Configfs GPIO Simulator

The configfs GPIO Simulator (gpio-sim) provides a way to create simulated GPIO chips for testing purposes. The lines exposed by these chips can be accessed using the standard GPIO character device interface as well as manipulated using sysfs attributes.

Creating simulated chips

The gpio-sim module registers a configfs subsystem called `'gpio-sim'`. For details of the configfs filesystem, please refer to the configfs documentation.

The user can create a hierarchy of configfs groups and items as well as modify values of exposed attributes. Once the chip is instantiated, this hierarchy will be translated to appropriate device properties. The general structure is:

Group: `/config/gpio-sim`

This is the top directory of the gpio-sim configfs tree.

Group: `/config/gpio-sim/gpio-device`

Attribute: `/config/gpio-sim/gpio-device/dev_name`

Attribute: `/config/gpio-sim/gpio-device/live`

This is a directory representing a GPIO platform device. The `'dev_name'` attribute is read-only and allows the user-space to read the platform device name (e.g. `'gpio-sim.0'`). The `'live'` attribute allows to trigger the actual creation of the device once it's fully configured. The accepted values are: `'1'` to enable the simulated device and `'0'` to disable and tear it down.

Group: `/config/gpio-sim/gpio-device/gpio-bankX`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/chip_name`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/num_lines`

This group represents a bank of GPIOs under the top platform device. The `'chip_name'` attribute is read-only and allows the user-space to read the device name of the bank device. The `'num_lines'` attribute allows to specify the number of lines exposed by this bank.

Group: `/config/gpio-sim/gpio-device/gpio-bankX/lineY`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/name`

This group represents a single line at the offset Y. The 'name' attribute allows to set the line name as represented by the 'gpio-line-names' property.

Item: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/hog`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/hog/name`

Attribute: `/config/gpio-sim/gpio-device/gpio-bankX/lineY/hog/direction`

This item makes the gpio-sim module hog the associated line. The `'name'` attribute specifies the in-kernel consumer name to use. The `'direction'` attribute specifies the hog direction and must be one of: `'input'`, `'output-high'` and `'output-low'`.

Inside each bank directory, there's a set of attributes that can be used to configure the new chip. Additionally the user can `mkdir()` subdirectories inside the chip's directory that allow to pass additional configuration for specific lines. The name of those subdirectories must take the form of: `'line<offset>'` (e.g. `'line0'`, `'line20'`, etc.) as the name will be used by the module to assign the config to the specific line at given offset.

Once the confiuration is complete, the `'live'` attribute must be set to 1 in order to instantiate the chip. It can be set back to 0 to destroy the simulated chip. The module will synchronously wait for the new simulated device to be successfully probed and if this doesn't happen, writing to `'live'` will result in an error.

Simulated GPIO chips can also be defined in device-tree. The compatible string must be: `"gpio-simulator"`. Supported properties are:

`"gpio-sim,label"` - chip label

Other standard GPIO properties (like `"gpio-line-names"`, `"ngpios"` or `"gpio-hog"`) are also supported. Please refer to the GPIO documentation for details.

An example device-tree code defining a GPIO simulator:

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\admin-guide\gpio\(linux-master)(Documentation)(admin-guide)(gpio)gpio-sim.rst`, **line 91)**

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block :: none

    gpio-sim {
        compatible = "gpio-simulator";

        bank0 {
            gpio-controller;
            #gpio-cells = <2>;
            ngpios = <16>;
            gpio-sim,label = "dt-bank0";
            gpio-line-names = "", "sim-foo", "", "sim-bar";
        };

        bank1 {
            gpio-controller;
            #gpio-cells = <2>;
            ngpios = <8>;
            gpio-sim,label = "dt-bank1";

            line3 {
                gpio-hog;
                gpios = <3 0>;
                output-high;
                line-name = "sim-hog-from-dt";
            };
        };
    };
```

Manipulating simulated lines

Each simulated GPIO chip creates a separate sysfs group under its device directory for each exposed line (e.g. `/sys/devices/platform/gpio-sim.X/gpiochipY/`). The name of each group is of the form: `'sim_gpioX'` where X is the offset of the line. Inside each group there are two attibutes:

`pull` - allows to read and set the current simulated pull setting for every line, when writing the value must be one of: `'pull-up'`, `'pull-down'`

`value` - allows to read the current value of the line which may be different from the pull if the line is being driven from user-space