ELEC 424/553

Mobile & Embedded Systems

Lecture 16

Device Trees, gpiod

URL: https://relicrecord.com/blog/bob-happy-little-trees-ross/

Notes

- Raspberry Pi wifi manager
- (don’t run sudo apt upgrade)

Question of The Day:

Why hasn’t Linux taken over the desktop?

Question of The Day - Why Hasn't Linux Taken Over The Desktop?

https://www.youtube.com/watch?v=5Qj8p-PEwbI

Question of The Day - Why Hasn’t Linux Taken Over The Desktop?

https://www.youtube.com/watch?v=KFKxlYNfT_o

Question of The Day - Why Hasn't Linux Taken Over The Desktop?

https://www.youtube.com/watch?v=qHGTs1NSB1s

Housekeeping

- **Assignment 2 Due Tonight**
  - Don’t forget to submit your code
- **Project 2** to be posted this week
- **Assignment 3** to be posted next week
- **Midterm** to be posted Monday Nov 6
- **Project 3**
- **Final Project**

Questions Will Be Answered & Answers Will Be Questioned

- Note on `mutex` - interesting slides here
- `copy_to_user()` concerns
    - Hardened usercopy (LWN, Red Hat)
    - Whitelisting (ServerWatch)
- Can we trust drivers?
    - Boyd-Wickizer & Zeldovich, “Tolerating Malicious Device Drivers in Linux”. In *Proceedings of the 2010 USENIX Annual Technical Conference*, Boston, MA, June 2010. Link

Objectives For Today

- Understand more about the Device Tree through examples
    - Syntax
    - Driver linking

Project 2 Goal: Modify Devicetree to Use `gpiod` To Toggle GPIO

- Modify an existing Devicetree (DT) overlay to give us access to a GPIO pin via gpiod functions in kernel space
  - Overlay: A file that overwrites portions of the platform's standard device tree
- Compile and install the overlay
- Write device driver to utilize GPIO pin via gpiod
  - Will rely on information provided in altered device tree so driver can link up with GPIO pin

Accessing GPIO

- **Kernel space**
  - `gpio`
  - **`gpiod`**
- **User space**
  - `sysfs`
  - `Char dev` [Kernel 4.8]
    - **`libgpiod`**
      - Command line & C program

Compiled Device Tree File - Pi

```
/dts-v1/;

/ {
    compatible = "raspberrypi,model-zero-w\0brcm,bcm2835";
    serial-number = "00000000064e4bf5";
    model = "Raspberry Pi Zero W Rev 1.1";
    memreserve = <0x1c000000 0x4000000>;
    interrupt-parent = <0x01>;
    #address-cells = <0x01>;
    #size-cells = <0x01>;

    reserved-memory {
        ranges;
        #address-cells = <0x01>;
        #size-cells = <0x01>;
        phandle = <0x2f>;
   ...
```

Compiled Device Tree File - BeagleBone Black

```
/dts-v1/;

/ {
    compatible = "ti,am335x-bone-black\0ti,am335x-bone\0ti,am33xx";
    serial-number = "2125SBB05081";
    model = "TI AM335x BeagleBone Black";
    interrupt-parent = < 0x01 >;
    #address-cells = < 0x01 >;
    #size-cells = < 0x01 >;

    clk_mcasp0_fixed {
      compatible = "fixed-clock";
      #clock-cells = < 0x00 >;
      phandle = < 0x2ca >;
      clock-frequency = < 0x1770000 >;
    };

    ...
```

Device Tree (DT) Bindings

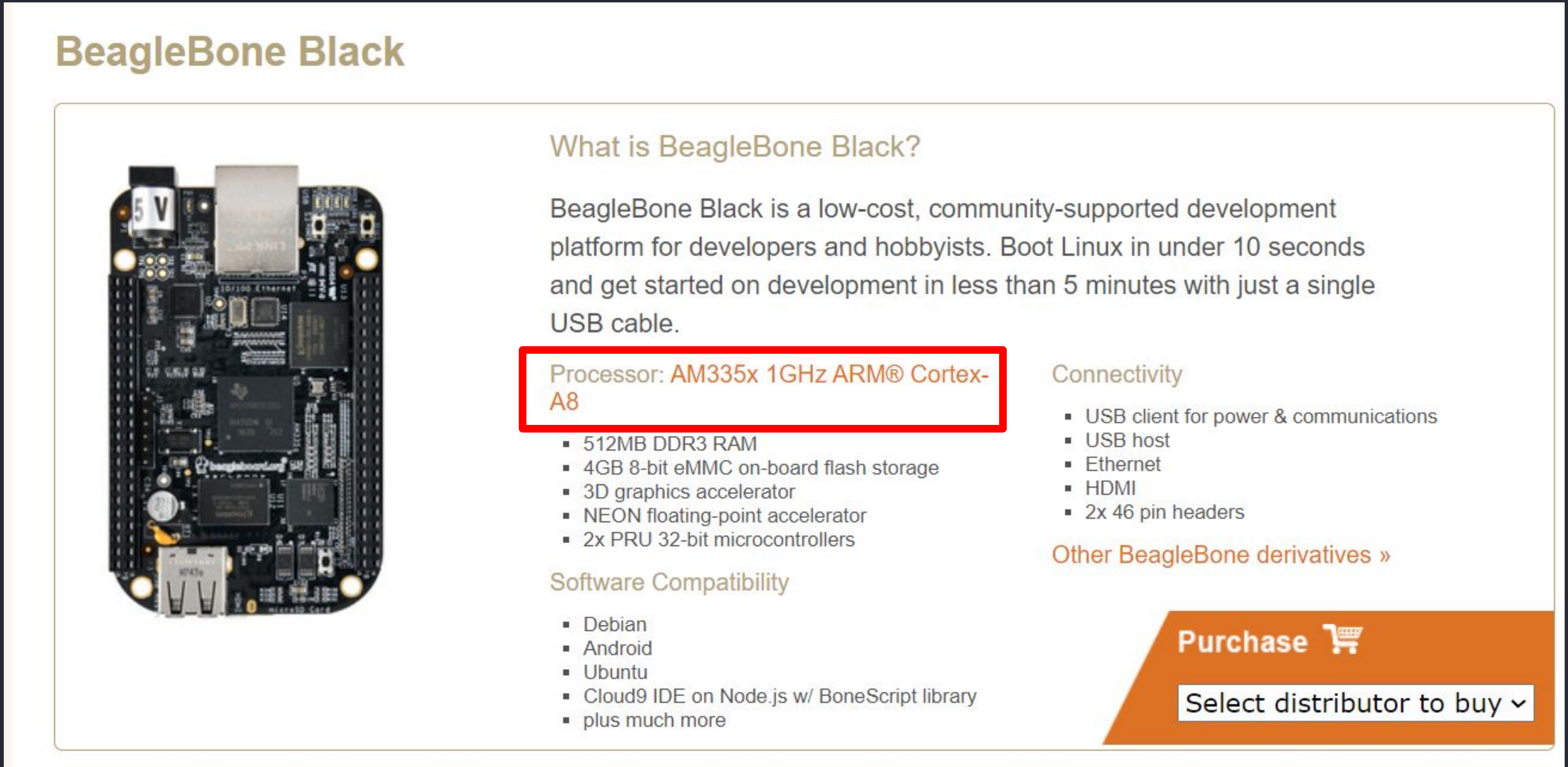
- A non-discoverable hardware description given by DT
- DT **bindings** provide required structure of description
- **`compatible`** property specifies binding to be used
    - Overwhelming detail in Documentation/devicetree/bindings
    - When in doubt, look at the binding
    - `compatible` also used to link with **driver**
- Drivers take the description and run with it

Marta Rybczyńska, “Device-tree schemas”. URL: https://lwn.net/Articles/771621/#:~:text=The%20device%2Dtree%20bindings%20define,which%20values%20they%20may%20have.

Specific Example: LEDs In Compiled Device Tree

```
leds {
     compatible = "gpio-leds";           /* Binding; Driver will also use this property */
     pinctrl-0 = < 0x20c >;
     pinctrl-names = "default";

      ...

     led4 {
           gpios = < 0x58 0x17 0x00 >;  /* Where in the world did this come from? */
           label = "beaglebone:green:usr2";
           default-state = "off";
           linux,default-trigger = "cpu0";
      };
```

Let’s Look At The Source Device Tree - Where Is The Source Code?



BeagleBone Black

What is BeagleBone Black?

BeagleBone Black is a low-cost, community-supported development platform for developers and hobbyists. Boot Linux in under 10 seconds and get started on development in less than 5 minutes with just a single USB cable.

Processor: AM335x 1GHz ARM® Cortex-A8

- 512MB DDR3 RAM
- 4GB 8-bit eMMC on-board flash storage
- 3D graphics accelerator
- NEON floating-point accelerator
- 2x PRU 32-bit microcontrollers

Software Compatibility

- Debian
- Android
- Ubuntu
- Cloud9 IDE on Node.js w/ BoneScript library
- plus much more

Connectivity

- USB client for power & communications
- USB host
- Ethernet
- HDMI
- 2x 46 pin headers

Other BeagleBone derivatives »

https://beagleboard.org/black

Let’s Look At The Source Device Tree - Where Is The Source Code?

Click:

https://elixir.bootlin.com/linux/latest/source

or Google “Bootlin Elixir”

Start with

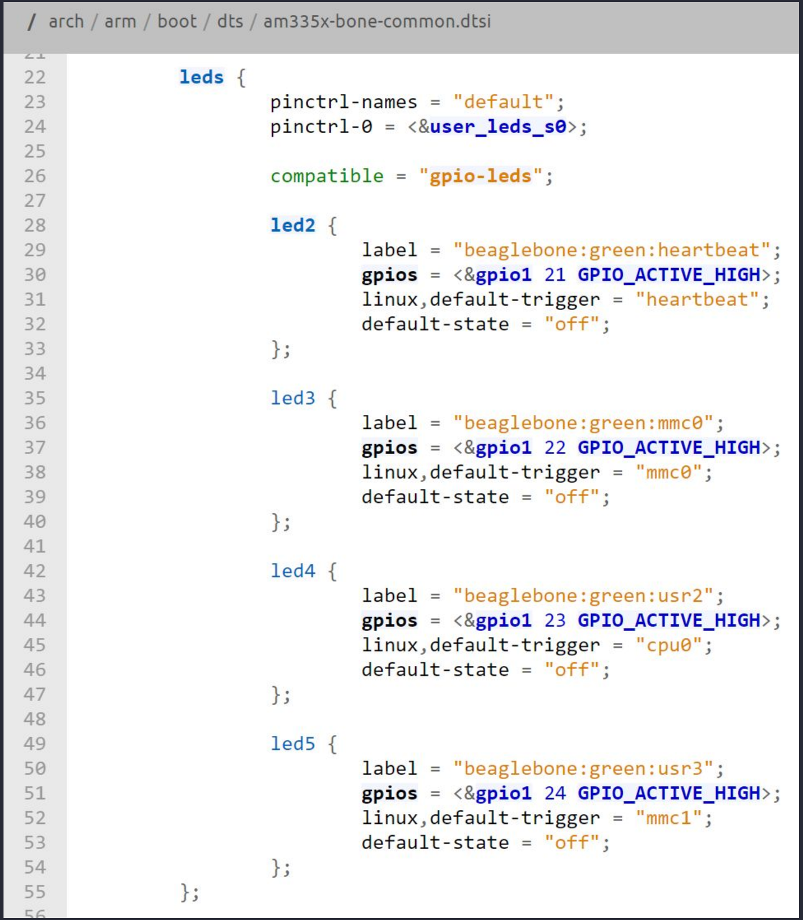
/arch/arm/boot/dts/am335x-boneblack.dts

File we are eventually looking for:

/arch/arm/boot/dts/am335x-bone-common.dtsi

Let’s Look At The Source Device Tree

```
leds {
     compatible = "gpio-leds";
     pinctrl-0 = < 0x20c >;
     pinctrl-names = "default";

      ...

     led4 {
           gpios = < 0x58 0x17 0x00 >;
           label = "beaglebone:green:usr2";
           default-state = "off";
           linux,default-trigger = "cpu0";
      };
```



/ arch / arm / boot / dts / am335x-bone-common.dtsi

```
	leds {
		pinctrl-names = "default";
		pinctrl-0 = <&user_leds_s0>;

		compatible = "gpio-leds";

		led2 {
			label = "beaglebone:green:heartbeat";
			gpios = <&gpio1 21 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "heartbeat";
			default-state = "off";
		};

		led3 {
			label = "beaglebone:green:mmc0";
			gpios = <&gpio1 22 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "mmc0";
			default-state = "off";
		};

		led4 {
			label = "beaglebone:green:usr2";
			gpios = <&gpio1 23 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "cpu0";
			default-state = "off";
		};

		led5 {
			label = "beaglebone:green:usr3";
			gpios = <&gpio1 24 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "mmc1";
			default-state = "off";
		};
	};
```

Source screenshot from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am335x-bone-common.dtsi

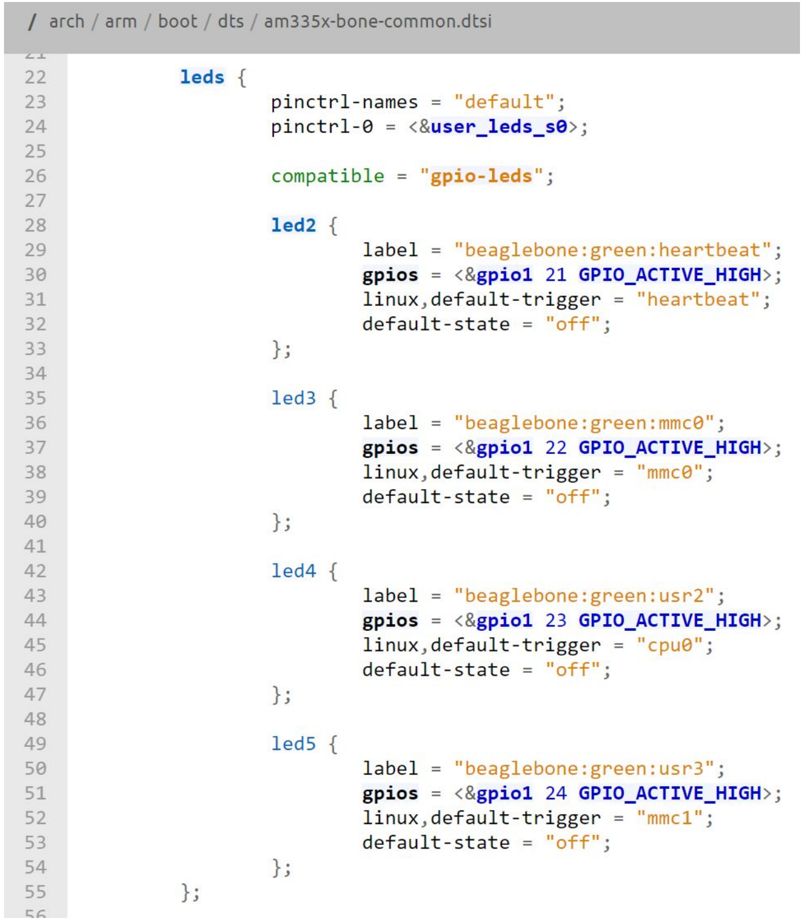
Check Out The Binding

Copying from (with my own bolding) https://www.kernel.org/doc/Documentation/devicetree/bindings/leds/leds-gpio.txt :

LEDs connected to GPIO lines

Required properties:
- **compatible** : should be "gpio-leds".

Each LED is represented as a sub-node of the gpio-leds device. Each node's name represents the name of the corresponding LED.



/ arch / arm / boot / dts / am335x-bone-common.dtsi

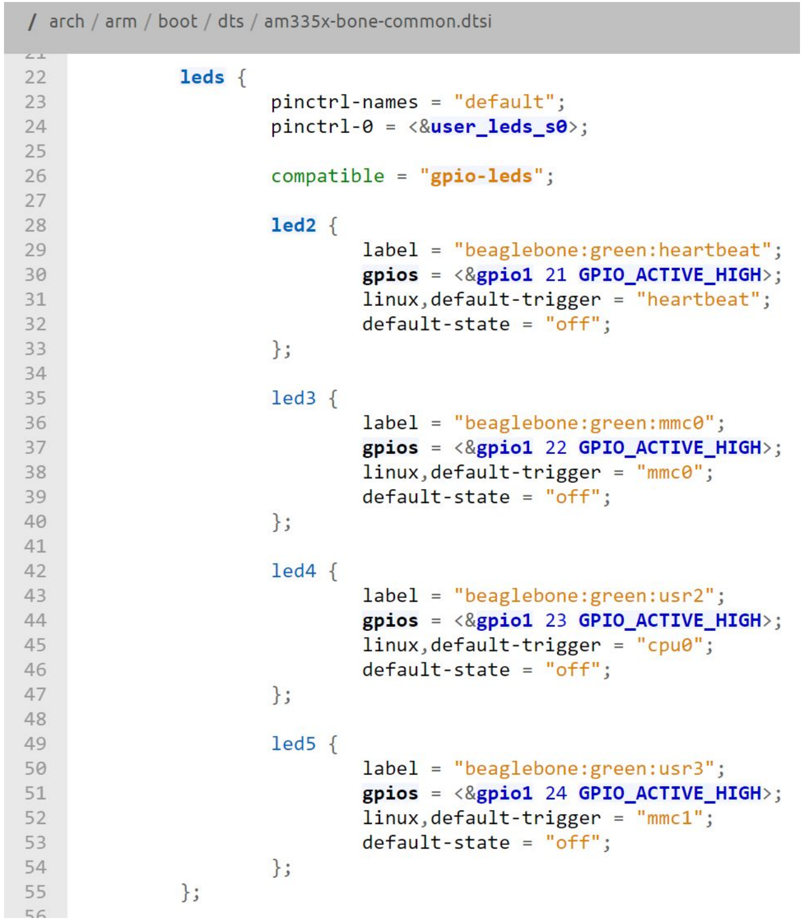
```
        leds {
                pinctrl-names = "default";
                pinctrl-0 = <&user_leds_s0>;

                compatible = "gpio-leds";

                led2 {
                        label = "beaglebone:green:heartbeat";
                        gpios = <&gpio1 21 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "heartbeat";
                        default-state = "off";
                };

                led3 {
                        label = "beaglebone:green:mmc0";
                        gpios = <&gpio1 22 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "mmc0";
                        default-state = "off";
                };

                led4 {
                        label = "beaglebone:green:usr2";
                        gpios = <&gpio1 23 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "cpu0";
                        default-state = "off";
                };

                led5 {
                        label = "beaglebone:green:usr3";
                        gpios = <&gpio1 24 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "mmc1";
                        default-state = "off";
                };
        };
```

Source screenshot from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am335x-bone-common.dtsi

Check Out The Binding (2)

Copying from (with my bolding/formatting)
https://www.kernel.org/doc/Documentation/devicetree/bindings/leds/leds-gpio.txt:

LED sub-node properties:
- **gpios** : Should specify the LED's GPIO, see "gpios property" in Documentation/devicetree/bindings/gpio/gpio.txt. Active low LEDs should be
  indicated using flags in the GPIO specifier.
- **function** : (optional)
  see Documentation/devicetree/bindings/leds/common.txt
- **color** : (optional)
  see Documentation/devicetree/bindings/leds/common.txt
- **label** : (optional)
  see Documentation/devicetree/bindings/leds/common.txt (deprecated)
- **linux,default-trigger** : (optional)
  see Documentation/devicetree/bindings/leds/common.txt
- **default-state**: (optional) The initial state of the LED.
  see Documentation/devicetree/bindings/leds/common.txt



/ arch / arm / boot / dts / am335x-bone-common.dtsi

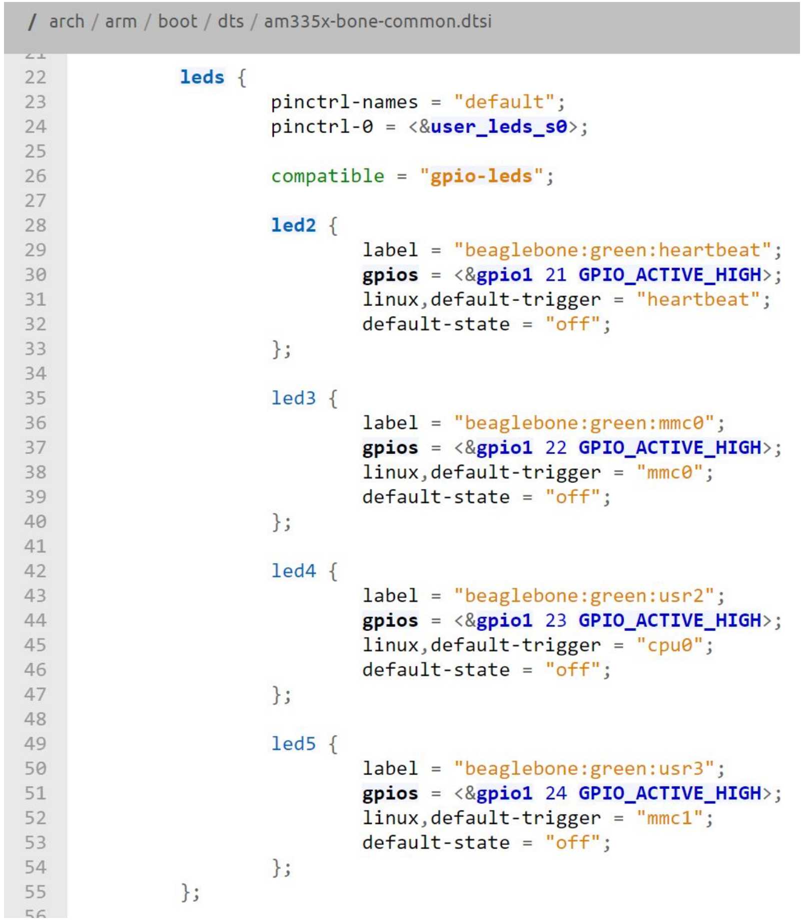
```
	leds {
		pinctrl-names = "default";
		pinctrl-0 = <&user_leds_s0>;

		compatible = "gpio-leds";

		led2 {
			label = "beaglebone:green:heartbeat";
			gpios = <&gpio1 21 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "heartbeat";
			default-state = "off";
		};

		led3 {
			label = "beaglebone:green:mmc0";
			gpios = <&gpio1 22 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "mmc0";
			default-state = "off";
		};

		led4 {
			label = "beaglebone:green:usr2";
			gpios = <&gpio1 23 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "cpu0";
			default-state = "off";
		};

		led5 {
			label = "beaglebone:green:usr3";
			gpios = <&gpio1 24 GPIO_ACTIVE_HIGH>;
			linux,default-trigger = "mmc1";
			default-state = "off";
		};
	};
```

Source screenshot from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am335x-bone-common.dtsi

Example

Copying from (with my bolding/formatting)
https://www.kernel.org/doc/Documentation/devicetree/bindings/leds/leds-gpio.txt:

```
leds {
        compatible = "gpio-leds";
        led0 {
                gpios = <&mcu_pio 0 GPIO_ACTIVE_LOW>;
                linux,default-trigger = "disk-activity";
                function = LED_FUNCTION_DISK;
        };

        led1 {
                gpios = <&mcu_pio 1 GPIO_ACTIVE_HIGH>;
                /* Keep LED on if BIOS detected hardware fault */
                default-state = "keep";
                function = LED_FUNCTION_FAULT;
        };
};
```



/ arch / arm / boot / dts / am335x-bone-common.dtsi

```
        leds {
                pinctrl-names = "default";
                pinctrl-0 = <&user_leds_s0>;

                compatible = "gpio-leds";

                led2 {
                        label = "beaglebone:green:heartbeat";
                        gpios = <&gpio1 21 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "heartbeat";
                        default-state = "off";
                };

                led3 {
                        label = "beaglebone:green:mmc0";
                        gpios = <&gpio1 22 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "mmc0";
                        default-state = "off";
                };

                led4 {
                        label = "beaglebone:green:usr2";
                        gpios = <&gpio1 23 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "cpu0";
                        default-state = "off";
                };

                led5 {
                        label = "beaglebone:green:usr3";
                        gpios = <&gpio1 24 GPIO_ACTIVE_HIGH>;
                        linux,default-trigger = "mmc1";
                        default-state = "off";
                };
        };
```

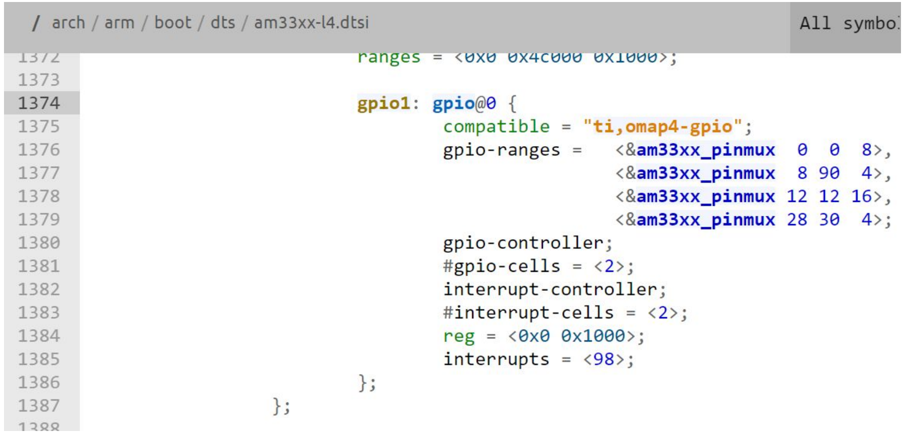
Source screenshot from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am335x-bone-common.dtsi

Look Back At Device Tree Source Code

Copying from (with my formatting/bold)
https://www.kernel.org/doc/Documentation/devicetree/bindings/gpio/gpio-omap.txt:

OMAP GPIO controller bindings
Required properties:

- **compatible**:
  - "ti,omap2-gpio" for OMAP2 controllers
  - "ti,omap3-gpio" for OMAP3 controllers
  - "ti,omap4-gpio" for OMAP4 controllers
- **reg** : Physical base address of the controller and length of memory mapped region.
- **gpio-controller** : Marks the device node as a GPIO controller.
- **#gpio-cells** : Should be two.
  - first cell is the pin number
  - second cell is used to specify optional parameters (unused)



```
                    ranges = <0x0 0x4c000 0x1000>;

                    gpio1: gpio@0 {
                            compatible = "ti,omap4-gpio";
                            gpio-ranges =   <&am33xx_pinmux  0  0  8>,
                                            <&am33xx_pinmux  8 90  4>,
                                            <&am33xx_pinmux 12 12 16>,
                                            <&am33xx_pinmux 28 30  4>;
                            gpio-controller;
                            #gpio-cells = <2>;
                            interrupt-controller;
                            #interrupt-cells = <2>;
                            reg = <0x0 0x1000>;
                            interrupts = <98>;
                    };
            };
```

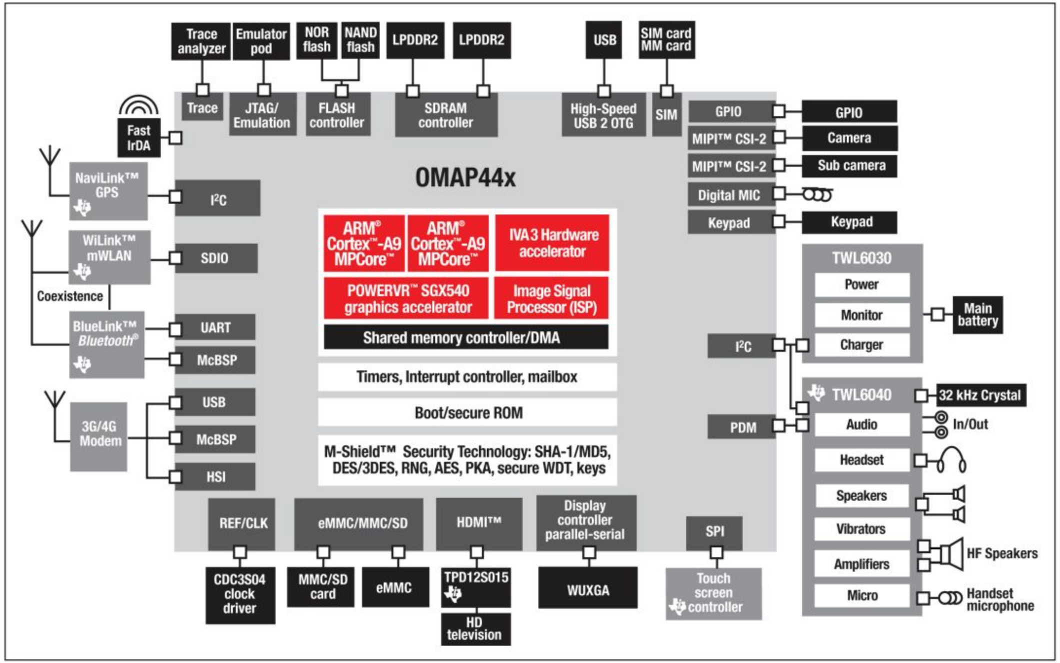
Bootlin Elixir screenshot of source from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am33xx-l4.dtsi#L1374

OMAP4?

OMAP™ 4 mobile applications platform

Product Bulletin

TI's new OMAP 4 platform supports development of planned features for the Smartphones and MIDs of tomorrow with tremendous performance and programmability to support new applications yet to be imagined.



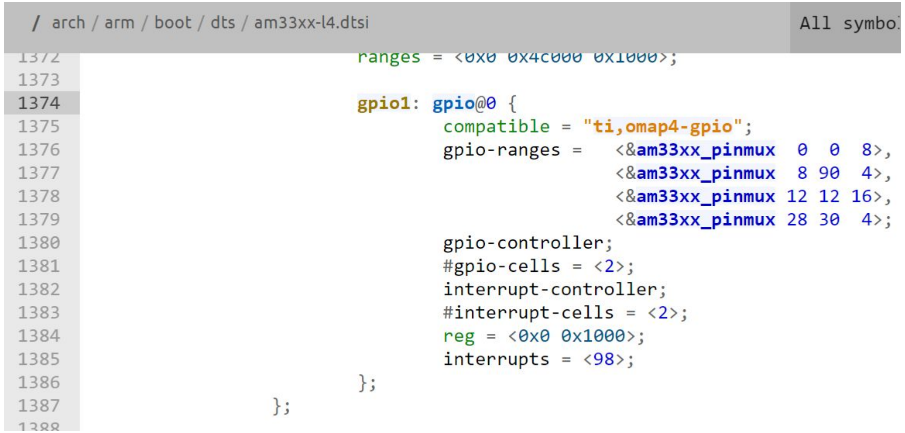
A OMAP44x system diagram

From OMAP 4 product bulletin. URL: https://www.ti.com/lit/ml/swpt034b/swpt034b.pdf?ts=1635425427118&ref_url=https%253A%252F%252Fwww.google.com%252F

Look Back At Device Tree Source Code (2)

Copying & modified from
https://www.kernel.org/doc/Documentation/devicetree/bindings/gpio/gpio-omap.txt:

```
OMAP GPIO controller bindings
Required properties:
- interrupt-controller: Mark the device node as
an interrupt controller.
- #interrupt-cells : Should be 2.
  The first cell is the GPIO number.
  The second cell is used to specify flags:
    bits[3:0] trigger type and level flags:
      1 = low-to-high edge triggered.
      2 = high-to-low edge triggered.
      4 = active high level-sensitive.
      8 = active low level-sensitive.
- interrupts : The interrupt the controller is
rising as output when an interrupt occures
```



```
                        ranges = <0x0 0x4c000 0x1000>;

                        gpio1: gpio@0 {
                                compatible = "ti,omap4-gpio";
                                gpio-ranges =   <&am33xx_pinmux  0  0  8>,
                                                <&am33xx_pinmux  8 90  4>,
                                                <&am33xx_pinmux 12 12 16>,
                                                <&am33xx_pinmux 28 30  4>;
                                gpio-controller;
                                #gpio-cells = <2>;
                                interrupt-controller;
                                #interrupt-cells = <2>;
                                reg = <0x0 0x1000>;
                                interrupts = <98>;
                        };
                };
```

Bootlin Elixir screenshot of source from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am33xx-l4.dtsi#L1374

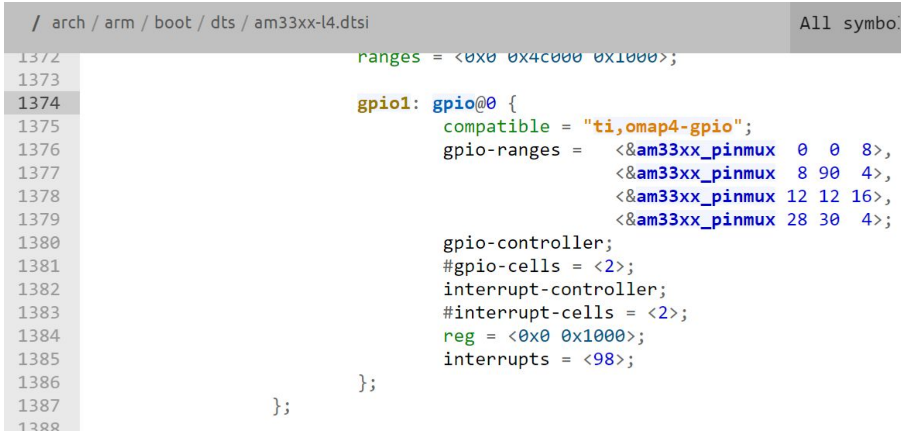
Look Back At Device Tree Source Code (3)

Copying from (with my formatting/bolding)
https://www.kernel.org/doc/Documentation/devicetree/bindings/gpio/gpio-omap.txt:

OMAP specific properties:

- **ti,hwmods**: Name of the hwmod associated to the GPIO: "gpio<X>", <X> being the 1-based instance number from the HW spec.
- **ti,gpio-always-on**: Indicates if a GPIO bank is always powered and so will never lose its logic state.

```
Example:
gpio0: gpio@44e07000 {
    compatible = "ti,omap4-gpio";
    reg = <0x44e07000 0x1000>;
    ti,hwmods = "gpio1";
    gpio-controller;
    #gpio-cells = <2>;
    interrupt-controller;
    #interrupt-cells = <2>;
    interrupts = <96>;
};
```



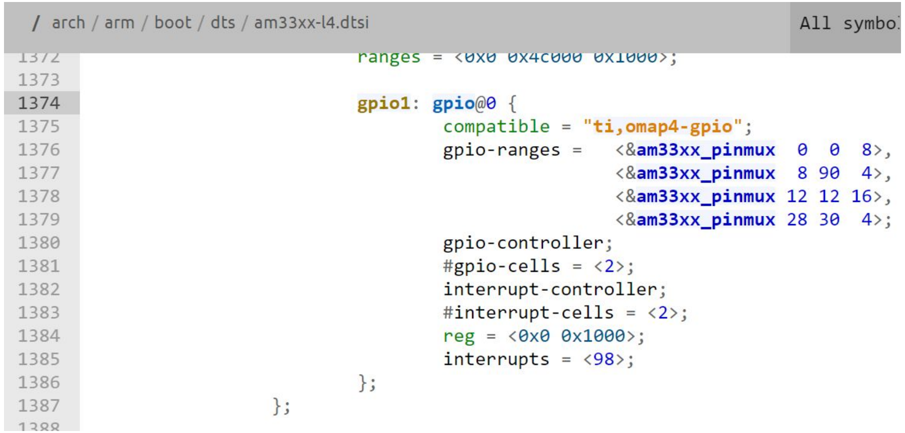
```
        ranges = <0x0 0x4c000 0x1000>;

        gpio1: gpio@0 {
                compatible = "ti,omap4-gpio";
                gpio-ranges =   <&am33xx_pinmux  0  0  8>,
                                <&am33xx_pinmux  8 90  4>,
                                <&am33xx_pinmux 12 12 16>,
                                <&am33xx_pinmux 28 30  4>;
                gpio-controller;
                #gpio-cells = <2>;
                interrupt-controller;
                #interrupt-cells = <2>;
                reg = <0x0 0x1000>;
                interrupts = <98>;
        };
};
```

Bootlin Elixir screenshot of source from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am33xx-l4.dtsi#L1374

Also Consider GPIO Binding

Copying from
https://elixir.bootlin.com/linux/latest/source/Documentation/devicetree/bindings/gpio/gpio.txt:

```
The following example could be used to describe
GPIO pins used as device enable
and bit-banged data signals:

    gpio1: gpio1 {
        gpio-controller;
        #gpio-cells = <2>;
    };
    [...]

    data-gpios = <&gpio1 12 0>,
                 <&gpio1 13 0>,
                 <&gpio1 14 0>,
                 <&gpio1 15 0>;
```

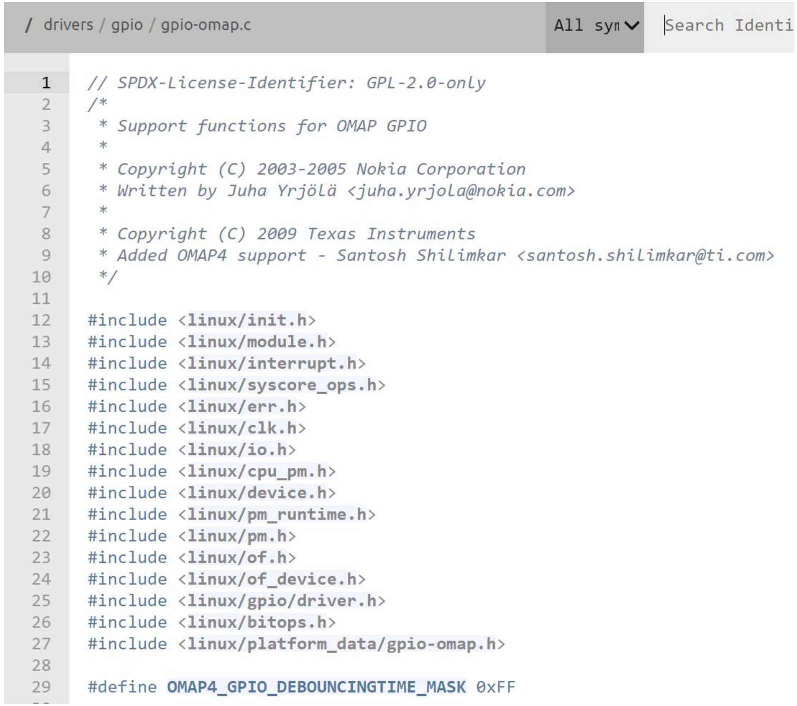


```
		ranges = <0x0 0x4c000 0x1000>;

		gpio1: gpio@0 {
			compatible = "ti,omap4-gpio";
			gpio-ranges =   <&am33xx_pinmux  0  0  8>,
					<&am33xx_pinmux  8 90  4>,
					<&am33xx_pinmux 12 12 16>,
					<&am33xx_pinmux 28 30  4>;
			gpio-controller;
			#gpio-cells = <2>;
			interrupt-controller;
			#interrupt-cells = <2>;
			reg = <0x0 0x1000>;
			interrupts = <98>;
		};
	};
```

Bootlin Elixir screenshot of source from: https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am33xx-l4.dtsi#L1374

GPIO Driver (Producer)

- **`drivers/gpio/gpio-omap.c`** (link)
- How is this linked with the gpio controller in the DT?

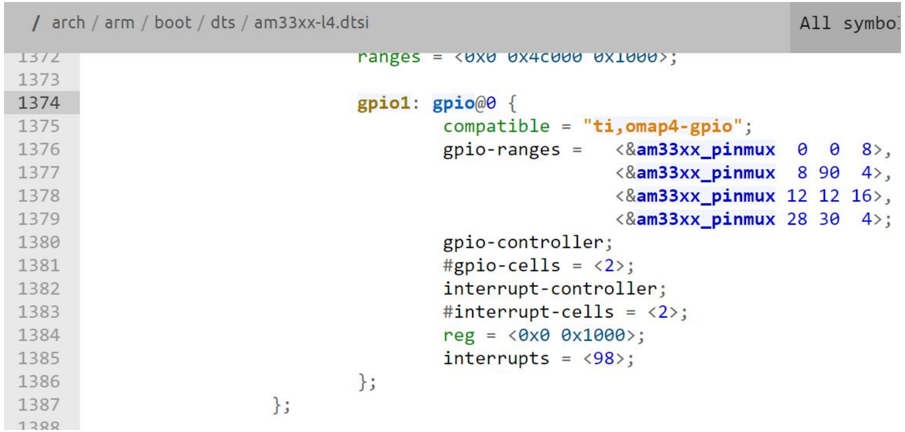


```
// SPDX-License-Identifier: GPL-2.0-only
/*
 * Support functions for OMAP GPIO
 *
 * Copyright (C) 2003-2005 Nokia Corporation
 * Written by Juha Yrjölä <juha.yrjola@nokia.com>
 *
 * Copyright (C) 2009 Texas Instruments
 * Added OMAP4 support - Santosh Shilimkar <santosh.shilimkar@ti.com>
 */

#include <linux/init.h>
#include <linux/module.h>
#include <linux/interrupt.h>
#include <linux/syscore_ops.h>
#include <linux/err.h>
#include <linux/clk.h>
#include <linux/io.h>
#include <linux/cpu_pm.h>
#include <linux/device.h>
#include <linux/pm_runtime.h>
#include <linux/pm.h>
#include <linux/of.h>
#include <linux/of_device.h>
#include <linux/gpio/driver.h>
#include <linux/bitops.h>
#include <linux/platform_data/gpio-omap.h>

#define OMAP4_GPIO_DEBOUNCINGTIME_MASK 0xFF
```

Bootlin Elixir screenshot of source from: https://elixir.bootlin.com/linux/latest/source/drivers/gpio/gpio-omap.c

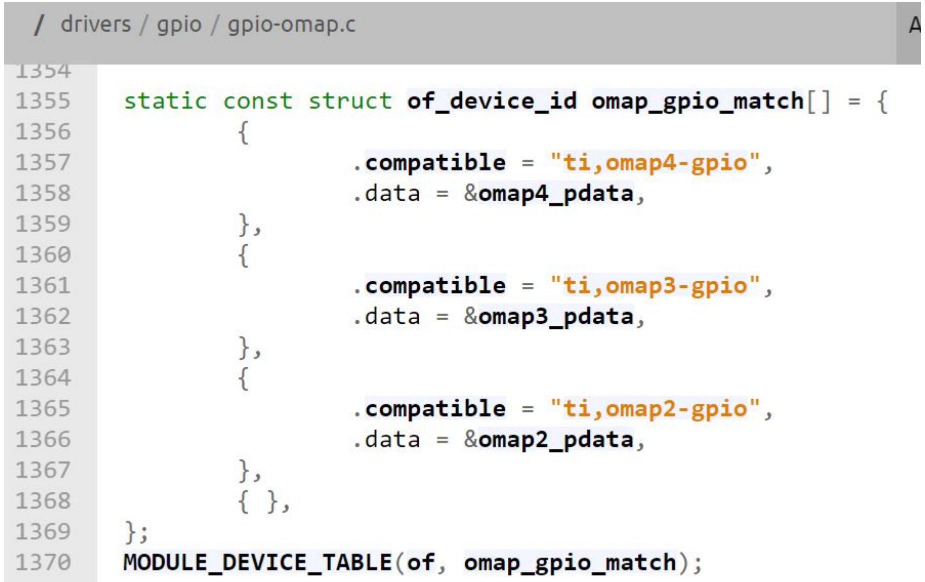
GPIO Driver (Producer)

- **`drivers/gpio/gpio-omap.c`** (link)
- How is this linked with the gpio controller in the DT?



```
                ranges = <0x0 0x4c000 0x1000>;

                gpio1: gpio@0 {
                        compatible = "ti,omap4-gpio";
                        gpio-ranges =   <&am33xx_pinmux  0  0  8>,
                                        <&am33xx_pinmux  8 90  4>,
                                        <&am33xx_pinmux 12 12 16>,
                                        <&am33xx_pinmux 28 30  4>;
                        gpio-controller;
                        #gpio-cells = <2>;
                        interrupt-controller;
                        #interrupt-cells = <2>;
                        reg = <0x0 0x1000>;
                        interrupts = <98>;
                };
        };
```

Bootlin Elixir screenshot of source from:
https://elixir.bootlin.com/linux/latest/source/arch/arm/boot/dts/am33xx-l4.dtsi#L1374



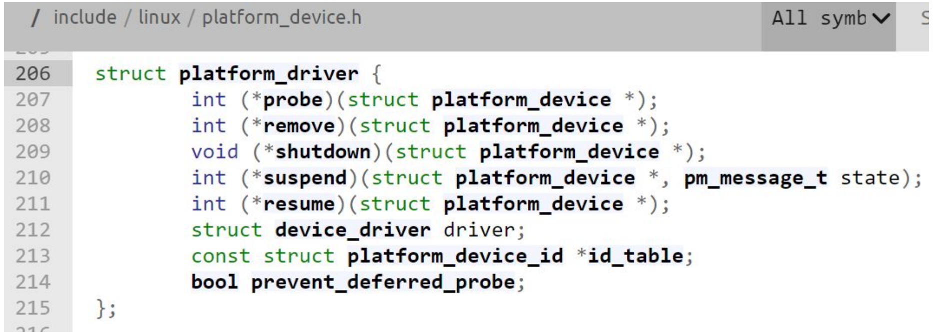
```
static const struct of_device_id omap_gpio_match[] = {
        {
                .compatible = "ti,omap4-gpio",
                .data = &omap4_pdata,
        },
        {
                .compatible = "ti,omap3-gpio",
                .data = &omap3_pdata,
        },
        {
                .compatible = "ti,omap2-gpio",
                .data = &omap2_pdata,
        },
        { },
};
MODULE_DEVICE_TABLE(of, omap_gpio_match);
```

Bootlin Elixir screenshot of source from:
https://elixir.bootlin.com/linux/latest/source/drivers/gpio/gpio-omap.c

More work must be done!

Platform Driver

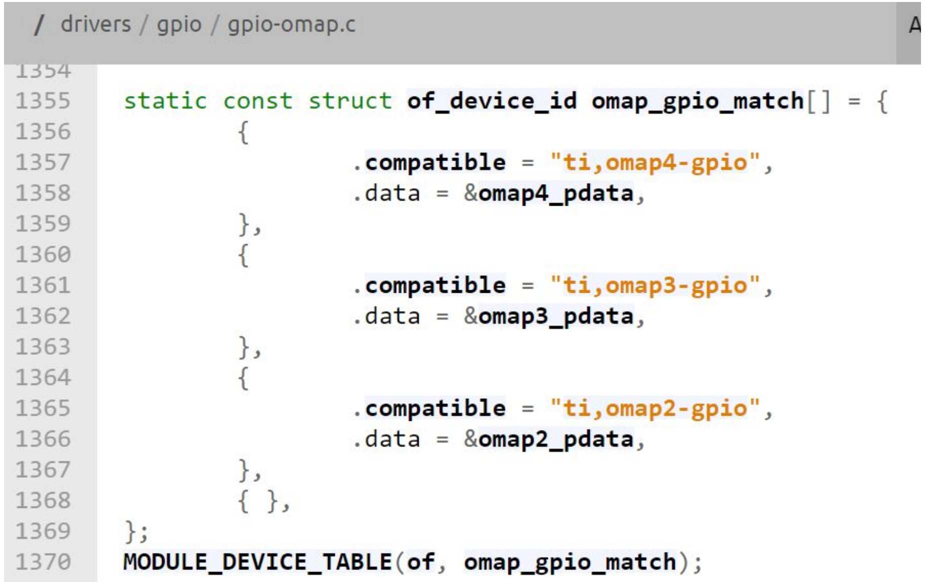
- Controllers in SoC platforms generally use platform drivers
    - Platform devices are seemingly autonomous units



/ include / linux / platform_device.h

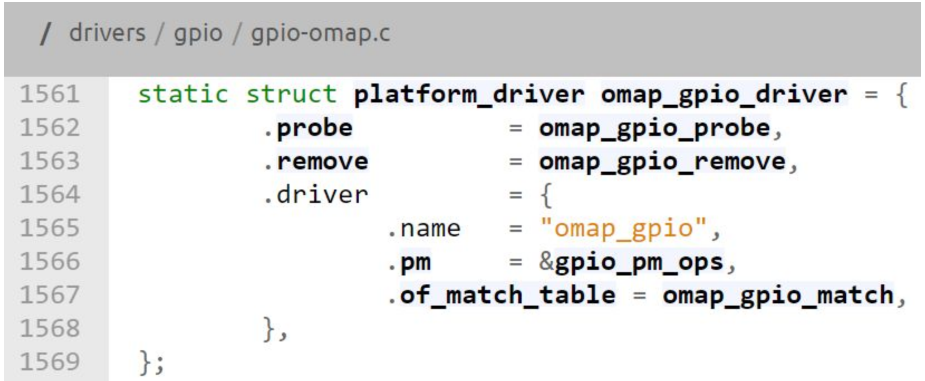
```
struct platform_driver {
        int (*probe)(struct platform_device *);
        int (*remove)(struct platform_device *);
        void (*shutdown)(struct platform_device *);
        int (*suspend)(struct platform_device *, pm_message_t state);
        int (*resume)(struct platform_device *);
        struct device_driver driver;
        const struct platform_device_id *id_table;
        bool prevent_deferred_probe;
};
```

Bootlin Elixir screenshot of source from:
https://elixir.bootlin.com/linux/latest/source/include/linux/platform_device.h#L205



/ drivers / gpio / gpio-omap.c

```
static const struct of_device_id omap_gpio_match[] = {
        {
                .compatible = "ti,omap4-gpio",
                .data = &omap4_pdata,
        },
        {
                .compatible = "ti,omap3-gpio",
                .data = &omap3_pdata,
        },
        {
                .compatible = "ti,omap2-gpio",
                .data = &omap2_pdata,
        },
        { },
};
MODULE_DEVICE_TABLE(of, omap_gpio_match);
```



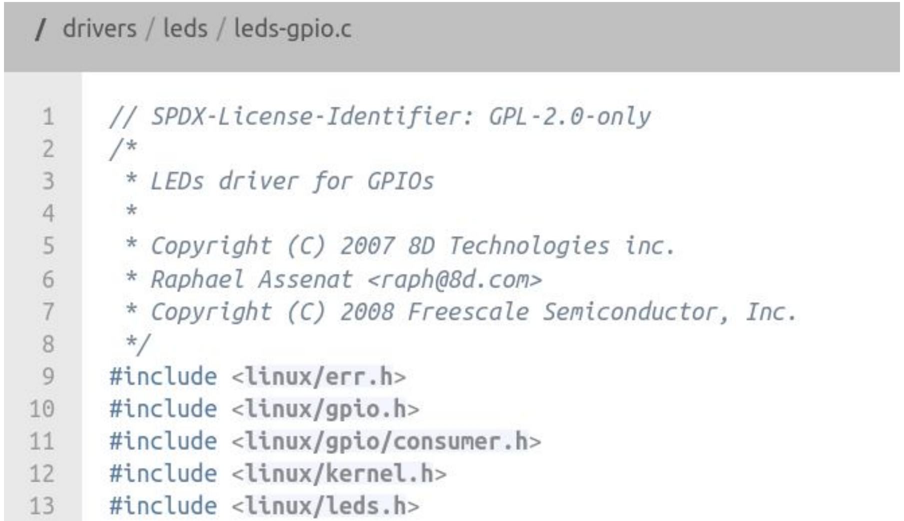
/ drivers / gpio / gpio-omap.c

```
static struct platform_driver omap_gpio_driver = {
        .probe          = omap_gpio_probe,
        .remove         = omap_gpio_remove,
        .driver         = {
                .name   = "omap_gpio",
                .pm     = &gpio_pm_ops,
                .of_match_table = omap_gpio_match,
        },
};
```

Bootlin Elixir screenshots above of source from:
https://elixir.bootlin.com/linux/latest/source/drivers/gpio/gpio-omap.c

What’s Going On For LED Driver (Consumer)?

- Check out **`drivers/leds/leds-gpio.c`** (link)



```
// SPDX-License-Identifier: GPL-2.0-only
/*
 * LEDs driver for GPIOs
 *
 * Copyright (C) 2007 8D Technologies inc.
 * Raphael Assenat <raph@8d.com>
 * Copyright (C) 2008 Freescale Semiconductor, Inc.
 */
#include <linux/err.h>
#include <linux/gpio.h>
#include <linux/gpio/consumer.h>
#include <linux/kernel.h>
#include <linux/leds.h>
```

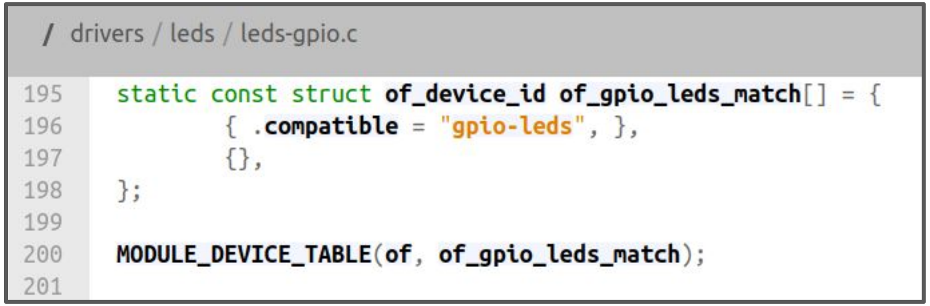
Image from: https://elixir.bootlin.com/linux/latest/source/drivers/leds/leds-gpio.c

How Does DT & Driver Matching Look For LED?

Device Tree

```
leds {
     compatible = "gpio-leds";
     pinctrl-0 = < 0x20c >;
     pinctrl-names = "default";

      ...

     led4 {
            gpios = < 0x58 0x17 0x00 >;
            label = "beaglebone:green:usr2";
            default-state = "off";
            linux,default-trigger = "cpu0";
      };
```

Driver (Consumer)

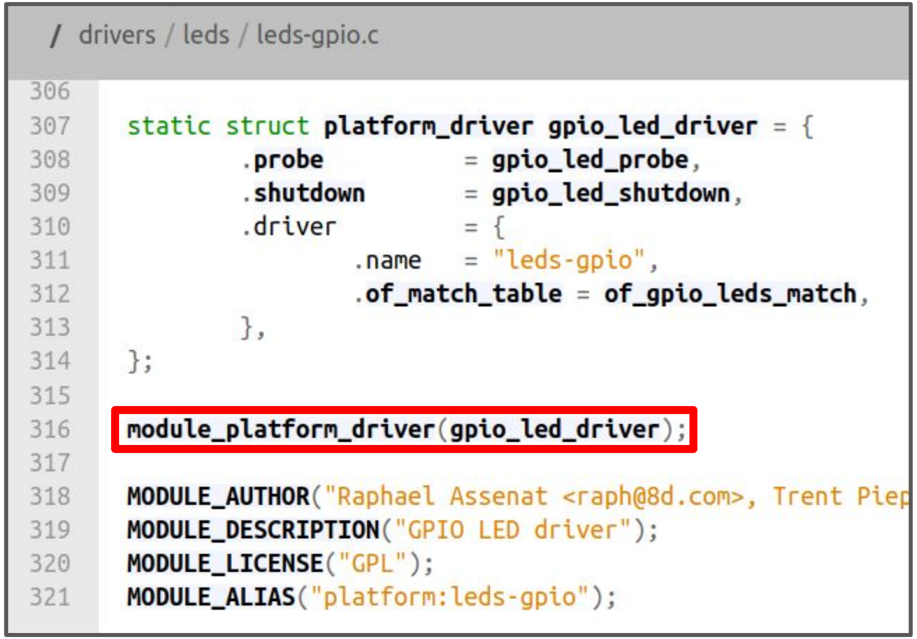


/ drivers / leds / leds-gpio.c

```
static const struct of_device_id of_gpio_leds_match[] = {
        { .compatible = "gpio-leds", },
        {},
};

MODULE_DEVICE_TABLE(of, of_gpio_leds_match);

```

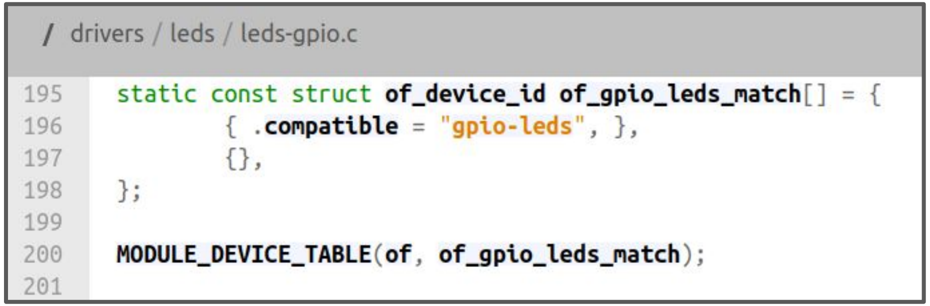
leds-gpio.c source from https://elixir.bootlin.com/linux/latest/source/drivers/leds/leds-gpio.c

Matching Device Tree Entry & Driver (Consumer)



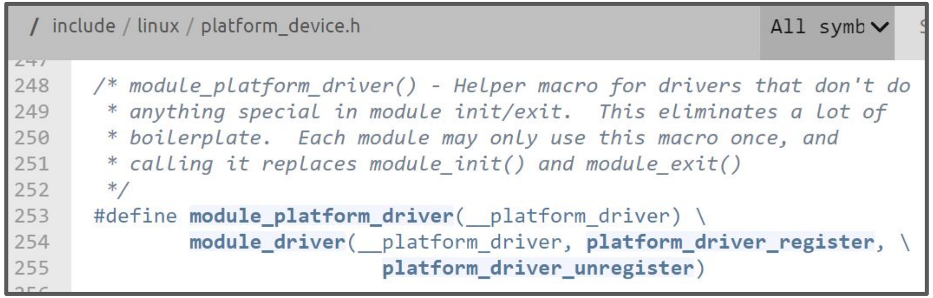
/ drivers / leds / leds-gpio.c

```

static struct platform_driver gpio_led_driver = {
	.probe		= gpio_led_probe,
	.shutdown	= gpio_led_shutdown,
	.driver		= {
		.name	= "leds-gpio",
		.of_match_table = of_gpio_leds_match,
	},
};

module_platform_driver(gpio_led_driver);

MODULE_AUTHOR("Raphael Assenat <raph@8d.com>, Trent Piep
MODULE_DESCRIPTION("GPIO LED driver");
MODULE_LICENSE("GPL");
MODULE_ALIAS("platform:leds-gpio");
```



/ drivers / leds / leds-gpio.c

```
static const struct of_device_id of_gpio_leds_match[] = {
	{ .compatible = "gpio-leds", },
	{},
};

MODULE_DEVICE_TABLE(of, of_gpio_leds_match);

```



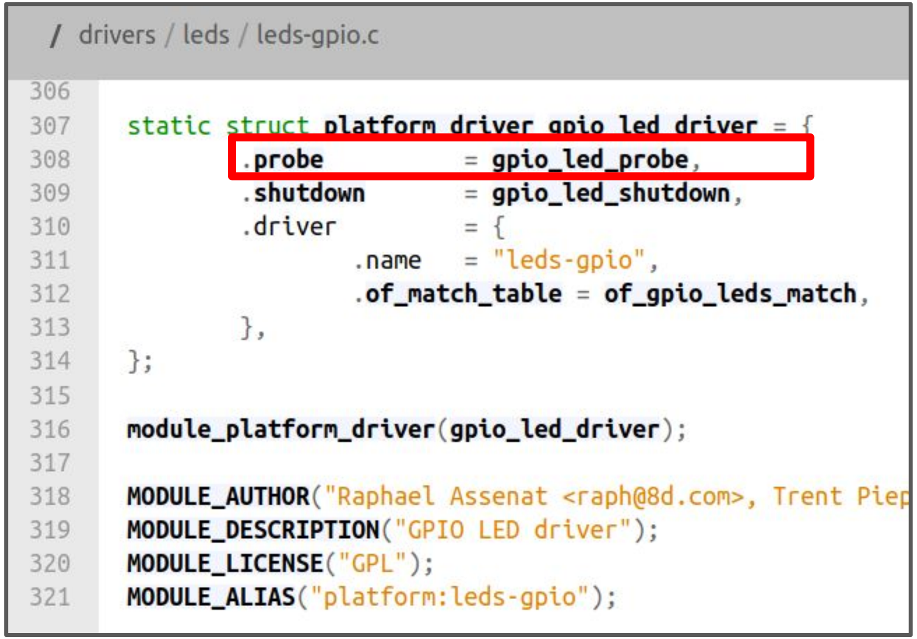
/ include / linux / platform_device.h

```
/* module_platform_driver() - Helper macro for drivers that don't do
 * anything special in module init/exit.  This eliminates a lot of
 * boilerplate.  Each module may only use this macro once, and
 * calling it replaces module_init() and module_exit()
 */
#define module_platform_driver(__platform_driver) \
	module_driver(__platform_driver, platform_driver_register, \
			platform_driver_unregister)
```

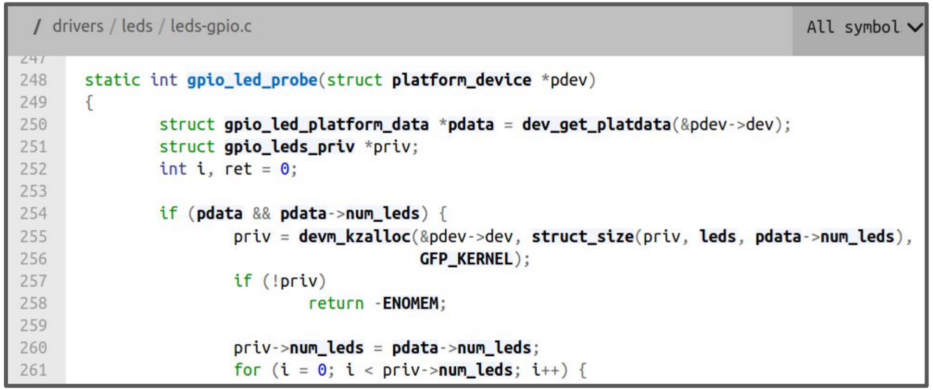
leds-gpio.c source from https://elixir.bootlin.com/linux/latest/source/drivers/leds/leds-gpio.c

Bootlin Elixir screenshot of platform_device source from:
https://elixir.bootlin.com/linux/latest/source/include/linux/platform_device.h#L205

Matching Device Tree Entry & Driver



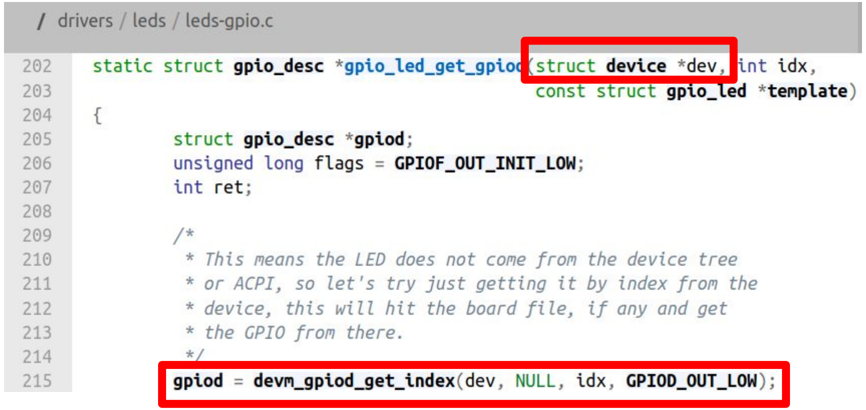
```

static struct platform_driver gpio_led_driver = {
        .probe          = gpio_led_probe,
        .shutdown       = gpio_led_shutdown,
        .driver         = {
                .name   = "leds-gpio",
                .of_match_table = of_gpio_leds_match,
        },
};

module_platform_driver(gpio_led_driver);

MODULE_AUTHOR("Raphael Assenat <raph@8d.com>, Trent Piep
MODULE_DESCRIPTION("GPIO LED driver");
MODULE_LICENSE("GPL");
MODULE_ALIAS("platform:leds-gpio");
```



```
static int gpio_led_probe(struct platform_device *pdev)
{
        struct gpio_led_platform_data *pdata = dev_get_platdata(&pdev->dev);
        struct gpio_leds_priv *priv;
        int i, ret = 0;

        if (pdata && pdata->num_leds) {
                priv = devm_kzalloc(&pdev->dev, struct_size(priv, leds, pdata->num_leds),
                                    GFP_KERNEL);
                if (!priv)
                        return -ENOMEM;

                priv->num_leds = pdata->num_leds;
                for (i = 0; i < priv->num_leds; i++) {
```

leds-gpio.c source from https://elixir.bootlin.com/linux/latest/source/drivers/leds/leds-gpio.c

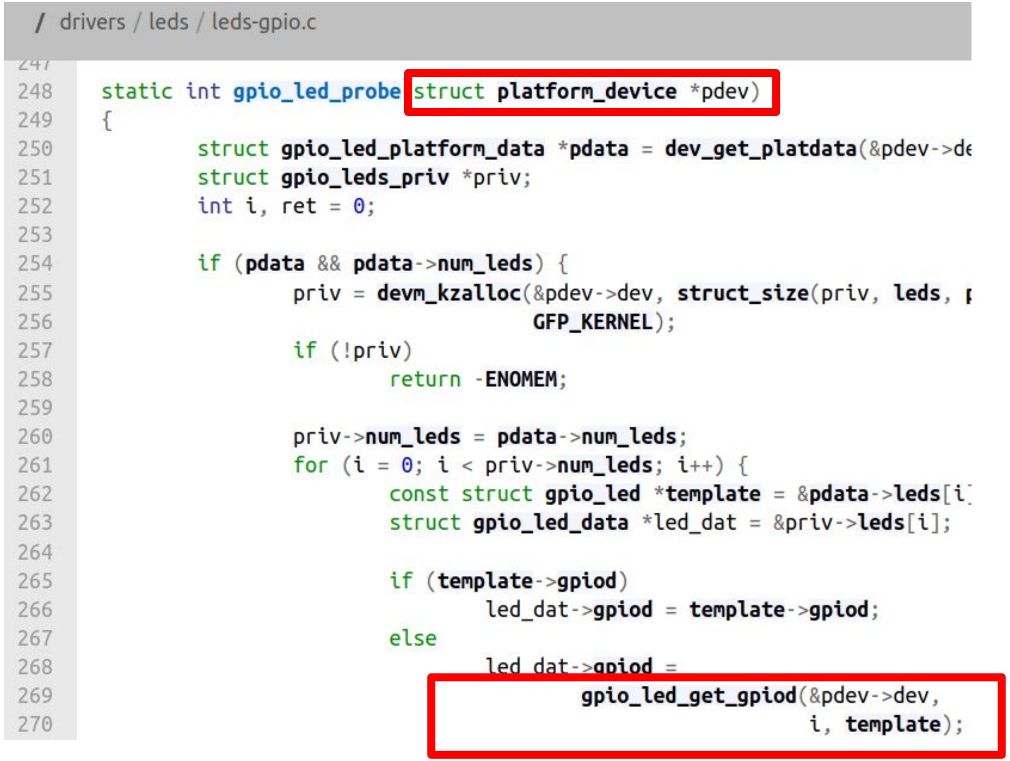
Matching Device Tree Entry & Driver



/ drivers / leds / leds-gpio.c

```
static struct gpio_desc *gpio_led_get_gpiod(struct device *dev, int idx,
                                            const struct gpio_led *template)
{
        struct gpio_desc *gpiod;
        unsigned long flags = GPIOF_OUT_INIT_LOW;
        int ret;

        /*
         * This means the LED does not come from the device tree
         * or ACPI, so let's try just getting it by index from the
         * device, this will hit the board file, if any and get
         * the GPIO from there.
         */
        gpiod = devm_gpiod_get_index(dev, NULL, idx, GPIOD_OUT_LOW);
```

We finally found **dev**



/ drivers / leds / leds-gpio.c

```
static int gpio_led_probe(struct platform_device *pdev)
{
        struct gpio_led_platform_data *pdata = dev_get_platdata(&pdev->de
        struct gpio_leds_priv *priv;
        int i, ret = 0;

        if (pdata && pdata->num_leds) {
                priv = devm_kzalloc(&pdev->dev, struct_size(priv, leds, p
                                    GFP_KERNEL);
                if (!priv)
                        return -ENOMEM;

                priv->num_leds = pdata->num_leds;
                for (i = 0; i < priv->num_leds; i++) {
                        const struct gpio_led *template = &pdata->leds[i
                        struct gpio_led_data *led_dat = &priv->leds[i];

                        if (template->gpiod)
                                led_dat->gpiod = template->gpiod;
                        else
                                led_dat->gpiod =
                                        gpio_led_get_gpiod(&pdev->dev,
                                                           i, template);
```

leds-gpio.c source from https://elixir.bootlin.com/linux/latest/source/drivers/leds/leds-gpio.c

More Device Tree: PWM