

30-Day FreeRTOS Course for ESP32 Using ESP-IDF

(Day 2)

“Setting Up ESP-IDF
with VS Code”





Table of Contents

Table of Contents

1. Overview
2. Why Use VS Code with ESP-IDF?
3. Install the ESP-IDF Extension
4. Create Your First ESP-IDF Project
5. Writing a Basic FreeRTOS Task
6. Building, Flashing, and Monitoring
7. Using menuconfig Inside VS Code
8. Summary
9. Challenge for Today



1. Overview

1. Overview

On Day 2, you'll learn how to:

- Set up ESP-IDF using the **official Visual Studio Code extension**
- Install the required toolchain, Python, Git, and ESP-IDF
- Create, configure, build, flash, and monitor your first ESP32 project
- Run your first FreeRTOS task inside Visual Studio Code

By the end of this lesson, you'll be ready to write, build, and debug FreeRTOS code on your ESP32 — all inside a full-featured graphical IDE.



2. Why Use VS Code with ESP- IDF?

2. Why Use VS Code with ESP-IDF?

While ESP-IDF works great from the command line, Visual Studio Code offers:



- ✓ Syntax highlighting and IntelliSense
- ✓ Built-in build/flash/monitor buttons
- ✓ Git integration
- ✓ IDF monitor with clickable logs
- ✓ Menuconfig GUI integration
- ✓ Task and peripheral debugging

The **official Espressif plugin** makes VS Code a powerful tool for FreeRTOS-based embedded development.




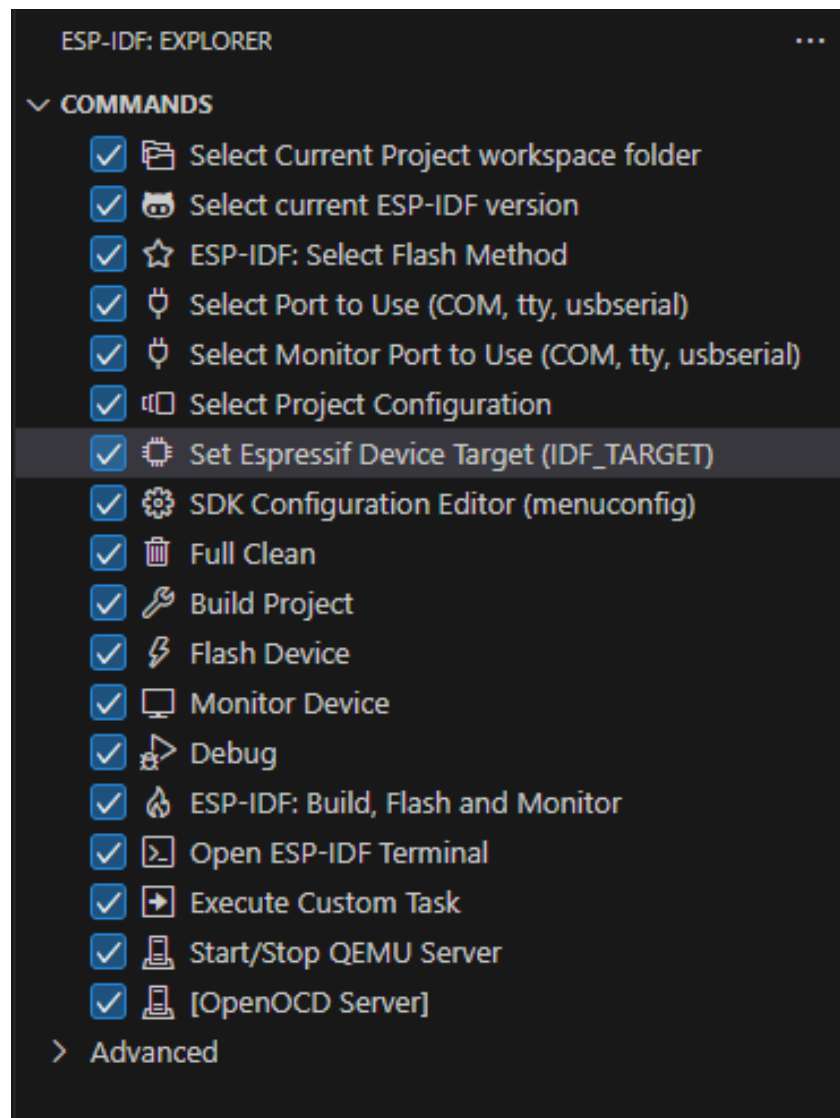
3. Install the ESP-IDF Extension

3. Install the ESP-IDF Extension

1. Download and install [Visual Studio Code](#).
2. Install ESP-IDF system prerequisites for your operating system:
 - Prerequisites for [MacOS and Linux](#).
 - For Windows there is no additional prerequisites.
3. In Visual Studio Code, Open the **Extensions** view by clicking on the Extension icon in the Activity Bar on the side of Visual Studio Code or the **View: Show Extensions** command (shortcut:   X or Ctrl+Shift+X).
4. Search for [ESP-IDF Extension](#).

3. Install the ESP-IDF Extension

5. Install the extension. After you install the extension, the Espressif icon  should appear in the VS Code Activity bar (left side set of icons). When you click the Espressif icon, you can see a list of basic commands provided by this extension.



3. Install the ESP-IDF Extension

6. From the command list, select **Configure ESP-IDF Extension** or press **F1** and type **Configure ESP-IDF Extension**. After, choose the **ESP-IDF: Configure ESP-IDF Extension** option.

NOTE: For versions of ESP-IDF < 5.0, spaces are not supported inside configured paths.



ESPRESSIF

ESP-IDF Extension for Visual Studio Code

Select download server:

Github



☐ Show all ESP-IDF tags

Select ESP-IDF version:

v5.4.2 (release version)



3. Install the ESP-IDF Extension



ESPRESSIF

ESP-IDF Extension for Visual Studio Code

Select download server:

Github



☐ Show all ESP-IDF tags

Select ESP-IDF version:

v5.4.2 (release version)



Enter ESP-IDF container directory:

C:\Users\YankiMainDesk\esp

\v5.4.2\esp-idf



Enter ESP-IDF Tools directory (IDF_TOOLS_PATH):

C:\Users\YankiMainDesk\.espressif



Install

3. Install the ESP-IDF Extension

7. Choose Express and select the download server:
 - **Espressif:** Faster speed in China using Espressif download servers links.
 - **Github:** Using github releases links.
8. Pick an ESP-IDF version to download or the `Find ESP-IDF in your system` option to search for existing ESP-IDF directory.
9. Choose the location for ESP-IDF Tools (also known as `IDF_TOOLS_PATH`) which is `$HOME\.espressif` on MacOS/Linux and `%USERPROFILE%\.espressif` on Windows by default.

3. Install the ESP-IDF Extension

10. If your operating system is MacOS/Linux, choose the system Python executable to create ESP-IDF virtual environment inside ESP-IDF Tools and install ESP-IDF Python package there.

NOTE: Windows users don't need to select a Python executable since it is going to be installed by this setup.

11. Make sure that `IDF_TOOLS_PATH` doesn't have any spaces to avoid any build issues. Also make sure that `IDF_TOOLS_PATH` is not the same directory as `IDF_PATH`.

3. Install the ESP-IDF Extension

12. You will see a page showing the setup progress status, including ESP-IDF download progress, ESP-IDF Tools download and install progress as well as the creation of a Python virtual environment.
13. If everything is installed correctly, you will see a message that all settings have been configured. You can start using the extension.



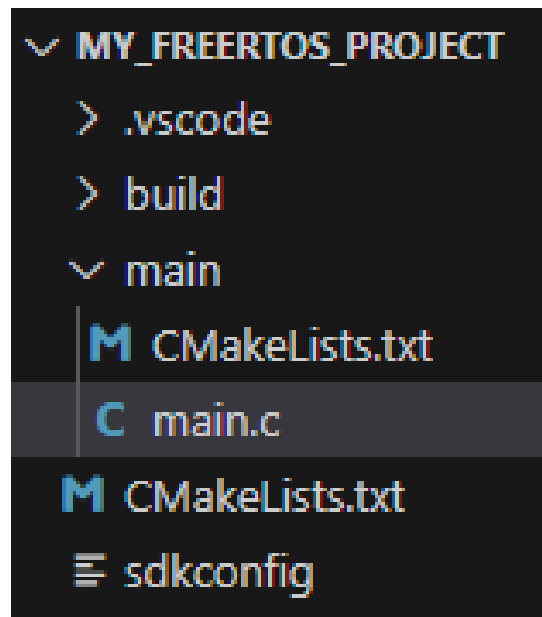
4. Create Your First ESP-IDF Project

4. Create Your First ESP-IDF Project

Once the setup is done:

1. Press Ctrl+Shift+P
2. Run: `ESP-IDF: Create project using template`
3. Select a folder and enter a project name, e.g., `my_freertos_project`
4. Choose the **template-app** template (we'll replace the code shortly)
5. Open the newly created folder in VS Code

Your project folder will look like this:





5. Writing a Basic FreeRTOS Task

5. Writing a Basic FreeRTOS Task

Replace the contents of `main/main.c` with:

```
1  #include <stdio.h>
2  #include "freertos/FreeRTOS.h"
3  #include "freertos/task.h"
4
5  void hello_task(void *pvParameter) {
6      while (1) {
7          printf("Day 2: Hello from FreeRTOS running inside Visual \
8              Studio Code!\n");
9          vTaskDelay(pdMS_TO_TICKS(1000));
10     }
11 }
12
13 void app_main(void) {
14     xTaskCreate(hello_task, "hello_task", 2048, NULL, 5, NULL);
15 }
```

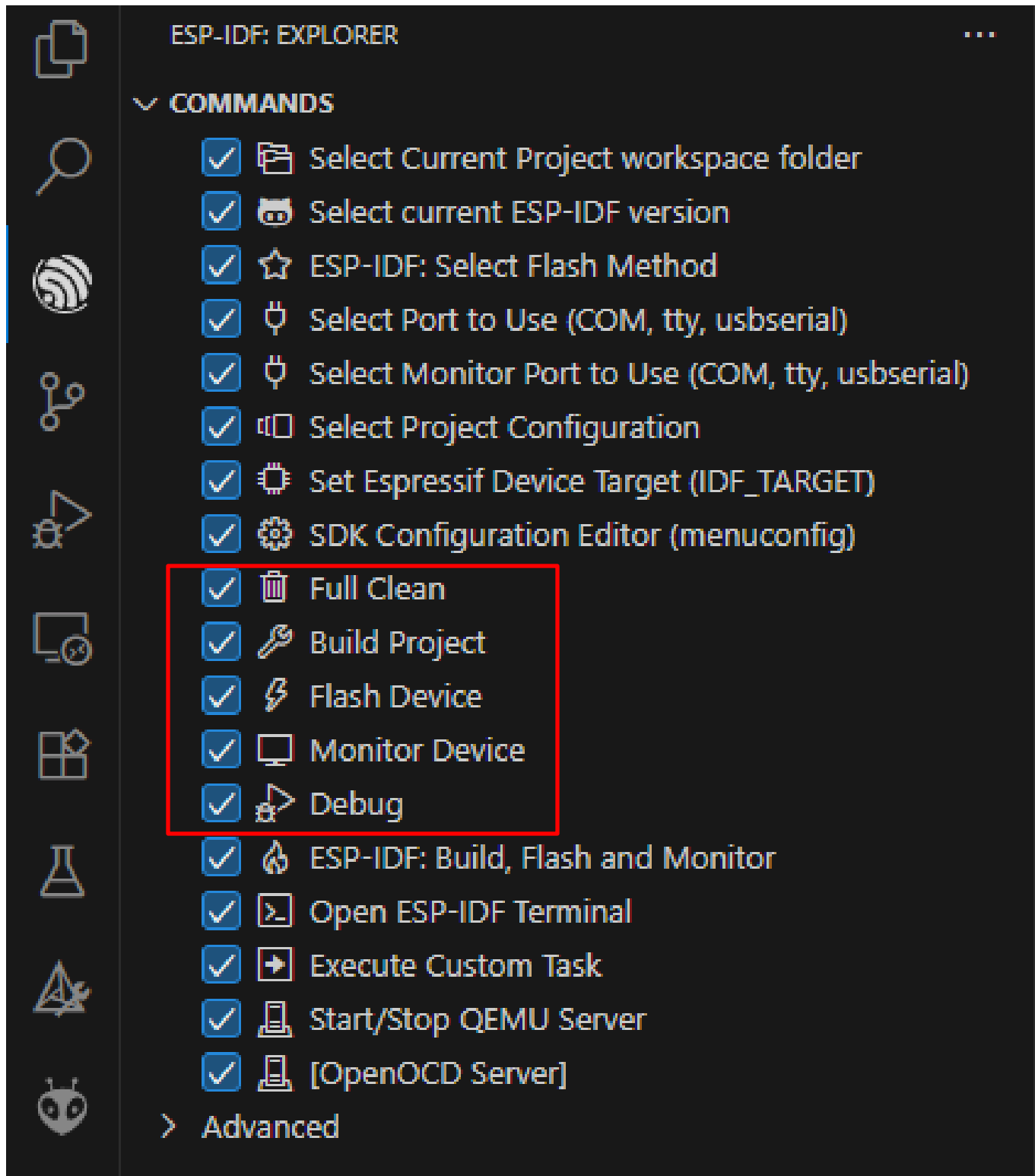
This is your first FreeRTOS task created.



6. Building, Flashing, and Monitoring

6. Building, Flashing, and Monitoring

Use the ESP-IDF plugin commands to build, Flash, and monitor your project.



6. Building, Flashing, and Monitoring



Build:

- Click the “**Build Project**” option on the bottom status bar, or
- Press `Ctrl+Shift+P` → `ESP-IDF: Build your project`



Flash:

- Connect your ESP32 board via USB
- Click “**Flash Device**”, or
- Press `Ctrl+Shift+P` → `ESP-IDF: Flash (UART) your project`



Monitor:

- Click “**Monitor Device**”, or
- Run `Ctrl+Shift+P` → `ESP-IDF: Monitor your device`

6. Building, Flashing, and Monitoring

Flash:

- Connect your ESP32 board via USB
- Click “**Flash Device**”, or
- Press `Ctrl+Shift+P` → `ESP-IDF: Flash (UART)`
your project

Monitor:

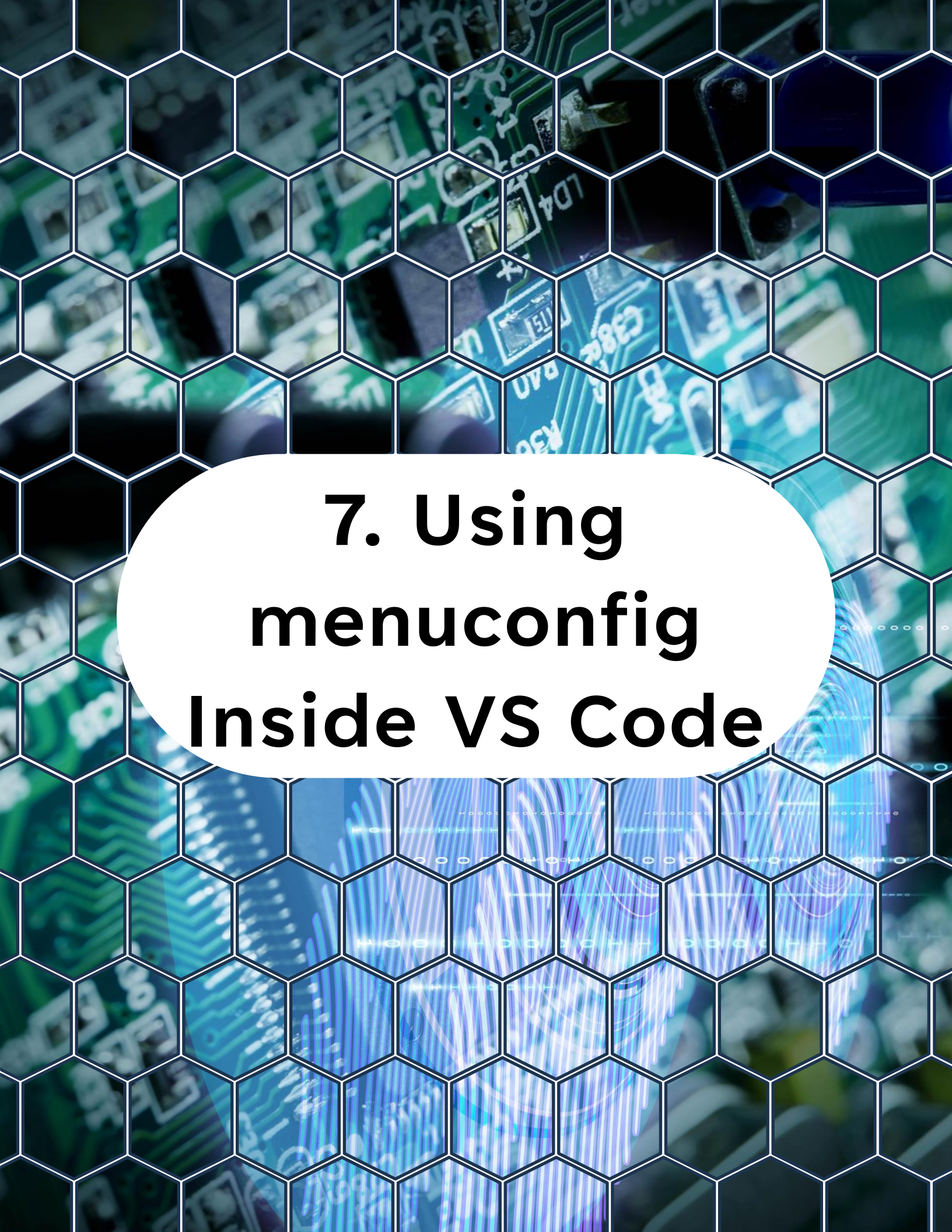
- Click “**Monitor Device**”, or
- Run `Ctrl+Shift+P` → `ESP-IDF: Monitor your device`

You should see:

Day 2: Hello from FreeRTOS running inside

Visual Studio Code!

repeated every second.



7. Using menuconfig Inside VS Code

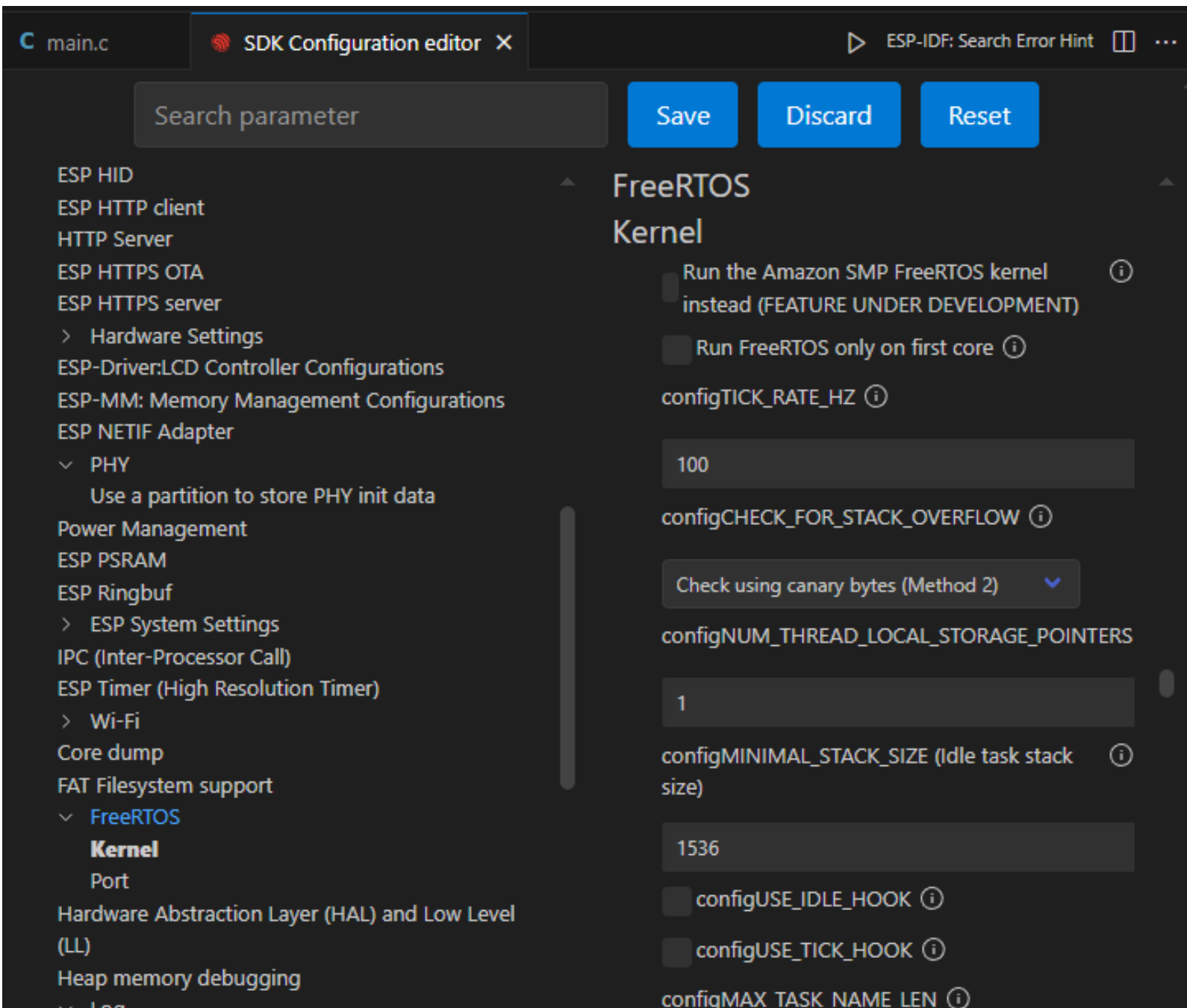
7. Using menuconfig Inside VS Code

Run:

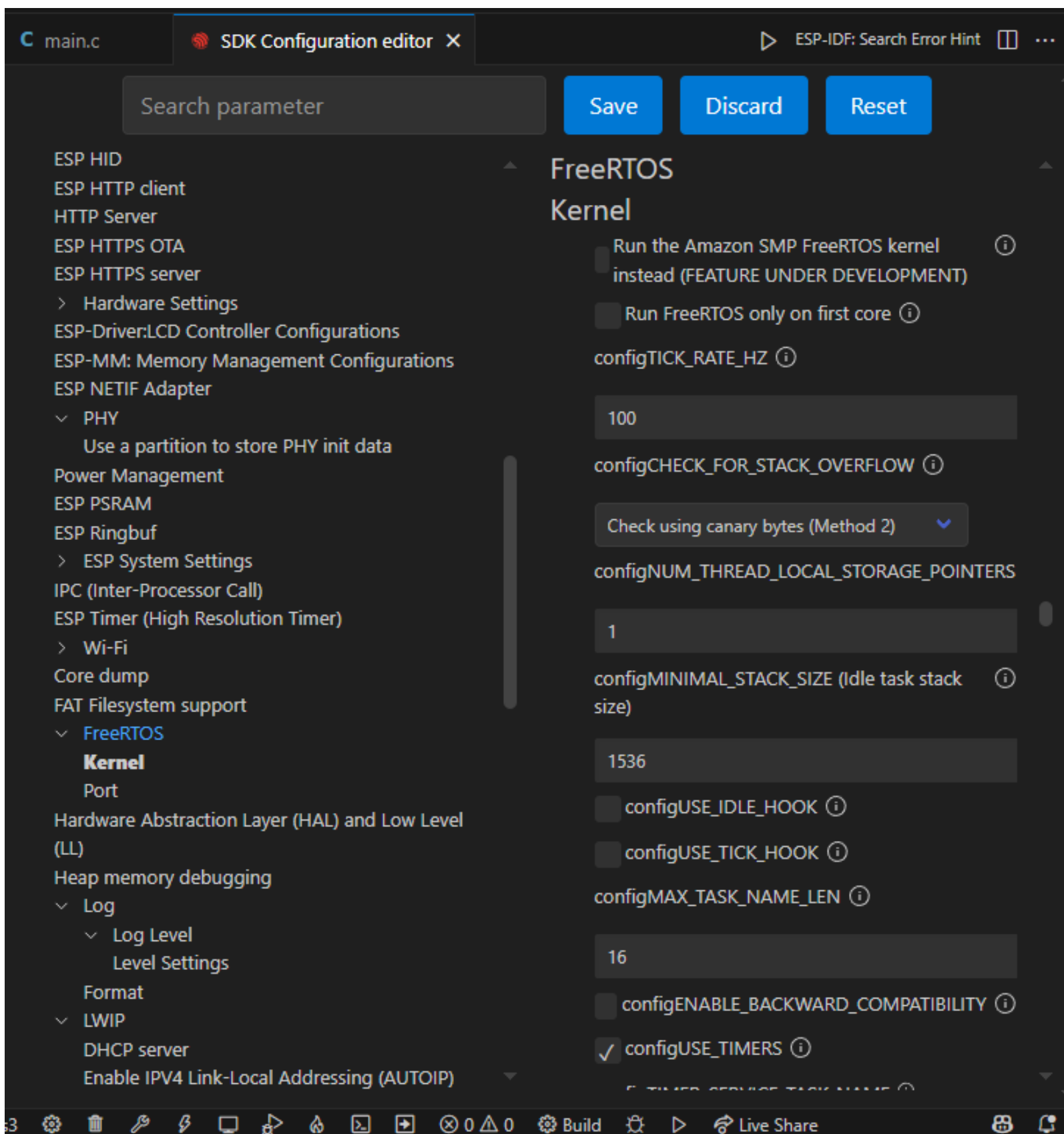
Ctrl+Shift+P → ESP-IDF: Launch Menuconfig

Navigate to:

Component config → FreeRTOS



7. Using menuconfig Inside VS Code



7. Using menuconfig Inside VS Code

From here you can:

- Change tick frequency
- Enable stack overflow detection
- Select memory allocator
- Tune task priority behavior
- Enable SMP features

We'll explore these options in future lessons.



8. Summary

8. Summary

Today, you successfully:

- Installed the official ESP-IDF plugin for Visual Studio Code
- Set up the complete toolchain and environment using the wizard
- Created your first ESP-IDF project
- Wrote and ran a FreeRTOS task
- Monitored output directly from VS Code

You're now equipped with a modern RTOS development environment!



9. Challenge for Today

9. Challenge for Today

Modify the example task to:

- Print a message every 2 seconds
- Add a counter to track how many times the message was printed
- Display the output in the terminal window

Example:

[2s] Hello from task (count = 5)