CB external PLL controller (master / slave)

Document version 17-12-2025

Jef Collin

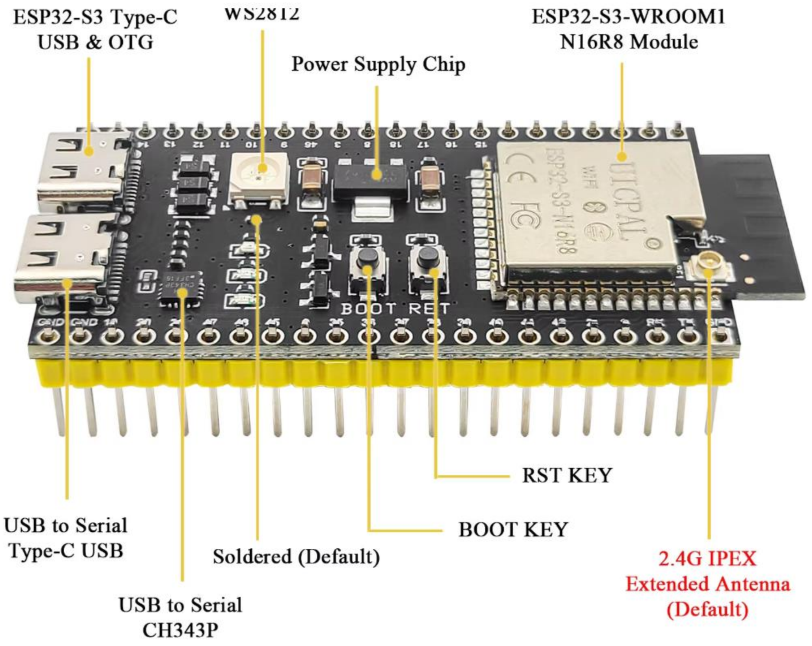
These are some notes on the construction and programming.

In the video you can see that I used the ESP32-WROVER devkit, after adding a few more features in the code the system was a bit unstable due to the high RAM demand from both LVGL and the Wi-Fi / webserver. I could not find a working balance and decided to redesign the master using an ESP32-S3 N16R8 because it has a better RAM segmentation management, a new PCB has been designed which replaced the one in my master unit.

Unfortunately that did not solve the issues but I traced it down to the transfer of the large HTML code block to the Wi-Fi client which used an large intermediate buffer, after changing the code to transfer directly from flash memory where the HTML is located there was no need for this buffer and I could allocate more memory to LVGL. The master is now stable in all modes .

The Master unit

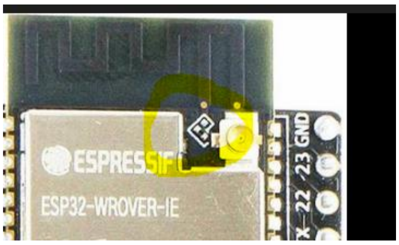
The master unit is very simple and uses a variant of the ESP32-S3 devkit module which has a IPEX connector for an external WiFi antenna.



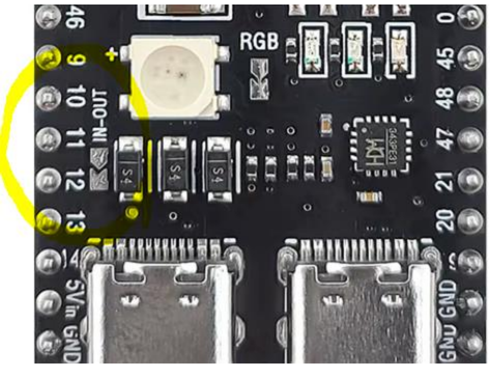
There are a few variations of this board, make sure it has the correct pinout and width.

For programming use the USB connector marked “USB to serial”.

Some versions have a selector for the antenna, as shown below there are 3 pads and only 2 are connected via a tiny 0 ohm resistor, this is the switchover between internal antenna and external. In order to use the external antenna connection you have to make sure the tiny resistor is between the left pad and the bottom right pad leading up to the connector. You can remove the resistor with a soldering iron but it is nearly impossible to place it back so a piece of wire is ok.



The master needs a decent 5v power supply, In case you want to test the project with only the USB cable connected short a solder bridge marked IN-OUT, this will provide 5V to the 5V pin of the module which is connected to the rest of the circuit. Warning: if you start in Wi-Fi mode (pressing the top encoder while powering on) and you have a poor quality or long USB cable, the USB voltage can sag to a point that it will cause a brown out detection and the esp32 will reboot, either use an external supply or keep the button pressed until the second reboot, it will work fine after that.



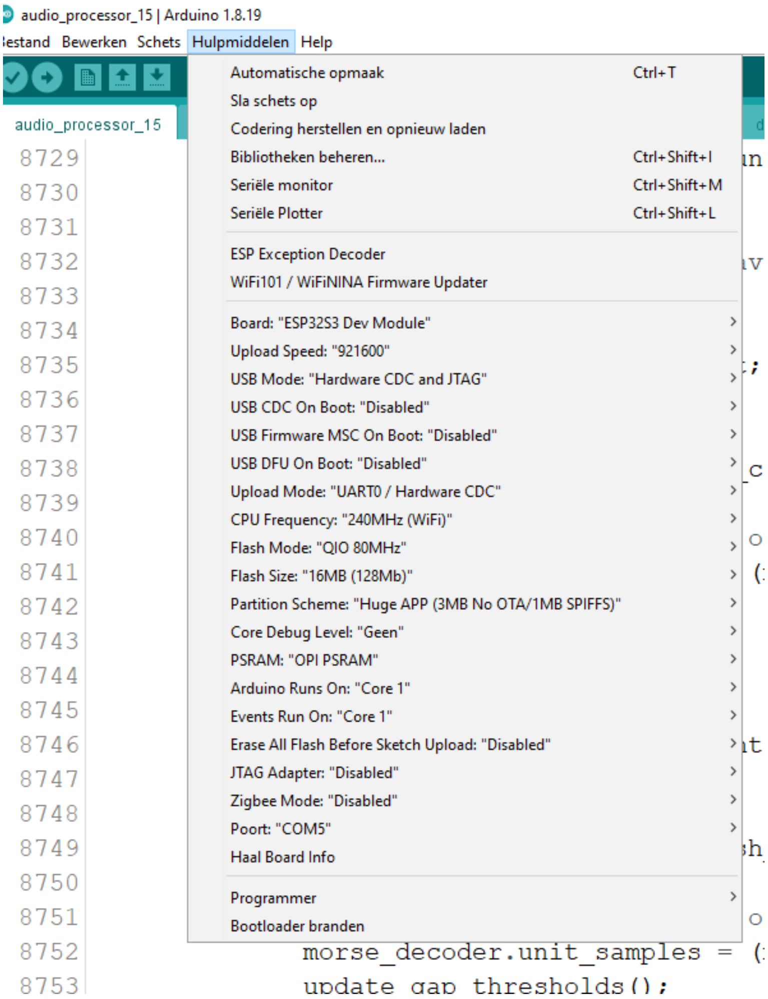
Programming the ESP32-S3 devkit

The ESP32 can be programmed in the usual way using the Arduino 1.8.19 development environment.

I strongly advise to use this version of Arduino IDE, install it in portable mode and install the ESP32 add-on libraries, in my modular tester repository you can find a PDF with instructions on how to setup.

Important: since the LVGL configuration is different from the speech processor and audio processor you need to setup the master in a separate portable setup, same for the slave unit code.

Select **ESP32S3 Dev Module** as board, make sure the other settings are correct (partition and PSRAM are important settings).


audio_processor_15 | Arduino 1.8.19
Bestand Bewerken Schets Hulpmiddelen Help
audio_processor_15
Automatische opmaak Ctrl+T
Sla schets op
Codering herstellen en opnieuw laden
Bibliotheken beheren... Ctrl+Shift+I
Seriële monitor Ctrl+Shift+M
Seriële Plotter Ctrl+Shift+L
ESP Exception Decoder
WiFi101 / WiFiNINA Firmware Updater
Haal Board Info
Programmer
Bootloader branden

Board: "ESP32S3 Dev Module"
Upload Speed: "921600"
USB Mode: "Hardware CDC and JTAG"
USB CDC On Boot: "Disabled"
USB Firmware MSC On Boot: "Disabled"
USB DFU On Boot: "Disabled"
Upload Mode: "UART0 / Hardware CDC"
CPU Frequency: "240MHz (WiFi)"
Flash Mode: "QIO 80MHz"
Flash Size: "16MB (128Mb)"
Partition Scheme: "Huge APP (3MB No OTA/1MB SPIFFS)"
Core Debug Level: "Geen"
PSRAM: "OPI PSRAM"
Arduino Runs On: "Core 1"
Events Run On: "Core 1"
Erase All Flash Before Sketch Upload: "Disabled"
JTAG Adapter: "Disabled"
Zigbee Mode: "Disabled"
Poort: "COM5"

```
    morse_decoder.unit_samples = (
    update_gap_thresholds();
```

Following libraries have to be installed, use the same version number to avoid compilation errors due to breaking changes between library versions. Once it compiles without errors you might try to upgrade the libraries. You can install them via the library manager, some that are not available you can find on github.

LVGL 9.4.0

LovyanGFX 1.2.9

SD (sparkfun)

ESPAsyncWebServer 3.8.1

AsyncTCP 3.4.9

Note on LVGL, a configuration include file lv_conf.h is provided, you need to copy that to the portable->sketchbook-> libraries folder (not to the lvgl folder itself). This contains configuration settings for LVGL 9.4.0, if you use another version you need to copy the template from the lvgl library folder, change the name to lv_conf.h and edit it, use the old version and compare, very important settings are:

#if 1 /* Set this to "1" to enable content */ (don't ask why but default it's on 0 and you get lots of errors)

#define LV_MEM_SIZE (70 * 1024U)   /**< [bytes] */ (this defines how much RAM space is reserved, too low it will hang during screen refresh or not compile, too high it will reduce the space for variables and cause compilation or other errors)

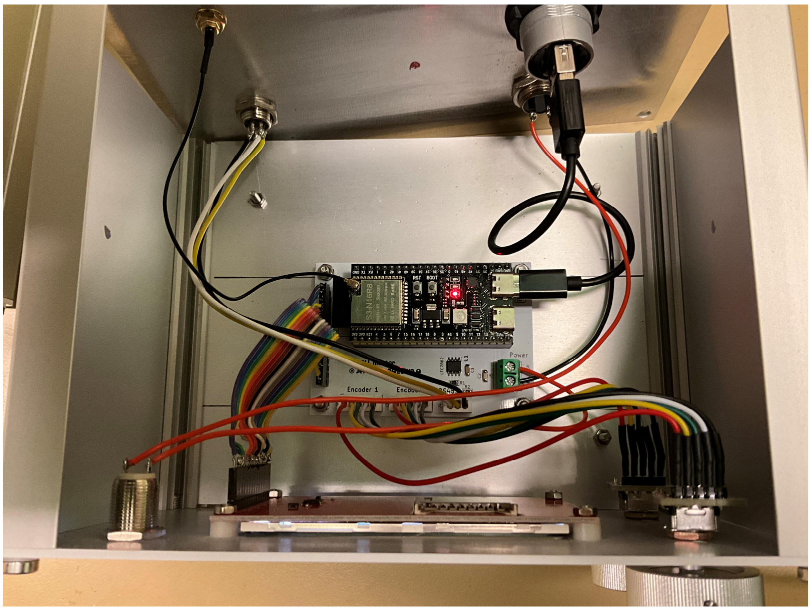
#define LV_FONT_MONTSERRAT_8 0

You need to set all the Montserrat fonts that are used by the program to 1.

Compilation of the program will take some time the first time, when compiling again with just code changes in the main program it will be shorter due to caching of compiled libraries.

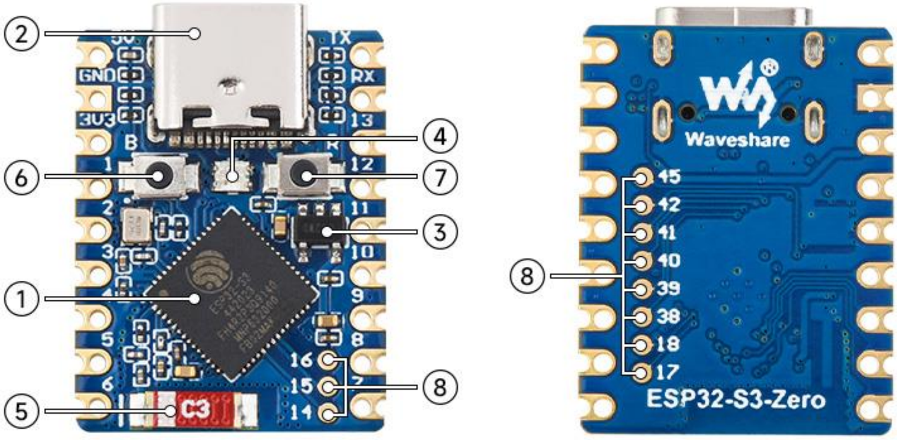
When uploading make sure Serial Monitor and Serial Plotter windows are closed otherwise the ESP32-S3 could go into download mode again after downloading.

A view of the new master PCB.



The Slave unit

The slave unit uses an ESP32-S3 zero.



As explained above you need to setup a separate portable development folder. The older "cb pll controller" project uses the same module, in it's repository you can find a document explaining the setup and programming.

Following libraries have to be installed:

Adafruit_MCP23X17 2.3.2

Adafruit_MCP4725 2.0.2

Note: MCP4725 comes in 4 versions with different I2C addresses so either pick the same as I did or change the address in the code.

Now the bad news…

This is (for now) a one-off personal project, I did not take many notes or pictures for a step by step guide as I don't think many people are brave enough to build this although several people have built the previous design and I installed it in several radios.

The video should explain a few things on how to install the slave PCB in a radio, you can also refer to the installation document in the older cb pll controller repository since it is based on that design except the PLL wiring is a little different, instead of removing the channel switch all traces to the PLL pins are cut to isolate the pins and both sides are connected to the controller PCB.

In the video I talk about adding a 100 ohm resistor between the ground connection of the RS485 link to break the ground loop, this resistor is now included on the master PCB.

Since I forgot the connection for the transmit detection and the parts were added on the bottom side of the PCB, they are now included in the released slave PCB design.