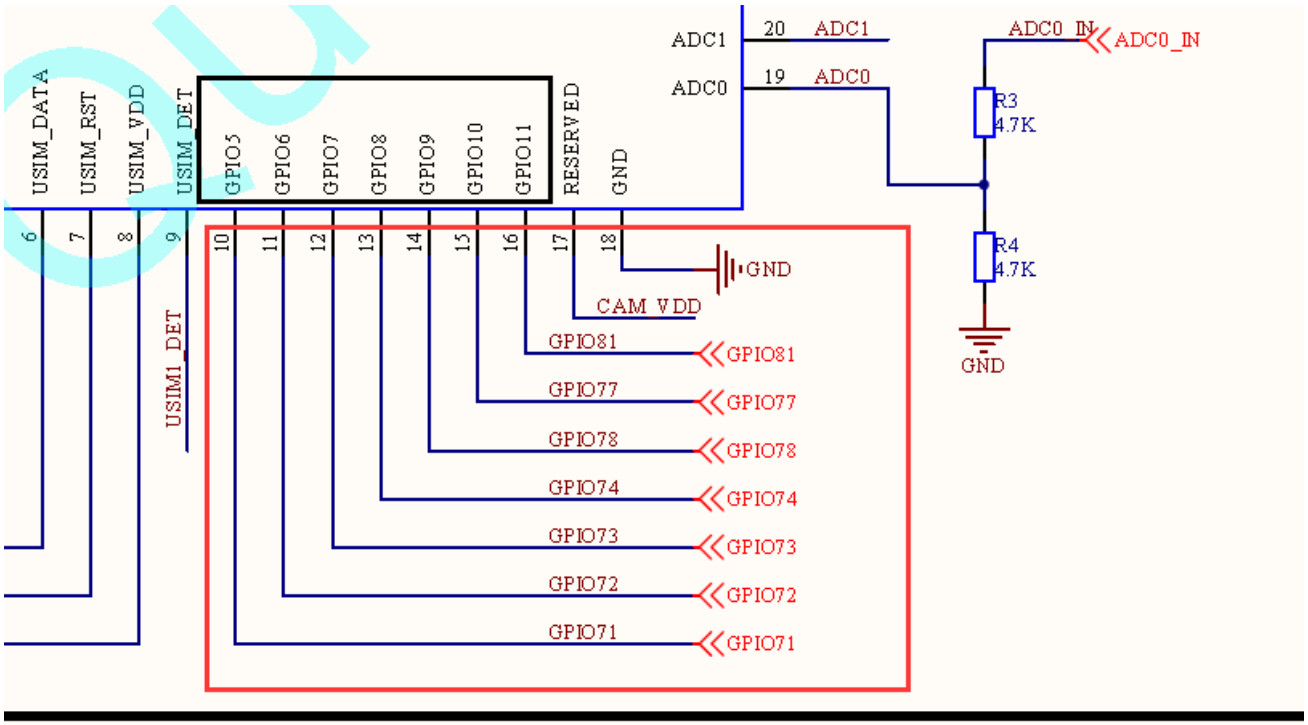
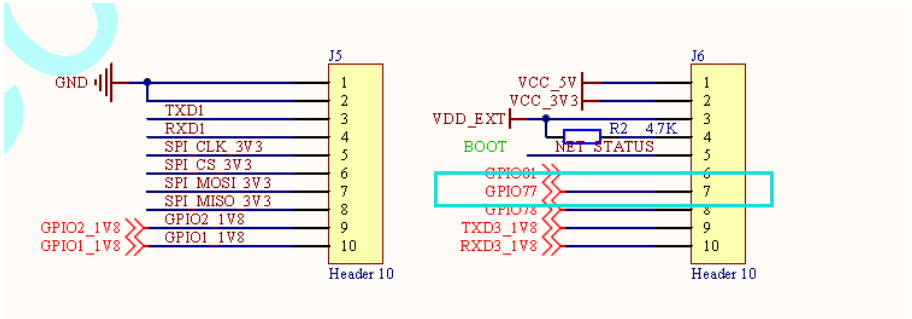
对照原理图，找开发板上是否引出了这个 GPIO 口

第一步，看这个 GPIO 与 PIN 脚的对应关系（位置在 API 库的 PIN 模块，左侧是 Quecpython 的 GPIO 命名，右侧是模组的 PIN 脚号），例如你想使用 QuecPython 的 GPIO6，那么看了这个图，你知道的是 GPIO6 对应的是模组的 PIN15；

		GPIO19 – 引脚号215 EC600SCN平台引脚对应关系如下（引脚号为模块外部引脚编号）： GPIO1 – 引脚号10 GPIO2 – 引脚号11 GPIO3 – 引脚号12 GPIO4 – 引脚号13 GPIO5 – 引脚号14 GPIO6 – 引脚号15 GPIO7 – 引脚号16 GPIO8 – 引脚号39 GPIO9 – 引脚号40 GPIO10 – 引脚号48 GPIO11 – 引脚号58 GPIO12 – 引脚号59 GPIO13 – 引脚号60 GPIO14 – 引脚号61
direction	int	IN – 输入模式，OUT – 输出模式
pullMode	int	PULL_DISABLE – 浮空模式 PULL_PU – 上拉模式

第二步看，EC600SV1.1 的原理图，（黑色部分无需关注，只需要关注红色圈的部分），对于刚刚说的 Quecpython 的 GPIO6（PIN15）,这里关注的是 PIN15 对应的 GPIO77（实际上 GPIO77 没有特别的含义，只是表示一个连接关系，也就是此处的 GPIO77 对应于下图中的 GPIO77）。





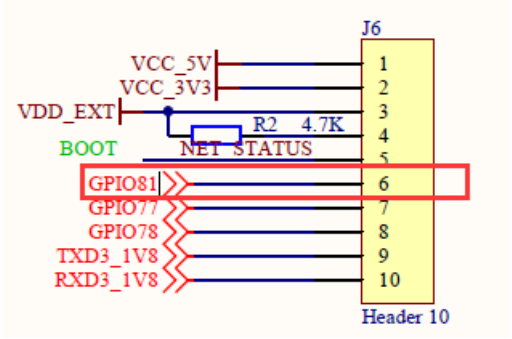
第三步，上方最后一个图的 GPIO77，也就对应开发板引出的 GPIO77(开发板的 J6 处)。

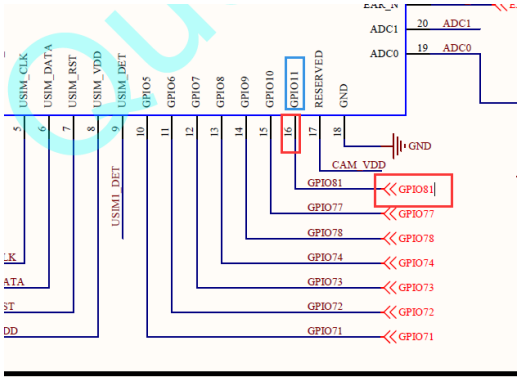
总结：配置 QuecPython 的 GPIO6，在开发板的 GPIO77 处查看配置是否生效。

对照开发板上的丝印，找具体是哪个 GPIO 口

例如我想找开发板上的 G81 对应 QuecPython 的哪个 GPIO 口（如果按照下面方法没有找到对应的 GPIO，那说明暂时还未开放）。

第一步，看原理图，开发板的 G81 对应原理图就是 GPIO81。如下图所示，GPIO81 对应的是 PIN16（请忽略蓝色框的 GPIO 标识，记住 PIN16）





第一步，看原理图，开发板的 G81 对应原理图就是 GPIO81。如下图所示，GPIO81 对应的是 PIN16**（请忽略蓝色框的 GPIO 标识，记住 PIN16）**

第二步，进入 QuecPython 的官网，找到 PIN 的 API 库（https://python.quectel.com/wiki/#/zh-cn/api/QuecPythonClasslib?id=pin）
如下图所示，**如果你要控制 G81，就需要配置 GPIO7。**

GPIOn	int	GPIO4 – 引脚号53 GPIO5 – 引脚号54 GPIO6 – 引脚号104 GPIO7 – 引脚号105 GPIO8 – 引脚号106 GPIO9 – 引脚号107 GPIO10 – 引脚号178 GPIO11 – 引脚号195 GPIO12 – 引脚号196 GPIO13 – 引脚号197 GPIO14 – 引脚号198 GPIO15 – 引脚号199 GPIO16 – 引脚号203 GPIO17 – 引脚号204 GPIO18 – 引脚号214 GPIO19 – 引脚号215 EC600SCN平台引脚对应关系如下（引脚号为模块外部引脚编号）： GPIO1 – 引脚号10 GPIO2 – 引脚号11 GPIO3 – 引脚号12 GPIO4 – 引脚号13 GPIO5 – 引脚号14 GPIO6 – 引脚号15 GPIO7 – 引脚号16 GPIO8 – 引脚号39 GPIO9 – 引脚号40 GPIO10 – 引脚号48 GPIO11 – 引脚号58 GPIO12 – 引脚号59 GPIO13 – 引脚号60 GPIO14 – 引脚号61
direction	int	IN – 输入模式，OUT – 输出模式
		PULL_DISABLE – 浮空模式