Enabling Zephyr on Your Hardware Platform

Diego Sueiro, Sepura / Embarcados
www.embarcados.com.br

OpenIoT Summit Europe 2018

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline

Preamble

- Source code examples based on master branch 1ec4b68;
- Some sources were stripped to fit on the screen;
- All examples based on the support for Zephyr running on the ARM Cortex M4 core embedded in the i.MX7 processor;
- This presentation will not cover how to add a new CPU core architecture support. But a good documentation on how to achieve this can be find here;
- Not all hardware aspects will be covered;

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

Adding new drivers

Adding a new board

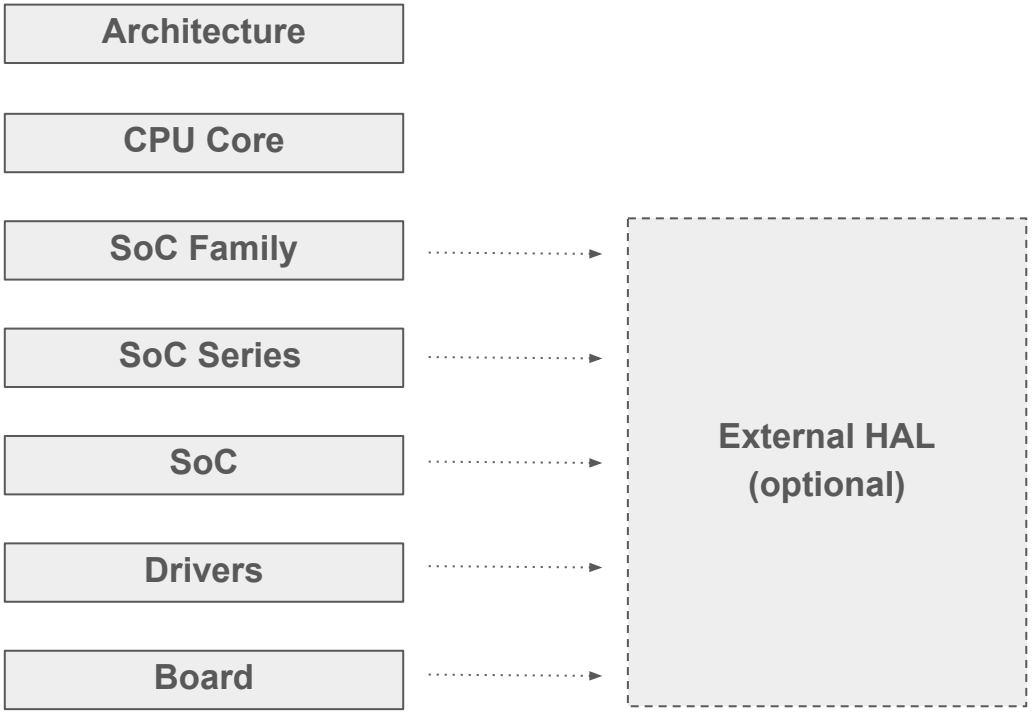
Debugging tips

Hardware support checklist

Contributing to mainline

Hardware support implementation in Zephyr

Hardware Configuration Hierarchy (bottom to top):



Hardware support implementation in Zephyr

Hardware Configuration Hierarchy:

Architecture: arc, arm, nios2, posix, riscv32, x86 and xtensa

CPU Core (a.k.a architecture):

- Implements: early boot sequence, interrupt and exception handling, thread context switching, thread creation and termination, CPU idling/power management, fault management, linker scripts and toolchains;
- Examples: ARCV2, CORTEX_M0, CORTEX_M0PLUS, CORTEX_M4, CORTEX_M7, CORTEX_M23, CORTEX_M33, NIOS2_GEN2, ATOM, MINUTEIA and APOLLO_LAKE.

SoC Family:

- Represents a single SoC type that can have more than one variations in terms of peripherals and features;
- Examples: KINETIS, IMX, SAM, SAM0, NRF, EXX32, LPC, TISIMPLELINK, STM32 and QUARK.

SoC Series:

- Represents the specific peripherals and features for the SoC family variations;
- Examples: KINETIS_K6X, KINETIS_KWX, KINETIS_KL2X, IMX_RT, IMX7_M4, IMX6_M4, NRF51X, NRF52X, EFM32WG, EFR32FG1P.

Hardware support implementation in Zephyr

Hardware Configuration Hierarchy (cont):

SoC:

- The actual SoC that is “soldered” in the hardware platform and its configuration;
- Examples: MKL25Z32VFM4, MCIMX7D5EVM10SC, SAMD20E14, EFM32WG990F256, LPC54114J256BD64.

Drivers:

- Include device model responsible for configuring and initialize drivers. Each driver follows a device model API and a specific driver type API;
- Examples: interrupt controller, timer, serial communications (UART, I2C etc) and random number generator.

Board:

- Includes a SoC and it’s associated peripherals and features including external components and devices;
- Examples: NRF51_BLENANO, NUCLEO_F103RB, COLIBRI_IMX7D_M4, 96B_CARBON, MIMXRT1050_EVK, HEXIWEAR_K64, QUARK_SE_C1000_BLE, CC2650_SENSORTAG, ADAFRUIT_TRINKET_M0 (more than 100 available).

Hardware support implementation in Zephyr

- Top level hardware configurations are defined via Kconfigs and the final processing results located in the files:

  `build/<board>/zephyr/.config`
  `build/<board>/zephyr/include/generated/autoconf.h`

- Low level hardware specific configurations are defined via device tree and the final processing results located in the files:

  `build/<board>/zephyr/include/generated/generated_dts_board.conf`
  `build/<board>/zephyr/include/generated/generated_dts_board.h`

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

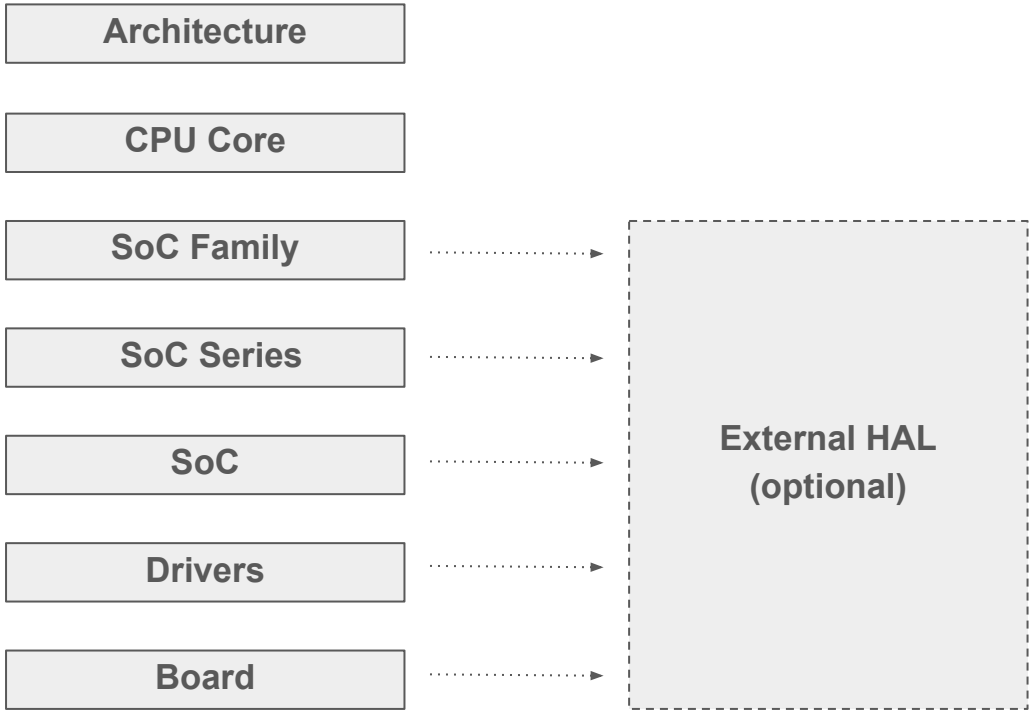
Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline



Adding a new HAL

- Added to support SoC, board and drivers implementations;
- Low level libraries mostly implemented by the SoC vendor to interface and configure the hardware;
- Different types of HAL, pros and cons covered at Maureen Helm's presentation: Using SoC Vendor HALs in the Zephyr Project - Video, slides;
- Needs to be approved by the Zephyr Technical Steering Committee for non-Apache 2.0;
- Located at: `ext/hal/<vendor>/<lib_name>/` ;

Adding a new HAL

- Just bug fixing modifications are allowed in these source/headers files;
- No standard coding style and directory structure;
- Almost all ARM devices follow the CMSIS standard headers for registers manipulation;
- Enabled with a config option, example: `CONFIG_HAS_IMX_HAL` ;
- Has a set of `Kconfig` and `CMakeLists.txt` files to determine what to include and compile;

Adding a new HAL

Example: i.XM7 ARM Cortex M4 core from NXP FreeRTOS BSP locate at `ext/hal/nxp/imx/` with the following structure:

```
.
├── CMakeLists.txt
├── devices
│   ├── MCIMX6X
│   │   ├── <...>
│   └── MCIMX7D
│       ├── <...>.c
│       ├── <...>.h
│       ├── CMakeLists.txt
│       ├── device_imx.h
│       └── MCIMX7D_M4.h
├── drivers
│   ├── <...>.c
│   ├── <...>.h
│   ├── CMakeLists.txt
├── Kconfig
└── README
```

Adding a new HAL

Example: i.XM7 ARM Cortex M4 (cont)

```
ext/hal/nxp/imx/README:
iMX7D and MX6SX Port
####################

Origin:
   <...>

Status:
   <...>

Purpose:
   <...>

Description:
   <...>

<...>
```

Follows the structure defined Contributing non-Apache 2.0 licensed components.

Adding a new HAL

Example: i.XM7 ARM Cortex M4 (cont)

ext/hal/nxp/imx/Kconfig:

```
config HAS_IMX_HAL
        bool
        select HAS_CMSIS
        depends on  SOC_FAMILY_IMX

if HAS_IMX_HAL

config HAS_IMX_RDC
        bool
        help
          Set if the RDC module is present in the
          SoC.

config HAS_IMX_CCM
        bool
        help
          Set if the CCM module is present in the
          SoC.
```

ext/hal/nxp/imx/Kconfig (cont):

```
config HAS_IMX_GPIO
        bool
        help
          Set if the GPIO module is present in the
          SoC.

config HAS_IMX_I2C
        bool
        help
          Set if the I2C module is present in the
          SoC.

endif # HAS_IMX_HAL
```

Adding a new HAL

Example: i.XM7 ARM Cortex M4 (cont)

ext/hal/nxp/imx/CMakeLists.txt:

```
# Translate the SoC name and part number into the imx device and cpu
# name respectively.
string(TOUPPER ${CONFIG_SOC} IMX_DEVICE)

zephyr_include_directories(devices/${IMX_DEVICE})

# Build imx drivers and utilities that can be used for multiple SoC's.
add_subdirectory(drivers)
add_subdirectory(devices/${IMX_DEVICE})
```

Adding a new HAL

Example: Toradex Colibri iMX7 Dual

HAL related generated configs

build/colibri_imx7d_m4/zephyr/.config:

```
CONFIG_HAS_IMX_HAL=y
CONFIG_HAS_IMX_GPIO=y
CONFIG_HAS_IMX_I2C=y
```

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

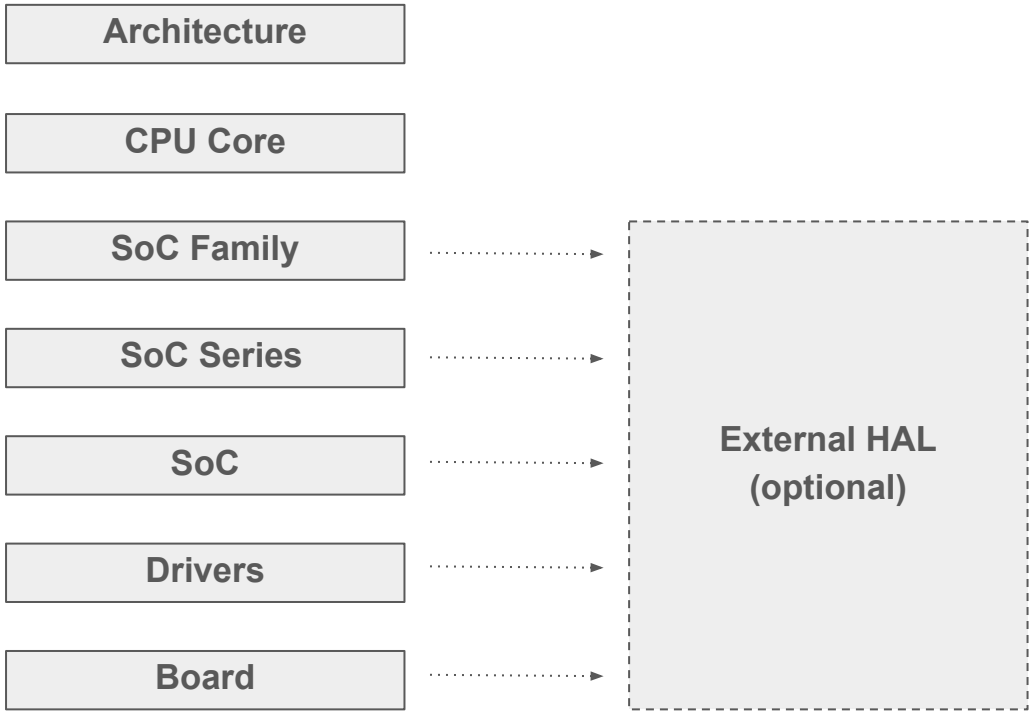
Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline



Adding a new SoC

- Defines the `SOC_FAMILY`, `SOC_SERIES`, `SOC` and `SOC_PART_NUMBER` configs;
- Located at: `soc/<architecture>/<soc_family>/<soc_series>/` ;
- SoC initialization like clocks, memories, cache, chip erratas, watchdog etc in a `soc.c` file;
- Called in the system initialization process with the level `PRE_KERNEL_1` and priority `0` ;
- Provides a `soc.h` header which will be often included by the board and drivers sources;
- Can extend functionalities not provided by the vendor HAL;
- Contains a set of Kconfig files, linker definitions, and device tree fixups.

Adding a new SoC

- Default Architecture, `SOC_FAMILY`, `SOC_SERIES` configs are selected in `boards/<architecture>/<board_name>/<board_name>_defconfig`.
  Example: Toradex Colibri iMX7 Dual

```
boards/arm/colibri_imx7d_m4/colibri_imx7d_m4_defconfig:
        CONFIG_ARM=y
        CONFIG_SOC_FAMILY_IMX=y
        CONFIG_SOC_SERIES_IMX7_M4=y
        <...>
```

- These default configs will dictate what Kconfigs will be sourced and which `CONFIG_` entries will be selected and generated for the SoC presented on the hardware platform;
- Has a dtsi defining peripherals and features properties presented in the SoC and is located at `dts/<architecture>/<vendor>/<vendor>_<soc_name>.dtsi` ;
- May have a `dts.fixup` file that contain mappings from existing Kconfig options to the actual underlying DTS derived configuration `#defines` .

Adding a new SoC

dtsi defining peripherals and features properties presented in the SoC
Example: i.XM7 ARM Cortex M4 locate dtsi at `dts/arm/nxp/`

dts/arm/nxp/nxp_imx7d_m4.dtsi:

```
#include <arm/armv7-m.dtsi>
#include <dt-bindings/gpio/gpio.h>
#include <dt-bindings/i2c/i2c.h>
#include <dt-bindings/rdc/imx_rdc.h>

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

                cpu@0 {
                        device_type = "cpu";
                        compatible = "arm,cortex-m4";
                        reg = <0>;
                };
        };
```

dts/arm/nxp/nxp_imx7d_m4.dtsi (cont):

```
        soc {
                <...>

                tcml_code: code@1fff8000 {
                        compatible = "nxp,imx-code-bus";
                        reg = <0x1fff8000 0x8000>;
                        label = "TCML CODE";
                };

                tcmu_sys: memory@20000000 {
                        device_type = "memory";
                        compatible = "nxp,imx-sys-bus";
                        reg = <0x20000000 0x8000>;
                        label = "TCMU SYSTEM";
                };

                <...>
```

Adding a new SoC

dtsi defining peripherals and features properties presented in the SoC
Example: i.XM7 ARM Cortex M4 locate dtsi at `dts/arm/nxp/`

dts/arm/nxp/nxp_imx7d_m4.dtsi (cont):

```
	<...>

	gpio7: gpio@30260000 {
		compatible = "nxp,imx-gpio";
		reg = <0x30260000 0x10000>;
		interrupts = <76 0>, <77 0>;
		label = "GPIO_7";
		rdc = <(RDC_DOMAIN_PERM(A7_DOMAIN_ID, RDC_DOMAIN_PERM_RW)|\
			RDC_DOMAIN_PERM(M4_DOMAIN_ID, RDC_DOMAIN_PERM_RW))>;
		gpio-controller;
		#gpio-cells = <2>;
		status = "disabled";
	};
	<...>

	&nvic {
		 arm,num-irq-priority-bits = <4>;
	};
```

Adding a new SoC

`dts.fixup` files contain mappings from existing Kconfig options to the actual underlying DTS derived configuration `#defines`.
Example: i.XM7 ARM Cortex M4

soc/arm/nxp_imx/mcimx7_m4/dts.fixup:

```
<...>
#define CONFIG_NUM_IRQ_PRIO_BITS              ARM_V7M_NVIC_E000E100_ARM_NUM_IRQ_PRIORITY_BITS
<...>
#define CONFIG_GPIO_IMX_PORT_7_NAME           NXP_IMX_GPIO_30260000_LABEL
#define CONFIG_GPIO_IMX_PORT_7_BASE_ADDRESS   NXP_IMX_GPIO_30260000_BASE_ADDRESS
#define CONFIG_GPIO_IMX_PORT_7_IRQ_0          NXP_IMX_GPIO_30260000_IRQ_0
#define CONFIG_GPIO_IMX_PORT_7_IRQ_0_PRI      NXP_IMX_GPIO_30260000_IRQ_0_PRIORITY
#define CONFIG_GPIO_IMX_PORT_7_IRQ_1          NXP_IMX_GPIO_30260000_IRQ_1
#define CONFIG_GPIO_IMX_PORT_7_IRQ_1_PRI      NXP_IMX_GPIO_30260000_IRQ_1_PRIORITY
<...>
#define CONFIG_UART_IMX_UART_2_NAME           NXP_IMX_UART_30890000_LABEL
#define CONFIG_UART_IMX_UART_2_BASE_ADDRESS   NXP_IMX_UART_30890000_BASE_ADDRESS
#define CONFIG_UART_IMX_UART_2_BAUD_RATE      NXP_IMX_UART_30890000_CURRENT_SPEED
#define CONFIG_UART_IMX_UART_2_IRQ_NUM        NXP_IMX_UART_30890000_IRQ_0
#define CONFIG_UART_IMX_UART_2_IRQ_PRI        NXP_IMX_UART_30890000_IRQ_0_PRIORITY
#define CONFIG_UART_IMX_UART_2_MODEM_MODE     NXP_IMX_UART_30890000_MODEM_MODE
<...>
```

Adding a new SoC

Example: i.XM7 ARM Cortex M4 SoC specific source code at `soc/arm/nxp_imx/mcimx7_m4/` with the following structure:

```
soc/arm/nxp_imx/
        .
        ├── CMakeLists.txt
        ├── Kconfig
        ├── Kconfig.defconfig
        ├── Kconfig.soc
        └── mcimx7_m4
            ├── CMakeLists.txt
            ├── dts.fixup
            ├── Kconfig.defconfig.mcimx7_m4
            ├── Kconfig.defconfig.series
            ├── Kconfig.series
            ├── Kconfig.soc
            ├── linker.ld
            ├── soc.c
            ├── soc_clk_freq.c
            ├── soc_clk_freq.h
            └── soc.h
```

Adding a new SoC

Kconfig processing order when `cmake -DBOARD=<BOARD_NAME> ../..` command is issued:

```
[00] $(BOARD_DIR)/<BOARD_NAME>_defconfig
[01] Kconfig -> [02]
[02] Kconfig.zephyr -> [03] | [04] | [05] | [08] | [11] | [17]
[03] $(BOARD_DIR)/Kconfig.defconfig
[04] boards/shields/*/Kconfig.defconfig
[05] $(SOC_DIR)/$(ARCH)/*/Kconfig.defconfig -> [06]
[06] $(SOC_DIR)/$(ARCH)/<SOC_FAMILY>/*/Kconfig.defconfig.series -> [07]
[07] $(SOC_DIR)/$(ARCH)/<SOC_FAMILY>/<SOC_SERIES>/Kconfig.defconfig.<SOC_SERIES>
[08] boards/Kconfig -> [09] | [10]
[09] $(BOARD_DIR)/Kconfig.board
[10] $(BOARD_DIR)/Kconfig
[11] $(SOC_DIR)/Kconfig -> [12] | [14] | [15]
[12] $(SOC_DIR)/$(ARCH)/*/Kconfig.soc -> [13]
[13] $(SOC_DIR)/$(ARCH)/<SOC_FAMILY>/*/Kconfig.series
[14] $(SOC_DIR)/$(ARCH)/Kconfig
[15] $(SOC_DIR)/$(ARCH)/*/Kconfig -> [16]
[16] $(SOC_DIR)/$(ARCH)/<SOC_FAMILY>/*/Kconfig.soc
[17] arch/Kconfig
```

Adding a new SoC

Kconfig processing order (cont)

Example: Toradex Colibri iMX7 Dual (`cmake -DBOARD=colibri_imx7d_m4 ../..`)

```
[00] boards/arm/colibri_imx7d_m4/colibri_imx7d_m4_defconfig
[01] Kconfig -> [02]
[02] Kconfig.zephyr -> [03] | [04] | [05] | [08] | [11]
[03] boards/arm/colibri_imx7d_m4/Kconfig.defconfig
[04] boards/shields/*/Kconfig.defconfig
[05] soc/arm/nxp_imx/Kconfig.defconfig -> [06]
[06] soc/arm/nxp_imx/mcimx7_m4/Kconfig.defconfig.series -> [07]
[07] soc/arm/nxp_imx/mcimx7_m4/Kconfig.defconfig.mcimx7_m4
[08] boards/Kconfig -> [09] | [10]
[09] boards/arm/colibri_imx7d_m4/Kconfig.board
[10] boards/arm/colibri_imx7d_m4/Kconfig
[11] soc/Kconfig -> [12] | [14] | [15]
[12] soc/arm/nxp_imx/Kconfig.soc -> [13]
[13] soc/arm/nxp_imx/mcimx7_m4/Kconfig.series
[14] soc/arm/Kconfig
[15] soc/arm/nxp_imx/Kconfig -> [16]
[16] soc/arm/nxp_imx/mcimx7_m4/Kconfig.soc
```

Adding a new SoC

SoC specific Kconfig files

Example: i.XM7 ARM Cortex M4

[00] boards/arm/colibri_imx7d_m4/colibri_imx7d_m4_defconfig
[01] Kconfig -> [02]
[02] Kconfig.zephyr -> [03] | [04] | [05] | [08] | [11]
[03] boards/arm/colibri_imx7d_m4/Kconfig.defconfig
[04] boards/shields/*/Kconfig.defconfig
[05] soc/arm/nxp_imx/Kconfig.defconfig -> [06]
[06] soc/arm/nxp_imx/mcimx7_m4/Kconfig.defconfig.series -> [07]
[07] soc/arm/nxp_imx/mcimx7_m4/Kconfig.defconfig.mcimx7_m4
[08] boards/Kconfig -> [09] | [10]
[09] boards/arm/colibri_imx7d_m4/Kconfig.board
~~[10] boards/arm/colibri_imx7d_m4/Kconfig~~
[11] soc/Kconfig -> [12] | [14] | [15]
[12] soc/arm/nxp_imx/Kconfig.soc -> [13]
[13] soc/arm/nxp_imx/mcimx7_m4/Kconfig.series
[14] soc/arm/Kconfig
[15] soc/arm/nxp_imx/Kconfig -> [16]
[16] soc/arm/nxp_imx/mcimx7_m4/Kconfig.soc

Adding a new SoC

Example: Toradex Colibri iMX7 Dual
SoC related generated configs

samples/subsys/shell/shell_module/build/colibri_imx7d_m4/zephyr/.config:

```
...
CONFIG_SOC="mcimx7d"
CONFIG_SOC_SERIES="mcimx7_m4"
CONFIG_NUM_IRQS=127
CONFIG_SYS_CLOCK_HW_CYCLES_PER_SEC=200000000
CONFIG_SOC_PART_NUMBER="MCIMX7D5EVM10SC"
...
CONFIG_CLOCK_CONTROL_IMX_CCM=y
CONFIG_GPIO_IMX=y
CONFIG_UART_IMX=y
CONFIG_SYS_CLOCK_TICKS_PER_SEC=1000
...
CONFIG_SOC_SERIES_IMX7_M4=y
CONFIG_SOC_FAMILY="nxp_imx"
CONFIG_SOC_FAMILY_IMX=y
CONFIG_SOC_MCIMX7_M4=y
CONFIG_SOC_PART_NUMBER_MCIMX7D5EVM10SC=y
CONFIG_SOC_PART_NUMBER_IMX7_M4="MCIMX7D5EVM10SC"
```

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

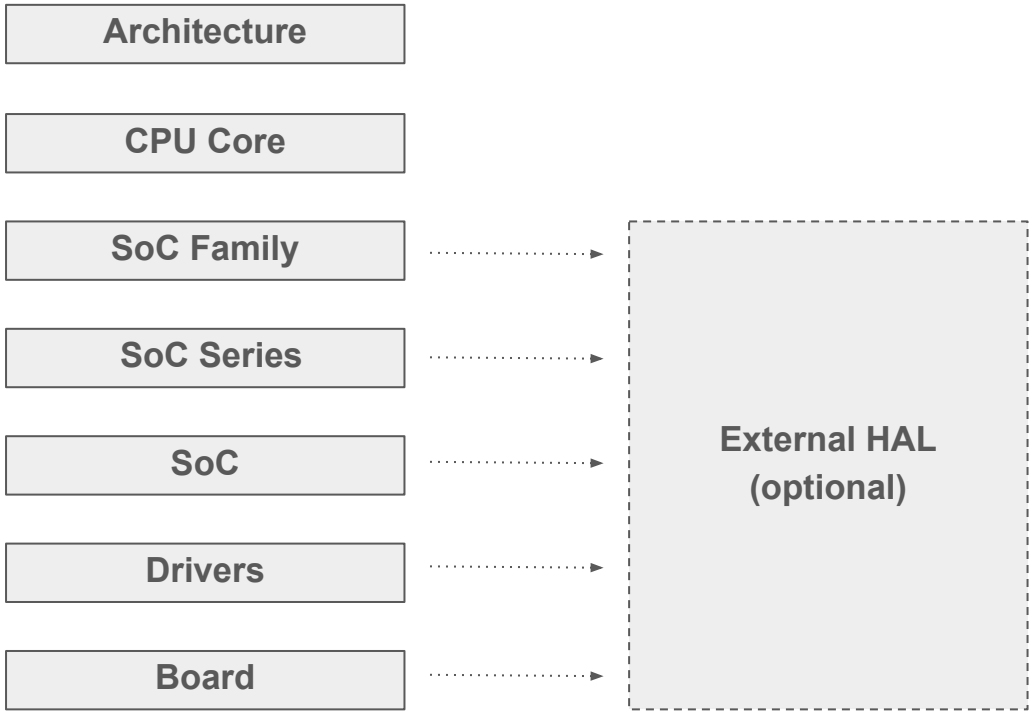
Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline



Adding a new Driver

- Provides interface to the hardware;
- Located at `drivers/<driver_type>/` ;
- Must implement the API exposed in `include/<driver_type>.h` ;
- One driver multiple instances;
- Selection and configuration done via Kconfigs and device tree;
- May use the vendor HAL (shim drivers);
- Initialization performed during the kernel boot.

Adding a new Driver

- Yaml file to describe the device tree nodes and properties;
- Device tree file to define driver properties and configurations;
- Good ramp up documentation available;
- Unfortunately we don’t have time to cover this topic in this presentation :-(

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

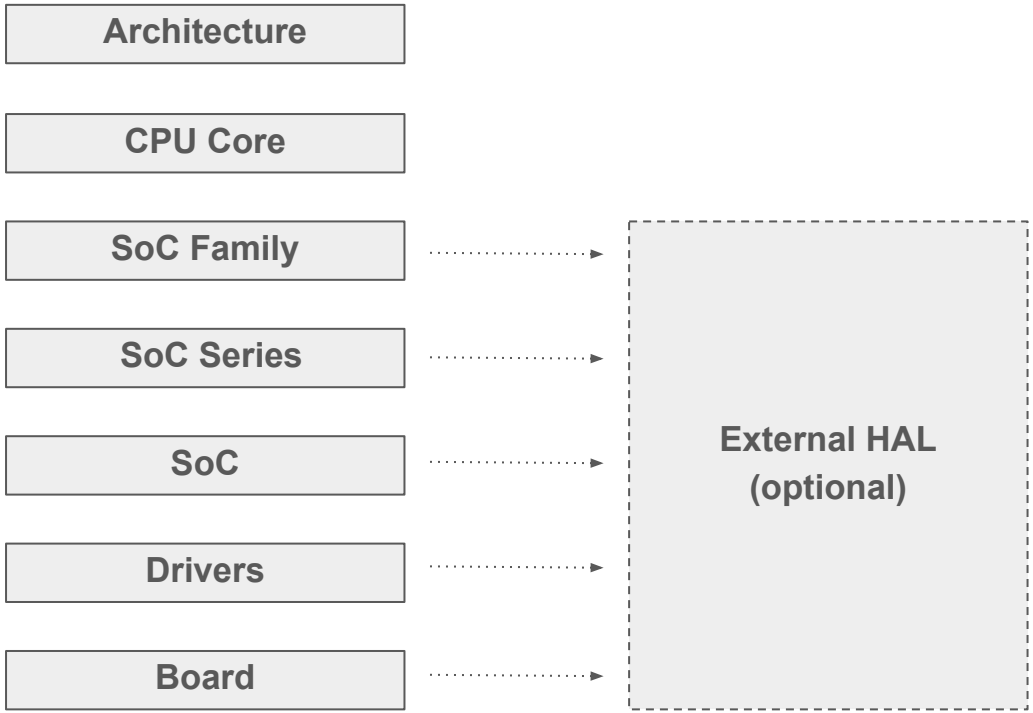
Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline



Adding a new Board

- Represents the application hardware platform;
- Located at `boards/<architecture>/<board_name>/` ;
- Extends the SoC and enable/disable its peripherals and functions and instantiate external devices via device tree (`<board_name>.dts`) and Kconfigs;
- Applies the pin muxing configuration;
- Contains a `board.h` to be used by the drivers;
- Contains a `<board_name>_defconfig` file to define which SoC and basic features and interfaces included;

Adding a new Board

- May set flash partitions layout in the `<board_name>.dts` file;
- May include a `dts.fixup` file which contain mappings from existing Kconfig options to the actual underlying DTS derived configuration `#defines` ;
- May include other source files to configure specific hardware and board features;
- May provide a `board.cmake` to instruct how to flash/debug ;
- Includes a `<board_name>.yaml` file to list the board properties: e.g. flash and ram sizes and toolchain used, etc;
- Must have documentation listing the supported features, interfaces etc.

Adding a new Board

Source code located at `boards/<architecture>/<board_name>/`
Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/:

```
.
├── board.h
├── CMakeLists.txt
├── colibri_imx7d_m4_defconfig
├── colibri_imx7d_m4.dts
├── colibri_imx7d_m4.yaml
├── doc
│   ├── colibri_imx7d_m4.rst
│   └── colibri_imx7d.png
├── Kconfig.board
├── Kconfig.defconfig
└── pinmux.c
```

Adding a new Board

Includes a `<board_name>.yaml` file to list the board properties: e.g. flash and ram sizes and toolchain used, etc
Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/colibri_imx7d_m4.yaml:

```
identifier: colibri_imx7d_m4
name: TORADEX Colibri IMX7D
type: mcu
arch: arm
ram: 32
flash: 32
toolchain:
  - zephyr
  - gnuarmemb
testing:
  ignore_tags:
    - net
    - bluetooth
```

Adding a new Board

Device tree `boards/<architecture>/<board_name>/<board_name>.dts` extending the SoC and setting external devices.
Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/colibri_imx7d_m4.dts:

```
/dts-v1/;

#include <nxp/nxp_imx7d_m4.dtsi>

/ {
        model = "TORADEX Colibri IMX7D board";
        compatible = "nxp,mcimx7d_m4";

        aliases {
                gpio-1 = &gpio1;
                gpio-2 = &gpio2;
                uart-2 = &uart2;
                led0   = &green_led;
                sw0    = &user_switch_1;
                i2c-4  = &i2c4;
                pwm-1  = &pwm1;
        };
```

boards/arm/colibri_imx7d_m4/colibri_imx7d_m4.dts (cont):

```
        chosen {
        #if defined(CONFIG_XIP)
                zephyr,flash = &tcml_code;
        #endif
                zephyr,sram = &tcmu_sys;
                zephyr,console = &uart2;
        };

        leds {
                compatible = "gpio-leds";
                green_led: led@0 {
                gpios = <&gpio1 2 GPIO_INT_ACTIVE_LOW>;
                label = "User LED1";
                };
        };
```

Adding a new Board

Device tree `boards/<architecture>/<board_name>/<board_name>.dts` extending the SoC and setting external devices.
Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/colibri_imx7d_m4.dts (cont):

```
                gpio_keys {
                        compatible = "gpio-keys";
                        user_switch_1: sw@0 {
                                gpios = <&gpio2 26
                                   GPIO_INT_ACTIVE_LOW>;
                                label = "User SW1";
                        };
                };
        };

        &uart2 {
                status = "ok";
                current-speed = <115200>;
                modem-mode = <64>;
        };
```

boards/arm/colibri_imx7d_m4/colibri_imx7d_m4.dts (cont):

```
        &gpio1 {
                status = "ok";
        };

        &gpio2 {
                status = "ok";
        };

        &i2c4 {
                status = "ok";
        };

        &pwm1 {
                status = "ok";
        };
```

Adding a new Board

`boards/<architecture>/<board_name>/Kconfig.board` file that basically defines the board config, list `SOC_SERIES` dependency and selects the `SOC_PART_NUMBER`

Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/Kconfig.board:

```
config BOARD_COLIBRI_IMX7D_M4
        bool "Toradex Colibri iMX7 Dual"
        depends on SOC_SERIES_IMX7_M4
        select SOC_PART_NUMBER_MCIMX7D5EVM10SC
```

Adding a new Board

`boards/<architecture>/<board_name>/Kconfig.defconfig` file with invisible symbols that selects hardware interfaces and features and sets its default values.
Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/Kconfig.defconfig:

```
if BOARD_COLIBRI_IMX7D_M4
config BOARD
        default "colibri_imx7d_m4"

if GPIO_IMX
config GPIO_IMX_PORT_1
        def_bool y
<..>
endif # GPIO_IMX

if UART_IMX
config UART_IMX_UART_2
        def_bool y
endif # UART_IMX
```

boards/arm/colibri_imx7d_m4/Kconfig.defconfig (cont):

```
if I2C_IMX
<...>
config I2C_4
        def_bool y
endif # I2C_IMX

if PWM_IMX
config PWM_1
        def_bool y
endif # PWM_IMX

endif # BOARD_COLIBRI_IMX7D_M4
```

Adding a new Board

`boards/<architecture>/<board_name>/<board_name>_defconfig` file with visible symbols that selects the architecture, SoC aspects, board config, top level interfaces and features.

Example: Toradex Colibri iMX7 Dual

boards/arm/colibri_imx7d_m4/colibri_imx7d_m4_defconfig:

```
CONFIG_ARM=y
CONFIG_SOC_FAMILY_IMX=y
CONFIG_SOC_SERIES_IMX7_M4=y
CONFIG_SOC_MCIMX7_M4=y
CONFIG_BOARD_COLIBRI_IMX7D_M4=y
CONFIG_CORTEX_M_SYSTICK=y
CONFIG_SERIAL_HAS_DRIVER=y
CONFIG_UART_CONSOLE=y
CONFIG_SERIAL=y
CONFIG_CONSOLE=y
CONFIG_CONSOLE_HAS_DRIVER=y
CONFIG_XIP=y
```

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline

Debugging tips

- Look at other source code reference (e.g. FreeRTOS) to understand what needs to be done to initialize the SoC;
- Try to print to UART (accessing the registers directly) in the SoC initialization to guarantee that the core is up and running;
- Implement the UART driver first, printk is life;
- Turn on the System Logging or Logger;
- Turn on asserts (`CONFIG_ASSERT`) to try to catch errors;
- Use a on-chip debugger (J-Link, ULINK etc).

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline

Hardware support checklist

- For a new HAL:
  - Add a Kconfig and CMakeLists.txt files for build configuration and source codes includes and selection;
  - Import all the source code but only compile and include what is needed;
- For a new SoC, files to add:
  - `dts/<architecture>/<vendor>/<vendor>_<soc_name>.dtsi`
  - `soc/<architecture>/<soc_family>/<soc_series>/`

```
.
├── CMakeLists.txt
├── dts.fixup
├── Kconfig.defconfig.<soc_series>
├── Kconfig.defconfig.series
├── Kconfig.series
├── Kconfig.soc
├── linker.ld
├── soc.c
└── soc.h
```

Hardware support checklist

- For a new Board, files to add:
  - `boards/<architecture>/<board_name>/`

```
.
├── board.h
├── CMakeLists.txt
├── <board_name>_defconfig
├── <board_name>.dts
├── <board_name>.yaml
├── doc
│   └── <board_name>.rst
├── Kconfig.board
├── Kconfig.defconfig
└── pinmux.c
```

Agenda

Hardware support implementation in Zephyr

Adding a new HAL

Adding a new SoC

Adding new drivers

Adding a new board

Debugging tips

Hardware support checklist

Contributing to mainline

Contributing to mainline

- Follow the coding style (except for vendor HAL or source inside `/ext` directory);
- Follow the commit guidelines;
- Follow the documentation guidelines;
- Run the sanitycheck before pushing;
- There is a good example of contribution workflow when submitting patches for review;

Contributing to mainline

- When adding a new hardware platform split the PR in different patches:
  - ext/hal: for adding a new hal
  - drivers: for adding a new driver
  - soc: for adding a new SoC
  - boards: for adding a new board

- Be patient.

References

- Zephyr docs:
  - Architecture Porting Guide
  - Board Porting Guide
  - Device Tree in Zephyr
  - Application Development Primer

- Using SoC Vendor HALs in the Zephyr Project - Maureen Helm, NXP Semiconductors - Embedded Linux Conference Europe 2017 - Video, slides.

THANK YOU !!!!

Questions?

Diego Sueiro, Embarcados
www.embarcados.com.br
diego.sueiro@gmail.com
linkedin.com/in/diegosueiro/

OpenIoT Summit Europe 2018