



# RTOS GPIO

## 开发指南

版本号: 1.0  
发布日期: 2020.7.9

## 版本历史

版本号	日期	制/修订人	内容描述
1.0	2020.7.9	Allwinner	1. 初版



# 目 录

<b>1 前言</b>	<b>1</b>
1.1 文档简介 . . . . .	1
1.2 目标读者 . . . . .	1
1.3 适用范围 . . . . .	1
<b>2 模块介绍</b>	<b>2</b>
2.1 模块功能介绍 . . . . .	2
2.2 相关术语介绍 . . . . .	2
2.2.1 硬件术语 . . . . .	2
2.2.2 软件术语 . . . . .	3
2.3 模块配置介绍 . . . . .	3
2.3.1 platform 配置说明 . . . . .	3
2.3.2 kernel menuconfig 配置说明 . . . . .	3
2.4 源码结构介绍 . . . . .	4
2.5 驱动框架介绍 . . . . .	4
<b>3 模块接口说明</b>	<b>5</b>
3.1 数据结构 . . . . .	5
3.1.1 引脚定义 gpio_pin_t . . . . .	5
3.1.2 引脚驱动能力 gpio_driving_level_t . . . . .	6
3.1.3 引脚上下拉 gpio_pull_status_t . . . . .	6
3.1.4 引脚数据 gpio_data_t . . . . .	6
3.1.5 引脚电压能力 gpio_power_mode_t . . . . .	7
3.1.6 中断模式 gpio_interrupt_mode_t . . . . .	7
3.2 hal_gpio_check_valid . . . . .	7
3.3 hal_gpio_get_data . . . . .	7
3.4 hal_gpio_set_data . . . . .	8
3.5 hal_gpio_set_direction . . . . .	8
3.6 hal_gpio_get_direction . . . . .	8
3.7 hal_gpio_set_pull . . . . .	9
3.8 hal_gpio_get_pull . . . . .	9
3.9 hal_gpio_set_driving_level . . . . .	9
3.10 hal_gpio_get_driving_level . . . . .	9
3.11 hal_gpio_pinmux_set_function . . . . .	10
3.12 hal_gpio_sel_vol_mode . . . . .	10
3.13 hal_gpio_set_debounce . . . . .	10
3.14 hal_gpio_to_irq . . . . .	11
3.15 hal_gpio_irq_request . . . . .	11
3.16 hal_gpio_irq_free . . . . .	11
3.17 hal_gpio_irq_enable . . . . .	12
3.18 hal_gpio_irq_disable . . . . .	12
3.19 hal_gpio_init . . . . .	12

**4 模块使用范例****13****5 FAQ****14**

# 1 前言

## 1.1 文档简介

介绍 RTOS 中 GPIO 驱动的接口及使用方法，为 GPIO 的使用者提供参考。

## 1.2 目标读者

GPIO 驱动、及应用层的开发/维护人员。

## 1.3 适用范围

表 1-1: 适用产品列表

产品名称	内核版本   驱	动文件
V459	Melis	hal_gpio.c
F133	Melis	hal_gpio.c
R328	FreeRTOS	hal_gpio.c
R329-DSP	FreeRTOS	hal_gpio.c

## 2 模块介绍

### 2.1 模块功能介绍

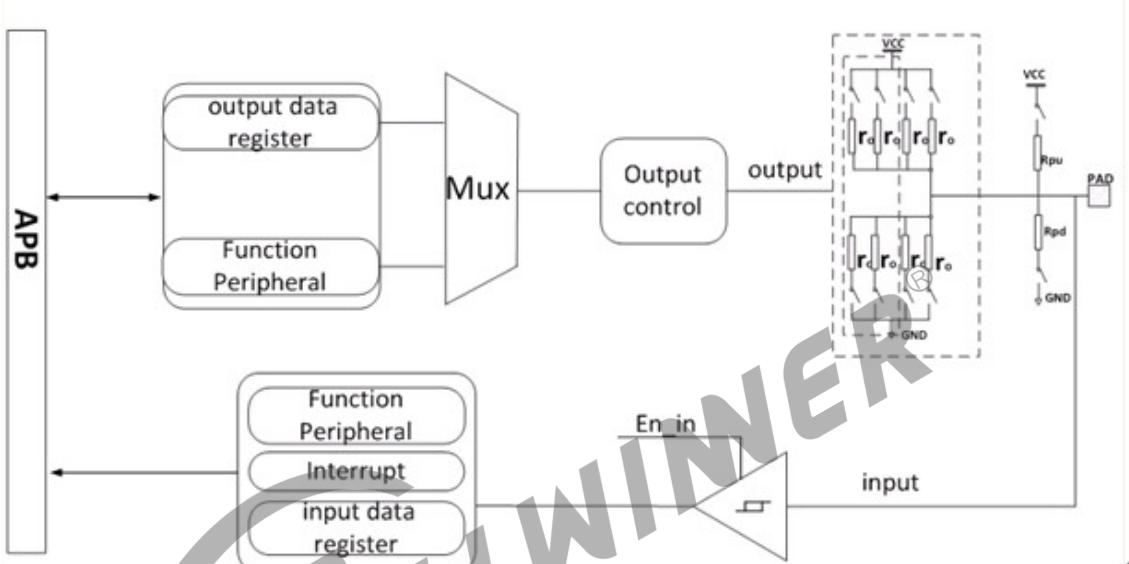


图 2-1: PORT 控制器

整个 GPIO 控制器由数字部分（GPIO 和外设接口）以及 IO 模拟部分（输出缓冲，双下拉，pad）组成。其中数字部分的输出可以通过 MUX 开关选择，模拟部分可以用来配置上下拉，驱动能力以及引脚输出电压等等。具体的规格如下：

- 可以在软件上配置各个引脚的状态
- 每个引脚都可以触发中断
- 可以配置上拉/下拉/无上下拉三种状态
- 每个引脚都可以配置 4 种驱动能力
- 可以配置边缘中断触发
- 最高 99 个中断

### 2.2 相关术语介绍

#### 2.2.1 硬件术语

---

术语	解释说明
MUX	multiplexer, 数据选择器

---

## 2.2.2 软件术语

---

术语	解释说明
HAL	Hardware Abstraction Layer, 硬件抽象层
RTOS	Real Time Operatiing System, 实时操作系统
GPIO	General Purpose Input/Output, 通用输入输出

---

## 2.3 模块配置介绍

### 2.3.1 platform 配置说明

在不同的 Sunxi 硬件平台中，GPIO 的设计有差别，但平台配置文件的信息基本类似，如下：

```
hal/source/gpio/sun8iw18/
└── gpio-sun8iw18.c
└── Makefile
└── platform-gpio.h

hal/source/gpio/sun8iw19/
└── gpio-sun8iw19.c
└── Makefile
└── platform-gpio.h
```

其中：

1. platform-gpio.h 主要包含 GPIO 控制器基地址、GPIO 中断号、pin 的声明等信息，这部分参考 GPIO SPEC 即可。
2. gpio-sunXX.c 主要包含每个平台的 GPIO 描述符配置，这部分也是参考 GPIO SPEC 即可。

### 2.3.2 kernel menuconfig 配置说明

```
[*] enable gpio driver
[*]   enable gpio hal APIs test command
```

图 2-2: GPIO menuconfig

## 2.4 源码结构介绍

```
hal/source/gpio/ ---- 驱动源码
├── gpio.h
├── hal_gpio.c
├── Kconfig
├── Makefile
└── sun8iw18
    ├── gpio-sun8iw18.c
    ├── Makefile
    └── platform-gpio.h
└── sun8iw19
    ├── gpio-sun8iw19.c
    ├── Makefile
    └── platform-gpio.h

include/hal/ ---- 驱动APIs声明头文件
└── hal_gpio.h

hal/test/gpio/ ---- 驱动APIs测试代码
├── Makefile
├── test_conf.h
└── test_gpio.c
```

## 2.5 驱动框架介绍

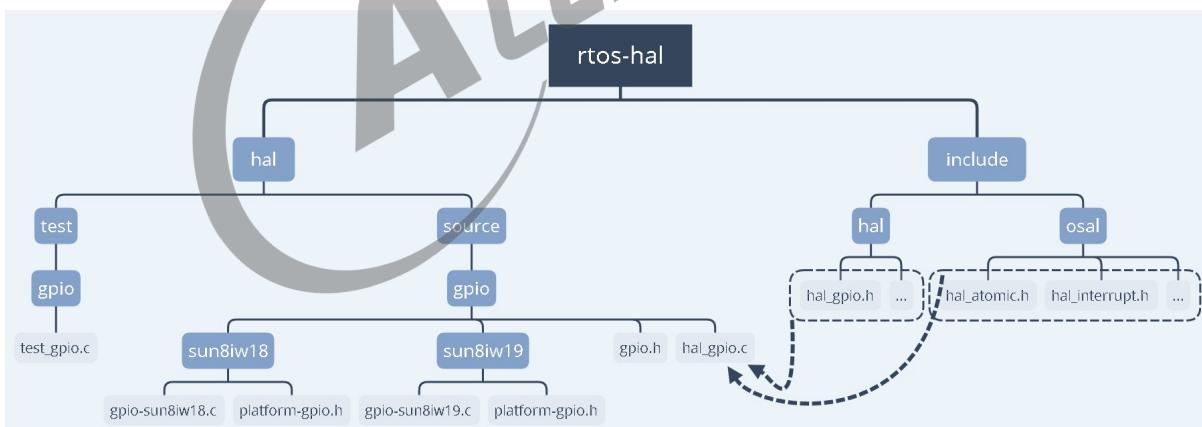


图 2-3: GPIO 驱动框架

## 3 模块接口说明

API	解释说明
hal_gpio_check_valid	判断 GPIO 的合法性
hal_gpio_get_data	获取指定 GPIO 的电平状态
hal_gpio_set_data	设置指定 GPIO 的电平状态
hal_gpio_set_direction	设置指定 GPIO 的 IO 模式
hal_gpio_get_direction	获取指定 GPIO 的 IO 模式
hal_gpio_set_pull	设置指定 GPIO 的上下拉状态
hal_gpio_get_pull	获取指定 GPIO 的上下拉状态
hal_gpio_set_driving_level	设置指定 GPIO 的驱动能力
hal_gpio_get_driving_level	获取指定 GPIO 的驱动能力
hal_gpio_pinmux_set_function	设置指定 GPIO 的复用功能
hal_gpio_sel_vol_mode	设置指定 GPIO 组的电压模式
hal_gpio_set_debounce	设置指定 GPIO 组的中断采样频率
hal_gpio_to_irq	获取指定 GPIO 的 IRQ 中断号
hal_gpio_irq_request	GPIO 中断申请
hal_gpio_irq_free	GPIO 中断释放
hal_gpio_irq_enable	使能 GPIO 中断
hal_gpio_irq_disable	关闭 GPIO 中断
hal_gpio_init	GPIO 驱动初始化

### 3.1 数据结构

由于 GPIO 需要配置每个引脚的引脚复用功能，中断类型，驱动能力，上下拉，输出/输入数据，输入/输出方向等等，所以对 GPIO 的这些配置都封装在一个 enum 枚举结构里面，方便使用。下面是一些配置的定义。想要了解更多的可以到 hal\_gpio.h 查看

#### 3.1.1 引脚定义 gpio\_pin\_t

该枚举定义了可用的每个引脚定义，在配置引脚的时候将相关参数传入则可，具体定义如下：

```
typedef enum
{
    GPIO_PC0 = GPIOC(0),
```

```
GPIO_PC1 = GPIOC(1),  
GPIO_PC2 = GPIOC(2),  
GPIO_PC3 = GPIOC(3),  
...  
GPIO_PL0 = GPIOL(0),  
GPIO_PL1 = GPIOL(1),  
GPIO_PL2 = GPIOL(2),  
GPIO_PL3 = GPIOL(3),  
GPIO_PL4 = GPIOL(4),  
GPIO_PL5 = GPIOL(5),  
} gpio_pin_t;
```

### 3.1.2 引脚驱动能力 gpio\_driving\_level\_t

该枚举定义了引脚的驱动能力的值，具体定义如下：

```
typedef enum  
{  
    GPIO_DRIVING_LEVEL0      = 0,          /**< Defines GPIO driving current as level0. */  
    GPIO_DRIVING_LEVEL1      = 1,          /**< Defines GPIO driving current as level1. */  
    GPIO_DRIVING_LEVEL2      = 2,          /**< Defines GPIO driving current as level2. */  
    GPIO_DRIVING_LEVEL3      = 3,          /**< Defines GPIO driving current as level3. */  
} gpio_driving_level_t;
```

### 3.1.3 引脚上下拉 gpio\_pull\_status\_t

该枚举定义了引脚的上下拉的值，具体定义如下：

```
typedef enum  
{  
    GPIO_PULL_DOWN_DISABLED   = 0,          /**< Defines GPIO pull up and pull down disable.  
     * /  
    GPIO_PULL_UP              = 1,          /**< Defines GPIO is pull up state. */  
    GPIO_PULL_DOWN             = 2,          /**< Defines GPIO is pull down state. */  
} gpio_pull_status_t;
```

### 3.1.4 引脚数据 gpio\_data\_t

该枚举定义引脚的输入输出数据，具体定义如下：

```
typedef enum  
{  
    GPIO_DATA_LOW   = 0,          /**< GPIO data low. */  
    GPIO_DATA_HIGH  = 1,          /**< GPIO data high. */  
} gpio_data_t;
```

### 3.1.5 引脚电压能力 gpio\_power\_mode\_t

该枚举定义了引脚的电压模式，可以配置成 1.8V 和 3.3V，具体定义如下

```
typedef enum
{
    POWER_MODE_330 = 0,
    POWER_MODE_180 = 1
} gpio_power_mode_t;
```

### 3.1.6 中断模式 gpio\_interrupt\_mode\_t

该枚举定义了引脚的中断模式，具体定义如下：

```
typedef enum
{
    IRQ_TYPE_NONE      = 0x00000000,
    IRQ_TYPE_EDGE_RISING = 0x00000001,
    IRQ_TYPE_EDGE_FALLING = 0x00000002,
    IRQ_TYPE_EDGE_BOTH  = (IRQ_TYPE_EDGE_FALLING | IRQ_TYPE_EDGE_RISING),
    IRQ_TYPE_LEVEL_HIGH = 0x00000004,
    IRQ_TYPE_LEVEL_LOW  = 0x00000008,
} gpio_interrupt_mode_t;
```

## 3.2 hal\_gpio\_check\_valid

- 作用：判断 GPIO 的合法性
- 参数：
  - pin id
- 返回：
  - true: 合法
  - false: 不合法

## 3.3 hal\_gpio\_get\_data

- 作用：获取指定 GPIO 的电平状态
- 参数：
  - 参数 1:pin id
  - 参数 2: 存放频率的指针变量

- 返回：
  - -1: 失败
  - 0: 成功

### 3.4 hal\_gpio\_set\_data

- 作用：设置指定 GPIO 的电平状态
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的电平高低
- 返回：
  - -1: 失败
  - 0: 成功

### 3.5 hal\_gpio\_set\_direction

- 作用：设置指定 GPIO 的 IO 模式
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的 IO 模式
- 返回：
  - -1: 失败
  - 0: 成功

### 3.6 hal\_gpio\_get\_direction

- 作用：获取指定 GPIO 的 IO 模式
- 参数：
  - 参数 1:pin id
  - 参数 2: 存放 IO 模式的指针变量
- 返回：
  - -1: 失败
  - 0: 成功

### 3.7 hal\_gpio\_set\_pull

- 作用：设置指定 GPIO 的上下拉状态
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的上下拉状态
- 返回：
  - -1: 失败
  - 0: 成功

### 3.8 hal\_gpio\_get\_pull

- 作用：获取指定 GPIO 的上下拉状态
- 参数：
  - 参数 1:pin id
  - 参数 2: 存放上下拉状态的指针变量
- 返回：
  - -1: 失败
  - 0: 成功

### 3.9 hal\_gpio\_set\_driving\_level

- 作用：设置指定 GPIO 的驱动能力
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的驱动能力
- 返回：
  - -1: 失败
  - 0: 成功

### 3.10 hal\_gpio\_get\_driving\_level

- 作用：获取指定 GPIO 的驱动能力

- 参数：
  - 参数 1:pin id
  - 参数 2: 存放驱动能力的指针变量
- 返回：
  - -1: 失败
  - 0: 成功

### 3.11 hal\_gpio\_pinmux\_set\_function

- 作用：设置指定 GPIO 的复用功能
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的复用功能
- 返回：
  - -1: 失败
  - 0: 成功

### 3.12 hal\_gpio\_sel\_vol\_mode

- 作用：设置指定 GPIO 组的电压模式
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的电压模式
- 返回：
  - -1: 失败
  - 0: 成功

### 3.13 hal\_gpio\_set\_debounce

- 作用：设置指定 GPIO 组的中断采样频率
- 参数：
  - 参数 1:pin id
  - 参数 2: 需设置的值 (bit0-clock select; bit6:4-clock pre-scale)

- 返回：
  - -1: 失败
  - 0: 成功

### 3.14 hal\_gpio\_to\_irq

- 作用：获取指定 GPIO 的 IRQ 中断号
- 参数：
  - 参数 1: pin id
  - 参数 2: 存放中断号的指针变量
- 返回：
  - -1: 失败
  - 0: 成功

### 3.15 hal\_gpio\_irq\_request

- 作用：GPIO 中断申请
- 参数：
  - 参数 1: 中断号
  - 参数 2: 中断处理函数
  - 参数 3: 中断触发模式
  - 参数 4: 私有数据指针变量
- 返回：
  - -1: 失败
  - 0: 成功

### 3.16 hal\_gpio\_irq\_free

- 作用：GPIO 中断释放
- 参数：
  - 参数 1: 中断号
- 返回：
  - -1: 失败
  - irq: 成功释放的 irq 号

### 3.17 hal\_gpio\_irq\_enable

- 作用：使能 GPIO 中断
- 参数：
  - 参数 1：中断号
- 返回：
  - -1：失败
  - 0：成功

### 3.18 hal\_gpio\_irq\_disable

- 作用：关闭 GPIO 中断
- 参数：
  - 参数 1：中断号
- 返回：
  - -1：失败
  - 0：成功

### 3.19 hal\_gpio\_init

- 作用：GPIO 驱动初始化
- 参数：
  - void
- 返回：
  - -1：失败
  - 0：成功

## 4 模块使用范例

可参考驱动 APIs 测试代码 (hal/test/gpio/)。



## 5 FAQ

无



## 著作权声明

版权所有 © 2021 珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

## 商标声明

  **全志科技**  (不完全列举) 均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

## 免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本文档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。