

SE-102

Zephyr: A Portable Software Platform



Matt Gordon

Staff Product Manager, IoT Embedded SW & OS,
Wireless



Introduction to Zephyr

Governance



Open-Source LF
Project Governed by
30+ Member
Companies



Platinum

Silver



antmicro



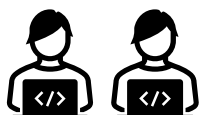
Memfault



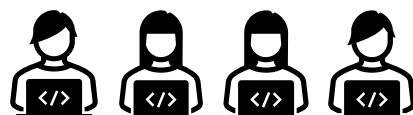
Community



Maintainer



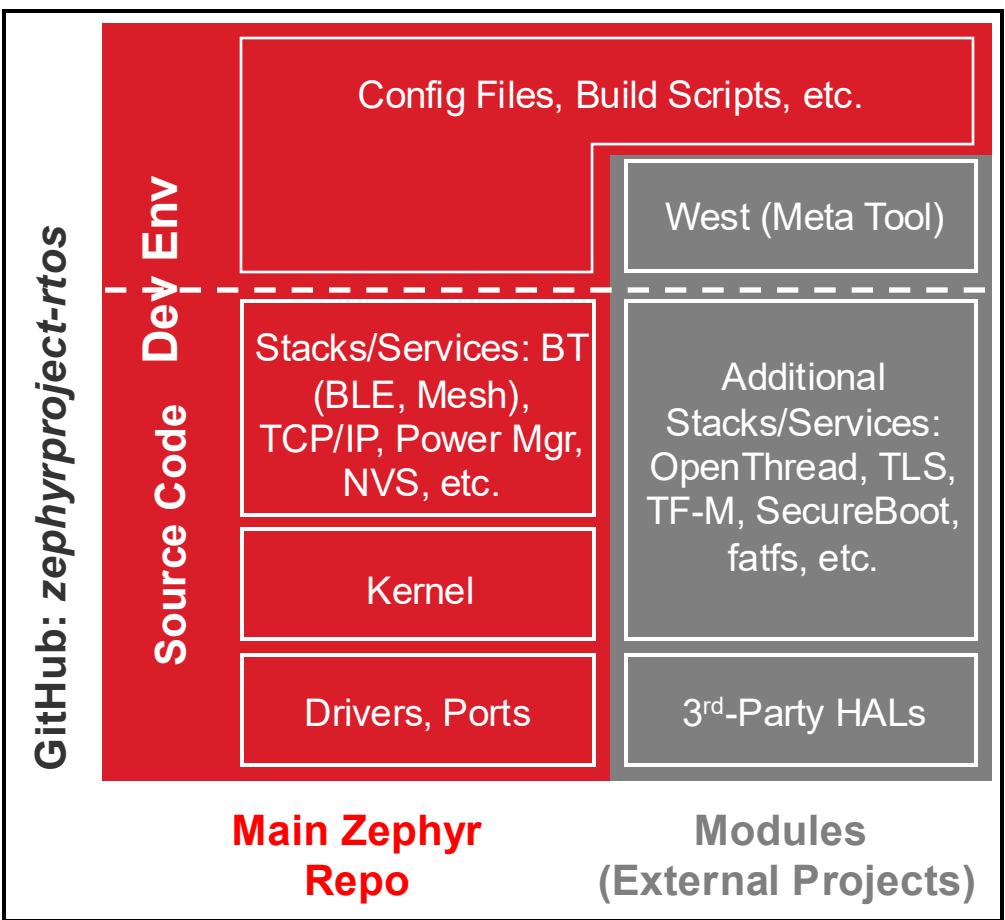
Collaborator



Contributor

Contributions from 1000+
Developers











Professional Development
Process w/ Code Reviews, CI,
LTS Releases, Safety &
Security Audits, Bluetooth
Qualification



Silicon Labs Zephyr Activity

- 2021 - Silicon Labs became project member**
 - Silver membership level
 - Internal efforts on Zephyr porting began earlier
- 2022 - Began working with Antmicro**
 - Longtime Zephyr member and contributor
 - Added several Silicon Labs kits to repo
- 2024 - Moved all efforts in - house**
 - Established dedicated Zephyr development team
 - Hired longtime Zephyr contributor Johan Hedberg
 - Began expanding support to more kits, technologies
- 2025 - Upgraded to Platinum membership**
 - Looking to drive low-power wireless initiatives
 - Johan now Chair of Bluetooth working group

Linux Foundation Zephyr Organizations Leaderboard (Q2 2025)*

	The Linux Foundation	8,887 - 1%
	NXP Semiconductors N.V.	7,780 - 1%
	STMicroelectronics Netherlands B.V.	6,281 - 1%
	ARM Corporation	5,786 - 1%
	Infineon Technologies International AG	5,478 - 0%
	Meta Platforms, Inc.	2,205 - 4%
	Google LLC	2,091 - 4%
	Silicon Laboratories Inc.	1,941 - 3%
	Zephyr Project	1,657 - 3%
	Nuvoton Electronics Corporation	1,478 - 0%

*Source: [LFX Insights](#)

An Open and Portable Solution

- **Portable APIs are common in the RTOS world**
 - Typical for kernel services: task management, mutexes, etc.
- **Zephyr offers a range of drivers and subsystems**
 - Power management, storage, I/O, etc.
 - Generally, these APIs are portable across HW
- **Minimizes work needed to port app code**
 - Many Zephyr sample apps can run on multiple devices
- **Also simplifies adoption of other SW modules**
 - With one Zephyr port, stacks can run on various devices
- **All code contributed to Zephyr is open-source**
 - Project requires open-source licensing
 - Most code licensed under Apache 2.0 terms
- **Limits barriers to adoption**
 - Also encourages diverse and active community

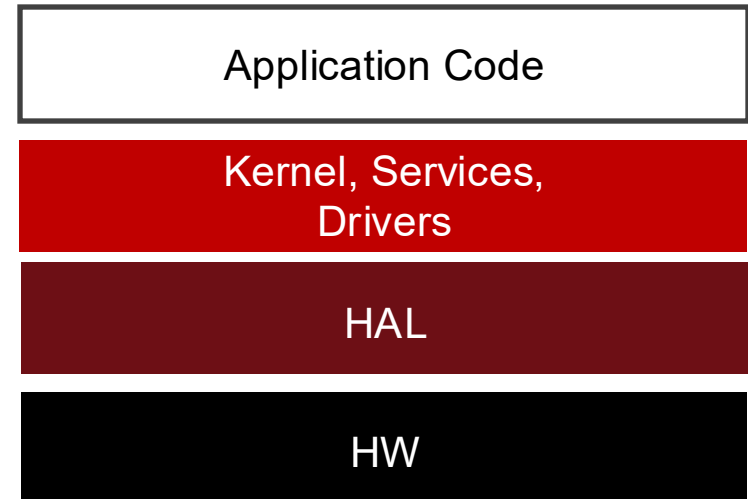
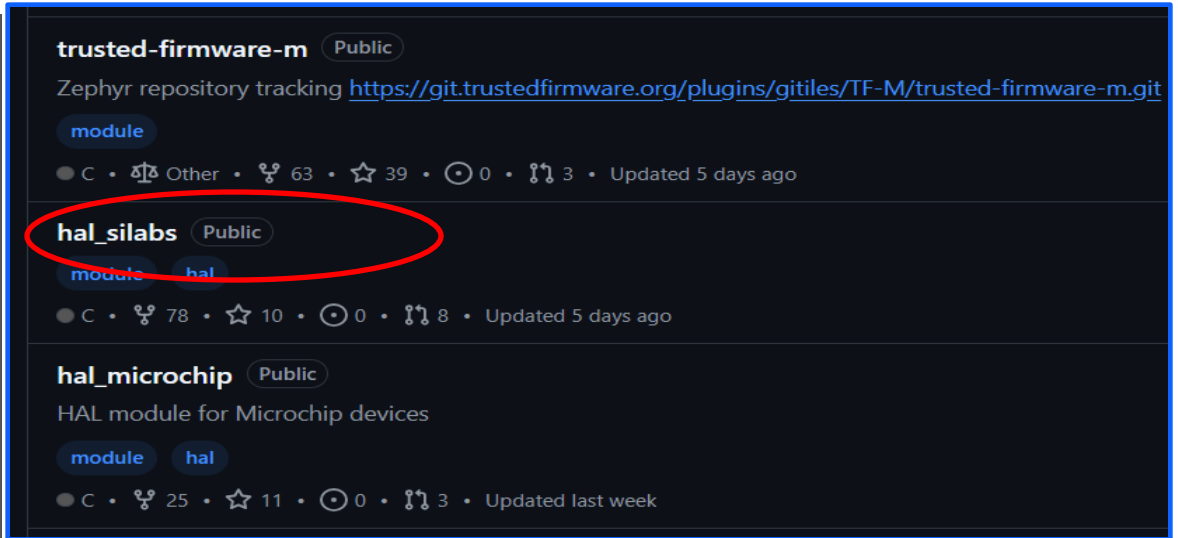
UART Example

```
while (1) {  
    k_sleep(K_SECONDS(5));  
    num_tx = (sys_rand32_get() % LOOP_ITER_MAX_TX) + 1;  
    LOG_INF("Loop %d: Sending %d packets",  
            loop_counter, num_tx);  
    for (int i = 0; i < num_tx; i++) {  
        tx_buf = net_buf_alloc(&tx_pool, K_FOREVER);  
        tx_len = snprintk(tx_buf->data,  
                           net_buf_tailroom(tx_buf), "Loop%d:  
                           Packet: %d\r\n", loop_counter, i);  
        net_buf_add(tx_buf, tx_len);  
        rc = uart_tx(uart_dev, tx_buf->data, tx_buf->len,  
                     SYS_FOREVER_US);  
        ...  
    }  
}
```

Red indicates portable functions

Porting Zephyr

- Early in its history, the Zephyr project decided to adopt a HAL model
 - Somewhat controversial, as HALs are vendor-specific
- With HALs, the process for Silicon vendors to support Zephyr is streamlined
 - Use their own HW-specific code as the basis of their drivers
- HAL availability allows bulk of Zephyr contributors to focus on core functionality
 - HW-specific code is primarily developed and tested by maintainers of HAL repos



Upstream vs Downstream

Upstream

- Zephyr public repo
- Silicon Labs HW is enabled
- Community support available



Downstream

- Silicon Labs Zephyr SDK
- Silicon Labs HW is supported
- Full technical support



- **Zephyr is developed in a public GitHub repo**
 - “Upstream” in open-source terms
 - Project has many contributors & repo is an active space
- **Silicon Labs will offer a Zephyr SDK**
 - The SDK will be a fork of the upstream repo
 - “Downstream,” in open-source terms
- **SDK will contain the officially supported code**
 - Fork chosen to give Silicon Labs more control over code
 - Simplifies our Tech Support and SQA efforts
- **Intention is not for a “hard” fork**
 - Upstream and downstream will not deviate considerably
- **Upstream-first approach is being followed**
 - Most downstream functionality will be available upstream
 - Support status is primary difference between repos

Peripheral Support Overview

Feature	EFR32	SiWx917
Power Manager	✓	✓
UART	✓	✓
SPI	✓	✓
I2C	✓	✓
I2S	N/A	✓
PWM	✓	✗
GPIO	✓	✓
ADC	✓	✓
Comparator	✓	✗
DAC	✓	✗
Timers	✓	✓

Feature (Cont.)	EFR32	SiWx917
DMA	✓	✓
PSRAM (via QSPI)	N/A	✓
Flash Read and Write	✓	✓
GUI	✗	✓
Crypto	✓	✓
Secure Boot	✓	✓

Kits Supported for Zephyr

EFR32

■ 25Q2

- xG29 Radio Board
 - BRD4414A

■ 25Q4

- **xG22 Starter Kit (SLWSTK6021A)**
- xG22/xGM22 Additional Radio Boards
 - BRD4311A
- xG22 Explorer Kit (BG22-EK4108A)
- **xG22 Thunderboard (SLTB010A)**
- **xG24 Pro Kit (xG24-PK6009A)**
- xG24 Additional Radio Boards
 - BRD4187C, BRD4316A, BRD4317A
- xG24 Explorer Kit (xG24-EK2703A)
- **xG24 Dev Kit (xG24-DK2601B)**
- **xG27 Pro Kit (xG27-PK6017A)**
- xG27 Additional Radio Boards
 - BRD4110B, BRD4111B
- **xG27 Dev Kit (xG27-DK2602A)**

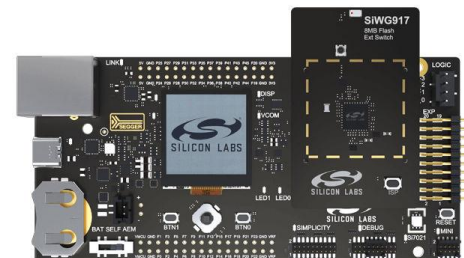
SiWx917

■ 25Q2

- Pro Kit (SiWx917-PK6031A)

■ 25Q4

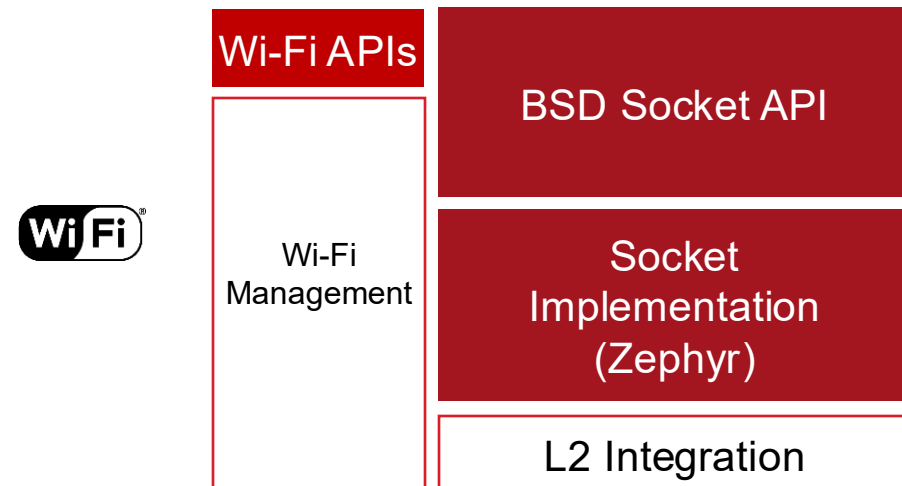
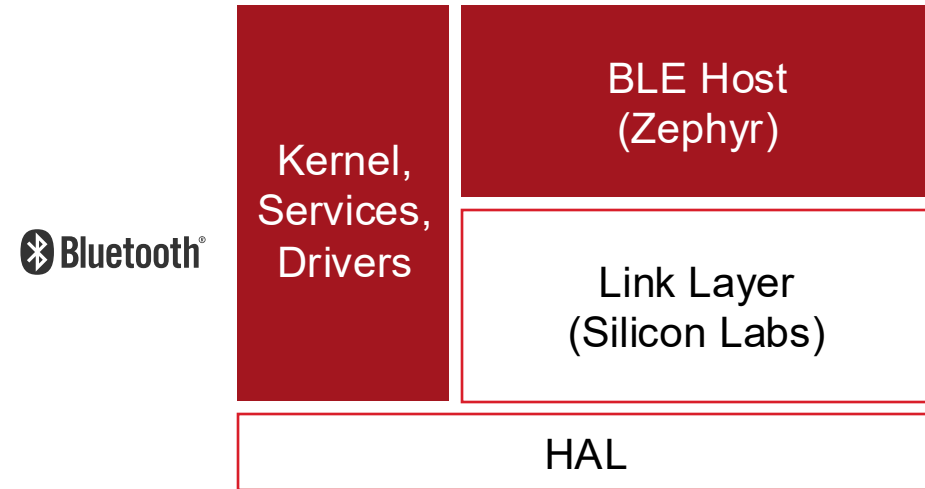
- Dev Kit (SiWx917-DK2605A)



Kits in bold will be supported in both upstream and downstream

Wireless Connectivity in Zephyr

- **Wireless connectivity central to Silicon Labs' Zephyr involvement**
 - Leverage our expertise in RF and wireless software
 - Silicon Labs engineer is current Chair of Zephyr Bluetooth Working Group
- **Initial focus in Zephyr support efforts has been BLE and Wi-Fi**
 - Zephyr stacks and APIs leveraged in both cases
 - Support for additional wireless technologies is currently being planned



Programming Model

- **Zephyr is based on a multi-threaded kernel**
 - At a high-level, somewhat similar to FreeRTOS
 - Multi-threading can be disabled, but it is required by many subsystems
- **Application code creates threads using kernel APIs**
 - `k_thread_create()` for creating threads dynamically
 - `K_THREAD_DEFINE()` for static threads
- **Each thread requires stack space and a priority**
 - Priorities can be negative or positive numbers
 - Negative priorities correspond to cooperative tasks that must yield CPU
 - Positive priorities are for pre-emptive tasks that take the CPU from one another
 - Range determined by `CONFIG_NUM_COOP_PRIORITIES` and `CONFIG_NUM_PREEMPT_PRIORITIES`
- **Various services provided for thread interaction**
 - Semaphores, mutexes, conditional variables, events, etc.
- **Extensive documentation provided for all Zephyr kernel services**
 - Code samples illustrating various services are available as well

Current Upstream Repo Status

EFR32xG24 Dev Kit (xG24-DK2601B)


Overview

The EFR32MG24 Mighty Gecko Board dev kit contains a Wireless System-On-Chip from the EFR32MG24 family built on an ARM Cortex®-M33F processor with excellent low power capabilities.

Hardware

- EFR32MG24B310F1536IM48-B Mighty Gecko SoC
- CPU core: ARM Cortex®-M33 with FPU
- Flash memory: 1536 kB
- RAM: 256 kB
- Transmit power: up to +20 dBm
- Operation frequency: 2.4 GHz
- Crystals for LFXO (32.768 kHz) and HFXO (38.4 MHz).
- On board devices:
 - Silicon Labs Si7021 relative humidity & temperature sensor

Board Overview



EFR32xG24 Dev Kit (xG24-DK2601B)

Name: xg24_dk2601b
Vendor: Silicon Laboratories
Architecture: arm
SoC: efr32mg24b310f1536im48

[Browse board sources](#)

- **Several kits and boards *enabled* upstream today**
 - BG22 Thunderboard
 - xG24 Dev Kit
 - xG24 Explorer Kit
 - xG24 Radio Board (RB4187C)
 - xG27 Dev Kit
 - xG29 Radio Board (RB4412A)
 - SiWx917 Radio Board (SLWRB4338A)
- **Available drivers listed in “Supported Boards”**
 - DMA, SPI, I2C, etc.
 - BLE enabled on EFR32, Wi-Fi & BLE on 917
 - Certain examples may not run due to HW limitations
 - <https://docs.zephyrproject.org/latest/boards/index.html#>
- **Code not fully supported with Apps resources**
 - Silicon Labs is official Maintainer of upstream repos
 - Zephyr Virtual Team will address issues, PRs, etc.

Getting Started

- **Upstream repo contains the code needed for Bluetooth development on EFR32 and Wi-Fi on SiWx917**
- As indicated previously, fully supported and tested code will be coming soon via downstream
- **The Zephyr project provides a generic getting-started guide**
 - Walks through the steps to clone the repo, install tools, etc.
- **Guide can be supplemented with content from relevant “Supported Boards” pages**
 - Indicate peripherals supported on each board, along with any changes to standard getting-started procedure

Getting Started Guide

Follow this guide to:

- Set up a command-line Zephyr development environment on Ubuntu, macOS, other Linux distributions are discussed in [Install Linux Host Dependencies](#))
- Get the source code
- Build, flash, and run a sample application

Select and Update OS

Click the operating system you are using.

Picking a Software Platform for Your Project



Choose Zephyr if you -

Place a high value on portability

Prefer the open-source development model

Are already familiar with Zephyr

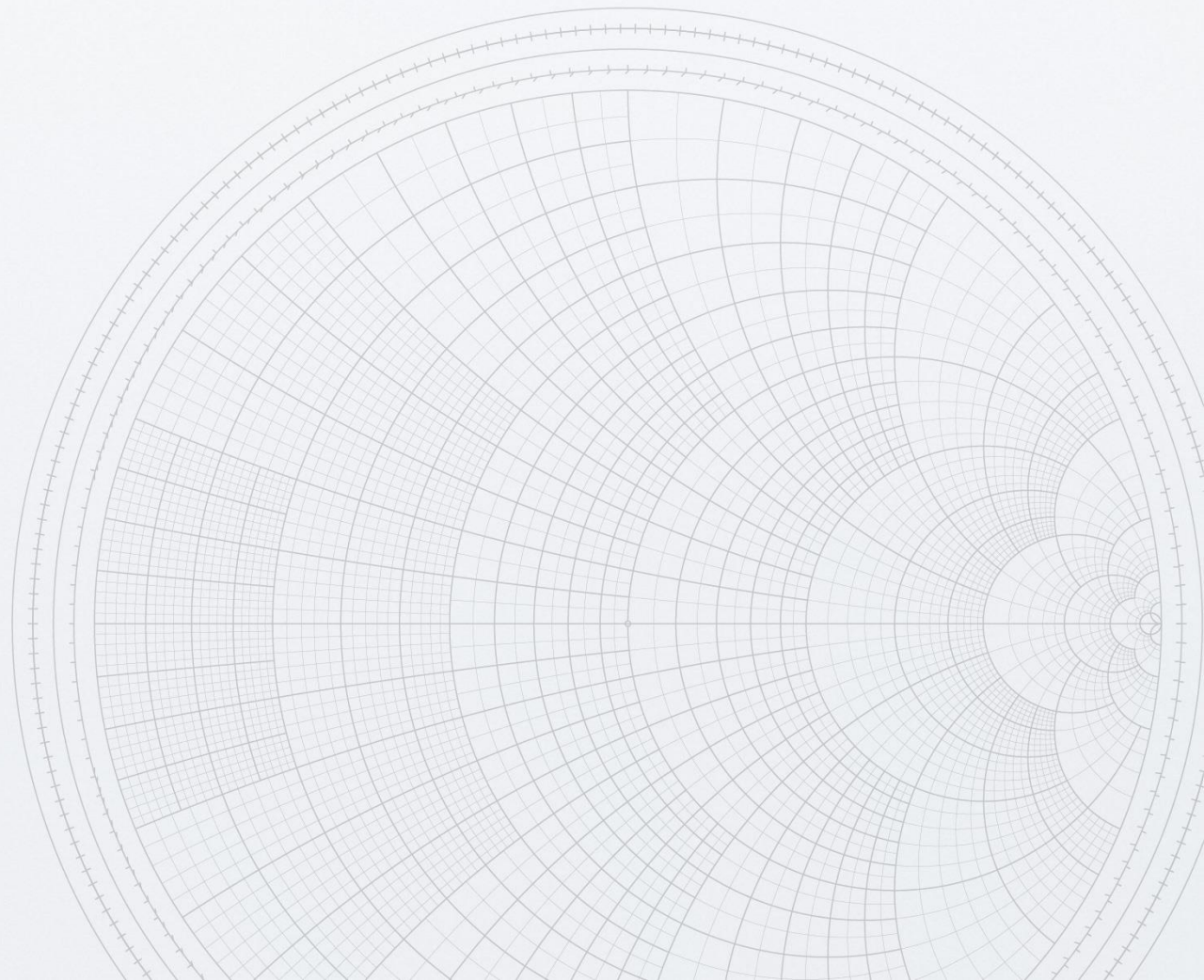
Choose Silicon Labs Simplicity SDK if you -

Require highly optimized performance

Favor graphical project generation/config tools

Are a FreeRTOS user or prefer bare-metal

Q&A





SILICON LABS

CONNECTED INTELLIGENCE