FeTAp-32 Build Manual

lspr98

April 26, 2025

Abstract

This document guides you through the process of building your own *FeTAp-32* from an old *FeTAp 611-2*. Please read through the whole document before starting the build, to see what awaits you. These instructions are just my recommendations, feel free to tinker if you want to do something different.

The FeTAp-32 is an ESP32C3 running ESPHOME with external components, sitting inside an old German retro rotary-dial telephone. Since its based on ESPHOME, it is modular and you may skip components that you do not want or add your own customizations. Each component (speaker, microphone, rotary-dial, etc...) has its own section in the manual. Optional components are marked with *(optional)* in the section title.

If a component requires 3D-printed parts, they are found as the first step of the sub-chapter. You can of course print them all together once you have decided what features you want to include. All 3D-printed parts are found in the *hardware* folder of the repository, both as *.step* and *.stl* files. The parts were designed in such a way that no additional screws or nuts are required, which sometimes came at the cost of ease-of-assembly. It is however definitely possible to assemble the whole device without a second person.

The manual is divided into two sections: the first section deals with building the hardware, the second is concerned with the software (building, flashing, adding to home assistant, etc...).

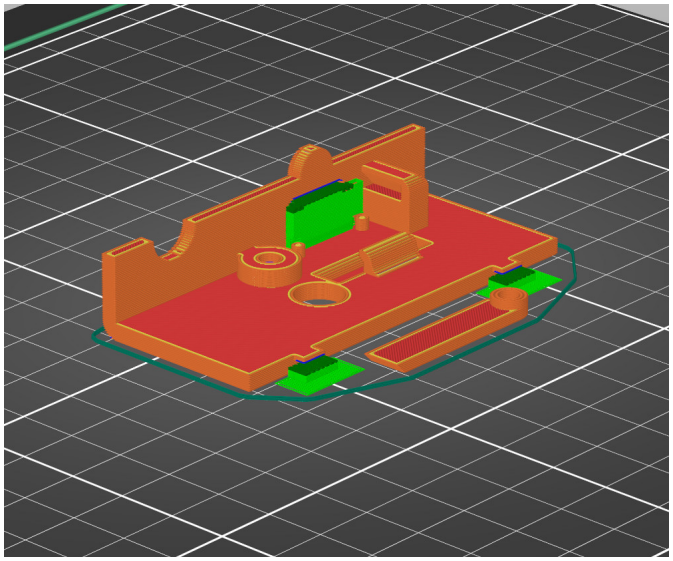
Contents

1 Hardware

1.1 USB-C port installation

1.1.1 3D-printed parts

Print **Connector_Cover v9.stl** and **Connector_Brace v3.stl** using a filament of your choice (mine is printed out of PLA+). Orient the parts as shown in the picture. **Enable support everywhere**. You may use **0.3mm layer height** to speed up the print.

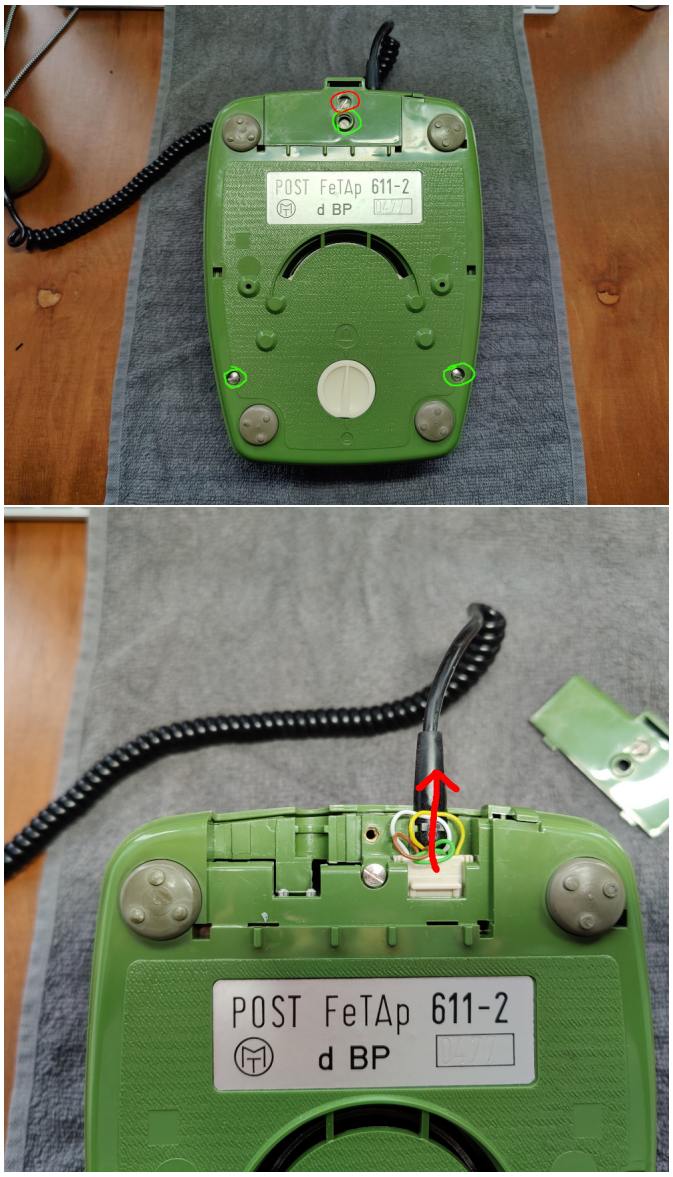


1.1.2 Opening the telephone

Turn the telephone on its back. Verify that the telephone actually is a FeTAp 611-2. There are 3 screws securing the housing (green) and one screw securing the port cover (red), see first image. The housing screws may be covered by seals to prevent modifying the telephone. In that case you need to remove the seals (**this will void your warranty**).

Next, unscrew the port cover screw (red) and remove the port cover (second image). Unplug any cable that is plugged in (third image). Be careful when removing the cables: they have a little notch in the rubber-end that acts as as stress-relief. You need to gently lift the rubber-part before pulling the plug.

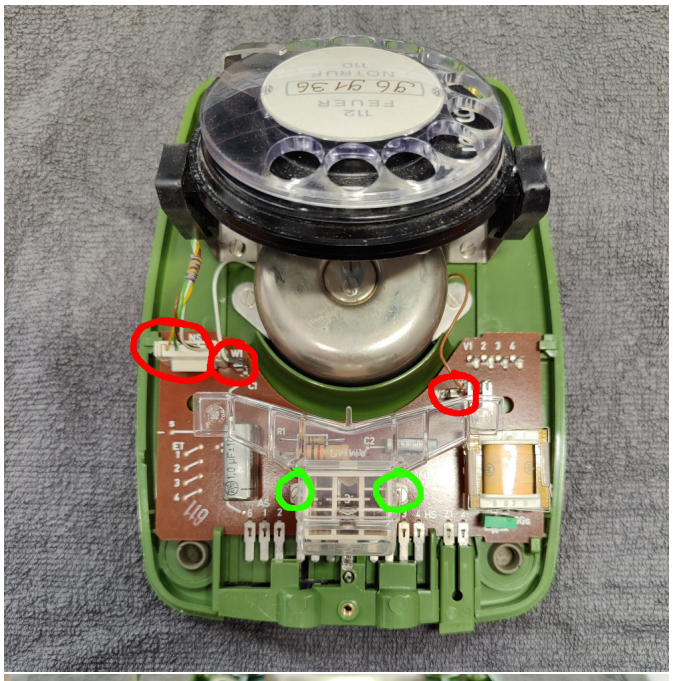
Finally, unscrew the 3 housing screws (green) and remove the housing.

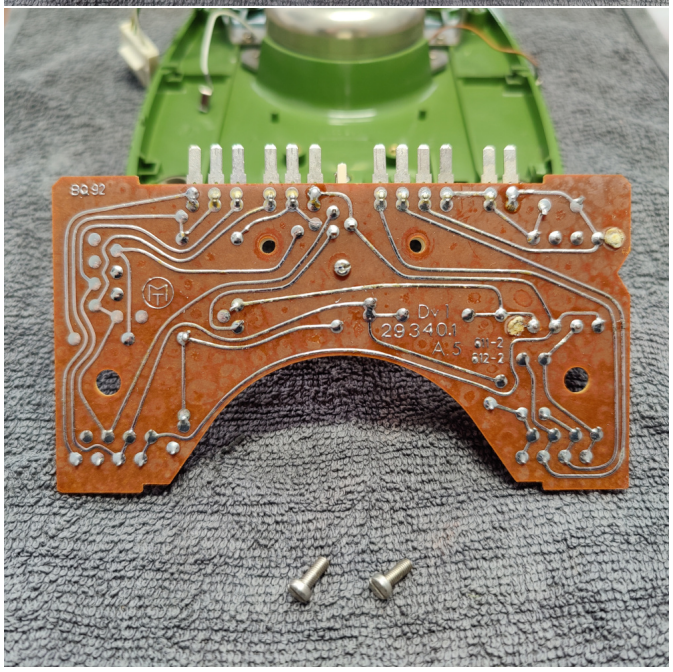


1.1.3 Removing the PCB

Unplug the 3 plugs (red) and unscrew the two screws (green) next to the handset mechanism. We will later re-install the PCB so don't lose the two screws!

Afterwards, carefully slide the PCB towards the front of the telephone and lift the PCB at an angle to remove it. The backside of the PCB and the two screws are shown in the second picture



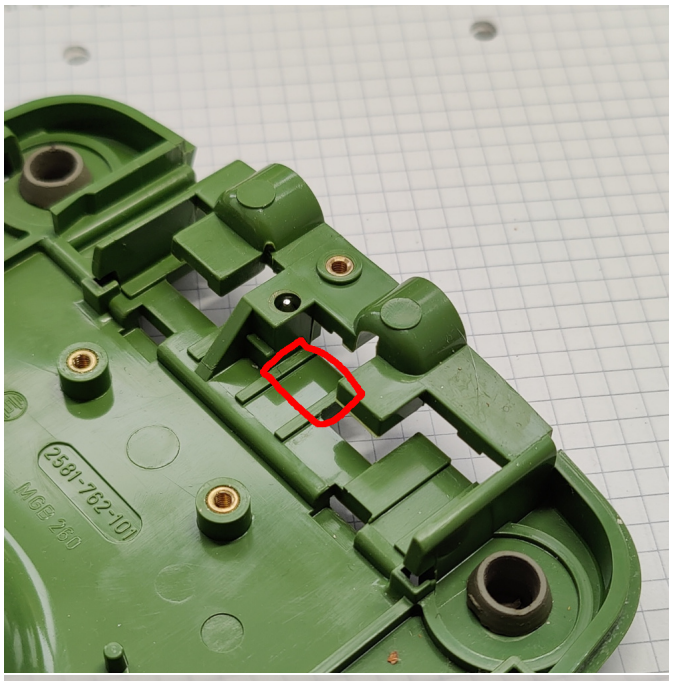


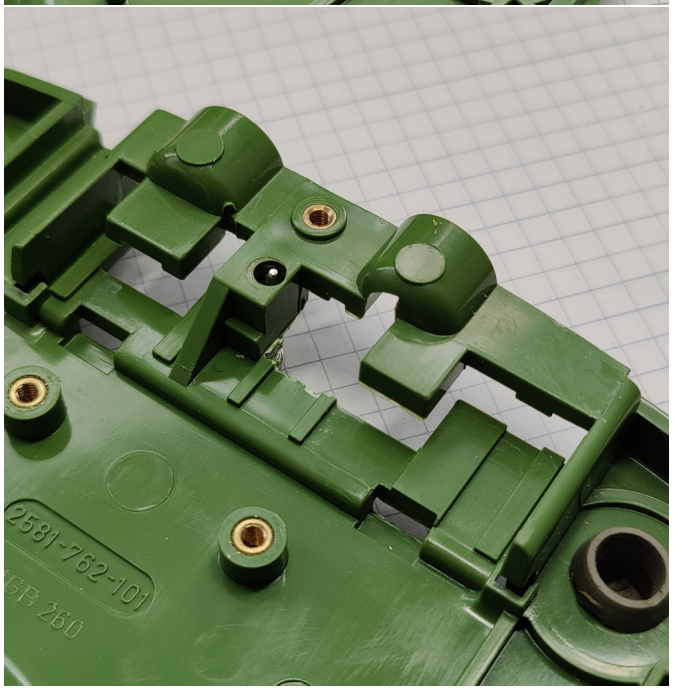
1.1.4 Making room for the USB-C board

The back of the base-plate needs to be adjusted slightly to make room for the USB-C breakout board.

A small area needs to be cut-off (marked in red on first image). Be careful not to break the back of the base-plate.

The result should look similar to the second image.

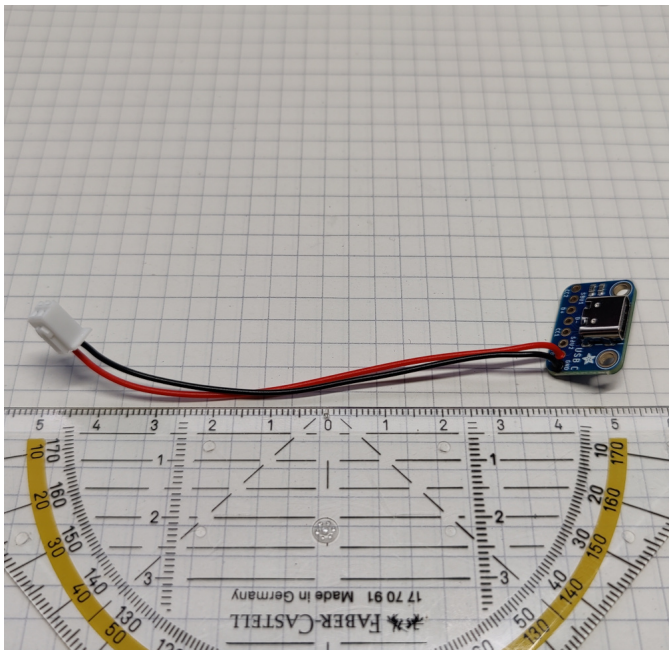




1.1.5 Preparing the USB-C board

Prepare two cables of approx. 10cm length. Crimp a 2-pin JST-XH female connector to one end and solder the other ends to the **GND** and **VBUS** pin holes.

Preferably, use a black cable for **GND** and a red cable for **VBUS**. The connector you just crimped will supply your FeTAp-32 with 5V.



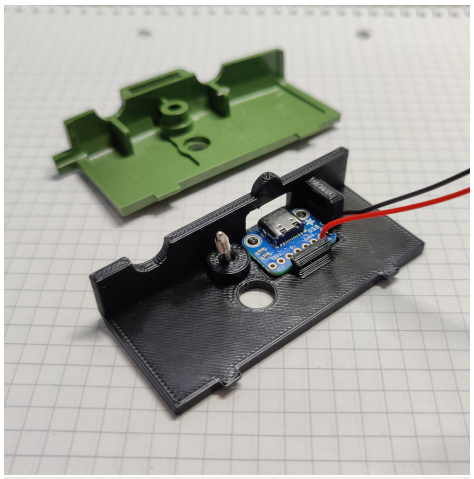
1.1.6 Installing the new port cover

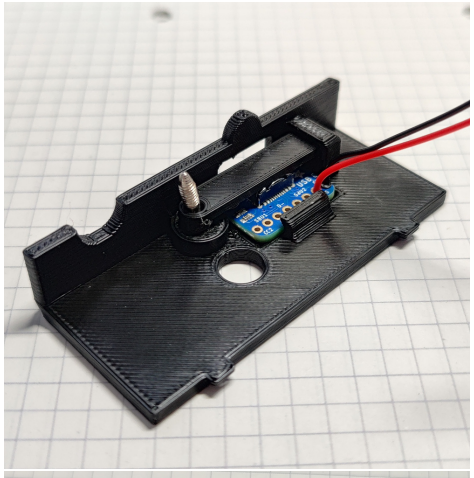
Slot the USB-C board into Connector_Cover v9.stl as shown in the first picture. The USB-C board should sit evenly on the 3D-printed part.

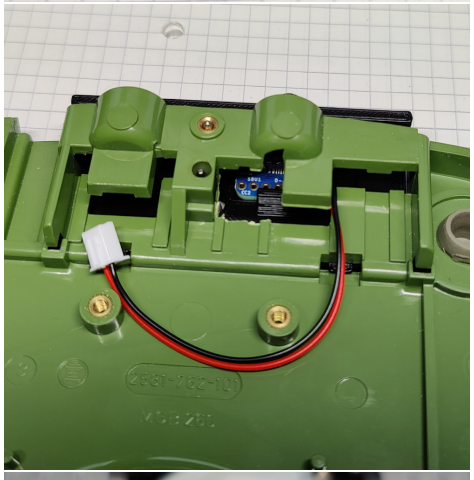
Next, use Connector_Brace v3.stl and the port cover screw from the original port cover to secure the USB-C board as shown in the second picture. The Connector_Brace v3.stl can only be installed in one direction.

Thread the connector and cable that you soldered to the USB-C port through the opening in the back of the base-plate and secure the new port cover with the port cover screw. The result should look similar to the third picture.

Finally, verify that you can plug and unplug a USB-C cable without the USB-C board coming loose and that the USB-C plug sits evently on the port cover (fourth picture).







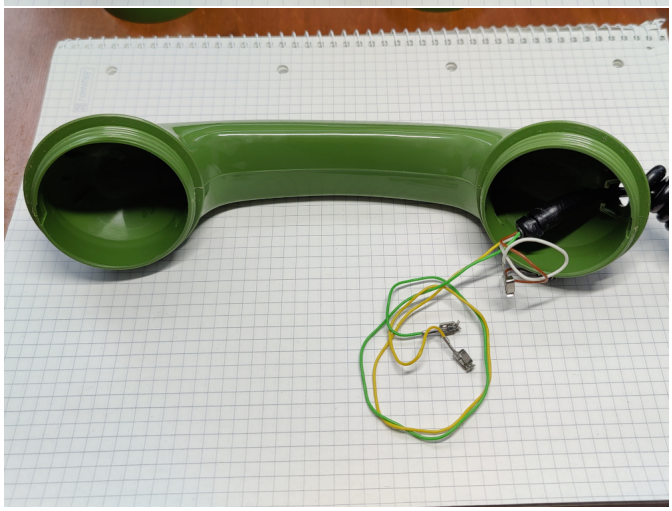
1.2 Handset preparation

1.2.1 Removing the old electronics

Verify that the handset is already disconnected from the telephone, which was part of the previous section.

Unscrew the microphone and speaker covers. Remove the old microphone and speaker as well as their rubber fittings (first picture).

Unplug the microphone and speaker and pull the speaker wires through the handset such that all wires are hanging out at the bottom of the handset (second picture).

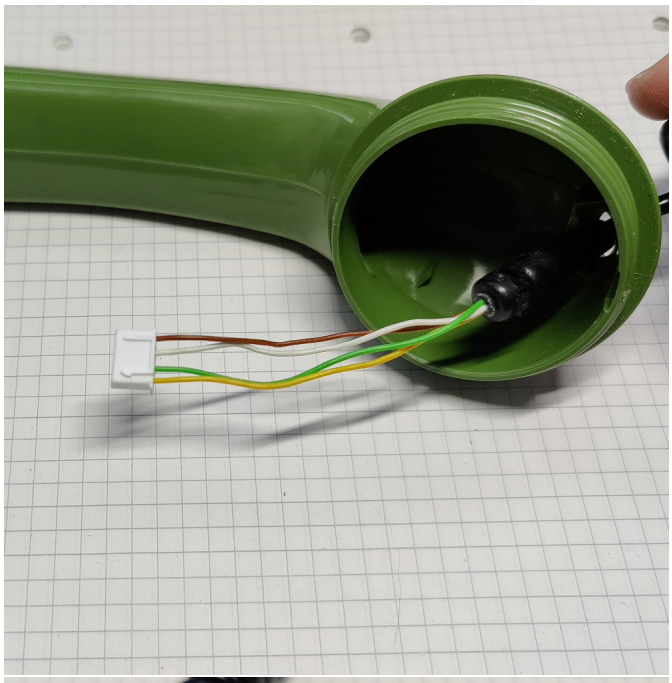


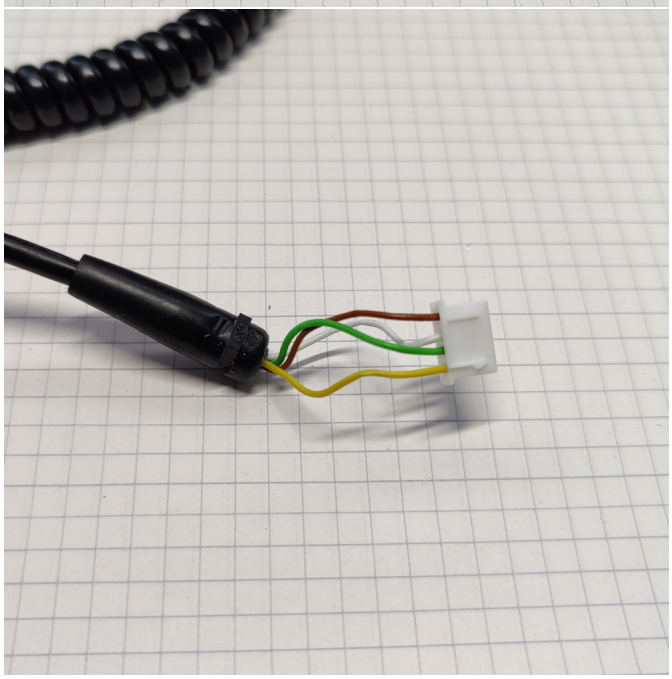
1.2.2 Crimping the handset connectors

There should be four wires running through the handset cord, colored brown, white, green and yellow.

I used **brown for ground** (GND), **white for 5V** (VBUS), **green for the handset-sensor pin** and **yellow for the rotary-dial sensor pin**. If you decide to use a different pin-out, make sure to accommodate for the changes in all other steps as well!

Crip a 4-pin JST-XH female connector to each side of the handset cord (first and second picture). You don't necessarily need to use the same pinning order on the connectors as long as all pins are connected correctly to the ESP in the end.

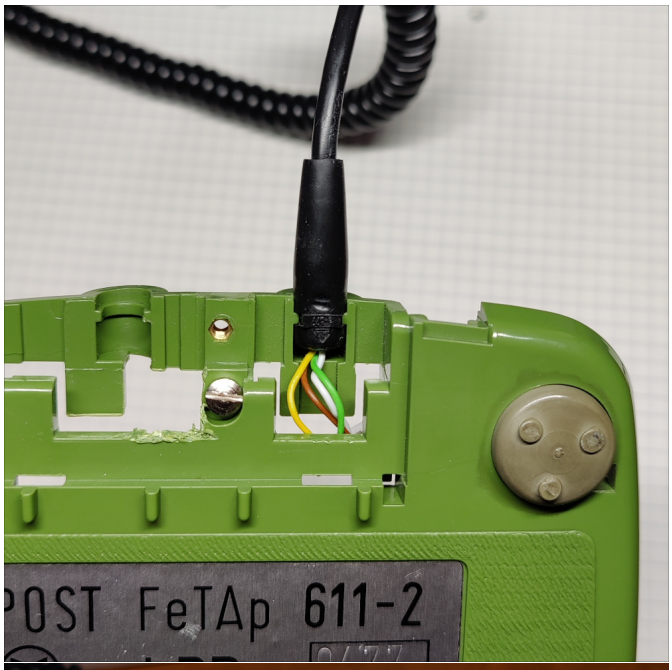


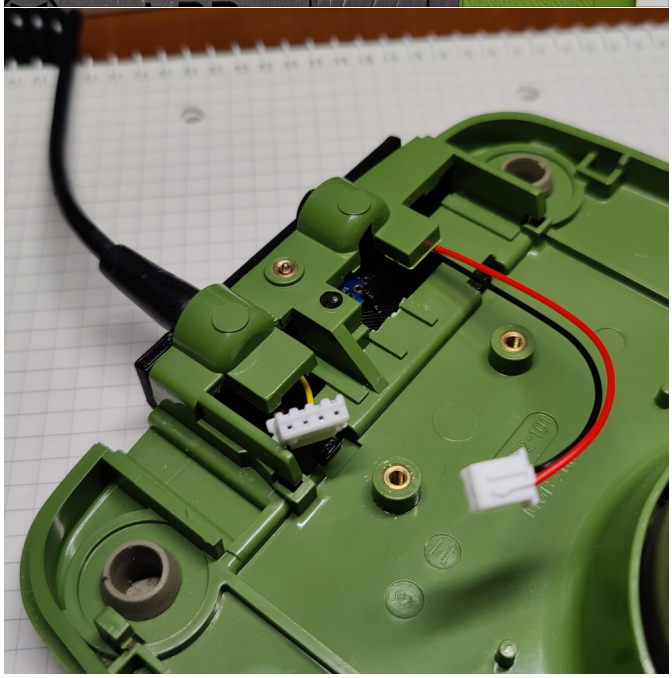


1.2.3 Reconnecting the handset

Unscrew the 3D-printed port cover again and re-insert the handset cable into its slot as shown in the first picture. Note that there is a little ridge in the rubber end that acts as a stress-relief for the cord.

Make sure to pull the JST connector through the opening in the back of the base-plate and re-screw the 3D-printed port cover on. The result should be similar to the second picture.





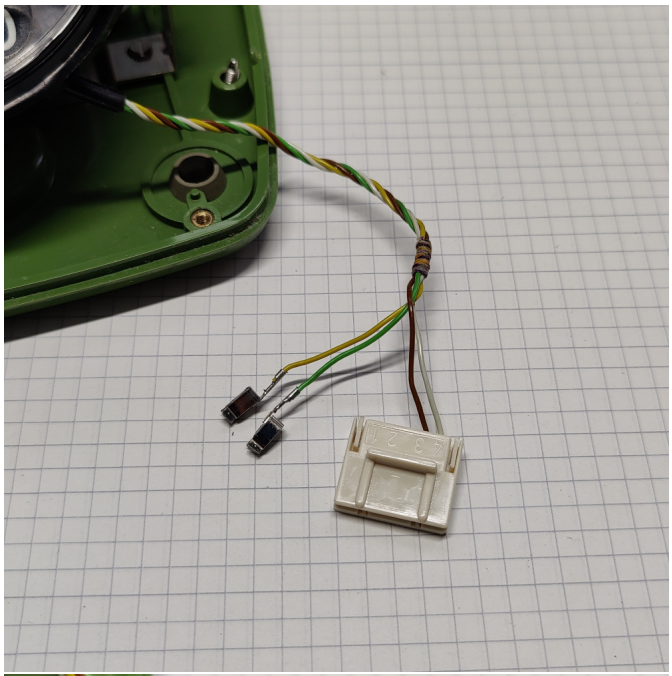
1.3 Rotary-Dial-Sensor preparation (optional)

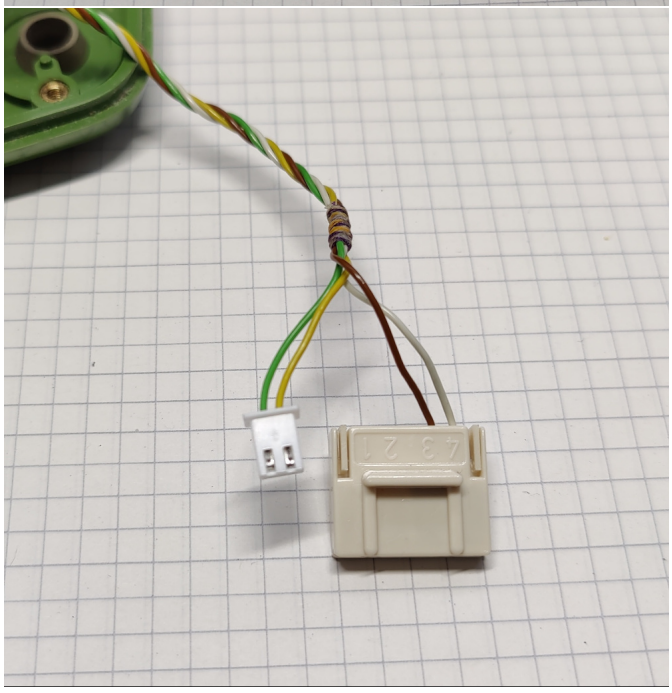
The rotary-dial sensor is optional but highly recommended, as it basically adds 10 individual virtual buttons through the numbers that can be dialed. It is also comparably low effort as you only need to crimp a connector. Its also very satisfying to trigger actions via the rotary-dial.

1.3.1 Crimping the rotary-dial connector

The rotary-dial has a 4-pin connector that we previously unplugged from the PCB. We only require the yellow and green wires.

Remove them green and yellow wires from the connector as shown in the first picture. Cut off the old metal contacts from these two wires and leave as much wire as possible to have room for error. Crimp a 2-pin JST-XH female connector to the yellow and green wire as shown in the second picture. **The pin order does not matter**, as it is basically just a contact that opens and closes in a certain pattern if the dial is spinning.





1.4 Handset-Sensor preparation (optional)

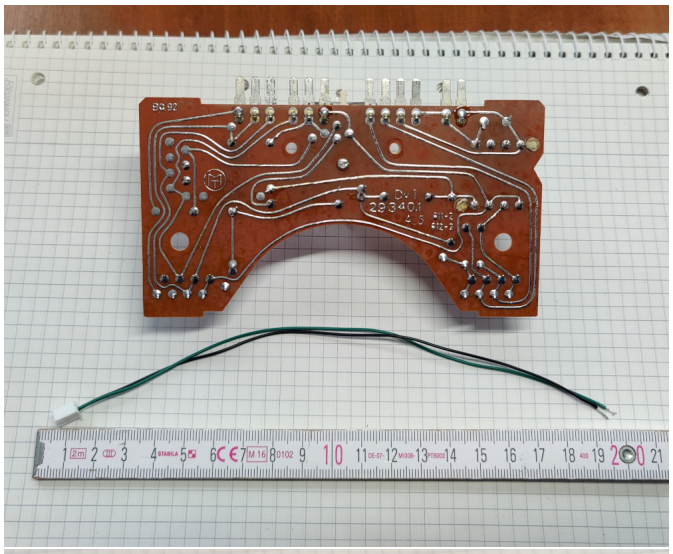
The handset-sensor senses if the handset is picked-up or put-down. It is optional but highly recommended, as it allows you to trigger custom actions or only activate the voice assistant (and microphone) if the handset is picked-up, removing the need for any wake-word detection.

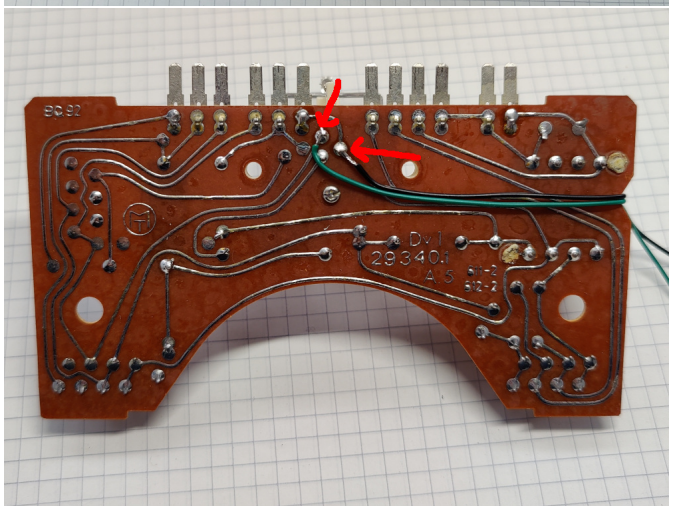
1.4.1 Hooking into the handset-sensor mechanism

The handset-sensor uses a spring-loaded contact that opens when it is pressed down (e.g. the handset is placed on top of the telephone). We need to attach two wires to be able to sense when that contact opens or closes. The PCB should already be removed from previous steps.

Prepare two wires, each approx. 20cm in length and crimp a 2-pin JST-XH female connector to them (see first picture).

Next, solder the two wires to the PCB as shown in the second picture. Make sure you do not bridge any other contacts while soldering.



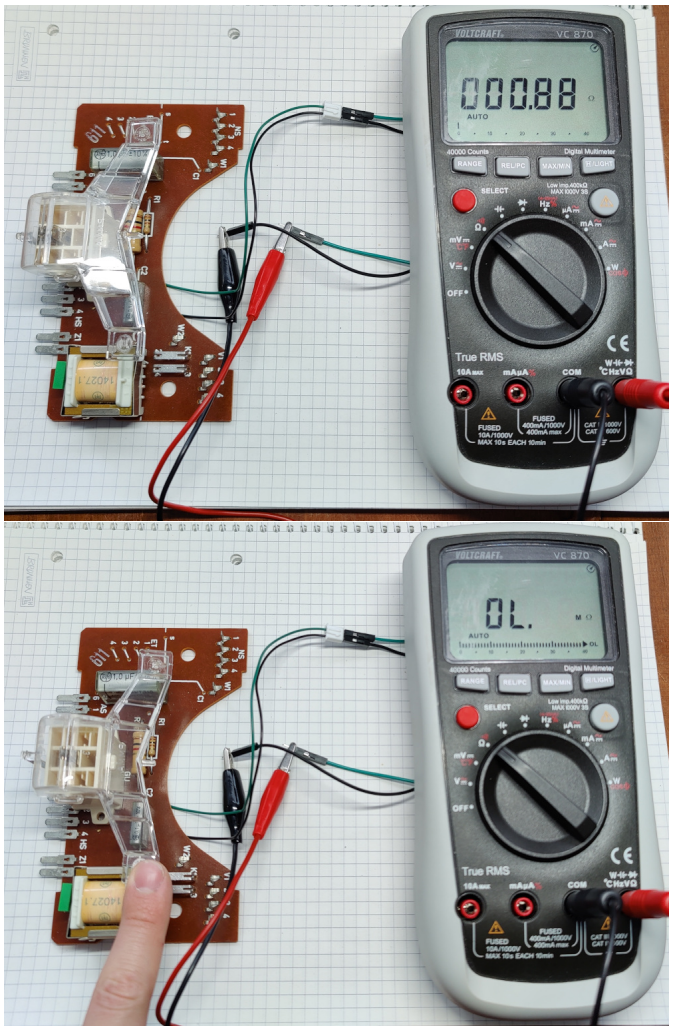


1.4.2 Function check (optional)

Use a multimeter and measure the resistance between the two wires you just soldered to the PCB.

Depending on the wear of your phone, the resistance should be close to zero when the handset-sensor is not pressed. In the first picture you can see my multimeter showing 0.8Ω when I am not pressing the contact down.

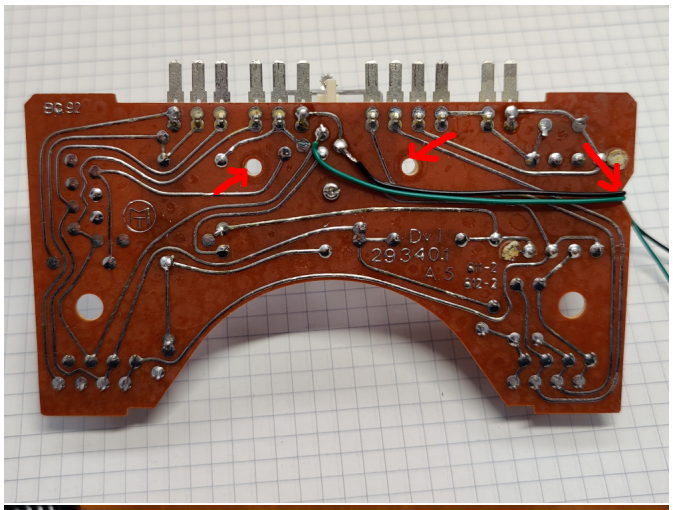
If the handset-sensor is pressed, the multimeter should show O.L. (open loop), meaning the contact open (see second image).



1.4.3 Installing the PCB

It is now time to put the PCB back into the telephone. **Be careful not to pinch the two wires you just soldered**. Route them past the two mounting holes and use the little ridge on one side of the PCB for cable management as shown in the first picture.

Slot in the PCB at an angle and secure it using the two screws next to the handset-sensor mechanism. **Make sure not to pinch any other cables too!**. If you did not skip any optional steps, you should end up with something similar to the second picture: There are four JST plugs hanging loose. Three 2-pin female plugs for power-input, rotary-dial-sensor and handset-sensor and one 4-pin female plug coming from the cord that attaches the handset.



1.5 Housing wiring

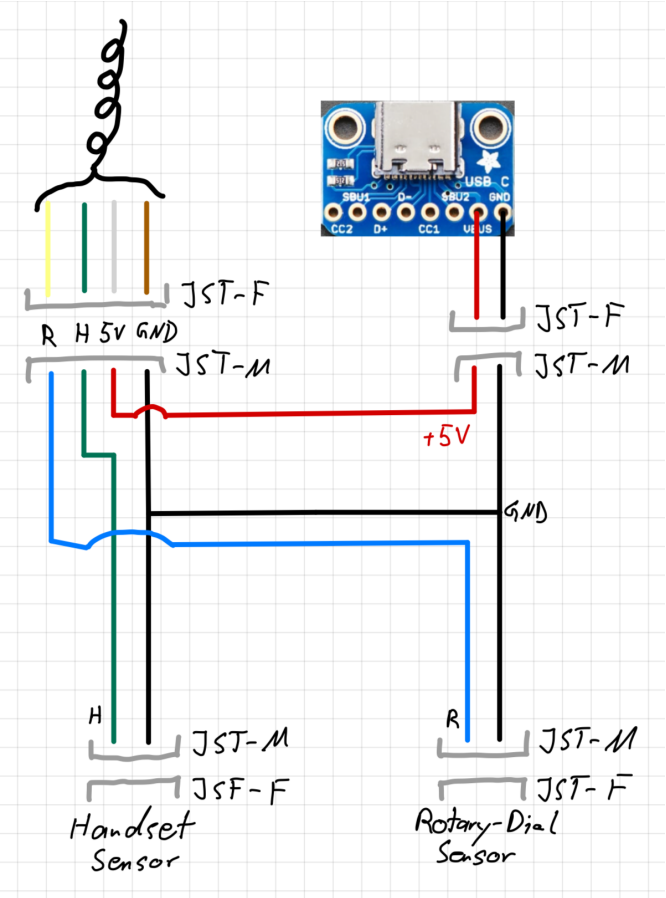
1.5.1 Housing cable harness

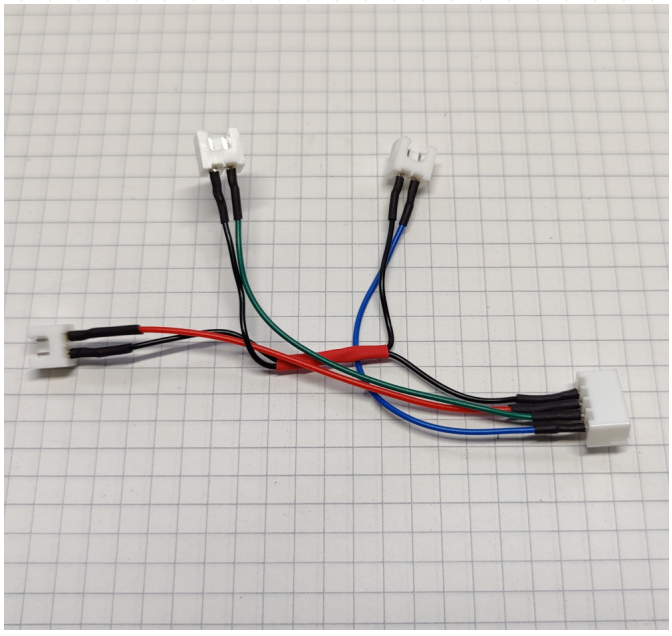
Next, you will have to create a connection between the connectors we just installed into the telephone. In the first picture, you can see a wiring diagram of how the components are supposed to be connected. If you did not install the optional components, you can of course ignore the respective connections.

The yellow, green, white and brown wires are the cables coming out of the handset-cord, which you terminated by a 4-pin JST female connector (JST-F) previously. Two wires running through the cord are used to supply 5V (white) and GND (brown) to the ESP32 and the other two are for the rotary-dial-sensor (R) and handset-sensor (H).

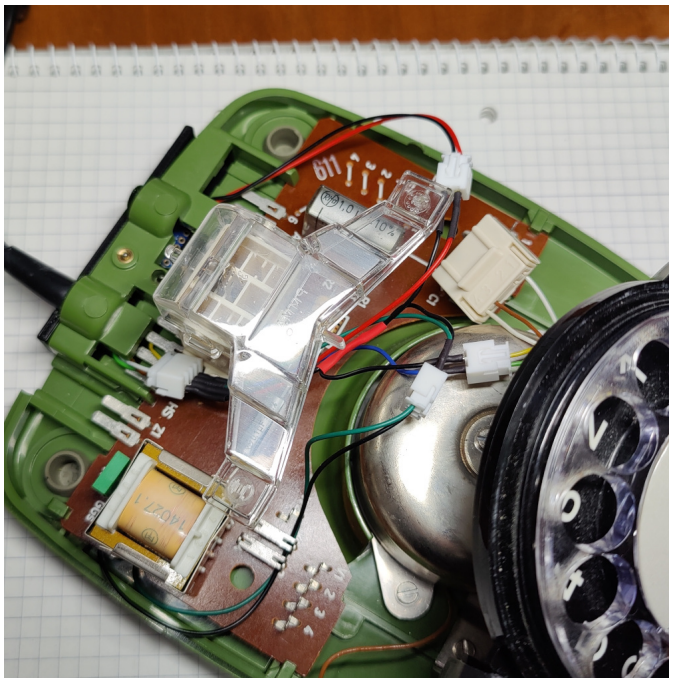
In the second picture you can see the cable harness I created for a *full-build* (all optional components installed). Since we only installed JST-female connectors, **the cable harness only uses JST-male connectors**. **If you chose a different pinning on the cord JST female connector before, be sure the reflect your changes!**

The **R-pin** is used for the rotary-dial sensor, and the **H-pin** is used for the handset-sensor. Since those two sensors are only contacts, switching the respective sensor-pin and the ground pin doesn't matter. **The cables for the harness are about 10cm long. Be sure to use shrink-tubing to avoid any shorts.**





1.5.2 Final checks and closing the housing



Once the cable harness is finished, **connect all components using the harness as shown in the picture**. Cable management is not that important, as it will be hidden anyway. However, **be sure that no cables are blocking the handset-sensor mechanism!**

Now is a good time to check if you wired everything up correctly before closing the housing. If you have a multimeter, measure the voltage between the brown and white wires on the handset-side. It should show approx. 5V when you connect a USB-C cable (which of course has to be connected to a PC or phone charging brick on the other end).

If you prepared the handset-sensor, measure the resistance between the brown and green wires on the handset-side. It should be close to zero when the handset-sensor is not pressed and it should show O.L. (open loop) when you press the handset-sensor.

If you prepared the rotary-dial-sensor, measure the resistance between the brown and yellow wires on the handset-side. It should be close to zero in idle. Now dial the number **1** (you have to turn the dial until you hit the metal limiter) and **very slowly** let it spin back by restricting its speed with your finger. You should see that the multimeter shows O.L. (open loop) or a very high resistance for a short amount of time.

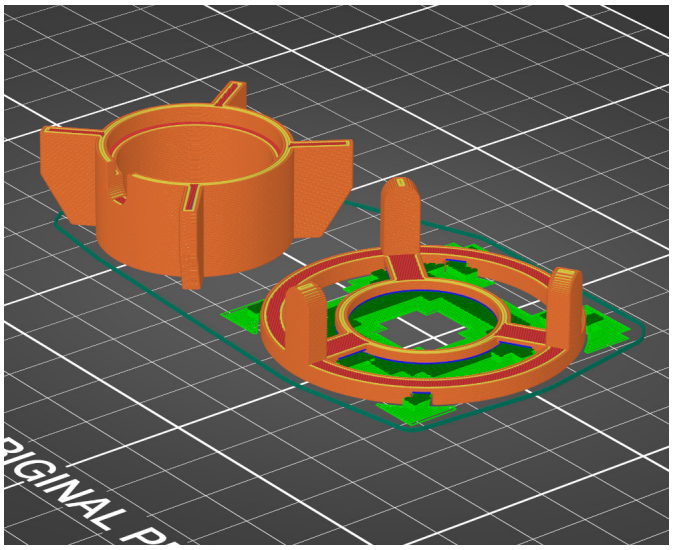
Finally, **place the housing back on the base-plate and secure it with the three housing screws on the back**. We are now finished with the housing and will continue with the handset (the fun part).

1.6 Speaker installation (optional)

The speaker is technically optional, but required for a full voice-assistant experience. If you do not install a speaker, you will get no feedback from the assistant.

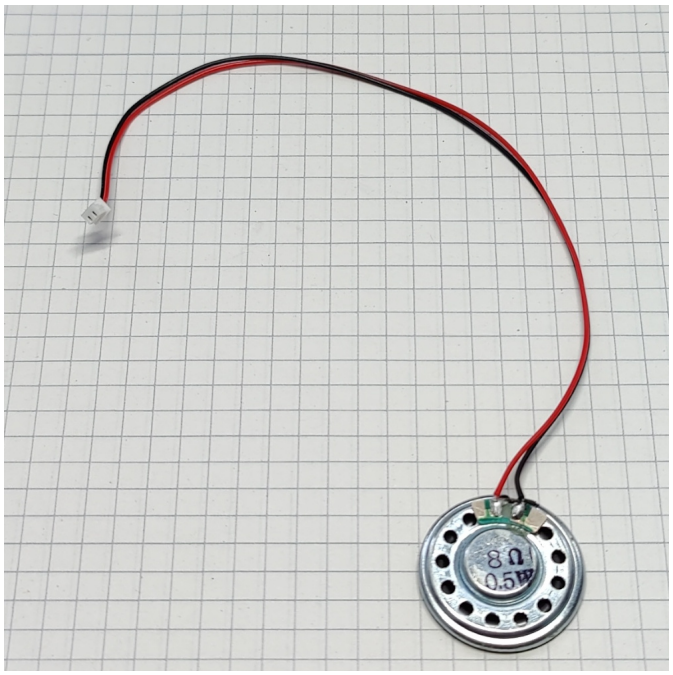
1.6.1 3D-printed parts

Since the new speaker is way smaller than the old one, a frame had to be designed to position the new speaker in the ear-piece compartment of the handset. You will need to print **Speaker_Brace v8.stl** and **Speaker_Insert v10.stl** with **supports enabled**. **Orient the parts as shown in the image**. Again, you can use any filament and I recommend 0.3mm layer height for faster printing. While the printer is running, you can already continue with the next step.



1.6.2 Soldering the speaker connector

Since the I2S amplifier uses a pico-blade connector, which I only managed to get precrimped (see BoM), we do not need to crimp this time. However, before soldering the wires to the speaker, **make sure the wires are at least 20cm long!** Otherwise, they will be too short to be connected to the amplifier later on. If they are too short, solder some extra length to them (don't forget shrink tubing) and finally **solder the wires to the speaker as shown in the image**.



1.6.3 Installing the speaker

Before installing the parts into the handset, take a closed look at the two 3D-printed parts. **Speaker_Brace v8.stl** is meant to go on top of **Speaker_Insert v10.stl** as shown in the first picture. Note that **they only fit together in a certain direction** (marked in red).

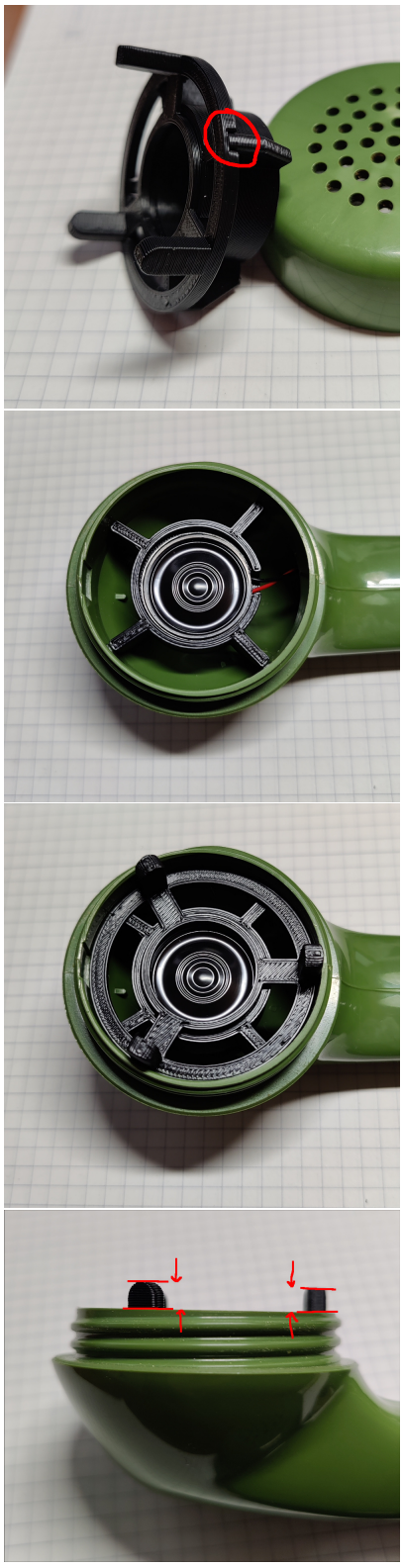
Take **Speaker_Insert v10.stl** and place it inside the ear piece compartment as shown in the second picture. There is an opening in **Speaker_Insert v10.stl** that is meant to be for the speaker cable, which should be aligned with the cable channel that runs through the handset.

Next, take the speaker and place it inside **Speaker_Insert v10.stl** facing upwards with the cables running through the opening of **Speaker_Insert v10.stl** into the cable channel (second picture). Verify that the speaker cable comes out at the lower end of the handset.

Then place **Speaker_Brace v8.stl** on top and make sure it is correctly aligned! (third picture)

Make sure that the three arms of **Speaker_Brace v8.stl** stick out 3-5mm from the edge of the threading as shown in the fourth picture. Verify that they are approximately equally high such that the speaker sits evenly. If they stick out too much or the speaker isn't even, the parts need repositioning.

Finally, you can screw the cover back on. The speaker is now ready!



1.7 Microphone preparation (optional)

The microphone is only required if you want voice-assistant capabilities. You don't need it if you only want to use the rotary-dial for example.

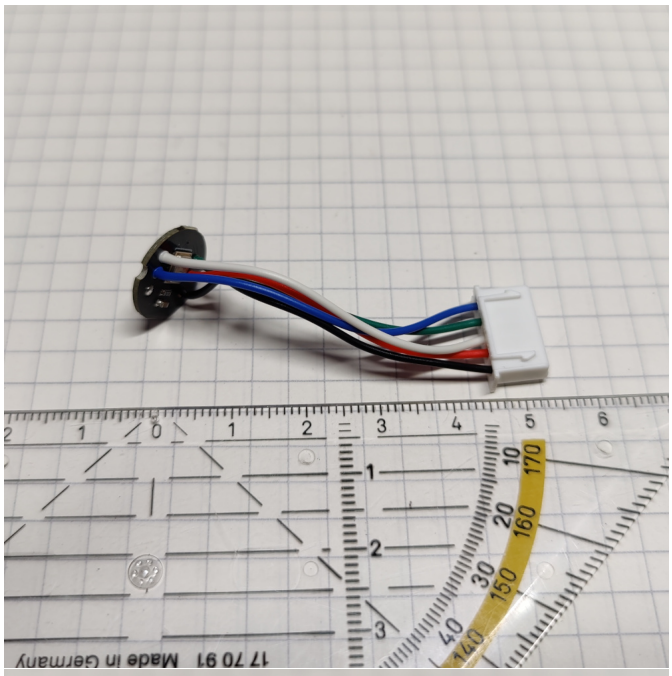
1.7.1 Soldering and crimping the microphone-side connections

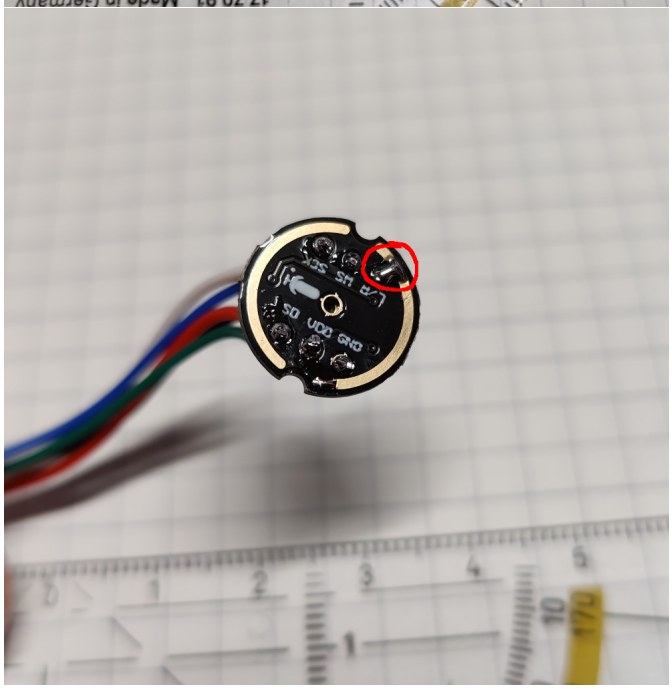
The microphone used is an INMP441 I2S digital microphone. There are five connections necessary to operate it: Ground, VCC (3.3V), data, word-select (or left/right clock) and bit-clock.

This means you need to **prepare 5 cables of approx. 5cm length and solder them to the GND, VCC, SD, WS and SCK pins of the microphone and crimp a JST-XH 5-pin female connector to the other end** as shown in the first picture. **Make sure that the cables are coming out of the back side of the microphone**.

The top side is the side with the small hole in the middle (microphone opening) and the microphone-symbol printed in silk-screen. I am using the following colour-code: **Black = GND, Red = VCC, White = SCK, Green = SD and Blue = WS**. You should **pull down the left/right selector pin by soldering it to the outer ground pad** of the microphone (red circle in second image).

For now, that is all that has to be done for the microphone. Put it aside, we will use it later.



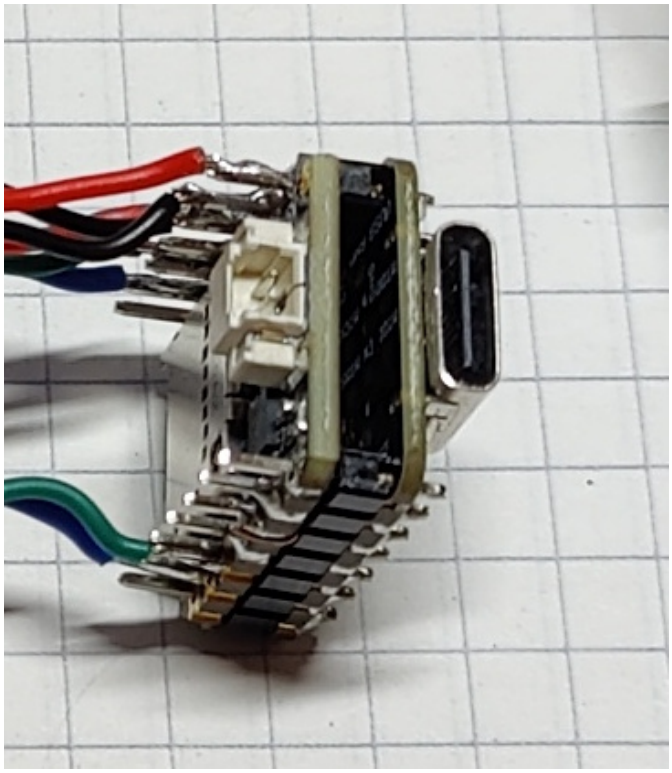


1.8 ESP preparation

1.8.1 ESP pin-header soldering

If your ESP did not come with pre-soldered pin headers, **start by soldering the pin headers to the back of the ESP**.

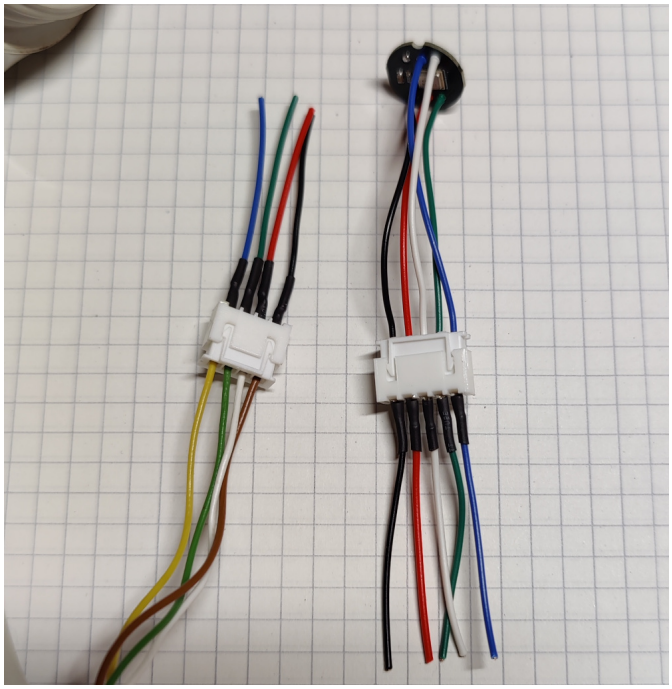
If you installed the speaker, **solder the I2S amplifier board to the back of the ESP** as shown in the picture (ignore the wires for now). The port of the speaker connector should be on the same side as the USB-C port of the ESP.



1.8.2 Handset cable harness

Next, you will need to create the connection from the JST-XH 4-pin female connector of the cord to the ESP (shown on the left in the image). **The cables are approx. 5cm long and use the following colour-coding: Black = GND, Red = VBUS, Green = Handset-Pin, Blue = Rotary-Dial-Pin**. They are attached to a JST-XH 4-pin male connector to connect to the JST-XH 4-pin female connector coming out of the cord on the handset-side.

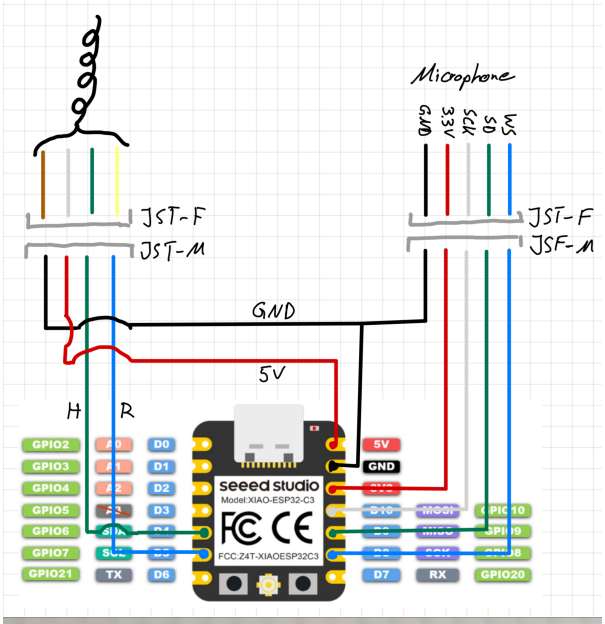
If you prepared a microphone, you also need to create a connection for the JST-XH 5-pin female connector of the microphone to the ESP (shown on the right in the image). **These cables are also approx. 5cm long and use the same colour-coding as for the microphone: Black = GND, Red = VCC, White = SCK, Green = SD and Blue = WS**. Attach a JST-XH 5-pin male connector to them to connect it to the microphone.

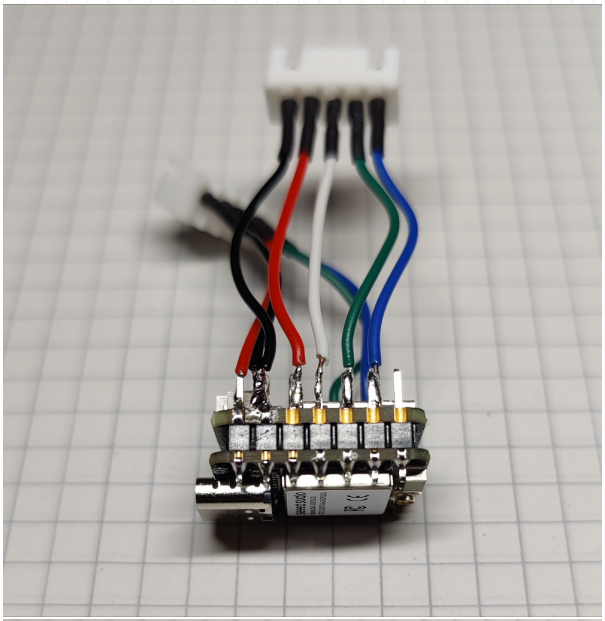


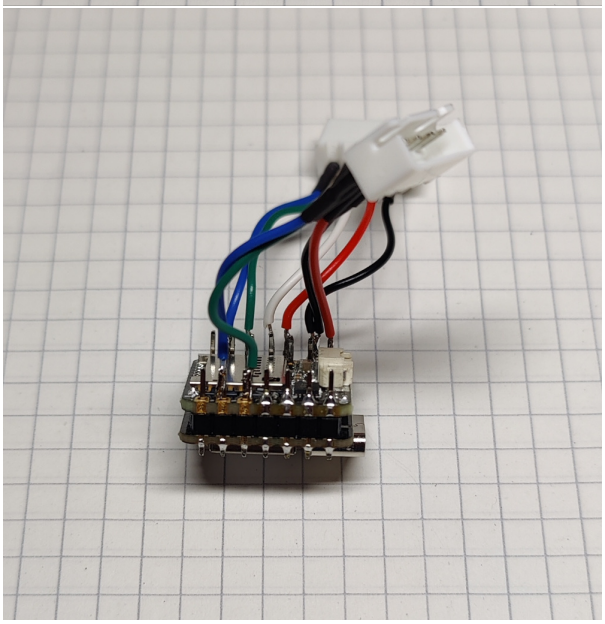
1.8.3 ESP connector soldering

In this step, you will solder the handset cable harness to the ESP. Please familiarize with the wiring diagram in the first picture before you begin. The microphone connections only apply if you prepared a microphone. The handset-pin (H) and rotary-dial-pin (R) also only apply, if you prepared them earlier.

I recommend soldering the microphone connector (right connector in diagram) first, as all its pins are on the same side. Secondly, solder the cord connector (left connector in diagram). The final product should look similar to the second and third picture.



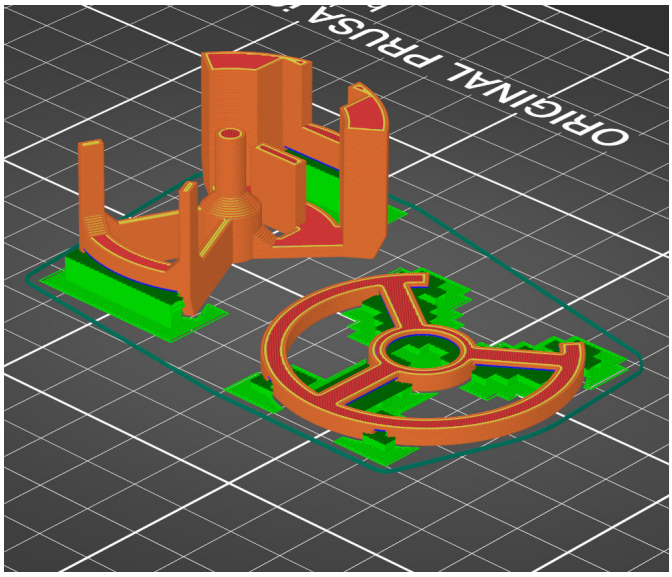




1.9 ESP installation

1.9.1 3D-printed parts

The ESP and microphone are housed inside the microphone compartment of the handset. Similar to the speaker, they will be sitting in a 3D-printed frame. Please print **Esp_Insert v1.stl** and **Esp_Brace v1.stl** (found in the hardware folder) with **supports enabled**. **Orient the parts as shown in the image**. Again, you can use any filament and I recommend 0.3mm layer height for faster printing.



1.9.2 Part Check

The following steps will require: The handset, the two 3D-printed parts (**Esp_Insert v1.stl** and **Esp_Brace v1.stl**), the ESP with handset cable harness and (optionally) amplifier connected, the ESP antenna and optionally the microphone. Check the picture and verify that you got all parts ready.

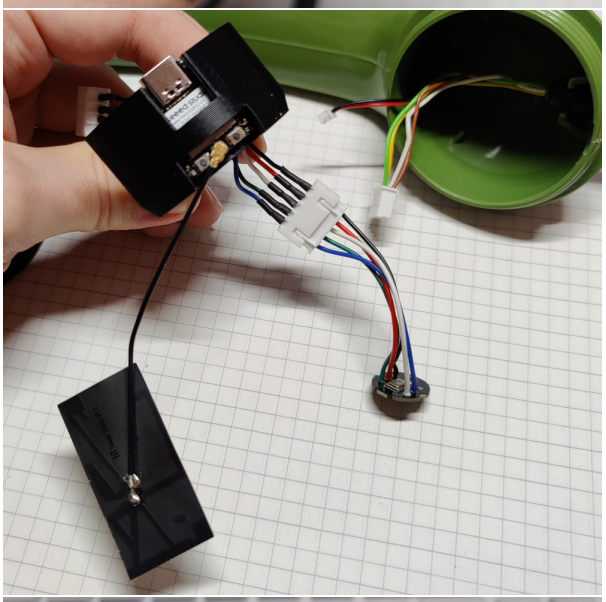
The following steps can be a bit tricky but are definitely possible with only two hands. Take your time.

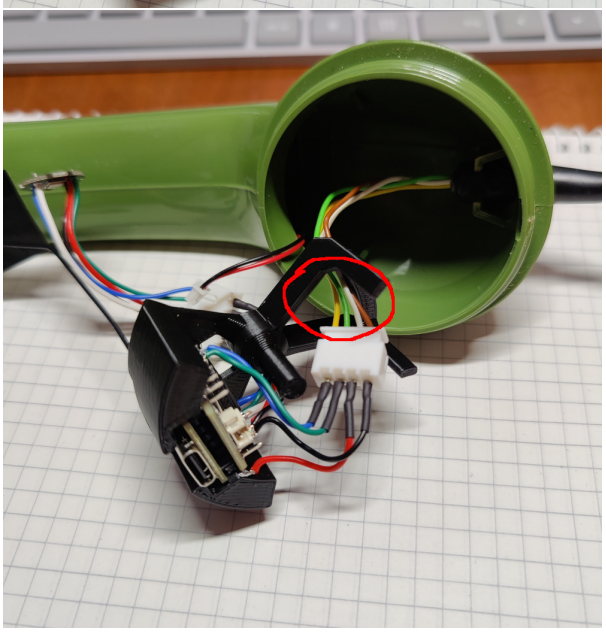
1.9.3 Connecting the ESP

Slot the ESP into **Esp_Insert v1.stl** as shown in the first picture. Make sure it is all the way in and the antenna connector is fully exposed.

Connect the ESP antenna and (optionally) the microphone as shown on the second picture.

Pull the JST connector of the cord through the hole in **Esp_Insert v1.stl** and connect it to the ESP as shown in the third picture.





1.9.4 Fitting the ESP

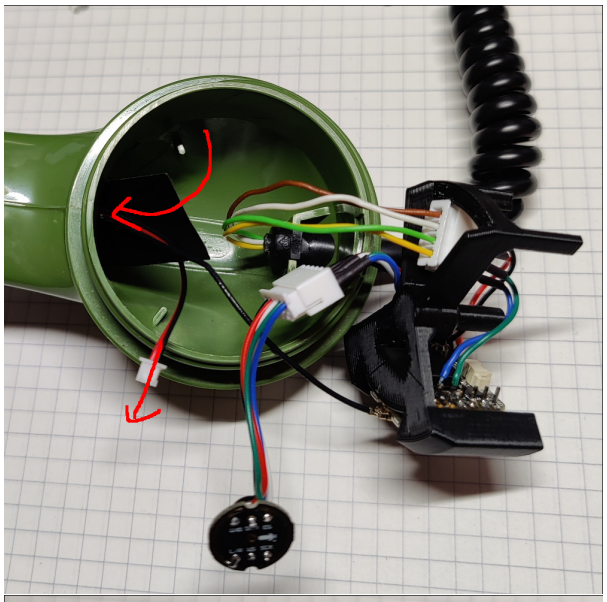
Carefully push the ESP antenna into the cable channel in the handset while making sure to hold onto the (optional) speaker connector as shown in the first picture.

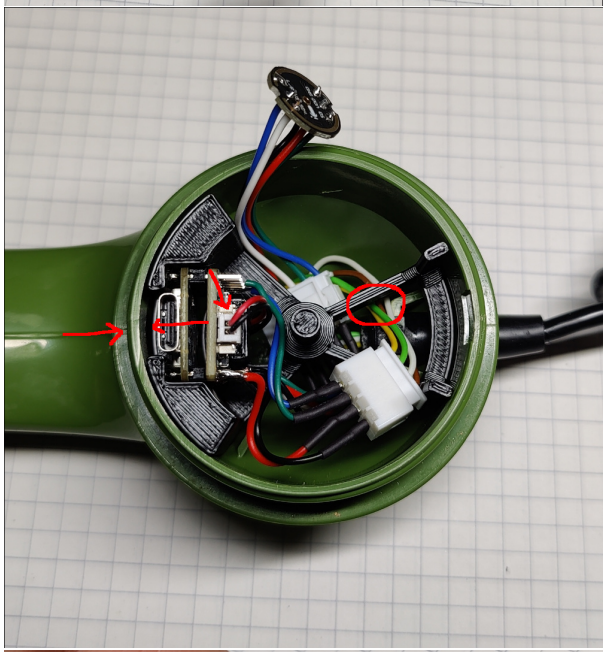
Next, loop the microphone around the bottom of Esp_Insert v1.stl and **pull the optional speaker connector through the smaller hole in Esp_Insert v1.stl** as seen in the second picture.

Connect the speaker to the amplifier if present.

Push the whole assembly into the microphone compartment while **making sure to align the USB-C port of the ESP with the center-seam of the handset**. Also verify that the **cord cables are not pinched** between the handset and the assembly (red circle in second picture).

If you connected a microphone, **push the microphone gently onto the middle arm** as shown in the third picture. The metal housing of the microphone sensor should be positioned right above the arm with no wires pinched in between.





1.9.5 Securing the ESP

Gently push down **Esp_Brace v1.stl**, make sure to align it correctly as shown in the first picture.

If you have a microphone, make sure the microphone stays centered and no cables are pinched by it. Check that the gap between the upper edge of **Esp_Brace v1.stl** and the edge of the handset thread is not larger than 3mm. If it is, there may be cables pinched or the parts are positioned incorrectly.

Finally, finish your build by screwing the cover for the mouthpiece back on. That's it! Your FeTAp-32 hardware is now finished!

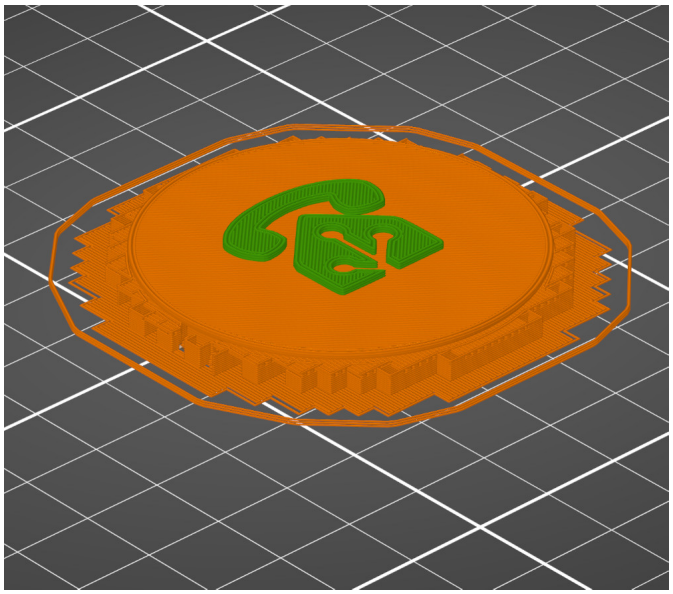
1.10 Emblem installation (optional)

1.10.1 3D-printed parts

If you want to customize the appearance of your FeTAp-32, you can print a new cover emblem for the rotary dial to make people aware that this is not just a normal old rotary telephone.

An emblem with the FeTAp-32 logo is included as **Emblem_Fetap32 v4.stl** which you can use. Make sure you **enable support only on the build-plate**. I also added a filament colour change to make the logo stand out (see picture).

There is also a file called **Emblem_Template v2.step**, which is an empty emblem for you to customize with a 3D modelling program of your choice.



1.10.2 Installing the emblem

Use your two thumbs to rotate the original dial cover counter-clockwise (see first image).

Afterwards, you should be able to just remove the cover (second image).

To install the 3d-printed emblem, just reverse the steps and rotate it clockwise to lock it into place. Make sure the two clips left and right are both locked (third image).

2 Software

2.1 Configuration

The ESP needs to connect to your network over wifi in order to communicate with Home-Assistant. To do that, you need to provide the SSID of your network (*wifi_ssid*) and the password (*wifi_password*). Edit the file **secrets.yaml** accordingly.

You should also set an access point ssid (*ap_ssid*) and access point password (*ap_password*). The access-point settings are used to create a hotspot in case the FeTAp-32 can't connect to your network.

Finally, you can also provide a password for over-the-air updates (*ota_password*), in case you want to modify the firmware without having to connect the FeTAp-32 to your PC again.

2.2 Building the firmware

Clone the repository to your PC and change into the root folder of the repository.

If you do not have esphome installed locally on your machine, follow the instructions on installing esphome locally: https://esphome.io/guides/installing_esphome.html

You should now have an active local virtual python environment (on ubuntu, this is indicated by e.g. *(venv)* in front of your command line prompt, where *venv* is the name of your environment) with esphome installed. You can then build the firmware by running the following command:

```
esphome build fetap32.yaml
```

If everything goes well, you should see an output that ends with

```
INFO Successfully compiled program.
```

Great, the firmware is now ready to be flashed onto the device.

2.3 Flashing the firmware

To flash the firmware, you need to directly connect the ESP with the computer using a USB-C cable. **Always make sure that the USB-c power port is unplugged before doing so.**

To connect the ESP with a computer, simply unscrew the microphone cover of the handset and plug the USB-C cable in as shown in the first picture.

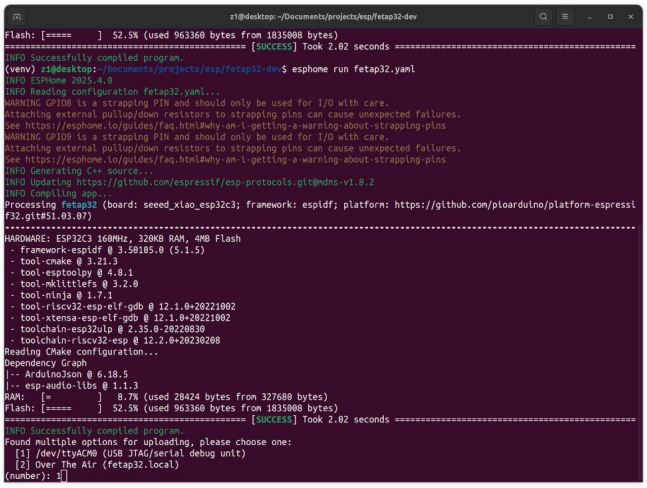
Once the device is connected to your computer, run the following command:

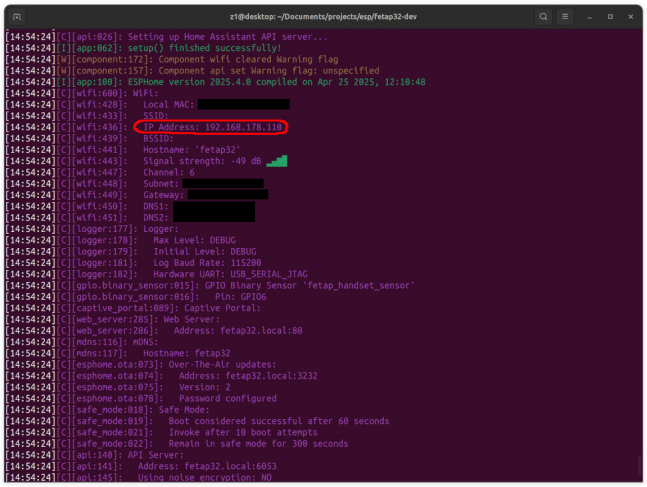
```
esphome run fetap32.yaml
```

You may be asked what upload option you would like to use. In that case, choose *USB JTAG* by entering the corresponding option number (in my example, the USB JTAG option has number one, as seen in the second picture) and confirm by pressing enter.

Once flashing completes, debug information from the ESP will be printed to the console and after a short while, it should connect to your wifi. Once it is connected, it will print the IP address that it got assigned (in my case 192.168.178.110, see second image). Take note of the address your FeTAp-32 got, as we may need it to add it to Home-Assistant.

The flashing process is now finished. You can unplug the USB-C cable from the ESP, screw the microphone cover back on and power the FeTAp-32 back on by plugging into the USB-C power port on the back of the device.





2.4 Home-Assistant Setup

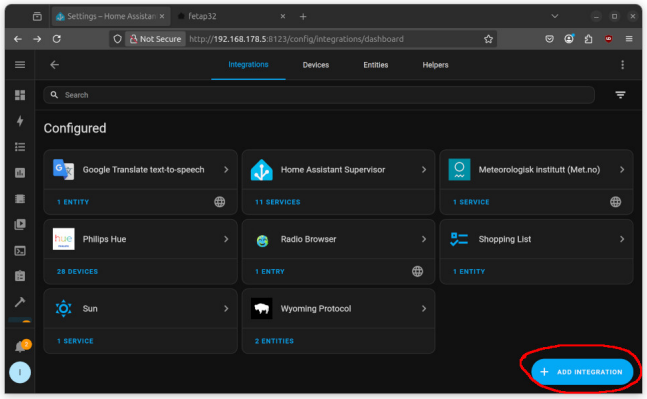
2.4.1 Adding the device

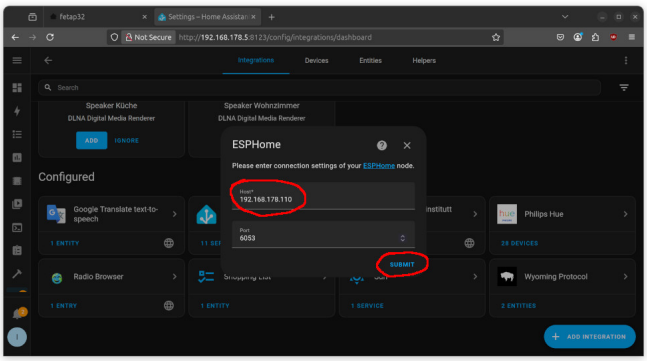
Open your Home-Assistant dashboard and go to **Settings** > **Devices & services** and press **Add Integration** in the lower right corner (see first picture).

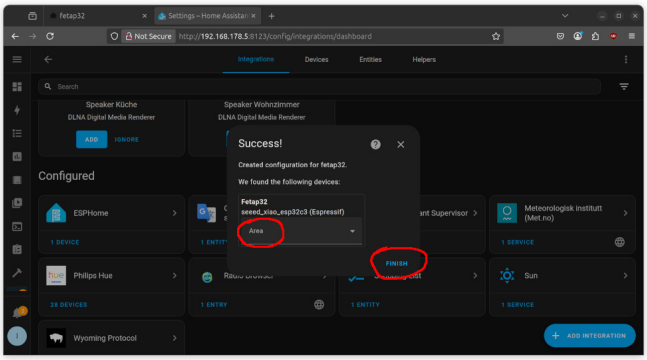
Search for *ESPHome* and select it. In the host field, enter the IP address of your FeTAp-32 (second picture). You can also try to use *fetap32.local* instead, however, that may not work depending on how your network is configured.

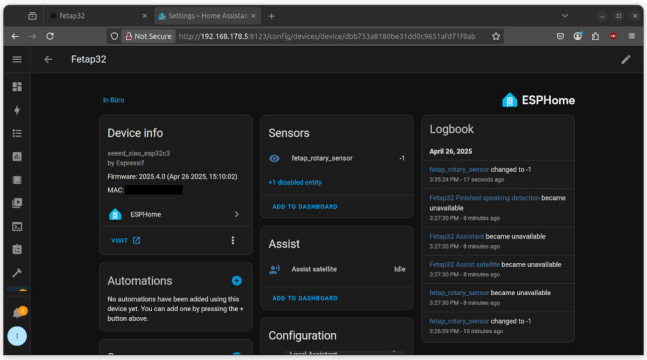
If Home-Assistant could reach your FeTAp-32, you should get a pop-up saying *Success!*. Now select the area for your voice-assistant and press finish to finalize the setup (third picture).

Your FeTAp-32 is now connected to Home-Assistant and ready to use. If you don't have a voice-assistant pipeline setup, you will need to do that first before being able to issue any voice commands (there are plenty of tutorials online). If you connected the rotary dial, try dialing in a number. It should briefly show the number value for the *fetap_rotary_sensor* before switching back to −1.









2.4.2 Rotary Dial Automation (optional)

If you connected the rotary dial sensor, you can setup a custom automation for each number. On the FeTAp-32 device overview screen, click on the blue plus sign next to **Automations**.

Then select **Create New Automation**.

Press **Add Trigger** and select **Entity** and **State**. Search for the **sensor.fetap_rotary_sensor** entity and select it.

Press **Add Condition** and select **Entity** and **State**. Search for the **sensor.fetap_rotary_sensor** entity and select it. In the **State** field, type in the number for that automation (0 - 9).

Press **Add Action** and configure the action you want to perform when that number is dialed. Once you are finished, the automation should look something like the image below. I configured the automation to turn on my light named **Trinity 2** when I dial the number 5. Finish your automation setup by pressing **Save**.