NANO低功耗 调试方法

Application Note for 32-bit NuMicro™ Family

Document Information

Abstract	该文档介绍nano中，怎样调进入超低功耗模式时的功耗.
Apply to	Nano Series.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

目录

1 介绍

Nano系列是以Cotex-m0为内核的超低功耗MCU。进入深度睡眠模式时，在所有IP都关闭的情况下，功耗为1uA；在RTC和段式LCD屏都工作的情况下，功耗为10uA左右。客户的系统板上往往不止一颗IC，如何让系统进入深度睡眠模式时功耗最低？一直都是客户最为关心的问题，也是出现问题最多的地方。本文将介绍一些设计原则和调试方法，希望对大家有所帮助。

系统进入低功耗只需要4行代码:

```
SYS_UnlockReg();/*解锁寄存器*/
CLK->PWRCTL |= (CLK_PWRCTL_PWRDOWN_EN | CLK_PWRCTL_DELY_EN);        /* Set power down bit */
SCB->SCR |= 0x04;                                                   /* Sleep Deep */
__WFI();
```

WFI指令执行之后，MCU就进入低功耗了。

由于客户系统板上还存在其它的IC，与这些IC相连的IO管脚要小心处理，防止这些IO脚漏电。下面主要介绍如何防止这些IO口漏电。相关寄存器将会在1.1节提及，希望大家牢记。

1.1 寄存器

相关寄存器包括：

- PWRCTL、AHBCLK 和 APBCLK 寄存器。
  - PWRCTL：时钟控制寄存器，用于使能/关闭时钟。进入 power down 时查看内部 10K(LIRC)和外部 32K(LXT)时钟是否关闭，如果不需要使用就关闭。这两个时钟硬件不会主动关闭，需要通过软件关闭。
  - AHBCLK 和 APBCLK：时钟使能寄存器，用于使能各个外设的时钟。进入 power down 时查看时钟开启的外设，掉电模式下不需要工作的 IP，应关闭其对应的时钟。这里需要通过软件关闭的只有模拟外设的时钟，或者是以 10K 和 32K 为时钟源的外设。
- RegLockAddr 寄存器
  - 进入 power down 之前要保证 RegUnLock = 1，就是寄存器解锁。
- GPIO 中的 PMD（模式寄存器），OFFD（关闭数字通路寄存器），PUEN（使能内部上拉）寄存器。
- 系统中的 IO 功能设定寄存器 MFP
- CONFIG0 寄存器

进入低功耗时，PWRCTL、AHBCLK、APBCLK、RegLockAddr，以及 CONFIG0 和模拟外设只需要注意一下就行了，查看一下应该关闭的都关闭了就行。

需要花费比较多时间的是 GPIO 寄存器和 MFP 寄存器的设定。在 1.2 节原则部分会详细介绍 GPIO 应该怎样设定。

下面是拿到板子，需要做低功耗，或者进入低功耗发现功耗不正常时一般的检查步骤

1.2 原则

第一步：查是否进入低功耗

- 进入低功耗之前注意需要先解锁
- 使用 Nu-Link 下载/调试/烧录 m0 之后，m0 会进入 debug 模式，在该模式下 m0 无法进入 power down。只有将 m0 断电，m0 才能退出 debug 模式。所以烧录之后要先断电再上电后才能看到 power down 的功耗。
- 不能在中断处理函数中进入 power down 模式
- 进入 power down 时 NANO100 系列 LDO 引脚的电压将会从 1.8V 变成 1.6V。
  - 如果 LDO 电压变成 1.6V 才说明 NANO 进入了低功耗模式。
  - 如果量到电压为 1.8V，则表示芯片烧录之后没有断电，或者是进入低功耗的函数设置错误，或者是进入了低功耗但是系统又被唤醒了，又或者是 RegLockAddr 寄存器没有解锁。
- 进入 power down 时 NANO1x2 系列 LDO 引脚的电压有 1.8V 和 1.6V 两种选择。
  - LDO_CTL 寄存器可以控制进入 power down 时 LDO 引脚的电压。
- 检查 CONFIG0 寄存器的设定
  - 特别是 CFOSC 栏位(CPU clock source select after reset)的值，防止没接外部晶振的情况下使能了外部晶振。
  - 另外查看 CBS（boot select）栏位，确保程序可以启动起来。经常会发生程序烧录在 APROM 但是选择的是从 LDROM 启动，而 LDROM 里面又没有程序可以跳到 APROM 里面，所以导致 APROM 里面的程序实际上根本没有被执行到。这里大家经常遇到的问题是在 keil debug 模式下程序运行正常，上电直接跑反而跑不起来，这是因为 keil 发现你的程序烧录在 APROM 里，会负责跳到 APROM 执行。

第二步：查引脚设定

- 没有引出的引脚，保持 GPIO，INPUT 模式，打开内部上拉(PUEN)。例如：NANO130RxxBN 是 64pin 封装的，PC.4 脚没有出来，PC->PUEN bit[4]要写‘1’，并切为 INPUT 模式。这里要特别注意：没有引出的脚只能设为 INPUT 并打开内部上拉，不能设为 OUTPUT 模式并输出低电平，切记！！
- 悬空的引脚，保持 GPIO，INPUT 模式，打开内部上拉(PUEN)
- 深度睡眠时不需要保持状态的引脚，统统设为 GPIO，OUTPUT 模式，并输出低电平。例如：SPI0 4 根引脚外接一个 IC，系统 power down 时该 IC 是断电的，则 SPI0 的 4 根脚就需要切为 GPIO 模式，并输出 0。

- 状态需要保持的引脚（保持输出低电平/高电平，或者保持输入低电平/高电平），切成 GPIO 功能，模式维持不变，同时不要使能内部上拉。
  - 需要片外 IC 通过 GPIO 唤醒的，该 GPIO 引脚保持输入模式不要动，也不要打开内部上拉(PUEN)。
  - 输入的引脚要特别小心，确认输入的电压是 0 或者 VDD，不能是中间的其它值，否则会漏电。如果切成输入引脚的输入电压不确定，则需关闭数字通路(OFFD)
  - 输出高电平的引脚也要特别小心，如果该引脚接到其它芯片上，要防止灌电到其它芯片。如果发现灌电，该脚就不能输出高
- 关闭片外 IC 的电源或者让其进入 power down。与片外 IC 相连的 MCU 引脚处理原则如下
  - 如果片外 IC 的电源是关闭的，需要把与之相连的引脚都切成 GPIO 功能，并输出 0
- 晶振引脚如果外接晶振了，在 power down 的时候保持配置为晶振模式，并且不能打开内部上拉；但是如果外部没有接晶振，悬空或者用作普通 GPIO 使用，一定不能使能晶振（PWRCON），并且 HXT_GAIN 和 HXT_SELXT 要清为 0，否则会漏电。然后输出 0 即可
- NANO100AN 的 ICE_DAT 和 ICE_CLK 脚进入 power down 之前需要切为 GPIO 功能，INPUT 模式，并打开内部上拉。唤醒之后再切为 ICE_DAT 和 ICE_CLK 功能。

第三步：查LCD耗电

- 外接的段式 LCD 屏，power down 时需要保持显示的话，LCD 引脚需要保持 LCD 模式，并关闭相应引脚的数字通路。但是如果 power down 时不需要屏继续显示，相应 COM/SEG/DH1/DH2/ LCD_V1/LCD_V2/LCD_V3 引脚需要设为 GPIO 功能，INPUT 模式，并关闭相应引脚的数字通路。
  - 段式屏如果使用 C-Type 模式，功耗会比 R-Type 模式高大约 5～6uA（这个值跟 R-type 阶梯电阻，C-Type 充电频率有关）
    - C-Type 的好处是当电池没电时，可以将电压稳住，屏显示不会模糊，但是比较耗电
    - R-Type 芯片内部带 200K 欧姆的阶梯电阻，如果屏允许更小的驱动电流，可以外接 1M 的电阻来减小漏电。
    - 在屏允许的范围内将 LCD 工作频率尽量调低（LCD_CTL 寄存器的 FREQ）
    - NANO1x2 有 Ext-C 模式，比 R-Type 还省电

第四步：查模拟IP设定

- 模拟外设需要通过软件关闭相应功能。例如：ADC 需要将 ADCR 寄存器的 ADEN 清 0，并且关闭数字通路(OFFD)，无需切为 GPIO 模式。

第五步：查晶振设定

- 内部 10K 和外部 32K，power down 时不需要的话，应通过软件将其关闭。

第六步：查原理图

- 检查板子上所有接地的部分，是否有可能从电源到地形成通路。如果必须形成通路，尽量把电阻调大一点，这样漏电会小一些。

注：请在 WFI 指令之前设定断点，查看寄存器的设定是否如预期，涉及的寄存器如 1.1 节所述。

1.3 NANO 的行为

进入power down的时候，MCU会主动关闭HIRC（内部高频振荡器）和HXT（外部高频晶振）两个高频时钟，软件不要去控制这两个晶振。外部32K和内部10K需要软件自己关闭。

这就导致使用HIRC和HXT为时钟源的外设在进入power down的时候会自动停止工作，但是使用外部32K或者内部10K作为时钟源的外设需由软件决定在系统进入power down的时候是否仍然工作。例如：系统进入power down的时候，RTC和WDT仍然工作，这就需要不能关闭32K和10K晶振。

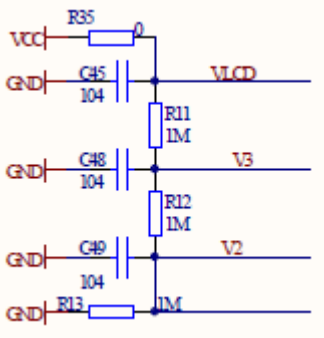
但是模拟外设是个特例，因为模拟电路工作不需要时钟。所以如果模拟外设功能使能，MCU虽然将晶振关闭，但是模拟电路部分仍会耗电。例如：ADC，系统进入power down之前需要写控制寄存器ADCR->ADEN将ADC功能关闭，否则会漏电。常见的模拟外设有：ADC、ACMP、LCD、DAC等。

进入power down的时候如果引脚设定有问题，NANO中上拉电阻造成每个IO脚的漏电大概20uA，这也是一个线索。

1.4 漏电电路

下面是一些常见的漏电电路，给大家参考。

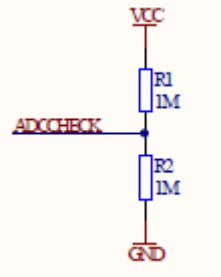
1.4.1 电路一



该电路从VCC->R35->R11->R12->R13形成一个通路。假设VCC=3.3V，这里漏电3.3V/3MΩ = 1.1uA。

这个是LCD的驱动电路，考虑到驱动能力，加1M可能已经差不多了。大家明白这里会耗电1.1uA就OK了。

1.4.2 电路二



该电路从VCC->R1->R2形成通路，假设VCC=3.3V，这里漏电3.3V/2MΩ = 1.65uA。

该电路是测量电源电压的电路，ADCCHECK引脚接到ADC0模拟输入引脚，power down的时候需要关闭数字通路(OFFD)。漏电小一点的解决办法：

a) R1 和 R2 尽量用大一点的电阻

b) 不用上面的电路，而是将 Avdd 接到 Vdd 做 ADC 参考，使用内部参考电压或者 Bangap 反推 Avdd 的方法。详细设定方法如下

i. NANO100 系列 ADC 通道 15 可以将内部参考电压作为输入

[15]	CHEN15	Analog Input Channel 15 Enable (Convert Int_VREF) 1 = Enabled. 0 = Disabled.

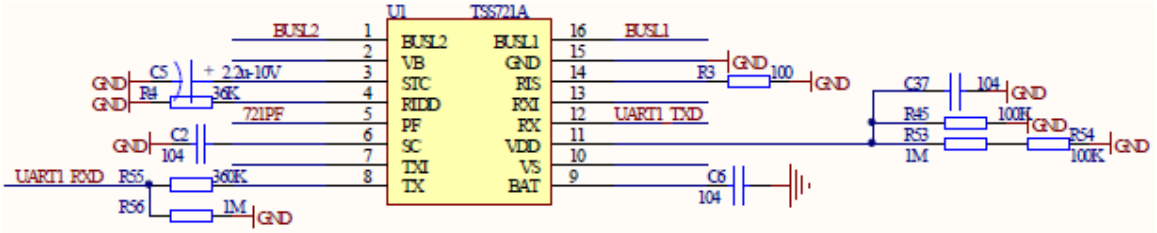
ii. 设定下面栏位为 00b 选择 Avdd 做参考

[17:16]	REFSEL	Reference Voltage Source Selection	
		00	Select power as reference voltage
		01	Select Int_VREF as reference voltage
		10	Select VREF as reference voltage

iii. Int_VREFCTL 寄存器可以设定使能内部参考电压，甚至内部参考的电压还可以选择，例如 2.5V 或者 1.8V

iv. ADC 采样值公式为： Vref * 采样值/4096 = 输入电压(Int_VREF) 。 此时 Int_VREF 电压已知，则可以倒推 Vref 的值

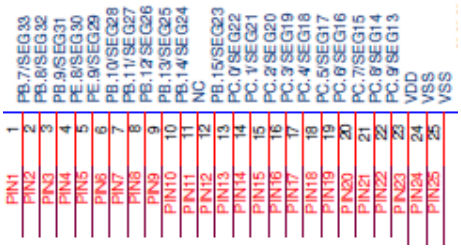
1.4.3 电路三



该电路从UART1_RXD->R56形成通路，假设VCC=3.3V，这里漏电3.3V/1MΩ = 3.3uA。

解决办法：power down时，将UART1_RXD引脚设为INPUT模式，但是不要打开内部上拉。

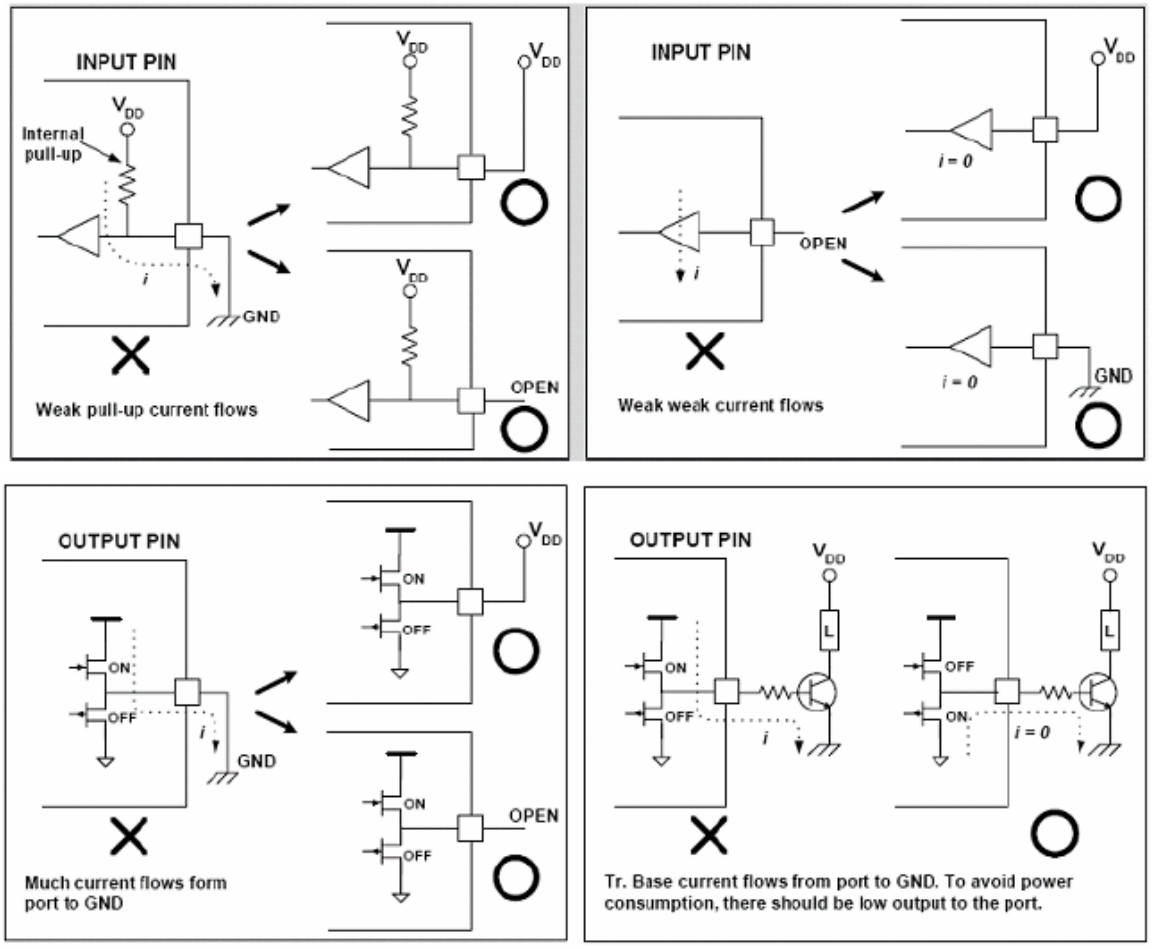
1.4.4 电路四



上图是NANO112 100pin的引脚图，PIN9是UART0_RX， PIN10是UART0_TX。常常有客户拿它们打印debug信息用。这就涉及一个问题，当不连接牛桥的时候，这UART0_RX脚就是悬空的，会漏电。

UART RXD 有可能floating的话，设成INPUT，打开内部上拉。这样不管有无外接UART线，都不会漏电。

1.4.5 电路五



上面这4副图详细介绍了GPIO输入模式，输出模式下外部正确的接法。

- ✧ 图 1 是 INPUT 的引脚，打开了内部上拉，同时外部接地了，这样是不对的。正确的应该像箭头所指，外部也上拉或者什么都不要接
- ✧ 图 2 是 INPUT 的引脚，没有打开内部上拉，外部也是悬空的，这样是不对的。正确的应该像箭头所指，接上拉或者下拉
- ✧ 图 3 是 OUTPUT 的引脚，输出高电平的时候，外部接地了，这样是不对的。正确的应该像箭头所指，外部上拉或者什么都不要接
- ✧ 图 4 是 OUTPUT 的引脚，输出高电平的时候，外部接 P-MOS，这样 Vdd 到地形成通路。像右边所指，输出低电平时就不会漏电

1.5 常见问题

下面这几种问题比较常见，大家按照上面的原则每一步都做了，可能功耗还是不对。这时候可以参考一下客户遇到的下面几种常见问题。

- 案例 1：进入 power down 时功耗高达 5.8mA

- 案例 2：打开 GPIO 的内部上拉，功耗反而变高
- 案例 3：打开内部上拉，功耗降低，但是耗电仍然偏高
- 案例 4：功耗会飘，手靠近板子，功耗会忽高忽低
- 案例 5：Power down 时，LCD 和 RTC 保持显示，正常情况下功耗应该是 10uA 左右，但是测得功耗 16uA
- 案例 6：大多情况下进入 power down 的功耗都是正常的 6uA 左右，但是某次进入 power down 时的功耗会高达 1mA
- 案例 7：寄存器设定都正常，但是功耗还是不对

1.6 解决方法

1.6.1 进入 power down 时功耗高达 5.8mA

客户板子上其它 IC 全部拿掉，只留新塘 MCU，但是进入低功耗是功耗仍然高达 5.8mA，和 TRM 上的 1uA 相差甚远。查看客户代码如下：

```
void PowerDownFunction(void)
{
    /* To check if all the debug messages are finished */
   UART_WAIT_TX_EMPTY(UART0);
    SCB->SCR = SCB_SCR_SLEEPDEEP_Msk;
    CLK->PWRCTL |= (CLK_PWRCTL_PD_EN_Msk | CLK_PWRCTL_WK_DLY_Msk );
    __WFI();
}
```

客户代码没有解锁寄存器，加上SYS_UnlockReg(); 之后就可以量到正确的电流。

```
void PowerDownFunction(void)
{
    /* To check if all the debug messages are finished */
   UART_WAIT_TX_EMPTY(UART0);
   SYS_UnlockReg();
    SCB->SCR = SCB_SCR_SLEEPDEEP_Msk;
    CLK->PWRCTL |= (CLK_PWRCTL_PD_EN_Msk | CLK_PWRCTL_WK_DLY_Msk );
    __WFI();
}
```

1.6.2 打开 IO 口的内部上拉，功耗反而变高

说明引脚有下拉，这个下拉可能是内部也可能是外部。该引脚输出低电平时芯片内部是接地的，此时打开内部上拉，就会导致漏电。外部下拉可能是电路板本身有下拉或者被外部芯片下拉。

外部有下拉不一定是明显的引脚加电阻接地，有可能是外部芯片在输出低。

- 如果是 MCU 在输出低，power down 的时候保持输出低就行，就别打开内部上拉了。
- 如果是外部芯片拉低的，该引脚就切成 GPIO 功能，INPUT 模式，但是不要打开内部上拉。

1.6.3 打开内部上拉，功耗降低，但是耗电仍然偏高

这就要考虑两种情况

- 是否 OFFD 没有关闭
- 是否模拟 IP，例如：ADC 没有关

OFFD 没有关，就会导致漏电
ADC 没有关，也会导致漏电
一根一根引脚把它们找出来，关闭就行了。

1.6.4 功耗会飘，手靠近板子，功耗会忽高忽低

这个问题的原因比较多，列出可能的3个原因

1. **可能是晶振引脚和 ICE 引脚设定不正确**

   进入power down的时候

   - 晶振引脚要维持晶振功能，并且不要打开内部上拉。如果没有接晶振的话，当普通 GPIO 脚处理
   - NANO0100AN 的 ICE_DAT 和 ICE_CLK 引脚需要设为 GPIO 功能，INPUT 模式，并打开内部上拉

   晶振引脚的处理，

   - 如果不接晶振并且 floating 时，需要切为 GPIO 模式，并输出低电平。NANOAN 需要保持晶振模式，不要打开上拉。
   - 如果晶振引脚需要作为 GPIO 使用，power down 时，GPIO 功能不再需要的，需要切 GPIO 功能，并输出低电平。

- 如果晶振引脚需要作为 GPIO 使用，power down 时，GPIO 功能需要保持的，可以保持 GPIO。

2. **可能是晶振引脚没有接晶振但是使能了晶振(PWRCON)**
   - 解决办法就是将晶振使能关闭，除了检查软件外还要检查 CONFIG0 寄存器，是否上电默认使能了
3. **可能是设为 INPUT 模式的引脚输入的电压是介于 VSS 和 VDD 之间的某个电压，导致芯片内部的 MOS 漏电**
   - 解决办法就是将所有 IO 的数字通路都关闭：OFFD = 0xFFFF0000;
   - 如果电流不再飘（功耗有可能会变高），说明就是有引脚输入了非预想电压
   - 然后依次使能一些引脚的数字通路，最后就可以定位是哪些引脚需要关闭数字通路（OFFD）

1.6.5 Power down 时，LCD 和 RTC 保持显示，正常情况下功耗应该是 10uA 左右，但是测得功耗 16uA

如果芯片内部IO口漏电，每根IO漏电20uA左右。而该板子多耗电6uA，所以一般不是芯片内部漏电，这就需要查整个板子的电路，是否有电源到地的通路。1.4节列出了几种漏电的情况，大家可以参考看看。

1.6.6 大多情况下进入 power down 的功耗都是正常的 6uA 左右，但是某次进入 power down 时的功耗会高达 1mA

先确定外部IC确实都断电，然后将用到的IP一个一个关闭。发现关闭ADC功能之后这个问题不再重现，所以锁定ADC。

ADC工作频率比CPU低的情况下，写到ADC寄存器中的数据要等2个ADC工作时钟才会起作用，所以在进入power down时，关闭ADC时钟之前（APBCLK），需要延迟2个ADC工作时钟。不然ADC可能还没有关闭，时钟没有了，就会漏电。

1. 6. 7 寄存器设定都对，但是功耗还是不对

检查引脚上是不是外接了其它仪器，查功耗的的时候，只能串电流表进来，其它的测量仪器都拿掉。

1.7 调试方法

下面详细为大家讲解一下低功耗软件的调试方法。以超声波热表为例，热表power down时整机功耗需要<10uA。2个按键和UART收到数据后唤醒。支持红外、RS485和MBUS抄表。外接一颗超声波芯片。

拿到板子之后，在不确定外围IC进入低功耗的方法时，或者该做的都做了，功耗就是降不到<10uA时，先把外围全部拆掉是比较好的方法。

1.7.1 确认 GPIO 的设定

将整个芯片的每个GPIO用途全部列成表格是一个比较好的习惯，这样可以防止出错。看似整理表格花了一些时间，但是这对后面功耗不正常时仔细检查GPIO设定很有帮助。

例如：下图是NANO112某个热表项目整理的引脚使用表

引脚	功能设定	状态	引脚	功能设定	状态	引脚	功能设定
PA.0	IN IO	OPENDRAIN	PB.0	CKO1		PC.0	SEG18
PA.1	not use		PB.1	EINT1		PC.1	SEG17
PA.2	EINT0		PB.2	GP22 RST	OUTOUT0-1	PC.2	SEG16
PA.3	not use		PB.3	Enable	OUTPUT 1	PC.3	SEG15
PA.4	not use		PB.4	not use		PC.4	SEG14
PA.5	not use		PB.5	not use		PC.5	SEG13
PA.6	not use		PB.6	not use		PC.6	SEG12
PA.7	not bond		PB.7	not bond		PC.7	SEG11
PA.8	not bond		PB.8	not bond		PC.8	SEG10
PA.9	not bond		PB.9	not bond		PC.9	SEG9
PA.10	not bond		PB.10	En_Start	INPUT	PC.10	not bond
PA.11	not bond		PB.11	En_Stop	OUTPUT	PC.11	not bond
PA.12	SPI1_MOSI		PB.12	GPIO_INPUT	FIRE	PC.12	not bond
PA.13	SPI1_MISO		PB.13	UART0_RX		PC.13	not bond
PA.14	SPI1_SCLK		PB.14	UART0_TX		PC.14	SEG8
PA.15	SPI1_SS		PB.15	SEG19		PC.15	SEG7
引脚	**功能设定**	**状态**	**引脚**	**功能设定**	**状态**	**引脚**	**功能设定**
PD.0	SEG6		PE.0	not bond		PF.0	X32I
PD.1	SEG5		PE.1	not bond		PF.1	X32O
PD.2	SEG4		PE.2	not bond		PF.2/XT1_IN	UART1_RX
PD.3	SEG3		PE.3	not bond		PF.3/XT1_OUT	UART1_TX
PD.4	SEG2		PE.4	not bond		PF.4	ICE_CLK
PD.5	SEG1		PE.5	not bond		PF.5	ICE_DAT
PD.6	SEG0		PE.6	not bond			
PD.7	COM0		PE.7	not bond			
PD.8	COM1		PE.8	not bond			
PD.9	COM2		PE.9	not bond			
PD.10	COM3						
PD.11/DH2	DH2						
PD.12/DH1	DH1						
PD.13	V1						
PD.14	V2						
PD.15	V3						

Power down时LCD保持，红外，UART0保持，UART1因为外接到一颗芯片上，而系统power down时该芯片是断电的，所以切GPIO防止漏电。SPI接口切GPIO。所以从上表可以看到PD、PE都不用动，PC只有没有绑定的几个脚需要打开上拉。需要仔细确认状态的就剩PA和PB和PF

- ✧ PA.0 是输入，为 open drain 用外部上拉，power down 时保持
- ✧ PA.2 是输入按键，使能内部上拉，power down 时保持
- ✧ PA.3 ~ PA11 保持 GPIO 模式，打开内部上拉
- ✧ PA.12 ~ PA.15 外接超声波芯片，系统 power down 时该芯片断电，所以全部切 GPIO，OUTPUT 0
- ✧ PB.0 为 32K 时钟输出，power down 时切 GPIO，OUTPUT 0
- ✧ PB.1 是输入按键，使能内部上拉，power down 时保持
- ✧ PB.2 和 PB.3 OUTPUT 0
- ✧ PB.4 ~ PB.9 保持 GPIO 模式，打开内部上拉
- ✧ PB.10 和 PB.11 OUTPUT 0
- ✧ PB.12 本是 INPUT，power down 时打开上拉
- ✧ PB.13 和 PB.14 保持。PB.13 是 UART0_RX 外接到 MBUS transceiver 上，如果 UART0_RX 是 floating 的，请打开上拉
- ✧ PF.2 和 PF.3 切 GPIO，OUTPUT 0
- ✧ 两根 ICE 脚切 GPIO， INPUT，并打开上拉

好了有了上面的说明，下面的代码就很好理解了

```
/* 配置GPIO寄存器*/
PA12 = 0;
PA13 = 0;
PA14 = 0;
PA15 = 0;
PB0 = 0;
PB2= 0;
PB3 = 0;
PB10 = 0;
PB11 = 0;
PF2 = 0;
PF3 = 0;
GPIOA->PMD = 0x02;/* PA.0 open drain */
GPIOB->PMD = 0x00500051; /* PB.0/PB.2/PB.3/PB.10/PB.11 output */
GPIOC->PMD = 0x0; /* 全部INPUT模式 */
GPIOD->PMD = 0x0; /* 全部INPUT模式 */
GPIOE->PMD = 0x0; /* 全部INPUT模式 */
GPIOF->PMD = 0x50; /* PF.2/PF.3 output */
```

```
GPIOA->PUEN = 0x7E;/* PA.1 ~ PA.6 打开上拉 */
GPIOB->PUEN = 0x13F2;/* PB.1/PB4/PB5/PB6/PB7/PB8/PB9/PB12打开上拉*/
GPIOC->PUEN = 0x3C00;/* PC.10 ~ PC.13 打开上拉 */
GPIOD->PUEN = 0x0;
GPIOE->PUEN = 0xFFFF;
GPIOF->PUEN = 0x3C;/* exclude GPF0 and GPF1 which are LXT OUT/IN */

GCR->PA_L_MFP = 0x100;/* PA.2为EINT0模式，其余全部GPIO模式 */
GCR->PA_H_MFP = 0x0; /* 全部GPIO模式 */
GCR->PB_L_MFP = 0x10; /* PB.1为EINT1模式，其余全部GPIO模式 */
GCR->PB_H_MFP = 0x87700000;/* PB15为SEG19，PB13和PB14为UART模式，其余全部GPIO模式 */
GCR->PC_L_MFP = 0x88888888;/* 保持SEGxx */
GCR->PC_H_MFP = 0x88000088; /* 保持SEGxx */
GCR->PD_L_MFP = 0x88888888; /* 保持SEGxx/COMx */
GCR->PD_H_MFP = 0x88800888;/* 保持DH2/DH1/V1/V2/V3 */
GCR->PE_L_MFP = 0x0; /* 全部GPIO模式 */
GCR->PE_H_MFP = 0x0; /* 全部GPIO模式 */
GCR->PF_L_MFP = 0x0000FF;/* PF0/PF1保持晶振模式，其它的切GPIO模式 */
/*进入power down*/
UnlockReg();
CLK->PWRCTL |= CLK_PWRCTL_LXT_EN;
CLK->PWRCTL |= (CLK_PWRCTL_PWRDOWN_EN | CLK_PWRCTL_DELY_EN);    /* Set power down bit */
SCB->SCR |= 0x04;                          /* Sleep Deep */
LockReg();
__WFI();  /* system really enter power down here ! */
```

其实上面的代码一部分可以挪到初始化函数里，真正power down时需要切的IO不多。都写在这里有个好处是进入debug时查看寄存器的值是否与你想的一样比较方便，一个一个比对就行了。

其实完全可以将代码精简为如下：

```
Void EnterPowerdown()
{
    /* 配置GPIO寄存器*/
    GPIOF->PUEN = 0x3C;/* exclude GPF0 and GPF1 which are LXT OUT/IN */
    GCR->PA_H_MFP = 0x0; /* 全部GPIO模式 */
    GCR->PB_L_MFP = 0x10; /* PB.1为EINT1模式，其余全部GPIO模式 */
    GCR->PF_L_MFP = 0x0000FF;/* PF0/PF1保持晶振模式，其它的切GPIO模式 */
    /*进入power down*/
```

```
    UnlockReg();
    CLK->PWRCTL |= CLK_PWRCTL_LXT_EN;
    CLK->PWRCTL |= (CLK_PWRCTL_PWRDOWN_EN | CLK_PWRCTL_DELY_EN); /* Set power down bit */
    SCB->SCR |= 0x04;                          /* Sleep Deep */
    LockReg();
    __WFI();  /* system really enter power down here ! */
    /*唤醒之后恢复状态*/
    GPIOF->PUEN = 0x0;
    GCR->PA_H_MFP |= 0x66660000;/* 恢复SPI1的4根引脚 */
    GCR->PB_L_MFP |= 0x1; /* CKO1 */
    GCR->PF_L_MFP |= 0xFF7700; /* UART1和ICE */
}
```

其它的都移到初始化函数里，系统上电之后初始化一次就行了。

GPIO的设定确定之后，最好连接Nu-Link进入debug状态实际看一下每个GPIO寄存器的状态。DOUT和DIN寄存器也要看一下，DOUT决定输出引脚的输出值，DIN是该引脚当前状态。如果某个引脚为INPUT并打开了上拉，可是DIN显示它的值是0就要特别小心了，可能外部有下拉。

1.7.2 查看寄存器设定

接上ICE，keil进入Debug模式，将断点放在__WFI指令这里，程序停在这里以后，逐一确认PWRCTL、AHBCLK和APBCLK以及GPIO的PMD（模式寄存器），OFFD（关闭数字通路寄存器），PUEN和MFP寄存器的内容是不是和你设定的一样。以及DOUT和DIN的值是不是如预期。低功耗设定原则就如1.2节所述。

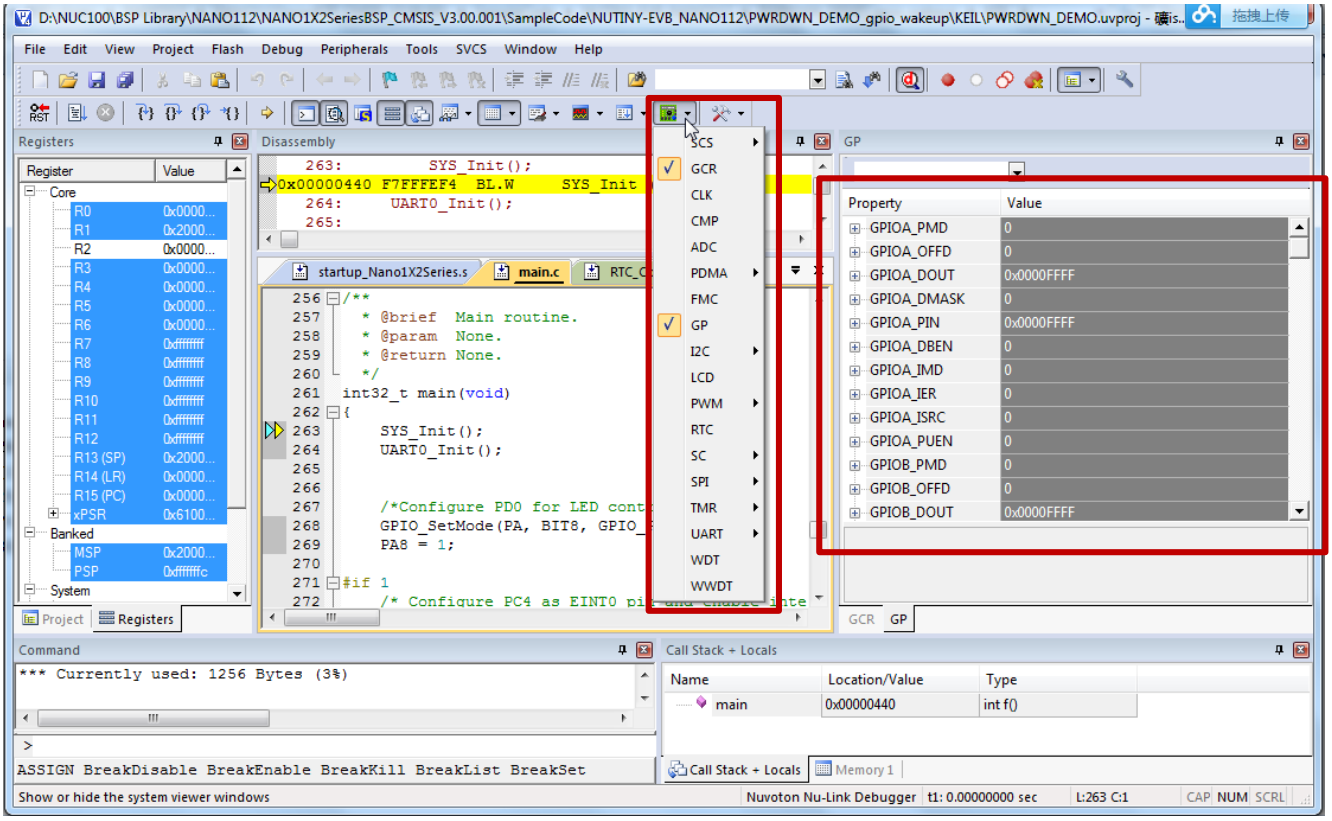


图 1-1 keil 调式额面

左边的红色框是寄存器列表，点击之后，该IP的寄存器列表会出现在右边。

一个一个寄存器比对，看跟你设定的是否一致。

1.7.3 模拟外设

查看一下模拟外设关闭了吗？模拟外设需要软件写该外设的寄存器，关闭该外设。例如：关闭ADC需要写ADCR寄存器bit[0] ADEN=0

1.7.4 晶振

晶振需要特别注意，如果没有接晶振，一定不能使能晶振。

1.7.5 其它 IC

板子上外接的其它器件也会影响功耗，如果到此功耗还是不如预期，可能是跟其它IC相接的引脚并不是和你想的一致。一根一根查一下，应该不难排除。

2 真实案例分析

下面分析几个真实的案例让大家体验一下低功耗的处理过程。

2.1 板子进入 power down 功耗 5mA，分析原因

1) 发现进入低功耗时 LDO 的电压是 1.77V，明显没有进入 power down
2) 经检查代码发现是使能了 RTC 唤醒
   - RTC 的寄存器要读/写的话需要使能 ACCESS Enable 寄存器，不然一些寄存器读都是不可以的。所以进入调试模式时，无法通过 TTR 寄存器查看是否使能了唤醒模式。只能查 RTC 相关代码
3) 关闭 RTC 唤醒之后，此时功耗 300uA。
4) 处理进入 power down 时的 IO 状态
   - 因为板子上所有的外设在 power down 时都是断电的，所以所有跟外设相连的 IO 统统设为 GPIO 功能（MFP），并输出 0。
   - 系统通过 PB.8 唤醒，该 IO 设为 INPUT 模式(PMD)，使能内部上拉。
   - 跟充电管理芯片相连的 IO 都设为 INPUT 模式，不使能内部上拉
   - 其他没有用到的 IO 都设为 INPUT，使能内部上拉（PUEN）
   - 测量电源电压的 ADC 引脚切成 GPIO 模式，INPUT，并关闭数字通路

   处理 IO 之后功耗 188uA

5) 板子上有一片 LDO 芯片，将 3.6V 转成 3.0V 给 MCU 使用，拿掉该 LDO 之后功耗 88uA。就是说该 LDO 耗掉了 100uA。
6) 充电电路从 VDD 到 VSS 有通路，拿掉了相连的电阻，之后功耗开始飘。低的时候到 5uA 高的时候到 20uA
   - 关闭所有 GPIO 的数字通路 OFFD，发现功耗固定在 4.2uA 不再飘
   - 依次使能 GPIO 的数字通路，发现与充电管理芯片相连的 2 根 PB.9 和 PB.10 的数字通路一定要关闭，不然功耗就会飘
   - 测量 PB.8 和 PB.9 的波形，发现这两根引脚的电压为 2.4V，会导致漏电，关闭这两根引脚的数字通路

至此，整个板子在进入低功耗时耗电4.2uA

2.2 板子放一个晚上功耗飘到了 100uA，分析原因

板子上只有新唐的NANO，功耗正常时2uA

经过分析发现没有外接高速晶振，但是使能了晶振（PWRCON）。该晶振是在CONFIG0中使能的，不是在软件里面使能，比较隐蔽。将晶振关闭之后电流没有再飘。

3 示例代码

该板子外接段式LCD屏，进入power down的时候LCD和RTC要保持工作，另外UART0收到数据唤醒系统，还有两个按键也可以唤醒系统。其实就是上面1.7.1的例子。

3.1 进入低功耗函数

```
Void EnterPowerdown()
{
    /* 配置GPIO寄存器*/
    GPIOF->PUEN = 0x3C;/* exclude GPF0 and GPF1 which are LXT OUT/IN */
    GCR->PA_H_MFP = 0x0; /* 全部GPIO模式 */
    GCR->PB_L_MFP = 0x10; /* PB.1为EINT1模式，其余全部GPIO模式 */
    GCR->PF_L_MFP = 0x0000FF;/* PF0/PF1保持晶振模式，其它的切GPIO模式 */
    /*进入power down*/
    UnlockReg();
    CLK->PWRCTL |= CLK_PWRCTL_LXT_EN;
    CLK->PWRCTL |= (CLK_PWRCTL_PWRDOWN_EN | CLK_PWRCTL_DELY_EN); /* Set power down bit */
    SCB->SCR |= 0x04;                           /* Sleep Deep */
    LockReg();
    __WFI();  /* system really enter power down here ! */
    /*唤醒之后恢复状态*/
    GPIOF->PUEN = 0x0;
    GCR->PA_H_MFP |= 0x66660000;/* 恢复SPI1的4根引脚 */
    GCR->PB_L_MFP |= 0x1; /* CKO1 */
    GCR->PF_L_MFP |= 0xFF7700; /* UART1和ICE */
}
```

Revision History

Date	Revision	Description
2014.1.21	1.00	1. Initially issued.
2014.4.11	1.10	2. 添加 AHBCLK 和 APBCLK 处理 添加一些有问题原理图分析
2014.4.25	1.12	3. 功耗飘时多添加 2 种肯能原因 如果量不到 LDO 为 1.6v 原因说明
2014.12.15	1.15	4. 添加一些补充说明，让客户更容易理解
2015.10.28	1.17	5. 整理章节布局，修改 typos，更新模板
2015.11.12	1.18	6. 整理章节布局

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.