DEVICE TREE

THE LANGUAGE TO DESCRIBE HARDWARE

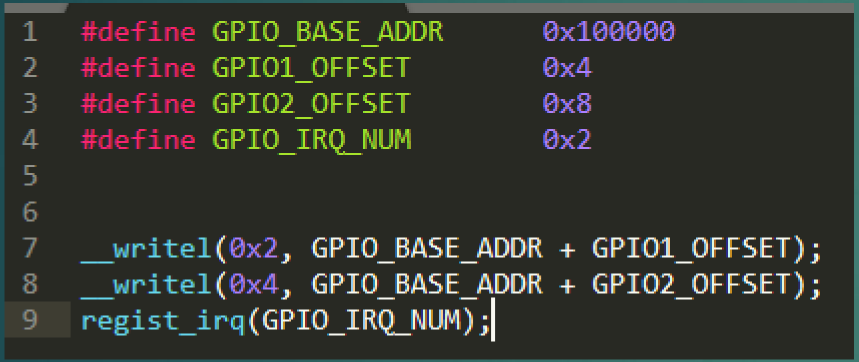
Author: Luu An Phu

AGENDA

- ✓ Introduction
- ✓ Booting flow with the Device Tree support
- ✓ Basic Device Tree syntax
- ✓ Examples with Device Tree
- ✓ General functions in device tree API

INTRODUCTION

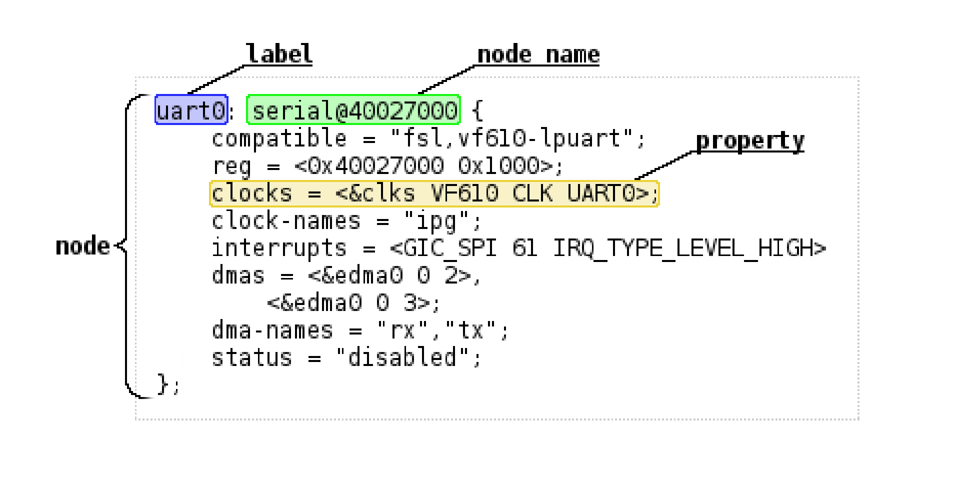
Old style of kernel code



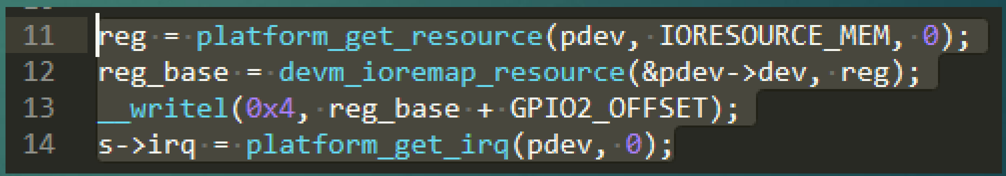
```
#define GPIO_BASE_ADDR      0x100000
#define GPIO1_OFFSET        0x4
#define GPIO2_OFFSET        0x8
#define GPIO_IRQ_NUM        0x2


__writel(0x2, GPIO_BASE_ADDR + GPIO1_OFFSET);
__writel(0x4, GPIO_BASE_ADDR + GPIO2_OFFSET);
regist_irq(GPIO_IRQ_NUM);
```

- ❖ Disadvantage
  - ➢ Hard to reuse source code

New style with device tree



```
uart0: serial@40027000 {
        compatible = "fsl,vf610-lpuart";
        reg = <0x40027000 0x1000>;
        clocks = <&clks VF610 CLK UART0>;
        clock-names = "ipg";
        interrupts = <GIC_SPI 61 IRQ_TYPE_LEVEL_HIGH>
        dmas = <&edma0 0 2>,
            <&edma0 0 3>;
        dma-names = "rx","tx";
        status = "disabled";
};
```

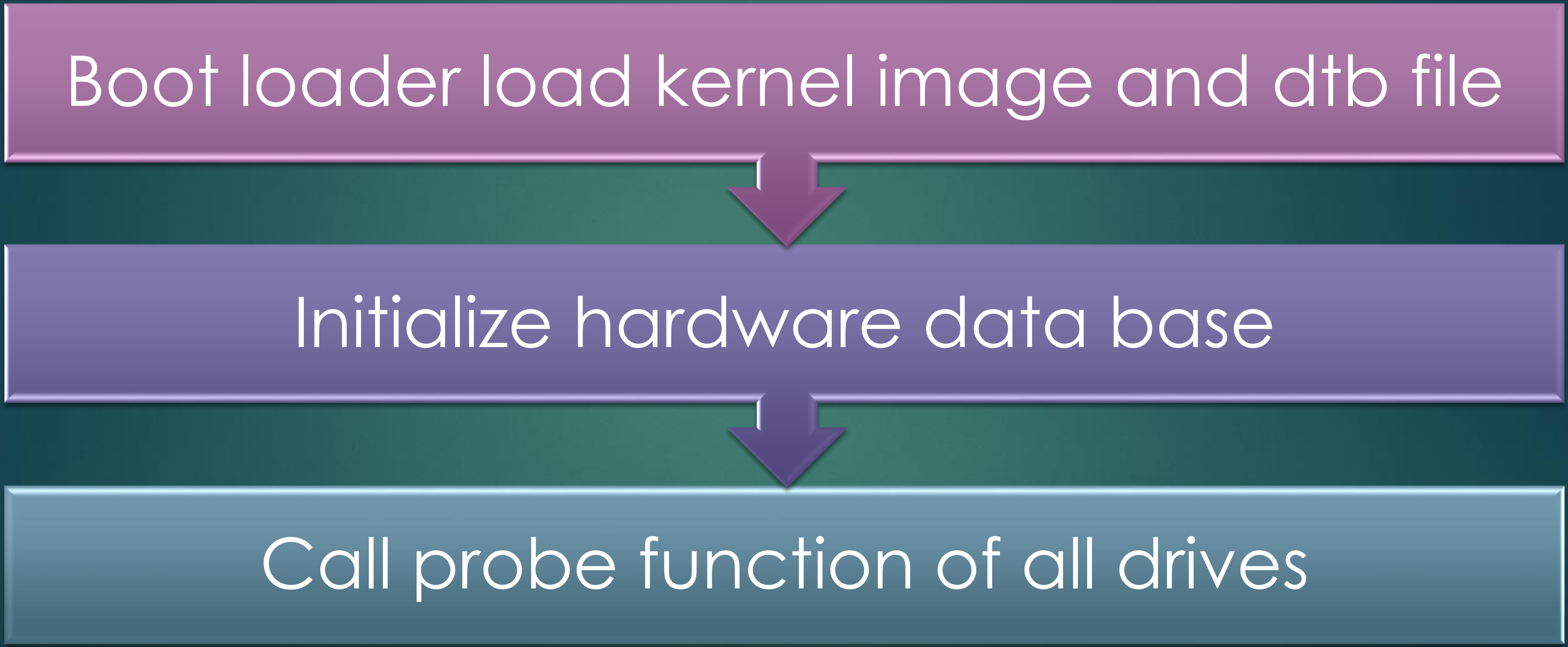


```
reg = platform_get_resource(pdev, IORESOURCE_MEM, 0);
reg_base = devm_ioremap_resource(&pdev->dev, reg);
__writel(0x4, reg_base + GPIO2_OFFSET);
s->irq = platform_get_irq(pdev, 0);
```

DEVICE TREE ADVANTAGE

- ✓ Simple to change the configuration
- ✓ Easily add support for new hardware
- ✓ Can reuse and over ride existing .dots files
- ✓ Easy to read and understand descriptions of hardware

BOOTING FLOW



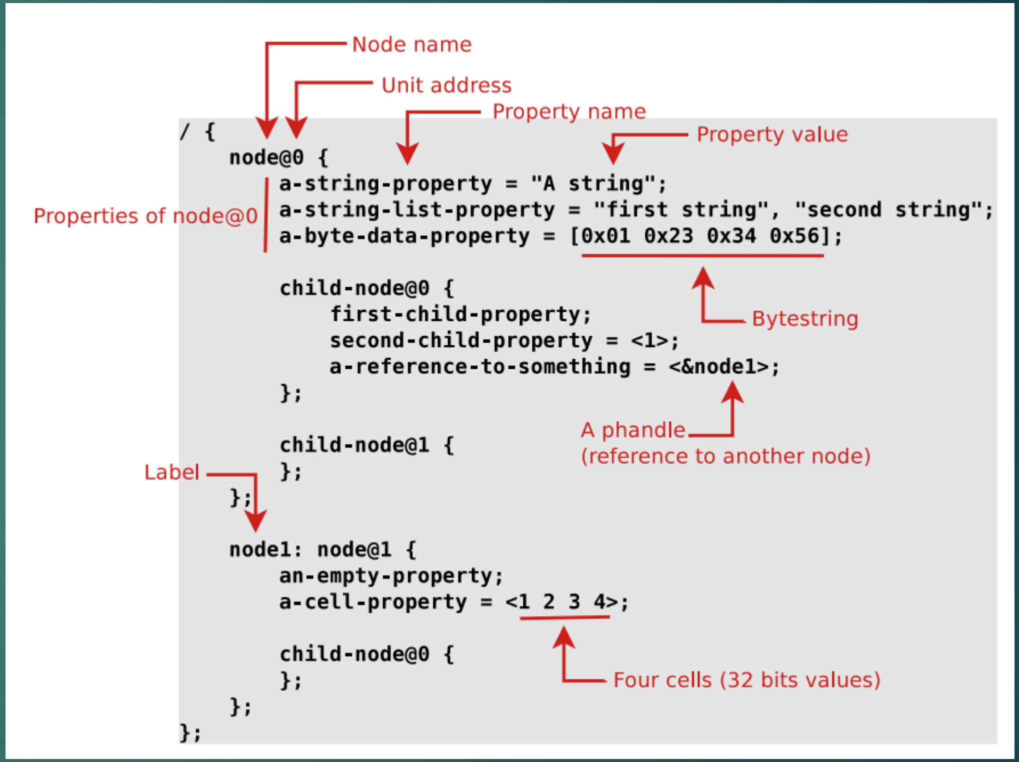
Boot loader load kernel image and dtb file

Initialize hardware data base

Call probe function of all drives

Basic Device Tree syntax

- Node name
- Unit address
- Property name
- Property value
- Value encode
- Reference with other node
- Node tuân theo cú pháp chuẩn.
- Node tự định nghĩa.



```
/ {
    node@0 {
        a-string-property = "A string";
        a-string-list-property = "first string", "second string";
        a-byte-data-property = [0x01 0x23 0x34 0x56];

        child-node@0 {
            first-child-property;
            second-child-property = <1>;
            a-reference-to-something = <&node1>;
        };

        child-node@1 {
        };
    };

    node1: node@1 {
        an-empty-property;
        a-cell-property = <1 2 3 4>;

        child-node@0 {
        };
    };
};
```

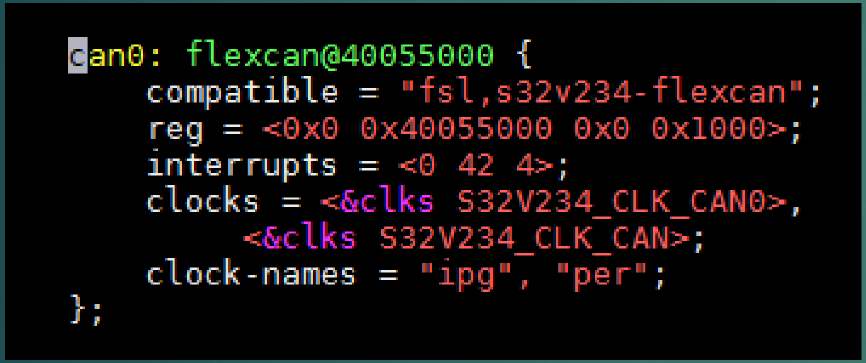
Device tree compilation

- Include other device tree
  - Reuse and override
- Device tree located in source code
- Tool to compile
- Device Tree Blob

- .dtsi files are included files, while .dts files are *final*
- arch/platform/boot/dts
- scripts/dtc
- .dts + .dtsi = .dtb

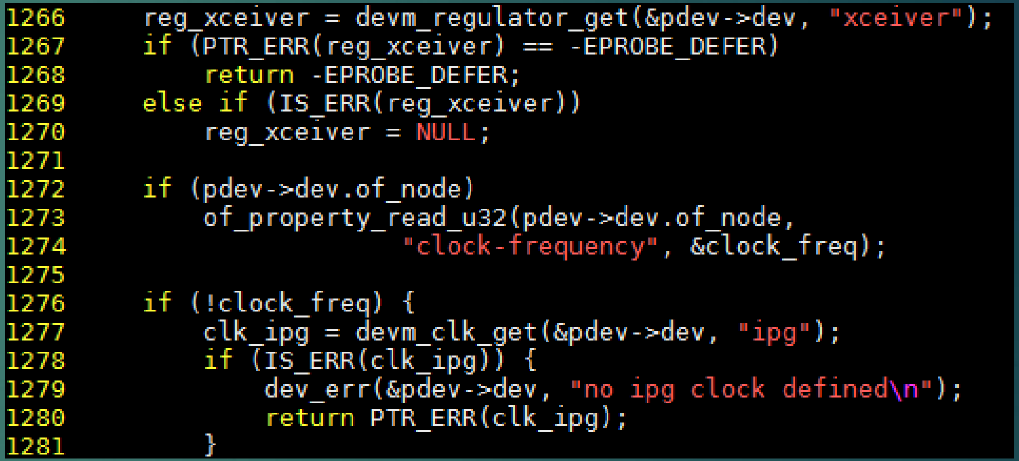
Examples with Device Tree

Device tree node



```
can0: flexcan@40055000 {
    compatible = "fsl,s32v234-flexcan";
    reg = <0x0 0x40055000 0x0 0x1000>;
    interrupts = <0 42 4>;
    clocks = <&clks S32V234_CLK_CAN0>,
         <&clks S32V234_CLK_CAN>;
    clock-names = "ipg", "per";
};
```

Driver code



```
    reg_xceiver = devm_regulator_get(&pdev->dev, "xceiver");
    if (PTR_ERR(reg_xceiver) == -EPROBE_DEFER)
        return -EPROBE_DEFER;
    else if (IS_ERR(reg_xceiver))
        reg_xceiver = NULL;

    if (pdev->dev.of_node)
        of_property_read_u32(pdev->dev.of_node,
                     "clock-frequency", &clock_freq);

    if (!clock_freq) {
        clk_ipg = devm_clk_get(&pdev->dev, "ipg");
        if (IS_ERR(clk_ipg)) {
            dev_err(&pdev->dev, "no ipg clock defined\n");
            return PTR_ERR(clk_ipg);
        }
```

General function in device tree API

- All functions are located at include/Linux/of.h
  - Getting a reference to the clock
    - clk_get(&pdev->dev, NULL)
  - Getting the I/O registers resource
    - platform_get_resource(pdev, IORESOURCE_MEM, 0)
  - Check some custom property
    - struct device_node *np = pdev->dev.of_node
    - of_get_property(np, "fsl,uart-has-rtscts", NULL)

SUMMARY

- Booting flow with device tree supported.
- How to a device tree blob is compiled.
- Device tree basic syntax
- Reuse and over ride in device tree
- Device tree API function

Practice

- Add 1 node để định nghĩa phần cứng cho đèn led và button.
  - Địa chỉ bắt đầu của các thanh ghi cho đèn led và button.
  - Số hiệu ngắt cho button.
- Viết lại driver để điều khiển led và button dựa vào device tree.
  - Ấn button thì đèn led sáng.
  - Nhả button thì đèn led tắt.

Reference

- Device tree advantage
- Device tree for dummy

Thank you