ESP32 LCD controller Wiki

Project uses VSCode frontend, running the Arduino CLI framework which is invoking esp-idf framework. The esp-idf framework is provided by Espressif and is based on FreeRTOS.

I used Arduino to have accesss to the readily available TFT and touch controller driver libraries as well as the LVGL and AsyncWebsockets libraries.

IMPORTANT!

When exiting the VM please use power off menu from inside the VM OS before closing the VM session. Otherwise you will corrupt the VM eventually. Using this method will prevent this.

1. Open VS Code, project is from /osboxes/Documents/PlatformIO/projects. Select the project you want to open. The Wilson Walton one is 210915\_1634-esp-wrover-kit

2. From within VSC, Click on Explorer on the left menu ribbon (2 Document Icon)

This will show the project and files within

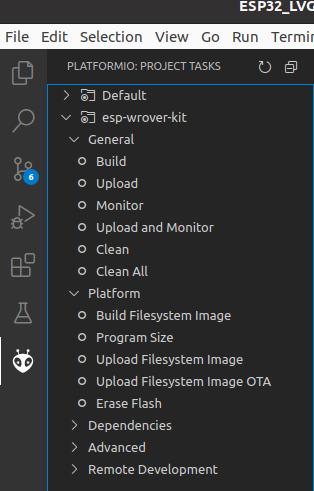
3. To build & upload to board there are 2 steps, A. builds the source and uploads the binary for the code and B. uploads the “data” files for the SPIFFS SPI flash file system into the SPI flash via the extension I added in Platform IO.

A. Click the little Ant icon on the left menu ribbon, it will open PlatformIO command menu

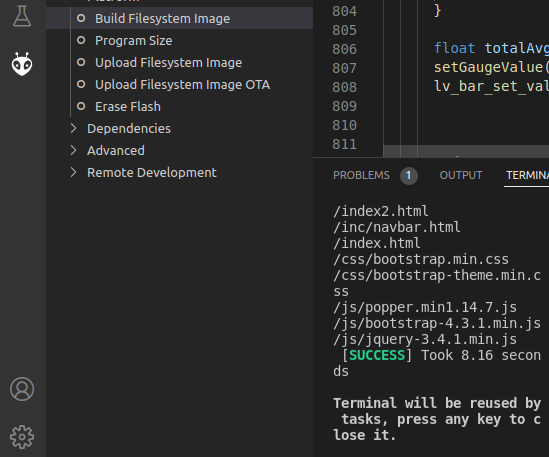
Select Upload and it will build and upload the program binaries

\* You will need to have setup the USB-Serial adapter prior to this. By default it uses *dev*tty/USB0. If you get an error connecting make sure the VM has control of the USB adapter. You can check this in the VM settings. If not give it access.

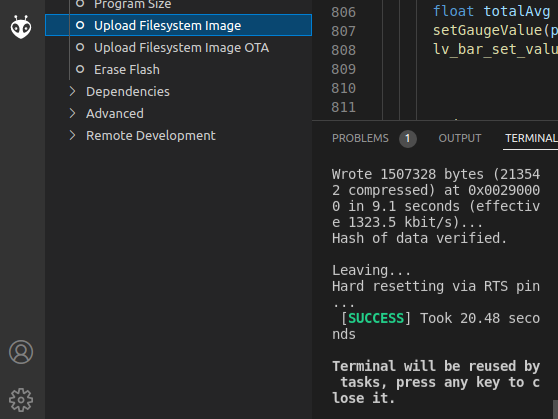
Also you may have to send the command to give Linux RW access to the USB-SER device. See special Info section below



B. Build & Upload Filesystem Image to SPI FLASH



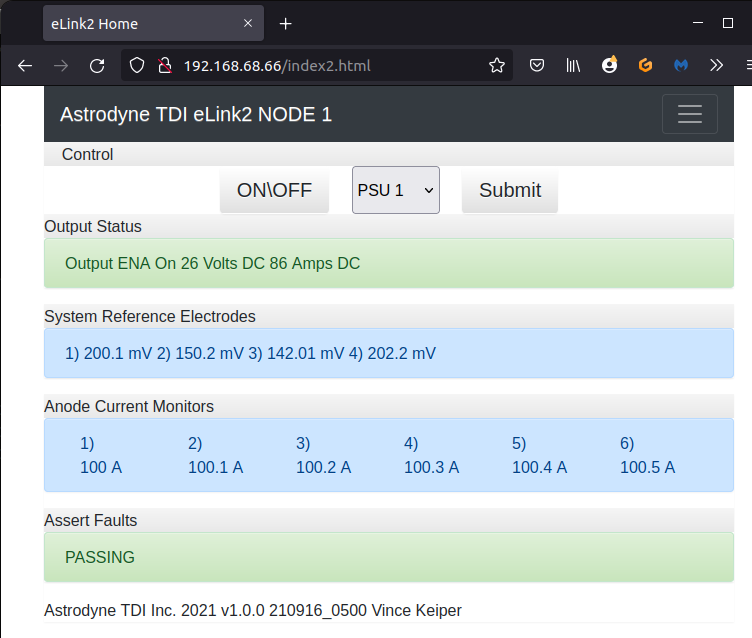
a. Click on the Build Filesystem Image menu item and it will take the folder and files from the “data” folder in the project and combine them into a single image file. From here it can be uploaded to the SPI Flash using this same extension.

b. Upload Filesystem Image, click on the menu item of the same name And look for results in terminal window within PlatformIO/VS Code. Look for SUCCESS

4. Wifi Connection Manager

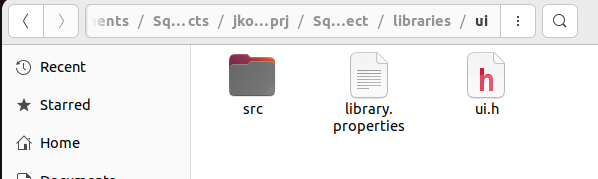
Select an active SSID from the WiFi dropdown and then enter the PWD (you can program the default password in FW to make this faster)

5. Website The main website is the assigned ip address /index2.html

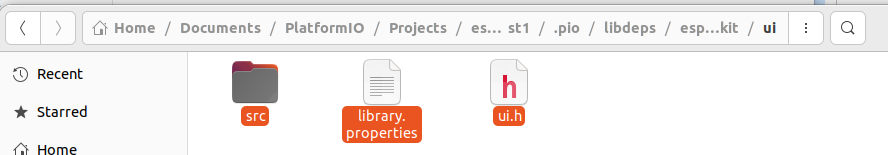


6. Squareline Studios (UI wysiwyg creator)

A. After creating UI select the “Export UI” menu, this will create all the files for that project. Go into the folder location and then copy the “src” folder and ui.h and lib properties files



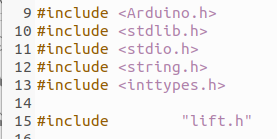
B. Move file to PlatformIO project in the .pio/libdeps/esp-wrover-kit/ui folder.



C. The ui\_events.c file needs to be compared against the ui\_events.cpp file and update the .cpp file with any new or modified code from the squareline editor made. If no events were modified, added, or removed you will not need to do anything.

\*The 1st time a project is created the ui\_events.c file needs to be renamed to ui\_events.cpp to be compiled properly. Also will need to add paths to Arduino.h and any other libraries that are usd within the events file.

Example:



7. PlatformIO Custom Boards File for ATDI WiFi Controller using the ESP321-S3 N8R2

Copy /docs/wt32\_plus\_esp32\_s3\_N8R2.json to the platformIO /espressif32 boards cfg file locations.

Default WIN location C:\Users\YOURUSERNAME\.platformio\platforms\espressif32\boards

8. Partition Table:

TODO: VGK

Special Info:

**To give USB serial port write access**

When you get "failed to open serial port….” its because Linux needs RW access and this needs to have this command sent via the Linux terminal

sudo chmod a+rw /dev/ttyUSB0

VM Login

sudo/ root Password 31911234

**To add a private library/ folder to the project** add into the /libs folder under the root of the project folder. To get platformIO to compile any source or find the headers you must add a line to the platformIO.ini file to tell it to add this folder with the “lib\_extra\_dirs” flag. I have done this already in this project.

lib\_extra\_dirs =

lib

Cant find <functional> This can happen when the source file is .C instead of .cpp. Change source file to .cpp and you will not get this error.