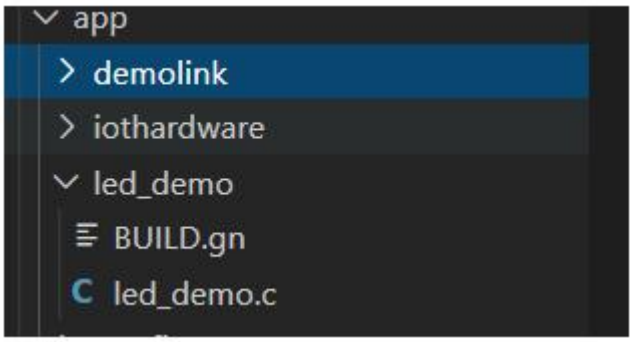
Hi3861 也提供了相关得 GPIO 口操作

先看最简单得 LED 灯闪烁操作

源码结构如下：



BUILD.gn 文件内容：

```
static_library("led_demo") {
    sources = [
        "led_demo.c"
    ]

    include_dirs = [
        "//utils/native/lite/include",
        "//kernel/liteos_m/components/cmsis/2.0",
        "//base/iot_hardware/interfaces/kits/wifiiot_lite"
    ]
}
```

led_demo.c 内容：

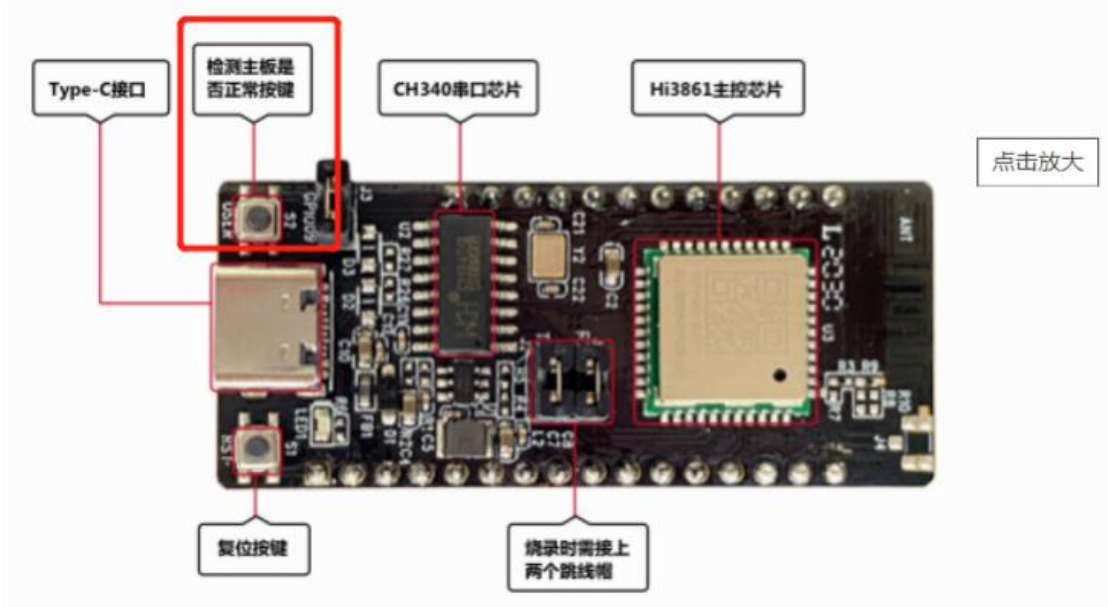
```
#include <unistd.h>
#include "stdio.h"
#include "ohos_init.h"
#include "cmsis_os2.h"
#include "wifiiot_gpio.h"
#include "wifiiot_gpio_ex.h"


#include <hi_types_base.h>
#include <hi_i2c.h>
#include <hi_early_debug.h>
#include <hi_stdlib.h>
```

```
#include "oled_demo.h"
#include "oledfont.h"

void *LedTask(const char *arg)
{

    (void)arg;
    while (1)
    {
        GpioSetOutputVal(WIFI_IOT_IO_NAME_GPIO_9, 0);
        usleep(300000);
        GpioSetOutputVal(WIFI_IOT_IO_NAME_GPIO_9, 1);
        usleep(300000);
    }

    return NULL;
}

void led_demo(void)
{
    //osThreadAttr_t attr;

    GpioInit();

    //复用引脚为 GPIO
    IoSetFunc(WIFI_IOT_IO_NAME_GPIO_9, WIFI_IOT_IO_FUNC_GPIO_9_GPIO);

    //设置为输出
    GpioSetDir(WIFI_IOT_IO_NAME_GPIO_9, WIFI_IOT_GPIO_DIR_OUT);



    attr.name = "LedTask";
    attr.attr_bits = 0U;
    attr.cb_mem = NULL;
    attr.cb_size = 0U;
    attr.stack_mem = NULL;
    attr.stack_size = 512;
    attr.priority = 26;

    if (osThreadNew((osThreadFunc_t)LedTask, NULL, &attr) == NULL) {
        printf("[LedExample] Falied to create LedTask!\n");
    }
```

```
}


//SYS_RUN(led_demo);

void oled_test(void)
{

}

SYS_RUN(oled_test);
```

另外 GPIO 口还可以作为输入，然后使用中断，示例代码如下：

这段示例代码用的开发板上面的 user 按键。



通过查阅原理图，我们可以看到 Hi3861 在 type-C 口附近有一个 user 按钮，如图，主要不要和复位按钮搞错了。user 按钮对应的是 GPIO5 引脚。

```
/* gpio callback func */
hi_void my_gpio_isr_func(hi_void *arg)
{
    hi_unref_param(arg);
    printf("----- gpio isr success -----\r\n");
```

```
}

/* 设置 按键中断响应 */
hi_void my_gpio_isr_demo(hi_void)
{
    hi_u32 ret;

    printf("----- gpio isr demo -----\r\n");

    (hi_void)hi_gpio_init();

    hi_io_set_func(HI_IO_NAME_GPIO_5, HI_IO_FUNC_GPIO_5_GPIO); /* uart1 rx */

    ret = hi_gpio_set_dir(HI_GPIO_IDX_5, HI_GPIO_DIR_IN);
    if (ret != HI_ERR_SUCCESS) {
        printf("===== ERROR ======gpio -> hi_gpio_set_dir1 ret:%d\r\n", ret);
        return;
    }

    ret = hi_gpio_register_isr_function(HI_GPIO_IDX_5, HI_INT_TYPE_EDGE,
                                        HI_GPIO_EDGE_RISE_LEVEL_HIGH,
my_gpio_isr_func, HI_NULL);
    if (ret != HI_ERR_SUCCESS) {
        printf("===== ERROR ======gpio -> hi_gpio_register_isr_function ret:%d\r\n", ret);
    }
}
```