Zephyr® Project
Developer Summit

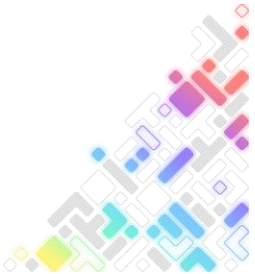
intel®

Future of Zephyr Device Model

Ederson de Souza, Intel Corporation

Agenda

- Current interest
- Deferred init
- Initialization order regarding services
- Device notifications
- Device unloading
- Questions



Device model issues and pull requests abound

Support peripheral deallocation at runtime #20012

kernel: device: add dts base device initialization priority #58296

Device initialization order that respects devicetree dependencies #22545

Refining Zephyr's Device Driver Model - Take II #22941

init sequence rework #6

init: specify dependencies between init entries #49

Allow for selective deferment of device initialization (aka manual init) #39896

t circular DTS node refere

Ability to use an API with device without extending its own API #22415

Adding support for multi-functional-devices #48934

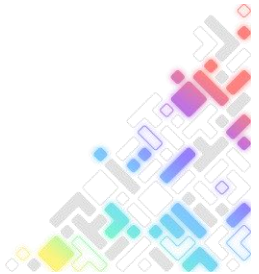
Device initialization improvements #34518

Device (de)initialization use case #40

Refining Zephyr's Device Driver Model #6393

What people want?

- Deferring device initialization
- Unloading/de-initializing devices
    - Power management integration/conflicts
- Initialization order regarding "services"
- API extensibility
- And more...



Please talk!

Deferred initialization

A look into the recent time

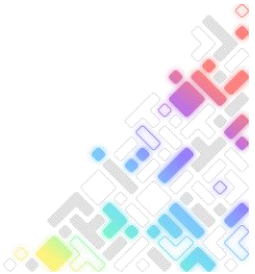
- Status: merged
- Really lengthy discussion - #39896 is from Oct/21 - maybe other discussion before
- A few PRs to tackle it
  - With more discussion!
  - PR discussion deviates from initial issue

```
# device tree
(...)
  status = "okay";
  zephyr,deferred-init;
};

/* Code */
ret = device_init(SOME_DEVICE);
```

Initialization order regarding services

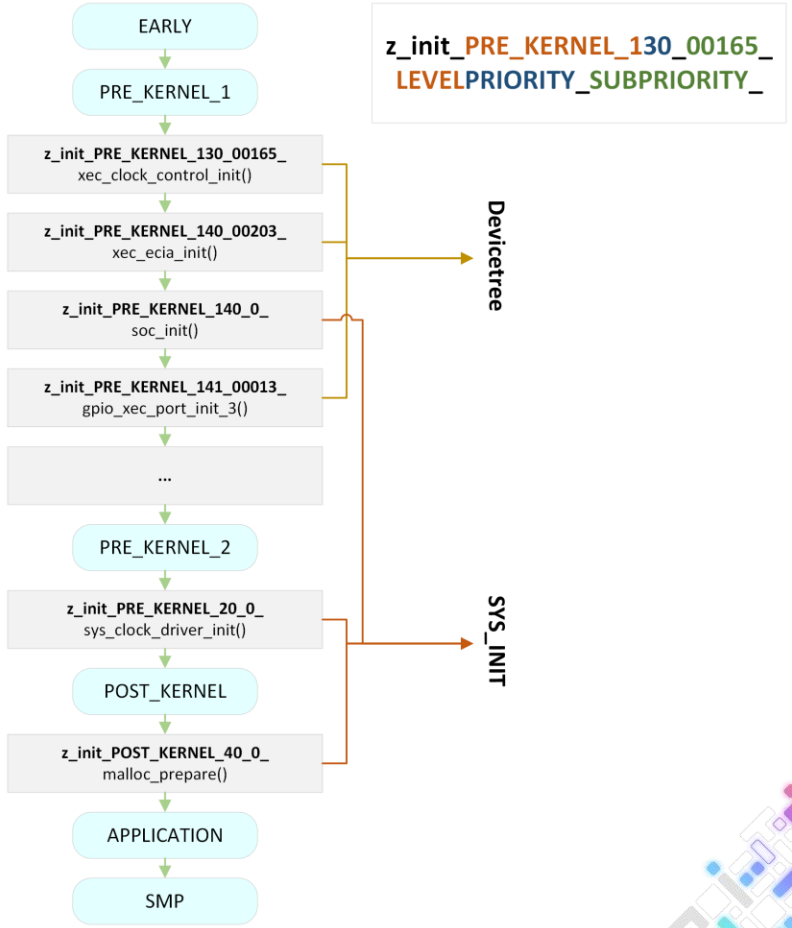
A look into the near future

- Device initialization order comes from devicetree
    - There's also a priority, so GPIO can be initialized after interrupts, for instance
- Things not on devicetree – calling them "services" here – are initialized via SYS_INIT
- Initialization level and priority inside that level are used to order SYS_INIT calls
- Numbers make it hard to track dependencies
    - Usually, priorities are tweaked after something doesn't work



How it is today

- "Init entries" containing the device and a function to perform initialization are created
- Sorted by LEVEL, PRIORITY and SUBPRIORITY
  - `KEEP(*(SORT(.z_init_<LEVEL>?_*)));`
- SUBPRIORITY keeps different instances of the same device in some order
- `west build -t initilevels` to see the order



A proposal

Get rid of numbers, track dependencies by name

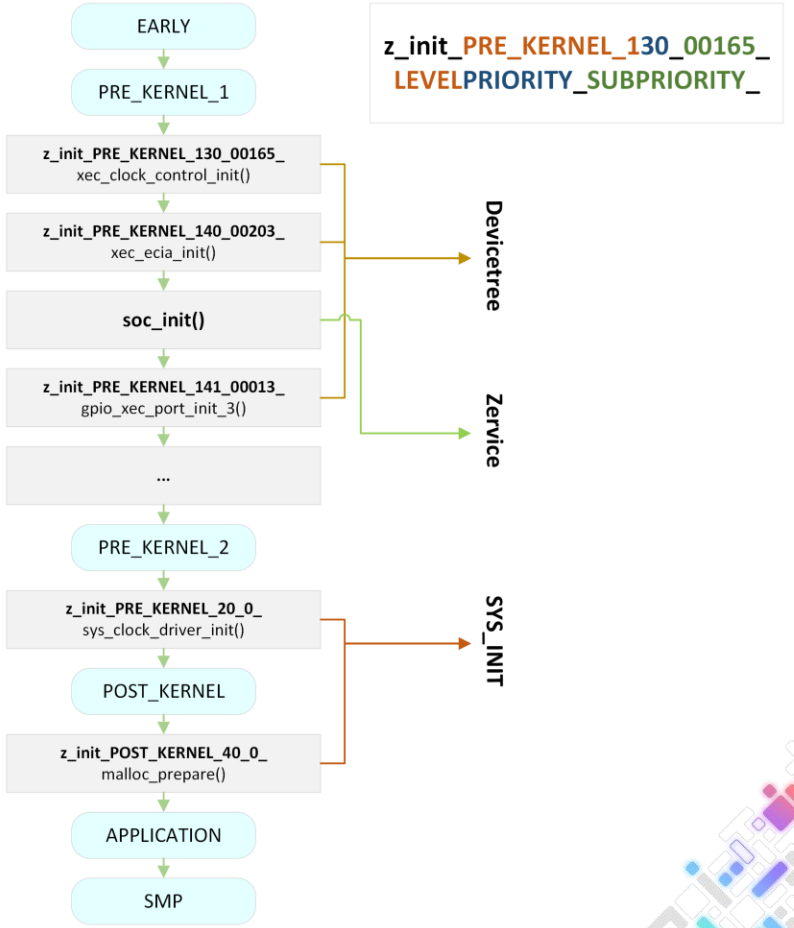
- Read info during build and generate a linker script snippet with entries topologically sorted
- Code defined dependencies
- Kconfig allows dependencies to be overwritten
- Integrate into current model, no need to rewrite the world

```
/* Code defined dependencies */
ZERVICE_DEFINE(soc_init, "microchip,xec-gpio-v2",
"microchip,xec-ecia");

# Kconfig allows dependencies to be overwritten
config ZERVICE_SOC_INIT_AFTER
    string "SoC initialization after"
    default "microchip,xec-ecia"

config ZERVICE_SOC_INIT_BEFORE
    string "SoC initialization before"
    default "microchip,xec-gpio-v2"

# prj.conf overwritting a dependency
CONFIG_ZERVICE_SOC_INIT_BEFORE=
"microchip,xec-gpio-v2#4"
```

ZERVICE?

How it becomes (I)

- "Init entries" are still created for devices, SYS_INIT and Zervices
- Devices and SYS_INIT ones are still sorted as of today in the *first build step*
- A new build script reads the first build step ELF map, Kconfig and devicetree info to find the place for the Zervices
- A linker script snippet is generated with all "init entries" in their final ordering, which is used in the *next build step*



How it becomes (II)

- One can even use current levels as dependencies
  - `ZERVICE_DEFINE(some_service, ”APPLICATION”, ”i2c_init;PRE_KERNEL_2”)`
- While seamless integration with current scheme is helpful, expectation is that eventually everything moves to the dependence-based mechanism
- It is possible to depend on a specific instance of a device, but currently this is done using the arbitrary devicetree instance number
  - Not good, as it doesn't relate with anything (#3 device may come before #0)
  - Device ordinal doesn't seem much better

```
/* Generated snippet sample */
__init_EARLY_start = .;
__init_PRE_KERNEL_1_start = .;
KEEP(*(.z_init_PRE_KERNEL_130_00165___init___device_dts_ord_165))
KEEP(*(.z_init_PRE_KERNEL_140_00203___init___device_dts_ord_203))
KEEP(*(.z_zervice_soc_init))
KEEP(*(.z_init_PRE_KERNEL_141_00013___init___device_dts_ord_13))
KEEP(*(.z_init_PRE_KERNEL_141_00014___init___device_dts_ord_14))
(...)
```

Relevant PR

https://github.com/zephyrproject-rtos/zephyr/pull/71470

Device notifications

A look into the not so near future

- Exploring ideas to notify about devices that failed to initialize – or were initialized late
    - Allow applications to track device state
- Used zbus to deliver notifications
    - Applications then subscribe to channels relating to a topic (device failed) or the device itself
- Why not?
    - Include zbus – but zbus should be light
    - Zbus is fully functional at Application level – is it really a problem?
- Help with another device issues?

Device unloading/de-initializing

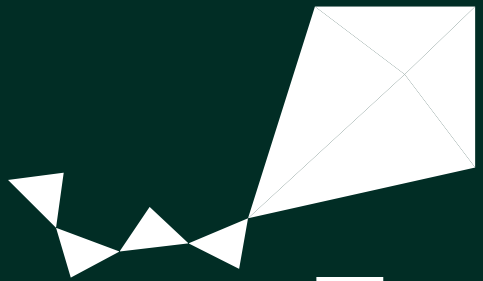
A look into the not so near future

- Devices can be initialized, why not the opposite?
    - Unloading/closing then on a Zephyr based bootloader before jumping to the next boot phase
- Overlap with Power Management?
    - Independent?
    - Interdependent?
    - One on top of the other? Which one?
- What to do about dependencies?
    - Mandatory and optional
- device_get()/device_put()
    - Changes across the board
    - Subtle errors
- Notification
    - Device notifies is going down, those (devices/application) who care subscribe and take appropriate action

Relevant PR

https://github.com/zephyrproject-rtos/zephyr/pull/71469

Questions? Comments? Suggestions?



Zephyr® Project

Developer Summit