简介：

MFPR 寄存器结构：

Figure 8:MFPR Structure

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved																PULL_SEL	PULLUP_EN	PULLDN_EN	DRIVE		Reserved	SLEEP_SEL	SLEEP_DATA	SLEEP_DIR	EDGE_CLEAR	EDGE_FALL_EN	EDGE_RISE_EN	SLEEP_SEL1	AF_SEL		
Default	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	1	0	0	0	0	?	0	0	0	1	0	0	?	0	0	0

大多数 ASR1603 I/O 引脚都是多路复用的。这些引脚也称为多功能引脚，因为它们中的每一个都分配了多个功能。每个引脚都有一个专用的多功能引脚寄存器（MFPR），用于控制其功能和行为。 MFPR 的主要目的是，从分配给多功能引脚的几个功能中，选择出一个。
其他的 ASR1603 I/O 引脚是专用引脚，仅分配给一个功能。这些引脚没有与它们相关的 MFPR。

任何未选择的功能在逻辑上都与用于输入和输出的引脚断开。在这种情况下，输出信号被忽略，并且输入信号在被馈送到内部设备时，被强制为默认状态。
在所有情况下，引脚的方向由硬件控制。不同功能的配置将引脚连接到其驱动逻辑，该逻辑负责确定引脚是输入还是输出。在 GPIO 功能的情况下，使用 GPIO 控制器的方向寄存器，通过 GPIO 功能控制方向。

可以将多个引脚分配给相同的功能，输出数据和使能信号被复制，输入信号被“或”在一起，或者如果低电平则为“非”（反向 AND），以形成单个功能。软件必须避免将多个引脚分配给同一功能的情况，或者，如果需要，可以管理任何复杂情况。

示例：

```
static unsigned int const gpio_key_mfp_cfgs_awake[] = {
    MFP_REG(GPIO_12) | MFP_AF0 | MFP_DRIVE_MEDIUM | MFP_PULL_HIGH | MFP_SLEEP_NONE | MFP_LPM_EDGE_NONE,
    MFP_EOC /*End of configuration, must have */
};
mfp_config((unsigned int*)gpio_key_mfp_cfgs_awake);
```

参数 1：

```
#define MFP_AF0          MFP(0x0000, 0, 0, 0, 0, 0, 0)
#define MFP_AF1          MFP(0x0000, 0, 0, 0, 0, 0, 1)
#define MFP_AF2          MFP(0x0000, 0, 0, 0, 0, 0, 2)
#define MFP_AF3          MFP(0x0000, 0, 0, 0, 0, 0, 3)
#define MFP_AF4          MFP(0x0000, 0, 0, 0, 0, 0, 4)
#define MFP_AF5          MFP(0x0000, 0, 0, 0, 0, 0, 5)
#define MFP_AF6          MFP(0x0000, 0, 0, 0, 0, 0, 6)
#define MFP_AF7          MFP(0x0000, 0, 0, 0, 0, 0, 7)
#define MFP_AF_MASK  MFP(0x0000, 0, 0, 0, 0, 0, 7)
```

以上这些，设置 GPIO 用作哪个功能。具体的功能表参见 Crane_GPIO_1106NEW.pdf

参数 2：

```
#define MFP_SLEEP_NONE              MFP(0x0000, 0, 0, 0, 0, 0, 0)
#define MFP_SLEEP_DIR               MFP(0x0000, 0, 0, 0, 0, 2, 0)
#define MFP_SLEEP_DATA              MFP(0x0000, 0, 0, 0, 0, 4, 0)
#define MFP_SLEEP_FLOAT             MFP(0x0000, 0, 0, 0, 0, 0xb, 0)
#define MFP_SLEEP_OUTPUT_HIGH       MFP(0x0000, 0, 0, 0, 0, 0xd, 0)
#define MFP_SLEEP_OUTPUT_LOW        MFP(0x0000, 0, 0, 0, 0, 0x9, 0)
#define MFP_SLEEP_MASK              MFP(0x0000, 0, 0, 0, 0, 0xf, 0)
```

MFP_SLEEP_NONE 不使能 pad 的 sleep 功能
MFP_SLEEP_DIR 不使能 sleep，只是设置为 sleep 时候，输入状态
MFP_SLEEP_DATA 不使能 sleep，只是设置为 sleep 时候，输出状态
MFP_SLEEP_FLOAT 使能 sleep 功能，设置为 sleep 时候，输入状态
MFP_SLEEP_OUTPUT_HIGH 使能 sleep 功能，设置为 sleep 时候，输出高状态
MFP_SLEEP_OUTPUT_LOW 使能 sleep 功能，设置为 sleep 时候，输出低状态
其他的用不到

参数 3：

```
#define MFP_LPM_EDGE_NONE   MFP(0x0000, 0, 0, 0, 4, 0, 0)
#define MFP_LPM_EDGE_RISE   MFP(0x0000, 0, 0, 0, 1, 0, 0)
#define MFP_LPM_EDGE_FALL   MFP(0x0000, 0, 0, 0, 2, 0, 0)
#define MFP_LPM_EDGE_BOTH   MFP(0x0000, 0, 0, 0, 3, 0, 0)
#define MFP_LPM_EDGE_MASK   MFP(0x0000, 0, 0, 0, 7, 0, 0)
```

MFP_LPM_EDGE_NONE 不使能 pad 的边沿检测功能
MFP_LPM_EDGE_RISE 使能 pad 的边沿检测功能，且是上升沿。并提供唤醒事件。
MFP_LPM_EDGE_FALL 使能 pad 的边沿检测功能，且是下降沿。并提供唤醒事件。
MFP_LPM_EDGE_BOTH 使能 pad 的边沿检测功能，且是双沿。并提供唤醒事件。

边沿检测逻辑的输出主要用于在应用子系统 D1/D2/D3 电源状态或通信子系统 D2 电源状态中提供唤醒事件，并且不连接到 GPIO 控制器的边沿检测功能。

示例：

```
static unsigned int const gpio_key_mfp_cfgs_sleep[] = {
    MFP_REG(GPIO_12) | MFP_AF0 | MFP_DRIVE_MEDIUM | MFP_PULL_HIGH |
MFP_SLEEP_NONE | MFP_LPM_EDGE_FALL,
    MFP_EOC
};
static unsigned int const gpio_key_mfp_cfgs_awake[] = {
    MFP_REG(GPIO_12) | MFP_AF0 | MFP_DRIVE_MEDIUM | MFP_PULL_HIGH |
MFP_SLEEP_NONE | MFP_LPM_EDGE_NONE,
    MFP_EOC
};
```

```
static int gpio_lpEnterD2Callback(void) {
	mfp_config((unsigned int*)gpio_key_mfp_cfgs_sleep);
	return TRUE;
}
static int gpio_lpExitD2Callback(BOOL ExitFromD2) {
	mfp_config((unsigned int*)gpio_key_mfp_cfgs_awake);
	return TRUE;
}

uiD2CallbackRegister(LP_MAX_ENTRYS - 1, gpio_lpEnterD2Callback, gpio_lpExitD2Callback);
```

参数 4：

```
#define MFP_DRIVE_VERY_SLOW  MFP(0x0000, 0, 0, 0, 0, 0, 0)
#define MFP_DRIVE_SLOW       MFP(0x0000, 0, 1, 0, 0, 0, 0)
#define MFP_DRIVE_MEDIUM     MFP(0x0000, 0, 2, 0, 0, 0, 0)
#define MFP_DRIVE_FAST       MFP(0x0000, 0, 3, 0, 0, 0, 0)
#define MFP_DRIVE_MASK       MFP(0x0000, 0, 3, 0, 0, 0, 0)
```

这些是 pad 的驱动能力

参数 5：

```
#define MFP_PULL_NONE    MFP(0x0000, 0, 0, 0, 0, 0, 0)
#define MFP_PULL_LOW     MFP(0x0000, 5, 0, 0, 0, 0, 0)
#define MFP_PULL_HIGH    MFP(0x0000, 6, 0, 0, 0, 0, 0)
#define MFP_PULL_BOTH    MFP(0x0000, 7, 0, 0, 0, 0, 0)
#define MFP_PULL_FLOAT   MFP(0x0000, 4, 0, 0, 0, 0, 0)
#define MFP_PULL_MASK    MFP(0x0000, 7, 0, 0, 0, 0, 0)
```

MFP_PULL_NONE 不使能内部的上拉下拉功能
MFP_PULL_LOW 使能下拉
MFP_PULL_HIGH 使能上拉
MFP_PULL_BOTH 使能上下拉
MFP_PULL_FLOAT 浮空功能

```
#define MFP_SLP_DI     MFP(0x0000, 0, 0, 1, 0, 0, 0)
```

这个目前没有用到