Overview

GPIO pins have no special purpose defined, and usually go unused by default. The idea is that sometimes the system integrator building a full system that uses the chip might find it useful to have a handful of additional digital control lines, and having these available from the chip can save the hassle of having to arrange additional circuitry to provide them.

A GPIO port is a group of GPIO pins (typically 8 GPIO pins) arranged in a group, and treated as a single port.

Accessing the GPIO pins through sysfs with mainline kernel

The GPIO pins can be accessed from user space using sysfs. To enabled this you need the following kernel option enabled: CONFIG_GPIO_SYSFS

```
Device Drivers  ---> GPIO Support  ---> /sys/class/gpio/... (sysfs interface)
```

To access a GPIO pin you first need to *export* it with

```
echo XX > /sys/class/gpio/export
```

with *XX* being the number of the desired pin. To obtain the correct number you have to calculate it from the pin name (like PH18)[1]:

```
(position of letter in alphabet - 1) * 32 + pin number
```

E.g for PH18 this would be (8 - 1) * 32 + 18 = 224 + 18 = 242 (since 'h' is the 8th letter).

Alternatively, you can read the mapping from debugfs with

```
cat /sys/kernel/debug/pinctrl/*/pinmux-pins
```

After you have successfully exported the pin you can access it through **/sys/class/gpio/gpio*NUMBER*** (in case of PH18 it's */sys/class/gpio/gpio242*).

With /sys/class/gpio/gpio*NUMBER*/direction you must set the pin to **out** or **in** using

```
echo "out" > /sys/class/gpio/gpio*NUMBER*/direction
```

and only then you can read/write the value with **/sys/class/gpio/gpio*NUMBER*/value**.

When you are done unexport the pin with

```
echo XX > /sys/class/gpio/unexport
```

Accessing the GPIO pins through character device with mainline kernel

The sysfs GPIO interface is now deprecated in favor of character devices /dev/gpiochipX

Verify that your system supports it with

```
ls /dev/gpiochip*
```

The easiest way to access the pins is with libgpiod and the set of tools it includes. https://git.kernel.org/pub/scm/libs/libgpiod/libgpiod.git/

You can read a pin with

```
sudo gpioget gpiochip0 XX
```

where XX is the pin number, calculated the same way as in the sysfs method (see section above).

Set a pin value with

```
sudo gpioset gpiochip0 XX=value
```

where value is 0 or 1. Note the value will return to default after gpioset exits. Consult *gpioset --help* for further options.

Monitor the state of a pin with

```
sudo gpiomon gpiochip0 XX
```

You can use gpioinfo to see how the pins are configured, but

```
 cat /sys/kernel/debug/pinctrl/*/pinmux-pins
```

give you also the pin mapping (see section above).