文档说明系列

PAN1080 HAL GPIO Sample Application Note

PAN-CLT-VER-B0, Rev 0.1

PanchipMicroelectronics

www.panchip.com

修订历史

版本	修订日期	描述
V0.1	2023-09-27	初始版本创建

目录

第1章 例程演示内容

1.1 测试内容

a) 寄存器默认值 （Register default value）
b) 四种功能模式 （Four I/O modes）
   1. 准双向输入输出模式 （Quasi-bidirectional mode）
   2. 推挽输出模式 （Push-pull output）
   3. 漏极开路输出模式 （Open-drain output）
   4. 高阻态输入模式 （Input-only with high impendence）
c) 五种中断触发方式 （Five types of interrupt condition）
   1. 低电平触发 （Low level trigger）
   2. 高电平触发 （High level trigger）
   3. 下降沿触发 （Falling edge trigger）
   4. 上升沿触发 （Rising edge trigger）
   5. 上升下降沿触发 （Both rising and falling edge trigger）
d) 唤醒功能 （Wake-up function）
e) 去抖功能 （De-bounce function）
f) 输出寄存器写锁定功能 （Digital-out register mask function）

1.2 环境配置

1.2.1 软件环境

例程工程文件：

<PAN1080-DK>\03_MCU\mcu_samples_hal\GPIO\keil\GPIO.uvprojx

例程源文件目录：

<PAN1080-DK>\03_MCU\mcu_samples_hal\GPIO\src

1.2.2 硬件环境

1、 PN1080 EVB 一块

2、 串口（打印接口，TX: P00，RX: P01，波特率 921600）

3、 SecureCRT（串口打印工具）

4、 逻辑分析仪（波形抓取工具）

5、 JLink（SWD 下载工具，SDCLK: P46，SDIO: P47）

6、 EVB 未占用的 IO 口：

a) 将待测的 GPIO PIN 任意两两相接，演示代码默认接线方式如下（gpio_common.c 中定义）：

Group1	Group2
P02	P03
P20	P21
P30	P31

b) EVB 上的某些引脚不可用于 GPIO 例程测试，例如 P00/P01 用作串口打印，P46/P47 用作 SWD 下载程序，P04/P05 外接了电容和按键。

第2章 例程演示流程

2.1 环境说明

打开演示工程，确保可以编译通过。编译烧录成功后，使用跳线帽将 P00 接到 TX0，P01 接到 RX0，然后通过 USB-TypeC 线与 PC 对应串口连接，并配置好串口工具，稍后通过输入测试命令观察演示结果。

2.2 演示代码流程

2.2.1 Push-pull Mode

1、 按分组将 Group1 和 Group2 引脚对应连接

2、 配置通用 GPIO 口

3、 配置 Group1 的引脚作为 Push-pull 输出，Group2 作为输入

4、 将 Group1 的引脚置低，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

5、 再将 Group1 的引脚置高，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

6、 将 Group1 和 Group2 的功能对调，Group2 的引脚作为 Push-pull 输出，Group1 作为输入，再进行以上操作（确保每个管脚的输入输出功能都能测试到）

2.2.2 Open-drain Mode

1、 按分组将 Group1 和 Group2 引脚对应连接

2、 配置通用 GPIO 口

3、 配置 Group1 的引脚作为 Open-drain 输出，并通过内部或外部上拉电阻拉高，Group2 作为输入

4、 将 Group1 的引脚置低，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

5、 再将 Group1 的引脚置高，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

6、 将 Group1 和 Group2 的功能对调，Group2 的引脚作为 Open-drain 输出，Group1 作为输入，再进行以上操作（确保每个管脚的输入输出功能都能测试到）

2.2.3 Quasi-bidirectional Mode

1、 按分组将 Group1 和 Group2 引脚对应连接

2、 配置通用 GPIO 口

3、 配置 Group1 的引脚作为 Quasi 模式，并通过内部或外部上拉电阻拉高，Group2 作为辅助测试 IO

4、 将 Group1 的引脚置低，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

5、 再将 Group1 的引脚置高，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

6、 将 Group2 配置为 Push-pull 输出，Group1 的引脚 DOUT 置高从而可以作为 Quasi 输入

7、 将 Group2 的引脚置低，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

8、 将 Group2 的引脚置高，测量 Group1 的引脚状态并读取 Group2 引脚状态打印出来并判断是否对应一致

9、 将 Group1 和 Group2 的功能对调，Group2 的引脚配置为 Quasi 模式，Group1 作为辅助测试 IO，再进行以上操作（确保每个管脚的输入输出功能都能测试到）

2.2.4 去抖测试

1、 按分组将 Group1 和 Group2 引脚对应连接

2、 配置通用 GPIO 口

3、 配置 Group1 的引脚作为输入，用来检测中断；Group2 为 Push-pull 输出，用来控制高电平维持时间

4、 Group1 使能上升沿中断触发条件，并开启去抖功能，配置去抖周期为 8 个 cycle 并修改周期多次测试

5、 通过 Group2 引脚提供 7 个上升沿，且每个上升沿高电平保持时间 1 个 nop 递增

6、 连续输入测试指令观察打印中断触发次数

2.2.5 中断测试

1、 按分组将 Group1 和 Group2 引脚对应连接

2、 配置通用 GPIO 口，Group1 的引脚作为输入，Group2 为 Push-pull 输出

3、 Group1 使能上升沿中断触发条件，Group2 引脚提供上升沿，观察是否有中断触发（通过 log 打印）

4、 Group1 使能下降沿中断触发条件，Group2 引脚提供下降沿，观察是否有中断触发（通过 log 打印）

5、 Group1 使能双边沿中断触发条件，Group2 引脚提供上升沿和下降沿，观察是否触发两次中断（通过 log 打印）

6、 Group1 使能高电平中断触发条件，Group2 引脚提供高电平，观察是否有中断触发（通

过 log 打印）

7、 Group1 使能低电平中断触发条件，Group2 引脚提供低电平，观察是否有中断触发（通过 log 打印）

2.2.6 唤醒测试

TBD

2.2.7 输出寄存器写锁定测试

1、 按分组将 Group1 和 Group2 引脚对应连接

2、 配置通用 GPIO 口

3、 配置 Group1 的引脚作为 Push-pull 输出，Group2 作为输入

4、 将 Group1 的引脚拉低，并使用 Group2 引脚检查是否拉低成功

5、 使能 Group1 引脚的输出寄存器写锁定（DOUT Write Mask）功能

6、 修改 Group1 引脚的输出寄存器对应位为 1，并使用 Group2 引脚检查是否拉高成功

7、 除能 Group1 引脚的输出寄存器写锁定（DOUT Write Mask）功能

8、 再修改 Group1 引脚的输出寄存器对应位为 1，并使用 Group2 引脚检查是否拉高成功

9、 使能 Group1 引脚的输出寄存器写锁定（DOUT Write Mask）功能

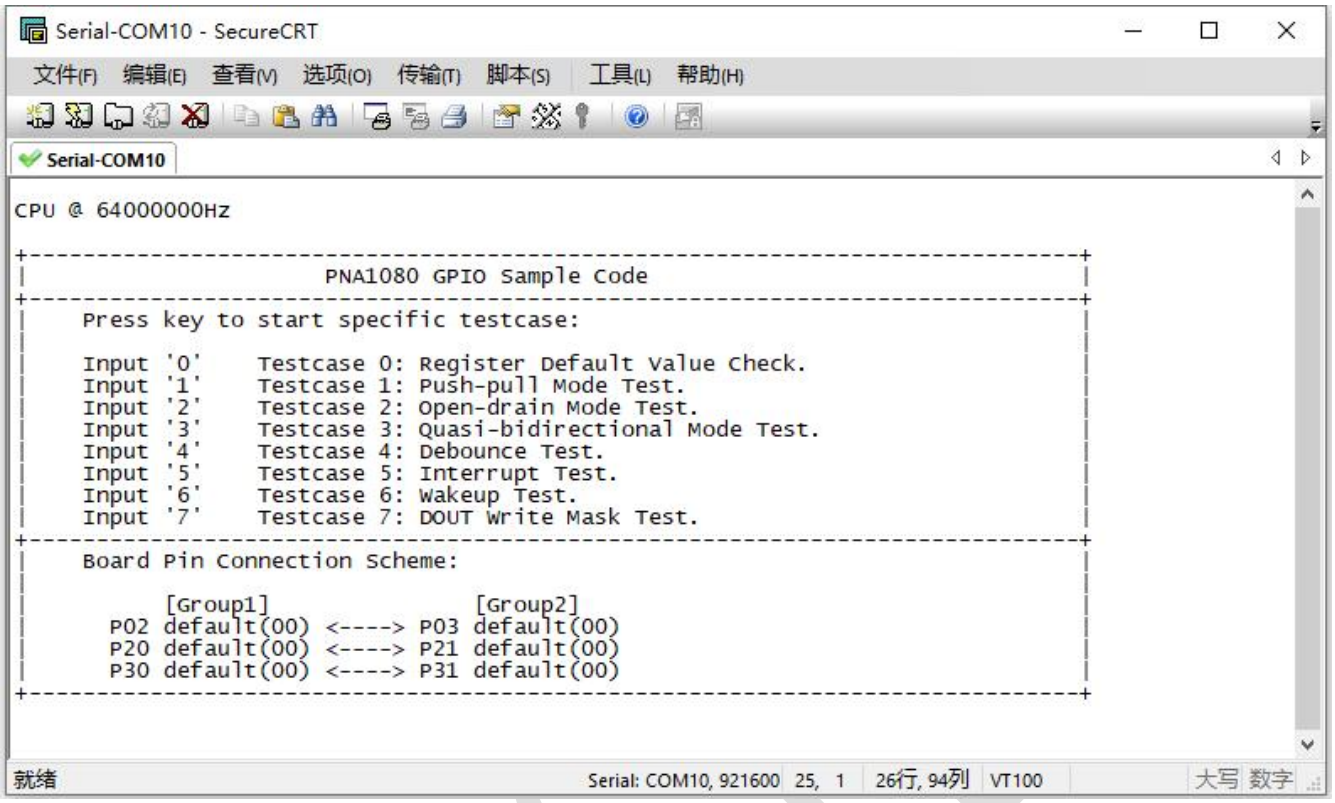
10、 修改 Group1 引脚的输出寄存器对应位为 0，并使用 Group2 引脚检查是否拉低成功

11、 除能 Group1 引脚的输出寄存器写锁定（DOUT Write Mask）功能

12、 再修改 Group1 引脚的输出寄存器对应位为 0，并使用 Group2 引脚检查是否拉低成功

2.3 演示步骤

2.3.1 UART 初始化



根据串口说明连接好串口；

初始化成功后，输出：

1. GPIO 端口分组信息；
2. 测试命令对应的测试用例。

2.3.2 基本功能验证

2.3.2.1 GPIO 所有寄存器默认状态

输入‘0’命令 打印所有寄存器默认值：

测试目的：

查看所有 GPIO 相关寄存器复位 Default 值状态。

测试预期：

除系统配置对 P0_DINOFF 和 P0_PIN 寄存器修改外，其他寄存器应和 PAN1080 Datasheet 上 GPIO 模块默认值一致。

测试现象：

```
0
GPIO Register Default Values:
--------------------------
P0_MODE     = 0x00000000
P1_MODE     = 0x00000000
P2_MODE     = 0x00000000
P3_MODE     = 0x00000000
P4_MODE     = 0x00000000
P5_MODE     = 0x00000000
--------------------------
P0_DINOFF   = 0x00fd0000
P1_DINOFF   = 0x00ff0000
P2_DINOFF   = 0x00ff0000
P3_DINOFF   = 0x00ff0000
P4_DINOFF   = 0x003f00c0
P5_DINOFF   = 0x00ff0000
--------------------------
P0_DOUT     = 0x000000ff
P1_DOUT     = 0x000000ff
P2_DOUT     = 0x000000ff
P3_DOUT     = 0x000000ff
P4_DOUT     = 0x000000ff
P5_DOUT     = 0x000000ff
--------------------------
P0_DATMSK   = 0x00000000
P1_DATMSK   = 0x00000000
P2_DATMSK   = 0x00000000
P3_DATMSK   = 0x00000000
P4_DATMSK   = 0x00000000
P5_DATMSK   = 0x00000000
--------------------------
P0_PIN      = 0x00000002
P1_PIN      = 0x00000000
P2_PIN      = 0x00000000
P3_PIN      = 0x00000000
P4_PIN      = 0x000000c0
P5_PIN      = 0x00000000
--------------------------
P0_DBEN     = 0x00000000
P1_DBEN     = 0x00000000
P2_DBEN     = 0x00000000
P3_DBEN     = 0x00000000
P4_DBEN     = 0x00000000
P5_DBEN     = 0x00000000
--------------------------
P0_INTTYPE  = 0x00000000
P1_INTTYPE  = 0x00000000
P2_INTTYPE  = 0x00000000
P3_INTTYPE  = 0x00000000
P4_INTTYPE  = 0x00000000
P5_INTTYPE  = 0x00000000
--------------------------
P0_INTEN    = 0x00000000
P1_INTEN    = 0x00000000
P2_INTEN    = 0x00000000
P3_INTEN    = 0x00000000
P4_INTEN    = 0x00000000
P5_INTEN    = 0x00000000
--------------------------
P0_INTSRC   = 0x00000000
P1_INTSRC   = 0x00000000
P2_INTSRC   = 0x00000000
P3_INTSRC   = 0x00000000
P4_INTSRC   = 0x00000000
P5_INTSRC   = 0x00000000
--------------------------
GPIO_DBCTL = 0x00000020
--------------------------
GPIO Test OK, Success case: 0.
```

测试分析：

参考芯片手册对比寄存器信息，大部分与手册一致，其中：

P4_6 与 P4_7 上电默认是 SWD_CLK 和 SWD_IO 功能（Digital Input），因此 P4_DINOFF 与 P4_PIN 相应这两个位与其他位不同；

P0_1 在测试工程中被配置为 UART_RX 功能（Digital Input），因此 P0_DINOFF 与 P0_PIN 相应位与其他位不同；

其他寄存器复位值与芯片手册默认值一致，符合预期。

2.3.2.2 GPIO 四种功能模式

输入‘1’命令 推挽输出测试 GPIO_PushPullModeTestCase1()：

测试目的：

按照分组信息一一对应连接相关引脚，配置 Group1 为推挽输出模式，Group2 为输入模式，分别将输出置低再置高，通过抓取输出引脚状态并读取输入引脚状态，验证推挽输出模式下引脚控制是否正确；然后再将 Group1 配置为输入，Group2 为推挽输出再测一遍。

测试预期：

打印输出每根测试引脚的推挽输出测试结果“GPIO Test OK, Success case: 1, Success Pin: P[xy]”，并且输入输出引脚状态对应相同，抓取输出引脚波形先置低后置高。

测试现象：

```
1
GPIO Test OK, Success case: 1, Success Pin: P02
GPIO Test OK, Success case: 1, Success Pin: P03
GPIO Test OK, Success case: 1, Success Pin: P20
GPIO Test OK, Success case: 1, Success Pin: P21
GPIO Test OK, Success case: 1, Success Pin: P30
GPIO Test OK, Success case: 1, Success Pin: P31
```

测试分析：

Group1 和 Group2 引脚分别作为 push-pull 输出，置低置高现象符合预期。

输入‘2’命令 开漏输出测试 GPIO_OpenDrainModeTestCase2()：

测试目的：

将 Group1 的引脚配置为 Open-drain 输出模式，并通过内部或外部上拉电阻拉高，Group2 配置为输入，通过拉高拉低输出端，打印输入输出状态，读取输出引脚信息，并 check 是否正确。

测试预期：

无论是使用内部上拉电阻（gpio_test.h 中打开宏 ENABLE_INTERNAL_PULLUP_RES）还是使用外部上拉电阻（gpio_test.h 中关闭宏 ENABLE_INTERNAL_PULLUP_RES 且确保芯片外部有接上拉电阻），均可打印输出“GPIO Test OK, Success case: 2, Success Pin: P[xy]”，输入输出

引脚状态对应相同。

测试现象：

```
2
GPIO Test OK, Success case: 2, Success Pin: P02
GPIO Test OK, Success case: 2, Success Pin: P03
GPIO Test OK, Success case: 2, Success Pin: P20
GPIO Test OK, Success case: 2, Success Pin: P21
GPIO Test OK, Success case: 2, Success Pin: P30
GPIO Test OK, Success case: 2, Success Pin: P31
```

测试分析：

拉高拉低输出引脚，check 输出引脚的输出值正确。

输入‘3’命令 准双向模式测试 GPIO_QuasiBidirectionalModeTestCase3()：

测试目的：

将 Group1 的引脚配置为 Quasi 输出模式，并通过内部或外部上拉电阻拉高，Group2 配置为输入，通过拉高拉低输出端，打印输入输出状态，读取输出引脚信息，并 check 是否正确，然后配置 Group1 引脚作为 Quasi 输入，Group2 为推挽输出，拉低再拉高输出端，check 输入端状态是否随之改变；将 Group1 与 Group2 配置的功能交换再测一遍。

测试预期：

无论是使用内部上拉电阻（gpio_test.h 中打开宏 ENABLE_INTERNAL_PULLUP_RES）还是使用外部上拉电阻（gpio_test.h 中关闭宏 ENABLE_INTERNAL_PULLUP_RES 且确保芯片外部有接上拉电阻），均可打印输出“GPIO Test OK, Success case: 3, Success Pin: P[xy]”，输入输出引脚状态对应相同。

测试现象：

```
3
GPIO Test OK, Success case: 3, Success Pin: P02
GPIO Test OK, Success case: 3, Success Pin: P03
GPIO Test OK, Success case: 3, Success Pin: P20
GPIO Test OK, Success case: 3, Success Pin: P21
GPIO Test OK, Success case: 3, Success Pin: P30
GPIO Test OK, Success case: 3, Success Pin: P31
```

测试分析：

Quasi 输出与输入模式，高低电平均验证 OK。

2.3.2.3 去抖功能测试

输入‘4’命令：

测试目的：

测试边沿触发条件输入信号的去抖功能，去抖周期控制。

测试预期：

增大去抖周期，中断产生变少。

测试现象：

```
4
Debounce Period - 8 Cycles HCLK
P0_2 INT occurred, type: GPIO_INT_RISING.
P0_2 INT occurred, type: GPIO_INT_RISING.
P0_2 INT occurred, type: GPIO_INT_RISING.
P0_2 INT occurred, type: GPIO_INT_RISING.
Debounce Period - 32 Cycles HCLK
P0_2 INT occurred, type: GPIO_INT_RISING.
GPIO Test OK, Success case: 4, Success Pin: P02
Debounce Period - 8 Cycles HCLK
P0_3 INT occurred, type: GPIO_INT_RISING.
P0_3 INT occurred, type: GPIO_INT_RISING.
Debounce Period - 32 Cycles HCLK
P0_3 INT occurred, type: GPIO_INT_RISING.
GPIO Test OK, Success case: 4, Success Pin: P03
Debounce Period - 8 Cycles HCLK
P2_0 INT occurred, type: GPIO_INT_RISING.
P2_0 INT occurred, type: GPIO_INT_RISING.
Debounce Period - 32 Cycles HCLK
P2_0 INT occurred, type: GPIO_INT_RISING.
GPIO Test OK, Success case: 4, Success Pin: P20
Debounce Period - 8 Cycles HCLK
P2_1 INT occurred, type: GPIO_INT_RISING.
P2_1 INT occurred, type: GPIO_INT_RISING.
Debounce Period - 32 Cycles HCLK
P2_1 INT occurred, type: GPIO_INT_RISING.
GPIO Test OK, Success case: 4, Success Pin: P21
Debounce Period - 8 Cycles HCLK
P3_0 INT occurred, type: GPIO_INT_RISING.
P3_0 INT occurred, type: GPIO_INT_RISING.
Debounce Period - 32 Cycles HCLK
P3_0 INT occurred, type: GPIO_INT_RISING.
GPIO Test OK, Success case: 4, Success Pin: P30
Debounce Period - 8 Cycles HCLK
P3_1 INT occurred, type: GPIO_INT_RISING.
P3_1 INT occurred, type: GPIO_INT_RISING.
Debounce Period - 32 Cycles HCLK
P3_1 INT occurred, type: GPIO_INT_RISING.
GPIO Test OK, Success case: 4, Success Pin: P31
```

测试分析：

去抖周期设置为 8 个 HCLK 时间时，每根引脚都触发了 2 次中断，增大去抖周期为 32 个 HCLK 时间，每根引脚只触发了 1 次中断，可见去抖周期设置不同，中断触发次数会发生改变，且中断次数随周期增大而减少，功能正常。

另外，注意到 P00 引脚，在去抖周期设置为 8 个 HCLK 时间时，触发了 4 次中断（而不是 2 次），产生这个现象的原因是模拟抖动的测试函数 SimulateBounceWave()第一次运行需要从 Flash 先搬移到芯片的 I-Cache 中，因此这段代码第一次运行速度较慢，导致各模拟抖动的持续时间比理论上要长，于是有些抖动被 GPIO 识别为正常信号。当模拟抖动函数第二次及以后运行的时候，由于其已经暂存在 Cache 中，运行速度变快，后续 PIN 的 debounce 测试结果就正常了。

2.3.2.4 GPIO 五种中断触发条件

输入‘5’命令：测试 5 种中断触发条件

测试目的：

分别验证：

a) 上升沿触发条件开启时，提供上升沿，是否能触发中断；
b) 下降沿触发条件开启时，提供下降沿，是否能触发中断；
c) 双边沿触发条件开启时，提供双边沿，是否能触发中断；
d) 高电平触发条件开启时，提供高电平，是否能触发中断；
e) 低电平触发条件开启时，提供低电平，是否能触发中断；

测试预期：

上升沿到来时，打印“P[x]_[y] INT occurred, type: GPIO_INT_RISING.”
下降沿到来时，打印“P[x]_[y] INT occurred, type: GPIO_INT_FALLING.”
双边沿到来时，打印两次“P[x]_[y] INT occurred, type: GPIO_INT_BOTH_EDGE.”
高电平到来时，打印“P[x]_[y] INT occurred, type: GPIO_INT_HIGH.”
低电平到来时，打印“P[x]_[y] INT occurred, type: GPIO_INT_LOW.”

测试现象：

```
5
P0_2 INT occurred, type: GPIO_INT_RISING.
P0_2 INT occurred, type: GPIO_INT_FALLING.
P0_2 INT occurred, type: GPIO_INT_BOTH_EDGE.
P0_2 INT occurred, type: GPIO_INT_BOTH_EDGE.
P0_2 INT occurred, type: GPIO_INT_HIGH.
P0_2 INT occurred, type: GPIO_INT_LOW.
GPIO Test OK, Success case: 5, Success Pin: P02
P0_3 INT occurred, type: GPIO_INT_RISING.
P0_3 INT occurred, type: GPIO_INT_FALLING.
P0_3 INT occurred, type: GPIO_INT_BOTH_EDGE.
P0_3 INT occurred, type: GPIO_INT_BOTH_EDGE.
P0_3 INT occurred, type: GPIO_INT_HIGH.
P0_3 INT occurred, type: GPIO_INT_LOW.
GPIO Test OK, Success case: 5, Success Pin: P03
P2_0 INT occurred, type: GPIO_INT_RISING.
P2_0 INT occurred, type: GPIO_INT_FALLING.
P2_0 INT occurred, type: GPIO_INT_BOTH_EDGE.
P2_0 INT occurred, type: GPIO_INT_BOTH_EDGE.
P2_0 INT occurred, type: GPIO_INT_HIGH.
P2_0 INT occurred, type: GPIO_INT_LOW.
GPIO Test OK, Success case: 5, Success Pin: P20
P2_1 INT occurred, type: GPIO_INT_RISING.
P2_1 INT occurred, type: GPIO_INT_FALLING.
P2_1 INT occurred, type: GPIO_INT_BOTH_EDGE.
P2_1 INT occurred, type: GPIO_INT_BOTH_EDGE.
P2_1 INT occurred, type: GPIO_INT_HIGH.
P2_1 INT occurred, type: GPIO_INT_LOW.
GPIO Test OK, Success case: 5, Success Pin: P21
P3_0 INT occurred, type: GPIO_INT_RISING.
P3_0 INT occurred, type: GPIO_INT_FALLING.
P3_0 INT occurred, type: GPIO_INT_BOTH_EDGE.
P3_0 INT occurred, type: GPIO_INT_BOTH_EDGE.
P3_0 INT occurred, type: GPIO_INT_HIGH.
P3_0 INT occurred, type: GPIO_INT_LOW.
GPIO Test OK, Success case: 5, Success Pin: P30
P3_1 INT occurred, type: GPIO_INT_RISING.
P3_1 INT occurred, type: GPIO_INT_FALLING.
P3_1 INT occurred, type: GPIO_INT_BOTH_EDGE.
P3_1 INT occurred, type: GPIO_INT_BOTH_EDGE.
P3_1 INT occurred, type: GPIO_INT_HIGH.
P3_1 INT occurred, type: GPIO_INT_LOW.
GPIO Test OK, Success case: 5, Success Pin: P31
```

测试分析：

每根引脚都触发了相应条件下的中断，符合预期。

2.3.2.5 唤醒功能测试

TBD

2.3.2.6 输出寄存器写锁定功能测试

输入‘7’命令：测试引脚输出寄存器写锁定功能

测试目的：

验证输出寄存器写锁定（DOUT Write Mask）功能。

测试预期：

默认情况下，GPIO 输出寄存器（DOUT）可以随便改写，当使能写锁定功能后，输出寄存器变为不可写。

测试现象：

```
7
GPIO Test OK, Success case: 7, Success Pin: P02
GPIO Test OK, Success case: 7, Success Pin: P03
GPIO Test OK, Success case: 7, Success Pin: P20
GPIO Test OK, Success case: 7, Success Pin: P21
GPIO Test OK, Success case: 7, Success Pin: P30
GPIO Test OK, Success case: 7, Success Pin: P31
```

测试分析：

测试用例的流程：

a) 将测试引脚输出寄存器清 0
b) 使能写锁定，然后向输出寄存器写 1，如果写成功则提示出错，写失败则符合预期；
c) 除能写锁定，然后向输出寄存器写 1，如果写失败则提示出错，写成功则符合预期；
d) 使能写锁定，然后向输出寄存器写 0，如果写成功则提示出错，写失败则符合预期；
e) 除能写锁定，然后向输出寄存器写 0，如果写失败则提示出错，写成功则符合预期。

所有 case 都符合预期，则打印 GPIO Test OK。

第3章 使用注意事项

1、 将 GPIO 引脚设置为输入模式或准双向模式后，需要 Enable Digital Path 方可接受数字信号输入；

2、 将 GPIO 引脚设置为推挽输出，若输出高电平，应延时一段时间（需根据实际情况设置，范围一般在 4~1000 NOP 之间）再去检测辅助测试引脚的输入电平是否被拉高；