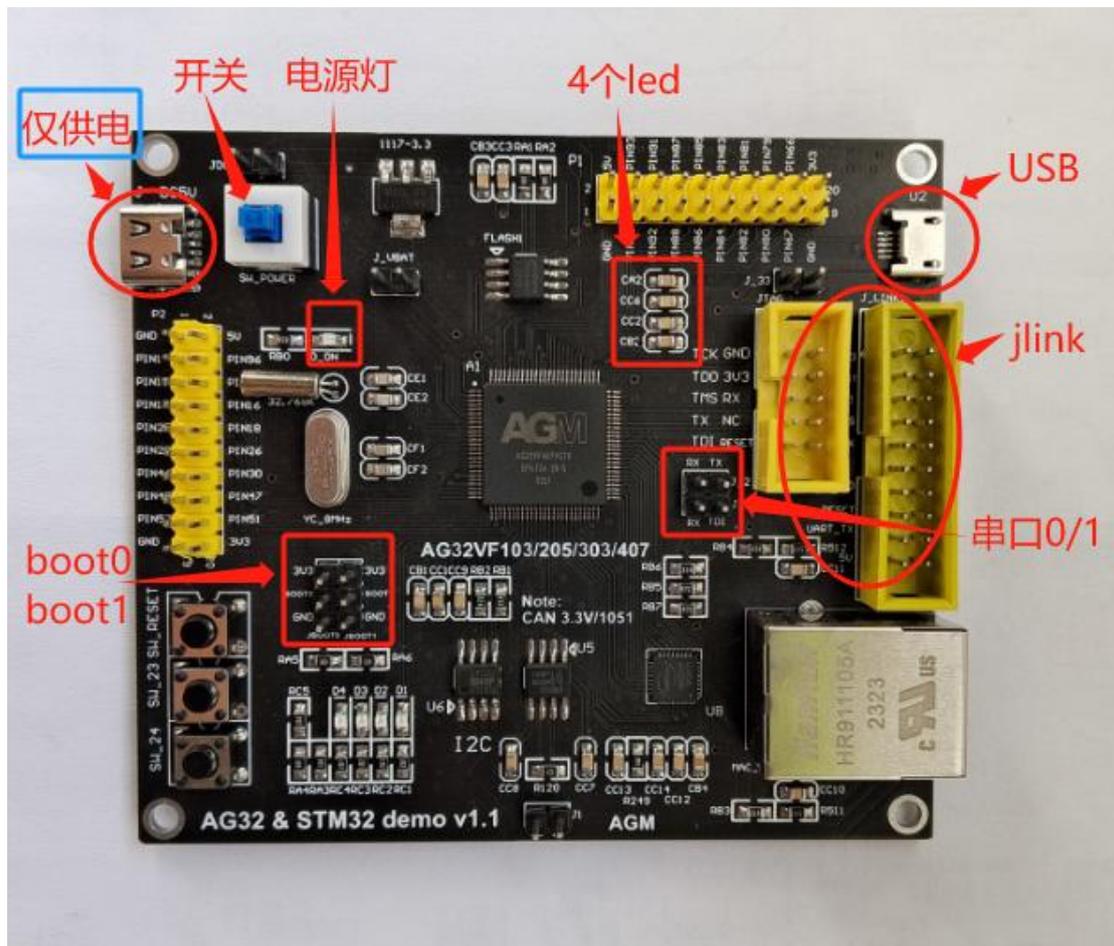
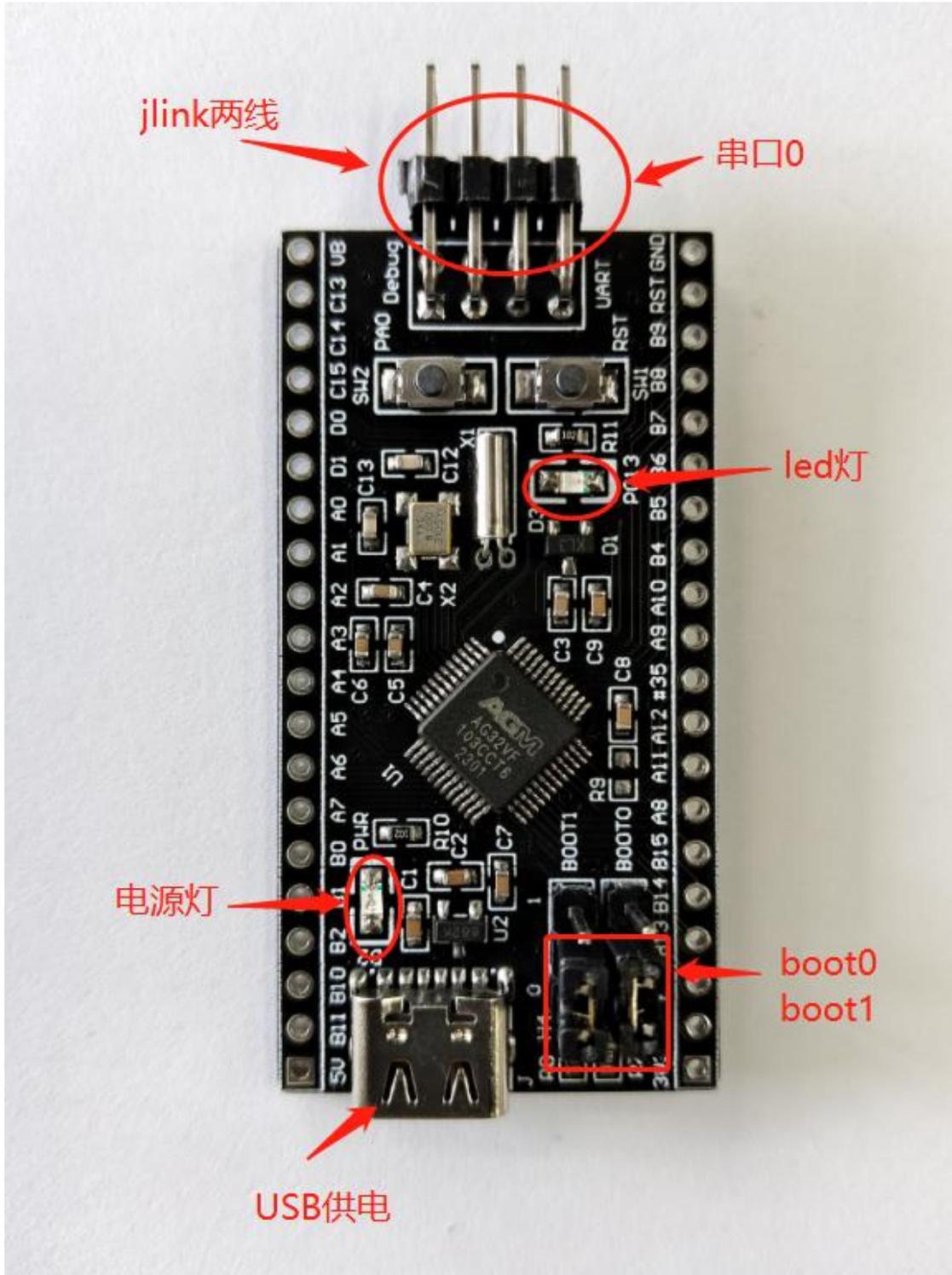


AG32 开发板使用入门

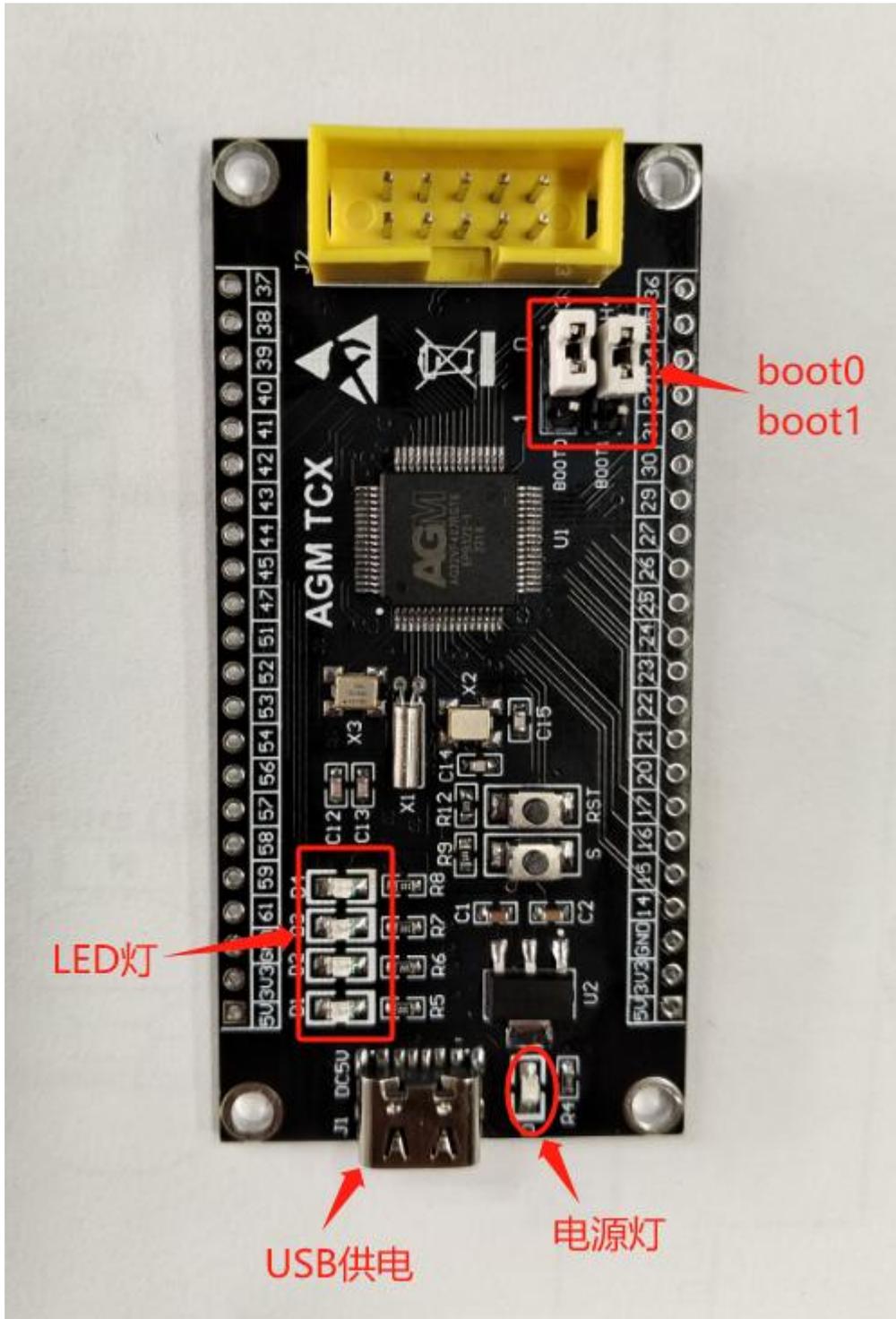
有三款开发板，分别是 100pin 的 AG32VF407、48pin 的 AG32VF103 和 64pin 的 AG32VF407。如下：



开发板 1: AG32VF407, 100pin



开发板 2: AG32VF103, 48pin



开发板 3: AG32VF407, 64pin

在使用开发板前，请确认已经安装好 SDK 开发环境。
安装环境过程，请参考文档《AG32 开发环境搭建.pdf》

以下先以开发板 1 为例描述使用过程。

开发板 2 和开发板 3 的使用略有不同，在后边会描述差异点。

注意：下边会涉及两个文件的修改：VE 文件和 platformio.ini 文件，修改后必须手动保存。
如果文件是只读属性，在弹出提醒时请点“覆盖”。

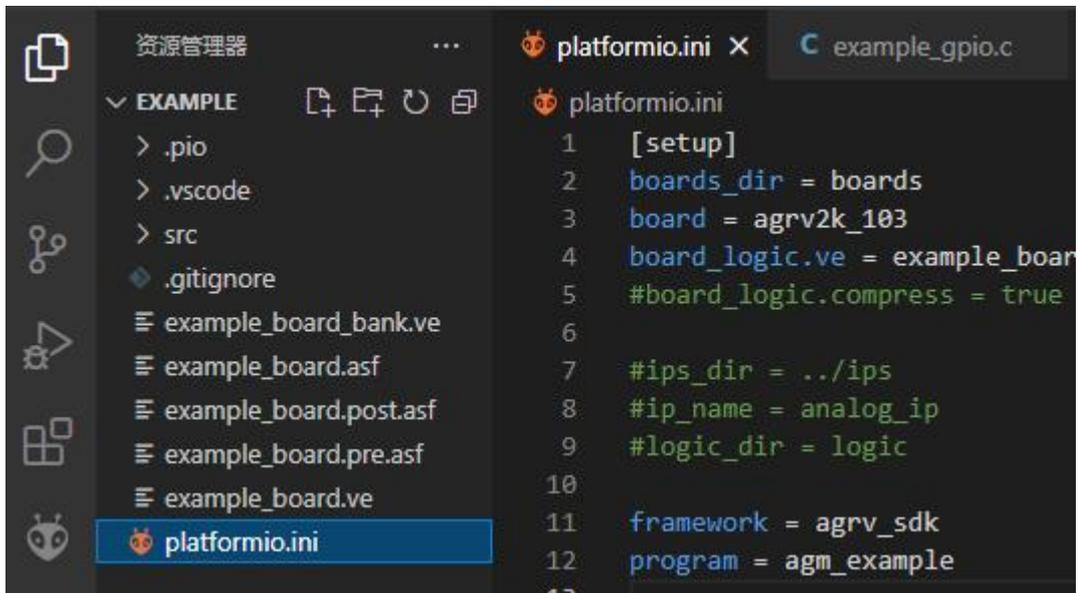
一、上电：

通过 USB 给开发板供电，可以看到板子上“电源灯”亮起。

二、使用 example 例程：

打开 example 例程，在 SDK 路径下：

D:\xxxx\AgRV_pio\platforms\AgRV\examples\example (注意这里的两重 example)



The screenshot shows a file explorer on the left with a tree view of a project directory. The right pane shows the contents of the platformio.ini file. The file explorer shows a directory named 'EXAMPLE' containing subdirectories '.pio', '.vscode', and 'src', and files like '.gitignore', 'example_board_bank.ve', 'example_board.asf', 'example_board.post.asf', 'example_board.pre.asf', 'example_board.ve', and 'platformio.ini'. The code editor shows the following configuration for platformio.ini:

```
platformio.ini
1 [setup]
2 boards_dir = boards
3 board = agrv2k_103
4 board_logic.ve = example_board
5 #board_logic.compress = true
6
7 #ips_dir = ../ips
8 #ip_name = analog_ip
9 #logic_dir = logic
10
11 framework = agrv_sdk
12 program = agm_example
13
```

由于开发板使用的是 407 芯片，需要先修改 platformio.ini 中的 board 类型：

board = agrv2k_103

修改为：

board = agrv2k_407

为了验证简单化，可以先把 example_board.ve 中的其他配置暂时删除，只留下 sysclk 和 led 的配置：

SYSCLK 100

HSECLK 8

GPIO4_1 PIN_32 # LED1

GPIO4_2 PIN_31 # LED2

如下图：

```
SYSCLK 100
HSECLK 8

GPIO4_1 PIN_32 # LED1
GPIO4_2 PIN_31 # LED2
```

三、烧录 VE 文件和代码 bin:

烧录程序需要使用 dap-link (AGM 专用) 或通用的 jlink; (串口烧录这里不做讨论)

Dap-link 和 Jlink 在跟开发板的连线上, 都是 jtag 的 swd 两线 (clk 和 tms) 模式。

配置上, 如果使用 Dap-link (AGM 专用), 需要在 platformio.ini 中的配置以下两行:

```
debug_tool = cmsis-dap-openocd
upload_protocol = cmsis-dap-openocd
```

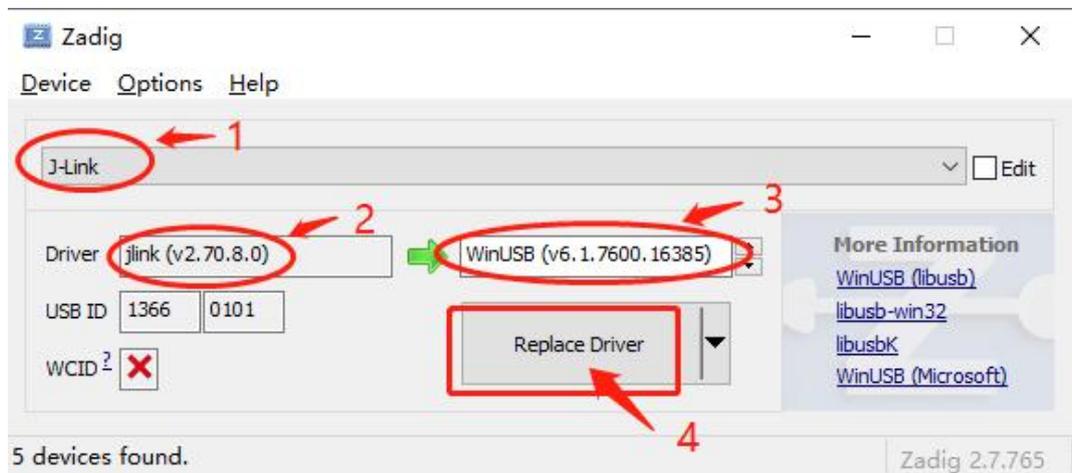
如果使用 Jlink, 需要在 platformio.ini 中的配置以下两行:

```
debug_tool = jlink-openocd
upload_protocol = jlink-openocd
```

如果使用 dap-link, 该烧录器是免驱动的, 不用安装任何驱动。

如果使用 Jlink, 需要在原有 Jlink 基础上安装插件 zadig。方法如下:

安装插件: 第一次使用 jlink, 需要先安装插件【zadig-2.8.exe】, 安装参考下图:
(该插件在 sdk 路径的根目录下)

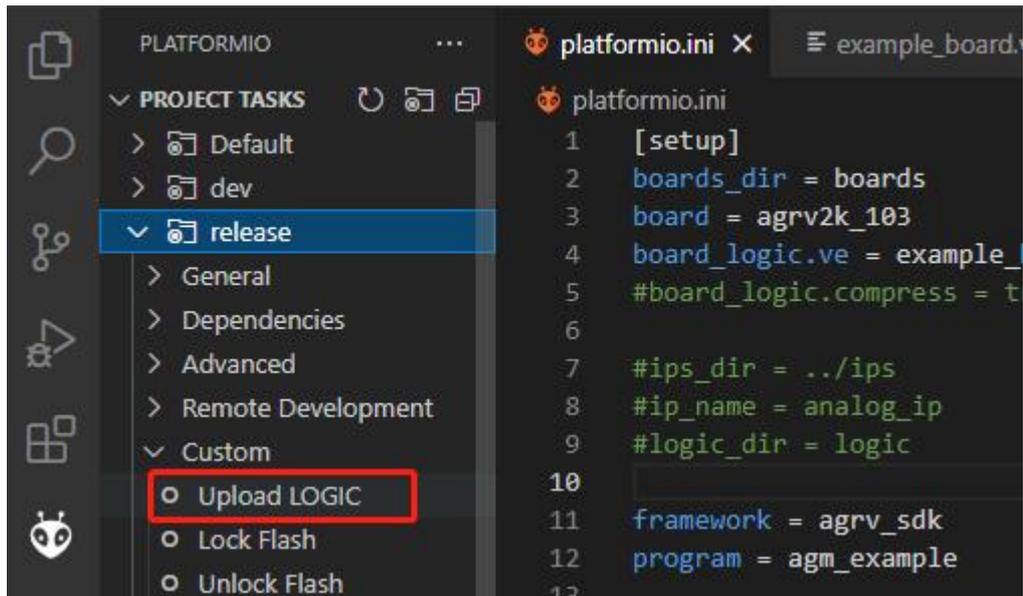


注: 如果第一步从下拉列表中找到【J-Link】项, 可以把下拉列表打开, 插拔 Jlink 几次, 找列表中的变化项。列表中的那个变化项, 就是要更新驱动的项目。

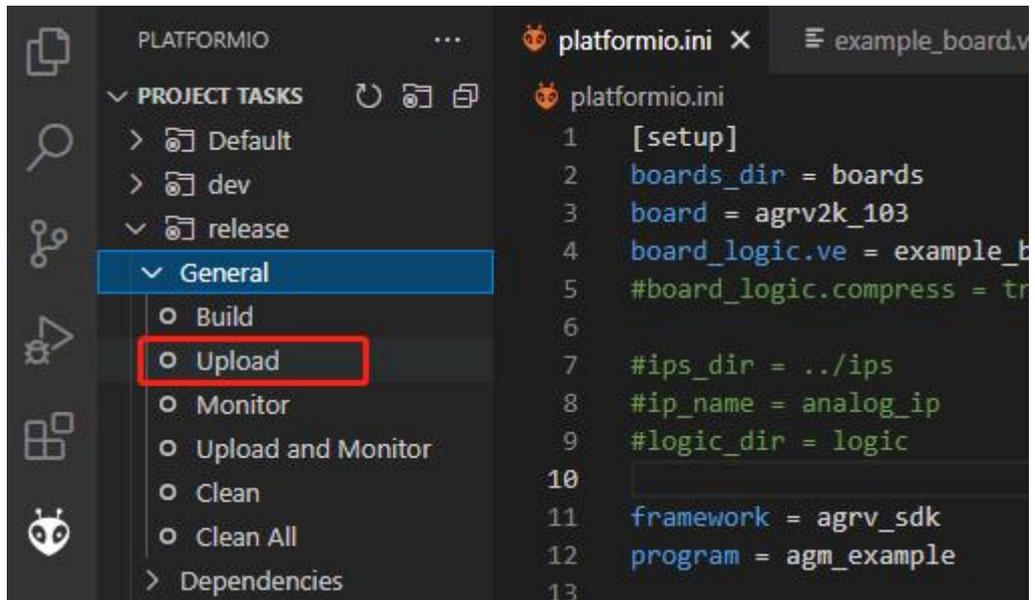
烧录:

新开发板第一次使用, 要先烧录 VE 配置。(不烧录 VE 而先烧录程序 bin, 会报错)

烧录 VE:

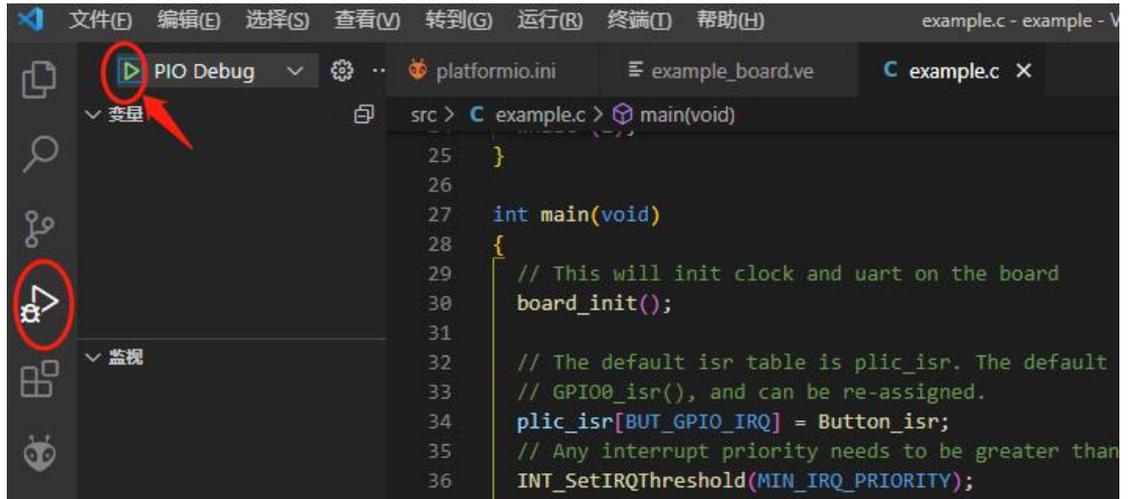


烧录程序：

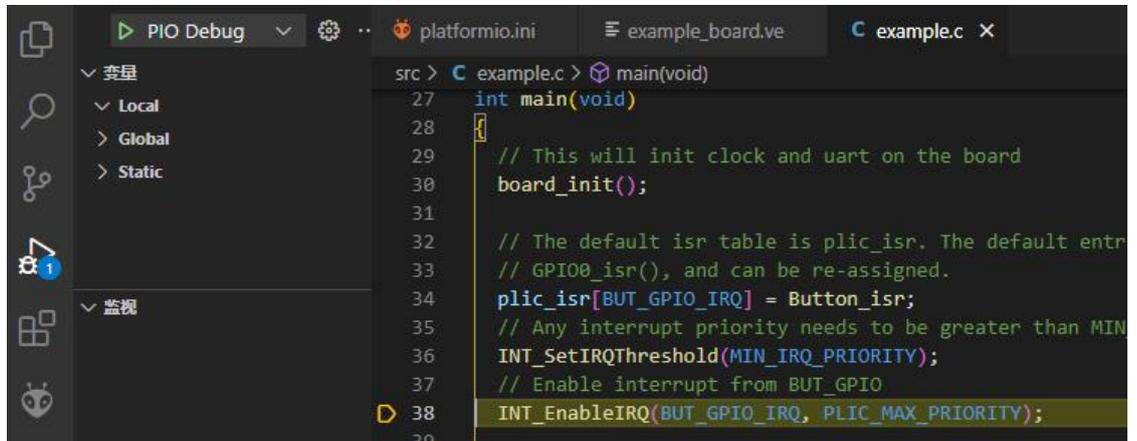


仿真：

点击仿真按钮，可以进入仿真调试。可单步运行到 main 函数的结尾。



单步状态下如图：



四、查看 led 灯：

在 example 样例程序的 main 函数中，最后是调用函数 TestGpio()。
进入 TestGpio()函数，里边是对 LED 灯的闪灯操作。

如果使用默认 example 程序，按前边的操作一路走下来，此时是可以看到左下角两个 led 灯一起闪烁的。

五、查看 log 输出：

在以上的基础上，修改以下三项：

1. Platformio.ini 中：

确认 logger_if 配置是打开的：

logger_if = UART0

build_flags = -DBAUD_RATE=115200

以上两项分别设置：log 输出通过 uart0 输出、输出的波特率是 115200.

2. Example_board.ve 中：

Copy 以下的串口 IO 配置到 ve 中去：

UART0_UARTRXD PIN_69

UART0_UARTTXD PIN_68

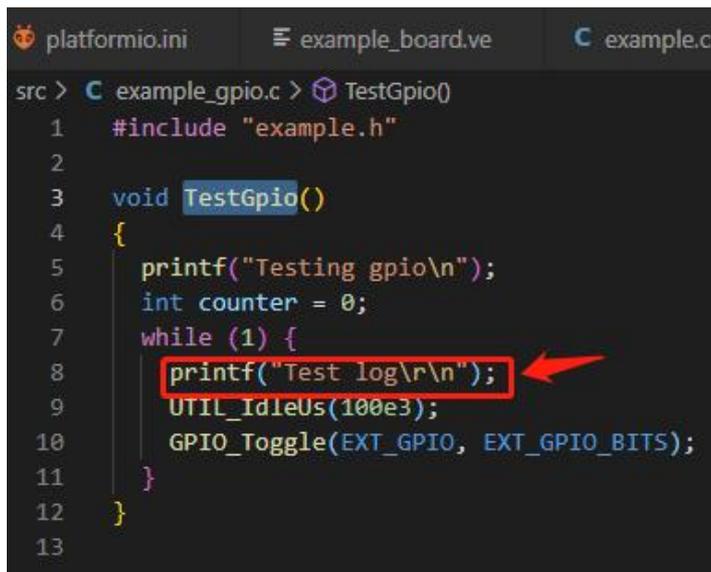
修改后图示如下：

```
SYSCLK 100
HSECLK 8

GPIO4_1 PIN_32 # LED1
GPIO4_2 PIN_31 # LED2

UART0_UARTRXD PIN_69
UART0_UARTTXD PIN_68
```

3. 在 example_gpio.c 中的 TestGpio()函数中，while(1)里增加一句 log:

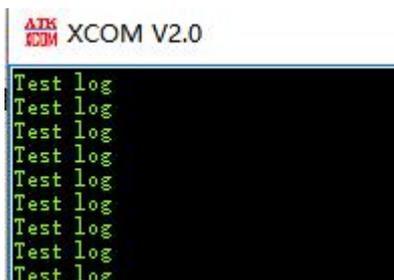


```
platformio.ini example_board.ve example.c
src > C example_gpio.c > TestGpio()
1 #include "example.h"
2
3 void TestGpio()
4 {
5     printf("Testing gpio\n");
6     int counter = 0;
7     while (1) {
8         printf("Test log\r\n");
9         UTIL_IdleUs(100e3);
10        GPIO_Toggle(EXT_GPIO, EXT_GPIO_BITS);
11    }
12 }
13
```

以上修改后，注意文件的保存。

最后，编译并烧录 VE，烧录程序 bin。（注：ve 和程序 两个都要烧录）

然后，用串口线，接到开发板的串口 0（参开发板 1 标识图）上，在 PC 端的串口工具（波特率 115200）上可以看到 log 的输出信息，如下图：



```
AT&K
XCOM V2.0
Test log
```

以上，只是展示了拿到开发板后验证 LED 灯和 log 通过串口 0 输出的样例。
更多的驱动使用，请参考文档《AG32 驱动的使用.pdf》

以下为开发板 2 的描述：

开发板 2 的所有操作都同开发板 1，这里列举和开发板 1 的差异点。

1. 开发板 2 是 jtag 两线连接（TMS 对应 IO，TCK 对应 clk）；
其实开发板 1 也可以是两线连接。

2. 开发板 2 在配置时，需要在 platformio.ini 修改两项：

```
board = agrv2k_103
```

```
board_logic.device = AGRV2KL48
```

3. 开发板 2 的 led 灯只有一个，对应 PIN_2（新开发板换成了 PIN_46）；
串口 0 对应 PIN_30 和 PIN_31；

VE 修改后对应关系：

```
GPIO4_2 PIN_2 # LED1
```

```
UART0_UARTRXD PIN_31
```

```
UART0_UARTTXD PIN_30
```

最终如图：

```
SYSCLK 100
HSECLK 8

GPIO4_2 PIN_2 # LED1

UART0_UARTRXD PIN_31
UART0_UARTTXD PIN_30
```

注意：上图的 PIN_2，是旧版的；新开发板换成了 PIN_46。具体是哪个依据拿的板子来定。

4. 注掉 example_board.asf 文件中对 PIN_23 的设置；

在 example_board.asf 文件中，有两行，是在 100PIN 中对 23 脚使用的样例，48PIN 里边不再需要，加#注掉即可。

```
example_board.asf X
example_board.asf
1 puts "***** DESIGN ASF *****"
2
3 # Enable PIN 23 (WKUP pin) to wake up device from standby. This funct
4 #set_config -pin PIN_23 CFG_WKUP_EN 1'b1
5 # Use falling edge of the WKUP pin. By default rising edge is used.
6 #set_config -pin PIN_23 CFG_WKUP_INV 1'b1
7
```

以上修改后，注意对修改过的文件的保存。

然后，烧录 VE，烧录程序，就可以看到 LED 闪烁，串口 log 输出。

如果开发板 2 出现烧录报错：Error: Error connecting DP: cannot read IDR，请检查 jlink 的两根线（TMS 和 TCK）连接是否正常。

注意：这个开发板的引脚 TMS 和 TCK，从板子正面看，它是对应上层的一层排针。串口 0 对应下层排针。

以下为开发板 3 的描述:

开发板 3 的所有操作都同开发板 1, 这里列举和开发板 1 的差异点。

1. 开发板 3 是 jtag 两线连接 (TMS 对应 IO, TCK 对应 clk);

其实开发板 1 也可以是两线连接。

2. 开发板 3 在配置时, 需要在 platformio.ini 修改两项:

board = agrv2k_407

board_logic.device = AGRV2KL64

3. 开发板 3 的 led 灯有 4 个, 对应 pin_8、pin_9、pin_10、pin_62;

串口 0 对应 PIN_42 和 PIN_43;

VE 修改后对应关系:

UART0_UARTRXD PIN_43

UART0_UARTTXD PIN_42

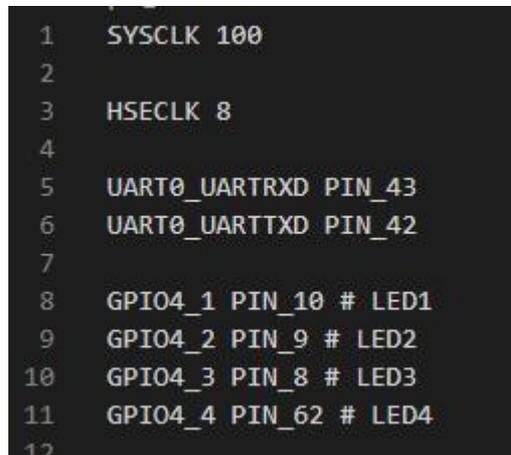
GPIO4_1 PIN_10 # LED1

GPIO4_2 PIN_9 # LED2

GPIO4_3 PIN_8 # LED3

GPIO4_4 PIN_62 # LED4

最终如图:



```
1 SYSClk 100
2
3 HSECLK 8
4
5 UART0_UARTRXD PIN_43
6 UART0_UARTTXD PIN_42
7
8 GPIO4_1 PIN_10 # LED1
9 GPIO4_2 PIN_9 # LED2
10 GPIO4_3 PIN_8 # LED3
11 GPIO4_4 PIN_62 # LED4
12
```

4. 注掉 example_board.asf 文件中对 PIN_23 的设置;

在 example_board.asf 文件中, 有两行, 是在 100PIN 中对 23 脚使用的样例, 64PIN 里边不再需要, 加#注掉即可。

```
example_board.asf X
example_board.asf
1 puts "***** DESIGN ASF *****"
2
3 # Enable PIN 23 (WKUP pin) to wake up device from standby. This funct
4 #set_config -pin PIN_23 CFG_WKUP_EN 1'b1
5 # Use falling edge of the WKUP pin. By default rising edge is used.
6 #set_config -pin PIN_23 CFG_WKUP_INV 1'b1
7
```

以上修改后，注意对修改文件的保存。

然后，烧录 VE，烧录程序，就可以看到 LED 闪烁，串口接 PIN42 后有 log 输出。