## 1. IMX6UL/6ULL**的GPIO操作方法**

CCM: Clock Controller Module (时钟控制模块)

IOMUXC ​: IOMUX Controller，IO复用控制器

GPIO: General-purpose input/output，通用的输入输出口

### 1.1 IMX6UL/6ULL的GPIO模块结构

参考资料：芯片手册《Chapter 26​: General Purpose Input/Output (GPIO)》

1. 有5组GPIO（GPIO1～GPIO5），每组引脚最多有32个，但是可能实际上并没有那么多。

GPIO1有32个引脚：GPIO1\_IO0~GPIO1\_IO31；

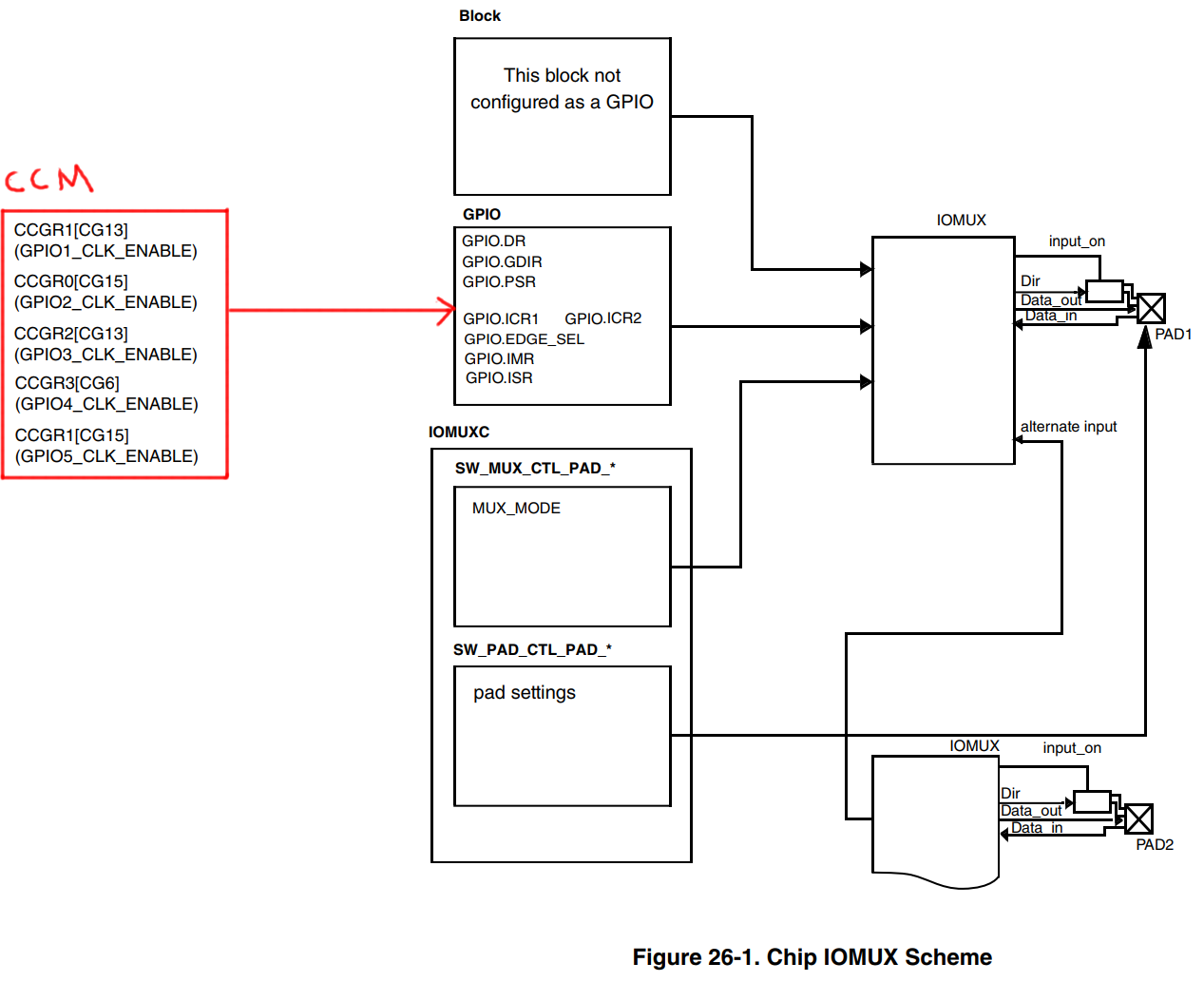
GPIO2有22个引脚：GPIO2\_IO0~GPIO2\_IO21；

GPIO3有29个引脚：GPIO3\_IO0~GPIO3\_IO28；

GPIO4有29个引脚：GPIO4\_IO0~GPIO4\_IO28；

GPIO5有12个引脚：GPIO5\_IO0~GPIO5\_IO11；

1. GPIO的控制涉及4大模块：CCM、IOMUXC、GPIO模块本身

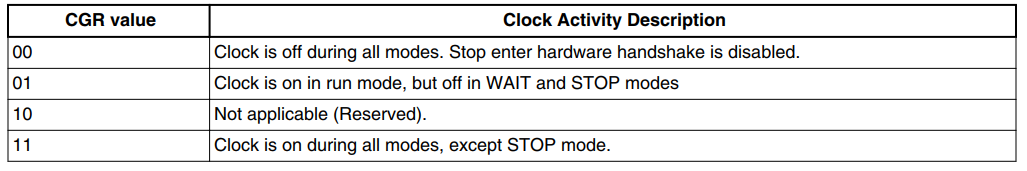


### 1.2 CCM用于设置是否向GPIO模块提供时钟

参考资料：芯片手册《Chapter 18​: Clock Controller Module (CCM)》

GPIOx要用CCM\_CCGRy寄存器中的2位来决定该组GPIO是否使能。哪组GPIO用哪个CCM\_CCGR寄存器来设置，请看上图红框部分。

CCM\_CCGR寄存器中某2位的取值含义如下：



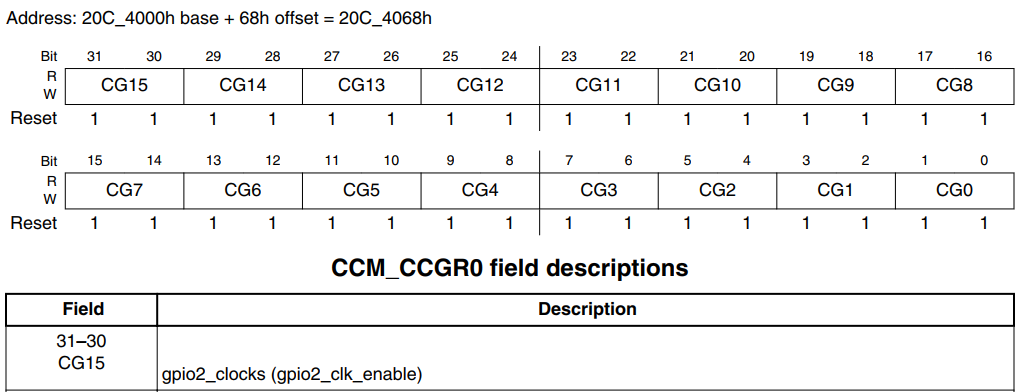
① 00：该GPIO模块全程被关闭

② 01：该GPIO模块在CPU run mode情况下是使能的；在WAIT或STOP模式下，关闭

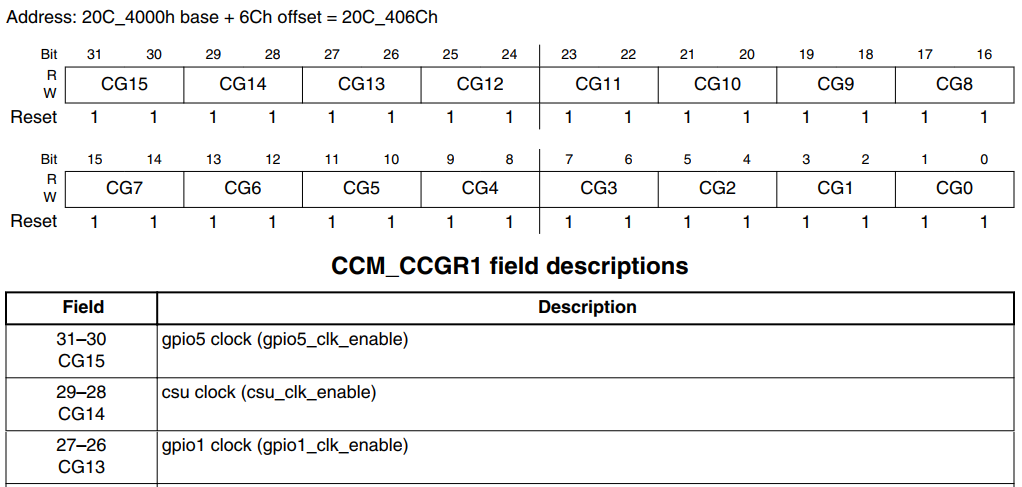
③ 10：保留

④ 11：该GPIO模块全程使能

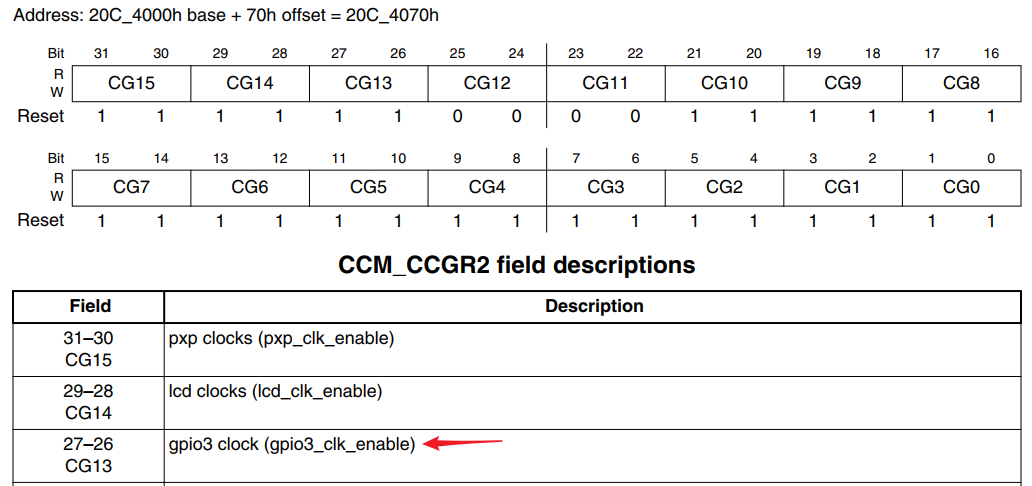
GPIO2时钟控制：



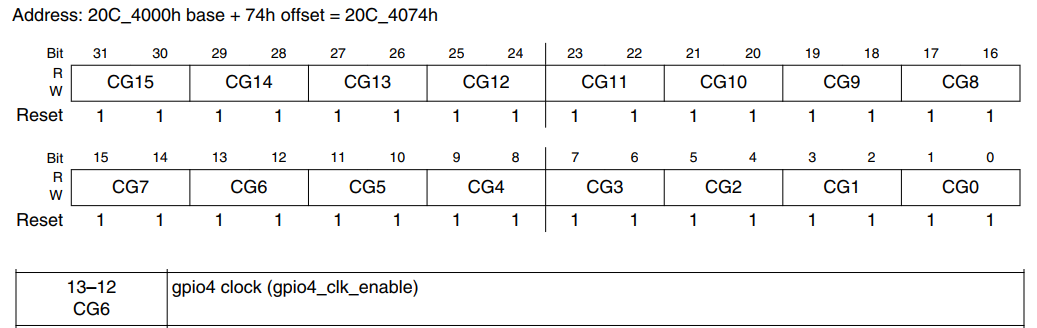
GPIO1、GPIO5时钟控制：



GPIO3时钟控制：

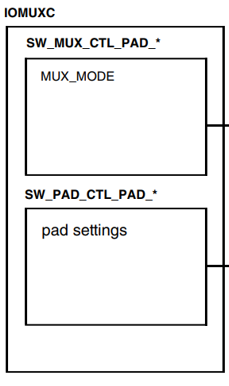


GPIO4时钟控制：



### 1.3 IOMUXC：引脚的模式(Mode、功能)

参考资料：芯片手册《Chapter 32​: IOMUX Controller (IOMUXC)》。



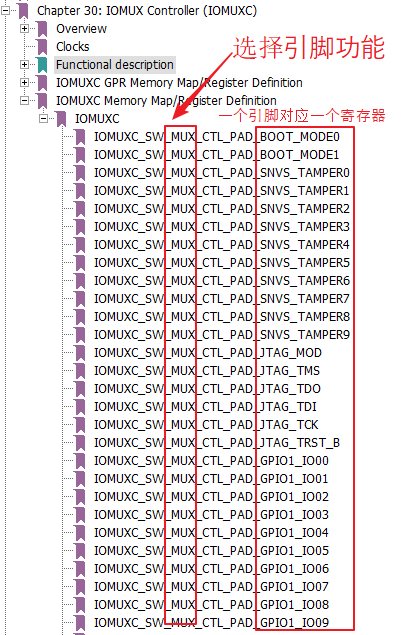
对于某个/某组引脚，IOMUXC中有2个寄存器用来设置它：

① 选择功能：

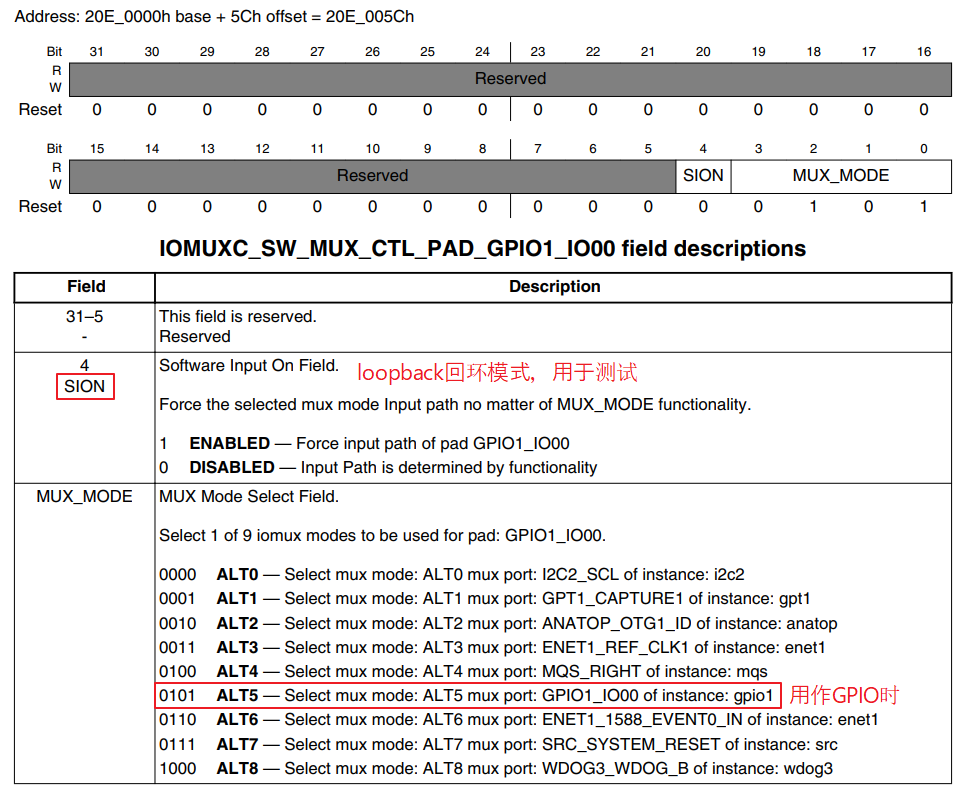
IOMUXC\_SW\_MUX\_CTL\_PAD\_<PADNAME> ：Mux pad xxx，选择某个pad的功能

IOMUXC\_SW\_MUX\_CTL\_GRP\_<GROUP NAME>：Mux grp xxx，选择某组引脚的功能

某个引脚，或是某组预设的引脚，都有8个可选的模式(alternate (ALT) MUX\_MODE)，



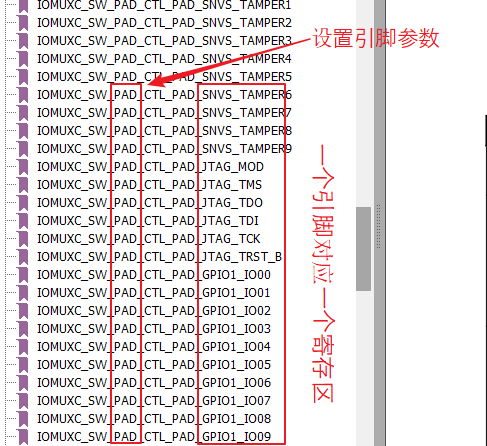
比如：



② 设置上下拉电阻等参数：

IOMUXC\_SW\_PAD\_CTL\_PAD\_<PAD\_NAME>：pad pad xxx，设置某个pad的参数

IOMUXC\_SW\_PAD\_CTL\_GRP\_<GROUP NAME>：pad grp xxx，设置某组引脚的参数

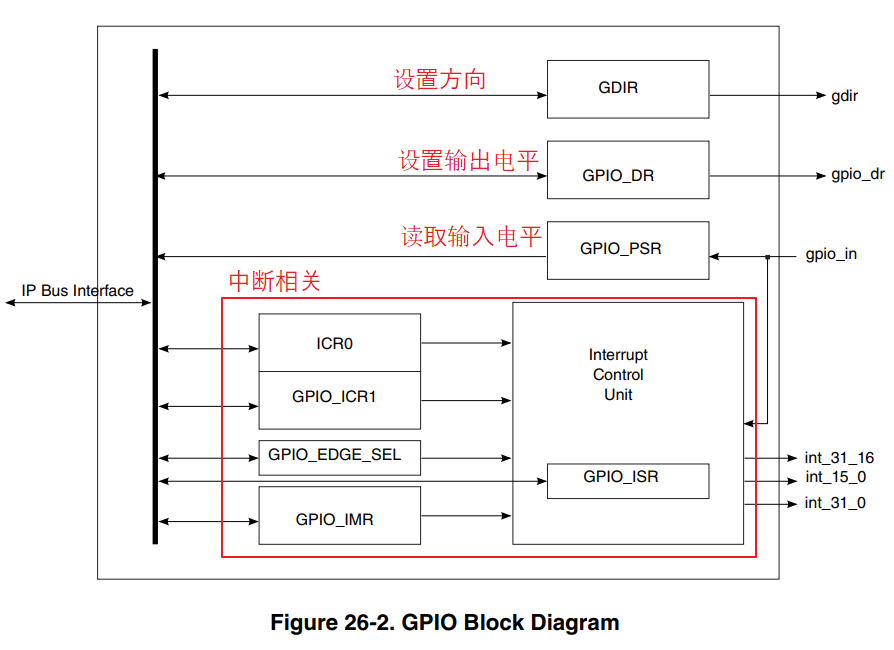


比如：



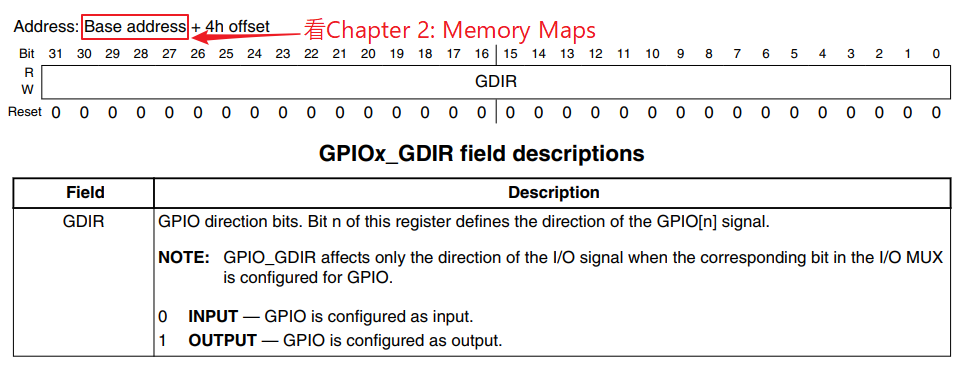
### 1.4 GPIO模块内部

框图如下：

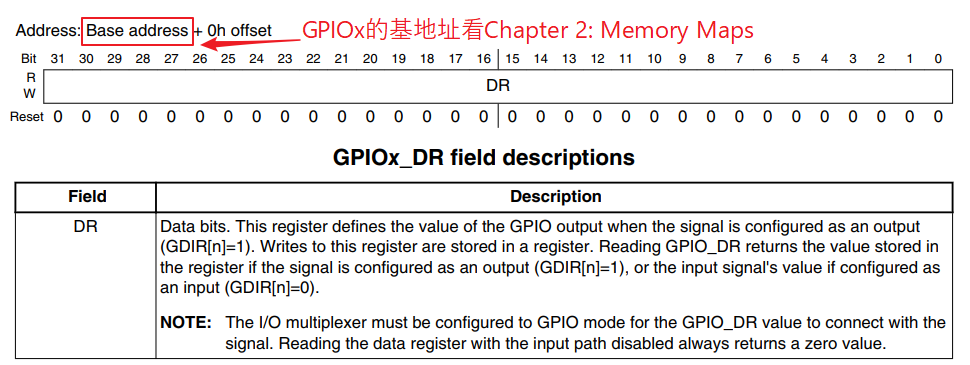


我们暂时只需要关心3个寄存器：

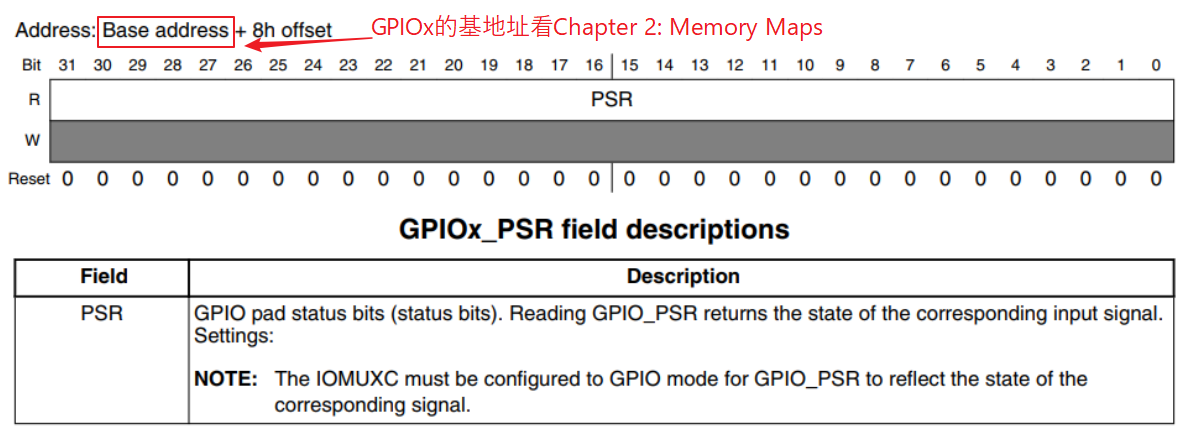
① GPIOx\_GDIR：设置引脚方向，每位对应一个引脚，1-output，0-input



② GPIOx\_GDIR：设置输出引脚的电平，每位对应一个引脚，1-高电平，0-低电平

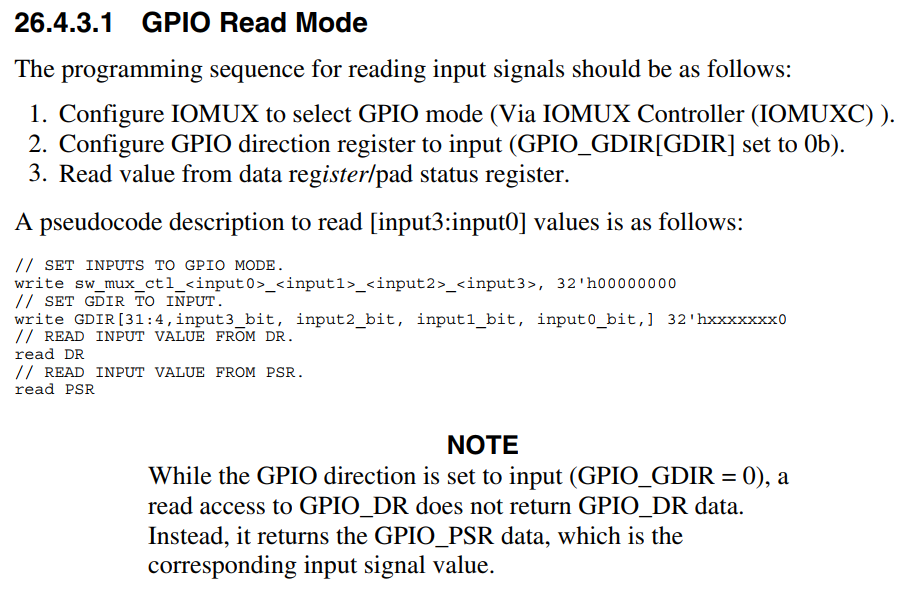


③ GPIOx\_PSR：读取引脚的电平，每位对应一个引脚，1-高电平，0-低电平



### 1.5 怎么编程

#### 1.5.1 读GPIO



翻译一下：

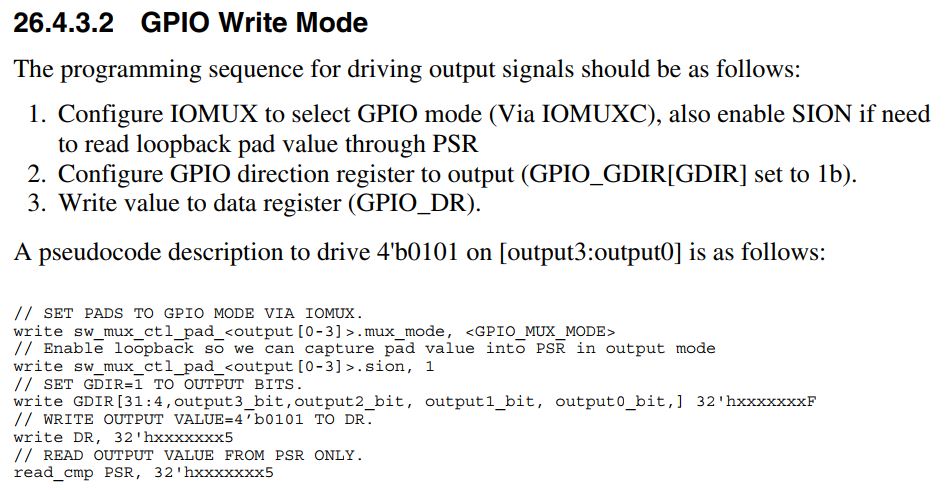
① 设置CCM\_CCGRx寄存器中某位使能对应的GPIO模块 // 默认是使能的，上图省略了

② 设置IOMUX来选择引脚用于GPIO

③ 设置GPIOx\_GDIR中某位为0，把该引脚设置为输入功能

④ 读GPIOx\_DR或GPIOx\_PSR得到某位的值（读GPIOx\_DR返回的是GPIOx\_PSR的值）

#### 1.5.2 写GPIO



翻译一下：

① 设置CCM\_CCGRx寄存器中某位使能对应的GPIO模块 // 默认是使能的，上图省略了

② 设置IOMUX来选择引脚用于GPIO

③ 设置GPIOx\_GDIR中某位为1，把该引脚设置为输出功能

④ 写GPIOx\_DR某位的值

需要注意的是，你可以设置该引脚的loopback功能，这样就可以从GPIOx\_PSR中读到引脚的有实电平；你从GPIOx\_DR中读回的只是上次设置的值，它并不能反应引脚的真实电平，比如可能因为硬件故障导致该引脚跟地短路了，你通过设置GPIOx\_DR让它输出高电平并不会起效果。