Advanced System Software
（先端システムソフトウェア）

#2 (2018/10/1)

CSC.T431, 2018-3Q
Mon/Thu 9:00-10:30, W832

Instructor: Takuo Watanabe（渡部卓雄）
Department of Computer Science

e-mail: takuo∅c.titech.ac.jp
https://titech-aos.github.io
ext: 3690, office: W8E-805

Agenda

- Programming Embedded Systems (1)
  - Development Environment
  - Basic Concepts
  - Code Structure
- Forming Development Teams

M5Stack

- An open-source modular toolkit for IoT devices
- An M5Stack module is an enclosed device that consists of an ESP32 SoC and several peripheral components, including three buttons, an LCD, and a speaker.
- http://m5stack.com

- In this class, we use M5Stack Gray, which has built-in 9-DOF motion sensor

M5Stack Gray H/W Spec.

- Processor: ESP32 SoC (Expressif)
  - CPU: Dual 240MHz LX6 (Tensilica)
  - Memory: 512KB
- Flash (program storage): 4MB
- Peripheral Components:
  - 320x240 Color TFT LCD, 3 buttons, speaker, 9-axis motion sensor, Li-po battery
- Ports:
  - USB-C, MBUS, Grove (I2C), micro-SD
- Size: 54 x 54 x 17mm
- Weight: 120g

ESP32 (1/2)

- Series of low-cost, low-power SoC with integrated WiFi & Bluetooth capabilities.
  - https://www.espressif.com
- Processors:
  - Dual 160/240MHz Xtensa LX6 (Tensilica)
  - ULP (Ultra Low Power) Co-processor
- Memory: 512KB SRAM
- Wireless
  - WiFi: 801.11 b/g/n (2.4GHz)
  - Bluetooth v4.2 BR/EDR & BLE

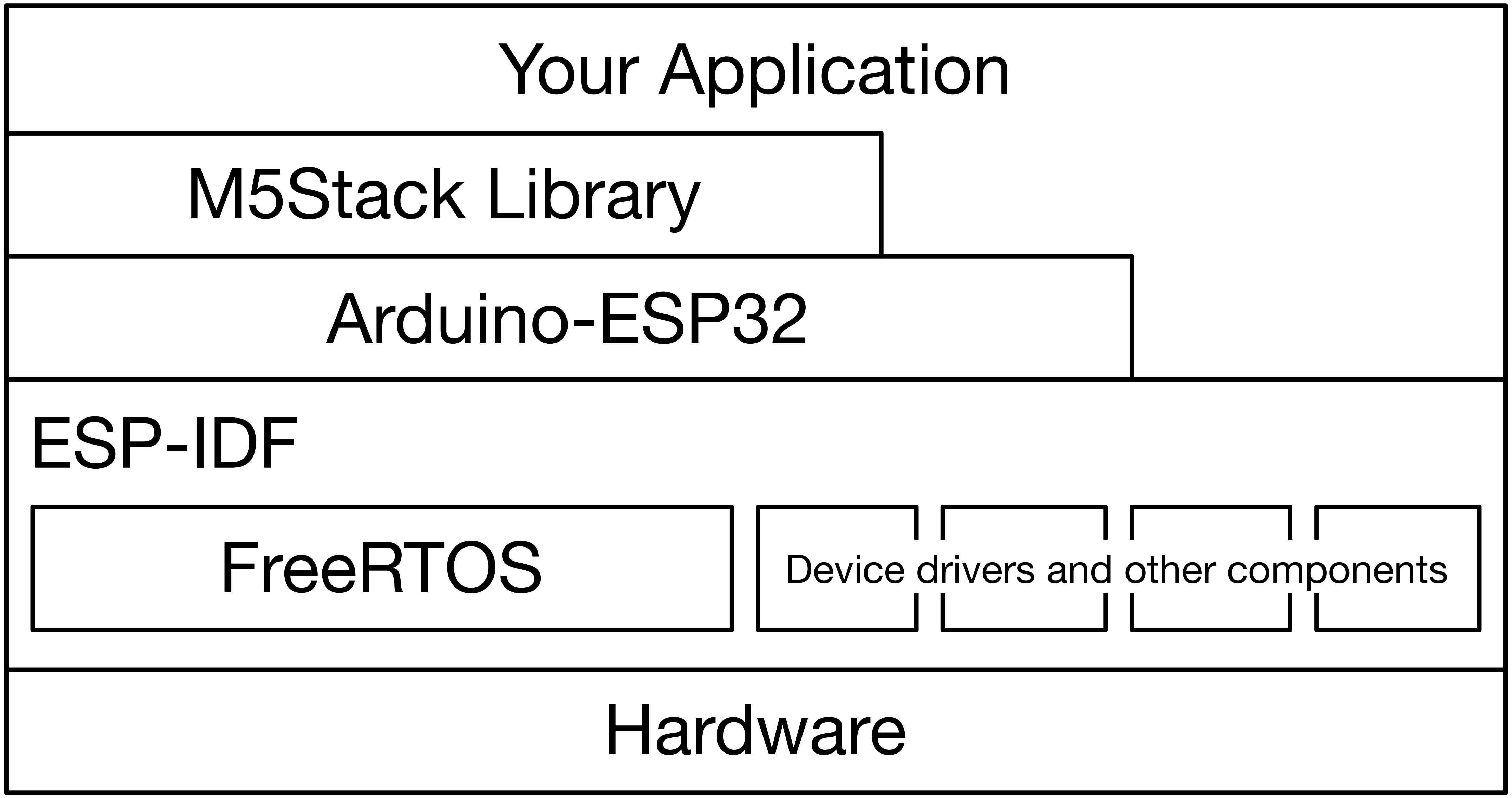
ESP32 (2/2)

- Peripheral I/F
  - 18 x 12bit ADC, 2 x 8bit DAC
  - 10 x capacitive touch sensors, temperature sensor
  - 4 x SPI, 2 x I2S, 2 x I2C, 3 x UART
  - SD/SDIO/CE-ATA/MMC/eMMC host controller
  - SDIO/SPI slave controller, Ethernet MAC I/F
  - CAN bus 2.0, 8 x Infrared TX/RX
  - Motor PWM, 16 x LED PWM, Hall Effect Sensor
  - Ultra low power analog pre-amplifier
- Security
  - IEEE 802.11 WFA, WPA/WPA2 and WAPI, Secure boot
  - Flash encryption, 1024bit OTP, Cryptographic H/W acceleration: AES, SHA-2, RSA, ESS, RNG

Required Tools/Libraries for Development

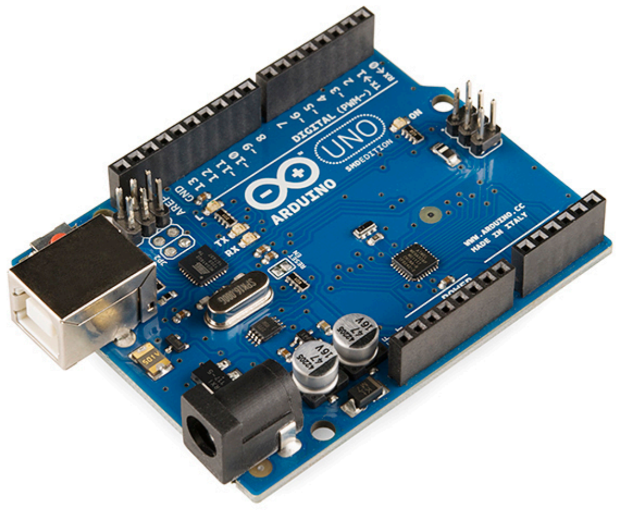
- Toolchain
  - GCC for LX6, binutils, gdb, etc.
- ESP-IDF
  - Espressif IoT Development Framework
  - Official Development Framework for ESP32
    - FreeRTOS, libraries, device drivers
    - command-line development tools
- Arduino Core for ESP32 (Arduino-ESP32)
  - Library for using ESP32 as Arduino
- M5Stack Library
  - Library for making use of M5Stack components (LCD, buttons, etc.)

Application Layer



Arduino

`https://www.arduino.cc`



- An open-source microcontroller-based development board and its software development framework
- Designed for people without engineering background
- Originally based on Atmel AVR

- Various processors (including ESP32) are now supported. From the software point of view, Arduino provides a common easy-to-develop abstraction layer for embedded systems.

using Arduino library

```
#include <Arduino.h>

#define LED_PIN 21

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    digitalWrite(LED_PIN, HIGH);
    delay(500);
    digitalWrite(LED_PIN, LOW);
    delay(500);
}
```

without using Arduino library

```
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/gpio.h"

#define LED_PIN 21

void blink_task(void *pvParameter) {
    gpio_pad_select_gpio(LED_PIN);
    gpio_set_direction(LED_PIN, GPIO_MODE_OUTPUT);
    for (;;) {
        gpio_set_level(LED_PIN, 0);
        vTaskDelay(500 / portTICK_PERIOD_MS);
        gpio_set_level(LED_PIN, 1);
        vTaskDelay(500 / portTICK_PERIOD_MS);
    }
}

void app_main() {
    xTaskCreate(&blink_task, "blink_task",
                configMINIMAL_STACK_SIZE, NULL, 5, NULL);
}
```

Basic Code Structure

Arduino code template

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

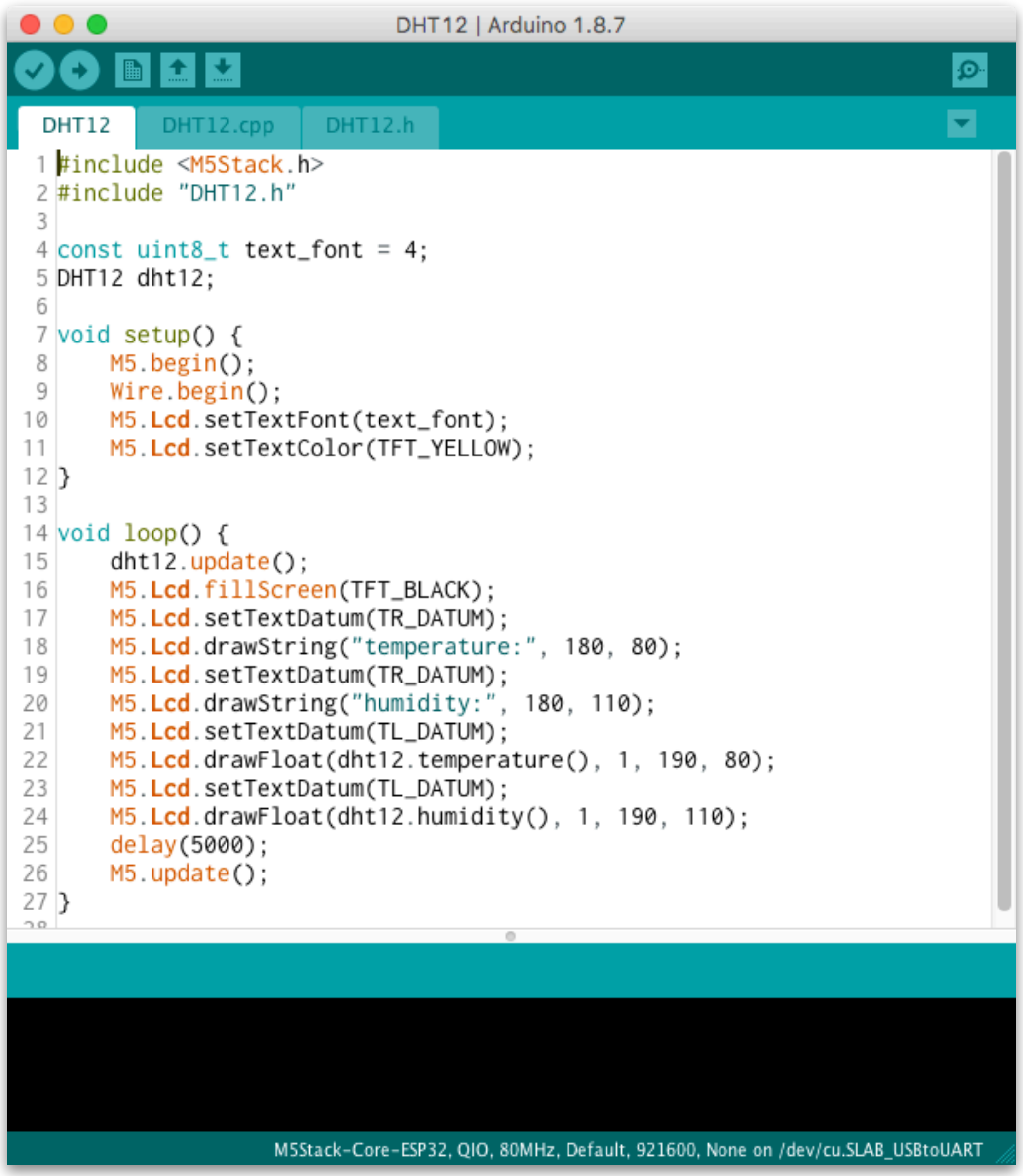
The application works like

```
int main() {
    setup();
    for (;;) loop();
}
```

The 'main' function in Arduino-ESP32

```
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "Arduino.h"

...

void loopTask(void *pvParameters) {
    setup();
    for (;;) {
        loop();
    }
}

extern "C" void app_main() {
    initArduino();
    xTaskCreatePinnedToCore(loopTask, "loopTask", 8192, NULL,
                            1, NULL, ARDUINO_RUNNING_CORE);
}
```

Arduino IDE



```
#include <M5Stack.h>
#include "DHT12.h"

const uint8_t text_font = 4;
DHT12 dht12;

void setup() {
    M5.begin();
    Wire.begin();
    M5.Lcd.setTextFont(text_font);
    M5.Lcd.setTextColor(TFT_YELLOW);
}

void loop() {
    dht12.update();
    M5.Lcd.fillScreen(TFT_BLACK);
    M5.Lcd.setTextDatum(TR_DATUM);
    M5.Lcd.drawString("temperature:", 180, 80);
    M5.Lcd.setTextDatum(TR_DATUM);
    M5.Lcd.drawString("humidity:", 180, 110);
    M5.Lcd.setTextDatum(TL_DATUM);
    M5.Lcd.drawFloat(dht12.temperature(), 1, 190, 80);
    M5.Lcd.setTextDatum(TL_DATUM);
    M5.Lcd.drawFloat(dht12.humidity(), 1, 190, 110);
    delay(5000);
    M5.update();
}
```

M5Stack-Core-ESP32, QIO, 80MHz, Default, 921600, None on /dev/cu.SLAB_USBtoUART

- A simple IDE for Arduino
- Available for Mac, Linux, and Windows
- Good starting point for developing Arduino-based applications
- Code editing facility is too simple

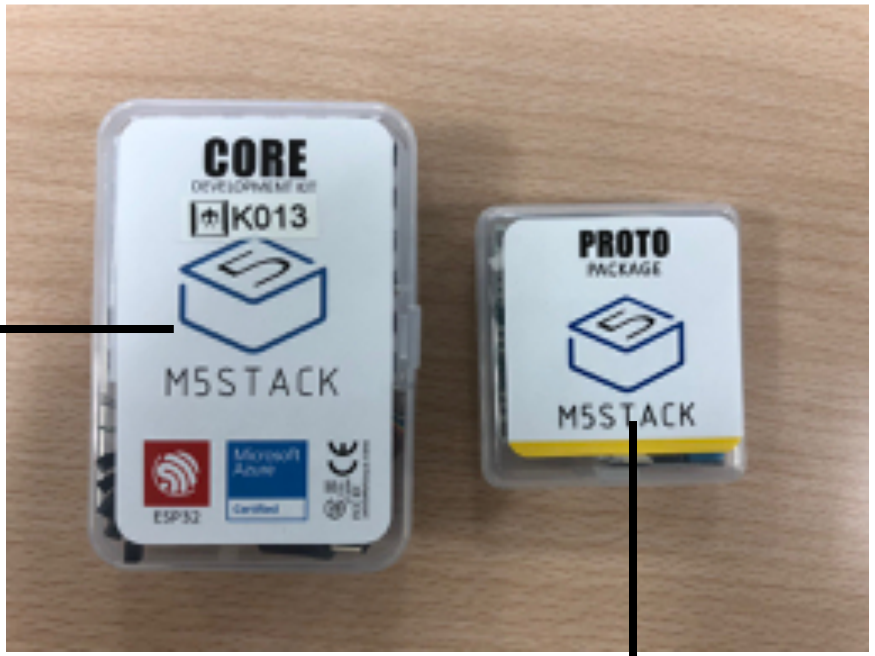
IDEs for Developing M5Stack Applications

- Arduino IDE
  - Simple GUI-based IDE for Arduino
- PletformIO
  - IDE for developing various devices
  - CUI or Editor Plugins (VSCode, Atom)
- ESP-IDF
  - CUI toolset distributed with ESP-IDF (framework)

- For any IDE, you need to additionally install the toolchain, framework and libraries for M5Stack.

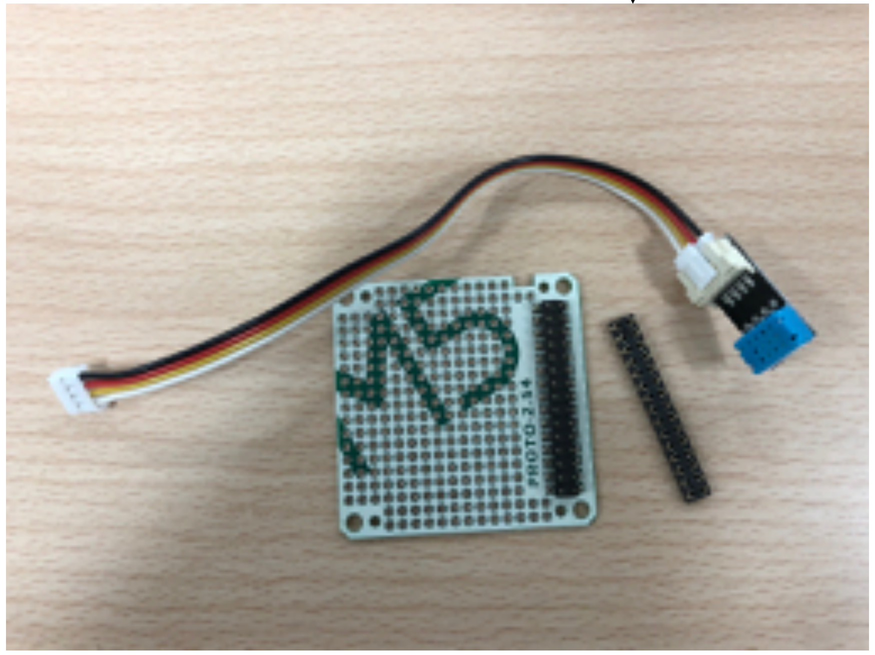
Development Kit

M5Stack Gray, USB-A to USB-C cable, GPIO cable



Two LEDs with built-in resistors (red or green or yellow)

DHT12 (temp. & humidity sensor), prototyping board, bus socket



Example Applications

- M5Stack Samples
  - https://github.com/titech-aos/M5Samples_Arduino
  - https://github.com/titech-aos/M5Samples_ESP-IDF
  - https://github.com/titech-aos/M5Samples_PIO
- How to run (w/ Arduino IDE)
  - 1. clone git repository

    ```
    git clone https://github.com/titech-aos/M5Samples_PIO.git
    ```
  - 2. open one of '.ino' file with Arduino IDE
  - 3. compile and upload

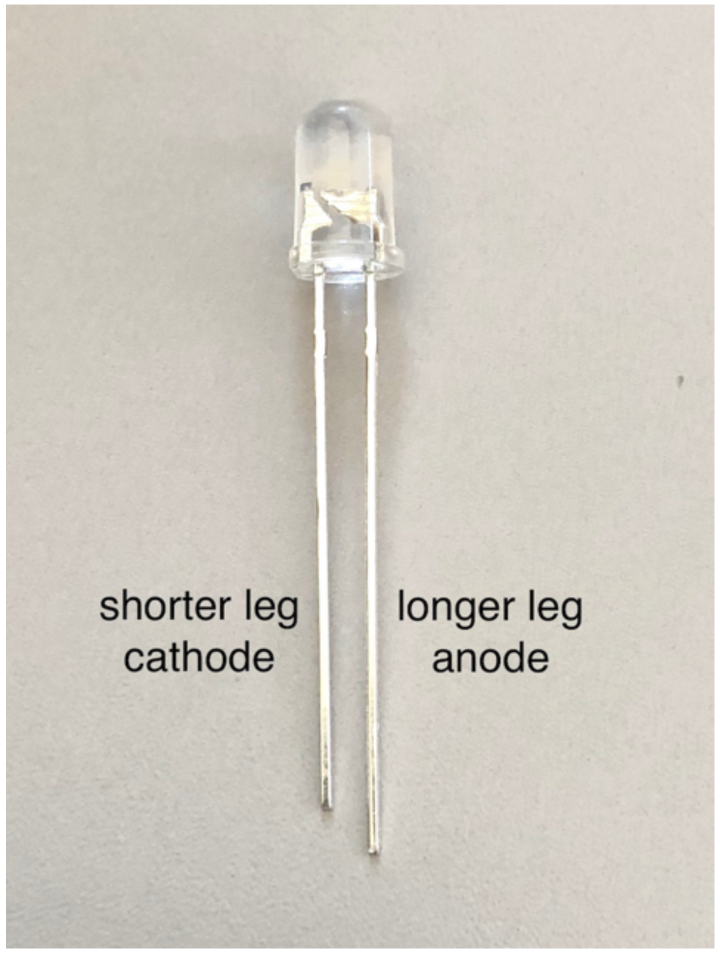
Blink

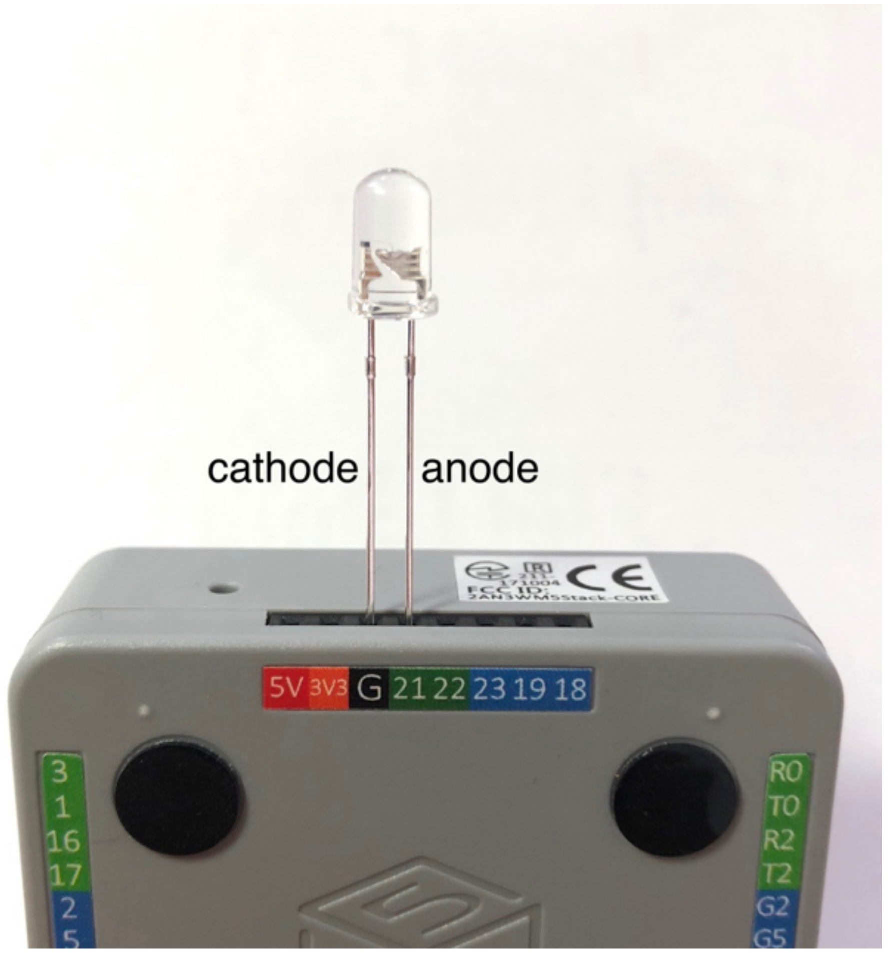
Repeatedly blinks an LED

```
#include <Arduino.h>

// This example uses GPIO 21 for an LED because the pin is in a
// physically convenient position in M5Stack Core. However, the
// pin is usually used as the SDA of I2C in ESP32. So you should
// make sure that nothing is connected to the I2C (Grove) port
// before running this code.
#define LED_PIN 21

void setup() {
    pinMode(LED_PIN, OUTPUT); // configures LED_PIN as an output
}

void loop() {
    digitalWrite(LED_PIN, HIGH); // turns the LED on
    delay(500);                  // waits for 500ms
    digitalWrite(LED_PIN, LOW);  // turns the LED off
    delay(500);                  // waits for 500ms
}
```

How to connect an LED

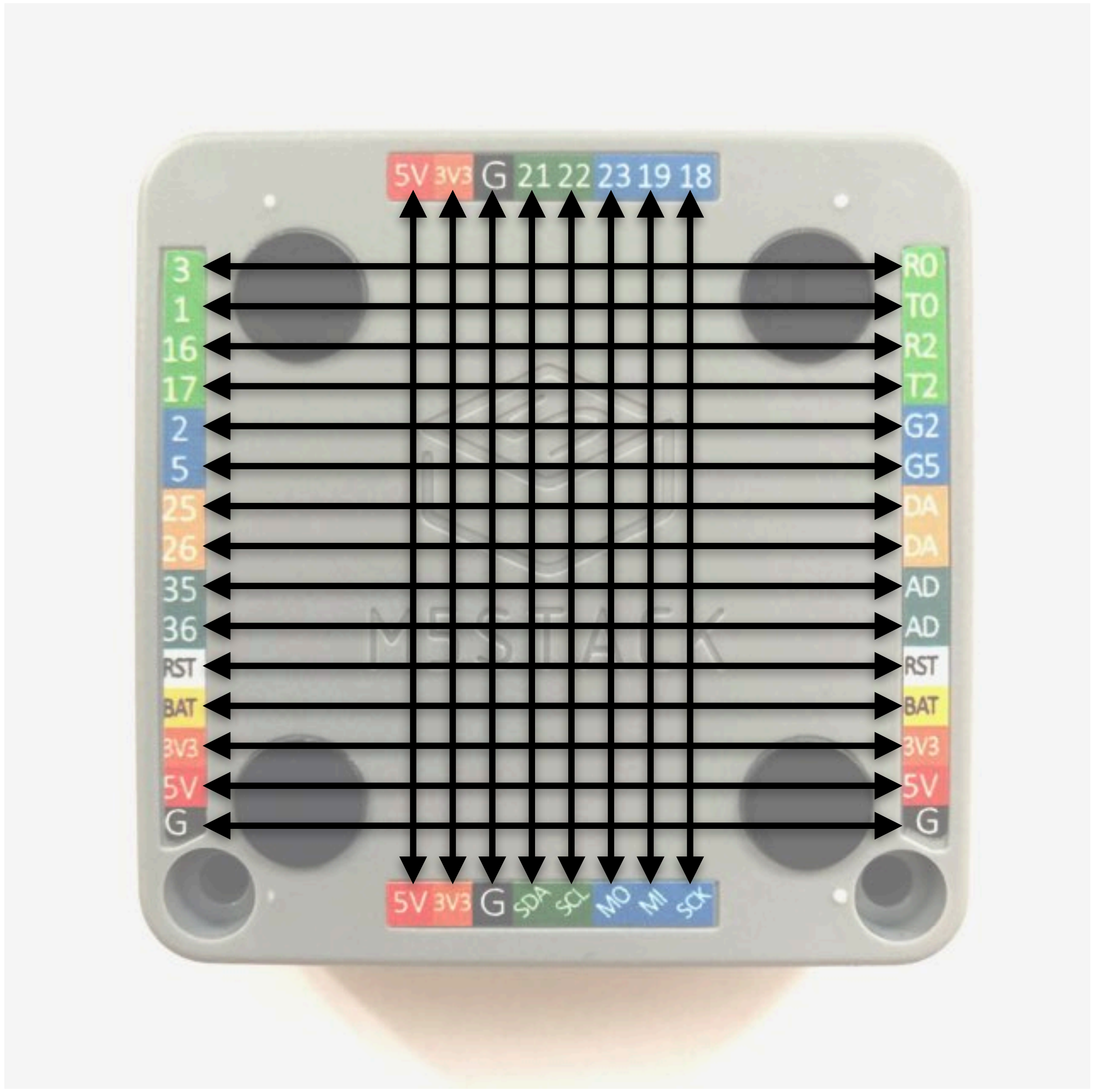
If you do this, make sure that nothing is connected to the Grove port.





LEDs in the kit have built-in resistors. So you can directly connect them to the GPIO pins. In this example, shorter and longer legs of an LED are respectively inserted into the ground (G) and GPIO 21 holes.

GPIO Pins



Each corresponding pair of the pins on opposite sides is connected.
ex) GPIO21 = SDA

Pins in the Grove port are connected to SDA, SCL, 5V and G.

WifiScan

```
#include <Arduino.h>
#include <WiFi.h>

void setup() {
    Serial.begin(115200);
    WiFi.mode(WIFI_STA);
    WiFi.disconnect();
}

void loop() {
    Serial.println();
    Serial.print("Scanning ... ");
    int n = WiFi.scanNetworks();
    Serial.println("done.");
    for (int i = 0; i < n; i++) {
        Serial.print(WiFi.BSSIDstr(i));
        Serial.print(" (");
        Serial.print(WiFi.RSSI(i));
        Serial.print(") ");
        Serial.println(WiFi.SSID(i));
    }
    delay(5000);
}
```

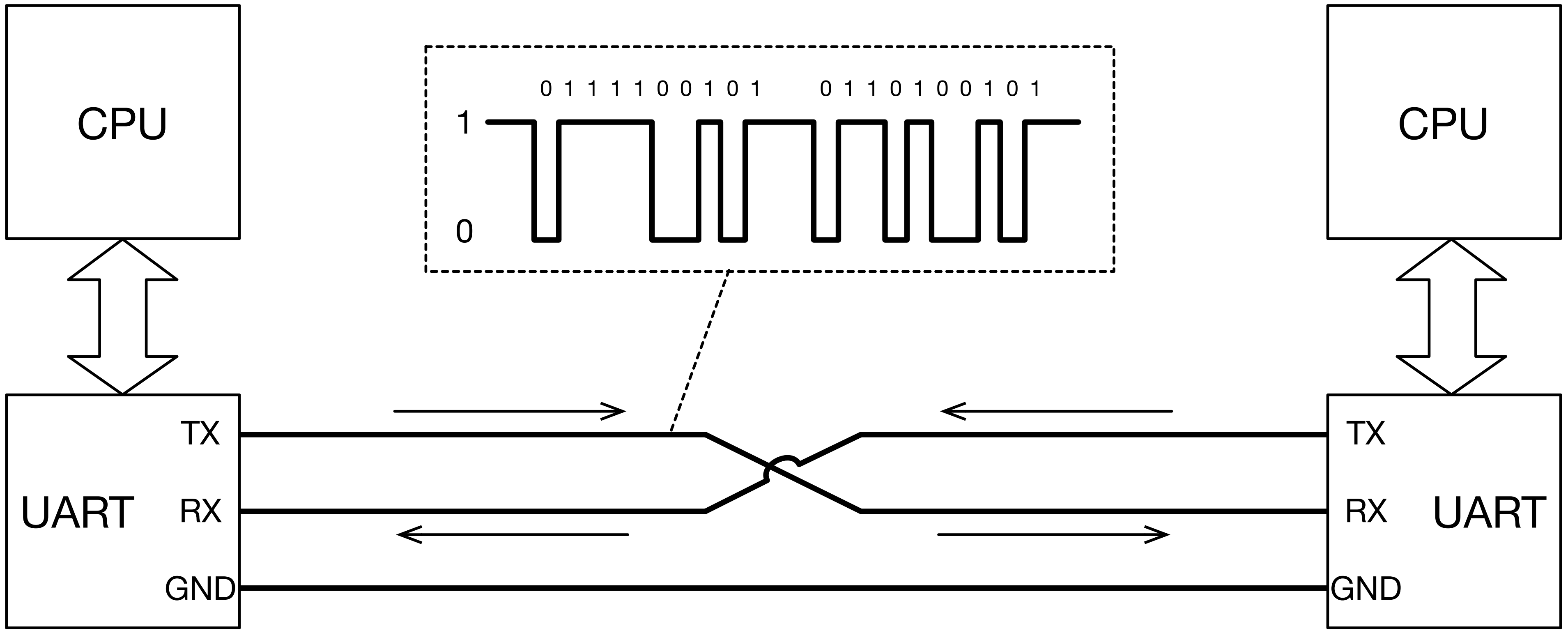
USB-Serial Port

- A virtual serial (UART) port is available via USB
- When you connect your PC to M5Stack, you should find a serial communication port in your PC
  - Mac) /dev/cu.SLAB_USBtoUART
  - Linux) /dev/ttyUSB0
  - Windows) COM?

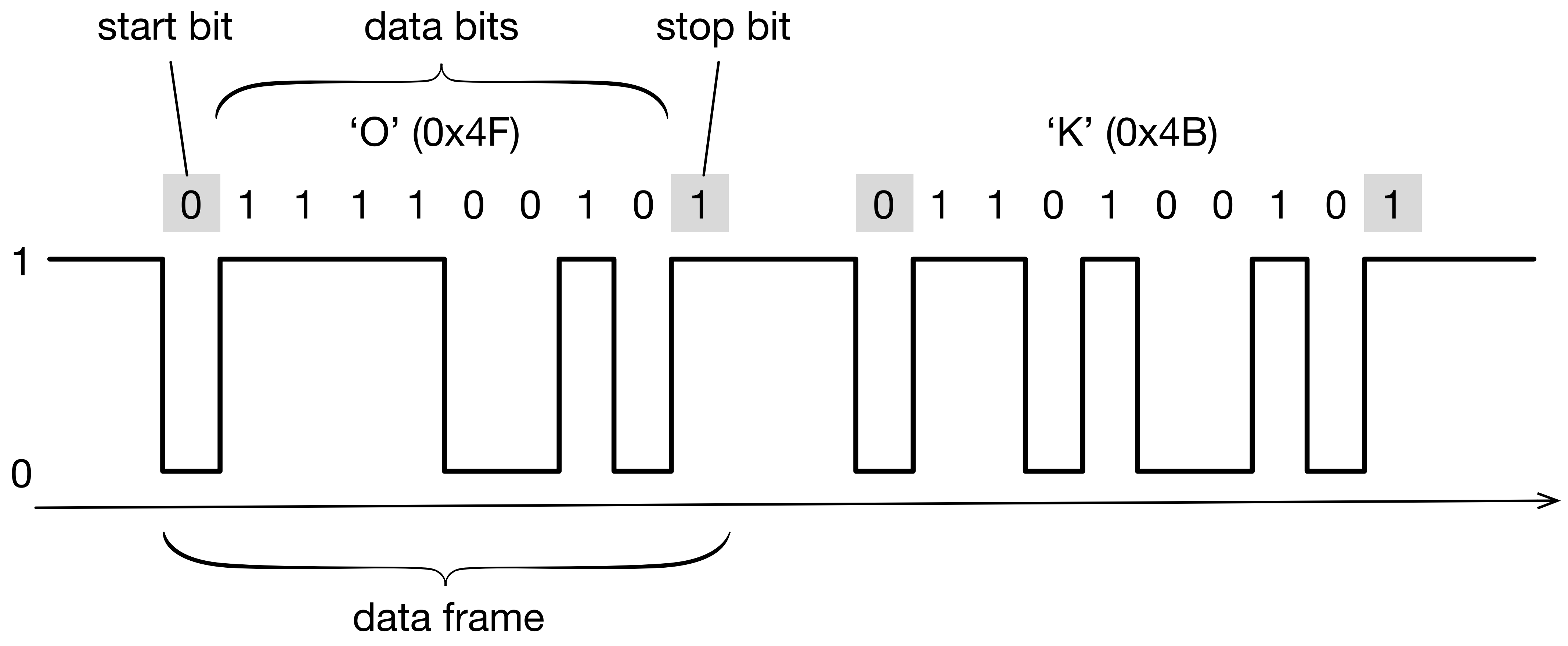
- Serial : default (USB-serial) serial port
  - begin(speed)
  - print, println, printf

UART

- Universal Asynchronous Receiver/Transmitter
- A (legacy) serial communication interface
- Binary bits correspond to voltage levels



UART Signal Format



HelloM5

A Greeting Application

```
#include <M5Stack.h>

const String text = "Hello M5";
const uint8_t text_font = 4;
int mx, my;

void setup() {
    M5.begin();
    M5.Lcd.setTextFont(text_font);
    M5.Lcd.setTextColor(TFT_YELLOW);
    mx = M5.Lcd.width() - M5.Lcd.textWidth(text);
    my = M5.Lcd.height() - M5.Lcd.fontHeight(text_font);
}

void loop() {
    M5.Lcd.fillScreen(TFT_BLACK);
    M5.Lcd.drawString(text, random(mx), random(my));
    delay(1000);
    M5.update();
}
```

Writing Code for M5Stack

- Include M5Stack.h instead of Arduino.h
    - Arduino.h is included in M5Stack.h
- M5 : the object that manages M5Stack H/W
    - M5.Lcd
    - M5.BtnA, M5.BtnB, M5.BtnC
    - M5.Speaker
    - M5.begin()
        - hardware initialization
    - M5.update()
        - status update for some devices (buttons, speaker)