Devicetrees in Zephyr

Navigating Hardware Diversity

Stefan Kratochwil

Karlsruhe · Köln · München · Hamburg · Berlin · Stuttgart · Pforzheim · Erlangen

inovex

Stefan Kratochwil

stefan.kratochwil@inovex.de

- Embedded Systems
- Operating Systems
- Software Architecture
- Heterogeneous Systems

About This Talk

Diversity in embedded platforms

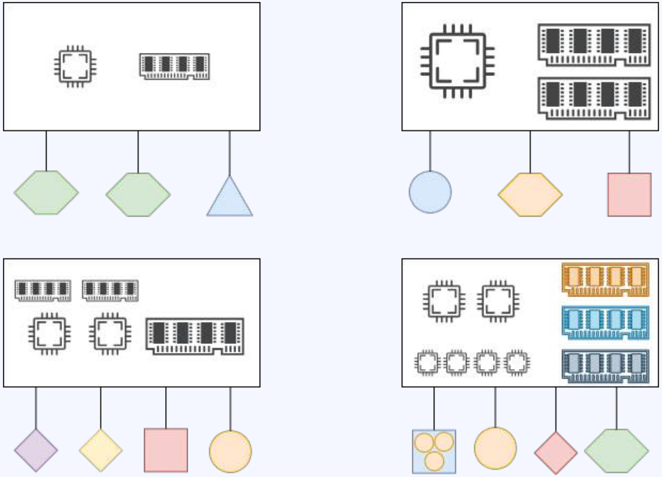
Software portability

The role of devicetrees

Implementation and use in Zephyr

Showcase inoCube:
    Description != Configuration

Variety of Embedded Platforms



Hello World

Power On Reset

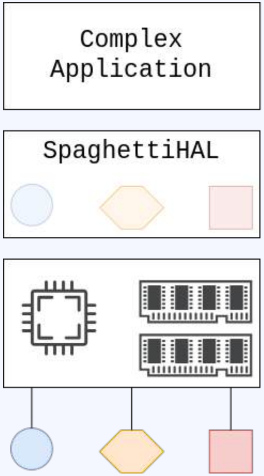
Operating System Initialization

Question 1: "Where am I?"

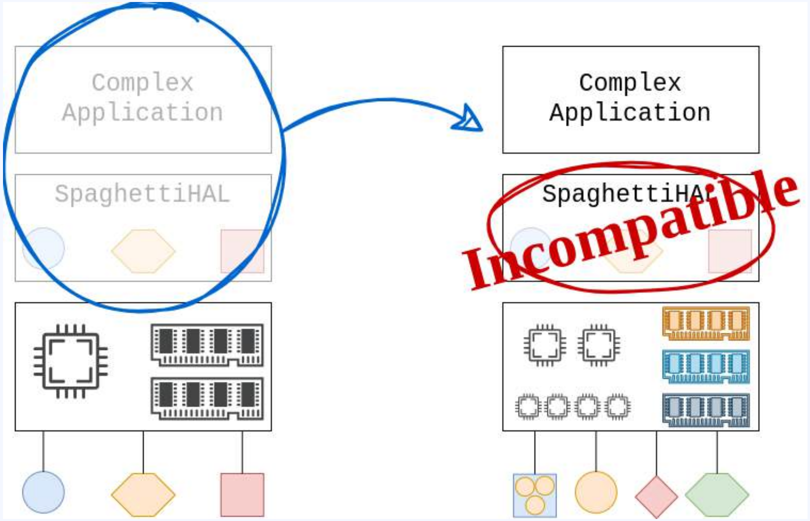
Question 2: "Where are my peripherals?"

Question 3: "Where are my drivers?"

→ Traditional solution: Hardcoded



Portability?



Introducing: `devicetree`

Hardware description format

Used in Linux, u-boot, **Zephyr**, and others

```
/ {
    cpus {
      cpu@0 {                             // Where am I?
        device_type = "cpu";
        compatible = "arm,cortexm4f";
      }
    };
    soc {                                 // Where are ...
      uart0: uart@40002000 {              // ... my peripherals?
        compatible = "nordic,nrf-uarte";  // (... my drivers?)
        reg = <0x40002000 0x1000>;
        interrupts = <2 NRF_DEFAULT_IRQ_PRIORITY>;
        status = "okay";
      };
    };
  };
```

Zephyr

https://www.zephyrproject.org/

https://www.devicetree.org/

inovex

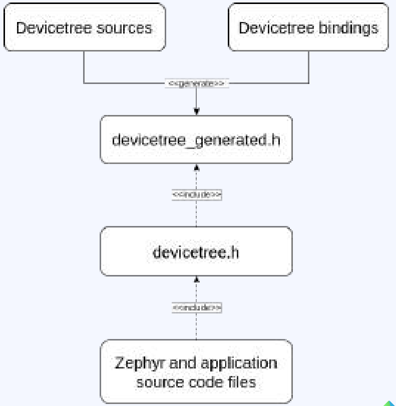
Usage in Zephyr

`*.dts` - Describes a specific instance of a board.

`*.dtsi` - Contains shareable elements among different boards.

`*.overlay` - Used to modify or extend specific devicetree nodes.

`*.yaml` - Bindings: Semantics for the devicetree.



Example: Hardware Description vs. Configuration

InoCube shall blink. We need:

- Drivers (led_strip, ws2812-spi) ✓
- DT-Bindings ✓
- DT-Node

```yaml
# dts/bindings/led_strip/ws2812.yaml
# Devicetree bindings file:
properties:
  chain-length:                # <-
    type: int
    required: true             # <-
```

```c
// include/zephyr/drivers/led_strip.h
// Prototype for led_strip API function:
typedef int (*led_api_update_rgb) (
    const struct device *dev,
    struct led_rgb *pixels,
    size_t num_pixels           // <-
);
```

Representation of `chain-length` inside inoCube application?

- Either during compile time:

```
// .dts file:
chain-length = <1>;

// Constant in .c file:
#define N_LEDS DT_PROP( DT_ALIAS(led_strip), chain_length) )
```

- Or ignore DT property and obtain it at runtime?

```
// Ignore devicetree, get property from somewhere else:
unsigned int chain_length = config_get_chain_length();
```

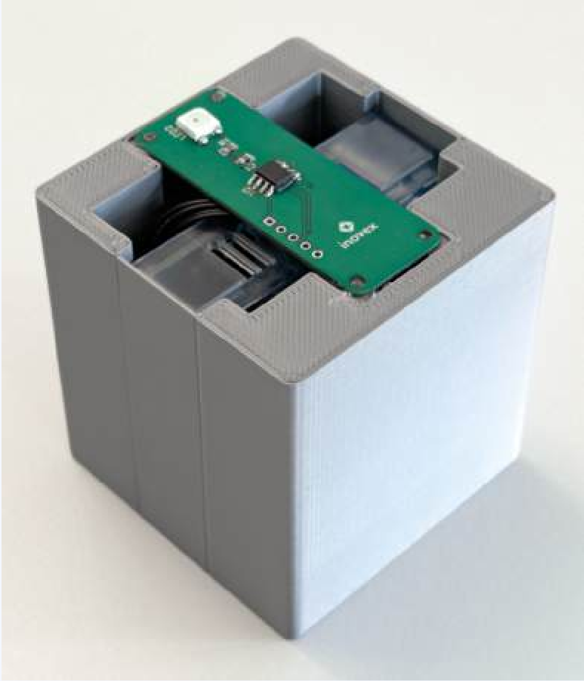
It Depends on the Use-case

Is it a fixed hardware property?
→ `devicetree`

Is the property user-modifyable?
→ Handle at runtime

InoCube:

```
led_strip: ws2812@0 {
   ...
   chain-length = <1>;
   ...
}
```



Thank you!

Stefan Kratochwil
stefan.kratochwil@inovex.de
Ludwig-Erhard-Allee 6
76131 Karlsruhe

inovex is an IT project center driven by innovation and quality, focusing its services on 'Digital Transformation'.

- founded 1999
- 500+ employees
- 8 offices across germany

www.inovex.de