



HiSpark-智能小车套件 电机编程指南

文档版本 00B01

发布日期 2020/8/3

版权所有 © 上海海思技术有限公司。保留一切权利。

非经本公司书面许可，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。

商标声明



HISILICON、海思和其他海思商标均为上海海思技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司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的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，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。

上海海思技术有限公司

地址：上海市青浦区金泽镇（西岑）水秀路 318 号 101 室 邮编：201718

网址：<http://www.hisilicon.com>



前言

概述

本文档主要介绍基于海思 WiFi 芯片 Hi3861 开发的 HiSpark-WiFi-IoT 套件演示指导书。

产品版本

与本文档相对应的主芯片版本如下。

产品名称	产品版本
Hi3861	V100R001C00SPC021

读者对象

本文档（本指南）主要适用于以下工程师：

- 软件开发工程师
- 硬件开发工程师

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修订日期	版本	修订说明
2020-08-3	00B01	第一次临时版本发布。



目 录

前 言.....	i
1 智能小车套件电机功能实现	4
1.1 电机硬件准备	4
1.2 电机硬件介绍	5
1.3 电机功能软件实现	6



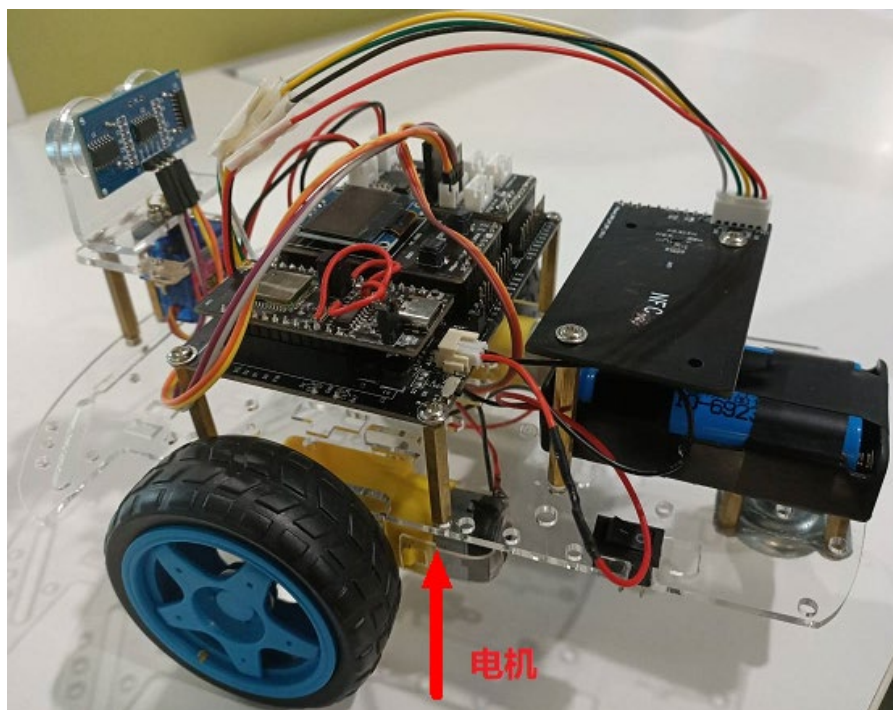
1 智能小车套件电机功能实现

1.1 电机硬件准备

图 1.1-1 电机



图 1.1-2 电机在智能小车上的安装





1.2 电机硬件介绍

所用驱动电机为 tt 直流减速马达，IO 引脚的连接如下：

电机 1：

IA ----- GPIO_0

IB ----- GPIO_1

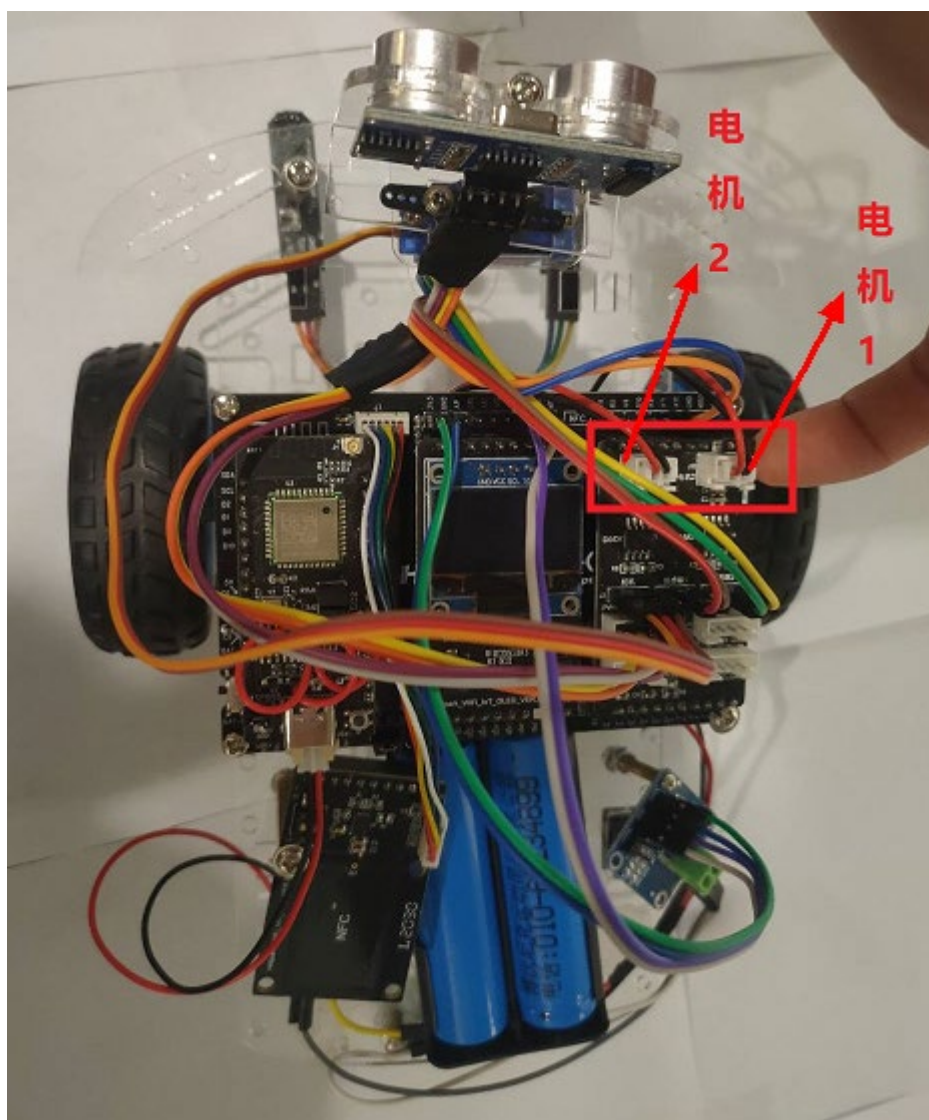
电机 2：

IA ----- GPIO_9

IB ----- GPIO_10

电机模块在小车上的连线如下所示：

图 1.2-1 电机模块接线图





1.3 电机功能软件实现

电机驱动实现为对 IB/IA 引脚拉高，其中一根引脚进行 PWM 输出，以此来控制电机转动。转动方向与电机所在位置有关。Demo 示例中，拉高 IB，IA 输出 PWM，小车后退；拉高 IA，IB 输出 PWM，小车前进。

Demo 中主要通过以下两个封装好的函数进行 IO 控制。

(1) PWM 控制：

hi_void pwm_control(hi_io_name id, hi_u8 val, hi_pwm_port port, hi_u16 duty);

函数实现如下：

```
hi_void pwm_control(hi_io_name id, hi_u8 val, hi_pwm_port port, hi_u16 duty)
{
    hi_io_set_func(id, val);
    hi_pwm_init(port);
    hi_pwm_set_clock(PWM_CLK_160M);
    hi_pwm_start(port, duty, PWM_FREQ_FREQUENCY);
}
```

hi_io_name id: gpio 硬件管脚编号，对应的是硬件管脚上的 GPIO pin 脚，枚举类型如下：

```
typedef enum {
    HI_IO_NAME_GPIO_0,    /**< GPIO0 */
    HI_IO_NAME_GPIO_1,    /**< GPIO1 */
    HI_IO_NAME_GPIO_2,    /**< GPIO2 */
    HI_IO_NAME_GPIO_3,    /**< GPIO3 */
    HI_IO_NAME_GPIO_4,    /**< GPIO4 */
    HI_IO_NAME_GPIO_5,    /**< GPIO5 */
    HI_IO_NAME_GPIO_6,    /**< GPIO6 */
    HI_IO_NAME_GPIO_7,    /**< GPIO7 */
    HI_IO_NAME_GPIO_8,    /**< GPIO8 */
    HI_IO_NAME_GPIO_9,    /**< GPIO9 */
    HI_IO_NAME_GPIO_10,   /**< GPIO10 */
    HI_IO_NAME_GPIO_11,   /**< GPIO11 */
    HI_IO_NAME_GPIO_12,   /**< GPIO12 */
    HI_IO_NAME_GPIO_13,   /**< GPIO13 */
    HI_IO_NAME_GPIO_14,   /**< GPIO14 */
    HI_IO_NAME_SFC_CSN,   /**< SFC_CSN */
}
```




```
HI_IO_NAME_SFC_IO1,    /**< SFC_IO1 */
HI_IO_NAME_SFC_IO2,    /**< SFC_IO2 */
HI_IO_NAME_SFC_IO0,    /**< SFC_IO0 */
HI_IO_NAME_SFC_CLK,    /**< SFC_CLK */
HI_IO_NAME_SFC_IO3,    /**< SFC_IO3 */
HI_IO_NAME_MAX,
} hi_io_name;
```

hi_u8 val: 对应 GPIO 引脚功能，如果该引脚复用，可以配置为其他的复用功能，如：UART/PWM/I2C/SPI/SDIO 等功能；

hi_pwm_port port: 对应 PWM 端口，取值范围为 hi_pwm_port 枚举值，枚举类型如下：

```
typedef enum {
    HI_PWM_PORT_PWM0 = 0,
    HI_PWM_PORT_PWM1 = 1,
    HI_PWM_PORT_PWM2 = 2,
    HI_PWM_PORT_PWM3 = 3,
    HI_PWM_PORT_PWM4 = 4,
    HI_PWM_PORT_PWM5 = 5,
    HI_PWM_PORT_MAX
} hi_pwm_port;
```

hi_u16 duty: 占空比计数值（此值可自己设置）

信号占空比为：duty/freq，占空比越小，转速越快，占空比越大，转速越慢。

(2) IO 引脚拉高/拉低控制：

hi_void gpio_control(hi_io_name gpio, hi_gpio_idx id, hi_gpio_dir dir, hi_gpio_value gpio_val, hi_u8 val);

函数实现如下：

```
hi_void gpio_control(hi_io_name gpio, hi_gpio_idx id, hi_gpio_dir dir, hi_gpio_value gpio_val, hi_u8 val)
{
    hi_io_set_func(gpio, val);
    hi_gpio_set_dir(id, dir);
    hi_gpio_set_output_val(id, gpio_val);
}
```

hi_io_name gpio: gpio 硬件管脚编号，对应的是硬件管脚上的 GPIO pin 脚，枚举类型如下：

```
typedef enum {
    HI_IO_NAME_GPIO_0,    /**< GPIO0 */
    HI_IO_NAME_GPIO_1,    /**< GPIO1 */
    HI_IO_NAME_GPIO_2,    /**< GPIO2 */
}
```



```

HI_IO_NAME_GPIO_3,    /**< GPIO3 */
HI_IO_NAME_GPIO_4,    /**< GPIO4 */
HI_IO_NAME_GPIO_5,    /**< GPIO5 */
HI_IO_NAME_GPIO_6,    /**< GPIO6 */
HI_IO_NAME_GPIO_7,    /**< GPIO7 */
HI_IO_NAME_GPIO_8,    /**< GPIO8 */
HI_IO_NAME_GPIO_9,    /**< GPIO9 */
HI_IO_NAME_GPIO_10,   /**< GPIO10 */
HI_IO_NAME_GPIO_11,   /**< GPIO11 */
HI_IO_NAME_GPIO_12,   /**< GPIO12 */
HI_IO_NAME_GPIO_13,   /**< GPIO13 */
HI_IO_NAME_GPIO_14,   /**< GPIO14 */
HI_IO_NAME_SFC_CSN,   /**< SFC_CSN */
HI_IO_NAME_SFC_IO1,   /**< SFC_IO1 */
HI_IO_NAME_SFC_IO2,   /**< SFC_IO2 */
HI_IO_NAME_SFC_IO0,   /**< SFC_IO0 */
HI_IO_NAME_SFC_CLK,   /**< SFC_CLK */
HI_IO_NAME_SFC_IO3,   /**< SFC_IO3 */
HI_IO_NAME_MAX,
} hi_io_name;

```

hi_gpio_idx id：硬件管脚编号，对应的是硬件管脚上的 GPIO pin 脚，枚举类型如下：

```

typedef enum {
    HI_GPIO_IDX_0, /**< GPIO0*/
    HI_GPIO_IDX_1, /**< GPIO1*/
    HI_GPIO_IDX_2, /**< GPIO2*/
    HI_GPIO_IDX_3, /**< GPIO3*/
    HI_GPIO_IDX_4, /**< GPIO4*/
    HI_GPIO_IDX_5, /**< GPIO5*/
    HI_GPIO_IDX_6, /**< GPIO6*/
    HI_GPIO_IDX_7, /**< GPIO7*/
    HI_GPIO_IDX_8, /**< GPIO8*/
    HI_GPIO_IDX_9, /**< GPIO9*/
    HI_GPIO_IDX_10, /**< GPIO10*/
}

```



```
HI_GPIO_IDX_11, /**< GPIO11*/
```

```
HI_GPIO_IDX_12, /**< GPIO12*/
```

```
HI_GPIO_IDX_13, /**< GPIO13*/
```

```
HI_GPIO_IDX_14, /**< GPIO14*/
```

```
HI_GPIO_IDX_MAX, /**< Maximum value, which cannot be used.CNcomment:最大值, 不可输入使用 CNend*/
```

```
} hi_gpio_idx;
```

hi_gpio_dir dir: dir, GPIO 输出方向

```
typedef enum {
```

```
    HI_GPIO_DIR_IN = 0, /**< Input.CNcomment:输入方向 CNend*/
```

```
    HI_GPIO_DIR_OUT /**< Output.CNcomment:输出方向 CNend*/
```

```
} hi_gpio_dir;
```

hi_gpio_value gpio_val : gpio 输出状态。

```
typedef enum {
```

```
    HI_GPIO_VALUE0 = 0, /**< Low level.CNcomment:低电平 CNend*/
```

```
    HI_GPIO_VALUE1 /**< High level.CNcomment:高电平 CNend*/
```

```
} hi_gpio_value;
```

hi_u8 val :对应 GPIO 引脚功能, 如果该引脚复用, 可以配置为其他的复用功能, 如: UART/PWM/I2C/SPI/SDIO 等功能;

在 demo 中示例:

前进: GPIO0/GPIO9 拉高, GPIO1/GPIO10 输出 PWM

```
hi_void car_go_forward(hi_void)
{
    correct_car_speed();
    gpio_control(HI_IO_NAME_GPIO_0, HI_GPIO_IDX_0, HI_GPIO_DIR_OUT, HI_GPIO_VALUE1, HI_IO_FUNC_GPIO_0_GPIO);
    pwm_control(HI_IO_NAME_GPIO_1, HI_IO_FUNC_GPIO_1_PWM4_OUT, HI_PWM_PORT_PWM4, g_car_speed);
    gpio_control(HI_IO_NAME_GPIO_9, HI_GPIO_IDX_9, HI_GPIO_DIR_OUT, HI_GPIO_VALUE1, HI_IO_FUNC_GPIO_9_GPIO);
    pwm_control(HI_IO_NAME_GPIO_10, HI_IO_FUNC_GPIO_10_PWM1_OUT, HI_PWM_PORT_PWM1, g_car_speed);
}
```

后退: GPIO1/GPIO10 拉高, GPIO0/GPIO9 输出 PWM

```
hi_void car_go_back(hi_void)
{
    correct_car_speed();
    pwm_control(HI_IO_NAME_GPIO_0, HI_IO_FUNC_GPIO_0_PWM3_OUT, HI_PWM_PORT_PWM3, g_car_speed);
    gpio_control(HI_IO_NAME_GPIO_1, HI_GPIO_IDX_1, HI_GPIO_DIR_OUT, HI_GPIO_VALUE1, HI_IO_FUNC_GPIO_1_GPIO);
    pwm_control(HI_IO_NAME_GPIO_9, HI_IO_FUNC_GPIO_9_PWM0_OUT, HI_PWM_PORT_PWM0, g_car_speed);
    gpio_control(HI_IO_NAME_GPIO_10, HI_GPIO_IDX_10, HI_GPIO_DIR_OUT, HI_GPIO_VALUE1, HI_IO_FUNC_GPIO_10_GPIO);
}
```



停止：GPIO0/GPIO9 拉低，GPIO1/GPIO10 输出 PWM （拉低 IB/IA 一根引脚即可）

```
hi_void car_stop(hi_void)
{
    correct_car_speed();
    pwm_control(HI_IO_NAME_GPIO_0, HI_IO_FUNC_GPIO_0_PWM3_OUT, HI_PWM_PORT_PWM3, PWM_DUTY_STOP);
    gpio_control(HI_IO_NAME_GPIO_1, HI_GPIO_IDX_1, HI_GPIO_DIR_OUT, HI_GPIO_VALUE0, HI_IO_FUNC_GPIO_1_GPIO);
    pwm_control(HI_IO_NAME_GPIO_9, HI_IO_FUNC_GPIO_9_PWM0_OUT, HI_PWM_PORT_PWM0, PWM_DUTY_STOP);
    gpio_control(HI_IO_NAME_GPIO_10, HI_GPIO_IDX_10, HI_GPIO_DIR_OUT, HI_GPIO_VALUE0, HI_IO_FUNC_GPIO_10_GPIO);
}
```

左转：GPIO1 拉低，GPIO9 拉高，GPIO0/GPIO10 输出 PWM

```
hi_void car_turn_left(hi_void)
{
    correct_car_speed();
    pwm_control(HI_IO_NAME_GPIO_0, HI_IO_FUNC_GPIO_0_PWM3_OUT, HI_PWM_PORT_PWM3, PWM_DUTY_LEFT_RIGHT);
    gpio_control(HI_IO_NAME_GPIO_1, HI_GPIO_IDX_1, HI_GPIO_DIR_OUT, HI_GPIO_VALUE0, HI_IO_FUNC_GPIO_1_GPIO);
    gpio_control(HI_IO_NAME_GPIO_9, HI_GPIO_IDX_9, HI_GPIO_DIR_OUT, HI_GPIO_VALUE1, HI_IO_FUNC_GPIO_9_GPIO);
    pwm_control(HI_IO_NAME_GPIO_10, HI_IO_FUNC_GPIO_10_PWM1_OUT, HI_PWM_PORT_PWM1, PWM_DUTY_LEFT_RIGHT);
}
```

右转：GPIO10 拉低，GPIO0 拉高，GPIO1/GPIO9 输出 PWM

```
hi_void car_turn_right(hi_void)
{
    correct_car_speed();
    gpio_control(HI_IO_NAME_GPIO_0, HI_GPIO_IDX_0, HI_GPIO_DIR_OUT, HI_GPIO_VALUE1, HI_IO_FUNC_GPIO_0_GPIO);
    pwm_control(HI_IO_NAME_GPIO_1, HI_IO_FUNC_GPIO_1_PWM4_OUT, HI_PWM_PORT_PWM4, PWM_DUTY_LEFT_RIGHT);
    pwm_control(HI_IO_NAME_GPIO_9, HI_IO_FUNC_GPIO_9_PWM0_OUT, HI_PWM_PORT_PWM0, PWM_DUTY_LEFT_RIGHT);
    gpio_control(HI_IO_NAME_GPIO_10, HI_GPIO_IDX_10, HI_GPIO_DIR_OUT, HI_GPIO_VALUE0, HI_IO_FUNC_GPIO_10_GPIO);
}
```