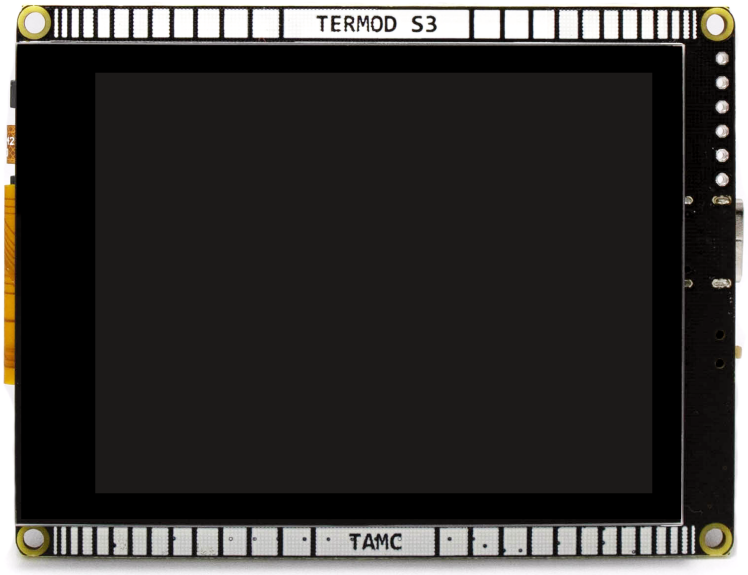
Termod S3

Release 1.0.0

TAMC

Sep 28, 2022

CONTENTS

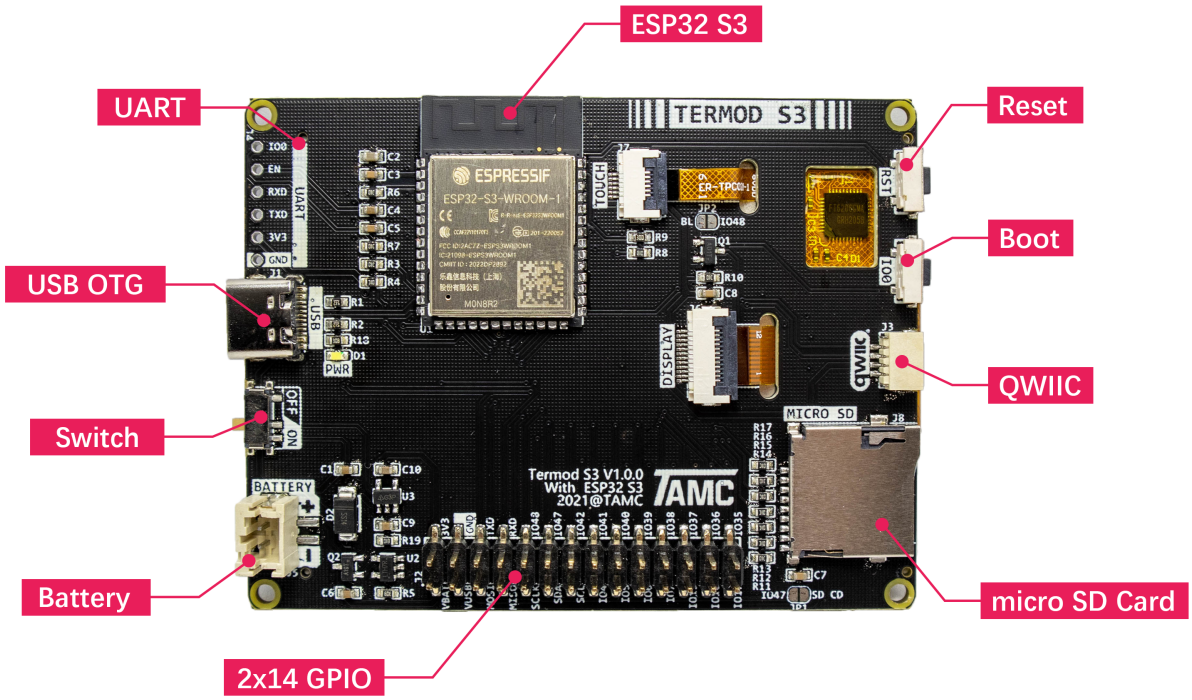


Termod S3 is a ESP32 S3 development board with 2.8 inch capacitive touch diaplay.

CHAPTER

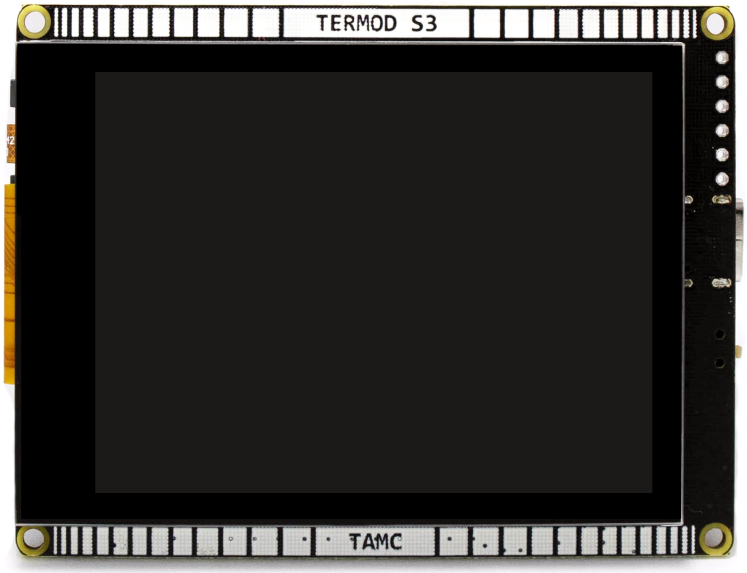
ONE

SPECIFICATIONS

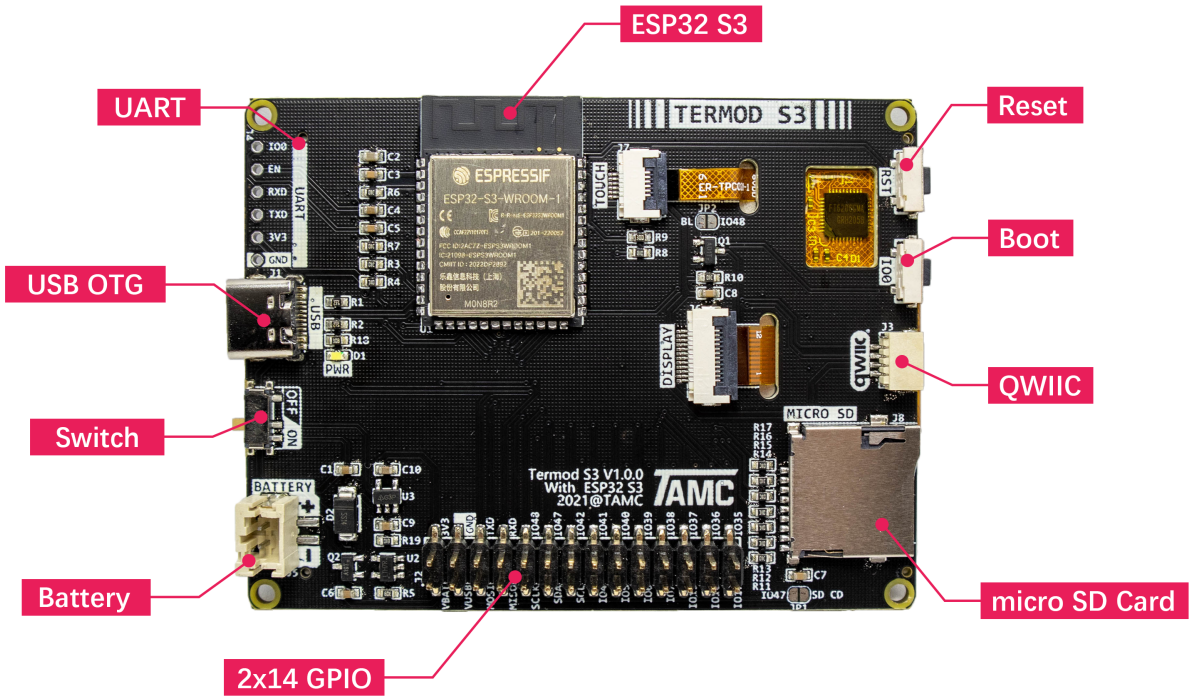


Power	PH2.0 2P, 3.3V-6V, Min 3.55V@600mA
USB	USB 2.0 Type-C, PD 5V
MCU	ESP32 S3
Flash	8MB
PSRAM	2MB
Display	2.8 Inch 320x240 IPS, SPI
Touch	FT6206 Capacitive IIC
Size	76x58mm
Mounting Holes	M2 x 4
SD Card	Micro SD with SPI Interface
Buttons	IO0 and Reset button

1.1 Hardware



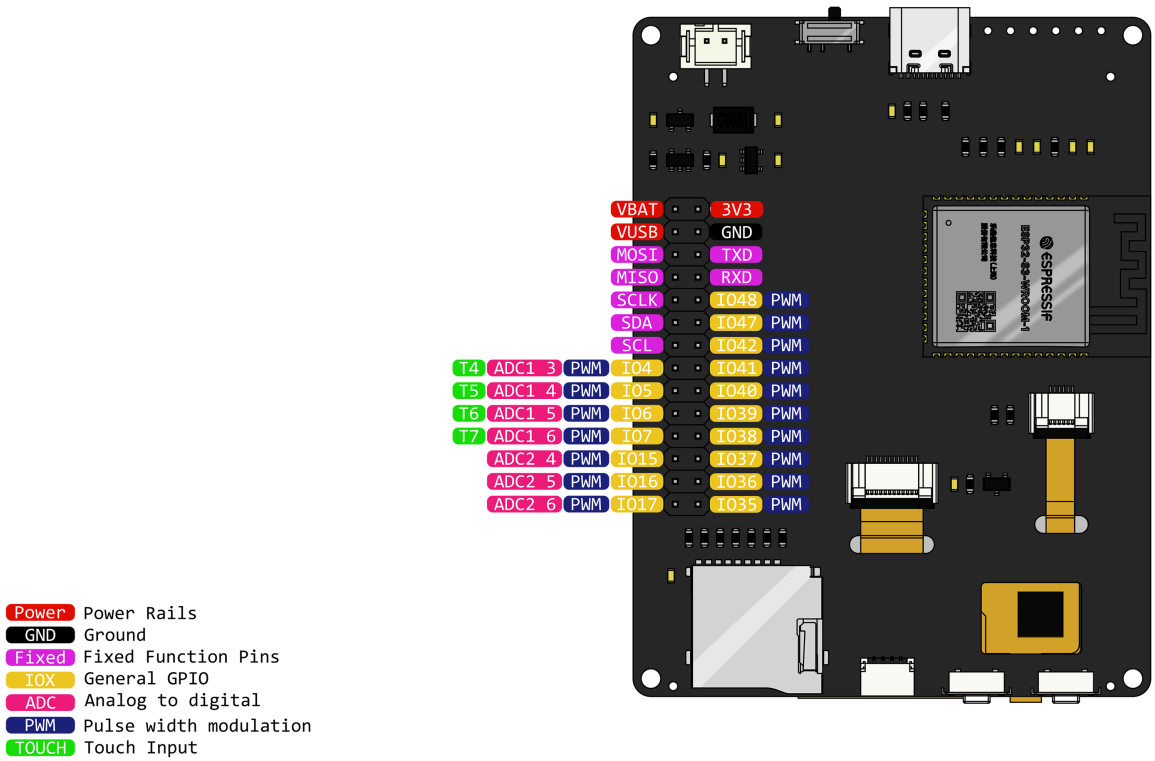
1.1.1 Specifications



Power	PH2.0 2P, 3.3V-6V, Min 3.55V@600mA
USB	USB 2.0 Type-C, PD 5V
MCU	ESP32 S3
Flash	8MB
PSRAM	2MB
Display	2.8 Inch 320x240 IPS, SPI
Touch	FT6206 Capacitive IIC
Size	76x58mm
Mounting Holes	M2 x 4
SD Card	Micro SD with SPI Interface
Buttons	IO0 and Reset button

1.1.2 Pinout

Termod S3 Pinout



Power Power Rails
GND Ground
Fixed Fixed Function Pins
IOX General GPIO
ADC Analog to digital
PWM Pulse width modulation
TOUCH Touch Input

1.1.3 Pin Assignment

General Pins

ESP32 S3	General
GPIO11	MOSI
GPIO13	MISO
GPIO12	SCLK
GPIO8	SDA
GPIO9	SCL
GPIO1	Battery Level
GPIO2	Charge Detect
GPIO0	Button
GPIO10	TFT CS
GPIO18	TFT D/C
GPIO14	TFT Reset
GPIO48	TFT Backlight (See *Selectable pins*)
GPIO21	uSD CS
GPIO47	uSD Card Detect (See *Selectable pins*)

LCD Pins

ESP32 S3	LCD
GPIO11	MOSI
GPIO13	MISO
GPIO12	SCLK
GPIO10	CS
GPIO18	D/C
GPIO14	Reset
GPIO48	Backlight (See *Selectable pins*)

FT6206 Touch Screeen Pins

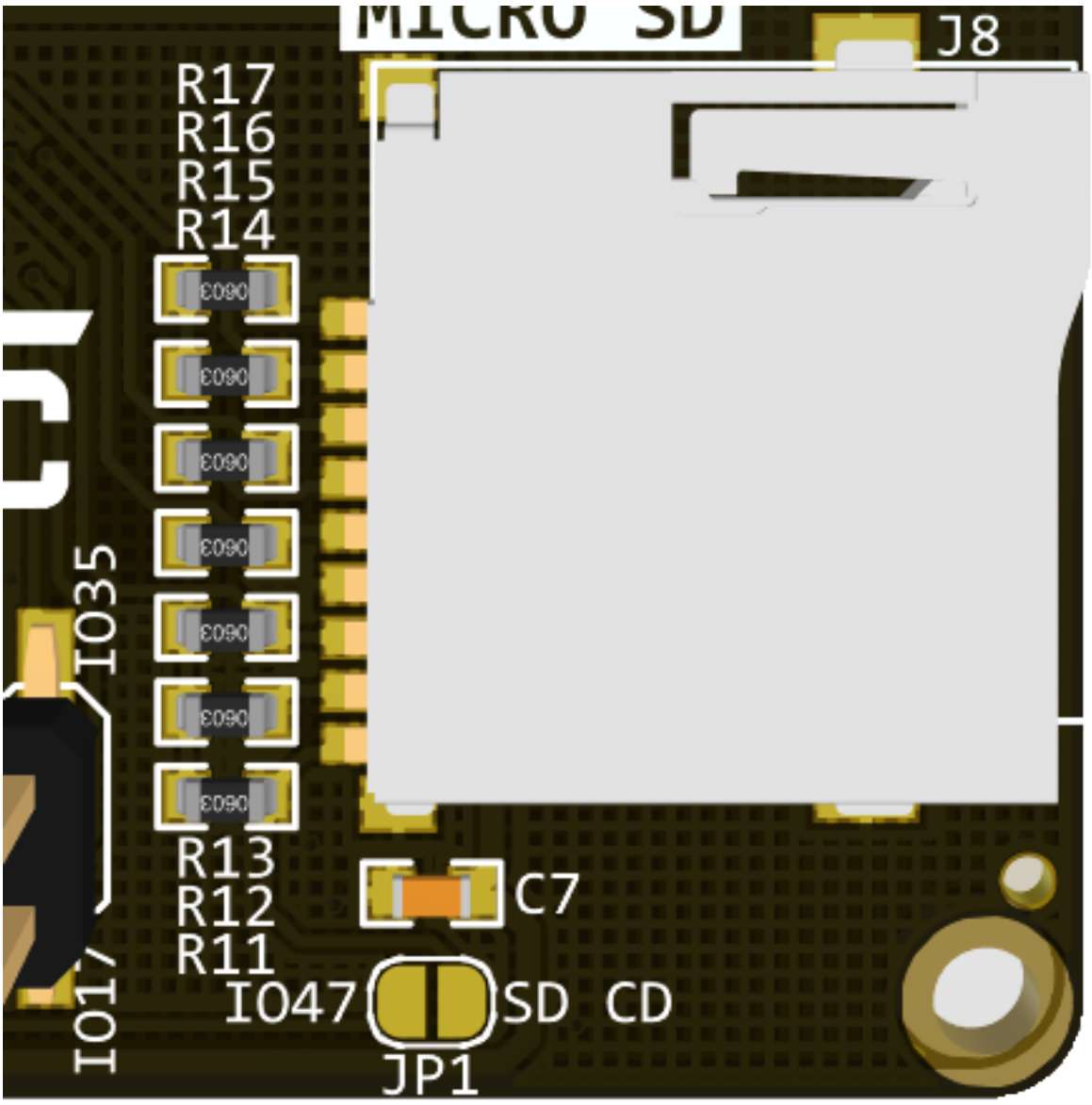
ESP32 S3	FT6206
GPIO8	SDA
GPIO9	SCL
NC	INT
NC	RST

Micro SD Card Pins

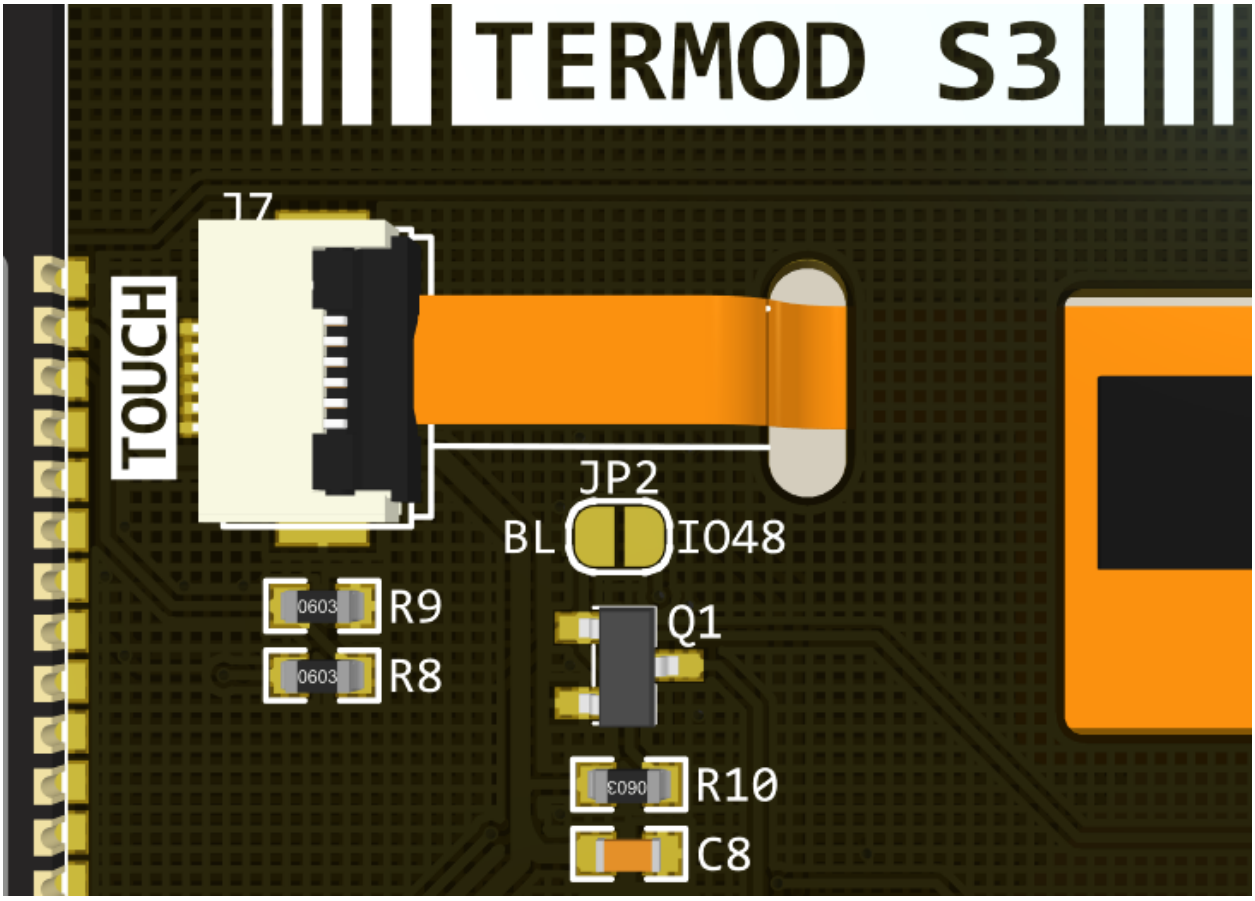
ESP32 S3	Micro SD Card
GPIO11	MOSI
GPIO13	MISO
GPIO12	SCLK
GPIO21	CS
GPIO47	Card Detect (See *Selectable pins*)

1.1.4 Selectable pins

JP1 and JP2 are solder pads for selecting functions.



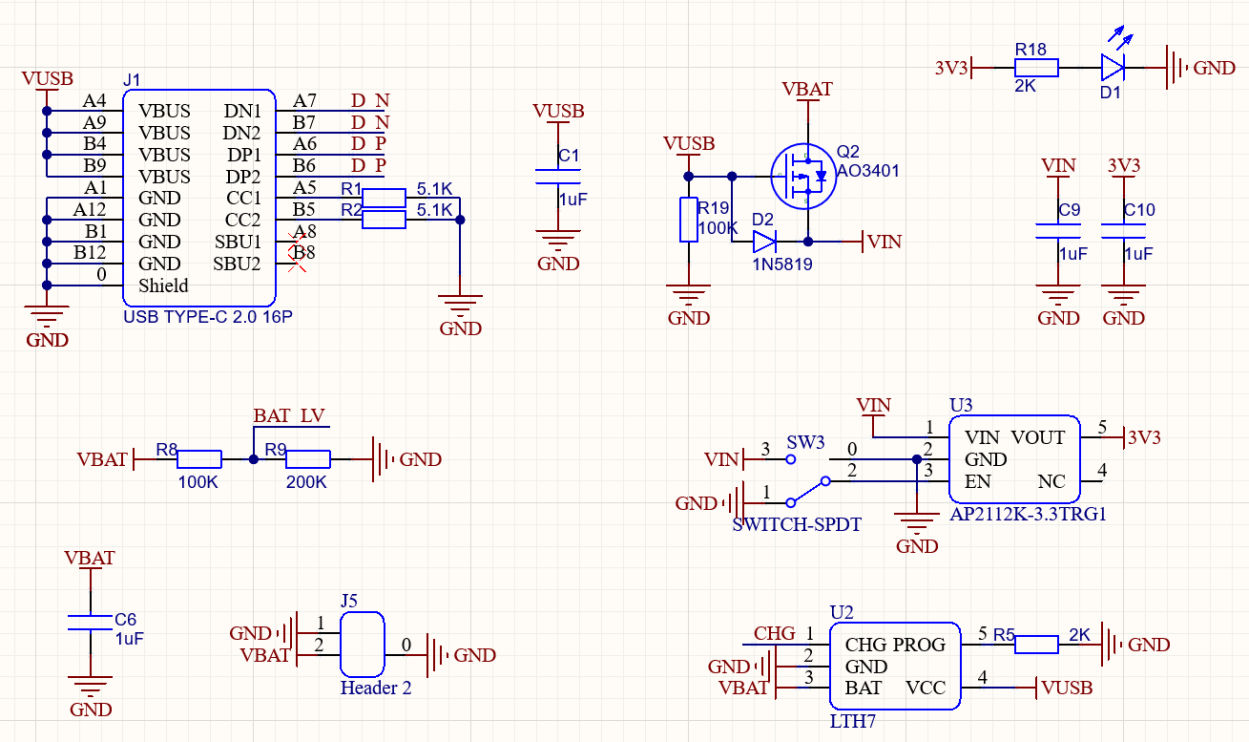
JP1 is for selecting the micro SD card detect pin. If you need to detect inserting a card, you can solder JP1 together, and reads IO47 for card detecting. IO47 will be pulled LOW when a card is inserted.



JP2 is for selecting the TFT backlight pin. If you need to control the backlight, you can solder JP2 together, and controls IO48 for backlight control. Set IO48 HIGH to turn on backlight.

1.1.5 Schematic

Power management



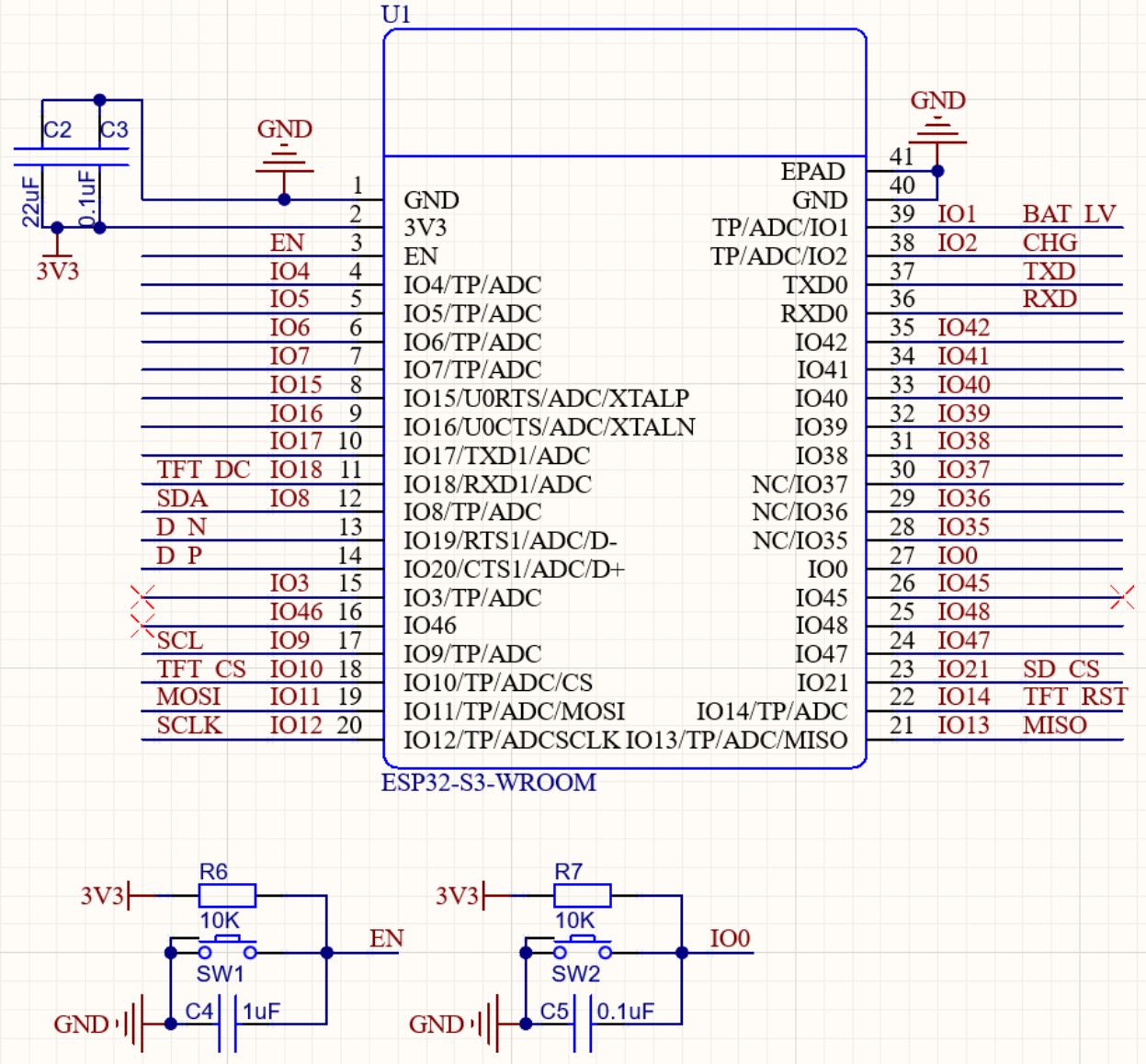
Power includes 2 inputs: 5V USB Type C and battery, joined together with a simple power selector, which cuts of the batteries when USB is pluged in.

A 3.3V power indicator LED `D1` to indicate the power status.

A 100K/200K voltage divider divide the battery voltage to IO1 `BAT`.

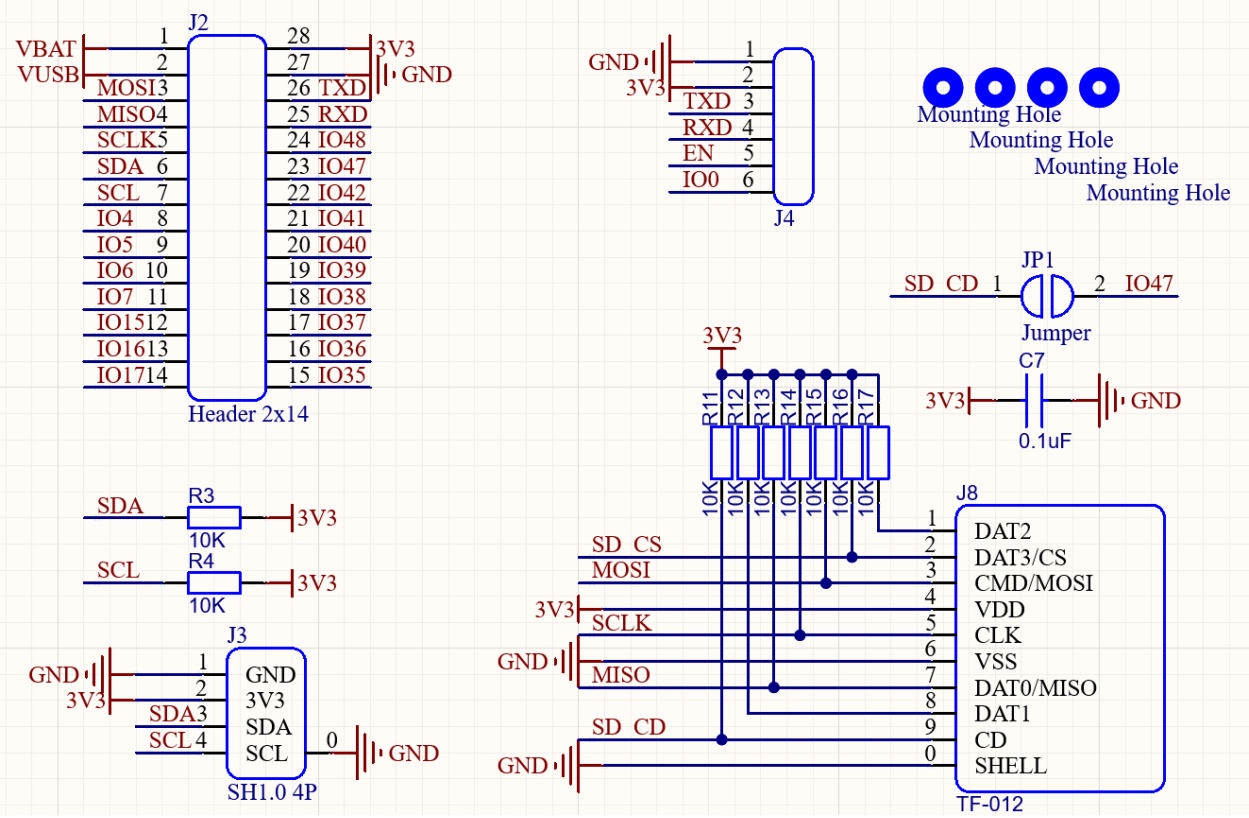
LTC4054 Lithium-ion battery charger. Charge signal is connected to IO2 `CHG`. LOW as charging.

ESP32 S3



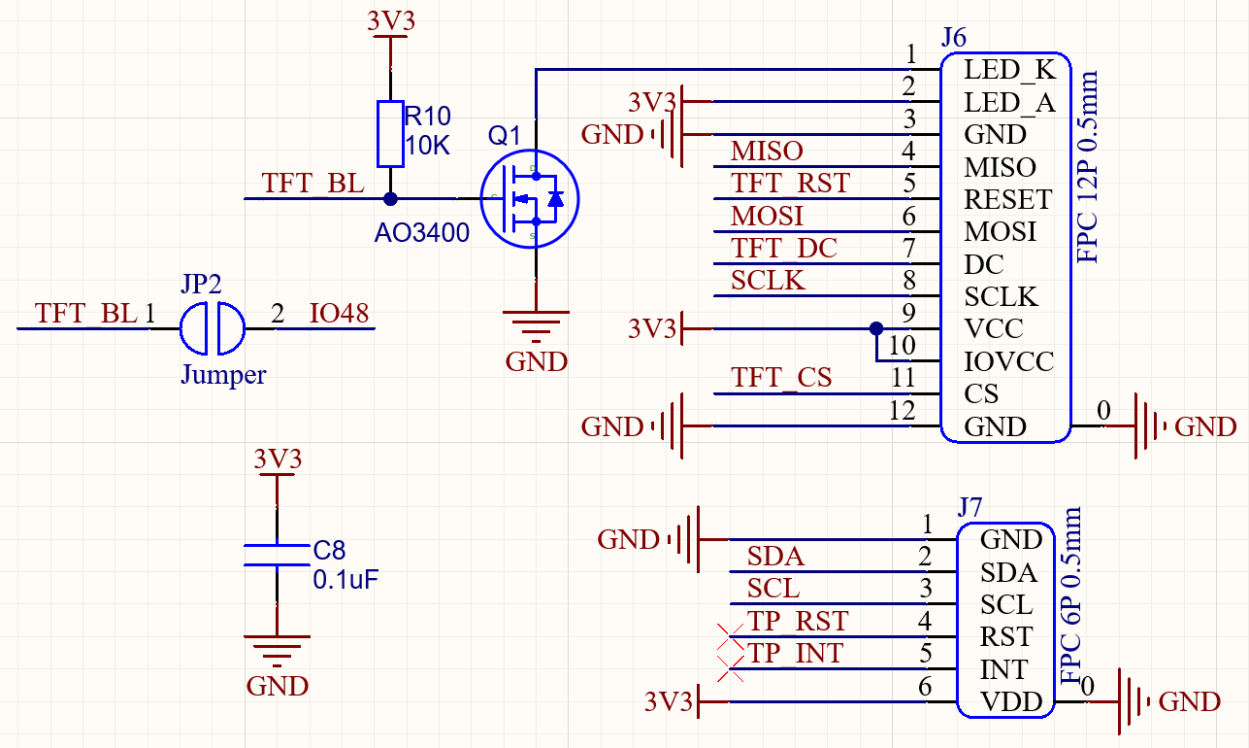
Simple setup for ESP32 S3 with buttons(`IO0` and `EN`).

Connectors



- J2: GPIO breakout connector: pin header 2x14 2.54mm.
- J3: I2C SH-1.0-4P connector compatible with Qwiic and STEMMA QT
- J4: Serial connector with `IO0` and `EN` for easy programming.
- J8: Micro SD Card connector.

Display & Touch Panel



- J6: ST7789V display with SPI interface.
- J7: FT6206 touch panel with I2C interface.
- NMOS Q1 to control the backlight.

1.1.6 Mechanics

- Drawing DXF: `termod-s3-v1.0.0-mechanical-drawing.dxf`
- Drawing PDF: `termod-s3-v1.0.0-mechanical-drawing.pdf`
- 3D Model: `termod-s3-v1.0.0-3d.step`

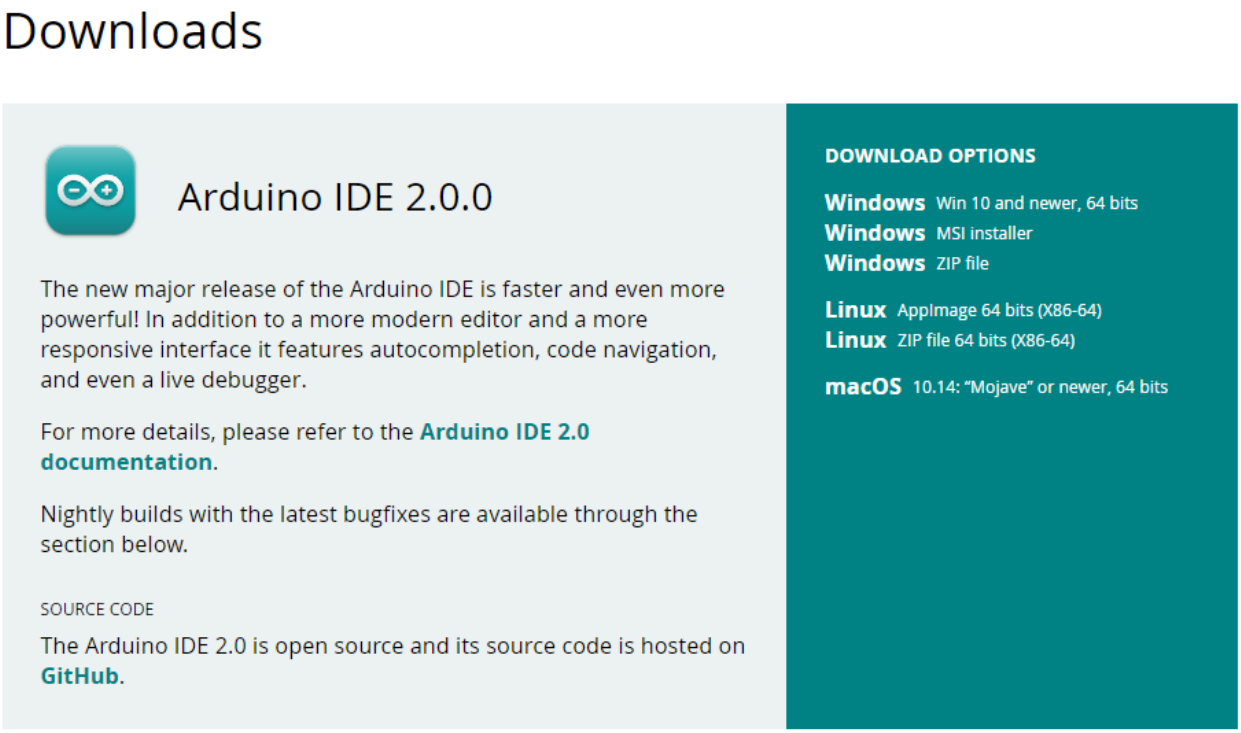
1.2 Arduino Usage

1.2.1 Getting Started with Arduino

Download Arduino IDE

Note: If you already installed Arduino IDE, skip this step. This tutorial is based on Arduino IDE 2.0.0, if yours is older, it is recommended to update.

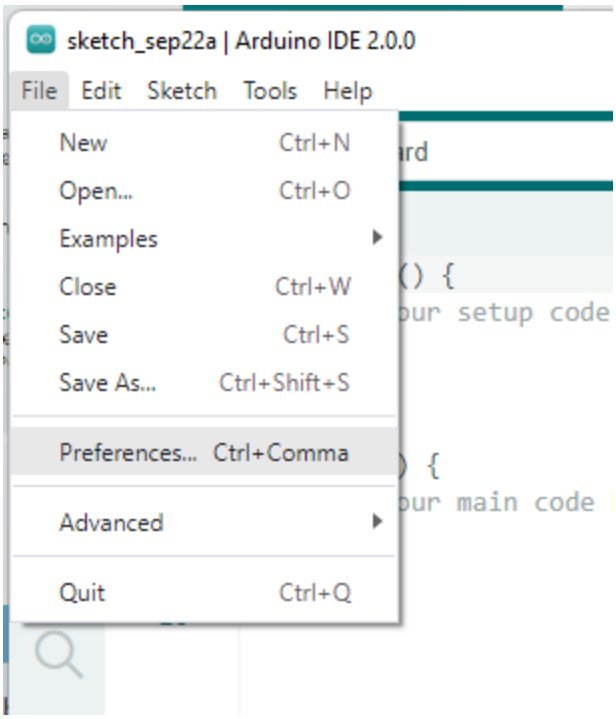
1. Turn to Arduino IDE download page, download and install Arduino IDE.



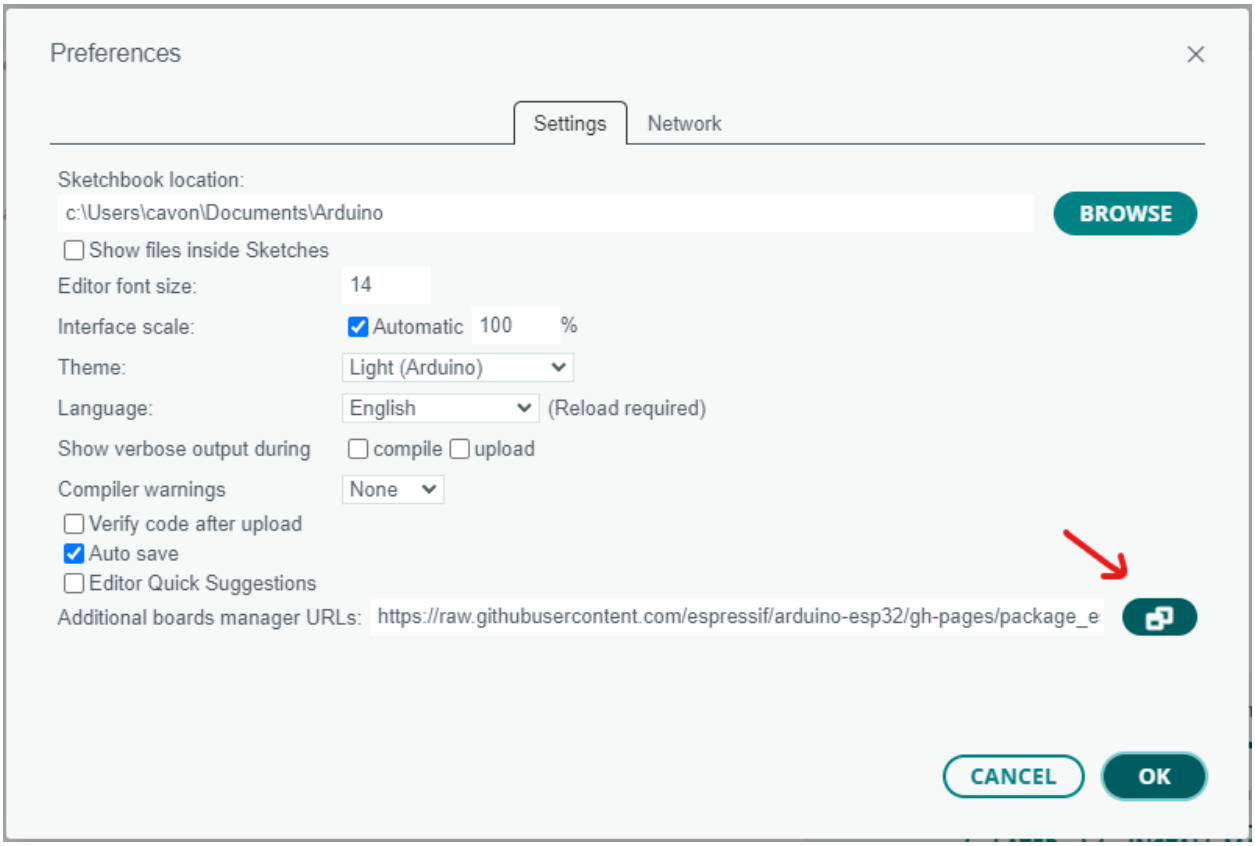
Add ESP32 Series

Note: If you have already added the latest ESP32 core, skip this step, or update it to the latest.

Open Arduino IDE. Click top left **File** menu, select **Preference**. Or for Mac, Click **Arduino** menu, select **Preference**.



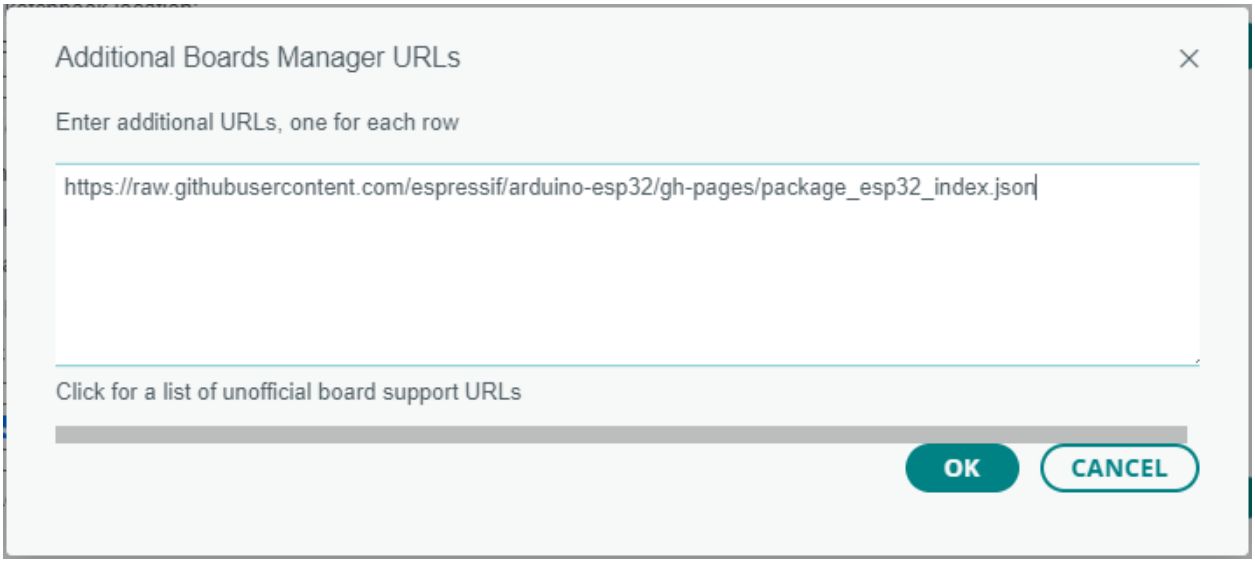
On **Preference** page, click the right-most icon at line **Additional Boards Manager URLS**.



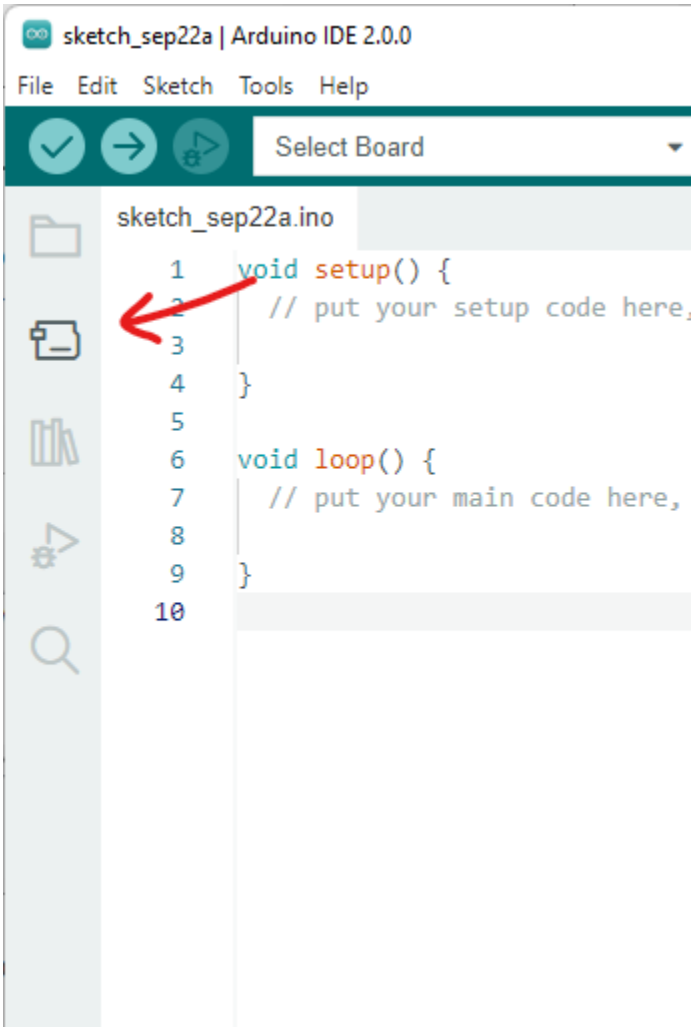
On **Additional Boards Manager URLS** page, Past the link

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

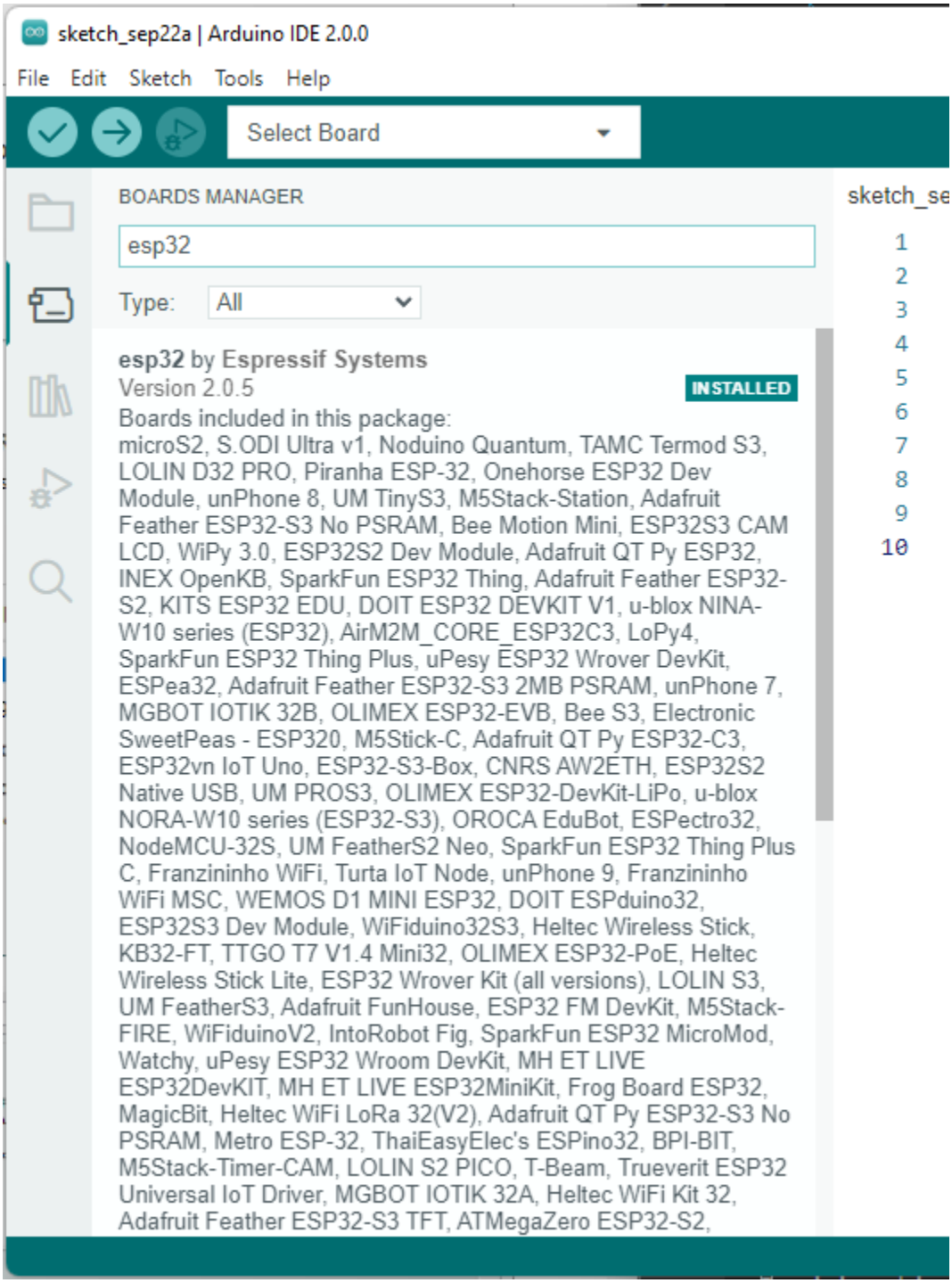
and Click **OK**



Close **Preference** window, Click Board Manager icon on the left



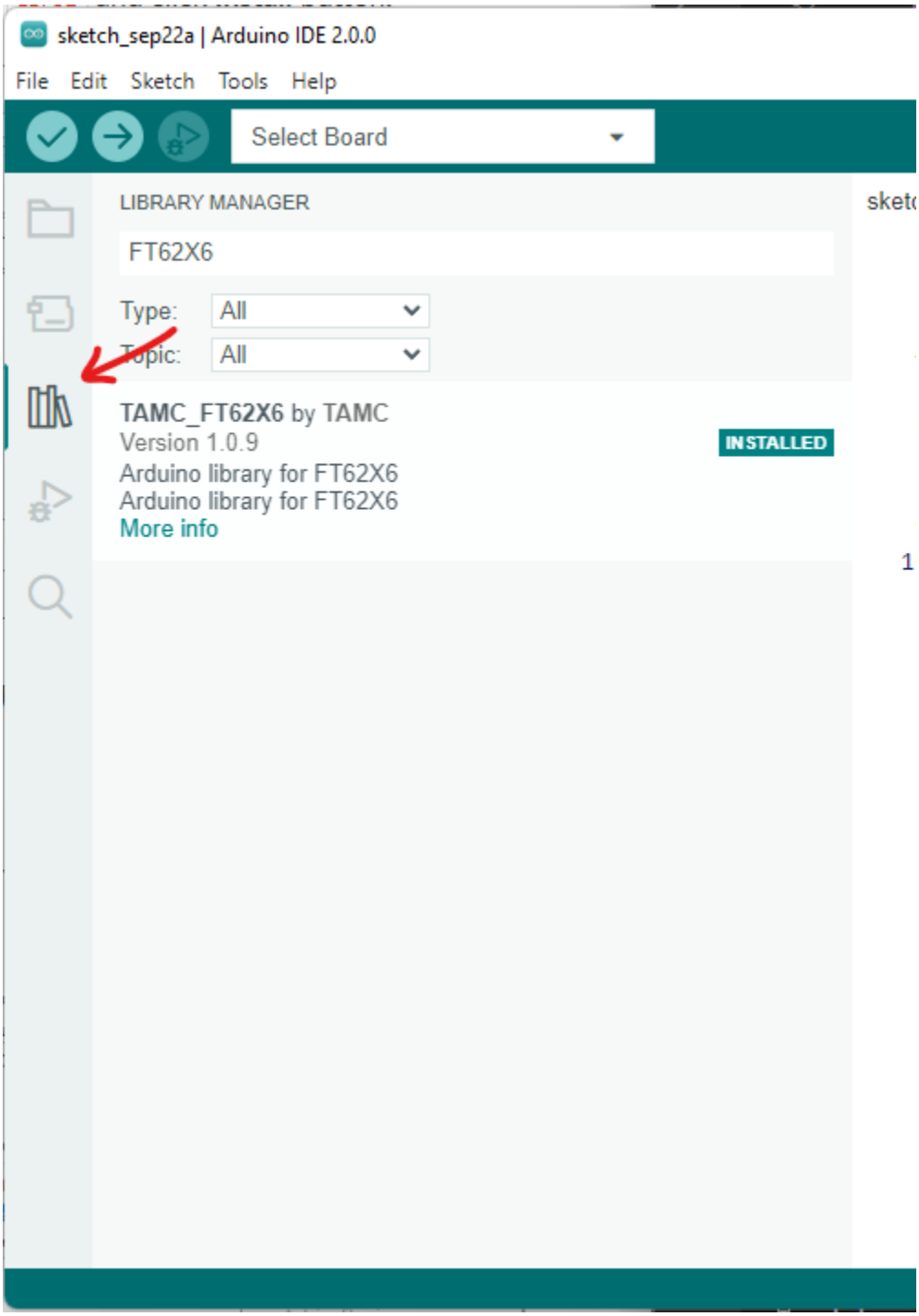
On **Boards Manger** side bar, search for ESP32 and click **Install** button. Or update it if it's not the latest version.


sketch_sep22a | Arduino IDE 2.0.0
File Edit Sketch Tools Help
Select Board
BOARDS MANAGER
esp32
Type: All
esp32 by Espressif Systems
Version 2.0.5
INSTALLED
Boards included in this package:
microS2, S.ODI Ultra v1, Noduino Quantum, TAMC Termod S3, LOLIN D32 PRO, Piranha ESP-32, Onehorse ESP32 Dev Module, unPhone 8, UM TinyS3, M5Stack-Station, Adafruit Feather ESP32-S3 No PSRAM, Bee Motion Mini, ESP32S3 CAM LCD, WiPy 3.0, ESP32S2 Dev Module, Adafruit QT Py ESP32, INEX OpenKB, SparkFun ESP32 Thing, Adafruit Feather ESP32-S2, KITS ESP32 EDU, DOIT ESP32 DEVKIT V1, u-blox NINA-W10 series (ESP32), AirM2M_CORE_ESP32C3, LoPy4, SparkFun ESP32 Thing Plus, uPesy ESP32 Wrover DevKit, ESPea32, Adafruit Feather ESP32-S3 2MB PSRAM, unPhone 7, MGBOT IOTIK 32B, OLIMEX ESP32-EVB, Bee S3, Electronic SweetPeas - ESP320, M5Stick-C, Adafruit QT Py ESP32-C3, ESP32vn IoT Uno, ESP32-S3-Box, CNRS AW2ETH, ESP32S2 Native USB, UM PROS3, OLIMEX ESP32-DevKit-LiPo, u-blox NORA-W10 series (ESP32-S3), OROCA EduBot, ESPectro32, NodeMCU-32S, UM FeatherS2 Neo, SparkFun ESP32 Thing Plus C, Franzininho WiFi, Turta IoT Node, unPhone 9, Franzininho WiFi MSC, WEMOS D1 MINI ESP32, DOIT ESPduino32, ESP32S3 Dev Module, WiFiduino32S3, Heltec Wireless Stick, KB32-FT, TTGO T7 V1.4 Mini32, OLIMEX ESP32-PoE, Heltec Wireless Stick Lite, ESP32 Wrover Kit (all versions), LOLIN S3, UM FeatherS3, Adafruit FunHouse, ESP32 FM DevKit, M5Stack-FIRE, WiFiduinoV2, IntoRobot Fig, SparkFun ESP32 MicroMod, Watchy, uPesy ESP32 Wroom DevKit, MH ET LIVE ESP32DevKIT, MH ET LIVE ESP32MiniKit, Frog Board ESP32, MagicBit, Heltec WiFi LoRa 32(V2), Adafruit QT Py ESP32-S3 No PSRAM, Metro ESP-32, ThaiEasyElec's ESPino32, BPI-BIT, M5Stack-Timer-CAM, LOLIN S2 PICO, T-Beam, Trueverit ESP32 Universal IoT Driver, MGBOT IOTIK 32A, Heltec WiFi Kit 32, Adafruit Feather ESP32-S3 TFT, ATMegaZero ESP32-S2,
sketch_se
1
2
3
4
5
6
7
8
9
10

Install FT62X6 Library

This is a library for touch screen.

Open Arduino IDE. Click Library Manager icon on the left, search for `TAMC_FT62X6` and click **Install** button. Or update it if it's not the latest version.

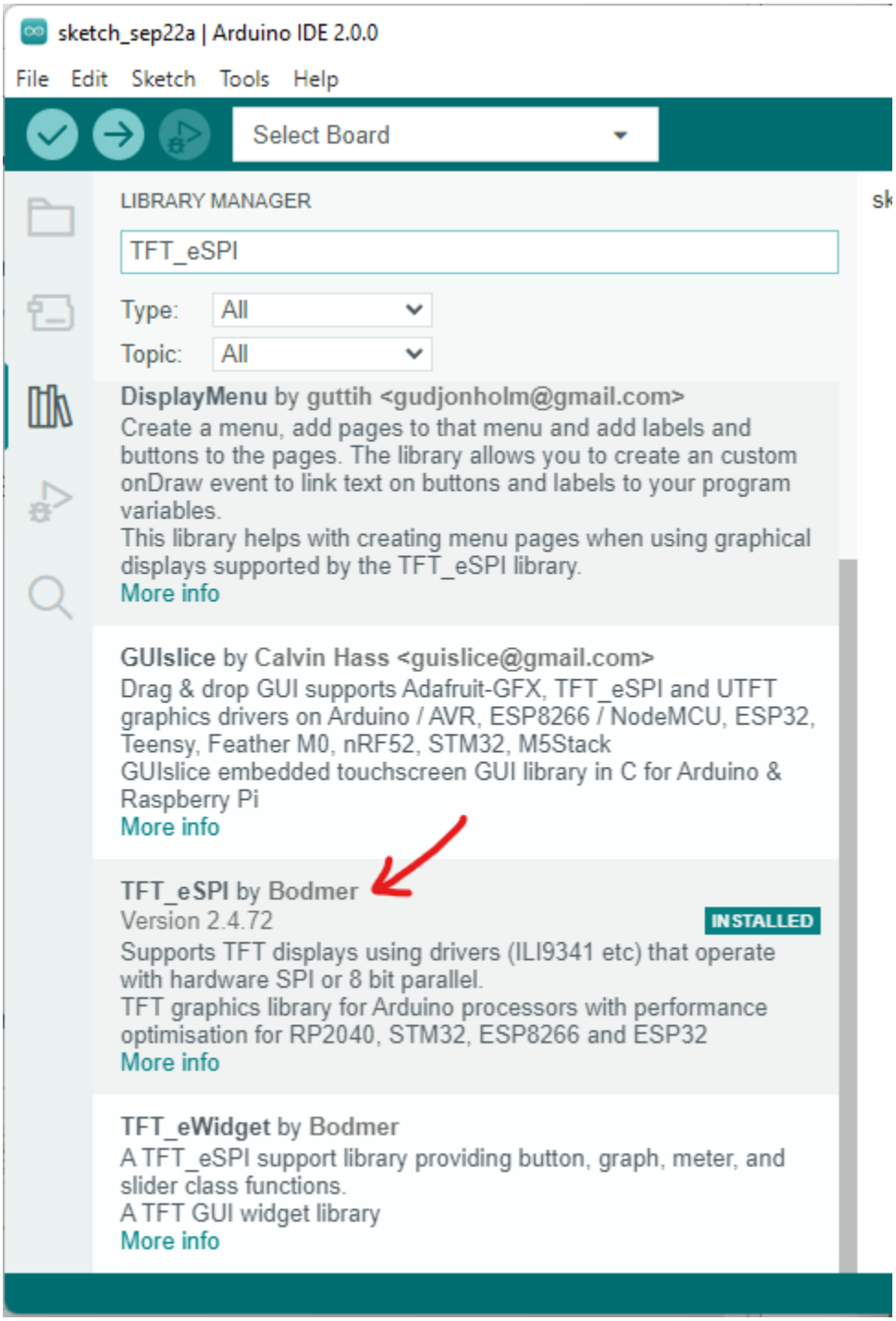


Install TFT_eSPI Library

This is a library for TFT display. There are also other options, but we recommand this one.

Install

Search again for `TFT_eSPI` and click **Install** button. Or update it if it's not the latest version.



Setup for TAMC Termod S3

After install TFT_eSPI Library, open File Explorer and go to Arduino library folder. Usually is under the following folders. If not, checkout **Sketchbook location** in **Preference** of Arduino IDE.

- For Windows: `C:\Users\<USER>\Documents\Arduino\libraries`
- For Mac: `/Users/<USER>/Documents/Arduino/libraries`
- For Linux: `/home/<USER>/Documents/Arduino/libraries`

Open `TFT_eSPI` folder, and open `User_Setup_Select.h` file with the editor you like.

Comment out the line: `#include <User_Setup.h>`, and uncomment the line: `#include <User_Setups/Setup300_TAMC_Termod_S3.h>`, and the file will look like this:

```
...

// #include <User_Setup.h>

...

#include <User_Setups/Setup300_TAMC_Termod_S3.h>

...
```

Save and close the file. Download Termod S3 setup and copy it to *User_Setups* folder.

`Setup300_TAMC_Termod_S3.h`

Done, but no need to close the File Explorer yet, you will need it later.

Install LVGL Library (Optional)

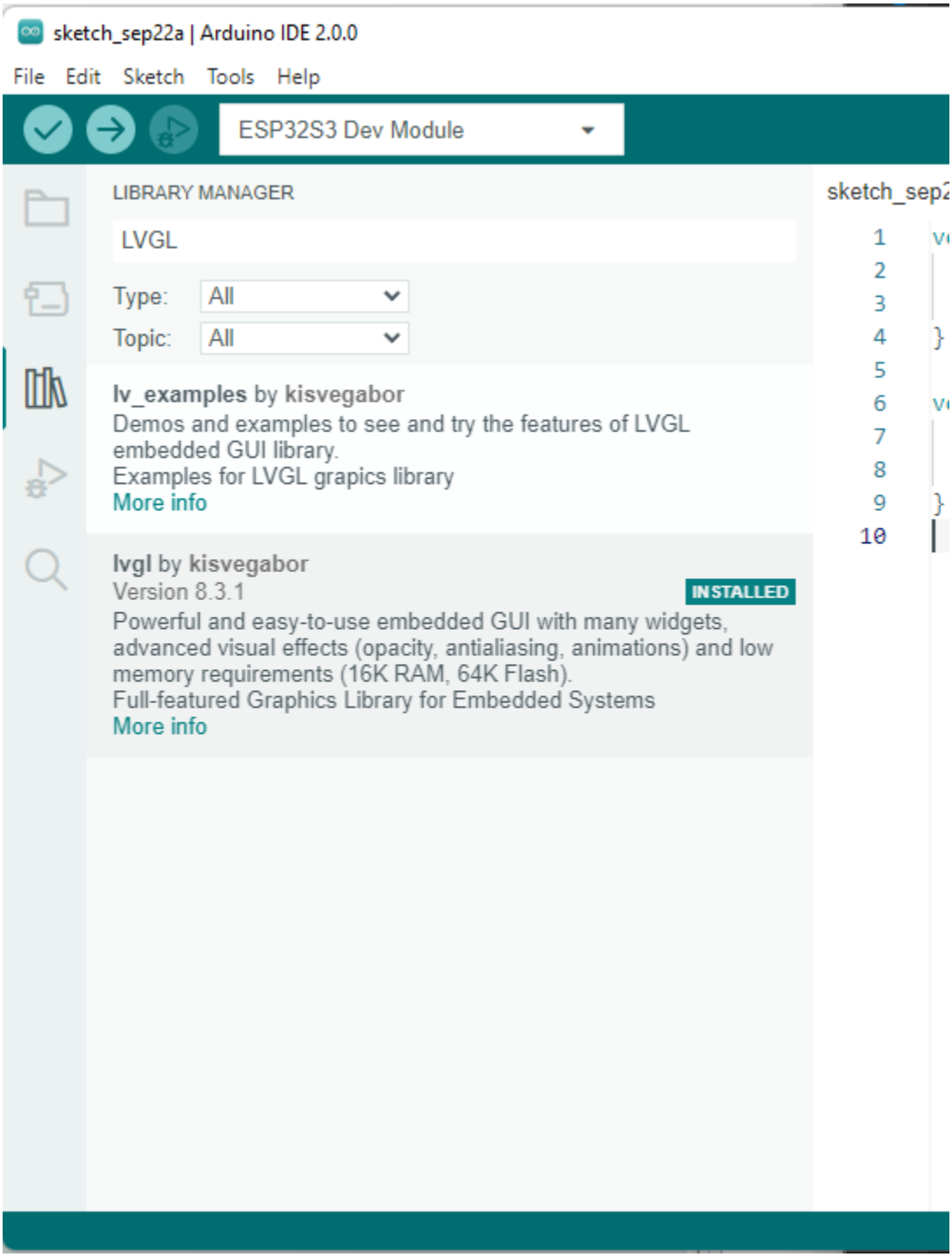
LVGL is an amazing GUI library, makes it easy to build modern UI.

Warning: Termod S3 uses SPI display, has not enough refresh rate to perfactly support LVGL. There will be some tearing when scrolling.

Install

On **Library Manager** tab, search for `LVGL`. Checkout if the version is 8.3.1, and click **INSTALL**. If not, select the version and click **INSTALL**.

Note: Why not the latest version? Because Termod S3 examples are develop under 8.3.1, and LVGL is under heavy development, there may be some breaking changes between versions. If you know what you are doing, you can try the latest version.



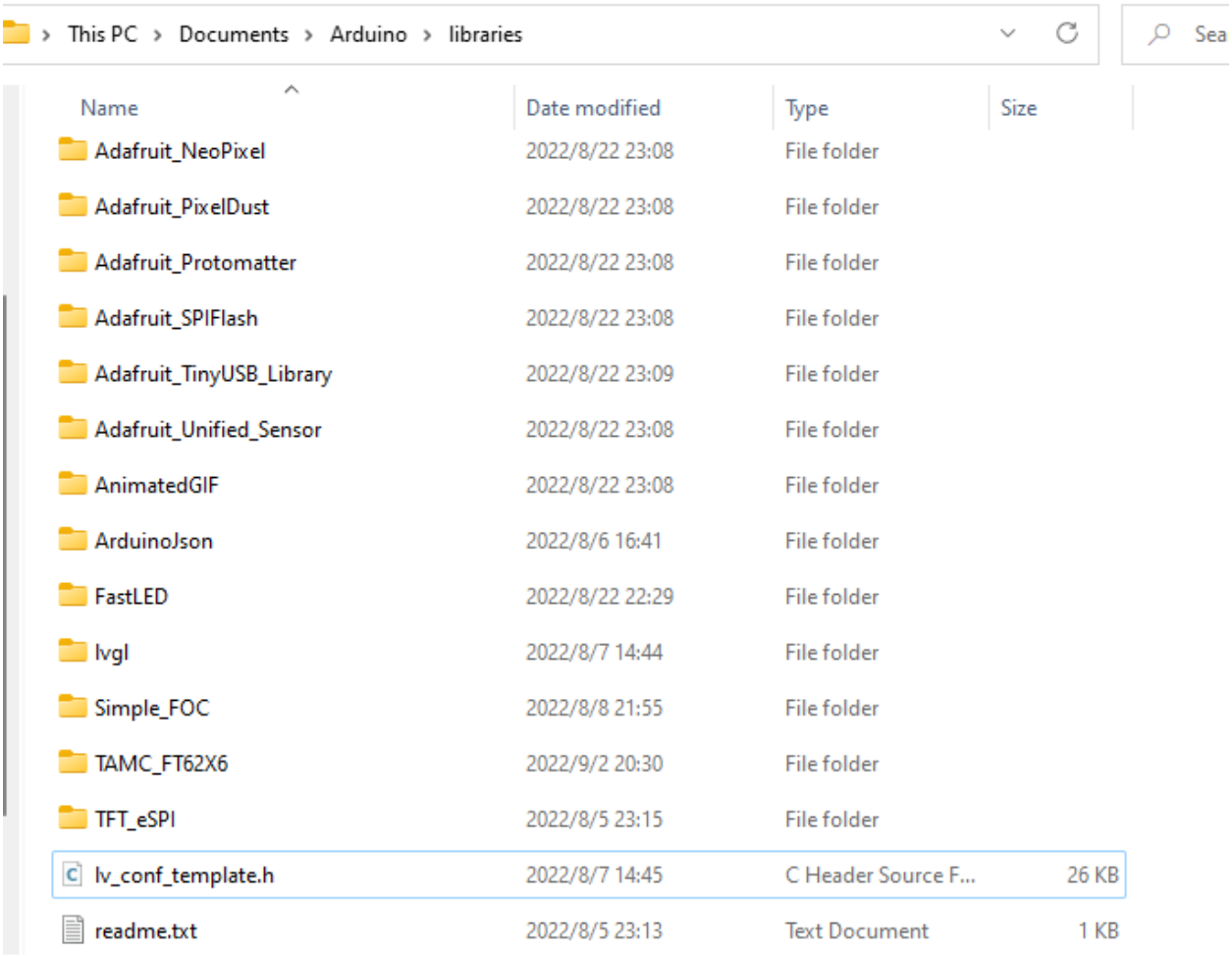
Setup LVGL

After install LVGL Library, open File Explorer and go to Arduino library folder. Usually is under the following folder.

- For Windows: `C:\Users\<USER>\Documents\Arduino\libraries`
- For Mac: `/Users/<USER>/Documents/Arduino/libraries`
- For Linux: `/home/<USER>/Documents/Arduino/libraries`

Open `lvgl` folder, and copy `lv_conf_template.h` file to Arduino library folder, alongside lvgl folder, not under lvgl.

Like this:



Then, rename it to `lv_conf.h`, open it with your favorate editor, and change first non-comment line `if 0` to `if 1`.

```
/**
 * @file lv_conf.h
 * Configuration file for v8.3.1
 */

/*
 * Copy this file as `lv_conf.h`
 * 1. simply next to the `lvgl` folder
 * 2. or any other places and
 *    - define `LV_CONF_INCLUDE_SIMPLE`
 *    - add the path as include path
 */

/* clang-format off */
#if 1 /*Set it to "1" to enable content*/

#ifndef LV_CONF_H
#define LV_CONF_H
```

(continues on next page)

(continued from previous page)

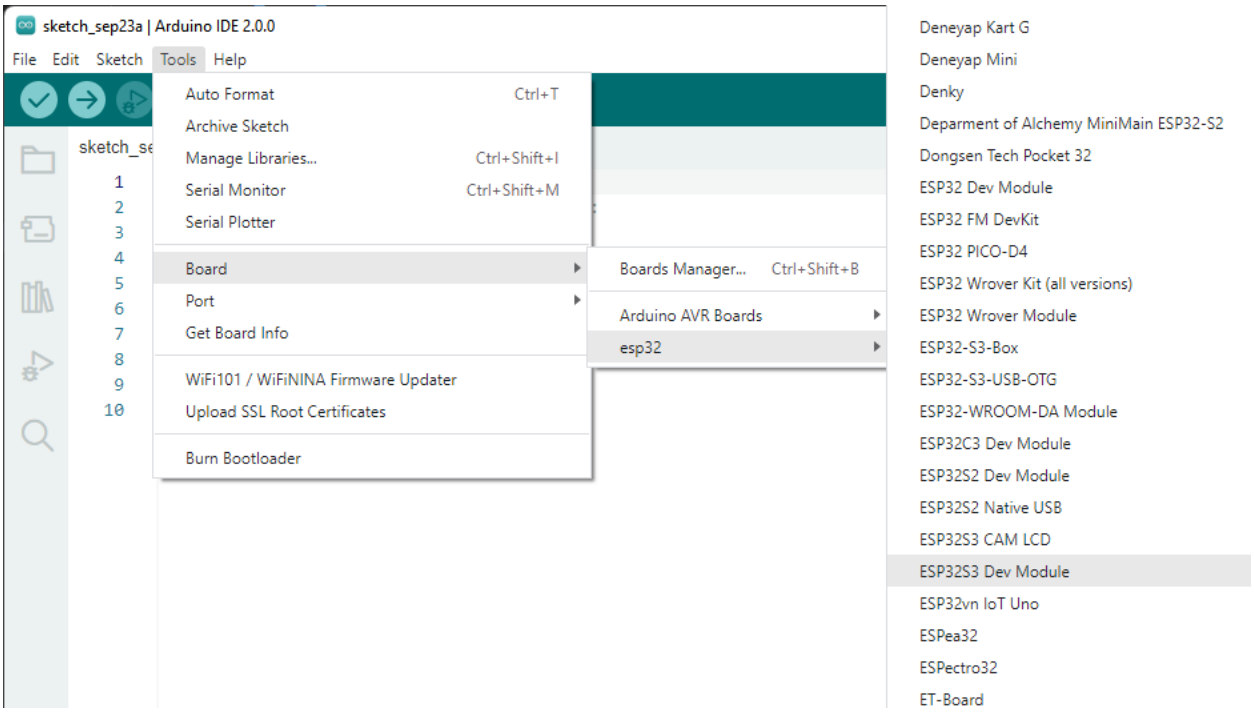
```
#include <stdint.h>

/*====================
   COLOR SETTINGS
 *====================*/

/*Color depth: 1 (1 byte per pixel), 8 (RGB332), 16 (RGB565), 32 (ARGB8888)*/
#define LV_COLOR_DEPTH 16

/*Swap the 2 bytes of RGB565 color. Useful if the display has an 8-bit int
```

This file contains the configuration options for LVGL. You can find more information about the options in the Configuration Reference. And done for LVGL setup.

Build and upload

Now everything is ready to build and upload. Make sure **Board** is set to `ESP32S3 Dev Module`.

Warning: You may see there's a TAMC Termod S3 board in the list, but it's not ready yet, don't use it as for esp32 core v2.0.5, will fix it in the next version.



Warning: You may notice there's a more easier way to select both port and device there in the new 2.0. But it's kinda problematic, board recognition may be not resulting the correct board, and it's not easy to select the correct port. So we recommand to use the old way.

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

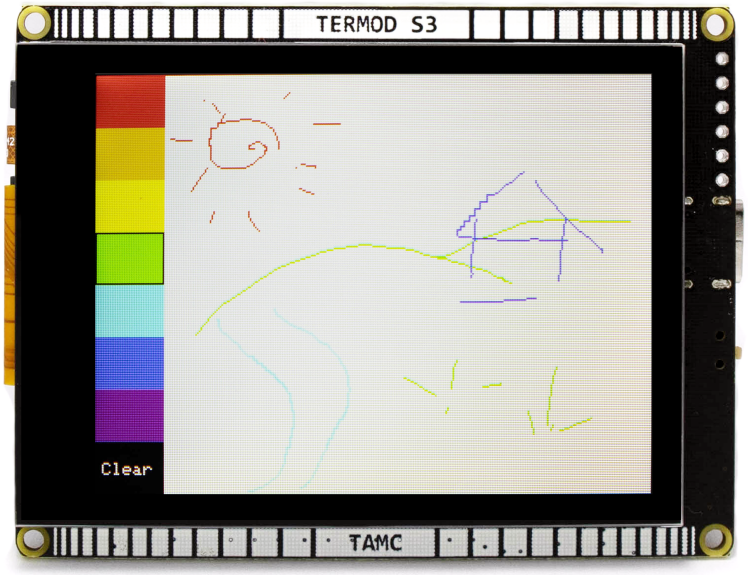
Open the downloaded folder, turn to examples, choose one example, and open it with Arduino IDE. Checkout more on *Examples*.

1.2.2 Examples

Now as you are ready, let's run some examples. You don't need to run all of them, just pick one or two that you like.

Simple Drawing APP

Painter



This example is a simple drawing app, with a bar of preset colors, and a slider to change the size of the brush. to choose and draw it on the right side of the screen.

With this example, you can learn how to use the touch screen with the display.

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Open `termod-s3/examples/draw/draw.ino` with Arduino IDE.

Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

draw.ino

```
#include <TFT_eSPI.h>
#include <TAMC_FT62X6.h>
#include <Wire.h>

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1

uint32_t currentColor = 0xF000;

TFT_eSPI tft = TFT_eSPI();
TAMC_FT62X6 tp = TAMC_FT62X6();

uint16_t colors[7] = {TFT_RED, TFT_ORANGE, TFT_YELLOW, TFT_GREEN, TFT_CYAN, TFT_BLUE,␣
↪TFT_PURPLE};

void drawButtons() {
  tft.fillRect(0, 0, 40, 30, TFT_RED);
  tft.fillRect(0, 30, 40, 30, TFT_ORANGE);
  tft.fillRect(0, 60, 40, 30, TFT_YELLOW);
  tft.fillRect(0, 90, 40, 30, TFT_GREEN);
  tft.fillRect(0, 120, 40, 30, TFT_CYAN);
  tft.fillRect(0, 150, 40, 30, TFT_BLUE);
  tft.fillRect(0, 180, 40, 30, TFT_PURPLE);
  tft.fillRect(0, 210, 40, 30, TFT_BLACK);
  tft.setCursor(4, 222);
  tft.setTextColor(TFT_WHITE);
  tft.setTextSize(1);
  tft.println("Clear");
}

void setup() {
  Wire.begin();
  tft.begin();
  tp.begin();
  tp.setRotation(DISPLAY_LANDSCAPE);
  tft.setRotation(DISPLAY_LANDSCAPE);
  tft.fillScreen(TFT_WHITE);
  drawButtons();
}
```

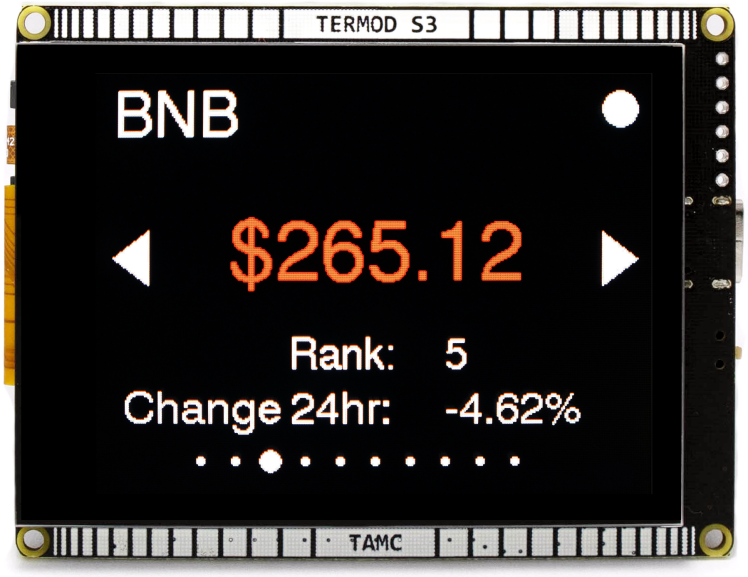
(continues on next page)

(continued from previous page)

```
void buttonPressed(int i) {
  drawButtons();
  if (i < 7) {
    tft.drawRect(0, i * 30, 40, 30, TFT_BLACK);
    currentColor = colors[i];
  } else {
    tft.fillRect(40, 0, 320, 240, TFT_WHITE);
  }
}

int lastX = -1;
int lastY = -1;
void loop() {
  int x = 0;
  int y = 0;
  tp.read();
  if (tp.isTouched) {
    x = tp.points[0].x;
    y = tp.points[0].y;
    // if touchstart
    if (lastX == -1) {
      if (x < 40) {
        buttonPressed(y / 30);
      }
    } else {
      if (x > 40) {
        tft.drawLine(lastX, lastY, x, y, currentColor);
      }
    }
    lastX = x;
    lastY = y;
  } else {
    lastX = -1;
    lastY = -1;
  }
}
```

Crypto Ticker

Crypto Ticker



This example have 10 crypto currencies to the display. Cycle through them by pressing the triangle button.

Data from Coin Cap API

Note: If you haven't download the code:

Download examples from github `termod-s3`

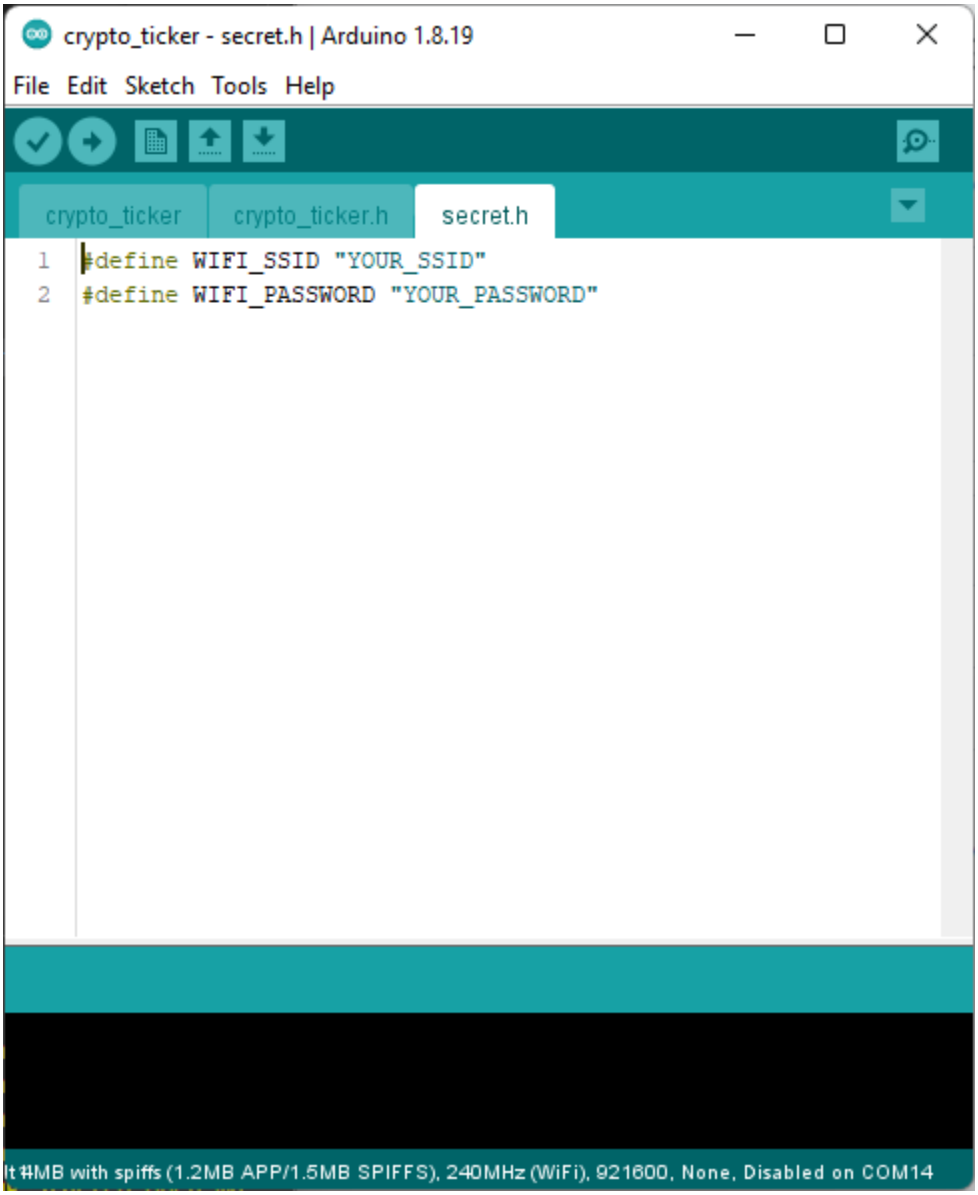
Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Open `termod-s3/examples/crypto_ticker/crypto_ticker.ino` with Arduino IDE.

As it require internet connection, you need to change the ssid and password to connect to your wifi network under `secret.h`.



Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

crypto_ticker.ino

crypto_ticker.h

secret.h

```
#include <TFT_eSPI.h>
#include <TAMC_FT62X6.h>

#include <WiFi.h>
#include <Wire.h>
#include <ArduinoJson.h>
#include <HTTPClient.h>
```

(continues on next page)

(continued from previous page)

```
#include <stdlib.h>

#include "crypto_ticker.h"
#include "secret.h"

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1

#define DISPLAY_WIDTH  240
#define DISPLAY_HEIGHT 320

// Instances
TFT_eSPI tft = TFT_eSPI();
TAMC_FT62X6 tp = TAMC_FT62X6();

// Global variables
HTTPClient http;
DynamicJsonDocument coinData(1024);
String header;

bool wifiFailed = false;

const char* coinId;
const char* coinSymbol;
const char* coinPrice;
const char* coinRank;
const char* coinChange24Hr;

String symbol = String(coinSymbol);
double price = 0;
String rank = String(coinRank);
double change24HrValue = round(String(coinChange24Hr).toFloat());

int currentCoinId = 0;
int lastCoinId = -1;

bool displayNeedReflash = false;
bool dataNeedReflash = false;

// Functions
void setStatus(int status);
void touchHandler();
bool coinGetData(String id);
void displayDrawMain(void);
void displayReflashData(void);
void displayDrawMessage(String msg);
void displayDrawMessage(String msg1, String msg2);
bool wifiInit();
String significentNumber(double f, int num);
```

(continues on next page)

(continued from previous page)

```
void setup(void) {
  Serial.begin(115200);
  Serial.println("Crypto Ticker Start!");
  Wire.begin();
  tft.init();
  tp.begin();
  tp.setRotation(DISPLAY_LANDSCAPE);
  tft.setRotation(DISPLAY_LANDSCAPE);
  tft.fillScreen(TFT_BLACK);
}

int lastX = -1;
int lastY = -1;
int retryCount = 0;
unsigned long previousMillis = 0;
unsigned long currentMillis = 0;
void loop() {
  if (WiFi.status() == WL_CONNECTED){
    currentMillis = millis();
    touchHandler();
    if (currentMillis - previousMillis > REFLASH_DELAY || dataNeedReflash) {
      bool success = false;
      setStatus(STATUS_BUSY);
      for (retryCount=0; retryCount<RETRY_COUNT; retryCount++){
        if (coinGetData(COINS[currentCoinId])){
          success = true;
          break;
        }
      }
      if (!success) {
        setStatus(STATUS_ERROR);
        displayDrawMessage("Get Data Error");
      } else {
        setStatus(STATUS_DONE);
        if (displayNeedReflash) {
          displayDrawMain();
          displayNeedReflash = false;
        }
        displayReflashData();
      }
      setStatus(STATUS_IDLE);
      dataNeedReflash = false;
    }
    if (currentMillis - previousMillis > LOOP_DELAY) {
      previousMillis = currentMillis;
    }
  }
  else {
    if (wifiInit()){
      Serial.println(WiFi.localIP().toString());
      String msg = String("IP: ") + WiFi.localIP().toString();
```

(continues on next page)

(continued from previous page)

```
      displayDrawMessage("Connected", msg);
      delay(2000);
      displayDrawMain();
    }
  }
}

// Status
void setStatus(int status) {
  tft.fillCircle(300, 20, 10, status_colors[status]);
}

/*
 * Touch Handler
 */
void touchHandler() {
  int x = 0;
  int y = 0;
  tp.read();
  if (tp.isTouched) {
    x = tp.points[0].x;
    y = tp.points[0].y;
    if (lastX == -1) {
      if (y > 70 && y < 140) {
        if (x > 0 && x < 50) {
          currentCoinId--;
          if (currentCoinId < 0) {
            currentCoinId = COINS_LENGTH - 1;
          }
          dataNeedReflash = true;
          displayNeedReflash = true;
        } else if (x > 270 && x < 320) {
          currentCoinId++;
          if (currentCoinId >= COINS_LENGTH) {
            currentCoinId = 0;
          }
          dataNeedReflash = true;
          displayNeedReflash = true;
        }
      }
      lastX = x;
      lastY = y;
    }
  } else {
    lastX = -1;
    lastY = -1;
  }
}

/*
 * Display
 */
```

(continues on next page)

(continued from previous page)

```
void displayDrawMain(void) {
  tft.fillScreen(TFT_BLACK);
  tft.setFreeFont(FF18);
  tft.setTextDatum(TR_DATUM);
  tft.setTextColor(TFT_WHITE);
  tft.drawString("Rank:", 170, 150, GFXFF);
  tft.drawString("Change 24hr:", 170, 180, 4);
  tft.fillTriangle(320 - 30, 90, 320 - 30, 120, 320 - 10, 105, TFT_PINK);
  tft.fillTriangle(      30, 90,       30, 120,       10, 105, TFT_PINK);
  for (int i=0; i<COINS_LENGTH; i++){
    if (i == currentCoinId){
      tft.fillCircle(60+(i*20), 220, 6, TFT_PINK);
    } else {
      tft.fillCircle(60+(i*20), 220, 2, TFT_WHITE);
    }
  }
}

void displayReflashData(void) {
  String id = String(coinId);

  String _symbol = String(coinSymbol);
  double _price = String(coinPrice).toFloat();
  String _rank = String(coinRank);
  double _change24HrValue = String(coinChange24Hr).toFloat();

  if (currentCoinId != lastCoinId) {
    symbol = _symbol;
    tft.fillRect(0, 0, 200, 50, TFT_BLACK);

    // Symbol
    tft.setFreeFont(FF19);
    tft.setTextDatum(TL_DATUM);
    tft.setTextColor(TFT_WHITE);
    tft.drawString(symbol, 10, 10, GFXFF);

    lastCoinId = currentCoinId;
  }

  // Price
  if (price != _price){
    if (_change24HrValue < 0){
      tft.setTextColor(TFT_RED);
    } else {
      tft.setTextColor(TFT_GREEN);
    }
    price = _price;
    String priceString = String("$") + significentNumber(price, 5);
    tft.fillRect(50, 80, 220, 50, TFT_BLACK);
    tft.setFreeFont(FF20);
    tft.setTextDatum(MC_DATUM);
    tft.drawString(priceString, 160, 100, GFXFF);
```

(continues on next page)

(continued from previous page)

```
  }

  // Rank
  if (rank != _rank){
    rank = _rank;
    tft.fillRect(200, 150, 320, 20, TFT_BLACK);
    tft.setFreeFont(FF18);
    tft.setTextDatum(TL_DATUM);
    tft.setTextColor(TFT_WHITE);
    tft.drawString(rank, 200, 150, GFXFF);
  }

  // Change 24 hour
  if (change24HrValue != _change24HrValue){
    change24HrValue = _change24HrValue;

    String change24Hr = significentNumber(change24HrValue, 4) + String("%");
    tft.fillRect(200, 180, 320, 20, TFT_BLACK);
    tft.setFreeFont(FF18);
    tft.setTextDatum(TL_DATUM);
    tft.setTextColor(TFT_WHITE);
    tft.drawString(change24Hr, 200, 180, 4);
  }
}

void displayDrawMessage(String msg) {
  tft.fillRoundRect(40, 50, 240, 140, 10, TFT_BLACK);
  tft.drawRoundRect(40, 50, 240, 140, 10, TFT_CYAN);
  tft.setFreeFont(FF17);
  tft.setTextDatum(MC_DATUM);
  tft.setTextColor(TFT_WHITE, TFT_BLACK);
  tft.drawString(msg, 160, 120, GFXFF);
  displayNeedReflash = true;
}

void displayDrawMessage(String msg1, String msg2) {
  tft.fillRoundRect(40, 50, 240, 140, 10, TFT_BLACK);
  tft.drawRoundRect(40, 50, 240, 140, 10, TFT_CYAN);
  tft.setFreeFont(FF17);
  tft.setTextDatum(MC_DATUM);
  tft.setTextColor(TFT_WHITE, TFT_BLACK);
  tft.drawString(msg1, 160, 110, GFXFF);
  tft.drawString(msg2, 160, 130, GFXFF);
  displayNeedReflash = true;
}

bool coinGetData(String id) {
  Serial.print("[HTTP] begin...\n");
  http.begin("https://api.coincap.io/v2/assets/" + id); //HTTP

  Serial.print("[HTTP] GET...\n");
  int httpCode = http.GET();
```

(continues on next page)

(continued from previous page)

```
  if(httpCode > 0) {
    Serial.printf("[HTTP] GET... code: %d\n", httpCode);
    if(httpCode == HTTP_CODE_OK) {
      String payload = http.getString();
      deserializeJson(coinData, payload);
      Serial.println(payload);
      coinId = coinData["data"]["id"];
      coinSymbol = coinData["data"]["symbol"];
      coinPrice = coinData["data"]["priceUsd"];
      coinRank = coinData["data"]["rank"];
      coinChange24Hr = coinData["data"]["changePercent24Hr"];
      http.end();
      return true;
    }
  } else {
    Serial.printf("[HTTP] GET... failed, error: %s\n", http.errorToString(httpCode).c_
↪str());
    http.end();
    return false;
  }
}

/*
 * Wi-Fi
 */
bool wifiInit() {
  displayDrawMessage("Connecting to", WIFI_SSID);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);

  return wifiConnecting(WIFI_SSID);
}

bool wifiConnecting(String msg){
  int WLcount = 0;
  while (1) {
    ++WLcount;
    delay(500);
    if (WLcount > WIFI_TIMEOUT * 2) {
      displayDrawMessage("Connection Failed");
      delay(2000);
      return false;
    }
    if (WiFi.status() == WL_CONNECTED){
      return true;
    }
  }
}

String significentNumber(double f, int num){
  String result = String(f, num);
  result = result.substring(0, num+1);
  if (result.indexOf(".") < 0 || result.indexOf(".") == 5){
```

(continues on next page)

(continued from previous page)

```
    result = result.substring(0, num);
  }
  return result;
}
```

```
#define WIFI_TIMEOUT 10 // Second

#define FF20 &FreeSans24pt7b
#define FF19 &FreeSans18pt7b
#define FF18 &FreeSans12pt7b
#define FF17 &FreeSans9pt7b
#define GFXFF 1

#define LOOP_DELAY        10000 // 100 ms
#define REFLASH_DELAY     10000 // 10 Second
#define RETRY_COUNT       10

#define STATUS_BUSY  0
#define STATUS_IDLE  1
#define STATUS_ERROR 2
#define STATUS_DONE  3

uint16_t status_colors[] = {TFT_YELLOW, TFT_WHITE, TFT_RED, TFT_GREEN};

String DOLLOR = "$";

/*
 * Coins
 */
#define COINS_LENGTH 10
String COINS[COINS_LENGTH] = {
  "bitcoin",
  "ethereum",
  "binance-coin",
  "cardano",
  "tether",
  "polkadot",
  "xrp",
  "uniswap",
  "litecoin",
  "chainlink",
};
```

```
#define WIFI_SSID "1002"
#define WIFI_PASSWORD "27148043"
```

LVGL Minimal Examples

This example shows a basic usage with LVGL.

In this example, we make a `lv_helper.cpp` and `lv_helper.h` makes it easy to implement LVGL in Arduino. And it is a minimal example for you to start with LVGL

Checkout the LVGL documentation for more information.

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Note: If you don't have lvgl installed, check this out: *Install LVGL Library (Optional)*.

Open `termod-s3/examples/lv_example/lv_example.ino` with Arduino IDE.

Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

lv_minimal_example.ino

lv_helper.cpp

lv_helper.h

```
#include "lv_helper.h"

void setup() {
  Serial.begin(115200);
  lh_init(DISPLAY_LANDSCAPE);
  Serial.println("LVGL Example: Ready");

  lv_obj_t* slider = lv_slider_create(lv_scr_act());
  lv_obj_align(slider, LV_ALIGN_CENTER, 0, 0);
}

void loop() {
  lv_timer_handler();
}
```

```
#include "lv_helper.h"

TFT_eSPI lh_tft = TFT_eSPI();
TAMC_FT62X6 lh_tp = TAMC_FT62X6();

static lv_disp_draw_buf_t lh_draw_buf;
static lv_color_t lh_buf[ DISPLAY_WIDTH * 10 ];
```

(continues on next page)

(continued from previous page)

```
static lv_disp_drv_t lh_disp_drv;
static lv_indev_drv_t lh_indev_drv;

uint16_t width, height;

void lh_init(int rotation){
  Wire.begin();
  lh_tp.begin();
  lv_init();
  lh_tft.begin();
  if (rotation == 1 || rotation == 3){
    width = DISPLAY_HEIGHT;
    height = DISPLAY_WIDTH;
  } else {
    width = DISPLAY_WIDTH;
    height = DISPLAY_HEIGHT;
  }
  lh_tft.setRotation(rotation);
  lh_tp.setRotation(rotation);

  lv_disp_draw_buf_init( &lh_draw_buf, lh_buf, NULL, DISPLAY_WIDTH * 10 );

  /*Initialize the display*/
  lv_disp_drv_init( &lh_disp_drv );
  /*Change the following line to your display resolution*/
  lh_disp_drv.hor_res = width;
  lh_disp_drv.ver_res = height;
  lh_disp_drv.flush_cb = lh_disp_flush;
  lh_disp_drv.draw_buf = &lh_draw_buf;
  lv_disp_drv_register( &lh_disp_drv );

  /*Initialize the (dummy) input device driver*/
  lv_indev_drv_init( &lh_indev_drv );
  lh_indev_drv.type = LV_INDEV_TYPE_POINTER;
  lh_indev_drv.read_cb = lh_touchpad_read;
  lv_indev_drv_register( &lh_indev_drv );
}

/* Display flushing */
void lh_disp_flush(lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p) {
  uint32_t w = (area->x2 - area->x1 + 1);
  uint32_t h = (area->y2 - area->y1 + 1);

  lh_tft.startWrite();
  lh_tft.setAddrWindow( area->x1, area->y1, w, h );
  lh_tft.pushColors( ( uint16_t * )&color_p->full, w * h, true );
  lh_tft.endWrite();

  lv_disp_flush_ready(disp);
}
/*Read the touchpad*/
void lh_touchpad_read(lv_indev_drv_t * indev_driver, lv_indev_data_t * data) {
```

(continues on next page)

(continued from previous page)

```
  lh_tp.read();

  if (!lh_tp.isTouched) {
    data->state = LV_INDEV_STATE_RELEASED;
  }
  else{
    data->state = LV_INDEV_STATE_PRESSED;
    /*Set the coordinates*/
    data->point.x = lh_tp.points[0].x;
    data->point.y = lh_tp.points[0].y;
  }
}
```

```
#ifndef LV_HELPER_H
#define LV_HELPER_H

#include <lvgl.h>
#include "TAMC_FT62X6.h"
#include "Wire.h"
#include <TFT_eSPI.h>

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1

#define DISPLAY_WIDTH  240
#define DISPLAY_HEIGHT 320

/* Display flushing */
void lh_disp_flush(lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p);
/*Read the touchpad*/
void lh_touchpad_read(lv_indev_drv_t * indev_driver, lv_indev_data_t * data);

void lh_init(int rotation);
#endif // LV_HELPER_H
```

LVGL Examples

This example shows some of the LVGL widgets usage.

Checkout the LVGL documentation for more information.

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Note: If you don't have lvgl installed, check this out: *Install LVGL Library (Optional)*.

Open `termod-s3/examples/lv_example/lv_example.ino` with Arduino IDE.

Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

lv_example.ino

lv_helper.cpp

lv_helper.h

```
#include "lv_helper.h"

lv_obj_t* arc;
lv_obj_t* slider;
lv_obj_t* arcValueLabel;

// Arc
void lv_example_arc_1(void) {
  arc = lv_arc_create(lv_scr_act());
  lv_obj_set_size(arc, 100, 100);
  lv_arc_set_rotation(arc, 135);
  lv_arc_set_bg_angles(arc, 0, 270);
  lv_arc_set_value(arc, 0);
  lv_arc_set_range(arc, 0, 100);
  lv_obj_align(arc, LV_ALIGN_TOP_MID, 0, 10);
  arcValueLabel = lv_label_create(arc);
  lv_label_set_text(arcValueLabel, String(0).c_str());
  lv_obj_center(arcValueLabel);
  lv_obj_add_event_cb(arc, arcValueChanged, LV_EVENT_VALUE_CHANGED, NULL);
}
static void arcValueChanged(lv_event_t* e){
  lv_obj_t* obj = lv_event_get_target(e);
  int value = (int)lv_arc_get_value(obj);
  lv_label_set_text(arcValueLabel, String(value).c_str());
  lv_bar_set_value(slider, value, LV_ANIM_OFF);
}

// Slider
void lv_example_slider_1(void) {
  slider = lv_slider_create(lv_scr_act());
  lv_obj_align(slider, LV_ALIGN_TOP_MID, 0, 130);
  lv_obj_add_event_cb(slider, sliderValueChanged, LV_EVENT_VALUE_CHANGED, NULL);
}
static void sliderValueChanged(lv_event_t* e){
  lv_obj_t* obj = lv_event_get_target(e);
  int value = (int)lv_slider_get_value(obj);
  lv_label_set_text(arcValueLabel, String(value).c_str());
  lv_arc_set_value(arc, value);
}
```

(continues on next page)

(continued from previous page)

```
// Button and toggle
void lv_example_btn_1(void){
  lv_obj_t * label;

  lv_obj_t * btn1 = lv_btn_create(lv_scr_act());
  lv_obj_add_event_cb(btn1, event_handler, LV_EVENT_ALL, NULL);
  lv_obj_align(btn1, LV_ALIGN_TOP_MID, 80, 170);

  label = lv_label_create(btn1);
  lv_label_set_text(label, "Button");
  lv_obj_center(label);

  lv_obj_t * btn2 = lv_btn_create(lv_scr_act());
  lv_obj_add_event_cb(btn2, event_handler, LV_EVENT_ALL, NULL);
  lv_obj_align(btn2, LV_ALIGN_TOP_MID, -80, 170);
  lv_obj_add_flag(btn2, LV_OBJ_FLAG_CHECKABLE);
  lv_obj_set_height(btn2, LV_SIZE_CONTENT);

  label = lv_label_create(btn2);
  lv_label_set_text(label, "Toggle");
  lv_obj_center(label);
}
static void event_handler(lv_event_t* e){
  lv_event_code_t code = lv_event_get_code(e);
  if(code == LV_EVENT_CLICKED) {
      Serial.println("Clicked");
  }
  else if(code == LV_EVENT_VALUE_CHANGED) {
      Serial.println("Toggled");
  }
}

void lv_example_checkbox_1(void) {
  lv_obj_t * checkboxs = lv_obj_create(lv_scr_act());
  lv_obj_set_flex_flow(checkboxs, LV_FLEX_FLOW_COLUMN);
  lv_obj_set_flex_align(checkboxs, LV_FLEX_ALIGN_CENTER, LV_FLEX_ALIGN_START, LV_FLEX_
↪ALIGN_CENTER);
  lv_obj_align(checkboxs, LV_ALIGN_TOP_MID, 0, 220);
  lv_obj_set_size(checkboxs, 200, 160);

  lv_obj_t * cb;
  cb = lv_checkbox_create(checkboxs);
  lv_checkbox_set_text(cb, "Apple");
  lv_obj_add_event_cb(cb, checkbox_event_handler, LV_EVENT_ALL, NULL);

  cb = lv_checkbox_create(checkboxs);
  lv_checkbox_set_text(cb, "Banana");
  lv_obj_add_state(cb, LV_STATE_CHECKED);
  lv_obj_add_event_cb(cb, checkbox_event_handler, LV_EVENT_ALL, NULL);

  cb = lv_checkbox_create(checkboxs);
```

(continues on next page)

(continued from previous page)

```
  lv_checkbox_set_text(cb, "Lemon");
  lv_obj_add_event_cb(cb, checkbox_event_handler, LV_EVENT_ALL, NULL);

  cb = lv_checkbox_create(checkboxs);
  lv_checkbox_set_text(cb, "Melon\nand a new line");
  lv_obj_add_event_cb(cb, checkbox_event_handler, LV_EVENT_ALL, NULL);

  lv_obj_update_layout(cb);
}
static void checkbox_event_handler(lv_event_t* e) {
  lv_event_code_t code = lv_event_get_code(e);
  lv_obj_t * obj = lv_event_get_target(e);
  if(code == LV_EVENT_VALUE_CHANGED) {
    const char * txt = lv_checkbox_get_text(obj);
    const char * state = lv_obj_get_state(obj) & LV_STATE_CHECKED ? "Checked" :
↪"Unchecked";
    Serial.printf("%s: %s\n", txt, state);
    Serial.flush();
  }
}

// Dropdown
void lv_example_dropdown_1(void) {
    lv_obj_t * dd = lv_dropdown_create(lv_scr_act());
    lv_dropdown_set_options(dd, "Apple\n"
                                "Banana\n"
                                "Orange\n"
                                "Cherry\n"
                                "Grape\n"
                                "Raspberry\n"
                                "Melon\n"
                                "Orange\n"
                                "Lemon\n"
                                "Nuts");

    lv_obj_align(dd, LV_ALIGN_TOP_MID, 0, 400);
    lv_obj_add_event_cb(dd, dropdown_event_handler, LV_EVENT_ALL, NULL);
}
static void dropdown_event_handler(lv_event_t* e) {
  lv_event_code_t code = lv_event_get_code(e);
  lv_obj_t * obj = lv_event_get_target(e);
  if(code == LV_EVENT_VALUE_CHANGED) {
    char buf[32];
    lv_dropdown_get_selected_str(obj, buf, sizeof(buf));
    Serial.printf("Option: %s\n", buf);
    Serial.flush();
  }
}

// Roller
void lv_example_roller_1(void) {
  lv_obj_t *roller1 = lv_roller_create(lv_scr_act());
```

(continues on next page)

(continued from previous page)

```
  lv_roller_set_options(
    roller1,
    "January\n"
    "February\n"
    "March\n"
    "April\n"
    "May\n"
    "June\n"
    "July\n"
    "August\n"
    "September\n"
    "October\n"
    "November\n"
    "December",
    LV_ROLLER_MODE_INFINITE
  );

  lv_roller_set_visible_row_count(roller1, 4);
  lv_obj_align(roller1, LV_ALIGN_TOP_MID, 0, 460);
  lv_obj_add_event_cb(roller1, roller_event_handler, LV_EVENT_ALL, NULL);
}
static void roller_event_handler(lv_event_t* e) {
  lv_event_code_t code = lv_event_get_code(e);
  lv_obj_t * obj = lv_event_get_target(e);
  if(code == LV_EVENT_VALUE_CHANGED) {
    char buf[32];
    lv_roller_get_selected_str(obj, buf, sizeof(buf));
    Serial.printf("Selected month: %s\n", buf);
    Serial.flush();
  }
}

// Switches
void lv_example_switch_1(void) {
  lv_obj_t * li;

  lv_obj_t * sw;
  lv_obj_t * label;

  li = lv_obj_create(lv_scr_act());
  lv_obj_clear_flag(li, LV_OBJ_FLAG_SCROLLABLE);
  lv_obj_set_size(li, 310, 50);
  lv_obj_align(li, LV_ALIGN_TOP_MID, 0, 610);
  sw = lv_switch_create(li);
  lv_obj_align(sw, LV_ALIGN_RIGHT_MID, -10, 0);
  lv_obj_add_event_cb(sw, switch_event_handler, LV_EVENT_ALL, NULL);
  label = lv_label_create(li);
  lv_label_set_text(label, "Switch 1");
  lv_obj_align(label, LV_ALIGN_LEFT_MID, 10, 0);

  li = lv_obj_create(lv_scr_act());
  lv_obj_clear_flag(li, LV_OBJ_FLAG_SCROLLABLE);
```

(continues on next page)

(continued from previous page)

```
  lv_obj_set_size(li, 310, 50);
  lv_obj_align(li, LV_ALIGN_TOP_MID, 0, 670);
  sw = lv_switch_create(li);
  lv_obj_add_state(sw, LV_STATE_CHECKED);
  lv_obj_align(sw, LV_ALIGN_RIGHT_MID, -10, 0);
  lv_obj_add_event_cb(sw, switch_event_handler, LV_EVENT_ALL, NULL);
  label = lv_label_create(li);
  lv_label_set_text(label, "Switch 2");
  lv_obj_align(label, LV_ALIGN_LEFT_MID, 10, 0);
}
static void switch_event_handler(lv_event_t* e) {
  lv_event_code_t code = lv_event_get_code(e);
  lv_obj_t * obj = lv_event_get_target(e);
  if(code == LV_EVENT_VALUE_CHANGED) {
    Serial.printf("State: %s\n", lv_obj_has_state(obj, LV_STATE_CHECKED) ? "On" : "Off");
    Serial.flush();
  }
}

// Calender
void lv_example_calendar_1(void) {
  lv_obj_t* calendar = lv_calendar_create(lv_scr_act());
  lv_obj_set_size(calendar, 300, 200);
  lv_obj_align(calendar, LV_ALIGN_TOP_MID, 0, 730);
  lv_obj_add_event_cb(calendar, calendar_event_handler, LV_EVENT_ALL, NULL);

  lv_calendar_set_today_date(calendar, 2022, 8, 9);
  lv_calendar_set_showed_date(calendar, 2022, 8);

  /*Highlight a few days*/
  static lv_calendar_date_t highlighted_days[3];       /*Only its pointer will be saved␣
↪so should be static*/
  highlighted_days[0].year = 2022;
  highlighted_days[0].month = 8;
  highlighted_days[0].day = 7;

  highlighted_days[1].year = 2022;
  highlighted_days[1].month = 2;
  highlighted_days[1].day = 11;

  highlighted_days[2].year = 2022;
  highlighted_days[2].month = 2;
  highlighted_days[2].day = 22;

  lv_calendar_set_highlighted_dates(calendar, highlighted_days, 3);

  lv_calendar_header_dropdown_create(calendar);
  // lv_calendar_header_arrow_create(calendar);
  // lv_calendar_set_showed_date(calendar, 2021, 10);
}
static void calendar_event_handler(lv_event_t* e) {
  lv_event_code_t code = lv_event_get_code(e);
```

(continues on next page)

(continued from previous page)

```
  lv_obj_t * obj = lv_event_get_current_target(e);

  if(code == LV_EVENT_VALUE_CHANGED) {
    lv_calendar_date_t date;
    if(lv_calendar_get_pressed_date(obj, &date)) {
      Serial.printf("Clicked date: %02d.%02d.%d\n", date.day, date.month, date.year);
      Serial.flush();
    }
  }
}

void lv_example_chart_1(void) {
  /*Create a chart*/
  lv_obj_t * chart;
  chart = lv_chart_create(lv_scr_act());
  lv_obj_set_size(chart, 200, 150);
  lv_obj_align(chart, LV_ALIGN_TOP_MID, 0, 940);
  lv_chart_set_type(chart, LV_CHART_TYPE_LINE);   /*Show lines and points too*/

  /*Add two data series*/
  lv_chart_series_t * ser1 = lv_chart_add_series(chart, lv_palette_main(LV_PALETTE_RED),␣
↪LV_CHART_AXIS_PRIMARY_Y);
  lv_chart_series_t * ser2 = lv_chart_add_series(chart, lv_palette_main(LV_PALETTE_
↪GREEN), LV_CHART_AXIS_SECONDARY_Y);

  /*Set the next points on 'ser1'*/
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 10);
  lv_chart_set_next_value(chart, ser1, 30);
  lv_chart_set_next_value(chart, ser1, 70);
  lv_chart_set_next_value(chart, ser1, 90);

  /*Directly set points on 'ser2'*/
  ser2->y_points[0] = 90;
  ser2->y_points[1] = 70;
  ser2->y_points[2] = 65;
  ser2->y_points[3] = 65;
  ser2->y_points[4] = 65;
  ser2->y_points[5] = 65;
  ser2->y_points[6] = 65;
  ser2->y_points[7] = 65;
  ser2->y_points[8] = 65;
  ser2->y_points[9] = 65;

  lv_chart_refresh(chart); /*Required after direct set*/
}
```

(continues on next page)

(continued from previous page)

```
void setup() {
  Serial.begin(115200);
  lh_init(DISPLAY_LANDSCAPE);
  Serial.println("LVGL Example: Ready");

  lv_example_arc_1();
  lv_example_slider_1();
  lv_example_btn_1();
  lv_example_checkbox_1();
  lv_example_dropdown_1();
  lv_example_roller_1();
  lv_example_switch_1();
  lv_example_calendar_1();
  lv_example_chart_1();
}

void loop() {
  lv_timer_handler();
}
```

```
#include "lv_helper.h"

TFT_eSPI lh_tft = TFT_eSPI();
TAMC_FT62X6 lh_tp = TAMC_FT62X6();

static lv_disp_draw_buf_t lh_draw_buf;
static lv_color_t lh_buf[ DISPLAY_WIDTH * 10 ];
static lv_disp_drv_t lh_disp_drv;
static lv_indev_drv_t lh_indev_drv;

uint16_t width, height;

void lh_init(int rotation){
  Wire.begin();
  lh_tp.begin();
  lv_init();
  lh_tft.begin();
  if (rotation == 1 || rotation == 3){
    width = DISPLAY_HEIGHT;
    height = DISPLAY_WIDTH;
  } else {
    width = DISPLAY_WIDTH;
    height = DISPLAY_HEIGHT;
  }
  lh_tft.setRotation(rotation);
  lh_tp.setRotation(rotation);

  lv_disp_draw_buf_init( &lh_draw_buf, lh_buf, NULL, DISPLAY_WIDTH * 10 );

  /*Initialize the display*/
  lv_disp_drv_init( &lh_disp_drv );
  /*Change the following line to your display resolution*/
```

(continues on next page)

(continued from previous page)

```
  lh_disp_drv.hor_res = width;
  lh_disp_drv.ver_res = height;
  lh_disp_drv.flush_cb = lh_disp_flush;
  lh_disp_drv.draw_buf = &lh_draw_buf;
  lv_disp_drv_register( &lh_disp_drv );

  /*Initialize the (dummy) input device driver*/
  lv_indev_drv_init( &lh_indev_drv );
  lh_indev_drv.type = LV_INDEV_TYPE_POINTER;
  lh_indev_drv.read_cb = lh_touchpad_read;
  lv_indev_drv_register( &lh_indev_drv );
}

/* Display flushing */
void lh_disp_flush(lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p) {
  uint32_t w = (area->x2 - area->x1 + 1);
  uint32_t h = (area->y2 - area->y1 + 1);

  lh_tft.startWrite();
  lh_tft.setAddrWindow( area->x1, area->y1, w, h );
  lh_tft.pushColors( ( uint16_t * )&color_p->full, w * h, true );
  lh_tft.endWrite();

  lv_disp_flush_ready(disp);
}
/*Read the touchpad*/
void lh_touchpad_read(lv_indev_drv_t * indev_driver, lv_indev_data_t * data) {
  lh_tp.read();

  if (!lh_tp.isTouched) {
    data->state = LV_INDEV_STATE_RELEASED;
  }
  else{
    data->state = LV_INDEV_STATE_PRESSED;
    /*Set the coordinates*/
    data->point.x = lh_tp.points[0].x;
    data->point.y = lh_tp.points[0].y;
  }
}
```

```
#ifndef LV_HELPER_H
#define LV_HELPER_H

#include <lvgl.h>
#include "TAMC_FT62X6.h"
#include "Wire.h"
#include <TFT_eSPI.h>

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1
```

(continues on next page)

(continued from previous page)

```
#define DISPLAY_WIDTH  240
#define DISPLAY_HEIGHT 320

/* Display flushing */
void lh_disp_flush(lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p);
/*Read the touchpad*/
void lh_touchpad_read(lv_indev_drv_t * indev_driver, lv_indev_data_t * data);

void lh_init(int rotation);
#endif // LV_HELPER_H
```

Handling micro SD Card

This example is basic usage of micro SD Card, it's copy from ESP32 example SD. Only adds `SD_CS` to `SD.begin()`

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Open `termod-s3/examples/sd_example/sd_example.ino` with Arduino IDE.

Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

sd_example.ino

```
#include "FS.h"
#include "SD.h"
#include "SPI.h"

static const uint8_t SD_CS = 21;

void listDir(fs::FS &fs, const char * dirname, uint8_t levels){
    Serial.printf("Listing directory: %s\n", dirname);

    File root = fs.open(dirname);
    if(!root){
        Serial.println("Failed to open directory");
        return;
    }
    if(!root.isDirectory()){
        Serial.println("Not a directory");
        return;
    }
```

(continues on next page)

(continued from previous page)

```
    File file = root.openNextFile();
    while(file){
        if(file.isDirectory()){
            Serial.print("  DIR : ");
            Serial.println(file.name());
            if(levels){
                listDir(fs, file.path(), levels -1);
            }
        } else {
            Serial.print("  FILE: ");
            Serial.print(file.name());
            Serial.print("  SIZE: ");
            Serial.println(file.size());
        }
        file = root.openNextFile();
    }
}

void createDir(fs::FS &fs, const char * path){
    Serial.printf("Creating Dir: %s\n", path);
    if(fs.mkdir(path)){
        Serial.println("Dir created");
    } else {
        Serial.println("mkdir failed");
    }
}

void removeDir(fs::FS &fs, const char * path){
    Serial.printf("Removing Dir: %s\n", path);
    if(fs.rmdir(path)){
        Serial.println("Dir removed");
    } else {
        Serial.println("rmdir failed");
    }
}

void readFile(fs::FS &fs, const char * path){
    Serial.printf("Reading file: %s\n", path);

    File file = fs.open(path);
    if(!file){
        Serial.println("Failed to open file for reading");
        return;
    }

    Serial.print("Read from file: ");
    while(file.available()){
        Serial.write(file.read());
    }
    file.close();
}
```

(continues on next page)

(continued from previous page)

```
void writeFile(fs::FS &fs, const char * path, const char * message){
    Serial.printf("Writing file: %s\n", path);

    File file = fs.open(path, FILE_WRITE);
    if(!file){
        Serial.println("Failed to open file for writing");
        return;
    }
    if(file.print(message)){
        Serial.println("File written");
    } else {
        Serial.println("Write failed");
    }
    file.close();
}

void appendFile(fs::FS &fs, const char * path, const char * message){
    Serial.printf("Appending to file: %s\n", path);

    File file = fs.open(path, FILE_APPEND);
    if(!file){
        Serial.println("Failed to open file for appending");
        return;
    }
    if(file.print(message)){
        Serial.println("Message appended");
    } else {
        Serial.println("Append failed");
    }
    file.close();
}

void renameFile(fs::FS &fs, const char * path1, const char * path2){
    Serial.printf("Renaming file %s to %s\n", path1, path2);
    if (fs.rename(path1, path2)) {
        Serial.println("File renamed");
    } else {
        Serial.println("Rename failed");
    }
}

void deleteFile(fs::FS &fs, const char * path){
    Serial.printf("Deleting file: %s\n", path);
    if(fs.remove(path)){
        Serial.println("File deleted");
    } else {
        Serial.println("Delete failed");
    }
}

void testFileIO(fs::FS &fs, const char * path){
    File file = fs.open(path);
```

(continues on next page)

(continued from previous page)

```
    static uint8_t buf[512];
    size_t len = 0;
    uint32_t start = millis();
    uint32_t end = start;
    if(file){
        len = file.size();
        size_t flen = len;
        start = millis();
        while(len){
            size_t toRead = len;
            if(toRead > 512){
                toRead = 512;
            }
            file.read(buf, toRead);
            len -= toRead;
        }
        end = millis() - start;
        Serial.printf("%u bytes read for %u ms\n", flen, end);
        file.close();
    } else {
        Serial.println("Failed to open file for reading");
    }


    file = fs.open(path, FILE_WRITE);
    if(!file){
        Serial.println("Failed to open file for writing");
        return;
    }

    size_t i;
    start = millis();
    for(i=0; i<2048; i++){
        file.write(buf, 512);
    }
    end = millis() - start;
    Serial.printf("%u bytes written for %u ms\n", 2048 * 512, end);
    file.close();
}

void setup(){
    Serial.begin(115200);
    if(!SD.begin(SD_CS)){
        Serial.println("Card Mount Failed");
        return;
    }
    uint8_t cardType = SD.cardType();

    if(cardType == CARD_NONE){
        Serial.println("No SD card attached");
        return;
    }
```

(continues on next page)

(continued from previous page)

```
    Serial.print("SD Card Type: ");
    if(cardType == CARD_MMC){
        Serial.println("MMC");
    } else if(cardType == CARD_SD){
        Serial.println("SDSC");
    } else if(cardType == CARD_SDHC){
        Serial.println("SDHC");
    } else {
        Serial.println("UNKNOWN");
    }

    uint64_t cardSize = SD.cardSize() / (1024 * 1024);
    Serial.printf("SD Card Size: %lluMB\n", cardSize);

    listDir(SD, "/", 0);
    createDir(SD, "/mydir");
    listDir(SD, "/", 0);
    removeDir(SD, "/mydir");
    listDir(SD, "/", 2);
    writeFile(SD, "/hello.txt", "Hello ");
    appendFile(SD, "/hello.txt", "World!\n");
    readFile(SD, "/hello.txt");
    deleteFile(SD, "/foo.txt");
    renameFile(SD, "/hello.txt", "/foo.txt");
    readFile(SD, "/foo.txt");
    testFileIO(SD, "/test.txt");
    Serial.printf("Total space: %lluMB\n", SD.totalBytes() / (1024 * 1024));
    Serial.printf("Used space: %lluMB\n", SD.usedBytes() / (1024 * 1024));
}

void loop(){

}
```

Reads and display battery infomations

This example is on how to read and display battery infomations.

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Open `termod-s3/examples/battery_info/battery_info.ino` with Arduino IDE.

Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

battery_info.ino

```
#include <TFT_eSPI.h>

TFT_eSPI tft = TFT_eSPI();

#define FF20 &FreeSans24pt7b
#define FF18 &FreeSans12pt7b

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1

static const uint8_t BAT_LV = 1;
static const uint8_t CHG = 2;

bool getChargingState() {
  return !digitalRead(CHG);
}

float getBatteryVoltage() {
  int analogVolt = analogReadMilliVolts(1);
  float voltage = analogVolt / 1000.0;
  voltage = voltage * (100.0 + 200.0) / 200.0;
  return voltage;
}

float getBatteryCapacity() {
  float voltage = getBatteryVoltage();
  float capacity = (voltage - 3.3) / (4.2 - 3.3) * 100.0;
  capacity = constrain(capacity, 0, 100);
  return capacity;
}

void setup() {
  Serial.begin(115200);
  tft.init();
  tft.setRotation(DISPLAY_LANDSCAPE);
  pinMode(CHG, INPUT_PULLUP);

  tft.fillScreen(TFT_BLACK);
  tft.drawRoundRect(40, 70, 240, 100, 10, TFT_WHITE);
  tft.drawRoundRect(270, 100, 20, 40, 4, TFT_WHITE);
  tft.fillRect(270, 100, 9, 40, TFT_BLACK); // cover the left side of the battery button
}

void loop() {
  float batteryCapacity = getBatteryCapacity();
  float batteryVoltage = getBatteryVoltage();
  bool isCharge = getChargingState();
  String batteryCapacityString = String(batteryCapacity) + String("%");
```

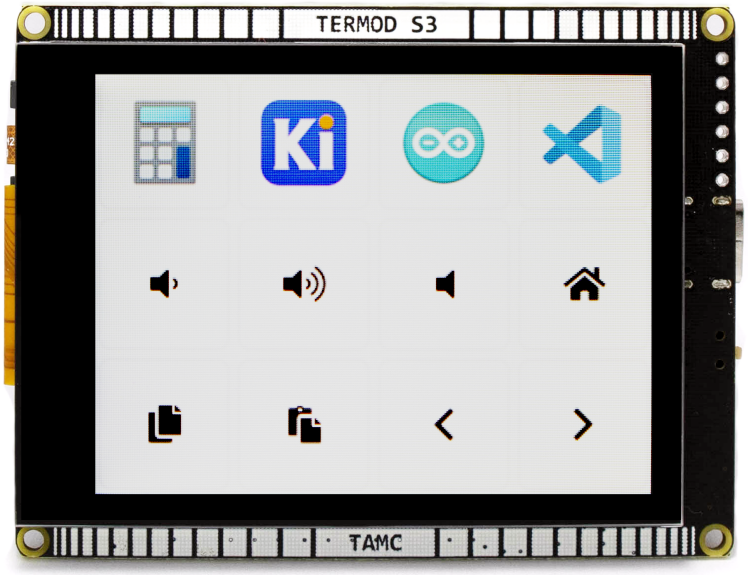
(continues on next page)

(continued from previous page)

```
  String batteryVoltageString = String(batteryVoltage) + String("V");
  Serial.print("Battery Capacitive: ");
  Serial.println(batteryCapacity);
  Serial.print("Battery Voltage: ");
  Serial.println(batteryVoltage);
  int width = batteryCapacity * 234.0 / 100.0;

  tft.fillRoundRect(43, 73, 234, 94, 6, TFT_BLACK);
  tft.fillRoundRect(43, 73, width, 94, 6, TFT_GREEN);
  tft.setFreeFont(FF20);
  tft.setTextDatum(MC_DATUM);
  tft.setTextColor(TFT_WHITE);
  tft.drawString(batteryCapacityString, 160, 120, 4);

  tft.fillRect(100, 190, 120, 20, TFT_RED);
  tft.setFreeFont(FF18);
  tft.setTextDatum(MC_DATUM);
  tft.setTextColor(TFT_WHITE);
  tft.drawString(batteryVoltageString, 160, 200, 2);
  tft.fillRect(100, 210, 120, 20, TFT_RED);
  if (isCharge) {
    Serial.println("Changing");
    tft.setFreeFont(FF18);
    tft.setTextDatum(MC_DATUM);
    tft.setTextColor(TFT_WHITE);
    tft.drawString("Charging", 160, 220, 2);
  }

  delay(1000);
}
```

Macro Pad

Macro Pad



Tutorial

This example shows how to use Termod S3 as a macro pad.

We use LVGL to make beautiful UI. Here also uses `lv_helper`

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Open `termod-s3/examples/macro_pad/macro_pad.ino` with Arduino IDE.

This example use a 22px font `LV_FONT_MONTSERRAT_22`, you need to enable it in `lv_conf.h`, the conf file mentioned in *Install LVGL Library (Optional)*.

Open the file, and find the following code, change the 0 to 1 to enable the font.

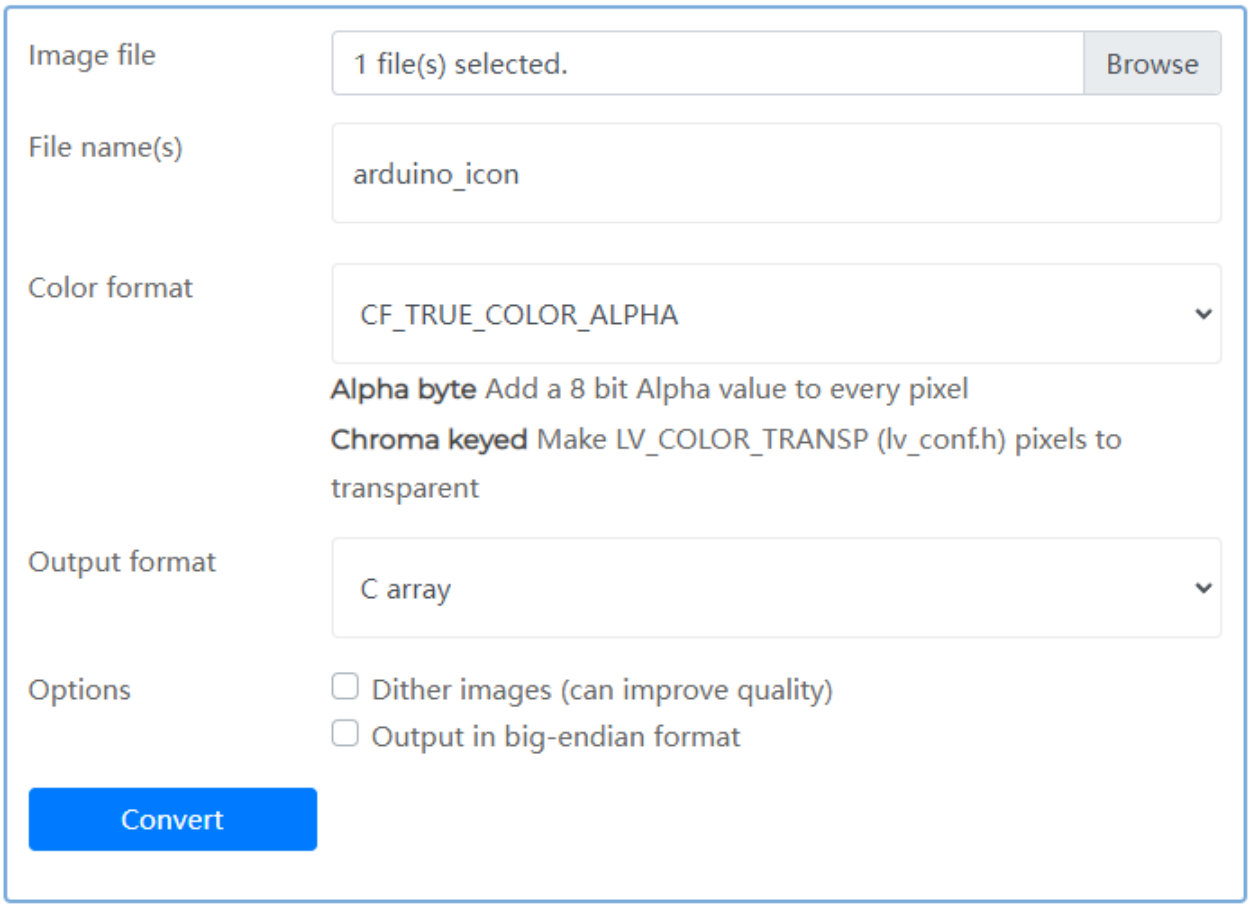
```
#define LV_FONT_MONTSERRAT_22 1
```

Make Sure that **USB Mode** is set to **USB OTG** under **Tools**, and Remember to select `ESP32S3 Dev Module` and port, then click upload.

Make Icons

To make your own icons, get a picture, better be a png with transparent background, resize it to about 50x50, then use lvgl online image converter to convert it to C array.

Set the output name, it will be the name of the image data variable, so make it "code friendly". Set Color format to `CF_TRUE_COLOR_ALPHA` and output to `C array`. Click `Convert`, it will download a `.c` file.



Then, copy the file to your project, and change the first few line, or it will raise compile error `fatal error: lvgl/lvgl.h: No such file or directory`

```
#if defined(LV_LVGL_H_INCLUDE_SIMPLE)
#include "lvgl.h"
#else
#include "lvgl/lvgl.h"
#endif
```

To

```
#include "lvgl.h"
```

Now add a line to `.ino` file to declare it.

```
LV_IMG_DECLARE(<name>);
```

That's it, you can now use it to create a button:

```
createIconButton(&<name>, 0, 0, <onPressed>, <onReleased>, <onTap>);
```

You can see all above in the example for a reference.

Create a shortcut

Some Apps have a keyboard shortcut like `CONSUMER_CONTROL_CALCULATOR`. You can launch it with `ConsumerControl`. Others you need to create a keyboard shortcut, and simulate the shortcut with Termod S3.

For Windows 10 and 11, you can make a keyboard shortcut to a desktop shortcut. First, create a shortcut of a app to desktop. Then, right click the shortcut, click `Properties`.

You will see a shortcut options, click on it and press a shortcut key, like `Ctrl+Alt+Shift+1`. Then click `Apply` and `OK`.

Then in code, simulate it like in the example `openKicad`.

```
void openKicad(_lv_event_t* event) {
    Keyboard.press(KEY_LEFT_CTRL);
    Keyboard.press(KEY_LEFT_ALT);
    Keyboard.press(KEY_LEFT_SHIFT);
    Keyboard.press('1');
    Keyboard.releaseAll();
}
```

You can change keys.

- To control keyboard, use `Keyboard`, checkout all keys under USBHIDKeyboard.h
- To control volume and music, use `ConsumerControl`, checkout all controls under USBHIDConsumerControl.h

Source code

macro_pad.ino

lv_helper.cpp

lv_helper.h

```
#if ARDUINO_USB_MODE
#warning This sketch should be used when USB is in OTG mode
void setup(){}
void loop(){}
#else

#include "lv_helper.h"
#include "USB.h"
#include "USBHIDKeyboard.h"
#include "USBHIDConsumerControl.h"

USBHIDConsumerControl ConsumerControl;
USBHIDKeyboard Keyboard;
```

(continues on next page)

(continued from previous page)

```
#define CONSUMER_CONTROL_INTERNET_BROWSER 0x0196

LV_IMG_DECLARE(calculator_icon);
LV_IMG_DECLARE(kicad_icon);
LV_IMG_DECLARE(arduino_icon);
LV_IMG_DECLARE(vscode_icon);

#define KEYBOARD_LAYOUT_MAC 0
#define KEYBOARD_LAYOUT_WINDOWS 1
// If you are using a Mac, set this to KEYBOARD_LAYOUT_MAC
#define KEYBOARD_LAYOUT KEYBOARD_LAYOUT_WINDOWS

#define LAYOUT_WIDTH 4
#define LAYOUT_HEIGHT 3
#define PADDING 2
#define BUTTON_WIDTH 320 / LAYOUT_WIDTH - (2 * PADDING)
#define BUTTON_HEIGHT 240 / LAYOUT_HEIGHT - (2 * PADDING)

static lv_style_t pressedStyle;

lv_obj_t* createButton(int x, int y, void (*onPressed)(_lv_event_t*), void␣
↪(*onReleased)(_lv_event_t*), void (*onTap)(_lv_event_t*));
void createTextButton(char* text, int x, int y, void (*onPressed)(_lv_event_t*), void␣
↪(*onReleased)(_lv_event_t*), void (*onTap)(_lv_event_t*));
void createIconButton(const lv_img_dsc_t *image, int x, int y, void (*onPressed)(_lv_
↪event_t*), void (*onReleased)(_lv_event_t*), void (*onTap)(_lv_event_t*));


void setup() {
  Serial.begin(115200);
  lh_init(DISPLAY_LANDSCAPE);

  // Create button style, when button is pressed, glow it
  lv_style_init(&pressedStyle);
  lv_style_set_border_color(&pressedStyle, lv_color_hex(0x33dddd));
  lv_style_set_shadow_color(&pressedStyle, lv_color_hex(0x33dddd));
  lv_style_set_shadow_width(&pressedStyle, 2);
  Keyboard.begin();
  ConsumerControl.begin();
  USB.begin();


  createIconButton(&calculator_icon, 0, 0, NULL, NULL, openCalculator);
  createIconButton(&kicad_icon, 1, 0, NULL, NULL, openKicad);
  createIconButton(&arduino_icon, 2, 0, NULL, NULL, openArduino);
  createIconButton(&vscode_icon, 3, 0, NULL, NULL, openVSCode);
  createTextButton(LV_SYMBOL_VOLUME_MID, 0, 1, volumeDownPressed, volumeDownReleased,␣
↪NULL);
  createTextButton(LV_SYMBOL_VOLUME_MAX, 1, 1, volumeUpPressed, volumeUpReleased, NULL);
  createTextButton(LV_SYMBOL_MUTE, 2, 1, NULL, NULL, mute);
  createTextButton(LV_SYMBOL_HOME, 3, 1, NULL, NULL, home);
  createTextButton(LV_SYMBOL_COPY, 0, 2, NULL, NULL, copy);
```

(continues on next page)

(continued from previous page)

```
  createTextButton(LV_SYMBOL_PASTE, 1, 2, NULL, NULL, paste);
  createTextButton(LV_SYMBOL_LEFT, 2, 2, NULL, NULL, leftDesktop);
  createTextButton(LV_SYMBOL_RIGHT, 3, 2, NULL, NULL, rightDesktop);
}

void loop() {
  lv_timer_handler();
}

void openCalculator(_lv_event_t* event) {
  ConsumerControl.press(CONSUMER_CONTROL_CALCULATOR);
  ConsumerControl.release();
}

void openKicad(_lv_event_t* event) {
  Keyboard.press(KEY_LEFT_CTRL);
  Keyboard.press(KEY_LEFT_ALT);
  Keyboard.press(KEY_LEFT_SHIFT);
  Keyboard.press('1');
  Keyboard.releaseAll();
}

void openArduino(_lv_event_t* event) {
  Keyboard.press(KEY_LEFT_CTRL);
  Keyboard.press(KEY_LEFT_ALT);
  Keyboard.press(KEY_LEFT_SHIFT);
  Keyboard.press('2');
  Keyboard.releaseAll();
}

void openVSCode(_lv_event_t* event) {
  Keyboard.press(KEY_LEFT_CTRL);
  Keyboard.press(KEY_LEFT_ALT);
  Keyboard.press(KEY_LEFT_SHIFT);
  Keyboard.press('3');
  Keyboard.releaseAll();
}

void volumeDownPressed(_lv_event_t* event) {
  ConsumerControl.press(CONSUMER_CONTROL_VOLUME_DECREMENT);
}
void volumeDownReleased(_lv_event_t* event) {
  ConsumerControl.release();
}
void volumeUpPressed(_lv_event_t* event) {
  ConsumerControl.press(CONSUMER_CONTROL_VOLUME_INCREMENT);
}
void volumeUpReleased(_lv_event_t* event) {
  ConsumerControl.release();
}
void mute(_lv_event_t* event) {
  ConsumerControl.press(CONSUMER_CONTROL_MUTE);
```

(continues on next page)

(continued from previous page)

```
  ConsumerControl.release();
}
void copy(_lv_event_t* event) {
  #if KEYBOARD_LAYOUT == KEYBOARD_LAYOUT_WINDOWS
  Keyboard.press(KEY_LEFT_CTRL);
  #else
  Keyboard.press(KEY_LEFT_GUI);
  #endif
  Keyboard.press('c');
  Keyboard.releaseAll();
}

void paste(_lv_event_t* event) {
  #if KEYBOARD_LAYOUT == KEYBOARD_LAYOUT_WINDOWS
  Keyboard.press(KEY_LEFT_CTRL);
  #else
  Keyboard.press(KEY_LEFT_GUI);
  #endif
  Keyboard.press('v');
  Keyboard.releaseAll();
}

void home(_lv_event_t* event) {
  #if KEYBOARD_LAYOUT == KEYBOARD_LAYOUT_WINDOWS
  Keyboard.press(KEY_LEFT_GUI);
  Keyboard.press('d');
  #endif
  Keyboard.releaseAll();
}

void leftDesktop(_lv_event_t* event) {
  Keyboard.press(KEY_LEFT_CTRL);
  #if KEYBOARD_LAYOUT == KEYBOARD_LAYOUT_WINDOWS
  Keyboard.press(KEY_LEFT_GUI);
  #endif
  Keyboard.press(KEY_LEFT_ARROW);
  Keyboard.releaseAll();
}

void rightDesktop(_lv_event_t* event) {
  Keyboard.press(KEY_LEFT_CTRL);
  #if KEYBOARD_LAYOUT == KEYBOARD_LAYOUT_WINDOWS
  Keyboard.press(KEY_LEFT_GUI);
  #endif
  Keyboard.press(KEY_RIGHT_ARROW);
  Keyboard.releaseAll();
}

// create a button
lv_obj_t* createButton(int x, int y, void (*onPressed)(_lv_event_t*), void␣
↪(*onReleased)(_lv_event_t*), void (*onTap)(_lv_event_t*)) {
  int top = x * 80 + PADDING;
```

(continues on next page)

(continued from previous page)

```
  int left = y * 80 + PADDING;

  lv_obj_t* btn = lv_obj_create(lv_scr_act());
  lv_obj_set_size(btn, BUTTON_WIDTH, BUTTON_HEIGHT);
  lv_obj_align(btn, LV_ALIGN_TOP_LEFT, top, left);
  lv_obj_clear_flag(btn, LV_OBJ_FLAG_SCROLLABLE);
  lv_obj_add_style(btn, &pressedStyle, LV_STATE_PRESSED);
  if (onPressed != NULL) {
    lv_obj_add_event_cb(btn, onPressed, LV_EVENT_PRESSED, NULL);
  }
  if (onReleased != NULL) {
    lv_obj_add_event_cb(btn, onReleased, LV_EVENT_RELEASED, NULL);
  }
  if (onTap != NULL) {
    lv_obj_add_event_cb(btn, onTap, LV_EVENT_CLICKED, NULL);
  }

  return btn;
}

void createTextButton(char* text, int x, int y, void (*onPressed)(_lv_event_t*), void␣
↪(*onReleased)(_lv_event_t*), void (*onTap)(_lv_event_t*)) {
  lv_obj_t* btn = createButton(x, y, onPressed, onReleased, onTap);
  lv_obj_t* label = lv_label_create(btn);
  lv_label_set_text(label, text);
  lv_obj_set_style_text_font(label, &lv_font_montserrat_22, 0);
  lv_obj_center(label);
}

void createIconButton(const lv_img_dsc_t *image, int x, int y, void (*onPressed)(_lv_
↪event_t*), void (*onReleased)(_lv_event_t*), void (*onTap)(_lv_event_t*)){
  lv_obj_t* btn = createButton(x, y, onPressed, onReleased, onTap);
  lv_obj_t* img = lv_img_create(btn);
  lv_img_set_src(img, image);
  lv_obj_center(img);
}

#endif /* ARDUINO_USB_MODE */
```

```
#include "lv_helper.h"

TFT_eSPI lh_tft = TFT_eSPI();
TAMC_FT62X6 lh_tp = TAMC_FT62X6();

static lv_disp_draw_buf_t lh_draw_buf;
static lv_color_t lh_buf[ DISPLAY_WIDTH * 10 ];
static lv_disp_drv_t lh_disp_drv;
static lv_indev_drv_t lh_indev_drv;

uint16_t width, height;

void lh_init(int rotation){
```

(continues on next page)

(continued from previous page)

```
  Wire.begin();
  lh_tp.begin();
  lv_init();
  lh_tft.begin();
  if (rotation == 1 || rotation == 3){
    width = DISPLAY_HEIGHT;
    height = DISPLAY_WIDTH;
  } else {
    width = DISPLAY_WIDTH;
    height = DISPLAY_HEIGHT;
  }
  lh_tft.setRotation(rotation);
  lh_tp.setRotation(rotation);

  lv_disp_draw_buf_init( &lh_draw_buf, lh_buf, NULL, DISPLAY_WIDTH * 10 );

  /*Initialize the display*/
  lv_disp_drv_init( &lh_disp_drv );
  /*Change the following line to your display resolution*/
  lh_disp_drv.hor_res = width;
  lh_disp_drv.ver_res = height;
  lh_disp_drv.flush_cb = lh_disp_flush;
  lh_disp_drv.draw_buf = &lh_draw_buf;
  lv_disp_drv_register( &lh_disp_drv );

  /*Initialize the (dummy) input device driver*/
  lv_indev_drv_init( &lh_indev_drv );
  lh_indev_drv.type = LV_INDEV_TYPE_POINTER;
  lh_indev_drv.read_cb = lh_touchpad_read;
  lv_indev_drv_register( &lh_indev_drv );
}

/* Display flushing */
void lh_disp_flush(lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p) {
  uint32_t w = (area->x2 - area->x1 + 1);
  uint32_t h = (area->y2 - area->y1 + 1);

  lh_tft.startWrite();
  lh_tft.setAddrWindow( area->x1, area->y1, w, h );
  lh_tft.pushColors( ( uint16_t * )&color_p->full, w * h, true );
  lh_tft.endWrite();

  lv_disp_flush_ready(disp);
}
/*Read the touchpad*/
void lh_touchpad_read(lv_indev_drv_t * indev_driver, lv_indev_data_t * data) {
  lh_tp.read();

  if (!lh_tp.isTouched) {
    data->state = LV_INDEV_STATE_RELEASED;
  }
  else{
```

(continues on next page)

(continued from previous page)

```
    data->state = LV_INDEV_STATE_PRESSED;
    /*Set the coordinates*/
    data->point.x = lh_tp.points[0].x;
    data->point.y = lh_tp.points[0].y;
  }
}
```

```
#ifndef LV_HELPER_H
#define LV_HELPER_H

#include <lvgl.h>
#include "TAMC_FT62X6.h"
#include "Wire.h"
#include <TFT_eSPI.h>

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1

#define DISPLAY_WIDTH  240
#define DISPLAY_HEIGHT 320

/* Display flushing */
void lh_disp_flush(lv_disp_drv_t *disp, const lv_area_t *area, lv_color_t *color_p);
/*Read the touchpad*/
void lh_touchpad_read(lv_indev_drv_t * indev_driver, lv_indev_data_t * data);

void lh_init(int rotation);
#endif // LV_HELPER_H
```

Factory Test

This example is for Factory test. to test every hardware is basicly working.

Note: If you haven't download the code:

Download examples from github `termod-s3`

Unzip the downloaded `termod-s3-main.zip`

Or just clone the repository

```
git clone https://github.com/TAMCTec/termod-s3.git
```

Open `termod-s3/examples/factory_test/factory_test.ino` with Arduino IDE.

Remember to select `ESP32S3 Dev Module` and port, then click upload.

Source code

factory_test.ino

```
#include <TFT_eSPI.h>
#include <TAMC_FT62X6.h>
#include <Wire.h>
#include "FS.h"
#include "SD.h"
#include "SPI.h"

#define DISPLAY_PORTRAIT 2
#define DISPLAY_LANDSCAPE 3
#define DISPLAY_PORTRAIT_FLIP 0
#define DISPLAY_LANDSCAPE_FLIP 1

TFT_eSPI tft = TFT_eSPI();
TAMC_FT62X6 tp = TAMC_FT62X6();

#define FF18 &FreeSans12pt7b
#define GFXFF 1

// io index
uint8_t i = 0;
uint8_t lastI = -1;
// last millis
uint32_t t = 0;

uint8_t rowHeight = 30;

String touchInfo;
String batteryInfo;
String sdCardInfo;
String buttonInfo;

static const uint8_t BAT_LV = 1;
static const uint8_t CHG = 2;
static const uint8_t SD_CS = 21;

bool getChargingState() {
  return !digitalRead(CHG);
}

float getBatteryVoltage() {
  int analogVolt = analogReadMilliVolts(1);
  float voltage = analogVolt / 1000.0;
  voltage = voltage * (100.0 + 200.0) / 200.0;
  return voltage;
}

float getBatteryCapacity() {
  float voltage = getBatteryVoltage();
  float capacity = (voltage - 3.3) / (4.2 - 3.3) * 100.0;
  capacity = constrain(capacity, 0, 100);
  return capacity;
}
```

(continues on next page)

(continued from previous page)

```
void setup() {
  Serial.begin(115200);
  // SD Card
  bool sdCardPresent = SD.begin(SD_CS);
  sdCardInfo += String("Mount") + (sdCardPresent ? "ed" : " Failed");

  Wire.begin();
  tft.init();
  if (!tp.begin()) {
    Serial.println("Touchscreen not found");
    while (1);
  }
  tp.setRotation(DISPLAY_LANDSCAPE);
  tft.setRotation(DISPLAY_LANDSCAPE);

  pinMode(0, INPUT_PULLUP);

  tft.setFreeFont(&FreeSans9pt7b);
  tft.setTextDatum(TL_DATUM);
  tft.setTextColor(TFT_WHITE);
  t = millis();
  tft.fillScreen(TFT_BLACK);
  uint8_t currentY = 10;
  tft.drawString("Factory test", 10, currentY, 1);
  currentY += rowHeight;
  tft.drawString("Touch:", 10, currentY, 1);
  currentY += rowHeight;
  tft.drawString("Battery:", 10, currentY, 1);
  currentY += rowHeight;
  tft.drawString("SD Card:", 10, currentY, 1);
  tft.drawString(sdCardInfo.c_str(), 110, currentY, 1);
  currentY += rowHeight;
  tft.drawString("Button IO0:", 10, currentY, 1);
  Serial.println("Hello");
}

void loop() {
  int x = 0;
  int y = 0;
  String newTouchInfo;
  String newBatteryInfo;
  String newSdCardInfo;
  String newButtonInfo;

  // Touch
  tp.read();
  if (tp.isTouched) {
    x = tp.points[0].x;
    y = tp.points[0].y;
    newTouchInfo += "[" + String(x) + ", " + String(y) + "]";
    if (tp.touches == 2){
      x = tp.points[1].x;
```

(continues on next page)

(continued from previous page)

```
      y = tp.points[1].y;
      newTouchInfo += ", [" + String(x) + ", " + String(y) + "]";
    }
  } else {
    newTouchInfo += "No touch";
  }

  // Battery
  float batteryVoltage = getBatteryVoltage();
  float batteryPercentage = getBatteryCapacity();
  bool charging = getChargingState();
  newBatteryInfo += String(batteryVoltage) + "V, " + String(batteryPercentage) + "%" +␣
↪(charging ? " (charging)" : "");

  // Button
  bool buttonPressed = digitalRead(0) == LOW;
  newButtonInfo += buttonPressed ? String("Pressed") : String("Released");

  uint8_t currentY = 10;
  currentY += rowHeight;
  if (newTouchInfo != touchInfo) {
    touchInfo = newTouchInfo;
    tft.fillRect(110, currentY, 210, rowHeight, TFT_BLACK);
    tft.drawString(touchInfo.c_str(), 110, currentY, 1);
  }
  currentY += rowHeight;
  if (newBatteryInfo != batteryInfo) {
    batteryInfo = newBatteryInfo;
    tft.fillRect(110, currentY, 210, rowHeight, TFT_BLACK);
    tft.drawString(batteryInfo.c_str(), 110, currentY, 1);
  }
  currentY += rowHeight;
  currentY += rowHeight;
  if (newButtonInfo != buttonInfo) {
    buttonInfo = newButtonInfo;
    tft.fillRect(110, currentY, 210, rowHeight, TFT_BLACK);
    tft.drawString(buttonInfo.c_str(), 110, currentY, 1);
  }
  delay(10);
}
```

1.2.3 Reference

Defines

USB_VID

USB_PID

EXTERNAL_NUM_INTERRUPTS

NUM_DIGITAL_PINS

NUM_ANALOG_INPUTS

BUILTIN_LED

LED_BUILTIN

RGB_BUILTIN

RGB_BRIGHTNESS

analogInputToDigitalPin(p)

digitalPinToInterrupt(p)

digitalPinHasPWM(p)

DISPLAY_PORTRAIT

DISPLAY_LANDSCAPE

DISPLAY_PORTRAIT_FLIP

DISPLAY_LANDSCAPE_FLIP

DISPLAY_WIDTH

DISPLAY_HEIGHT

Functions

float **getBatteryVoltage**()

Get battery voltage in volts

Returns
Battery voltage in volts

float **getBatteryCapacity**()

Get battery level in percent

Returns
Battery level in percent(0-100)

bool **getChargingState**()

Get battery charge state

Returns
Battery charge state(true=charging, false=not charging)

void **setOnChargeStart**(void (*func)())

Set on charge start callback

Parameters
func – On charge start Callback function

void **setOnChargeEnd**(void (*func)())

Set on charge end callback

Parameters
func – On charge end Callback function

Variables

static const uint8_t **LED_BUILTIN** = SOC_GPIO_PIN_COUNT + 48

static const uint8_t **TX** = 43

static const uint8_t **RX** = 44

static const uint8_t **SDA** = 8

static const uint8_t **SCL** = 9

static const uint8_t **SS** = 10

static const uint8_t **MOSI** = 11

static const uint8_t **MISO** = 13

static const uint8_t **SCK** = 12

static const uint8_t **A0** = 1

static const uint8_t **A1** = 2

static const uint8_t **A2** = 3

static const uint8_t **A3** = 4

static const uint8_t **A4** = 5

static const uint8_t **A5** = 6

static const uint8_t **A6** = 7

static const uint8_t **A7** = 8

static const uint8_t **A8** = 9

static const uint8_t **A9** = 10

static const uint8_t **A10** = 11

static const uint8_t **A11** = 12

static const uint8_t **A12** = 13

static const uint8_t **A13** = 14

static const uint8_t **A14** = 15

static const uint8_t **A15** = 16

static const uint8_t **A16** = 17

static const uint8_t **A17** = 18

static const uint8_t **A18** = 19

static const uint8_t **A19** = 20

static const uint8_t **T1** = 1

static const uint8_t **T2** = 2

static const uint8_t **T3** = 3

static const uint8_t **T4** = 4

static const uint8_t **T5** = 5

static const uint8_t **T6** = 6

static const uint8_t **T7** = 7

static const uint8_t **T8** = 8

static const uint8_t **T9** = 9

static const uint8_t **T10** = 10

static const uint8_t **T11** = 11

static const uint8_t **T12** = 12

static const uint8_t **T13** = 13

static const uint8_t **T14** = 14

static const uint8_t **BAT_LV** = 1

static const uint8_t **CHG** = 2

static const uint8_t **TFT_CS** = 10

static const uint8_t **TFT_DC** = 18

static const uint8_t **TFT_RST** = 14

static const uint8_t **TFT_BCKL** = 48

static const uint8_t **SD_CS** = 21

static const uint8_t **SD_CD** = 47

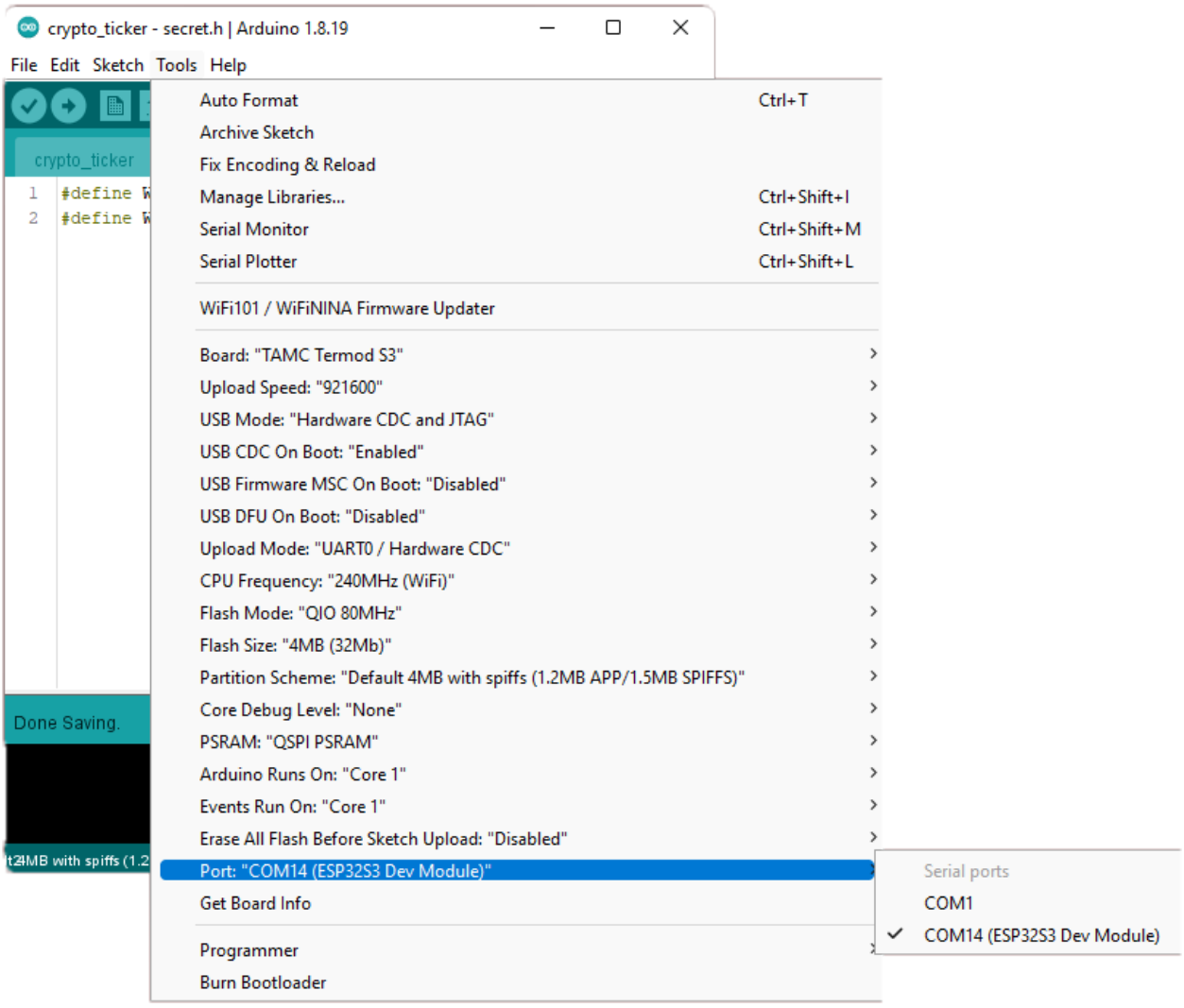
1.3 ESP-IDF Usage (Comming soon)

Coming soon…

1.4 FAQ

1.4.1 Error opening serial port 'COM15'.

Make sure you have choose the coresponding port under **Tools** => **Port**, the port wil have a (ESP32 Dev Module) after it.



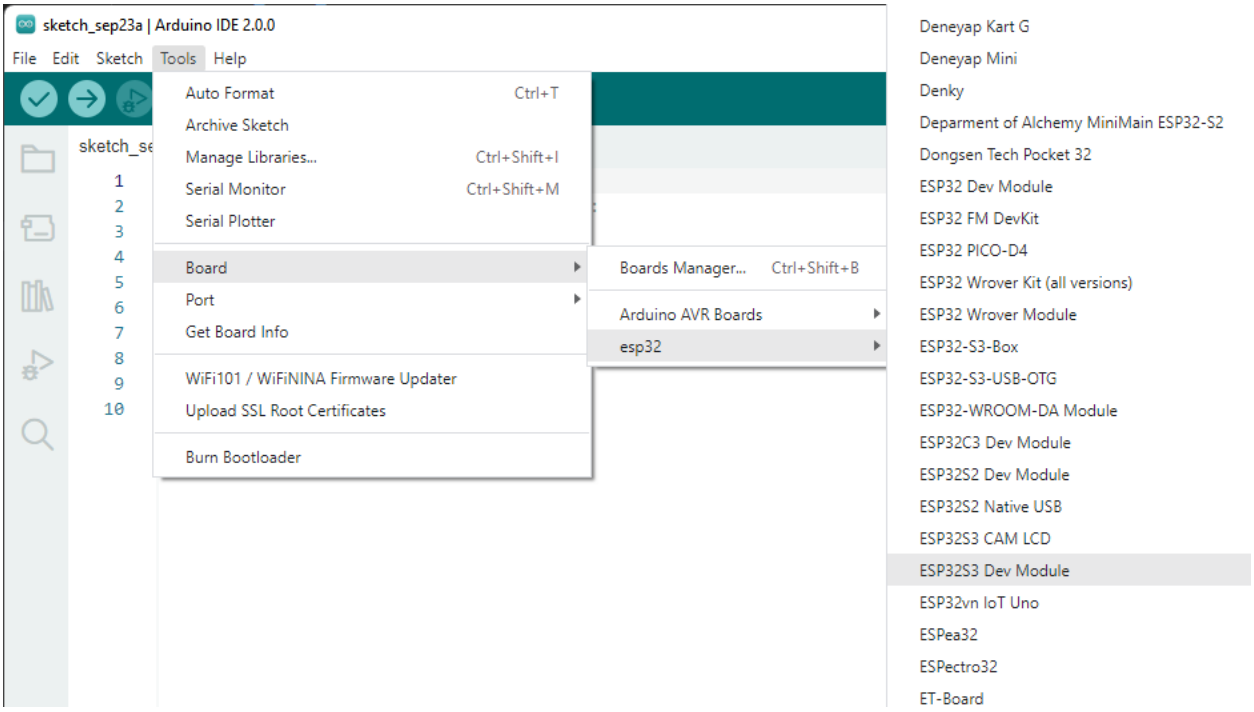
1.4.2 XXXX is not defined

if something like these is not defined:

- BAT_LV
- CHG
- TFT_CS
- TFT_DC
- TFT_RST

- TFT_BCKL
- SD_CS
- SD_CD
- DISPLAY_PORTRAIT
- DISPLAY_LANDSCAPE
- DISPLAY_PORTRAIT_FLIP
- DISPLAY_LANDSCAPE_FLIP
- DISPLAY_WIDTH
- DISPLAY_HEIGHT
- getBatteryVoltage
- getBatteryCapacity
- getChargingState
- setOnChargeStart
- setOnChargeEnd

Make sure you have select TAMC Termod S3 under **Tools** => **Board**.



1.4.3 the selected serial port [22712] Failed to execute script 'esptool' due to unhandled exception! does not exist or your board is not connected

Make sure you have choose the coresponding port under **Tools** => **Port**, the port wil have a (ESP32 Dev Module) after it.

If you do have the correct port selected, try forcing te board to flash mode, by holding down th IO0 button, and press and release the reset button, then release the IO0 button. After it's in flash mode, make sure check again the port is selected, as the port number might change.

After a manual reset, the board is not able to restart after upload done. yYou also need to manually reset the board after upload.

INDEX

U