# Getting Started With Zephyr RTOS

Quick Start Guide

## Introduction

The Zephyr Real Time Operating System is a small OS designed for embedded systems with limited resources. It can run on platforms with as little as 8KB of RAM and is compatible with a growing lineup of hundreds of development boards.

https://docs.zephyrproject.org/latest/introduction/index.html

While Zephyr can run on Windows and MacOS based systems, Linux based Ubuntu systems are more commonly used and offer enhanced capabilities. The next section contains a brief guide on installing Ubuntu 18.04.5 LTS (desktop). The remainder of the guide will apply to both Windows and Ubuntu environments for Microchip embedded controllers.

## 1 Ubuntu Installation

1. Create bootable USB thumb drive by following steps at this link: https://ubuntu.com/tutorials/create-a-usb-stick-on-windows#1-overview
2. Connect the USB drive to computer and reboot. Ubuntu should begin to install automatically. Otherwise, go to system BIOS and manually select boot drive.
3. Follow the steps at: https://ubuntu.com/tutorials/install-ubuntu-desktop#1-overview for setup details
4. Be sure to create an admin account in Ubuntu. This will give you the sudo access needed to install Zephyr's software and dependencies.
5. Finally, update all apps but **do not update Ubuntu OS**
6. Note: **Ubuntu version 18.04.5 LTS (desktop)** must be used with Zephyr. The latest Ubuntu 20.04 incorporates a different version of Python that will not work with our setup.

## 2 Set up Zephyr Environment

After successfully installing Ubuntu to your system, we can move to setting up the Zephyr environment. There are several dependencies that must be installed and a few roadblocks to watch out for that we'll describe below.

1. Follow steps in the Zephyr Getting Started Guide to install all necessary dependencies.
2. In steps 3-4, you may get the error: **"Command "python setup.py egg_infor" failed with error code 1 in /tmp/pip-build-18guqaf2/cryptography/"**

If this happens, PIP may need to be updated. Enter the following commands into Terminal:

**Cd~**
**Python3 –m pip install –U pip**

3. In section 5: "Build the Blinky Sample" "reel_board" can be used in place of "<your-board-name>"

## 3 Microchip EC-Specific Zephyr Environment Setup

Next, we'll want to install all the firmware needed for you system to interface with Microchip's MEC15xx family of controllers. Those steps are as follows:

1. The following link will walk you through getting the **embedded controller firmware framework code** for our MEC15xx controllers

https://intel.github.io/ecfw-zephyr/reference/getting_started.html#getting-ec-fw-framework-code
(Some of these steps will have been completed during the Zephyr Environment Setup)

2. In Step 5: "EC SoC vendor-specific setup", follow steps in "Programming and Debugging Setup" at the link below:

https://docs.zephyrproject.org/latest/boards/arm/mec1501modular_assy6885/doc/index.html#programming-and-debugging

Microchip EC SPI Generator tools can be downloaded from: https://github.com/MicrochipTech/CPGZephyrDocs

- Unzip the file after download
- Set environment variable **(this needs to be done each time you open the terminal to compile code for SPI image generation)**

**For MEC1501 Linux®:**
export EVERGLADES_SPI_GEN=~/CPGZephyrDocs/MEC1051/SPI_image_gen/everglades_spi_gen_lin64

**For MEC1501 Windows®:**
export EVERGLADES_SPI_GEN=~/CPGZephyrDocs/MEC1051/SPI_image_gen/everglades_spi_gen

**For MEC152x, use:**
**export EVERGLADES_SPI_GEN=~/CPGZephyrDocs/MEC152x/SPI_image_gen/everglades_spi_gen_RomE**

**Note:** Make sure to set permission of the spi_gen file you intend to use. Otherwise, you will get the following error when you build the FW:
/bin/sh: 1: /home/mchp/CPGZephyDocs/MEC152x/SPI_image_gen/everglades_spi_gen_RomE: Permission denied
Ninja: build stopped: subcommand failed
FATAL ERROR: command exited with status 1: /usr/bin/cmake –build /home/mchp/sandbox/ecfw-zephyr/build

To set permission:

- Right click on the spi_gen file to open properties
- At "Permissions" tap, check "Allow executing file as program"

3. Continue to "Getting EC FW framework code" section

4. "Werror" is usually enabled in newer compiler versions, and it causes logging errors during compilation. To disable Werror:

- Open "~/sandbox/ecfw-zephyr/CMakeLists.txt"
- Comment out the line "zephyr_compile_options(-Werror)"

5. **For Zephyr v2.3 branch only:** Replace "~/sandbox/zephyr_snapshot/boards/arm/mec1501modular_assy6885/support/spi_cfg.txt" with:
"~/CPGZephyrDocs/MEC152x/SPI_image_gen/spi_cfg.txt"
(copy paste over)

6. Follow steps in "Building and flashing" section to build code

- Note that you need to run this each time a new terminal is opened

cd ~/sandbox/zephyr_snapshot
Source zephyr-env.sh

- When switching boards or switching makefile generation, you need to clean the build folder first:

cd ecfw-zephyr
rm –rf build

- Build the application

cd ~/sandbox/ecfw-zephyr
west build -c -p auto -b mec1501modular_assy6885

**Note:** if EVERGLADES_SPI_GEN is not set up, you will only get *.elf after compiling the code
Use this command to see if EVERGLADES_SPI_GEN is set up with the right path:
printenv | grep SPI_GEN

You should see something like this returned:
EVERGLADES_SPI_GEN=/home/mchp/CPGZephyrDoc/MEC152x/SPI_image_gen/everglades_spi_gen_RomE

## 4 Summary: Command Sequence to Build Code

** spi_cfg.txt in ~/sandbox/zephyr_snapshot/boards/arm/mec1501modular_assy6885/support/spi_cfg.txt is for MEC150x. **To generate SPI image for MEC152x, replace this file with** ~/CPGZephyrDocs/MEC152x/SPI_image_gen/spi_cfg.txt **

cd ~/sandbox/zephyr_snapshot
source zephyr-env.sh
cd ~/sandbox/ecfw-zephyr
**For MEC1501 (Obsolete)**
export EVERGLADES_SPI_GEN=~/CPGZephyrDocs/MEC1501/SPI_Image_gen/everglades_spi_gen_lin64

or

**For MEC1521**
export EVERGLADES_SPI_GEN=~/CPGZephyrDocs/MEC152x/SPI_Image_gen/everglades_spi_gen_RomE

rm –rf build
west build –c –p auto –b mec1501modular_assy6885
See successful build log file for reference

## 5 Internal Flash (MEC1527 Programming)

- Use MEC1527_Flash_Utility_v1.32
- Copy spi_image.bin from Ubuntu ~/sandbox/ecfw-zephyr/build/zephyr to Windows and place the file in the same folder where the MEC1527 Flash Utility v1.32 resides
- In CMD, cd to MEC1527_Flash_Utility_v1.32
- At prompt, issue the command: MEC1527_Flash_utility –w –f spi_image.bin

You should see the output in the image to the right when flash programming is successful.

```
MEC1527 Flash Programming Utility v1.32
The argument of -w : Write SPI-Flash
The argument of -f (filename) : spi_image.bin

Flash programming Utility

========flash_option is PROGRAM_FLASH======
Transfer bytes 0KB - 128KB to MEC1527...
Transfer bytes 128KB - 256KB to MEC1527...
Transfer bytes 256KB - 384KB to MEC1527...
Transfer bytes 384KB - 512KB to MEC1527...

***Flash Programming process completed SUCCESSFUL!

Pass
0
```

## 6 Board Bring-Up

- After programming flash and resetting the board, LED1, LED7 and LED8 should be lit. If any of the LED's are not lit, check board power jumper options
- Zephyr code built for MEC15xx outputs are UART1 – load JP75 [10-11, 13-14]
- Terminal Setting
  Speed: 115200
  Data: 8 bits
  Parity: None
  Stop bits: 1

You should see "Hello World! mec1501modular_assy6885" in terminal output

## 7 Contribution and Coding Style Guidelines

As an open-source project, community members can submit patches and other code directly to the Zephyr repository. To maintain ubiquity and legibility throughout the project, follow Zephyr's "Contribution Guidelines" when making any code submissions, updates/patches, or bug reports.