



## **SIM8905** Series Smart Module GPIO Configure Guide

Version: 2.02

Release Date: April 9, 2019

# About Document

## Document Information

Document	
Title	SIM8905 Series Smart Module GPIO Configure Guide
Version	2.02
Document Type	Application Note
Document Status	Released/Confidential

## Revision History

Revision	Date	Owner	Status / Comments
1.00	April 8, 2018	Cheng.Zhou	First Release.
2.00	May14, 2018	Liuyuan.Zhang	Second Release
2.02	April 4, 2019	Liuyuan.Zhang	Third Release

## Related Documents

### This document applies to the following products:

Name	Type	Size (mm)	Comments
SIM8905A/E/AU	Smart Module	40.5*40.5*2.8	N/A

## Copyrights

This document contains proprietary technical information which is the property of SIMCom Wireless. Copying of this document and giving it to others and the using or communication of the contents thereof, are forbidden without express authority. Offenders are liable to the payment of damages. All rights reserved in the event of grant of a patent or the registration of a utility model or design. All specification supplied herein are subject to change without notice at any time.

# Contents

<b>About Document .....</b>	<b>2</b>
Document Information.....	2
Revision History.....	2
Related Documents .....	2
<b>Contents.....</b>	<b>3</b>
<b>1 Purpose of this document.....</b>	<b>4</b>
<b>2 Configure GPIO Process .....</b>	<b>4</b>
2.1 Configure Dts File .....	4
2.2 Add Driver File.....	5
<b>Contact.....</b>	<b>9</b>

# 1 Purpose of this document

General Purpose Input Output (general input / output) is referred to as GPIO . This document directs how to configure GPIO for SIM8905 series smart module.

## 2 Configure GPIO Process

### 2.1 Configure Dts File

The main steps to add GPIO is to modify DTS files and drivers. For example, in order to add a new GPIO2, the DTS file that needs to be modified is as follows:

kernel/arch/arm/boot/dts/qcom/msm8909.dtsi

```
sim8905_gpio {
    qcom,pins = <&gp 2>;
    qcom,num-grp-pins = <1>;
    qcom,pin-func = <0>;
    label = "sim8905_gpio";
    sim8905_act: active {
        drive-strength = <2>;
        bias-disable = <0>;
    };
};

sim8905_gpio {
    compatible = "sim8905_gpio";
    sim8905_gpio,gpio2 = <&msm_gpio 2 0>;
    qcom,num-grp-pins = <1>;
    qcom,pin-func = <0>;
    pinctrl-names = "sim8905_gpio";
    pinctrl-0 = <&sim8905_act>;
    label = "sim8905_gpio";
    sim8905_act: active {
        drive-strength = <2>;
        bias-disable = <0>;
    };
};
```

## 2.2 Add Driver File

The driver file `sim8905_gpio.c` needs to be added in the path: `/kernel/drivers/misc/`

For example:

```
#include <linux/types.h>
#include <linux/pm.h>
#include <linux/gpio.h>
#include <linux/slab.h>
#include <linux/init.h>
#include <linux/platform_device.h>
#include <linux/fsl_devices.h>
#include <asm/setup.h>
#include <linux/of.h>
#include <linux/of_gpio.h>
#include <linux/stat.h>
#include <linux/module.h>
#include <linux/err.h>
#include <linux/spinlock.h>
#include <linux/err.h>
#include <linux/regulator/consumer.h>

int sim8905_gpio2 = -1;

static struct class * sim8905_gpios_class = NULL;
static struct device * sim8905_gpio2_dev = NULL;

#define CTL_POWER_ON    "1"
#define CTL_POWER_OFF   "0"

static ssize_t sim8905_gpio2_show(struct device *dev, struct device_attribute *attr, char *buf)
{
    int gpio_state = gpio_get_value(sim8905_gpio2);
    sprintf(buf, "%d\n", gpio_state);
    return strlen(buf);
}

static ssize_t sim8905_gpio2_store(struct device *dev,
    struct device_attribute *attr, const char *buf,
    size_t count)
{
    if(!strcmp(buf, CTL_POWER_ON, strlen(CTL_POWER_ON))) {
        gpio_set_value(sim8905_gpio2, 1);
    } else if(!strcmp(buf, CTL_POWER_OFF, strlen(CTL_POWER_OFF))) {
```

```

        gpio_set_value(sim8905_gpio2, 0);
    }
    return count;
}

static struct device_attribute sim8905_gpio2_dev_attr = {
    .attr = {
        .name = "state",
        .mode = S_IRWXU|S_IRWXG|S_IRWXO,
    },
    .show = sim8905_gpio2_show,
    .store = sim8905_gpio2_store,
};

static int sim8905_gpio_probe(struct platform_device *pdev)
{
    int ret = 0;
    printk("zc enter sim8905_gpio_probe \n");
    sim8905_gpio0 = of_get_named_gpio(pdev->dev.of_node, "qcom, sim8905_gpio0", 0);

    sim8905_gpio2 = of_get_named_gpio(pdev->dev.of_node, "qcom, sim8905_gpio2", 0);

    if (sim8905_gpio2 < 0)
        printk("sim8905_gpio2 is not available \n");

    ret = gpio_request(sim8905_gpio2, " sim8905_gpio2");
    if(0 != ret) {
        printk("gpio request %d failed.", sim8905_gpio2);
        goto fail1;
    }

    //Set output
    gpio_direction_output(sim8905_gpio2, 0);
    //node create
    sim8905_gpios_class = class_create(THIS_MODULE, " sim8905_gpios");
    if(IS_ERR(sim8905_gpios_class))
    {
        ret = PTR_ERR(sim8905_gpios_class);
        printk("Failed to create class.\n");
        return ret;
    }

    sim8905_gpio2_dev=device_create(sim8905_gpios_class, NULL, 0, NULL, " sim8905_gpio2");

```

```
if (IS_ERR(sim8905_gpio2_dev))
{
    ret = PTR_ERR(sim8905_gpios_class);
    printk("Failed to create device(sim8905_gpio2_dev)!\n");
    return ret;
}
ret = device_create_file(sim8905_gpio2_dev, & sim8905_gpio2_dev_attr);
if(ret)
{
    pr_err("%s: sim8905_gpio2 creat sysfs failed\n",__func__);
    return ret;
}
printk("enter sim8905_gpio_probe, ok \n");
fail1:
    return ret;
}

static int sim8905_gpio_remove(struct platform_device *pdev)
{
    device_destroy(sim8905_gpios_class, 0);
    class_destroy(sim8905_gpios_class);
    device_remove_file(sim8905_gpio0_dev, & sim8905_gpio0_dev_attr);
    return 0;
}

static int sim8905_gpio_suspend(struct platform_device *pdev,pm_message_t state)
{
    return 0;
}

static int sim8905_gpio_resume(struct platform_device *pdev)
{
    return 0;
}

static struct of_device_id sim8905_gpio_dt_match[] = {
    { .compatible = " sim8905_gpio",},
    { },
};
MODULE_DEVICE_TABLE(of, sim8905_gpio_dt_match);

static struct platform_driver gpio_power_driver = {
    .driver = {
        .name = " sim8905_gpio",
```



```
.owner = THIS_MODULE,
.of_match_table = of_match_ptr(sim8905_gpio_dt_match),
},
.probe = sim8905_gpio_probe,
.remove = sim8905_gpio_remove,
.suspend = sim8905_gpio_suspend,
.resume = sim8905_gpio_resume,
};

static __init int gpio_power_init(void)
{
    return platform_driver_register(&gpio_power_driver);
}

static void __exit gpio_power_exit(void)
{
    platform_driver_unregister(&gpio_power_driver);
}

module_init(gpio_power_init);
module_exit(gpio_power_exit);
MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("sim8905_gpio Driver");
MODULE_AUTHOR("vicent.zhou <vicent.zhou@google.com>");
```



# Contact

## Headquarters

Add: Building B, No.633 Jinzhong Road, Changning District, Shanghai P.R.China 200335

Tel: +86-21-31575100

Fax: +86 21 31575200

Email: [support@simcom.com](mailto:support@simcom.com)

## Technical Support

### EMEA

#### West Europe

[we-support@simcom.com](mailto:we-support@simcom.com)

#### East Europe

[ee-support@simcom.com](mailto:ee-support@simcom.com)

#### Middle East

[me-support@simcom.com](mailto:me-support@simcom.com)

#### Africa

[af-support@simcom.com](mailto:af-support@simcom.com)

### APAC

#### ASEAN

[asean-support@simcom.com](mailto:asean-support@simcom.com)

#### Australia and New Zealand

[anz-support@simcom.com](mailto:anz-support@simcom.com)

#### Big China

[China-support@simcom.com](mailto:China-support@simcom.com)

### America

#### North America

[us-support@simcom.com](mailto:us-support@simcom.com)

#### Central and South America

[la-support@simcom.com](mailto:la-support@simcom.com)