3.1 点亮 LED 灯

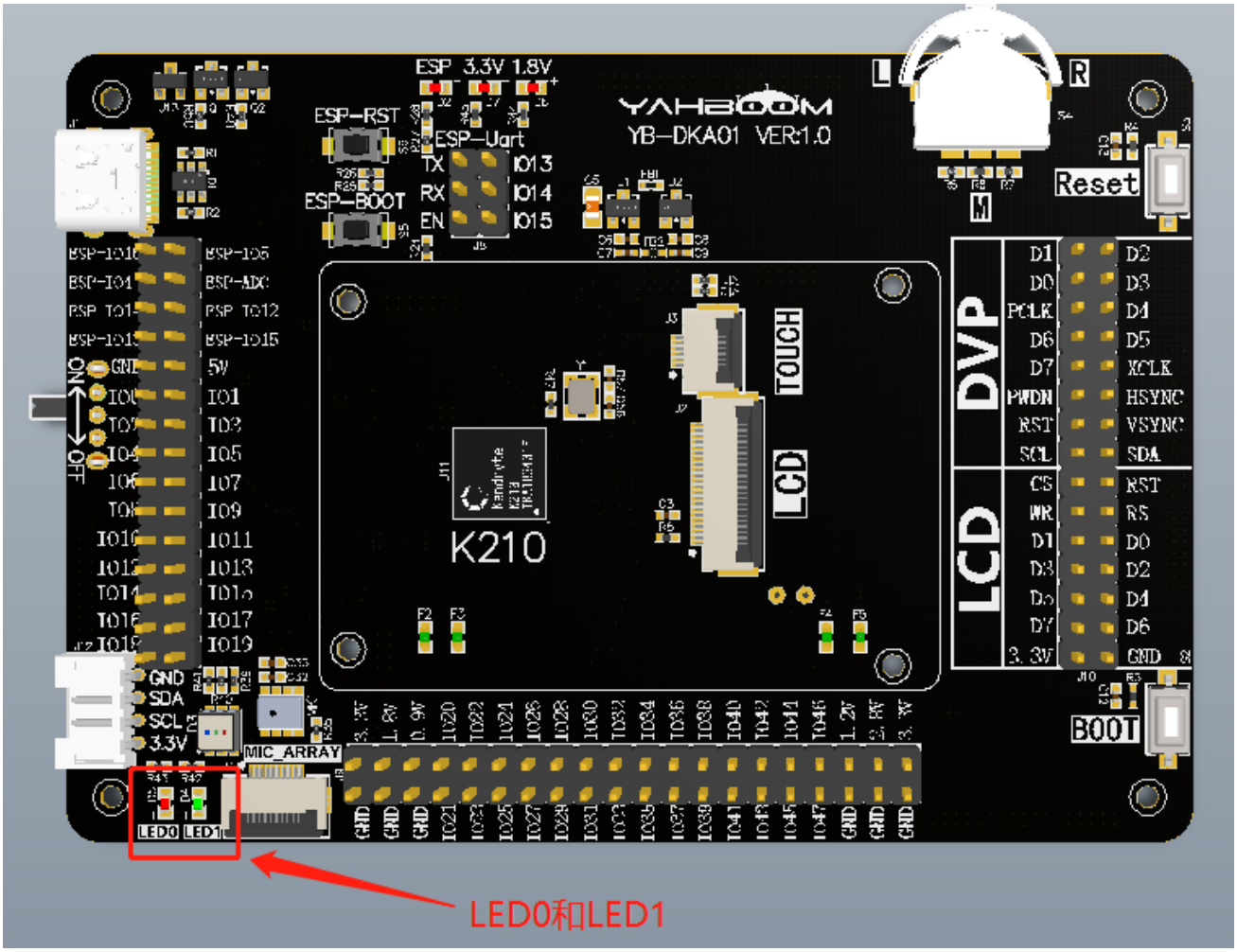
一、实验目的

本节课主要学习 K210 最基础的功能，FPIOA 引脚映射和点亮 LED 灯。

二、实验准备

1.实验元件

LED0 和 LED1

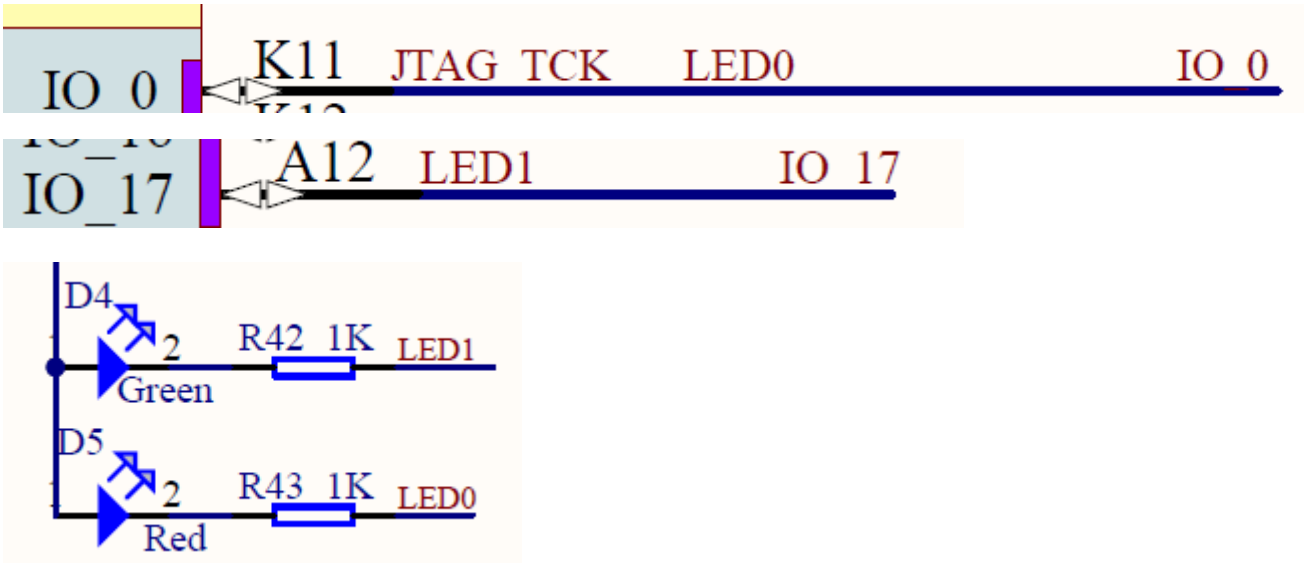


2.元件特性

LED0 为红灯，LED1 为绿灯。两颗 LED 灯都是低电平点亮，高电平熄灭。

3.硬件连接

K210 开发板出厂默认已经焊接好 LED0 和 LED1。LED0 连接的是 IO0，LED1 连接的是 IO17。



4. SDK 中对应 API 功能

对应的头文件 gpio.h

通用 gpio 共 8 个，使用同一个中断源，可配置输入输出信号，可配置触发 IO 口总中断，边沿触发和电平触发。每隔 IO 可以分配到 FPIOA 上 48 个管教之一。

为用户提供以下接口：

- gpio_init：GPIO 口初始化
- gpio_set_drive_mode：设置 GPIO 口输入或输出模式
- gpio_set_pin：设置 GPIO 引脚电平高/低
- gpio_get_pin：读取 GPIO 引脚电平

5. 什么是 FPIOA 呢？

FPIOA （现场可编程 IO 阵列）允许用户将 255 个内部功能映射到芯片外围的 48 个自由 IO 上：

- 支持IO 的可编程功能选择
- 支持IO 输出的8 种驱动能力选择
- 支持IO 的内部上拉电阻选择
- 支持IO 的内部下拉电阻选择
- 支持IO 输入的内部施密特触发器设置

- 支持IO 输出的斜率控制
- 支持内部输入逻辑的电平设置

三、实验原理

LED（Light Emitting Diode）也称为发光二极管，是一种能够将电能转化为可见光的固态的半导体器件，它可以直接把电转化为光。LED 的内部是一个半导体晶片，晶片的一端附在一个支架上，一端是负极，另一端连接电源的正极，使整个晶片被环氧树脂封装起来。只需要给正极输入正极电压，负极接地，形成回路就可以点亮 LED。

半导体晶片由两部分组成，一部分是 P 型半导体，在它里面空穴占主导地位，另一端是 N 型半导体，在这边主要是电子。但这两种半导体连接起来的时候，它们之间就形成一个 P-N 结。当电流通过导线作用于这个晶片的时候，电子就会被推向 P 区，在 P 区里电子跟空穴复合，然后就会以光子的形式发出能量，这就是 LED 灯发光的原理。而光的波长也就是光的颜色，是由形成 P-N 结的材料决定的。

四、实验过程

1. 首先根据上面的硬件连接引脚图，K210 的硬件引脚和软件功能使用的是 FPIOA 映射关系。

这里要注意的是程序里操作的都是软件引脚，所以需要先把硬件引脚映射成软件 GPIO 功能，操作的时候直接操作软件 GPIO 即可。

```
void hardware_init(void)
{
    fpioa_set_function(PIN_LED_0, FUNC_LED0);
    fpioa_set_function(PIN_LED_1, FUNC_LED1);
}
```

```
#ifndef _PIN_CONFIG_H_
#define _PIN_CONFIG_H_
/*****************************HEAR-FILE*********************
#include "fpioa.h"

/*****************************HARDWARE-PIN******************
// 硬件IO口，与原理图对应
#define PIN_LED_0               (0)
#define PIN_LED_1               (17)

/*****************************SOFTWARE-GPIO*****************
// 软件GPIO口，与程序对应
#define LED0_GPIONUM            (0)
#define LED1_GPIONUM            (1)

/*****************************FUNC-GPIO*********************
// GPIO口的功能，绑定到硬件IO口
#define FUNC_LED0               (FUNC_GPIO0 + LED0_GPIONUM)
#define FUNC_LED1               (FUNC_GPIO0 + LED1_GPIONUM)

#endif /* _PIN_CONFIG_H_ */
```

2.main 函数是 K210 芯片的入口函数，所有程序都从这里开始执行，首先初始化硬件引脚，然后使能 GPIO 时钟，再设置 LED0 和 LED1 为输出模式，接着设置 LED0 和 LED1 的电平为高电平，表示熄灭状态。

```
int main(void)
{
    hardware_init();// 硬件引脚初始化

    gpio_init();    // 使能GPIO的时钟

    // 设置LED0和LED1的GPIO模式为输出
    gpio_set_drive_mode(LED0_GPIONUM, GPIO_DM_OUTPUT);
    gpio_set_drive_mode(LED1_GPIONUM, GPIO_DM_OUTPUT);

    // 先关闭LED0和LED1
    gpio_pin_value_t value = GPIO_PV_HIGH;
    gpio_set_pin(LED0_GPIONUM, value);
    gpio_set_pin(LED1_GPIONUM, value);
```

最后在 while 循环中每隔一秒切换修改 value 的值，让 LED0 和 LED1 交替点亮。

```
    while (1)
    {
        sleep(1);
        gpio_set_pin(LED0_GPIONUM, value);
        gpio_set_pin(LED1_GPIONUM, value = !value);
    }
    return 0;
}
```

3.编译调试，烧录运行

把本课程资料中的 gpio_led 复制到 SDK 中的 src 目录下，然后进入 build 目录，运行以下命令编译。

cmake .. -DPROJ=gpio_led -G "MinGW Makefiles"

make

```
PS C:\K210\SDK\kendryte-standalone-sdk-develop\build> cmake .. -DPROJ=gpio_led -G "MinGW Makefiles"
PROJ = gpio_led
-- Check for RISCV toolchain ...
-- Using C:/K210/kendryte-toolchain/bin RISCV toolchain
SOURCE_FILES=C:/K210/SDK/kendryte-standalone-sdk-develop/src/gpio_led/main.c
```

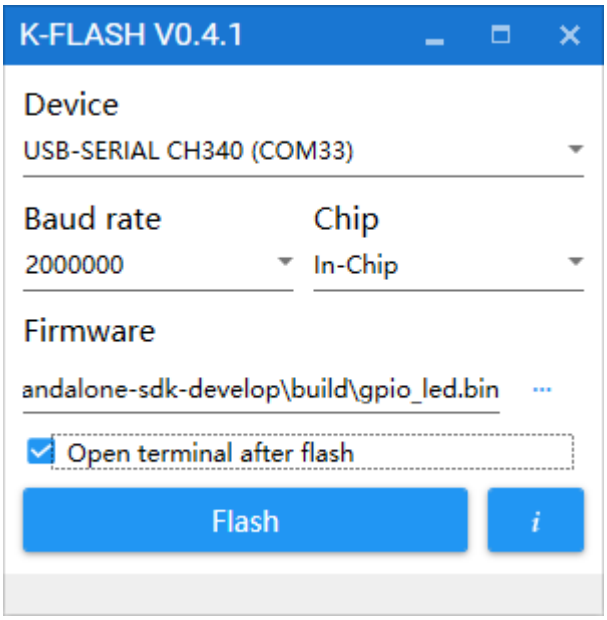
```
-- Configuring done
-- Generating done
-- Build files have been written to: C:/K210/SDK/kendryte-standalone-sdk-develop/build
PS C:\K210\SDK\kendryte-standalone-sdk-develop\build> make
```

```
[ 97%] Building C object CMakeFiles/gpio_led.dir/src/gpio_led/main.c.obj
[100%] Linking C executable gpio_led
Generating .bin file ...
[100%] Built target gpio_led
PS C:\K210\SDK\kendryte-standalone-sdk-develop\build>
```

编译完成后，在 build 文件夹下会生成 gpio_led.bin 文件。

使用 type-C 数据线连接电脑与 K210 开发板，打开 kflash，选择对应的设备，再将 gpio_led.bin 文件烧录到 K210 开发板上。

注意要勾选‘Open terminal after flash’，这样烧录完成固件后就会弹出一个终端可以查看调试的信息。



五、实验现象

LED0 和 LED1 两个灯交替点亮。先亮绿灯 1 秒，然后绿灯熄灭，红灯亮一秒后熄灭，绿灯再亮起，以此循环。

六、实验总结

1.K210 芯片使用 FPIOA 可编程阵列，所以每次使用硬件 IO 口前都需要对硬件 IO 口进行引脚映射。而且在软件中调用的也是软件映射后的软件 GPIO。

2.K210 芯片与其他单片机芯片同样是从 main 函数开始运行。

3.使用 GPIO 前需要设置 GPIO 的输入输出模式。

4.LED 灯是低电平点亮的，给 LED 引脚设置低电平时点亮，设置高电平时熄灭。

附：API

对应的头文件 gpio.h

gpio_init

描述

初始化 GPIO。

函数原型

```
int gpio_init(void)
```

返回值

返回值	描述
0	成功
非 0	失败

gpio_set_drive_mode

描述

设置 GPIO 驱动模式。

函数原型

```
void gpio_set_drive_mode(uint8_t pin, gpio_drive_mode_t mode)
```

参数

参数名称	描述	输入输出
pin	GPIO 管脚	输入
mode	GPIO 驱动模式	输入

返回值

无。

gpio_set_pin

描述

设置 GPIO 管脚值。

函数原型

```
void gpio_set_pin(uint8_t pin, gpio_pin_value_t value)
```

参数

参数名称	描述	输入输出
pin	GPIO 管脚	输入
value	GPIO 值	输入

返回值

无。

gpio_get_pin

描述

获取 GPIO 管脚值。

函数原型

```
gpio_pin_value_t gpio_get_pin(uint8_t pin)
```

参数

参数名称	描述	输入输出
pin	GPIO 管脚	输入

返回值

获取的 GPIO 管脚值。

数据类型

相关数据类型、数据结构定义如下：

- gpio_drive_mode_t：GPIO 驱动模式。
- gpio_pin_value_t：GPIO 值。

gpio_drive_mode_t

描述

GPIO 驱动模式。

定义

```
typedef enum _gpio_drive_mode
{
    GPIO_DM_INPUT,
    GPIO_DM_INPUT_PULL_DOWN,
    GPIO_DM_INPUT_PULL_UP,
    GPIO_DM_OUTPUT,
} gpio_drive_mode_t;
```

成员

成员名称	描述
GPIO_DM_INPUT	输入
GPIO_DM_INPUT_PULL_DOWN	输入下拉
GPIO_DM_INPUT_PULL_UP	输入上拉
GPIO_DM_OUTPUT	输出

gpio_pin_value_t

描述

GPIO 值。

定义

```
typedef enum _gpio_pin_value
{
    GPIO_PV_LOW,
    GPIO_PV_HIGH
} gpio_pin_value_t;
```

成员

成员名称	描述
GPIO_PV_LOW	低
GPIO_PV_HIGH	高