|  |
| --- |
| OpenCPU |
| SDK IO API 说明 |
| 版本1.1日期2025-02-26 |

版权声明

版权所有 © 深圳市有方科技股份有限公司2025。深圳市有方科技股份有限公司保留所有权利。

未经深圳市有方科技股份有限公司书面同意，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

是深圳市有方科技股份有限公司所有商标。

本文档中出现的其他商标，由商标所有者所有。

说明

本文档的使用对象为系统工程师，开发工程师及测试工程师。

本设计指南为客户产品设计提供支持，客户须按照本文中的规范和参数进行产品设计和调试。如因客户操作不当造成的人身伤害和财产损失，有方概不承担责任。

由于产品版本升级或其它原因，本文档内容会在不预先通知的情况下进行必要的更新。

除非另有约定，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市有方科技股份有限公司为客户提供全方位的技术支持，任何垂询请直接联系您的客户经理或发送邮件至以下邮箱：

Sales@neoway.com

Support@neoway.com

公司网址：http://www.neoway.com

目录

[关于本文档 iv](#_Toc191393386)

[范围 iv](#_Toc191393387)

[读者对象 iv](#_Toc191393388)

[修订记录 iv](#_Toc191393389)

[符号约定 iv](#_Toc191393390)

[1 概述 5](#_Toc191393391)

[2 数据结构 6](#_Toc191393392)

[2.1 UART 6](#_Toc191393393)

[2.1.1 typedef enum nwy\_flowctrl\_e 6](#_Toc191393394)

[2.1.2 typedef enum nwy\_paritybits\_e 6](#_Toc191393395)

[2.1.3 typedef struct nwy\_uartdcb\_t 6](#_Toc191393396)

[2.1.4 typedef enum nwy\_baudrate\_e 6](#_Toc191393397)

[2.2 I2C 7](#_Toc191393398)

[2.2.1 typedef enum nwy\_i2c\_mode\_e 7](#_Toc191393399)

[2.3 SPI 7](#_Toc191393400)

[2.4 GPIO 7](#_Toc191393401)

[2.4.1 typedef enum nwy\_pin\_edge\_e 7](#_Toc191393402)

[2.4.2 typedef enum nwy\_dir\_mode\_e 7](#_Toc191393403)

[2.4.3 typedef enum nwy\_value\_e 7](#_Toc191393404)

[2.4.4 typedef enum nwy\_pin\_pull\_e 7](#_Toc191393405)

[2.5 ADC 8](#_Toc191393406)

[2.5.1 typedef enum nwy\_adc\_e 8](#_Toc191393407)

[2.5.2 typedef enum nwy\_adc\_aux\_scale\_e 8](#_Toc191393408)

[3 函数 9](#_Toc191393409)

[3.1 UART 9](#_Toc191393410)

[3.2 I2C 11](#_Toc191393411)

[3.3 SPI 12](#_Toc191393412)

[3.4 GPIO 14](#_Toc191393413)

[3.5 ADC 16](#_Toc191393414)

关于本文档

范围

本文档对应产品为N706B系列（RTOS系统）。

读者对象

本文档的使用对象为系统工程师，开发工程师及测试工程师。

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 变更 | 作者 |
| 1.0 | 2023-05 | 初始版本 | ZhangYanhong |
| 1.1 | 2025-02 | 2.4.1、2.4.4增加注释 | Lai Hongxiao |

符号约定

|  |  |
| --- | --- |
| 符号 | 含义 |
|  | 危险或警告，用户必须遵从的规则，否则会造成模块或客户设备不可逆的故障损坏，甚至可能造成人员身体伤害。 |
|  | 注意，警示用户使用模块时应该特别注意的地方，如不遵从，模块或客户设备可能出现故障。 |
|  | 说明或提示，提供模块使用的意见或建议。 |

# 概述

本文档介绍了OpenCSDK中IO驱动相关的API, 主要包括了以下几个方面.

* UART

主要用于UART初始化配置、数据传输以及回调函数注册；

* I2C

主要用于I2C初始化配置以及数据传输；

* SPI

主要用于SPI初始化配置以及数据传输；

* GPIO

主要用于GPIO参数配置以及中断函数注册；

* ADC

主要用于ADC进行电压值采样；

# 数据结构

## UART

### typedef enum nwy\_flowctrl\_e

typedef enum {

FC\_NONE = 0, // None Flow Control

FC\_RTSCTS, // Hardware Flow Control (rtscts)

FC\_XONXOFF // Software Flow Control (xon/xoff)

}nwy\_flowctrl\_e;

### typedef enum nwy\_paritybits\_e

typedef enum {

PB\_NONE = 0,

PB\_ODD,

PB\_EVEN,

PB\_SPACE,

PB\_MARK

}nwy\_paritybits\_e;

### typedef struct nwy\_uartdcb\_t

typedef struct {

unsigned int baudrate;

unsigned int databits;

unsigned int stopbits;

nwy\_paritybits\_e parity;

nwy\_flowctrl\_e flowctrl;

}nwy\_uartdcb\_t;

### typedef enum nwy\_baudrate\_e

typedef enum {

BR\_300 = 300,

BR\_600 = 600,

BR\_1200 = 1200,

BR\_2400 = 2400,

BR\_4800 = 4800,

BR\_9600 = 9600,

BR\_19200 = 19200,

BR\_38400 = 38400,

BR\_57600 = 57600,

BR\_115200 = 115200,

BR\_230400 = 230400,

BR\_460800 = 460800,

BR\_921600 = 921600

}nwy\_baudrate\_e;

## I2C

### typedef enum nwy\_i2c\_mode\_e

typedef enum

{

NWY\_I2C\_BPS\_100K, ///< normal 100Kbps

NWY\_I2C\_BPS\_400K, ///< fast 400Kbps

NWY\_I2C\_BPS\_3P5M, ///< high speed 3.5Mbps

} nwy\_i2c\_mode\_e;

## SPI

## GPIO

### typedef enum nwy\_pin\_edge\_e

typedef enum {

PIN\_NO\_EDGE = 0,

PIN\_RISING\_EDGE,

PIN\_FALLING\_EDGE,

PIN\_BOTH\_EDGE,

PIN\_HIGH\_LEVEL,

PIN\_LOW\_LEVEL,

} nwy\_pin\_edge\_e;

定义名称为nwy\_pin\_edge\_e 的枚举类型，用于表示引脚的中断触发方式。

### typedef enum nwy\_dir\_mode\_e

typedef enum {

PIN\_DIRECTION\_IN = 0,

PIN\_DIRECTION\_OUT,

} nwy\_dir\_mode\_e;

### typedef enum nwy\_value\_e

typedef enum {

PIN\_LEVEL\_LOW = 0,

PIN\_LEVEL\_HIGH,

} nwy\_value\_e;

### typedef enum nwy\_pin\_pull\_e

typedef enum {

PIN\_PULL\_DISABLE,

PIN\_PULL\_PU,

PIN\_PULL\_PD,

}nwy\_pin\_pull\_e;

定义名称为nwy\_pin\_pull\_e 的枚举类型，用于表示引脚的上下拉状态。

## ADC

### typedef enum nwy\_adc\_e

typedef enum

{

NWY\_ADC\_CHANNEL1 = 1,

NWY\_ADC\_CHANNEL2 = 2,

NWY\_ADC\_CHANNEL3 = 3,

NWY\_ADC\_CHANNEL\_VBAT = 4 // recommended to use this to obtain battery voltage

} nwy\_adc\_e;

### typedef enum nwy\_adc\_aux\_scale\_e

typedef enum

{

NWY\_ADC\_SCALE\_1V250 = 0,

NWY\_ADC\_SCALE\_2V444 = 1,

NWY\_ADC\_SCALE\_3V233 = 2,

NWY\_ADC\_SCALE\_5V000 = 3 //MAX

} nwy\_adc\_aux\_scale\_e;

# 函数

## UART

本章节提供UART相关的操作接口。

此接口函数定义位于nwy\_uart\_api.h，用于进行UART相关操作。

* + - 1. nwy\_uart\_open

|  |  |
| --- | --- |
| 格式 | int nwy\_uart\_open (const char\* name, unsigned int baudrate, nwy\_flowctrl\_e flowctrl); |
| 描述 | 打开UART设备并进行初始化配置 |
| 参数说明 | name:UART设备名  baudrate：UART波特率  flowctrl：流控 |
| 返回值 | 成功：UART设备fd  失败：<0 |

* + - 1. nwy\_uart\_read

|  |  |
| --- | --- |
| 格式 | int nwy\_uart\_read (int fd,unsigned char\* buf, unsigned int buf\_len) |
| 描述 | UART数据读取 |
| 参数说明 | fd: UART设备fd  buf: 指向存储读取数据的指针  buf\_len: buf的大小 |
| 返回值 | >=0: 实际读取数据的字节数  <0: 读取失败 |
| 说明 | 当前不支持！ |

* + - 1. nwy\_uart\_write

|  |  |
| --- | --- |
| 格式 | int nwy\_uart\_write (int fd, const unsigned char\* buf, unsigned int buf\_len) |
| 描述 | UART数据写入 |
| 参数说明 | fd: UART设备fd  buf: 指向写入数据的指针  buf\_len: 写入数据的大小 |
| 返回值 | >=0: 实际写入数据的字节数  <0: 写入失败 |

* + - 1. nwy\_uart\_dcb\_set

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_uart\_dcb\_set(int fd, nwy\_uartdcb\_t \*dcb) |
| 描述 | 设置UART dcb参数 |
| 参数说明 | fd: UART设备fd  dcb: 指向dcb结构体的指针 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |

* + - 1. nwy\_uart\_dcb\_get

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_uart\_dcb\_get(int fd, nwy\_uartdcb\_t \*dcb) |
| 描述 | 获取UART dcb参数 |
| 参数说明 | fd: UART设备fd  dcb: 指向dcb结构体的指针 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |

* + - 1. nwy\_uart\_close

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_uart\_close(int fd) |
| 描述 | 关闭UART设备 |
| 参数说明 | fd: UART设备fd |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |

* + - 1. nwy\_uart\_rx\_cb\_register

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_uart\_rx\_cb\_register(int fd, nwy\_uart\_rx\_callback cb) |
| 描述 | 该函数用于配置轮询模式或者中断模式，为UART 读取数据的回调函数 |
| 参数说明 | fd: UART设备fd  cb: 需要注册的回调函数，若为NULL，则为轮询模式。 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_uart\_tx\_cb\_register

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_uart\_tx\_cb\_register(int fd, nwy\_uart\_tx\_callback cb) |
| 描述 | 该函数用于配置轮询模式或者中断模式，为UART 写入数据的回调函数 |
| 参数说明 | fd: UART设备fd  cb: 需要注册的回调函数，若为NULL，则为轮询模式。 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_uart\_rx\_frame\_timeout\_set

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_uart\_rx\_frame\_timeout\_set(int fd, unsigned int time) |
| 描述 | 设置接收数据回调函数的超时时间 |
| 参数说明 | fd: UART设备fd  time: 超时时间 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |

## I2C

此接口函数定义位于nwy\_i2c\_api.h，用于进行i2c相关操作。

* + - 1. nwy\_i2c\_init

|  |  |
| --- | --- |
| 格式 | int nwy\_i2c\_init(const char \* i2cDev, nwy\_i2c\_mode\_e mode) |
| 描述 | 初始化i2c |
| 参数说明 | i2cDev：i2c设备名  mode：I2C 的工作模式 |
| 返回值 | 成功：i2c设备句柄  失败：<0 |

* + - 1. nwy\_i2c\_read

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_i2c\_read(int fd, unsigned char slaveAddr, unsigned short ofstAddr,  unsigned char\* ptrBuff, unsigned short length) |
| 描述 | 从指定I2C slave设备的目标地址读取指定长度数据 |
| 参数说明 | fd：i2c 设备句柄  slaveAddr：slave 设备地址  ofstAddr ：寄存器地址  ptrBuff： 数据存储地址  length：数据字节数 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他枚举值 |

* + - 1. nwy\_i2c\_write

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_i2c\_write(int fd, unsigned char slaveAddr, unsigned short ofstAddr,  unsigned char\* ptrData, unsigned short length) |
| 描述 | 往指定I2C slave设备的目标地址写入指定长度数据。 |
| 参数说明 | fd：i2c 设备句柄  slaveAddr：slave 设备地址  ofstAddr ：寄存器地址  ptrData： 数据存储地址  length：数据字节数 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他枚举值 |

* + - 1. nwy\_i2c\_deinit

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_i2c\_deinit(int fd) |
| 描述 | 释放掉i2c设备 |
| 参数说明 | fd：i2c 设备句柄 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他枚举值 |

## SPI

此接口函数定义位于nwy\_spi\_api.h，用于SPI相关操作。

* + - 1. nwy\_spi\_init

|  |  |
| --- | --- |
| 格式 | int nwy\_spi\_init(char \*spibus, uint8\_t mode, uint32\_t speed, uint8\_t bits) |
| 描述 | 打开并配置SPI总线 |
| 参数说明 | spibus：SPI总线名，支持："SPI1"、"SPI2"  mode：总线模式，mode0~mode3  speed：总线速率  bits：字长 |
| 返回值 | 成功：spi设备句柄  失败：<0 |

* + - 1. nwy\_spi\_read

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_spi\_read(int hd, void \*sendaddr, void \*readaddr, uint32\_t len) |
| 描述 | SPI读数据 |
| 参数说明 | hd：spi总线句柄  sendaddr：数据地址  readaddr：数据地址  len：传输大小 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_spi\_write

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_spi\_write(int hd, void \*sendaddr, uint32\_t len) |
| 描述 | SPI写数据 |
| 参数说明 | hd：spi总线句柄  sendaddr：数据地址  len：传输大小 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_spi\_transfer

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_spi\_transfer(int hd, uint8\_t cs, uint8\_t \*tx, uint8\_t \*rx, uint32\_t size) |
| 描述 | SPI数据传输 |
| 参数说明 | hd：spi总线句柄  cs：cs参数选择  tx：指向发送数据的指针  rx：指向接收数据的指针  size：字节数 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_spi\_deinit

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_spi\_deinit(int hd) |
| 描述 | 关闭spi设备 |
| 参数说明 | hd：spi总线句柄 |
| 返回值 | 成功：偏移量  失败：其他 |

## GPIO

此接口函数定义位于nwy\_gpio\_api.h，用于进行GPIO相关操作。

* + - 1. nwy\_gpio\_direction\_get

|  |  |
| --- | --- |
| 格式 | int nwy\_gpio\_direction\_get(uint32 gpio\_id) |
| 描述 | 获取GPIO的引脚方向。 |
| 参数说明 | gpio\_id：gpio编号 |
| 返回值 | 0 代表输入；  1 代表输出。  -1 获取失败 |

* + - 1. nwy\_gpio\_direction\_set

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_direction\_set(uint32 gpio\_id, nwy\_dir\_mode\_e dirct) |
| 描述 | 设置GPIO的引脚方向 |
| 参数说明 | gpio\_id：GPIO编号  dirct：引脚方向设置 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_gpio\_value\_get

|  |  |
| --- | --- |
| 格式 | int nwy\_gpio\_value\_get(uint32 gpio\_id) |
| 描述 | 获取GPIO的引脚电平状态 |
| 参数说明 | gpio\_id：GPIO编号 |
| 返回值 | 0 代表低电平；  1 代表高电平；  失败: NWY\_ERROR |

* + - 1. nwy\_gpio\_value\_set

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_value\_set(uint32 gpio\_id, nwy\_value\_e value) |
| 描述 | 设置GPIO的引脚电平状态。 |
| 参数说明 | gpio\_id：GPIO编号  value：引脚电平设置 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_gpio\_pull\_get

|  |  |
| --- | --- |
| 格式 | int nwy\_gpio\_pull\_get (uint32\_t gpio\_id) |
| 描述 | 获取GPIO的上拉/下拉属性 |
| 参数说明 | gpio\_id：GPIO编号 |
| 返回值 | 0 代表无上下拉；  1 代表上拉；  2 代表下拉  失败: NWY\_ERROR |
| 说明 | 当前不支持！ |

* + - 1. nwy\_gpio\_pull\_set

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_pull\_set (uint32\_t gpio\_id, nwy\_pin\_pull\_e pull) |
| 描述 | 设置GPIO 上拉/下拉属性 |
| 参数说明 | gpio\_id：GPIO编号  pul：引脚上下拉设置 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |

* + - 1. nwy\_gpio\_irq\_register

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_irq\_register (uint32\_t gpio\_id, nwy\_pin\_edge\_e pin\_edge, nwy\_pin\_pull\_e pin\_pull, void \*eint\_cb, void \*wakeup\_eint\_cb) |
| 描述 | 设置GPIO 中断配置 |
| 参数说明 | gpio\_id：GPIO编号  pin\_edge：中断的触发方式选择  pin\_pull：引脚上下拉设置  eint\_cb：中断回调函数，正常触发中断会调用此函数。  wakeup\_eint\_cb：唤醒中断回调函数，休眠状态下，外部中断唤醒系统会调用此函数。 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |
| 说明 | wakeup\_eint\_cb当前不支持，设为NULL! |

* + - 1. nwy\_gpio\_irq\_enable

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_irq\_enable(uint32\_t gpio\_id) |
| 描述 | 使能GPIO 中断，该中断对应的回调函数为*eint\_cb* |
| 参数说明 | gpio\_n: GPIO编号 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |

* + - 1. nwy\_gpio\_irq\_disable

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_irq\_disable(uint32\_t gpio\_id) |
| 描述 | 禁用GPIO 中断，该中断对应的回调函数为*eint\_cb* |
| 参数说明 | gpio\_n: GPIO编号 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |

* + - 1. nwy\_gpio\_irq\_wakeup\_enable

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_irq\_wakeup\_enable (uint32\_t gpio\_id) |
| 描述 | 该函数用于启用所配置的GPIO 唤醒中断，该中断对应的回调函数为wakeup\_eint\_cb |
| 参数说明 | gpio\_n: GPIO编号 |
| 返回值 | 成功：NWY\_SUCCESS  失败：其他 |
| 说明 | 当前不支持！ |

* + - 1. nwy\_gpio\_irq\_wakeup\_disable

|  |  |
| --- | --- |
| 格式 | nwy\_error\_e nwy\_gpio\_irq\_wakeup\_disable (uint32\_t gpio\_id) |
| 描述 | 该函数用于禁用所配置的GPIO 唤醒中断，该中断对应的回调函数为*wakeup\_eint\_cb* |
| 参数说明 | gpio\_n: GPIO编号 |
| 返回值 | 成功：NWY\_SUCESS  失败：其他 |
| 说明 | 当前不支持！ |

## ADC

此接口函数定义位于nwy\_adc\_api.h，用于进行ADC相关操作。

* + - 1. nwy\_adc\_get

|  |  |
| --- | --- |
| 格式 | int nwy\_adc\_get(nwy\_adc\_e channel,nwy\_adc\_aux\_scale\_e scale) |
| 描述 | 获取ADC value |
| 参数说明 | Channel：ADC通道  Scale: ADC量程 |
| 返回值 | >=0: 采集到的电压值。（单位mV，误差约± 2%）  <0：其他 |