Contribution ID: **190**　　Type: **not specified**

How to extract a bare metal flavor of code out of Zephyr to use in RTOS ?

Zephyr RTOS offers a rich ecosystem, but embedded engineers often face the challenge of porting code across environments—from Zephyr to another RTOS or even to bare metal. This discussion is about a practical guide to extracting a "bare metal flavor"of code out of Zephyr so that it can run independently of Zephyr's driver and subsystem layers.

NOTE: The HAL approach isn't the right solution, it is making the code stay out of common code base, not following Zephyr standards or coding principles, not gaining advantages of opensource community for reviews of design architecture and the source code.

We will explore:

1. Structuring Zephyr projects so application and peripheral code can be isolated cleanly
2. Writing drivers and hardware access layers that can compile both inside and outside Zephyr
3. Techniques for minimizing dependencies on Zephyr APIs
4. A case study: extracting a Zephyr-based peripheral driver and reusing it in a bare-metal project
5. Beyond the mechanics, we'll discuss the philosophy of portability—how to strike the right balance between leveraging Zephyr abstractions and keeping code flexible enough to live elsewhere.

Primary author: SYED MOHAMMED, Khasim (Texas Instruments)

Co-authors: Mr SINGH, Amneesh (Texas Instruments); Mr DAVIS, Andrew (Texas Instruments); Mr TRIPATHY, Soumya (Texas Instruments)

Presenter: SYED MOHAMMED, Khasim (Texas Instruments)

Session Classification: Embedded & Internet of Things MC

Track Classification: Embedded & Internet of Things MC