Linux GPIO Framework

GPIOLIB Kernel API

Bill Gatliff
`bgat@billgatliff.com`

Freelance Embedded Systems Developer

Overview

Roadmap:

- The GPIO kernel API
- Asserting GPIO outputs, reading inputs
- Examples

Kernel vs. User API

User API:

- Attributes in subdirectories under `/sys/class/gpio`
- Must “export” and “unexport” pins
- Some pins are already spoken for by kernel drivers, etc.

Kernel API:

- Functions that are not visible to user applications
- Low-level, very close to the hardware

GPIO Channels that “Sleep”

Ways of implementing GPIO:

- SOC peripherals
- Bus expansion chips

Expansion chips:

- SPI, I2C, USB, etc.
- Usually require sleeping during communications

GPIO Channels that “Sleep”

Sleeping during communications:

- Prohibits their use from interrupt handlers!
- Prohibits their use from timer handlers!

“Does this channel sleep?”

- Test with `gpio_cansleep()`

`gpio_direction_output()`

Sets a GPIO pin for output:

- Qualify pin number with `gpio_is_valid()` first
- Asserts specified output, to prevent glitches
- Can be used to emulate an open-drain pin

```
int gpio_direction_output(unsigned gpio, int value)
```

`gpio_set_value()`

Asserts the value of a GPIO line:

- Pin must already be configured for output

```
int gpio_set_value(unsigned gpio, int value)
```

`gpio_direction_input()`

Sets a GPIO pin for input:

- Qualify pin number with `gpio_is_valid()` first
- Can also be used to emulate an open-drain pin

```
int gpio_direction_input(unsigned gpio)
```

`gpio_get_value()`

Reads the value of a pin:

- Might not match the value being asserted!

```
int gpio_get_value(unsigned gpio)
```

gpio_cansleep()

Lets the platform tell users:

- ... so users can pick an alternate API if needed

```
int gpio_cansleep(unsigned gpio)
```

gpio_[get|set]_value_cansleep()

Use in sleep-safe contexts:

- Interface implementations
- Kernel threads
- ...

```
int gpio_set_value_cansleep(unsigned gpio, int value)
int gpio_get_value_cansleep(unsigned gpio)
```

leds-gpio.c:58

```
if (led_dat->can_sleep) {
  led_dat->new_level = level;
  schedule_work(&led_dat->work);
} else
  gpio_set_value(led_dat->gpio, level);
...
led_dat->can_sleep = gpio_cansleep(cur_led->gpio);
```

`gpio_[request|free]()`

Reservation functions:

- Prevent resource conflicts
- Faciliate debugging, debugfs
- Annotation via `label` parameter

```
int gpio_request(unsigned gpio, const char *label);
void gpio_free(unsigned gpio);
```

gpiolib.c:773

GPIO numbers are of type `unsigned int`:

- Different namespace from interrupt sources
- Map between them if platform supports
- Returns -1 if the GPIO can't be used as an interrupt
- `request_irq()`

```
int gpio_to_irq(unsigned gpio);
int irq_to_gpio(unsigned irq);
```

Recap

The GPIO kernel API:

- Assert outputs, read inputs
- Check of the implementation requires sleeping

Linux GPIO Framework

GPIOLIB Kernel API

Bill Gatliff
`bgat@billgatliff.com`

Freelance Embedded Systems Developer