Apollo3 & Apollo4 Family Zephyr Training

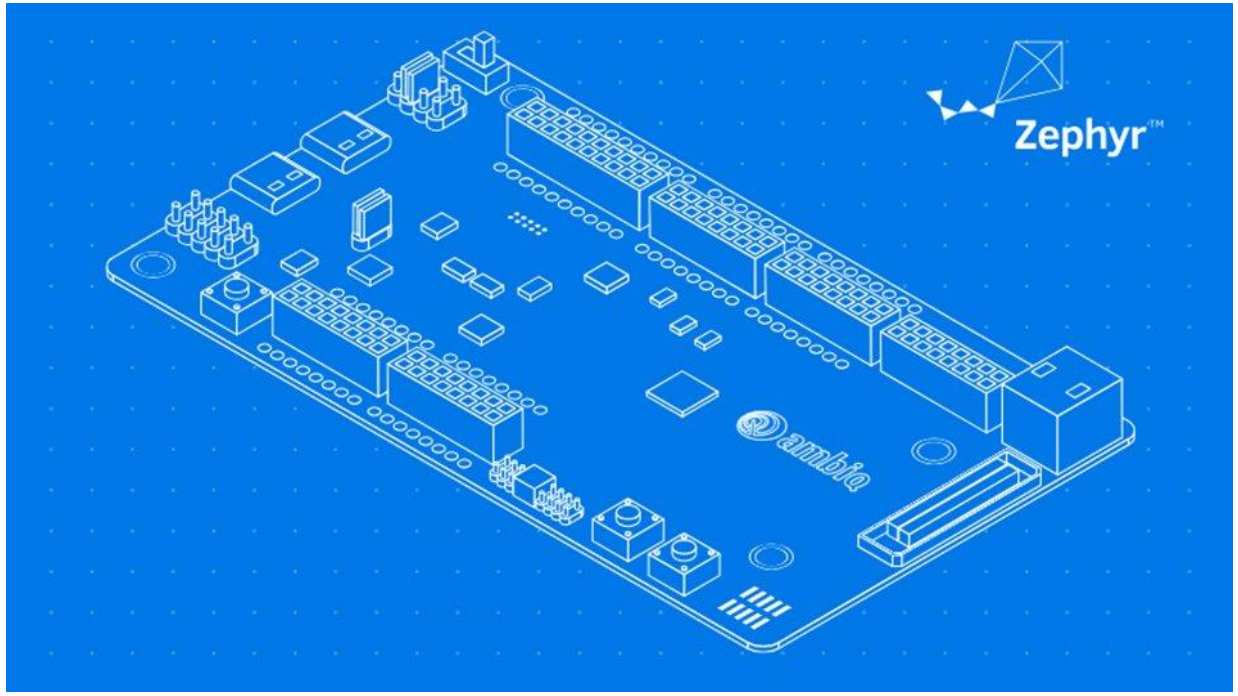
January 2025

Agenda

- Overview of Zephyr Ecosystem
- Ambiq Components in Zephyr
- Getting Started with Zephyr

Prerequisites

- Please try out the Zephyr build/test environment
  - See Getting Started Guide — Zephyr Project Documentation.
  - Slides #30 - 32 in this deck also provides some pointers
- Clone Ambiq's fork of Zephyr (Ambiq-Stable)
  - See GitHub - AmbiqMicro/ambiqzephyr: Ambiq fork of Primary Git Repository for the Zephyr Project.
- Links are included in the README page on https://github.com/AmbiqMicro/ambiqzephyr



Overview of Zephyr Ecosystem

Zephyr Overview

Zephyr OS

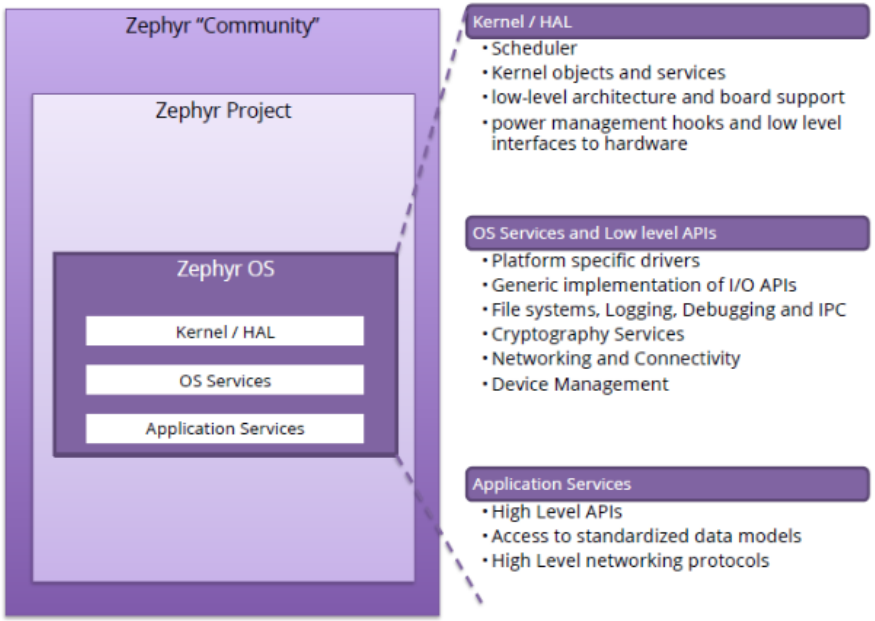
- The kernel and HAL
- OS Services such as IPC, Logging, file systems, crypto

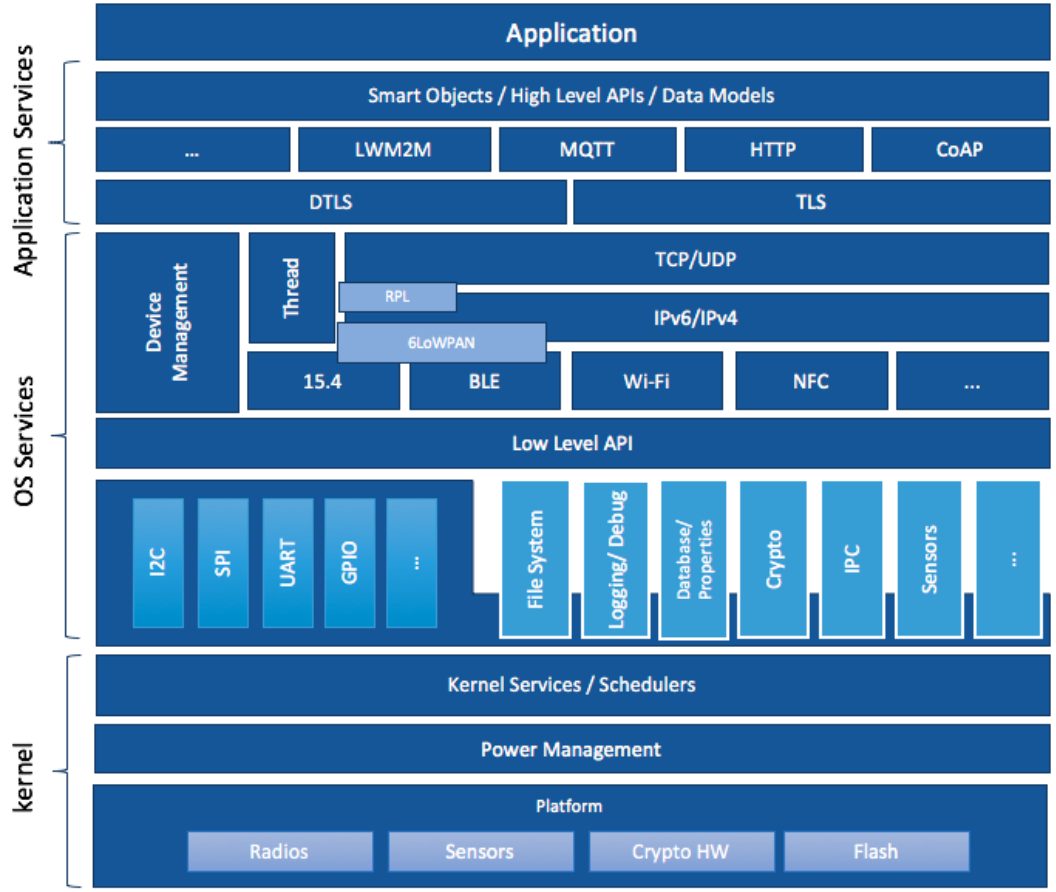
Zephyr Project

- SDK, tools and development environment
- Additional middleware and features
- Device Management and Bootloader

Zephyr Community

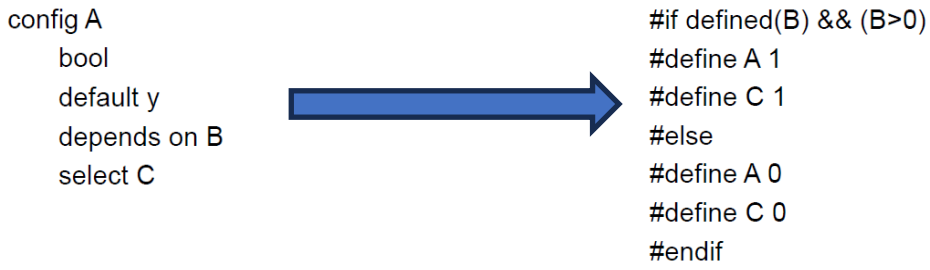
- 3rd Party modules and libraries
- Support for Zephyr in 3rd party projects, for example: Jerryscript, Micropython, Iotivity





Zephyr Build and Test Frameworks

- West Build System
  - Zephyr includes the “West" tool for the base code repository pulling and modules update (west init, west update). The “West" tool integrates CMAKE command to build the application and program the execution file to the board (west build, west flash). Refer to West (Zephyr’s meta-tool) — Zephyr Project Documentation.
  - Zephyr uses Kconfig to configurate the kernel and subsystem at build time to adapt the specific application and board needs. Configuration options (often called symbols) are defined in Kconfig files and the output from Kconfig is a header file autoconf.h. The type of kconfig can be bool, int, hex or string. Different configurations can be dependent through "depends on" and "select" keyword. Refer to Configuration System (Kconfig) — Zephyr Project Documentation.



- Ztest and Twister Test Frameworks
  - Zephyr testing framework (Ztest) can be used to create a testsuite with multiple test functions to verify the architecture, board, peripheral driver API, etc. Ztest provides many assertion/assumption/expectation macros to identify the current test is passed, skipped or failed. Refer to Test Framework — Zephyr Project Documentation.
  - Twister scans for the set of test applications in the specific path and attempts to execute them. It can build the samples/testcases for the target boards and flash them to the board, and output test results. Tests are detected by the presence of a testcase.yaml or a sample.yaml files in the application's project directory. Refer to Test Runner (Twister) — Zephyr Project Documentation.

Primary Git Repos

- Zephyr main repo

https://github.com/zephyrproject-rtos/zephyr

- Ambiq HAL repo

https://github.com/zephyrproject-rtos/hal_ambiq

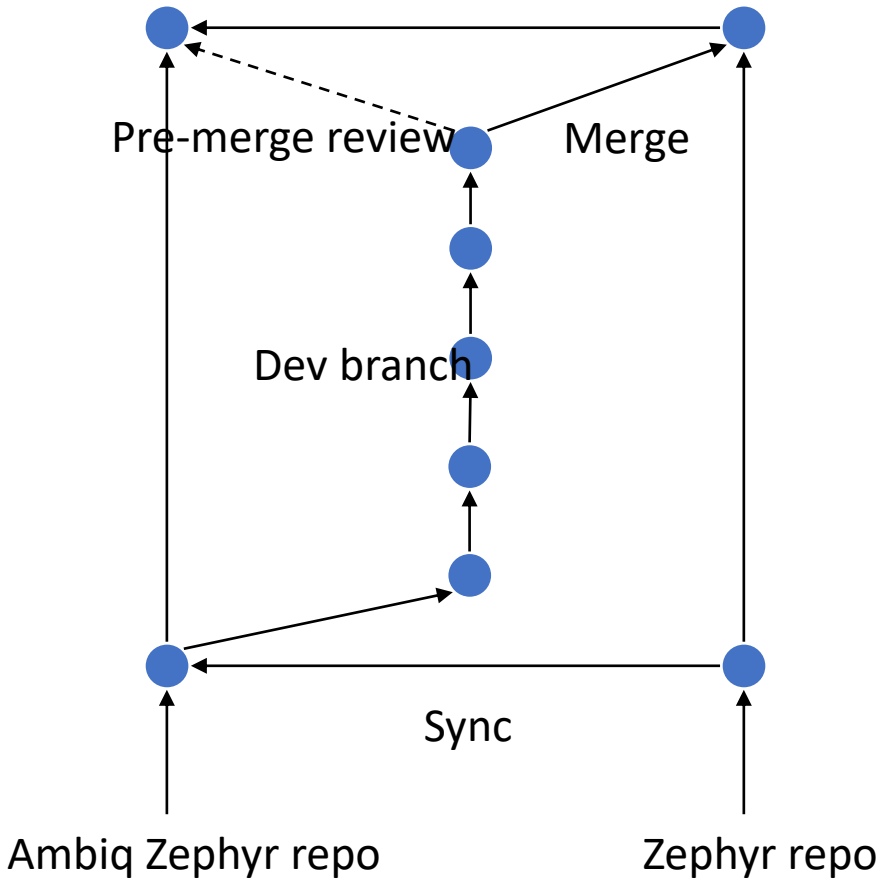
- Ambiq Zephyr development repo

https://github.com/AmbiqMicro/ambiqzephyr

- Ambiq HAL development repo

https://github.com/AmbiqMicro/ambiqhal_ambiq

Ambiq dev branch: ambiq-stable

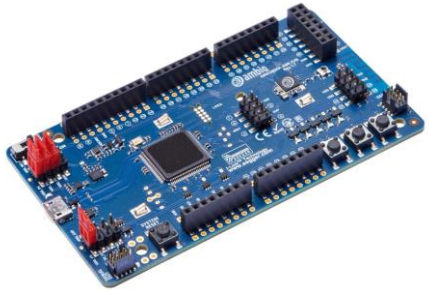


SoC and BSP Supported in Zephyr Ecosystem

Ambiq MCU:

- Apollo3 Blue
- Apollo3 Blue Plus
- Apollo4 Plus
- Apollo4 Blue Plus

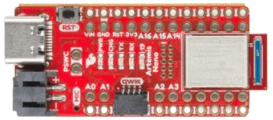
zephyr\soc\ambiq



Evaluation board:

- Apollo3 EVB (AMA3B1KK-KBR)
- Apollo3P EVB (AMA3B2KK-KBR)
- Apollo4P EVB (AMAP42KK-KBR)
- Apollo4P Blue KXR EVB (AMAP42KK-KXR)

zephyr\boards\ambiq



Third-party board:

- SparkFun RedBoard Artemis Nano
- RAK11720

More are on the way…

Drivers Supported In the Latest Ambiq Zephyr

	Apollo3 Blue	Apollo3 Blue Plus	Apollo4 Plus	Apollo4 Blue Plus
GPADC	✓	✓	✓	✓
I2C	✓	✓	✓	✓
BLE HCI	✓	✓	-	✓
SPI	✓	✓	✓	✓
MSPI	✓	✓	✓	✓
RTC	✓	✓	✓	✓
PDM				
I2S	-	-		
Timer	✓	✓	✓	✓
USB	-	-	✓	✓
UART	✓	✓	✓	✓
SDIO	-	-		
DSI	-	-		
Counter	✓	✓	✓	✓
HWINFO			✓	✓

Ambiq Components in Zephyr

ADC

- Apollo3 14-bit 10 channel ADC, Apollo4 12-bit 10 channel ADC
- Ambiq APIs

```
#ifdef CONFIG_ADC_ASYNC
#define ADC_AMBIQ_DRIVER_API(n)                                                \
    static const struct adc_driver_api adc_ambiq_driver_api_##n = {            \
        .channel_setup = adc_ambiq_channel_setup,                              \
        .read = adc_ambiq_read,                                                \
        .read_async = adc_ambiq_read_async,                                    \
        .ref_internal = DT_INST_PROP(n, internal_vref_mv),                     \
    };
#else
#define ADC_AMBIQ_DRIVER_API(n)                                                \
    static const struct adc_driver_api adc_ambiq_driver_api_##n = {            \
        .channel_setup = adc_ambiq_channel_setup,                              \
        .read = adc_ambiq_read,                                                \
        .ref_internal = DT_INST_PROP(n, internal_vref_mv),                     \
    };
#endif
```

```
adc0: adc@50010000 {
    reg = <0x50010000 0x400>;
    interrupts = <19 0>;
    interrupt-names = "ADC";
    channel-count = <10>;
    vref-mv = <1500>;
    status = "disabled";
    #io-channel-cells = <1>;
    ambiq,pwrcfg = <&pwrcfg 0x8 0x200>;
};
```

- Samples
  - zephyr\tests\drivers\adc
- Reference
  - Zephyr API Documentation: ADC driver APIs (zephyrproject.org)

Counter

Drivers Support

- Apollo3 CTIMER/Apollo4 TIMER
- 32bits, 8 for Apollo3, 16 for Apollo4
- Ambiq APIs

```
static const struct counter_driver_api counter_api = {
    .start = counter_ambiq_start,
    .stop = counter_ambiq_stop,
    .get_value = counter_ambiq_get_value,
    .set_alarm = counter_ambiq_set_alarm,
    .cancel_alarm = counter_ambiq_cancel_alarm,
    .set_top_value = counter_ambiq_set_top_value,
    .get_pending_int = counter_ambiq_get_pending_int,
    .get_top_value = counter_ambiq_get_top_value,
};
```

- Samples
  - zephyr\samples\drivers\counter\alarm
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__counter__interface.html

Flash Controller

Drivers Support

- 1MB flash memory for Apollo3, 2MB for Apollo3p and Apollo4p
- Ambiq APIs

```
static const struct flash_driver_api flash_ambiq_driver_api = {
    .read = flash_ambiq_read,
    .write = flash_ambiq_write,
    .erase = flash_ambiq_erase,
    .get_parameters = flash_ambiq_get_parameters,
#ifdef CONFIG_FLASH_PAGE_LAYOUT
    .page_layout = flash_ambiq_pages_layout,
#endif
};
```

- Samples
  - Zephyr\tests\drivers\flash
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__flash__interface.html

GPIO

Drivers Support

- 74 Pins for Apollo3, 128 Pins for Apollo4
- Pinctrl
- Ambiq APIs

```
static const struct gpio_driver_api ambiq_gpio_drv_api = {
    .pin_configure = ambiq_gpio_pin_configure,
#ifdef CONFIG_GPIO_GET_CONFIG
    .pin_get_config = ambiq_gpio_get_config,
#endif
    .port_get_raw = ambiq_gpio_port_get_raw,
    .port_set_masked_raw = ambiq_gpio_port_set_masked_raw,
    .port_set_bits_raw = ambiq_gpio_port_set_bits_raw,
    .port_clear_bits_raw = ambiq_gpio_port_clear_bits_raw,
    .port_toggle_bits = ambiq_gpio_port_toggle_bits,
    .pin_interrupt_configure = ambiq_gpio_pin_interrupt_configure,
    .manage_callback = ambiq_gpio_manage_callback,
#ifdef CONFIG_GPIO_GET_DIRECTION
    .port_get_direction = ambiq_gpio_port_get_direction,
#endif
};
```

- Samples
  - zephyr\samples\basic\blinky
  - zephyr\samples\basic\button
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__gpio__interface.html

RTC

Drivers Support

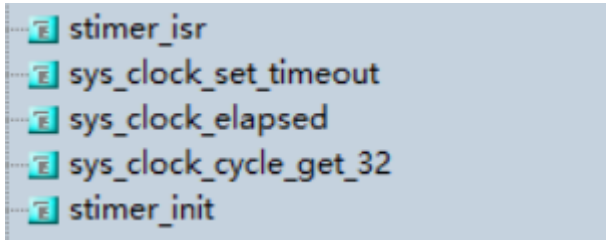
- RTC
  - AP3 has rollover capability in HW, AP4 does not have rollover capability in HW (see ERR129)
- Ambiq APIs

```
static const struct rtc_driver_api ambiq_rtc_driver_api = {
    .set_time = ambiq_rtc_set_time,
    .get_time = ambiq_rtc_get_time,
    /* RTC_UPDATE not supported */
#ifdef CONFIG_RTC_ALARM
    .alarm_get_supported_fields = ambiq_rtc_alarm_get_supported_fields,
    .alarm_set_time = ambiq_rtc_alarm_set_time,
    .alarm_get_time = ambiq_rtc_alarm_get_time,
    .alarm_is_pending = ambiq_rtc_alarm_is_pending,
    .alarm_set_callback = ambiq_rtc_alarm_set_callback,
#endif
};
```

- Samples
  - Zephyr\tests\drivers\rtc_api
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__rtc__interface.html

Timer

- Ambiq STIMER
- Ambiq APIs




```
&stimer0 {
    clk-source = <3>;
};
```

- Samples
  - zephyr\tests\kernel\timer
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__timer__apis.html

Watchdog

Drivers Support

- Ambiq APIs

```
static const struct wdt_driver_api wdt_ambiq_driver_api = {
    .setup = wdt_ambiq_setup,
    .disable = wdt_ambiq_disable,
    .install_timeout = wdt_ambiq_install_timeout,
    .feed = wdt_ambiq_feed,
};
```

- Samples
  - zephyr\samples\drivers\watchdog
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__watchdog__interface.html

I2C Controller

Drivers Support

- Ambiq IOM
- 6 instances for Apollo3, 8 instances for Apollo4
- Ambiq APIs

```
static const struct i2c_driver_api i2c_ambiq_driver_api = {
    .configure = i2c_ambiq_configure,
    .transfer = i2c_ambiq_transfer,
};
```

- DMA
  - CONFIG_I2C_AMBIQ_DMA
  - CONFIG_I2C_DMA_TCB_BUFFER_SIZE
- Samples
  - Display samples with touch screen (need apollo display shield cards)
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__i2c__interface.html

SPI Controller

- Ambiq IOM
- 6 instances for Apollo3, 8 instances for Apollo4
- Ambiq APIs

```
static const struct spi_driver_api spi_ambiq_driver_api = {
    .transceive = spi_ambiq_transceive,
    .release = spi_ambiq_release,
};
```

```
&spi0 {
    compatible = "ambiq,spi";
    pinctrl-0 = <&spi0_default>;
    pinctrl-names = "default";
    cs-gpios = <&gpio0_31 11 GPIO_ACTIVE_LOW>;
    clock-frequency = <DT_FREQ_M(1)>;
    status = "okay";
};
```

- DMA
  - CONFIG_SPI_AMBIQ_DMA
  - CONFIG_SPI_DMA_TCB_BUFFER_SIZE
- Samples
  - Samples with external Flash/PSRAM (need apollo memory shield cards)
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__spi__interface.html

SPI Device

- Ambiq IOS
- Half duplex, no DMA
- Ambiq APIs

```
static const struct spi_driver_api spi_ambiq_driver_api = {
    .transceive = spi_ambiq_transceive,
    .release = spi_ambiq_release,
};
```

```
dut_spis: &spid0 {
    compatible = "ambiq,spid";
    pinctrl-0 = <&spid0_default>;
    pinctrl-names = "default";
    status = "okay";
    int-gpios = <&gpio0_31 10 GPIO_ACTIVE_HIGH>;
};
```

HWINFO API

- Implements retrieving device ID
- Implements retrieving reset cause
  - Ambiq parts cannot "clear" reset cause
- Samples

```
ssize_t z_impl_hwinfo_get_device_id(uint8_t *buffer, size_t length);
int z_impl_hwinfo_get_reset_cause(uint32_t *cause);
int z_impl_hwinfo_clear_reset_cause(void);
int z_impl_hwinfo_get_supported_reset_cause(uint32_t *supported);
```

MSPI

- MSPI(Multi-bit SPI) up to octal for Apollo3 and Apollo3p, hex for Apollo4p
- 1 instance for Apollo3, 3 instances for Apollo3p and Apollo4p
- Ambiq APIs

```
static struct mspi_driver_api mspi_ambiq_driver_api = {
    .config              = mspi_ambiq_config,
    .dev_config          = mspi_ambiq_dev_config,
    .xip_config          = mspi_ambiq_xip_config,
    .scramble_config     = mspi_ambiq_scramble_config,
    .timing_config       = mspi_ambiq_timing_config,
    .get_channel_status  = mspi_ambiq_get_channel_status,
    .register_callback   = mspi_ambiq_register_callback,
    .transceive          = mspi_ambiq_transceive,
};
```

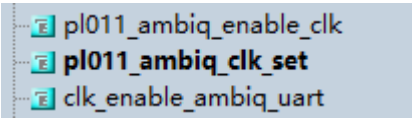
- Transfer Modes
  - PIO
  - DMA
  - XIP
- Samples - (need ambiq memory shield cards)
  - zephyr\samples\drivers\mspi
  - zephyr\samples\drivers\memc
  - zephyr\tests\drivers\mspi
- Reference
  - https://docs.zephyrproject.org/latest/doxygen/html/group__mspi__interface.html

```
&mspi1 {

    pinctrl-0 = <&mspi1_default>;
    pinctrl-1 = <&mspi1_sleep>;
    pinctrl-2 = <&mspi1_psram>;
    pinctrl-3 = <&mspi1_flash>;
    pinctrl-names = "default","sleep","psram","flash";
    status = "okay";

    ce-gpios = <&gpio64_95 5 GPIO_ACTIVE_LOW>,
           <&gpio32_63 18 GPIO_ACTIVE_LOW>;

    cmdq-buffer-location = ".mspi_buff";
    cmdq-buffer-size = <256>;

    aps64041: aps64041@0 {
        compatible = "ambiq,mspi-device", "mspi-aps64041";
        size = <DT_SIZE_M(64)>;
        reg = <0>;
        status = "disabled";
        mspi-max-frequency = <48000000>;
        mspi-io-mode = "MSPI_IO_MODE_QUAD";
        mspi-data-rate = "MSPI_DATA_RATE_SINGLE";
        mspi-hardware-ce-num = <0>;
        read-command = <0xEB>;
        write-command = <0x38>;
        command-length = "INSTR_1_BYTE";
        address-length = "ADDR_3_BYTE";
        rx-dummy = <6>;
        tx-dummy = <0>;
        xip-config = <1 0 0 0>;
        ce-break-config = <1024 3>;
        ambiq,timing-config-mask = <3>;
        ambiq,timing-config = <0 6 0 0 0 0 0 0>;
    };
```

UART

Drivers Support

- 2 instances for Apollo3, 4 instances for Apollo4
- Use general ARM PL011 UART driver
- UART CONSOLE
- Ambiq APIs



- Samples
  - zephyr\tests\drivers\uart
  - zephyr\tests\drivers\console
- Reference
  - https://docs.zephyrproject.org/latest/build/dts/api/bindings/serial/arm,pl011.html

USB

- 1 USB device instance for Apollo4P, unavailable for Apollo3
- USB is supported through the newer USB Device Controller (UDC) driver API. Older USB Device Core (DC) driver API is not supported.
- Numbers of endpoints available excluding CONTROL endpoints:
  - 5 IN endpoints
  - 5 OUT endpoints
- Transfer mode: PIO
- Samples
  - zephyr\samples\subsys\usb\cdc_acm
  - zephyr\samples\subsys\usb\console
  - zephyr\samples\subsys\usb\hid-keyboard
  - zephyr\samples\subsys\usb\hid-mouse
  - zephyr\samples\subsys\usb\mass
  - zephyr\samples\subsys\usb\webusb-next

Make sure vddusb control GPIO are updated accordingly in board's dts for custom boards.

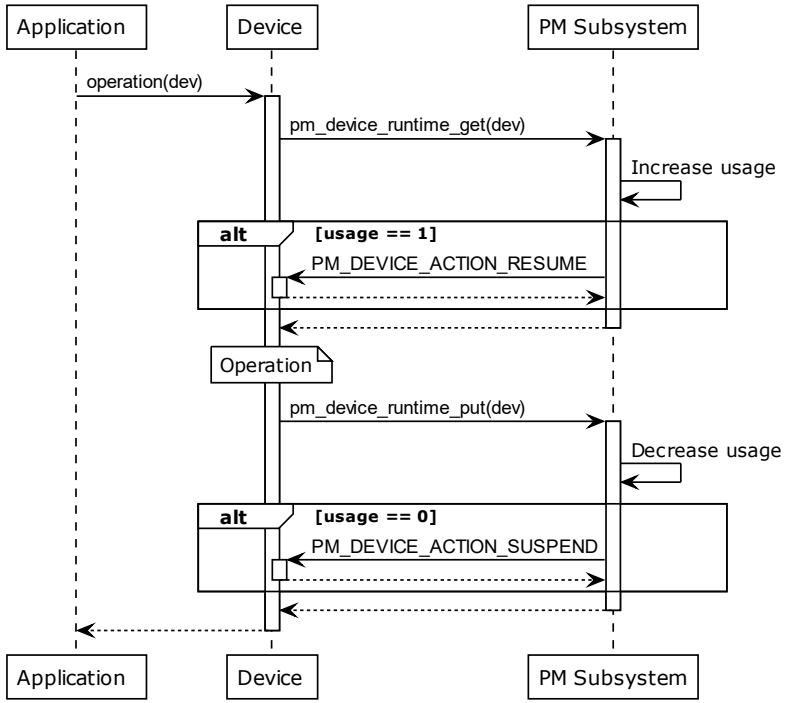
```
zephyr_udc0: &usb {
    vddusb33-gpios = <&gpio0_31 13 (GPIO_PULL_UP)>;
    vddusb0p9-gpios = <&gpio0_31 15 (GPIO_PULL_UP)>;
    status = "okay";
};
```

Power Management

- Zephyr supports two types of device power management
  - Device Runtime Power Management
  - System Power Management

- Device run time power management
  - Device power state is controlled by device driver, independent of the system power status.
  - Saving more power: device can be suspended even when the CPU is active.
  - Currently supported drivers: UART, SPI, I2C, MSPI, ADC

- Reference
  - https://docs.zephyrproject.org/latest/services/pm/device_runtime.html




```c
static int mydev_pm_action(...)
{
    switch (action) {
    case PM_DEVICE_ACTION_SUSPEND:
        /* Preserve register to SRAM and
         *power off device */
        Preserve_Register_value_to_RAM();
        Power_off_device();
        break;
    case PM_DEVICE_ACTION_RESUME:
        /* Power on device and restore register
         * from previously saved values*/
        Power_on_device();
        Restore_Register_from_RAM();
        break;
    default:
        return -ENOTSUP;
    }

    return 0;
}
```

Power Management

- System power management
  - **PM_STATE_SUSPEND_TO_IDLE**
    - Entered after the scheduled system idle time > 105us but < 2125us
    - Cotex-M4: normal sleep
    - Cache: active
    - SRAM: active
    - Flash: standby
  - **PM_STATE_SUSPEND_TO_RAM**
    - Entered after the scheduled system idle time >= 2125us
    - Cotex-M4: deep sleep
    - Cache: power off
    - SRAM: retention
    - Flash: power off
  - Reference

https://docs.zephyrproject.org/latest/services/pm/system.html

ambiq_apollo3_blue.dtsi

```
cpus {
    #address-cells = <1>;
    #size-cells = <0>;

    cpu0: cpu@0 {
        compatible = "arm,cortex-m4f";
        reg = <0>;
        cpu-power-states = <&idle &suspend_to_ram>;
    };

    power-states {
        idle: idle {
            compatible = "zephyr,power-state";
            power-state-name = "suspend-to-idle";
            /* As Apollo3blue datasheet, run_to_sleep and sleep_to_run transition
             * time are both lower than 1us, but considering the software overhead
             * we set a bigger value.
             */
            min-residency-us = <100>;
            exit-latency-us = <5>;
        };

        suspend_to_ram: suspend_to_ram {
            compatible = "zephyr,power-state";
            power-state-name = "suspend-to-ram";
            /* As Apollo3blue datasheet, run_to_deepsleep transition time is lower than
             * 1us and deepsleep_to_run transition time is about 25us,
             * but considering the software overhead, we set a bigger value.
             */
            min-residency-us = <2000>;
            exit-latency-us = <125>;
        };
    };
};
```

BLE Support

Zephyr Bluetooth Feature

Zephyr comes integrated with a feature-rich and highly configurable Bluetooth stack (Bluetooth — Zephyr Project Documentation).

- Bluetooth v5.3 compliant
  - Portable to all architectures supported by Zephyr
  - Support for all combinations of Host and Controller builds
    - Controller-only (HCI) over UART, SPI, USB and IPC physical transports
    - **Host-only over UART, SPI, and IPC**
    - Combined (Host + Controller)
- Bluetooth-SIG qualified (Bluetooth Qualification — Zephyr Project Documentation):
- Bluetooth Low Energy Controller support (LE Link Layer)
- Bluetooth Host support
  - GAP, GATT
  - Pairing, NVS
  - Mesh
  - **Clean HCI driver abstraction**
- LE Audio

BLE Support

The Apollo Blue family SoC includes a low power Bluetooth low energy subsystem, which contains a 2.4 GHz RF transceiver, modem, baseband, 32-bit processor and Host Controller Interface (HCI) to the host.

The Apollo Blue family SoC BLEIF module is used to interface with the embedded BLE Core module and supports read and write transactions to the BLE Core. The transactions are based on SPI bus, which are wired internally.

- **Driver Adaption:**
  - **SPI**: implement the IOM based SPI driver for Apollo4 Blue series or one independent BLEIF ((different with IOM) based SPI driver (*spi_driver_api*) for Apollo3 Blue series;
  - **HCI**: on top of SPI-IOM and SPI-BLEIF driver we finish the Zephyr standard HCI driver adaption for Apollo Blue to make it run the Zephyr BLE host stack and talk to the embedded BLE controller via implemented HCI driver API (*bt_hci_driver*).
  - **Flash controller**: implement flash driver (*flash_driver_api*) to support allocating some internal flash memory for data storage (the Bluetooth applications may need the NVS for bonding data storage).

In Bluetooth applications, it is most likely that the user just needs to call the API defined in the Zephyr BLE host stack (e.g., bt_enable(), bt_le_adv_start() to initialize the Bluetooth, start BLE advertising, etc.), no need to call HCI driver API.

- **Samples:**
  - *zephyr/samples/bluetooth*. User can run *peripheral_hr* sample to start Bluetooth development.

Getting Started with Zephyr

Get Started with Ambiq SoC

- **Select OS (Ubuntu, macOS, Windows) and install the main dependencies (CMake, Python, Devicetree compiler)**
  - The installation method for these tools is different between different OS. Please refer to section "Install dependencies" of Getting Started Guide — Zephyr Project Documentation.
- **Get Zephyr code via west tool and install Python dependencies**
  - Create one empty folder and "cd" to this path
  - Execute "west init -m https://github.com/AmbiqMicro/ambiqzephyr --mr main"
  - Execute "west update" to get/update the Zephyr modules
  - pip install –r YOUR_ZEPHYRPROJECT_PATH\zephyr\script\requirement.txt
- **Install Zephyr SDK which contains the toolchain and other programs required to build the Zephyr applications.**
  - The installation method of Zephyr SDK is different between different OS. Please refer to Section "Install Zephyr SDK" of Getting Started Guide — Zephyr Project Documentation.
  - Make sure **ZEPHYR_TOOLCHAIN_VARIANT** or **ZEPHYR_SDK_INSTALL_DIR** environment variables are already set. If not, set them manually.

Get Started with Ambiq SoC

- **Add Jlink.exe to the PATH environment variable of your PC, build and flash the sample**
  - Execute “west build -b <board_name> <sample_name> -p” to build the application
  - Execute “west flash” to program the binary to your plugged Apollo3/4 EVB.
- **Debugging**
  - For example, open Ozone, select the corresponding device, for example, "AMA3B1KK-KBR" for supported apollo3_evb, use SWD to connect, choose the ELF path, e.g., zephyr\build\zephyr\zephyr.elf, then download the binary and start debugging. VS Code can be used for Zephyr application debug as well.

Appendix

Appendix

- Zephyr development documents
  - https://docs.zephyrproject.org/latest/develop/index.html
- Renode designer
  - https://designer.antmicro.com/hardware/vendors/ambiq
- Ambiq SoC documents
  - https://contentportal.ambiq.com/soc

ambiq

Thank You!