and Device Name should be included at the top of your main firmware.

The screenshot shows the Blynk Console interface for managing datastreams. At the top, there are tabs for Info, Metadata, Datastreams (which is selected), Events, Automations, Web Dashboard, and Mobile Dashboard. A search bar labeled "Search datastream" is present, along with a green button for "New Datastream". Below the header, a table lists three datastreams:

	ID	Name	Alias	Color	Pin	Data Type	Units	Is Raw	Min	Max	Actions
1	1	AnimationNumber	AnimationNumber	Blue	V0	Integer		false	0	16	
2	2	MessageNumber	MessageNumber	Dark Blue	V1	Integer		false	0	10	
3	3	TextSize	TextSize	Orange	V2	Integer		false	1	2	

The screenshot shows the "Virtual Pin Datastream" configuration dialog. The "NAME" field contains "AnimationNumber" and the "ALIAS" field also contains "AnimationNumber" with a blue color swatch. Other fields include "PIN" (V0), "DATA TYPE" (Integer), "UNITS" (None), "MIN" (0), "MAX" (16), and "DEFAULT VALUE" (0). A "Save" button is at the bottom right of the dialog.

The screenshot shows the Blynk cloud dashboard interface. At the top, there are tabs for 'Info', 'Metadata', and 'Datastream'. The 'Datastream' tab is selected, and a modal window titled 'Virtual Pin Datastream' is open. The first datastream configuration is for 'MessageNumber':
- Name: MessageNumber
- Alias: i-MessageNumber
- Pin: V1
- Data Type: Integer
- Units: None
- Min: 0
- Max: 10
- Default Value: 0
The second datastream configuration is for 'TextSize':
- Name: TextSize
- Alias: TextSize
- Pin: V2
- Data Type: Integer
- Units: None
- Min: 1
- Max: 2
- Default Value: 0

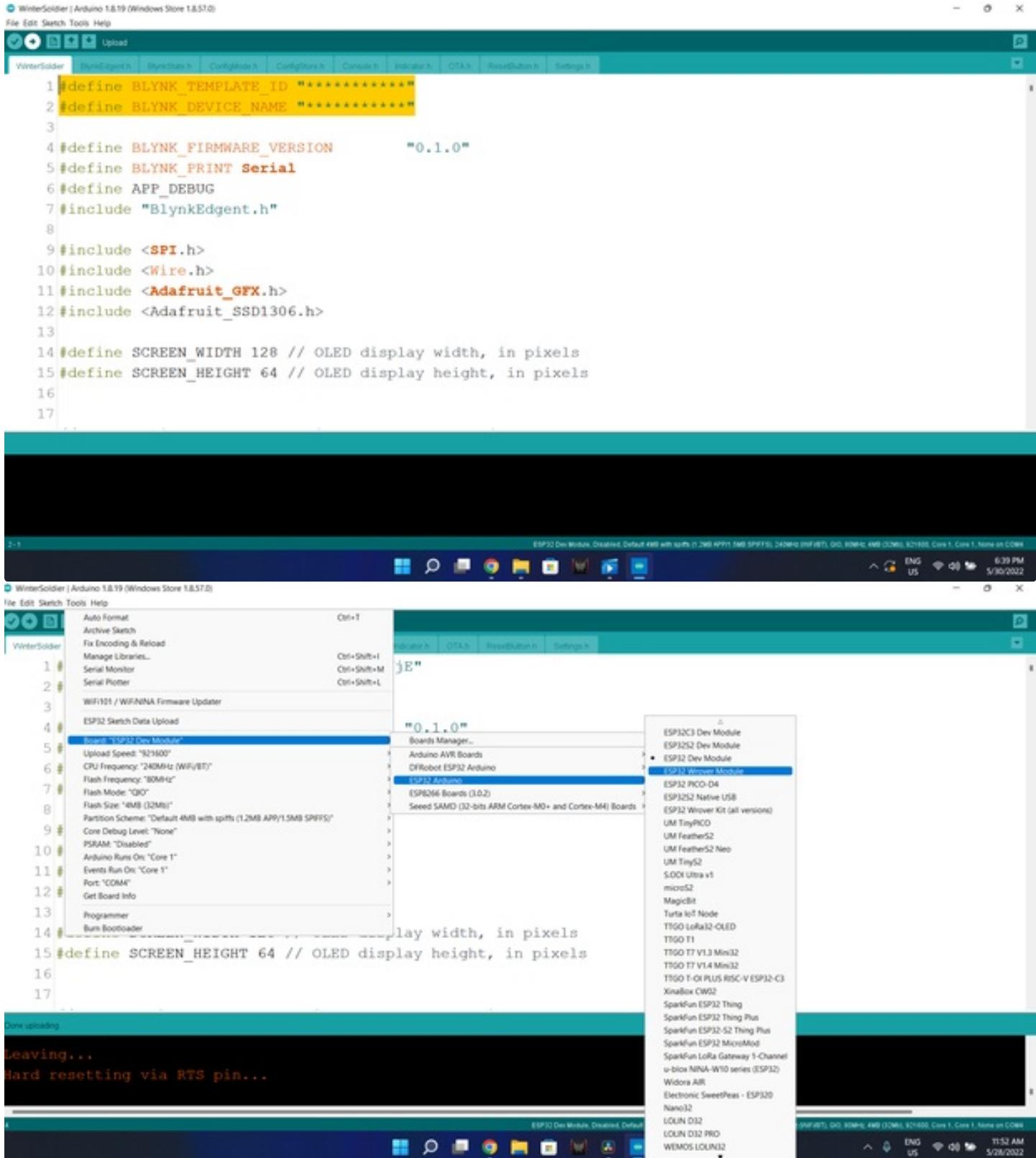
Step 7: Arduino Code

- Download the [Arduino Code](#) and extract the files.
- Open the code in Arduino IDE.
- Make sure that you have installed Blynk library in the arduino IDE.

- Paste the Template ID and Name From the blynk template that you have created.

```
#define BLYNK_TEMPLATE_ID "*****"
#define BLYNK_DEVICE_NAME "*****"
```

- Now choose the board type as ESP32, choose the right com port and upload the code.





Auto Format

Archive Sketch

Fix Encoding & Reload

Manage Libraries...

Serial Monitor

Serial Plotter

WIFI101 / WiFi-INA Firmware Updater

ESP32 Sketch Data Upload

Board: "ESP32 Dev Module"

Upload Speed: "921600"

CPU Frequency: "240MHz (WiFi/BT)"

Flash Frequency: "80MHz"

Flash Mode: "QIO"

Flash Size: "4MB (32Mbit)"

Partition Scheme: "Default-4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"

Core Debug Level: "None"

PSRAM: "Disabled"

Arduino Runs On: "Core 1"

Events Run On: "Core 1"

Port: "COM4"

Get Board Info

Programmer

Burn Bootloader

#define SCREEN_WIDTH 128 // OLED display width, in pixels

#define SCREEN_HEIGHT 64 // OLED display height, in pixels

16

17

Done uploading

Leaving...

Hard resetting via RTS pin...

```

Ctrl+T
Indicator.h GCODE.h ResetButton.h Settings.h
je"
"0.1.0"

Board: "ESP32 Dev Module"
Upload Speed: "921600"
CPU Frequency: "240MHz (WiFi/BT)"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Flash Size: "4MB (32Mbit)"
Partition Scheme: "Default-4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"
Core Debug Level: "None"
PSRAM: "Disabled"
Arduino Runs On: "Core 1"
Events Run On: "Core 1"
Port: "COM4"
Get Board Info
Programmer
Burn Bootloader
#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels
16
17

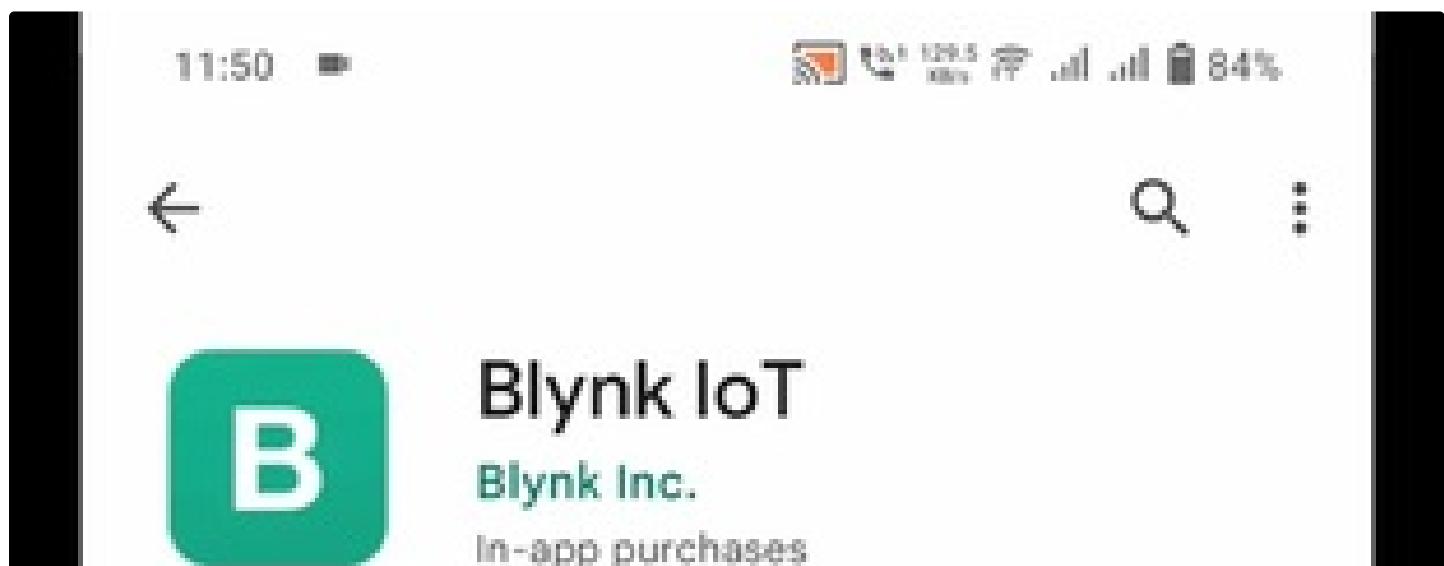
Done uploading
Leaving...
Hard resetting via RTS pin...

```

ESP32 Dev Module, Default-4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 1.0MHz (SWAP), QIO, 80MHz, 4MB (32Mbit), Core 1, Core 1, None on COM4

Step 8: Dashboard Setup

- Install the Blynk IOT app on you smart phone(Android/ IOS).
- Login with the same blynk account.
- Click on the 3 lines on top right side and click on add new device (Make sure the ESP32 is turned ON).
- You will find the device that you have created, click on it.
- Enter your Wi-Fi credentials and click on connect.
- After your device is connected setup the dashboard.
- Drag and drop 3 sliders, resize them as you want and assign each slider a DataStream that you have created.



[Uninstall](#)[Open](#)

What's new •

Last updated May 3, 2022



Minor fixes

Rate this app

Tell others what you think

[Write a review](#)

Developer contact



About this app



One App for All Devices

[Tools](#)

3.8 ★

1K reviews ⓘ

3+

Rated for 3+ ⓘ

100K+

Downloads

11:50 ■



Smart Sense 72E15



Offline



Devices



Notifications

11:52



83%



Add new device

11:52



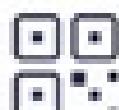
Add new device

Add new device



Connect to Wi-Fi

For devices using Wi-Fi to connect to the Internet



Scan QR-code



Have a QR-code or other number to
activate device?

11:52

83%



Check your device



Power On your device and make sure

that indicator is blinking

Connect to device

Blynk IoT app wants to use a temporary
Wi-Fi network to connect to your device



Blynk Mask-AA85E

Disconnected

[Cancel](#)

↻ Manual Connect

Ready

11:52

4G+ 83%



Choose your Wi-Fi network



ACE IoT



SRI PRABHU AAWAS



Sagar



JHAVERS 2.4G_EXT



softcore



JHAVERS 2.4G



Navaratna



SRI PRABHU TRADERS

11:53

4G+ 83%



Wi-Fi setup



Choose Wi-Fi network your device
will use

ACE IoT

Change



Remember the network for other devices

IP address settings

Continue

11:53

3G 1.27 4G+ 5G 83%



Configuring device



Configuring Wi-Fi



Reconnecting back to cloud



Waiting for device online

Cancel

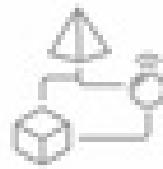
11:53



B Blynk



Mask



Smart Sense 72E15



Offline



Devices



Notifications

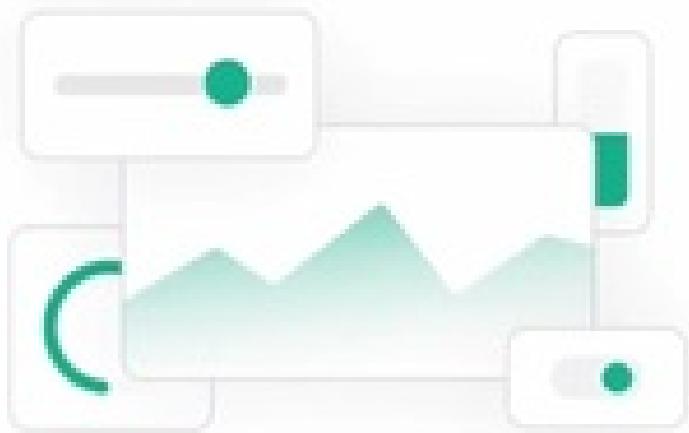
11:53



Mask



...



Setup your mobile dashboard

Start adding mobile widgets by tapping the
button below

Setup Dashboard

11:53



Widget Box



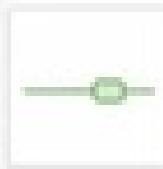
Icon Button

UPGRADE



Image Button

UPGRADE



Slider



Vertical Slider

UPGRADE



Step Slider

UPGRADE





Vertical
Step Slider

UPGRADE



Joystick

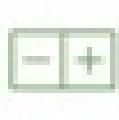


zeRGBa



RGB Light Control

UPGRADE



Step H

UPGRADE



Step V

UPGRADE



Slope Control

UPGRADE



Controls

UPGRADE



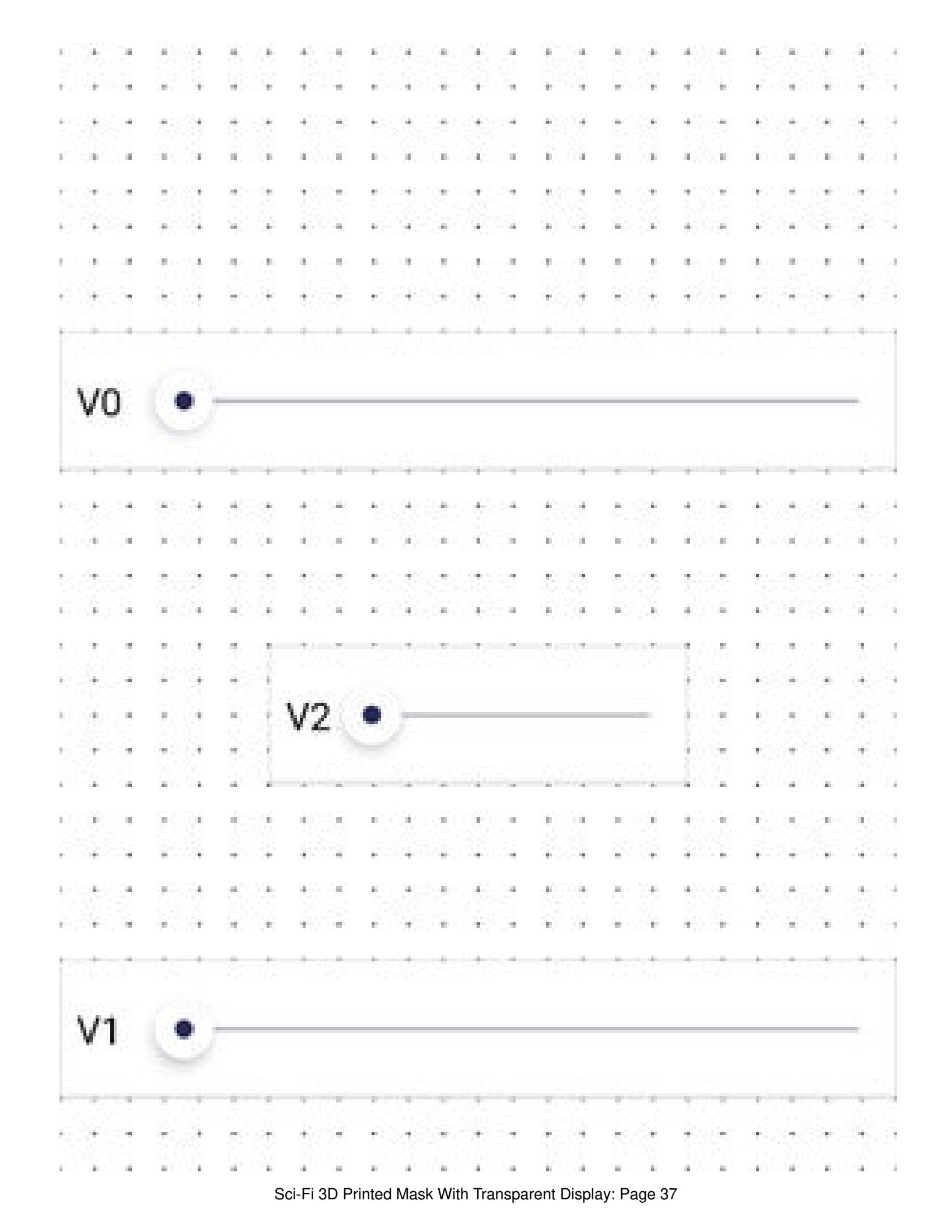
18:32

2.76 3.76 48%



Developer Mode





V0

V2

V1

18:32

5G 4G WiFi 4G 4G 48%



Slider Settings



Title (optional)



TITLE ALIGNMENT



DATASTREAM



AnimationNumber



SHOW VALUE

OFF



ON

ALIGNMENT



SEND ON RELEASE

OFF



ON



Delete

18:32

48%



Slider Settings



Title (optional)



TITLE ALIGNMENT



DATASTREAM



TextSize



SHOW VALUE

OFF



ON

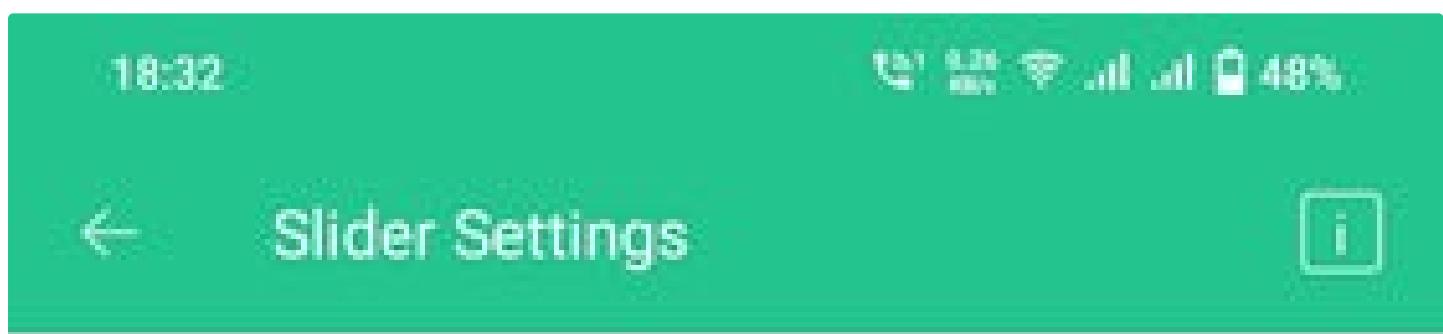
ALIGNMENT



SEND ON RELEASE



Delete



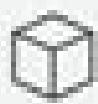
Title (optional)



TITLE ALIGNMENT



DATASTREAM



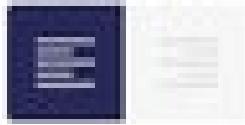
MessageNumber



SHOW VALUE



ALIGNMENT



SEND ON RELEASE





Delete

Step 9: Working

- The Animation Number slider is for switching between different animation, and the Animation Number 16 is for Displaying the Text.
- The text number is for switching between 11 different fixed text(In Arduino Code).
- Text Size is for choosing between 2 different text sizes.

18:32

48%



Mask

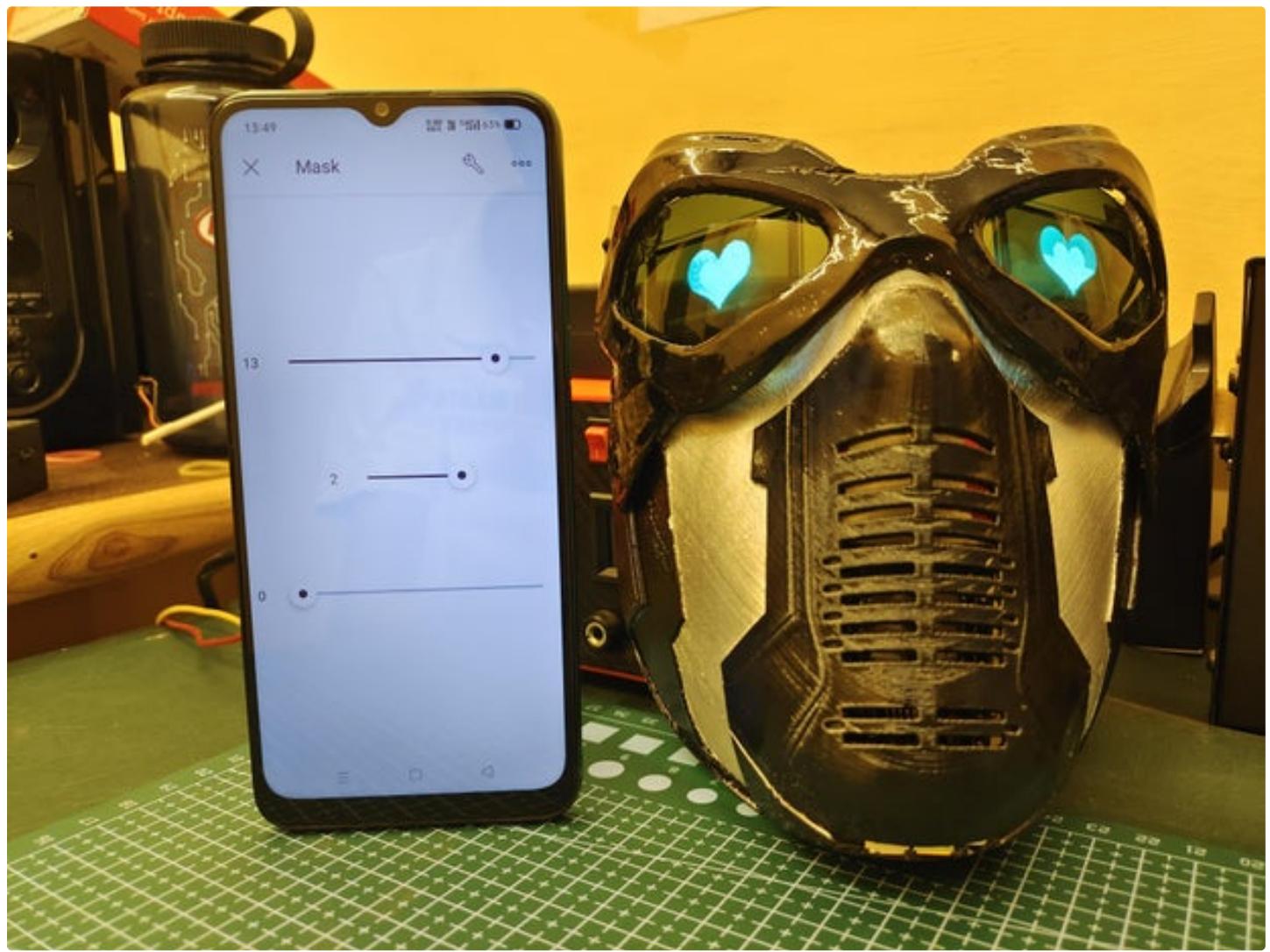


...

15

2

0



Sci-Fi 3D Printed Mask With Transparent Display: Page 45



Step 10: Code Explanation

- Resize your image or gif using [Resizing Tool](#) into 128x64 Pixels.
- If you are using gif Split the gif into frames using [Frame Splitter Tool](#).
- You can skip the frames that you feel unnecessary.
- Now download the frames as zip.
- Extract the zip file.
- Open the images in [Arduino Code Converter](#).
- If your content is black as the above gif then enable the **Invert image colors**.
- Click on Generate Code and copy the code.
- open the notepad paste the code and rename the prefix of the array for convenience.

For Example I have converted the frame name

```
epd_bitmap_frame_00_delay_0
```

to

```
emoji0
```

- Rename all the frames according to your convenience.
 - Now copy paste the code in Arduino code and call the frames where ever you want to display that frame.

```

if(Animation_Number == 1){
    display.clearDisplay();
    display.drawBitmap(0,0,emoji0, 128, 64, 1);
    display.display();
    delay(frame_delay);
}

if(Animation_Number == 2){
    display.clearDisplay();
    display.drawBitmap(0,0,emoji1, 128, 64, 1);
    display.display();
    delay(frame_delay);
}

```

- You can change the text by replacing the following texts in the code

```

if(MessageNumber ==0){
    Message ="Hello!";
}
if(MessageNumber ==1){
    Message ="Hello World!";
}
if(MessageNumber ==2){
    Message ="Pretty";
}if(MessageNumber ==3){
    Message ="Good";
}
if(MessageNumber ==4){
    Message ="Love U";
}
if(MessageNumber ==5){
    Message ="Hate U";
}
if(MessageNumber ==6){
    Message ="?";
}
if(MessageNumber ==7){
    Message ="Yes";
}
if(MessageNumber ==8){
    Message ="No";
}
if(MessageNumber ==9){
    Message ="Hi!";
}
if(MessageNumber ==10){
    Message ="How Are You??";
}

```

- In the code

```
int frame_delay =100;
```

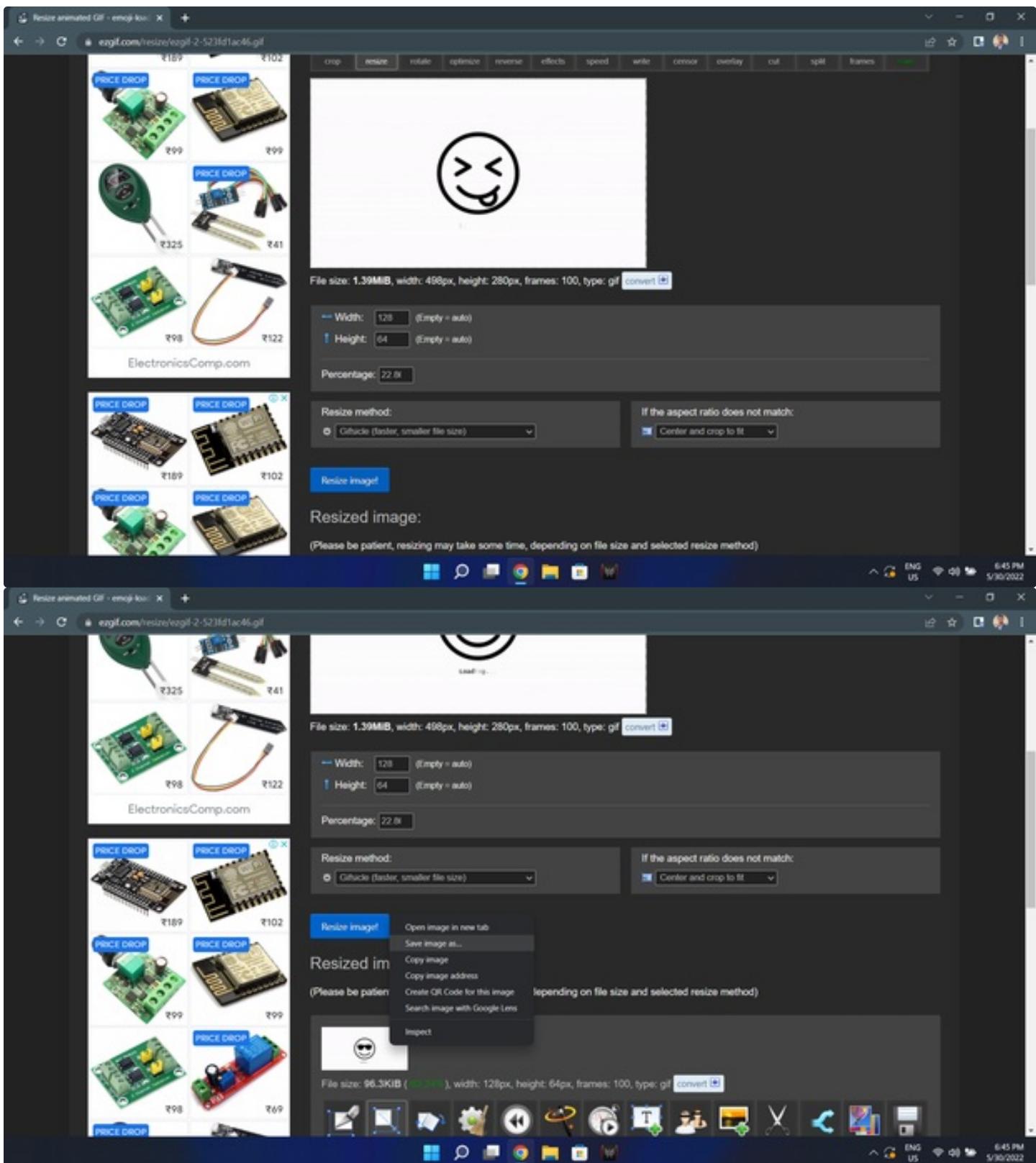
is for the delay between the animating frames, here 100 stands for 0.1 sec delay, if its 1000 it stands for 1 sec delay.

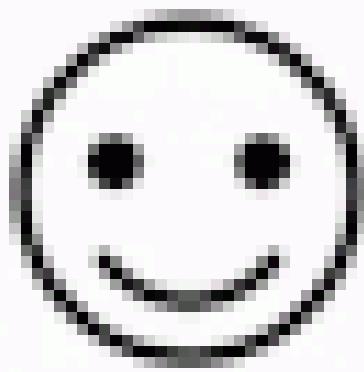


Screenshot of the EZGIF.COM Animated GIF resizer interface.

The interface includes:

- EZGIF.COM** logo with the tagline "ANIMATED GIFS MADE EASY".
- A banner for **borzo** with the text "Reduce your logistic costs! Send multiple deliveries with a single courier" and a "Sign up now" button.
- A navigation bar with various tools: GIF Maker, Video to GIF, Resize, Rotate, Crop, Cut, Optimize, Effects, Split, Add text, WebP, APNG, AVIF.
- A preview area showing a grid of small images related to price drops on electronic components.
- A main form titled "Animated GIF resizer" with sections for "Select image" (Upload image from your computer or paste image URL) and "Supported image types" (GiF, JPG, PNG, BMP, WebP, APNG, HEIC, FLIF, AVIF, MNG). It also specifies a "Max file size: 50MB".
- A note at the bottom: "For permanent links you can use: <https://ezgif.com/resize?url=https://example.com/source-image.gif>".





Screenshot of a web browser showing the EZGIF.COM website. The page title is "Split GIF image in frames". The main content area displays the "GIF frame extractor (splitter)" tool. On the left, there is a sidebar with an advertisement for Adobe Creative Cloud featuring a woman's face and the text "Not a pro? Not a problem. Create the unimaginable with new All features in Creative Cloud." Below the sidebar is a thumbnail image of a woman's face. The main right-hand section has a dark background with a light gray form. It includes a "Select image" section with a "Choose File" button and a "No file chosen" message, and another section for pasting an image URL with a "https://example.com/source-image.gif" placeholder. Below these are instructions: "Supported image types: animated GIF, WebP, APNG, FLIF, MNG, AVIF" and "Max file size: 50MB". At the bottom, there is a note: "For permanent links you can use: https://ezgif.com/split?url=https://example.com/source-image.gif". The browser interface shows multiple tabs at the top, including "Resize animated GIF - emoji-face" and "Split GIF image in frames". The status bar at the bottom right shows system information like battery level, signal strength, and the date/time (6:45 PM, 5/30/2022).

The screenshot shows a web-based application for extracting frames from a GIF. At the top, there's a toolbar with various editing tools: crop, resize, rotate, optimize, reverse, effects, speed, write, censor, overlay, cut, split, and frames. Below the toolbar, a single frame is displayed with a sad face emoji. The file details are shown as: File size: 96.34KB, width: 128px, height: 64px, frames: 100, type: gif. A "convert" button with a plus sign is also present. Under "Split options", there's a checked checkbox for "Redraw every frame with details from previous frames". A "Split to frames" button is located below this section. The main area is titled "Split images:" and shows a grid of 100 small frames. Each frame contains a different electronic component or device with a "PRICE DROP" label and a price value. The bottom of the grid has a "Microsoft Shop with Microsoft Rewards" advertisement with a "DOWNLOAD" button. The entire interface is set against a dark background.

Waiting for pagead2.googlesyndication.com...

Resize animated GIF - esgil-2-71 | Split GIF image in frames - esgil-2-71

GIF frame extractor (splitter)

PRICE DROP ₹189 PRICE DROP ₹99

PRICE DROP ₹102 PRICE DROP ₹99

PRICE DROP ₹98 ₹3,999

PRICE DROP ₹325 ₹41

ElectronicsComp.com

File size: 96.34KB, width: 128px, height: 64px, frames: 100, type: gif convert +

Split options:

Redraw every frame with details from previous frames ✓

Split to frames

Split images:

Microsoft Shop with Microsoft Rewards DOWNLOAD

Animated GIF, APNG, WebP, FLIF, AVIF and MNG frame splitter (extractor/decompiler)

Waiting for pagead2.googlesyndication.com...

Resize animated GIF - esgil-2-71 | Split GIF image in frames - esgil-2-71

Split images:

Edit animation Download frames as ZIP

😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊
😊	😊	😊	😊	😊	😊

ElectronicsComp.com

6:46 PM 5/30/2022

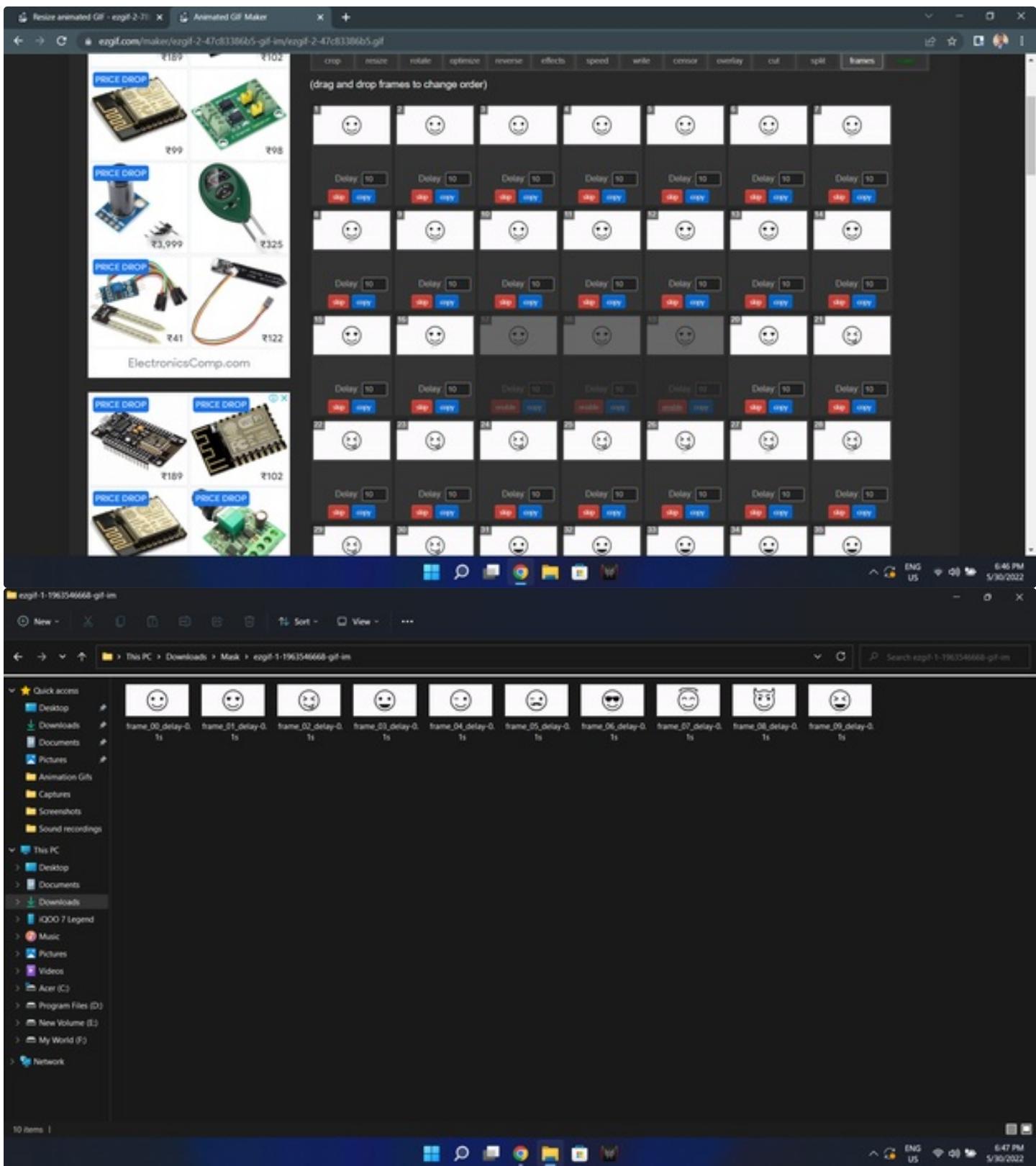


image2cpp is a simple tool to change images into byte arrays (or your array back into an image) for use with Arduino and (monochrome) displays such as OLEDs. It was originally made to work with the Adafruit OLED library. An example sketch for Arduino and this library can be found [here](#).

More info (and credits) can be found in the [Github repository](#). This is also where you can report any [issues](#) you might come across.

This tool also works offline. Simply save this page to your computer and open the file in your browser.

1. Select image or **1. Paste byte array**

2. Image Settings

Canvas size(s): No files selected

Background color: White Black Transparent

Invert image colors

Brightness / alpha threshold:

(+25% if the brightness of a pixel is above the given level the pixel becomes white, otherwise it's black)

1. Paste byte array

128 X 64 px

[Read as horizontal](#) [Read as vertical](#)

2. Image Settings

Canvas size(s): No files selected

Background color: White Black Transparent

Invert image colors

Brightness / alpha threshold:

(+25% if the brightness of a pixel is above the given level the pixel becomes white, otherwise it's black)

1. Paste byte array

128 X 64 px

[Read as horizontal](#) [Read as vertical](#)

2. Image Settings

Canvas size(s): No files selected

Background color: White Black Transparent

Invert image colors

Brightness / alpha threshold:

(+25% if the brightness of a pixel is above the given level the pixel becomes white, otherwise it's black)

Screenshot of the image2cpp application interface showing the "Split GIF image in frames - egypt" tab. The interface includes:

- Frame List:** A list of 12 frames from frame_00 to frame_11, each 128x64 pixels.
- Background color:** Radio buttons for White (selected), Black, or Transparent.
- Invert image colors:** A checked checkbox.
- Brightness / alpha threshold:** A slider set to 128.
- Scaling:** A dropdown menu set to "original size".
- Center:** Unchecked checkboxes for horizontally and vertically.
- Rotate image:** Unchecked checkboxes for rotate 180 degrees.
- Flip:** Unchecked checkboxes for horizontally and vertically.
- Note:** "Note: centering the image only works when using a canvas larger than the original image."

3. Preview

The preview shows 10 frames of an emoji sequence. The first row contains five frames: smiley, smiley, sad face, smiley, and neutral face. The second row contains five frames: neutral face, sunglasses, smiley, cat face, and sad face.

Screenshot of the image2cpp application interface showing the "Split GIF image in frames - egypt" tab. The interface includes:

- Frame List:** A list of 12 frames from frame_00 to frame_11, each 128x64 pixels.
- Background color:** Radio buttons for White (selected), Black, or Transparent.
- Invert image colors:** A checked checkbox.
- Brightness / alpha threshold:** A slider set to 128.
- Scaling:** A dropdown menu set to "original size".
- Center:** Unchecked checkboxes for horizontally and vertically.
- Rotate image:** Unchecked checkboxes for rotate 180 degrees.
- Flip:** Unchecked checkboxes for horizontally and vertically.
- Note:** "Note: centering the image only works when using a canvas larger than the original image."

3. Preview

The preview shows 10 frames of an emoji sequence against a black background. The first row contains five frames: smiley, smiley, sad face, smiley, and neutral face. The second row contains five frames: neutral face, sunglasses, smiley, cat face, and sad face.

Resizing animated GIF - espgf-2-71 | Split GIF image in frames - espgf | image2cpp

<img alt="A screenshot of a web-based tool for generating Arduino code from images. At the top, five small black squares with white smiley faces are shown. Below them is a section titled '4. Output' with a dropdown menu set to 'Arduino code'. A note says 'Adds some extra Arduino code around the output for easy copy-paste into this example. If multiple Images are loaded, generates a byte array for each and appends a counter to the identifier.' Under 'Identifier/Prefix:' is a field containing 'epd_bitmap_'. The 'Draw mode:' dropdown is set to 'Horizontal - 1 bit per pixel'. A note below says 'If your image looks all messed up on your display, like the image below, try using a different mode.' Below this is a diagram showing a 16x8 grid of pixels being mapped to a 16x8 grid of memory locations. At the bottom are two buttons: 'Generate code' and 'Copy Output'. The 'Copy Output' button is highlighted. A large window below shows the generated Arduino code. The code starts with a comment // 'frame_00_delay_0', followed by a const unsigned char array named 'epd_bitmap_frame_00_delay_0' containing 128 bytes of hex values. It then defines a PROGMEM variable 'PROGMEM = {'. The code continues with many more hex values, ending with a closing brace '}'. Below this is a multi-line comment // Array of all bitmaps for convenience. (Total bytes used to store images in PROGMEM = 38400). It defines a const int 'epd_bitmap_allArray_LENGTH = 10;' and a const unsigned char array 'epd_bitmap_allArray[10] = {' followed by ten entries for 'epd_bitmap_frame_xx_delay_yy'. The bottom of the screen shows a Windows taskbar with icons for Start, Search, Task View, File Explorer, Edge, Google Chrome, File Manager, and File Explorer. The system tray shows battery level, signal strength, and the date/time (5/30/2022, 6:48 PM). The status bar at the bottom left says 'In 685 Col 3'.</p>

WinterSoldier | Arduino 1.8.7r9 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help

Winterrider | Sketch | Serial Monitor | ConfigMode.h | ConfigSpiral.h | Console.h | Indicator.h | OTA.h | ResetButton.h | Settings.h

```
19 #define OLED_CLK    18 //SCL
20 #define OLED_MOSI   23 //SDA
21 #define OLED_RESET  13 //RES
22 #define OLED_DC     26 //DC
23 #define OLED_CS     27 //
24 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, OLED_MOSI, OLED_CLK, OLED_DC, OLED_RESET, OLED_CS);
25
26 int frame_delay =100;
27
28 // 'frame_00_delay=0', 128x64px
29 const unsigned char mando0[] PROGMEM = {
30 0x00, 0x00,
31 0x00, 0x00,
32 0x00, 0x00,
33 0x00, 0x00,
34 0x00, 0x00,
35 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

ESP32 Dev Module, Disassembled, Default 480 MHz with 2.7V 4.9V 5.0V SPIFFS, 24MHz 8MHz I2C, 0.000s, 0.000ms, 0.000ms, Core 1, Core 2, RAM 1.000MB

ENG US ⌂ 8:51 PM 5/30/2023

```
8473 const unsigned char emoji0[] PROGMEM = {  
8474 0x00,  
8475 0x00,  
8476 0x00,  
8477 0x00,  
8478 0x00,  
8479 0x00,  
8480 0x00,  
8481 0x00,  
8482 0x00,  
8483 0x00,  
8484 0x00,  
8485 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
8486 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xfe, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
8487 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0xfc, 0x1f, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00,  
8488 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xc0, 0x01, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00,  
8489 0x00, 0x00, 0x00, 0x00, 0x0e, 0x00, 0x00, 0x70, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}
```

6473 ESP32 Dev Module, Created: Default (480 MHz 4MB SPIFFS), 240MHz (WIFI), QIO, 80MHz (4MB SPIFFS), 80MHz, Core 1, Core 1, None on COM1

6474 ENG US 6:51 PM 5/30/2022

```
14694 }  
14695  
14696 if(Animation_Number == 1){  
14697   display.clearDisplay();  
14698   display.drawBitmap(0,0,emoji0, 128, 64, 1);  
14699   display.display();  
14700   delay(frame_delay);  
14701 }  
14702  
14703 if(Animation_Number == 2){  
14704   display.clearDisplay();  
14705   display.drawBitmap(0,0,emoji1, 128, 64, 1);  
14706   display.display();  
14707   delay(frame_delay);  
14708 }  
14709  
14710 if(Animation_Number == 3){
```

Updates available for some of your boards and libraries

6474 ESP32 Dev Module, Created: Default (480 MHz 4MB SPIFFS), 240MHz (WIFI), QIO, 80MHz (4MB SPIFFS), 80MHz, Core 1, Core 1, None on COM1

6475 ENG US 6:52 PM 5/30/2022

```
14541 if (MessageNumber ==0) {
14542     Message ="Hello!";
14543 }
14544 if (MessageNumber ==1){
14545     Message ="Hello World!";
14546 }
14547 if (MessageNumber ==2) {
14548     Message ="Pretty";
14549 }if (MessageNumber ==3) {
14550     Message ="Good";
14551 }
14552 if (MessageNumber ==4) {
14553     Message ="Love U";
14554 }
14555 if (MessageNumber ==5) {
14556     Message ="Hate U";
14557 }
```