Zephyr Training – Community Edition

Day1

Prerequisites

Make sure that host tools and the extensions are installed, if not, follow the instructions on the last page. Or check out this video: https://www.youtube.com/watch?v=WtQWrpUYF1g

Instructions

These labs are designed as guided, hands-on exercises. Throughout the material, you will encounter "fill-in-the-blank" sections where you must replace the three dots "..." with the actual value, TODO markers, and partially implemented code. Your task is to complete these missing pieces by applying the concepts learned.

Note:

Time during the training may be limited, so you might not fully complete all labs during the session.

All hands-on exercises can be continued afterward at your own pace. Solutions are also provided for all labs

Table of Contents

Lab 1 – Hello World

The purpose of this lab is to get started with zephyr and the hands on labs environment, it is important to do theses steps as all the following labs are based on this one.

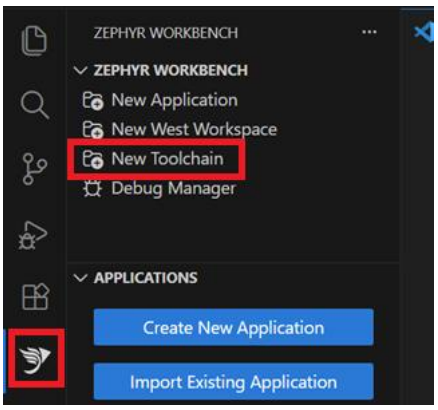
In case you have faced any problem, make sure that you have the right host requirements and installed the dependencies.

The lab consists of 7 steps, make sure that you follow them.

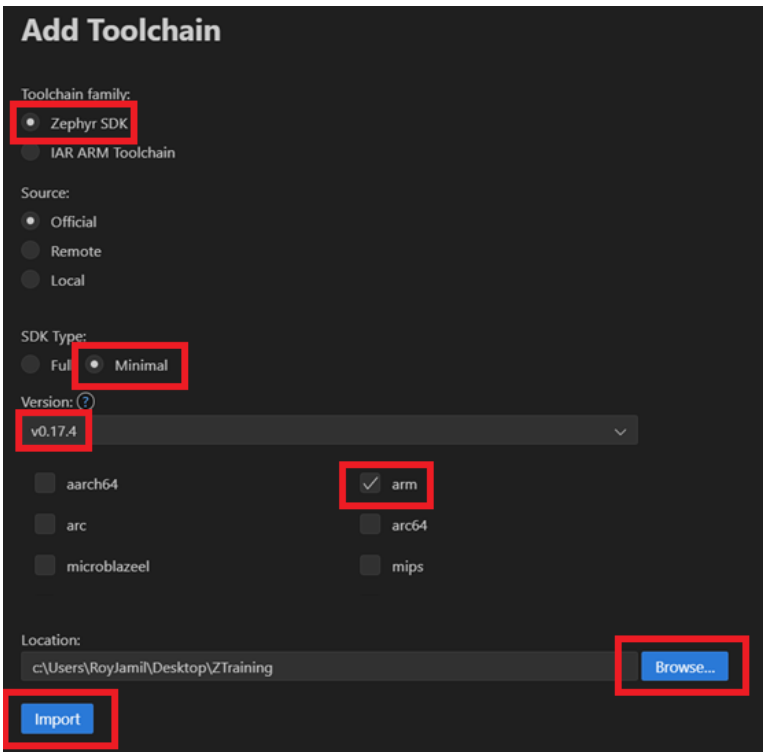
Note: in this lab, you may have already done some of the steps, but follow till the end, so that confirm that we all have a working environment

1- Import Zephyr SDK – (if not already imported)

- Open the workbench extension, then click on “New Toolchain”



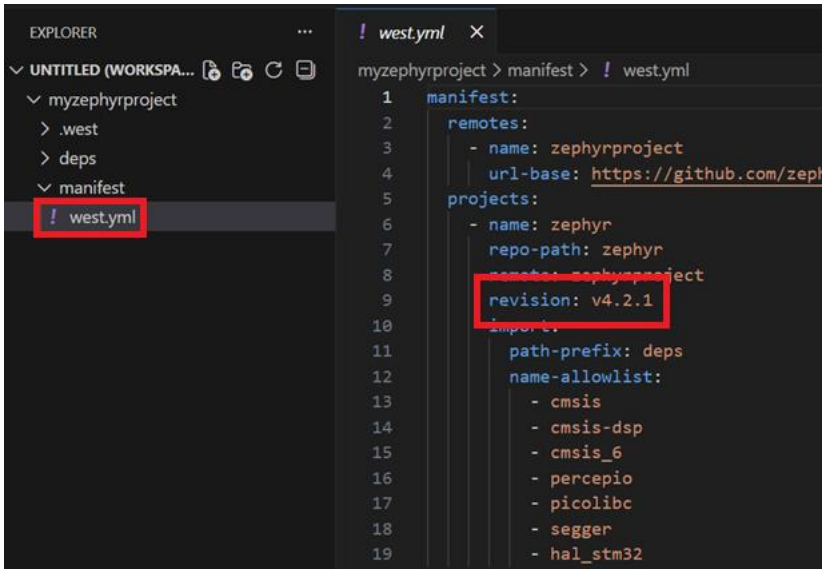
2- Import the Zephyr SDK v 0.17.4 for ARM (if not already imported)



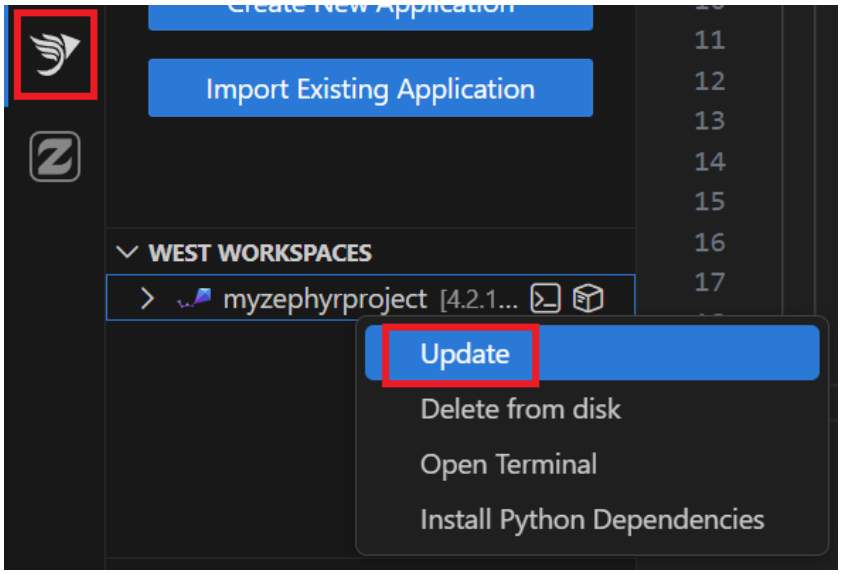
3- Import the West Workspace

3.1 - If you already have a working west workspace, make sure you are on the version 4.2.1, if not go to the section 3.2

- Open the west.yml in: <west workspace>/manifest/west.yml
    - Check the revision, set it to v4.2.1 if it is different

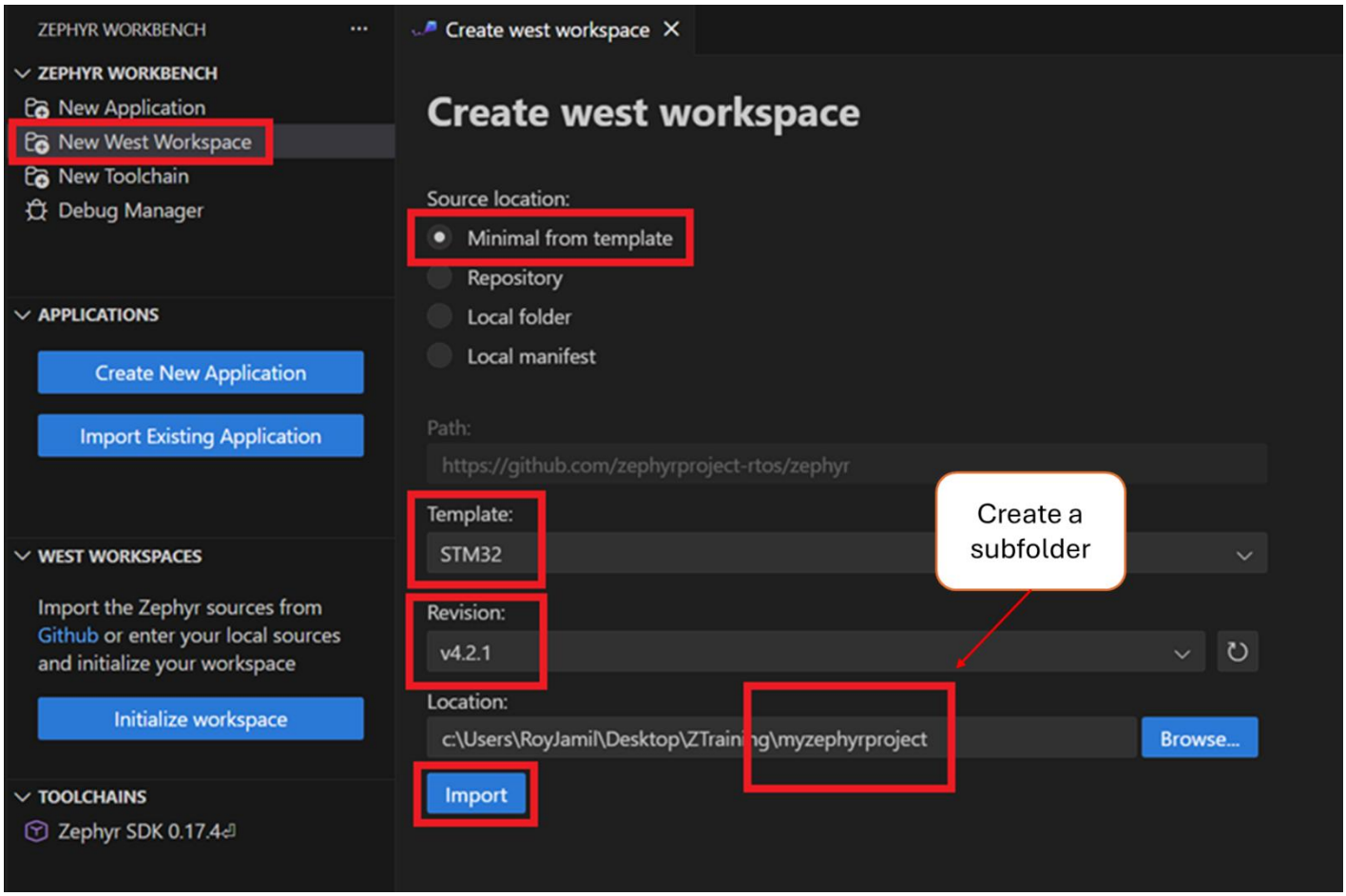


- Update west workspace:
    - Right click on the west workspace already imported then => update

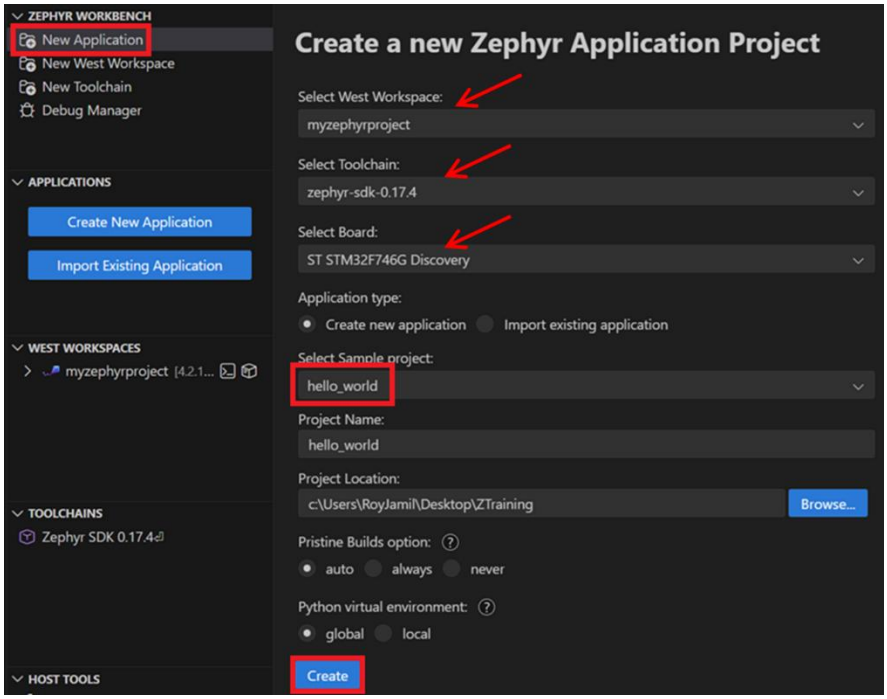


3.2 - If you have not already imported the workspace you need to import it (skip, if you already imported it):

- “New West Workspace” => minimal => STM32 => v4.2.1 => specify the folder (put it in a subfolder) => Click “Import”

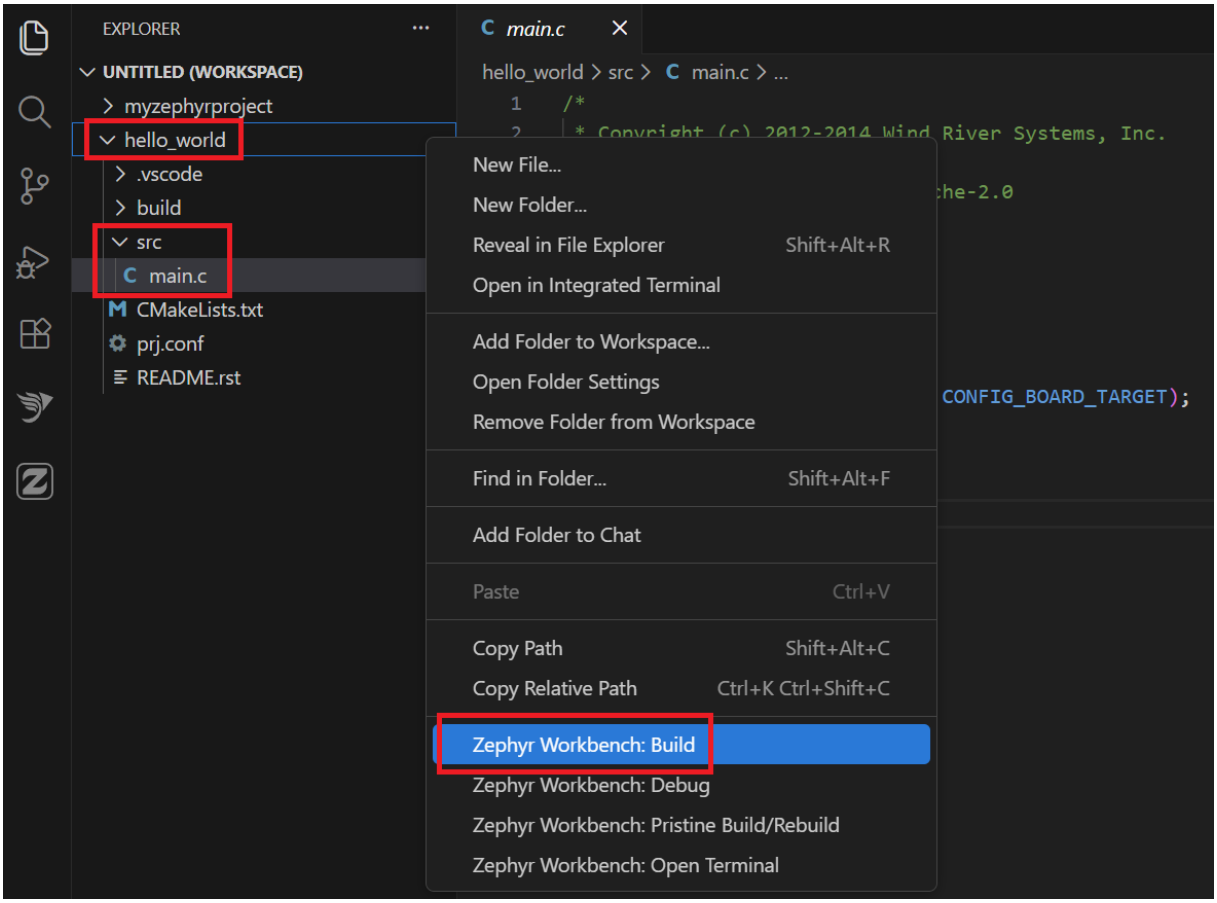


4- Create the Hello world application:

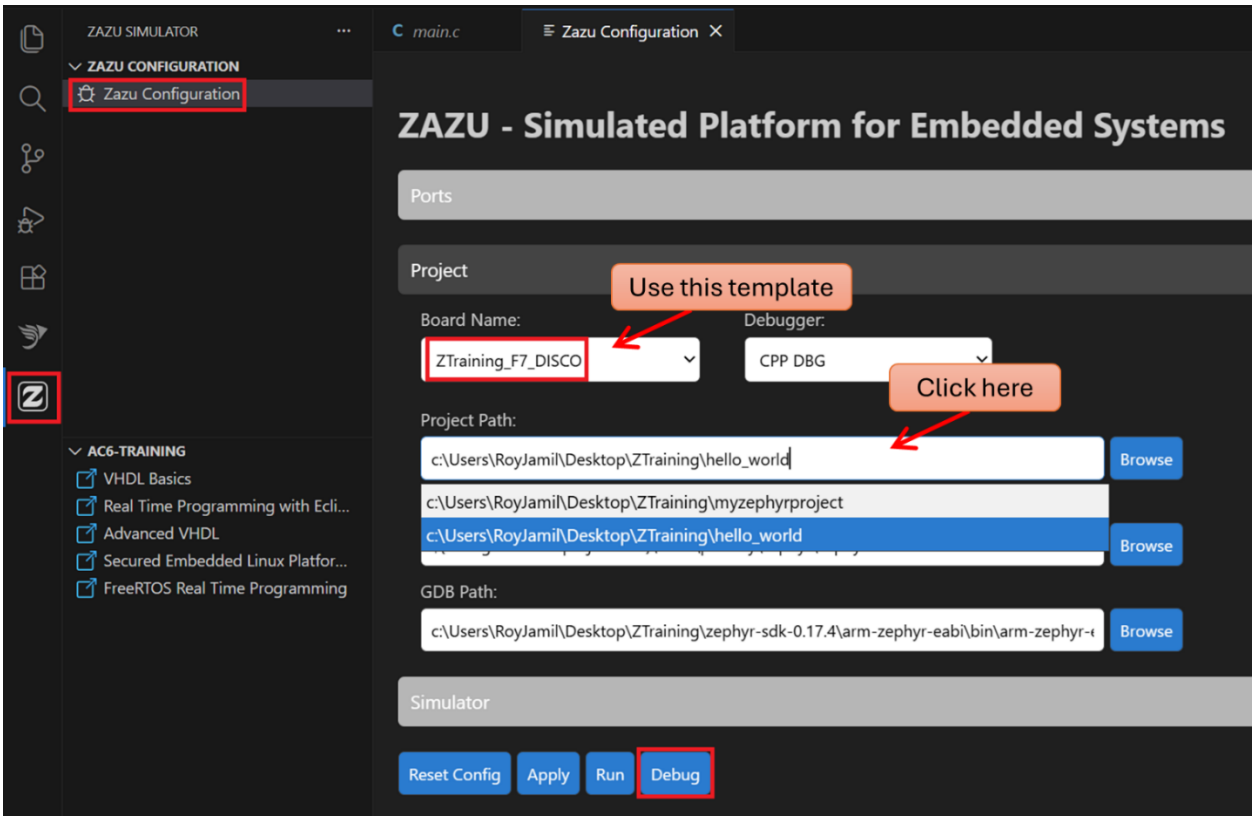


5- Build the application:

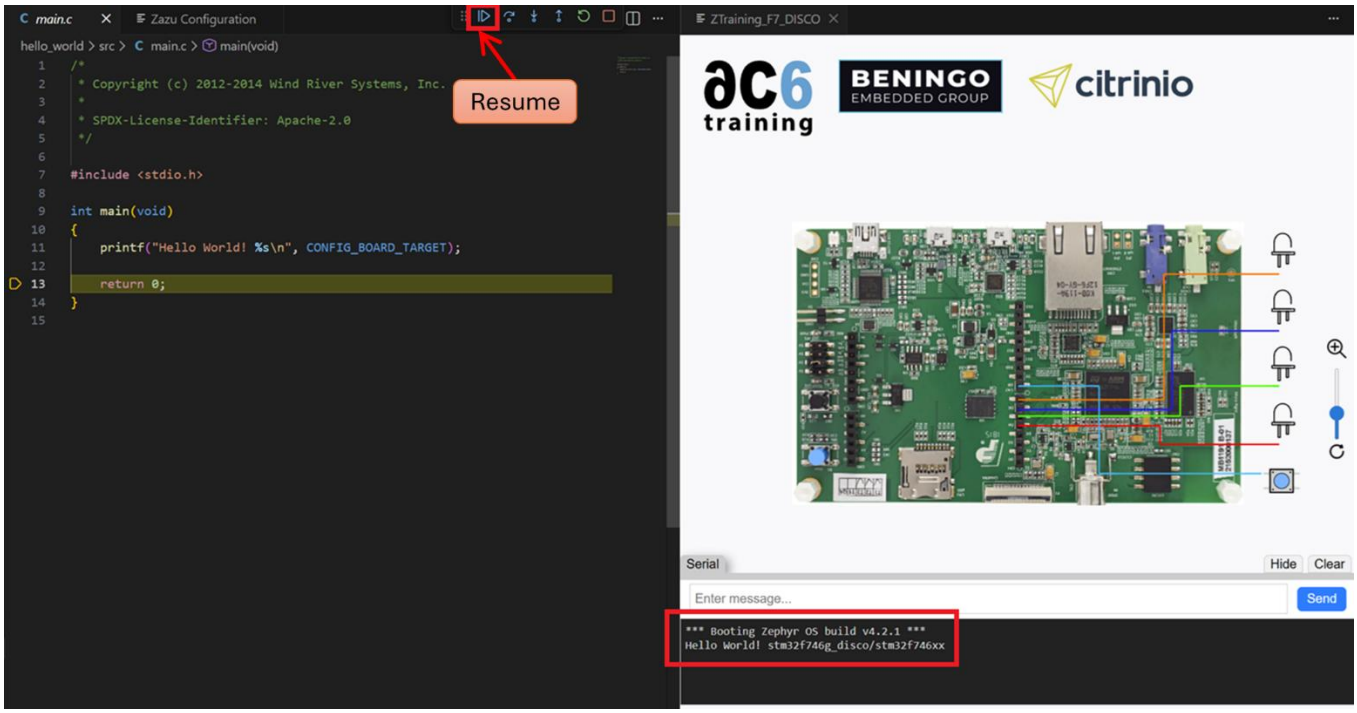
- Right click on the Hello World application, then build



6- Run and Debug:



7- Application Running:

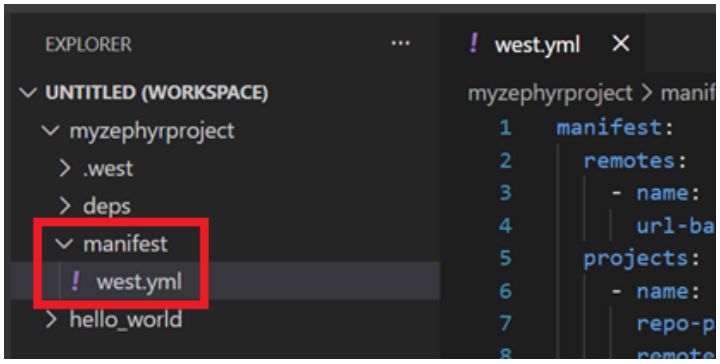


Lab2 – Import Labs using west

- Learn how to use west workspace
- Import Labs code and PDFs for the upcoming days

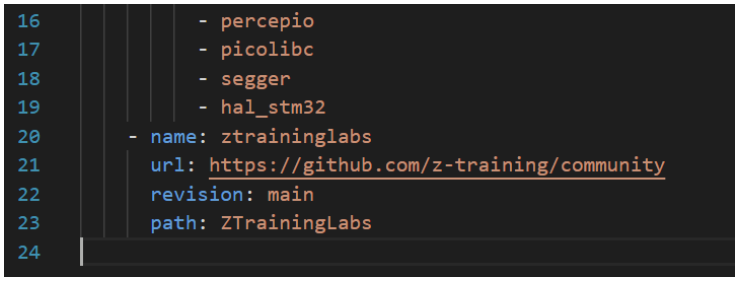
1- Open your workspace's west manifest file (west.yml)

a. Under the manifest folder
b. Open west.yml



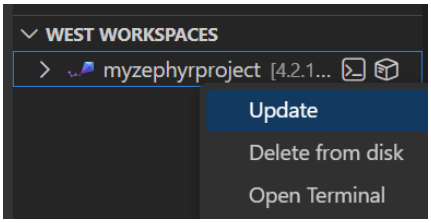
2- Add the following project entry

```
projects:
  - name: ztraininglabs
    url: https://github.com/z-training/community
    revision: main
    path: ZTrainingLabs
```

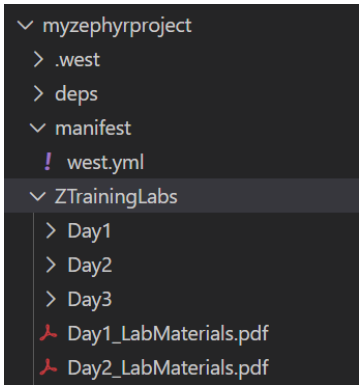


3- Update west workspace:

a. Go to the Workbench extension
b. Locate the west workspace we are using
c. Right click
d. Update



4- You should see a folder called ZTraininglabs in the west workspace containing the labs' example code

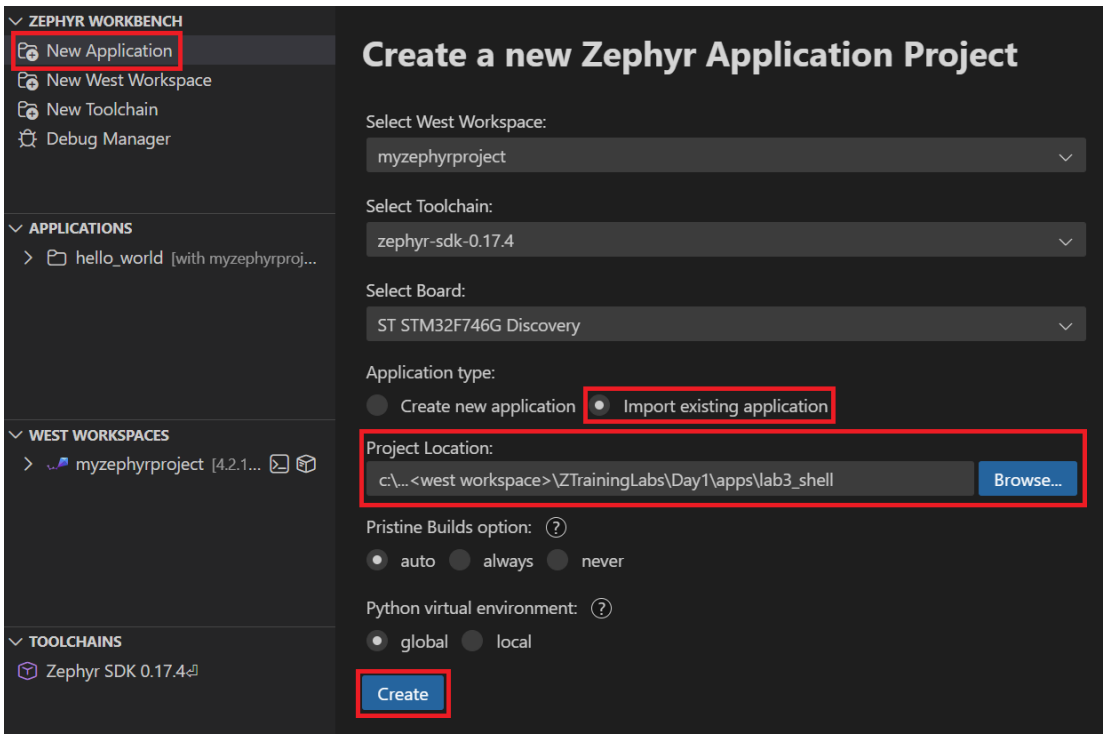


Lab3 – Shell and Kconfig

- Enable and configure the Zephyr shell using Kconfig.
- Create custom shell commands for interactive debugging.
- Practice structuring features using prj.conf and application code.

1- Import existing application:

a. Click on "New Application"
b. Set the west workspace and the Zephyr SDK
c. Select the `stm32f746g_disco` board
d. Select `Import existing application`
e. Give the path the `lab3_shell` folder, that is in your workspace



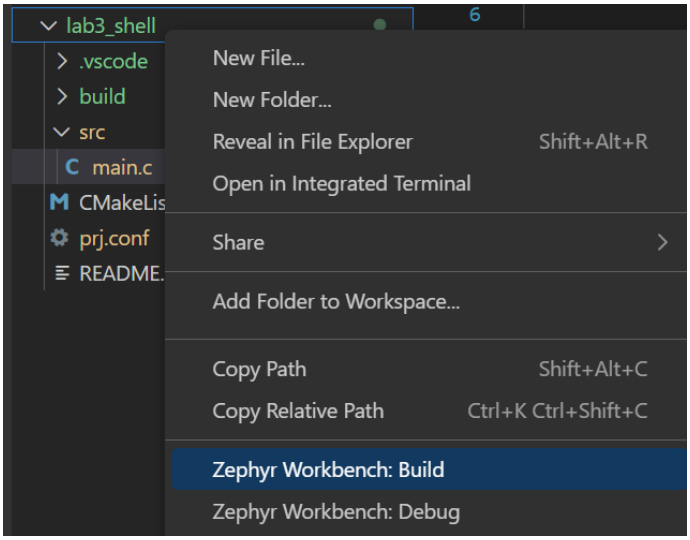
2- Procedure

a. Open lab3_shell/src/prj.conf
    i. Enable Shell in `prj.conf`: `CONFIG_SHELL=y`
b. Open lab3_shell/src/main.c
    i. Complete the missing code, indicated using "…"

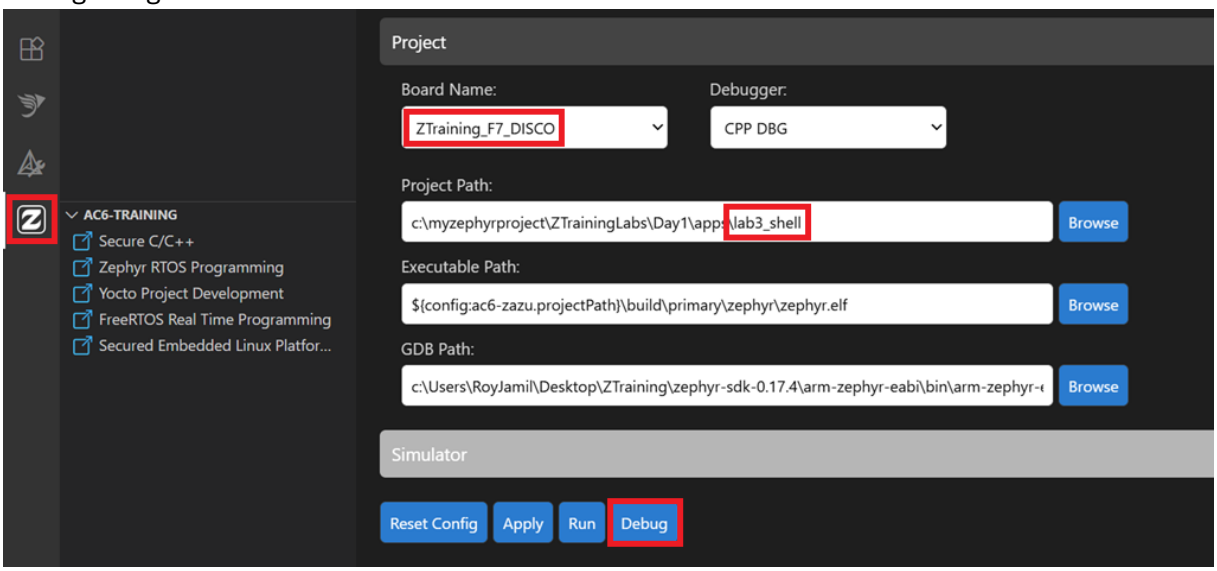
```
/* Subcommands available under "hakuna" */
SHELL_STATIC_SUBCMD_SET_CREATE(sub_hakuna,
    SHELL_CMD(matata, NULL, "Print Hakuna Matata message", cmd_matata),
    SHELL_SUBCMD_SET_END
);

/* Register main command */
SHELL_CMD_REGISTER(hakuna, &sub_hakuna,
                   "Hakuna command root", cmd_hakuna);
```

3- Build and debug

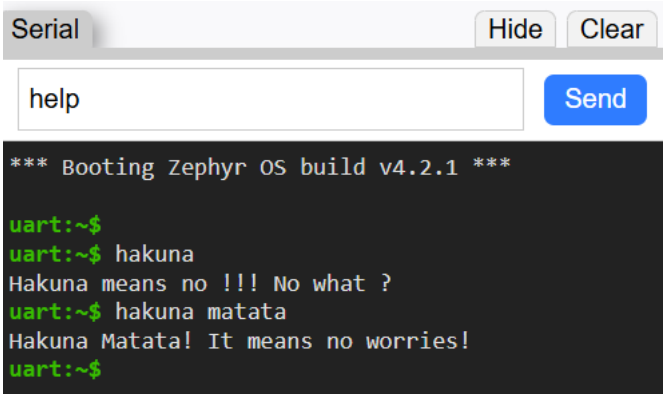
a. Right click on the application, then build



b. Debug using Zazu Simulator



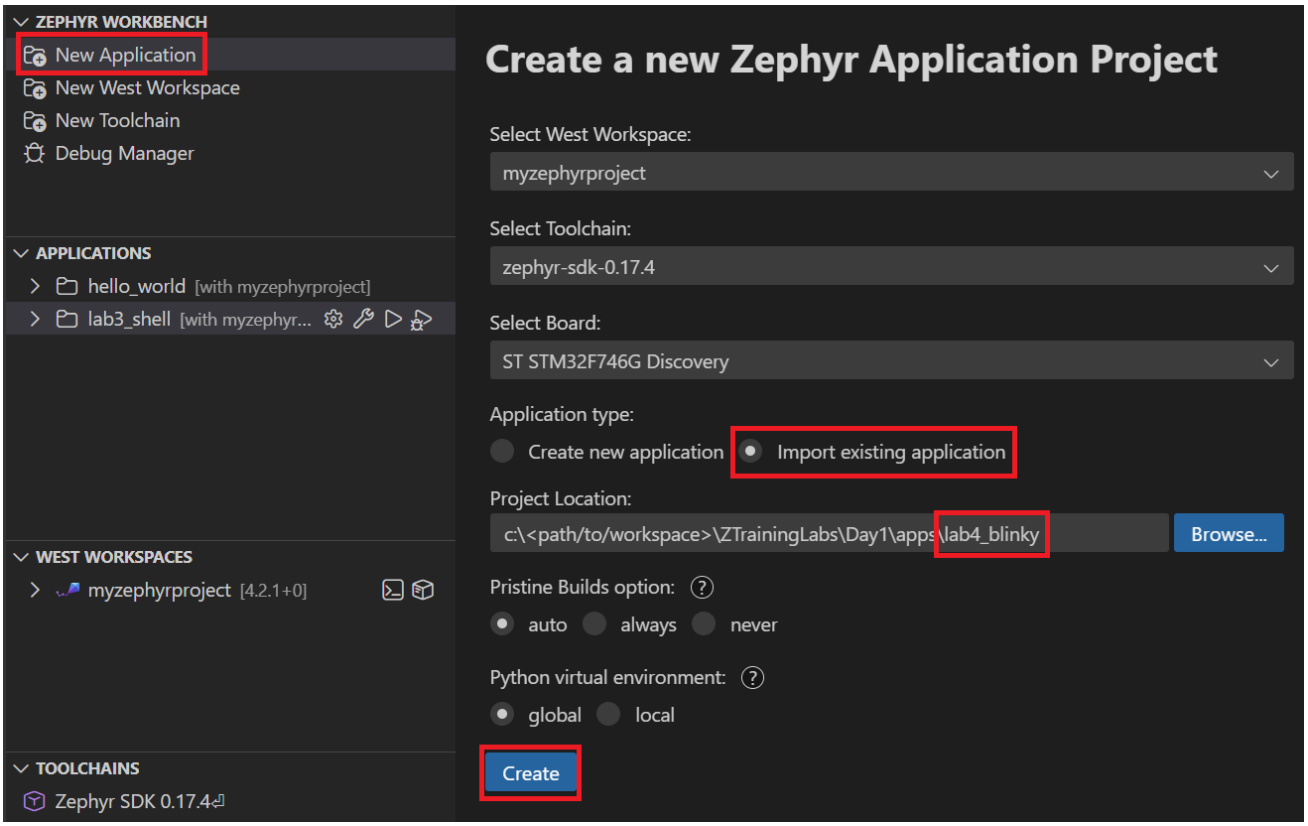
c. Test it



Lab4 – Devicetree and blinky

- Understand how hardware is described using Devicetree.
- Map external LEDs or peripherals through aliases and nodes.
- Implement a blinking LED application based on Devicetree definitions

1- Import existing application



2- Procedure

a. Check out the provided overlay
   i. lab4_blinky/boards/stm32f746g_disco.overlay
b. Provide the node id from the alias

```
/* Get LEDs from Devicetree */
static const struct gpio_dt_spec leds[] = {
    GPIO_DT_SPEC_GET(DT_ALIAS(my_led0), gpios),
    GPIO_DT_SPEC_GET(DT_ALIAS(my_led1), gpios),
    GPIO_DT_SPEC_GET(DT_ALIAS(my_led2), gpios),
    GPIO_DT_SPEC_GET(DT_ALIAS(my_led3), gpios),
};
```

c. Configure the gpio as output:

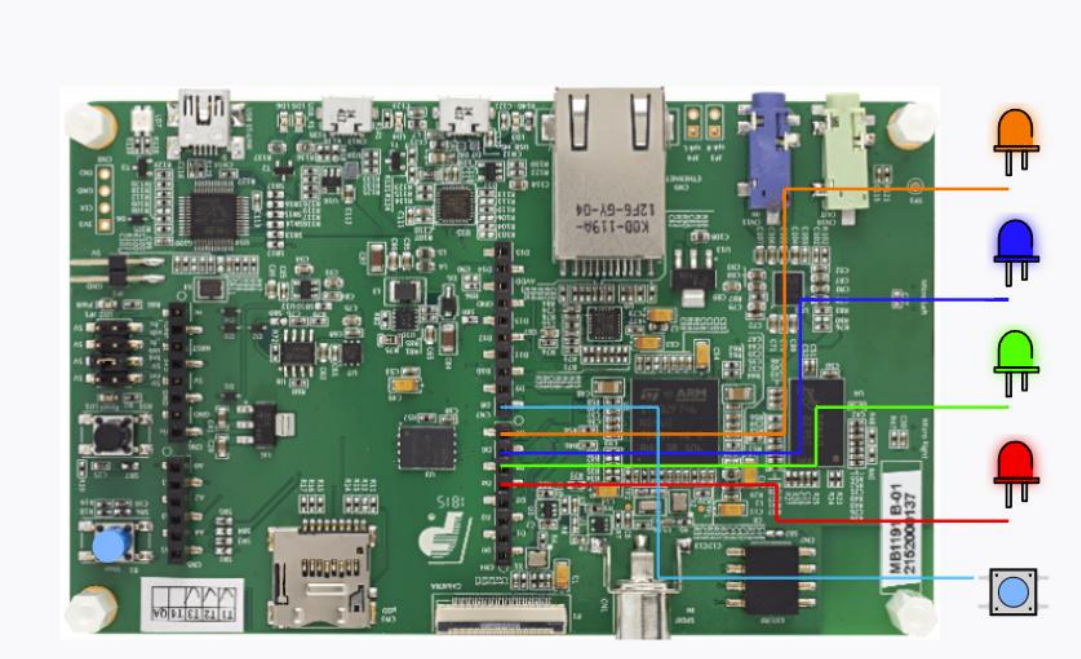
```
/* Configure the gpio */
gpio_pin_configure_dt(&leds[i], GPIO_OUTPUT_INACTIVE);
```

d. Toggle the GPIO

```
/* Toggle the gpio */
gpio_pin_toggle_dt(&leds[i]);
```

e. Build and debug the application, similar to what we have done in the previous labs

3- Expected output: the leds will blink



Software installation

The following software need to be installed onto each delegate PC (only Windows 10/11, Ubuntu > 20.04 or MacOS (ARM)) - If your system is not supported, please use the provided Docker image:

- Visual Studio Code (VSCode)
- Workbench for Zephyr Extension
- Zazu Simulator
- Percepio View

A) VSCode (if not already installed)

- Download the installer from

  https://code.visualstudio.com
- Install using the default options
- Alternatively, you may use the VSCode portable (this avoids modifying your existing VSCode configuration)

NOTE: You can follow this YouTube video to install and test the extensions B) and C):
https://youtu.be/WtQWrpUYF1g

B) Workbench for Zephyr extension

- Open VSCode
- Click on “Extensions” icon on the left 
- Search for “Workbench for Zephyr”
- Click on install
- Open Zephyr Workbench Extension using this icon 

  Under the “HOST TOOLS”

  Click on “Install host tools”

C) Zazu Simulator extension

- Click on “Extensions” icon on the left 
- *Search for “Zazu Simulator”* 
- *Click on install*
- *Install dependencies under “DEPENDENCIES” panel*

D) Percepio View

- Visit this link: https://traceviewer.io/get-view/?target=Zephyr
- Download and install the tool
- Optional: Register to access more tracing functionalities