Create a file gpio\_led.dts (Device Tree Source) with the following content:

/dts-v1/;

/plugin/;

/ {

compatible = "brcm,bcm2835";

fragment@0 {

target = <&gpio>;

\_\_overlay\_\_ {

led\_gpio = <&gpio 17 0>; // GPIO 17 is used for the LED

};

Explanation:

* led\_gpio = <&gpio 17 0>;: This defines GPIO 17 to control an LED. The &gpio refers to the GPIO controller (which is a part of the Raspberry Pi's hardware).

**Step 2: Compile the Device Tree Overlay**

To compile the Device Tree source into a binary .dtbo file (Device Tree Overlay), run the following command:

dtc -@ -I dts -O dtb -o gpio\_led.dtbo gpio\_led.dts

};

};

**Step 3: Load the Device Tree Overlay**

Copy the gpio\_led.dtbo file to the /boot/overlays/ directory:

sudo cp gpio\_led.dtbo /boot/overlays/

You can load the overlay via the /boot/config.txt file by adding the following line:

dtoverlay=gpio\_led

Alternatively, you can load the overlay dynamically using the following command:

sudo dtoverlay gpio\_led

Step 4: Kernel Module to Use the GPIO from Device Tree

Now, write the kernel module that will read the GPIO pin information from the Device Tree and control the LED. Create the file gpio\_led\_driver.c:

#include <linux/module.h>

#include <linux/kernel.h>

#include <linux/init.h>

#include <linux/gpio.h>

#include <linux/of.h> // For Device Tree functions

#include <linux/of\_gpio.h> // For GPIO handling in Device Tree

static int led\_gpio = -1; // GPIO pin for LED (default -1)

static int \_\_init gpio\_led\_init(void)

{

struct device\_node \*node;

int ret;

// Get the device tree node for the GPIO

node = of\_find\_node\_by\_name(NULL, "gpio\_led");

if (!node) {

pr\_err("GPIO LED node not found in Device Tree\n");

return -ENODEV;

}

// Get the GPIO from the device tree

led\_gpio = of\_get\_named\_gpio(node, "led\_gpio", 0);

if (led\_gpio < 0) {

pr\_err("Failed to get GPIO from Device Tree\n");

return led\_gpio;

}

pr\_info("LED GPIO found: %d\n", led\_gpio);

// Request the GPIO and set it as output

ret = gpio\_request(led\_gpio, "LED GPIO");

if (ret) {

pr\_err("Failed to request GPIO %d\n", led\_gpio);

return ret;

}

ret = gpio\_direction\_output(led\_gpio, 0); // Start with LED OFF

if (ret) {

pr\_err("Failed to set GPIO direction\n");

gpio\_free(led\_gpio);

return ret;

}

// Toggle the LED (turn ON and then OFF after a delay)

gpio\_set\_value(led\_gpio, 1); // Turn LED ON

msleep(1000); // Wait for 1 second

gpio\_set\_value(led\_gpio, 0); // Turn LED OFF

return 0;

}

static void \_\_exit gpio\_led\_exit(void)

{

if (led\_gpio >= 0) {

gpio\_set\_value(led\_gpio, 0); // Ensure the LED is OFF

gpio\_free(led\_gpio); // Free the GPIO

}

pr\_info("GPIO LED driver removed\n");

}

module\_init(gpio\_led\_init);

module\_exit(gpio\_led\_exit);

MODULE\_LICENSE("GPL");

MODULE\_AUTHOR("Your Name");

MODULE\_DESCRIPTION("A simple driver for controlling GPIO LED using Device Tree");

**Step 5: Compile and Test the Kernel Module**

**1. Create a Makefile:**

obj-m += gpio\_led\_driver.o

all:

make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:

make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean

2. Compile the module:

Make

sudo insmod gpio\_led\_driver.ko

dmesg

You should see something like:

bash

Copy code

[ 124.567890] LED GPIO found: 17

[ 124.567899] GPIO LED driver loaded

sudo rmmod gpio\_led\_driver

**Verify the GPIO is Controlled**

When the module is inserted, the LED connected to GPIO 17 will turn on for 1 second and then turn off. The kernel logs will display messages confirming the operations.