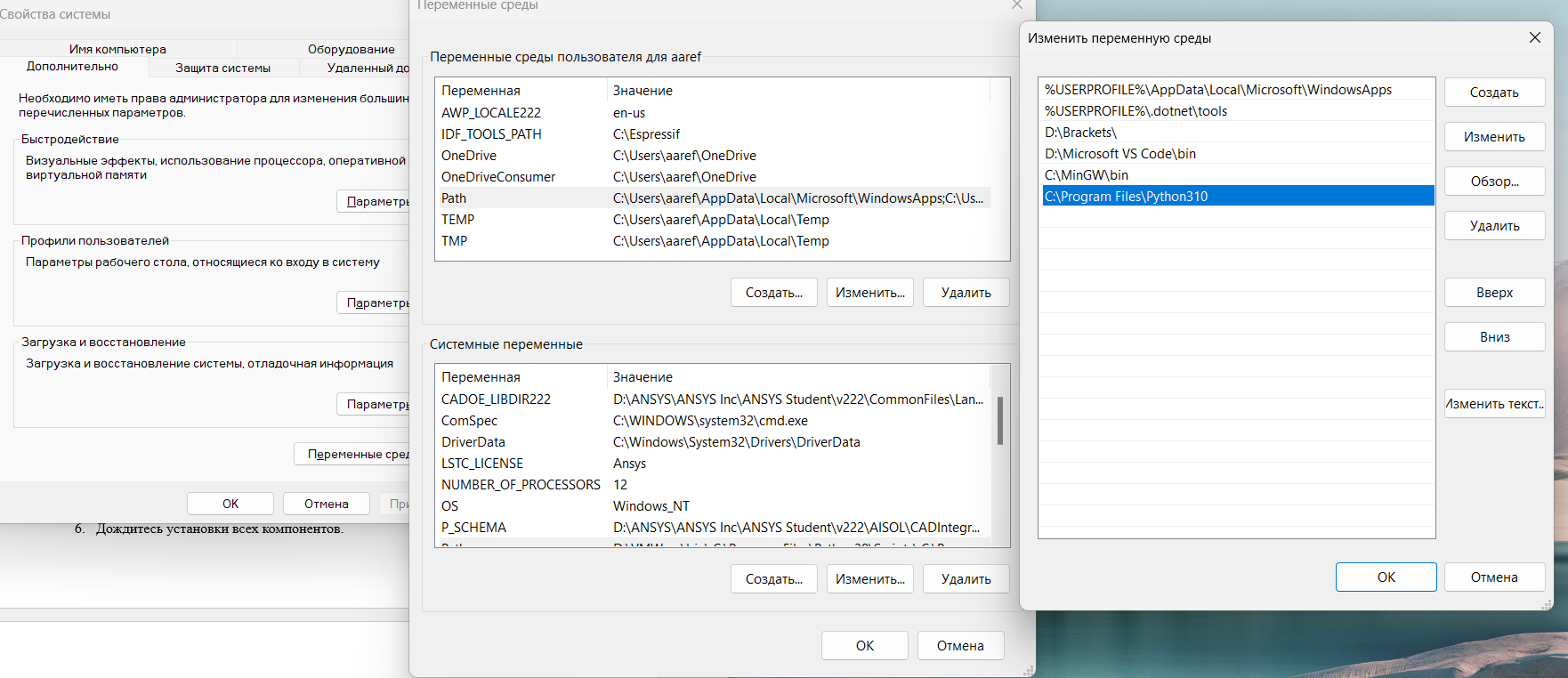
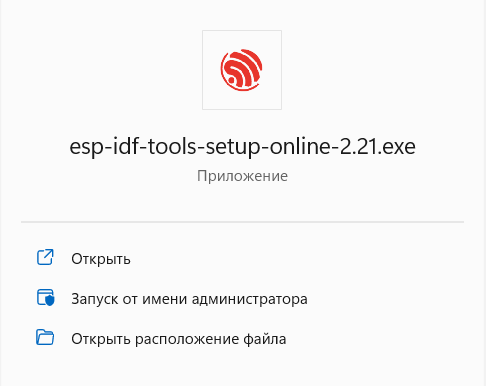
Инструкция для работы с МК ESP32 DevKitC V4 Wroom 32U

**Задача 1. Научиться работать с ESP32, установить среду разработки, научиться запускать дебаг (при наличии)**

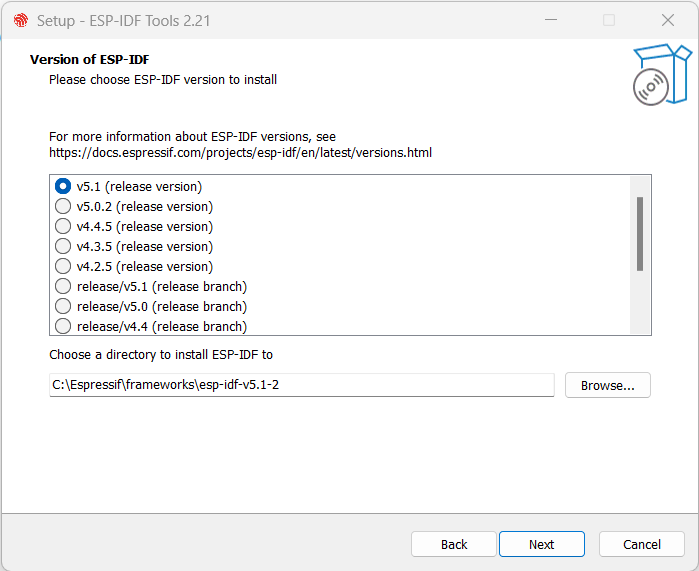
1. Для того, чтобы скачать инсталлер для работы с ESP-IDF, перейдите по [ссы­­­­­лке](https://dl.espressif.com/dl/esp-idf/?idf=4.4) и загрузите Universal Online Installer. Инструкция от компании [здесь](https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/windows-setup.html).
2. В переменные среды для пользователя и системные в переменные Path необходимо добавить путь до директории с Python 3.11. Если Python не установлен, то необходимо скачать его с официального сайта и установить.



1. Скачайте и установите последнюю версию CMake, ninja и MinGW.
2. В меню «Пуск» введите ESP и нажмите открыть

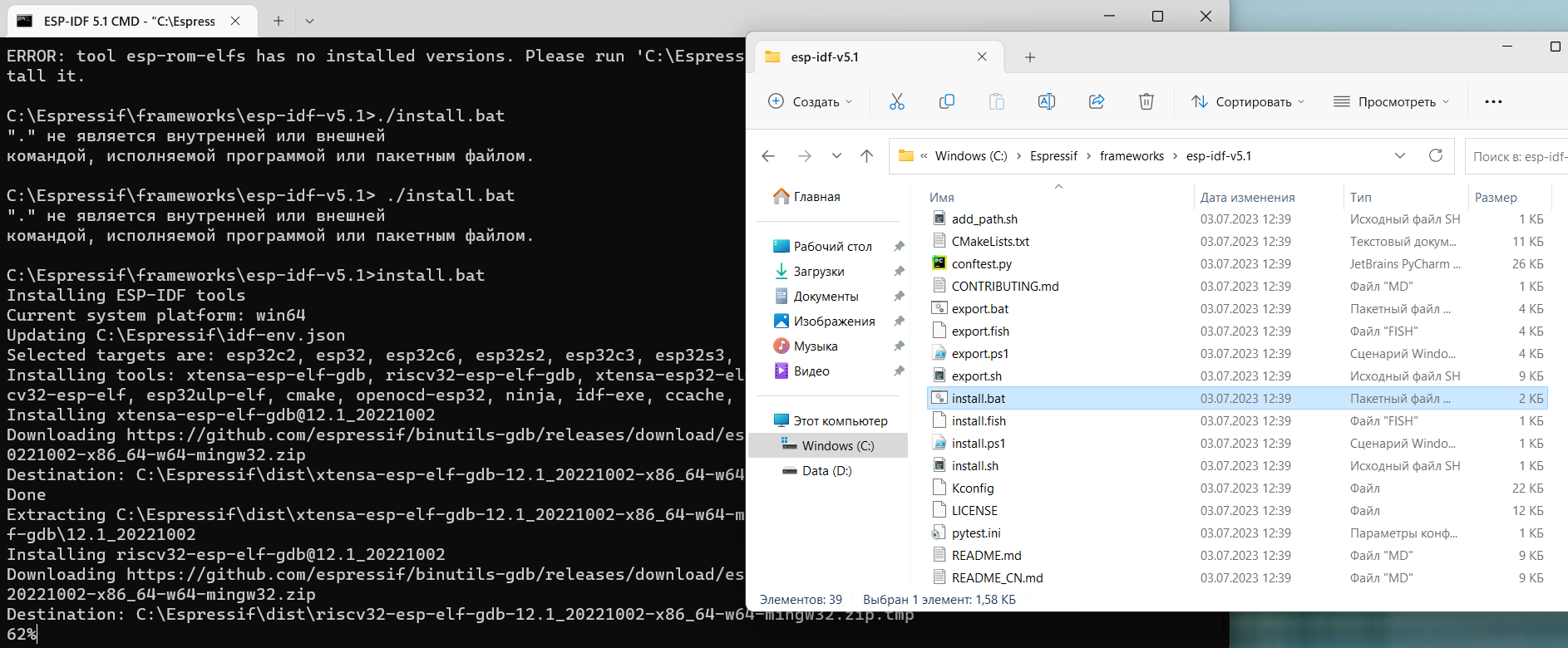


1. Откройте скачанный файл и нажимайте «Далее (Next)» пока не увидите данное окно.

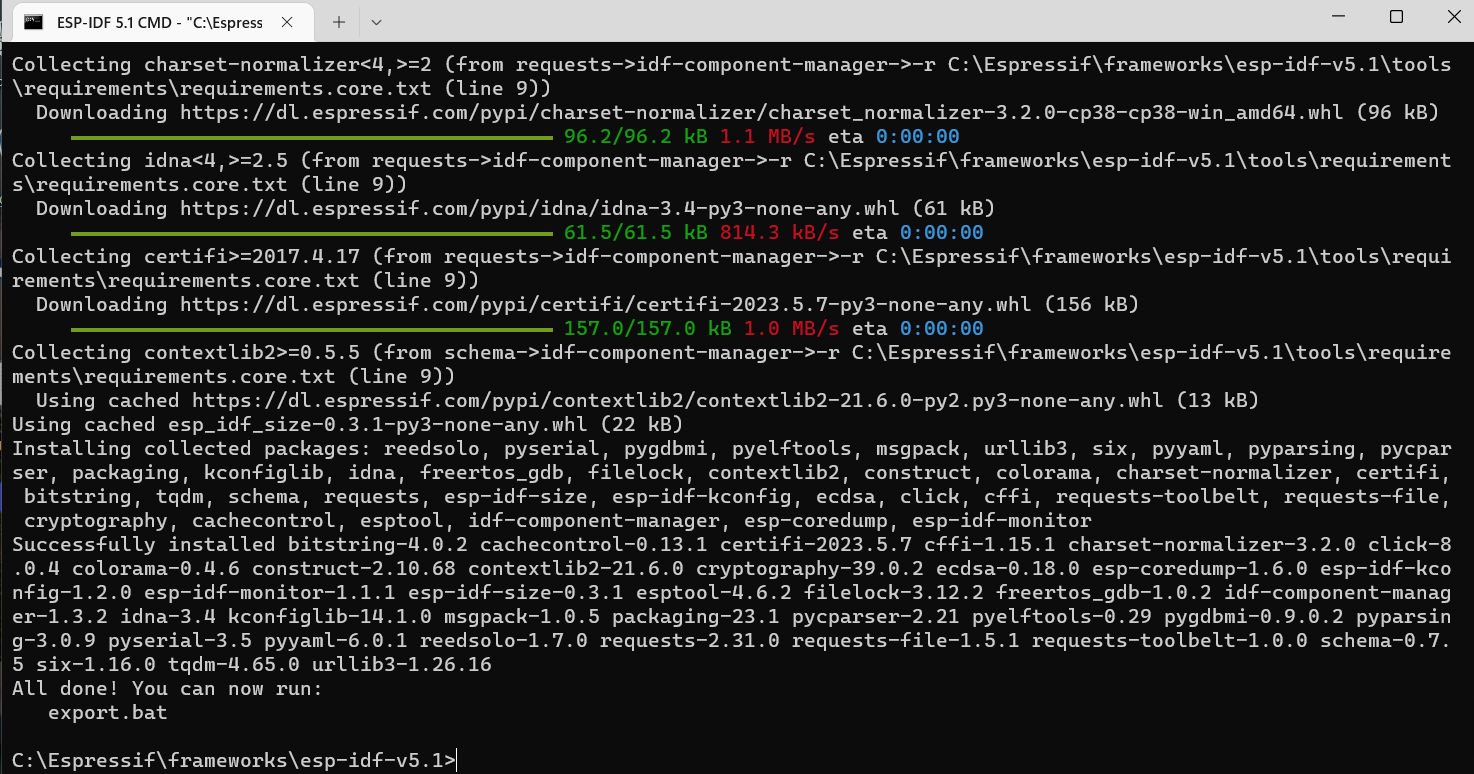


Выберите версию v5.1 (release version), укажите путь для вашего проекта и дождитесь загрузки. Выбираем Full Installation.

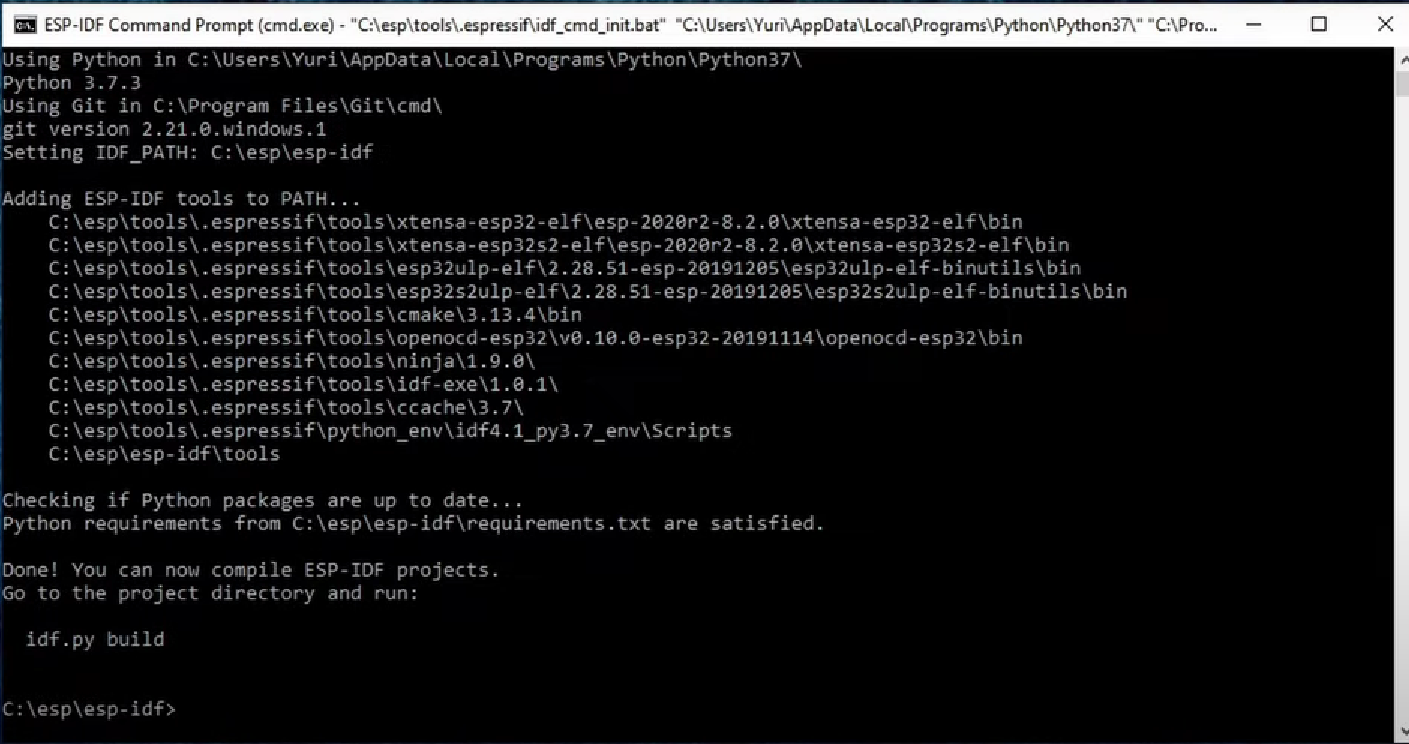
Из терминала перейдите в директорию Espressif/frameworks/esp-idf-v5.1 и запустите install.bat и дождитесь установки.



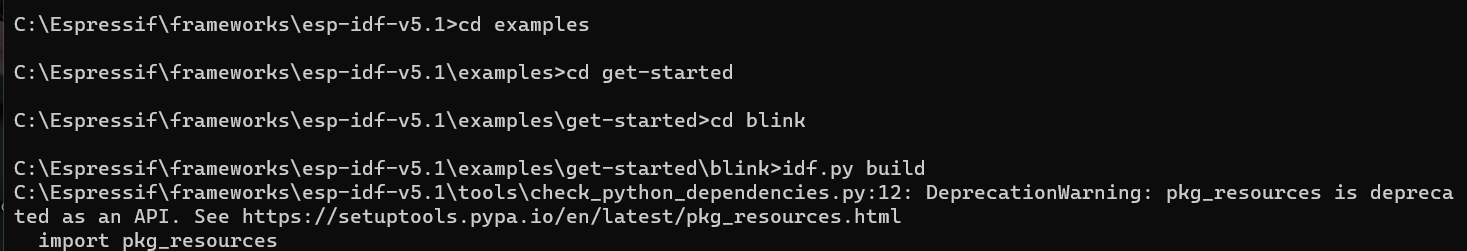
Далее запустите в директории export.bat.



1. Дождитесь установки всех компонентов.



1. Подключите МК к ПК с помощью кабеля USB-A to Micro-USB.
2. Соберите схему со светодиодом и резистором, подключив его к пину 21.
3. Зайдите в путь проекта командой cd PATH/TO/PROJECT, далее в папку Espressif/frameworks/esp-idf-v5.1/examples/get-started/blink/main и откройте файл blink\_example\_main.c и в коде измените 22 строку, где написано #define BLINK\_GPIO и поставьте значение 21.
4. Введите в консоли команду cd.. и введите команду idf.py build.

****

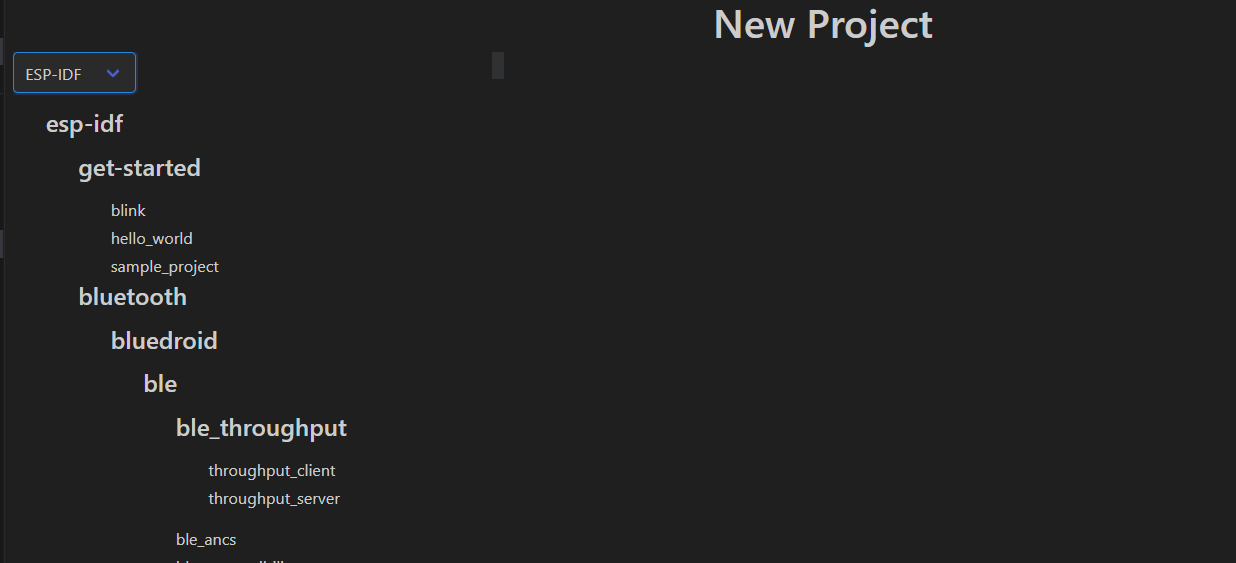
1. После завершения загрузки введите idf.py -p (PORT) flash , где PORT – COM порт для подключения микроконтроллера. Когда в консоли появится надпись Connection… необходимо зажать кнопку BOOT на плате.
2. Для отображения информации о МК введите idf.py -p (PORT) monitor.
3. Как итог: светодиод моргает с периодичностью 1 сек.

Дебаг возможен только сторонними аппаратными средствами.

ЛИБО:

Удобнее будет воспользоваться VS Code.

Переходим во вкладку Extensions и скачиваем Espressif IDF. После установки нажимаем в верхнем меню View – Command Pallete – New Project.

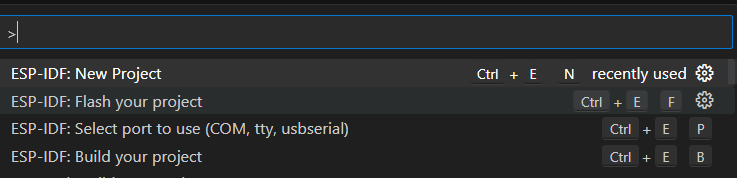


В качестве примера выберите blink. В окне снизу справа нажмите OK.

После создания проекта, написания кода – нажимаем на шестеренку снизу ESP-IDF SDK Configuration Editor (menuconfig).



После загрузки нажимаем в верху экрана View – Command Palette и последовательно выполняем Select port to use и кликаем на подключенный COMx порт с ESP32, потом выбираем Build your project и после всего выбираем Flash your project. Все должно сработать без ошибок.



**Задача 2. Научиться пользоваться портами GPIO (например кнопкой и светодиодом). (Архив 2\_1\_blink.rar и Архив 2\_2\_blink.rar)**

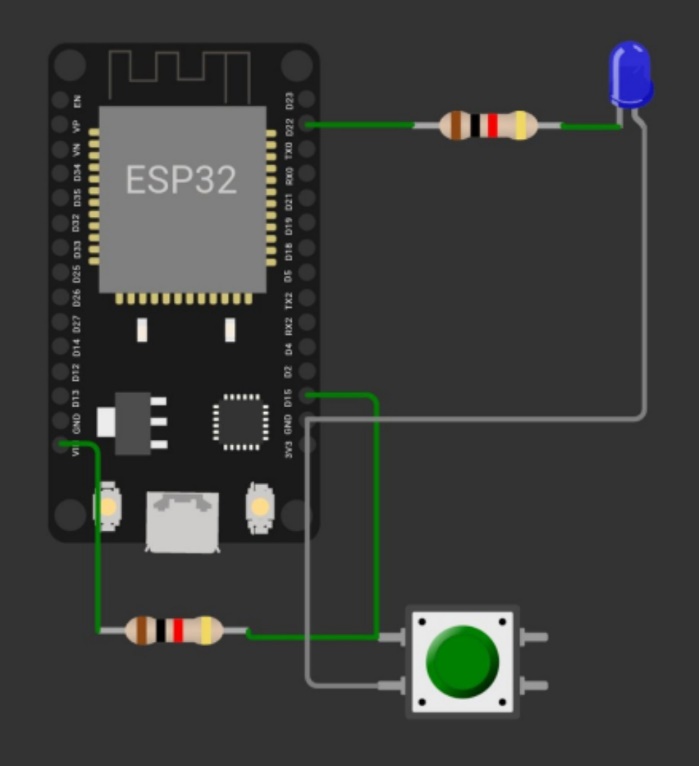
Светодиод отвечает за питание на плате, не настраиваемый.

На плате имеются две кнопки: EN и BOOT. EN отвечает за перезагрузку платы. BOOT необходима для прошивки платы.

В качестве примера мы собрали схему, состоящую из подтягивающего резистора на 3 кОм, резистор для светодиода 3 кОм и сам светодиод, тактовая кнопка на два выхода.

По схеме мы подключили к пину 22 резистор, последовательно подключили светодиод, вторая нога которого уходит на землю. Также к 16 пину подключили кнопку через подтягивающий резистор.

*Схема*:



*Код для МК*: (**Архив 2\_1\_blink.rar)**

В конечном итоге светодиод будет моргать с задержкой 0.25 секунд, а при нажатии кнопки задержка будет составлять 3 секунды, после чего задержка вернется в начальное значение.

Код: меняет задержку по нажатию на кнопку. Если кнопка не нажата, то задержка 0.25 с, если кнопка была нажата, то теперь задержка 3 секунды. **(Архив 2\_2\_blink.rar)**

**Задача 3. Научиться использовать прерывания. (Архив 3\_interrupt.rar)**

Код: каждые 3 секунды срабатывает прерывание и вызывается обработчик прерываний – функция timer\_isr\_callback.

**Задача 4. Научиться подключаться по Wi-Fi к смартфону, реализовать управление платой через смартфон (например, при нажатии кнопки на смартфоне – загорается светодиод, подключенный к плате, и наоборот).** (**архив 4\_1\_wifi.rar и архив 4\_2\_wifi.rar)**

Код **(архив 4\_1\_wifi.rar)** инициализирует Wi-Fi модуль на плате и переключает его в режим точки доступа. Название точки доступа – «ESP32», пароль «password».

Функция wifi-event-handler – обработчик нашей точки доступа, при подключении пользователя к ней будет выдаваться сообщение в логи о том, что подключен пользователь с определенным MAC-адресом, ему назначен идентификатор, а также будет загораться светодиод, подключенный через резистор к 22 пину.

Функция wifi\_init\_softap инициализирует точку доступа, устанавливает обработчик (в функции esp\_event\_handler\_instance\_register()). В структуре wifi\_config задается название, длина названия, пароль, максимальное количество подключений.

Код ниже позволяет управлять светодиодом со смартфона и наоборот. (**архив 4\_2\_wifi.rar)**

Создание проекта: для его работы необходимо воспользоваться для удобства расширением ESP-IDF Explorer в VS Code. Создаем проект с названием WiFi (например), в поле директории проекта выбираем путь до Espressif/frameworks/esp-idf-v5.1/examples/get-started, выбираем ESP32 module. Потом в поле расширения выбираем ESP-IDF и нажимаем Create project using template sample\_project. Создадим в проекте в main папке файл с названием «Kconfig.projbuild» и туда вставляем код:

menu "Example Configuration"

config ESP\_WIFI\_SSID

string "WiFi SSID"

default "myssid"

help

SSID (network name) for the example to connect to.

config ESP\_WIFI\_PASSWORD

string "WiFi Password"

default "mypassword"

help

WiFi password (WPA or WPA2) for the example to use.

config ESP\_MAXIMUM\_RETRY

int "Maximum retry"

default 5

help

Set the Maximum retry to avoid station reconnecting to the AP unlimited when the AP is really inexistent.

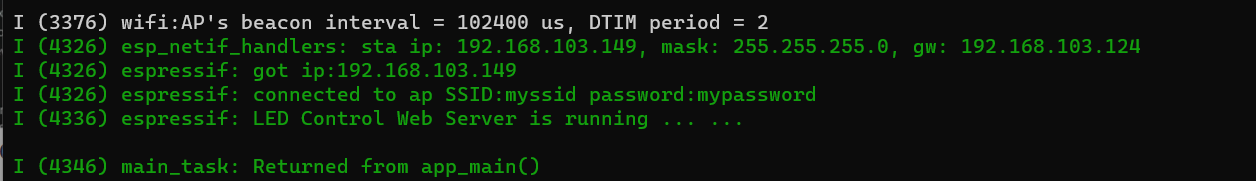
endmenu

В данном файле можно ввести необходимый SSID и пароль, к которой будет подключаться микроконтроллер.

После этого в файле main.c вставляем код и сохраняем проект

*Принцип работы:*

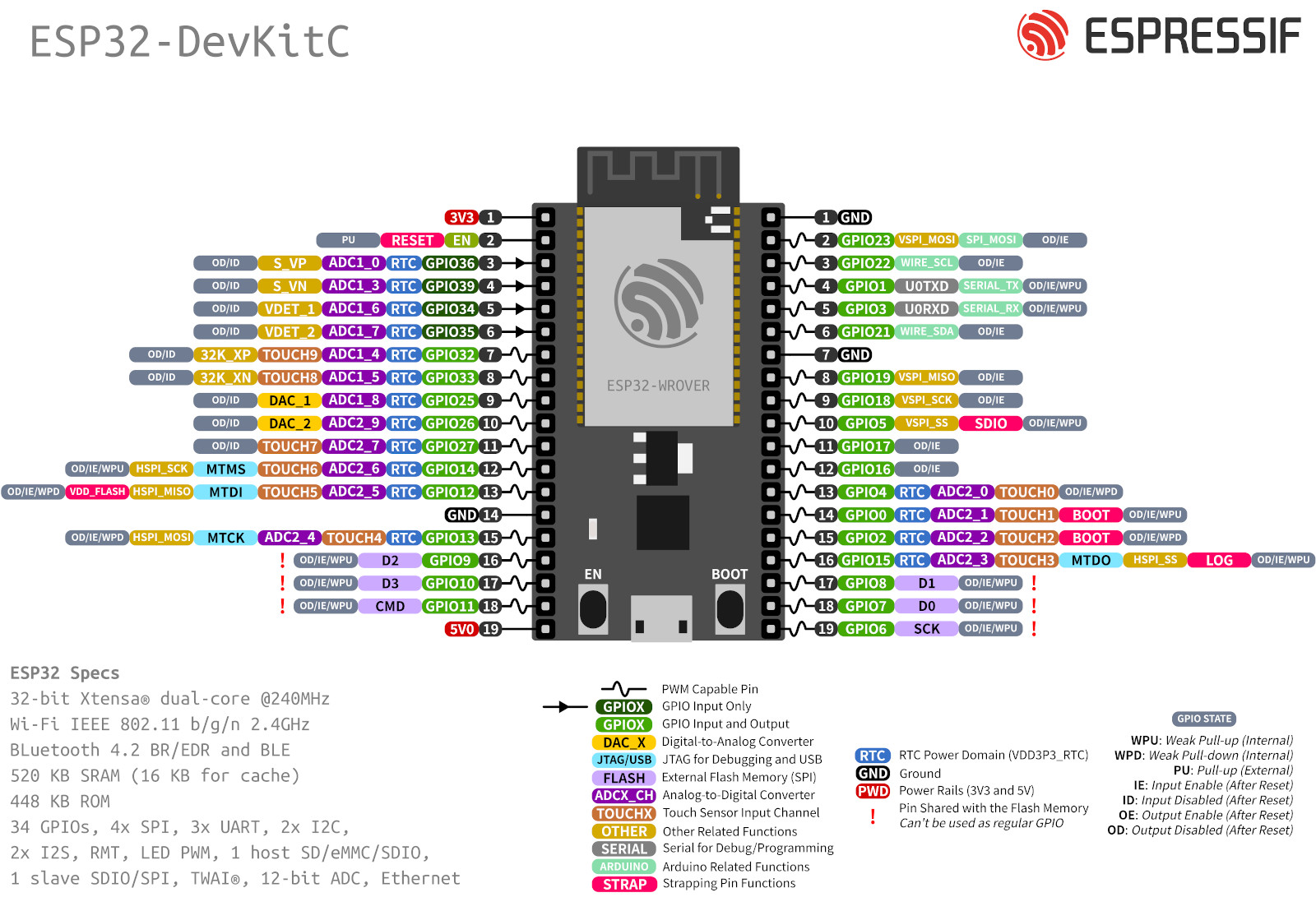
После того, как мы забилдили проект (idf.py build), зашили код (idf.py -p COMx flash) и запустили его (командой idf.py -p COMx monitor) микроконтроллер пытается подключаться к сети myssid с паролем mypassword (данные настройки можно сменить в файле как описано выше), в логах monitor’а будет выдан IP-адрес (на рисунке ниже это строка got ip: 192.168.103.149).



Теперь необходимо с устройства, которое является точкой доступа, перейти по адресу 192.168.103.149 (в данном примере), откроется сайт и при нажатии на кнопку ON будет загораться светодиод на 22 пине.

**Задача 5. Запустить АЦП на плате. (архив 5\_adc.rar)**

Микроконтроллер ESP32 DevKit V4 Wroom 32U обладает двумя 12-разрядными АЦП, всего 18 аналоговых каналов.

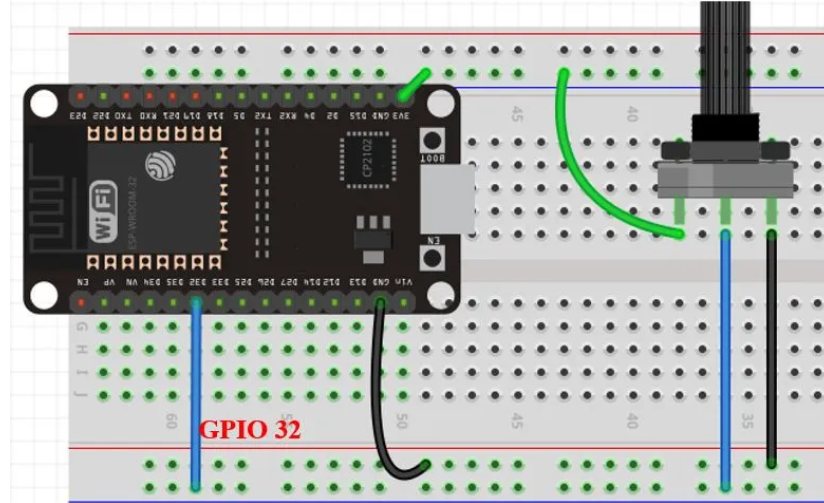


В качестве примера мы используем канал 1 и пин 4 (GPIO 32).

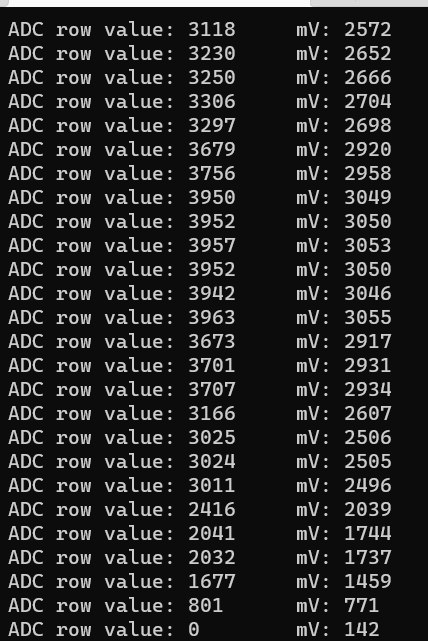
Код выводит в idf.py -p COMx monitor сырое значение АЦП и обработанное в мВ.

| **Затухание** | **Параметр затухания** | **Диапазон входного напряжения** |
| --- | --- | --- |
| 0dB | ADC\_ATTEN\_DB\_0 | ~100-950 mV |
| 2.5dB | ADC\_ATTEN\_DB\_2\_5 | ~100-1250mV |
| 6dB | ADC\_ATTEN\_DB\_6 | ~150-1750mV |
| 11dB | ADC\_ATTEN\_DB\_11 | ~150-2450mV |

Схема, необходимая для работы:



В мониторе получим такие строки, значение которых будет меняться в зависимости от потенциометра:



**Задача 6. Научиться выводить показания АЦП на смартфон. (Архив 6\_adc.rar)**

Для вывода показаний АЦП мы используем код для веб-сервера из задачи 4, немного его модифицировав. А также возьмем код и схему из задачи 5.

Как итог мы собираем проект (команда idf.py build), зашиваем (idf.py -p COMx flash) и запускаем монитор (idf.py -p COMx monitor). В логах монитора будет выдан устройству ip-адрес, на который мы заходим и видим данные с АЦП в реальном времени.

**Задача 7. Научиться использовать Bluetooth на ESP32. (архив 7\_bluetooth.rar)**

Код реализует управление портами GPIO через Bluetooth. Так, создается точка Bluetooth с именем ESP\_SPP\_ACCEPTOR, с помощью телефона подключаемся к ней и заходим в приложение Bluetooth Terminal, там подключаемся к нашему устройству и посылаем команды «on red led», «on green led», «off red led», «off green led». Для работы собираем схему из двух светодиодов и резисторов для них на 21 и 22 портах.