NXP Semiconductors
Application Note

Document Number: AN12085
Rev.1, 01/2019

如何使用 i.MX RT 低功耗功能

1. 介绍

本文档介绍了 i.MX RT 系列低功耗应用的设计要点。低功耗是 i.MX RT 的一个重要特性。

2. 芯片概述

本节介绍 i.MX RT 芯片的主要特性。

2.1. i.MX RT 芯片概述

i.MX RT 是基于 Cortex-M7 的芯片，运行速度高达 600 MHz，可提供高 CPU 性能和最佳实时响应。

- 基于 Cortex-M7 的处理器运行速度高达 600 MHz。
- 具有内部DCDC 和LDO 的高级电源管理模块，可降低外部电源的复杂性并简化电源排序。
- 各种存储器接口，包括SDRAM、Raw NAND FLASH、NOR 闪存、SD/eMMC。
- 用于连接外围设备的各种其他接口，例如 WLAN、Bluetooth™、GPS、显示器和相机传感器。
- 丰富的音视频功能，包括LCD 显示、基本2D 图形、摄像头接口、S/PDIF 和I2S 音频接口。
- 提供丰富的外设模块，如SPI、I2C、CAN、以太网、Flex-Timers、ADC等。
- 针对工业HMI、电机控制和家用电器领域。

目录

3. 低功耗概述

以下部分介绍电源、运行模式和低功耗模式。

3.1. 电源

表 1 列出了 i.MX RT 的外部电源轨。

Table 1. 外部电源轨

电源轨	描述
DCDC_IN	DCDC电源
VDD_HIGH_IN	模拟电源
VDD_SNVS_IN	SNVS和 RTC电源
USB_OTG1_VBUS USB_OTG2_VBUS	USB VBUS电源
VDDA_ADC	12位 ADC电源
VDDA_IN	LDO 2P5和 LDO 1P1电源
NVCC_SD0	SDIO1 bank 中GPIO 的电源(3.3 V 模式)
	SDIO1 bank 中GPIO 的电源(1.8 V 模式)
NVCC_SD1	SDIO2 bank 中GPIO 的电源 (3.3 V 模式)
	SDIO2 bank 中GPIO 的电源(1.8 V 模式)
NVCC_GPIO	GPIO bank里 GPIO的IO电源
NVCC_EMC	EMC bank里 GPIO的IO电源

3.2. 运行模式定义

表 2 列出运行模式定义。

Table 2. 运行模式定义

运行模式	定义
Overdrive run模式	• CPU 以 600 MHz 运行，过载电压为 1.275 V • 总线频率为 150 MHz • 所有外设均已启用并以目标频率运行
Full speed run模式	• CPU 以 528 MHz 运行，满载，电压降至 1.15 V • 总线频率为 132 MHz • 所有外设均已启用并以目标频率运行
Low speed run模式	• CPU 以 132 MHz 运行，电压降至 1.15 V • 总线频率为 66 MHz • 某些 PLL 掉电 • 20% 的外设处于活动状态，其他处于低功耗模式
Low power run模式	• CPU 以 24 MHz 运行，电压降至 0.95V • 总线频率为 12 MHz • 所有 PLL 断电，XTAL 24 M 掉电，OSC RC 24 M 启用 • 高速外围设备掉电

3.3. 运行模式配置

表 3 描述运行模式配置。

Table 3. 运行模式配置

	超速运行	全速运行	低速运行	低功耗运行
CCM LPM 模式	运行	运行	运行	运行
CPU 核	600 MHz	528 MHz	132 MHz	24 MHz
L1 缓存	开	开	开	开
FlexRAM	开	开	开	开
SOC 电压	1.275 V	1.15 V	1.15 V	0.95 V
模拟 LDO	开	开	开	在弱模式下
24MHz XTAL OSC	开	开	开	关
24MHz RC OSC	关	关	关	开
系统 PLL	开	开	开	关
其他 PLLs	开	开	根据需要开启	根据需要开启
模块时钟	开	开	根据需要开启	外围时钟关闭
RTC32K	开	开	开	开

3.4. 低功耗模式定义

表 4 列出了低功耗模式定义。

Table 4. 低功耗模式定义

低功耗模式	定义
System Idle模式	• CPU 可以在没有线程运行时自动进入该模式 • 所有外围设备都可以保持活动状态 • CPU 仅进入WFI模式，它会保留其状态，因此中断响应可以非常短
Low Power Idle模式	• 比系统空闲模式功耗低得多，退出所需的时间更长 • 所有 PLL 关闭，模拟模块在低功耗模式下运行 • 所有高速外围设备均采用电源门控，低速外围设备可保持低频率运行
Suspend模式	• 最省电、退出时间最长的模式 • 关闭所有 PLL，关闭 XTAL，关闭除 32 K 时钟以外的所有时钟 • 所有高速外设都是电源门控的，低速外设是时钟门控的
SNVS模式	• 所有 SOC 数字逻辑、模拟模块都关断， SNVS 域除外 • 32 KHz RTC有效

3.5. 低功耗模式配置

表 5 描述低功耗模式配置。

Table 5. 低功耗模式配置

	系统空闲	低功耗空闲	暂停	SNVS
CCM LPM模式	等候	等候	停止	-
Arm 核 (PDM7)	WFI	WFI	掉电	关
L1 缓存	开	开	掉电	关
FlexRAM (PDRET)	开	开	开	关
FlexRAM (PDRAM0)	开	开	掉电	关
FlexRAM (PDRAM1)	开/关	开/关	掉电	关
VDD_SOC_IN电压	1.15 V	0.95 V	0.925 V	关
SYS PLL	开	掉电	掉电	关
其他 PLL	掉电	掉电	掉电	关
24MHz XTAL OSC	开	关	关	关
24MHz RC OSC	关	开	关	关
LDO 2P5	开	关	关	关
LDO 1P1	开	关	关	关
WEAK 2P5	关	开	关	关
WEAK 1P1	关	开	关	关
带隙	开	关	关	关
低功耗带隙	开	开	开	关
AHB时钟	33 MHz	12 MHz	关	关
IPG 时钟	33 MHz	12 MHz	关	关
PER 时钟	33 MHz	12 MHz	关	关
模块时钟	根据需要开启	根据需要开启	关	关
RTC 32K	开	开	开	开

注意

在进入低功耗模式之前，每个模块都已按照配置表中的描述启用或禁用。 但是，对于唤醒，请确保恢复成原始设置。

3.5.1. 唤醒源

表 6 描述唤醒源。

Table 6. 唤醒源

	系统空闲	低功耗空闲	暂停	SNVS
GPIO 唤醒	是	是	是	-是(仅1引脚)
RTC 唤醒	是	是	是	是
USB 远程唤醒	是	是	是	否
其他唤醒源	是	是	否	否

注意

无论模式是系统空闲、低功耗空闲还是暂停，用户都必须先在 GPC 模块中使能唤醒中断，否则唤醒将会失败。.

SNVS模式下只能通过CPIO5_IO00引脚来将系统唤醒。

本文档不包括关于如何使用USB远程唤醒的部分。

4. 电源模式切换示例

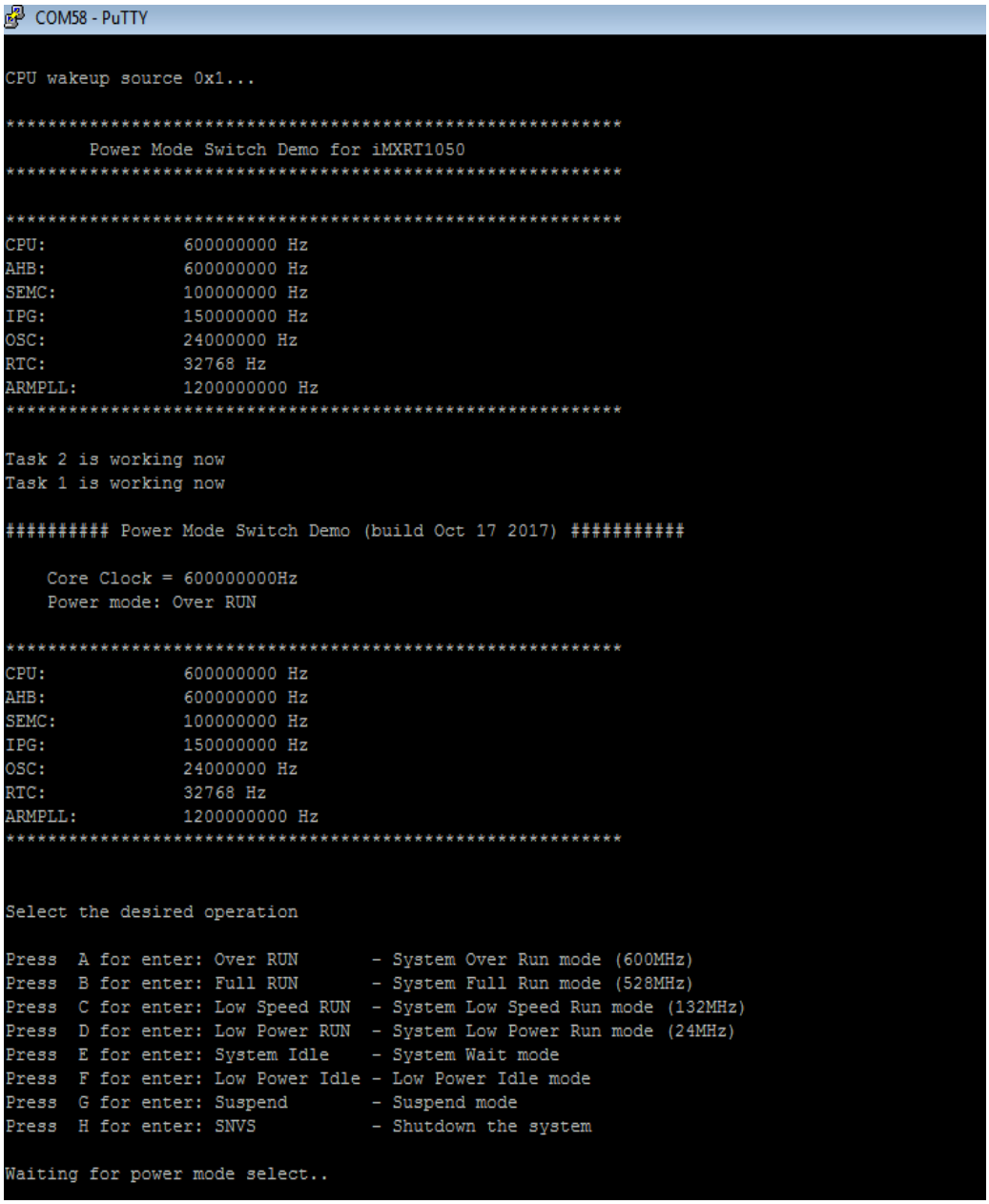
本节提供应用说明，电源模式切换菜单

设计功耗模式切换示例目的在于模拟客户低功耗应用案例。

特征:

- FreeRTOS 和tickless功能支持。
- 支持所有运行和低功耗模式切换。
- 支持WFI 指令后运行模式和暂停模式复位（由芯片控制）。
- 模拟两个正在运行的任务并在电源模式切换中显示任务状态。

4.1.1. 电源模式切换菜单



在此菜单中，用户可以切换电源模式以及从空闲、暂停模式中唤醒。

4.1.2. 任务状态显示

当用户选择转换到新的电源模式时，任务状态将会被显示出来。

1. 开始工作

```
Task 2 is working now
Task 1 is working now
```

2. 电源模式转换

   例如,
   Over Run模式 => System Idle模式

```
Waiting for power mode select..

WorkingTask 2: Transfer from Over RUN to System Idle
WorkingTask 1: Transfer from Over RUN to System Idle
```

3. 唤醒后转回Over Run模式

```
Waiting for key press..

Switch SW8 from off to on to wake up.
WorkingTask 2: Transfer from System Idle to Over RUN
WorkingTask 1: Transfer from System Idle to Over RUN

Next loop
```

4.1.3. 唤醒源

有两个唤醒源可供系统空闲、低功耗空闲和暂停模式选择使用，分别是SW8 键（在 EVK 板上）和 GPT 定时器。

```
Waiting for power mode select..

WorkingTask 2: Transfer from Over RUN to System Idle
WorkingTask 1: Transfer from Over RUN to System Idle
Select the wake up source:
Press T for GPT - GPT Timer
Press S for switch/button SW8.
```

- SW8 用户键唤醒
- GPT 定时器唤醒

当选择GPT定时器作为唤醒源时，另一个菜单会要求用户输入1 - 9秒作为唤醒前的等待时间。

```
Select the wake up timeout in seconds.
The allowed range is 1s ~ 9s.
Eg. enter 5 to wake up in 5 seconds.

Waiting for input timeout value...

3
Will wakeup in 3 seconds.
WorkingTask 2: Transfer from Low Power Idle to Over RUN
WorkingTask 1: Transfer from Low Power Idle to Over RUN

Next loop
```

5. 低功耗应用设计

本章将提供一些设计低功耗应用的指南。 此外，我们将给出一些关于低功耗模块启用和禁用的代码示例。

5.1. 总体描述

进入低功耗模式是一组启用/禁用模块的操作。 应注意低功耗模式配置。

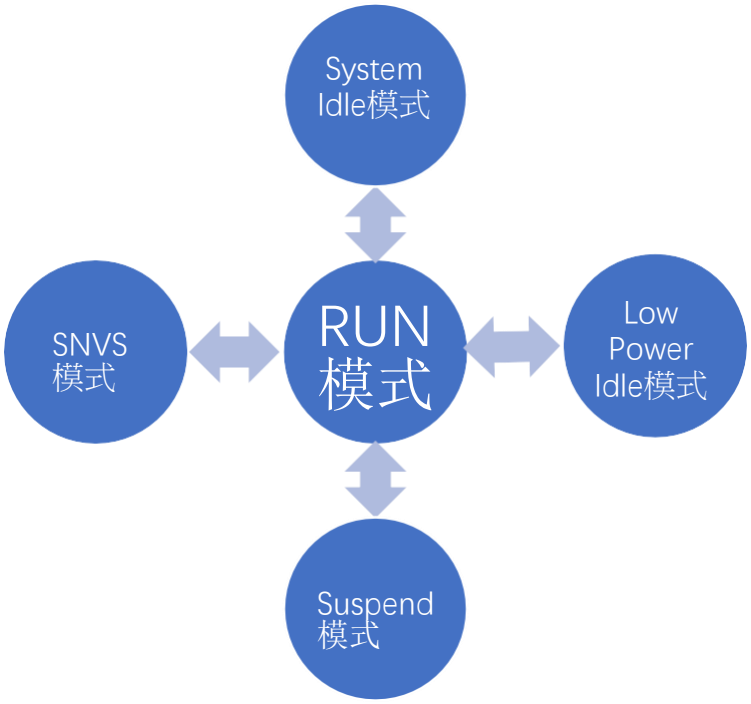
Table 7. 低功耗配置

	系统空闲	低功耗空闲	暂停	SNVS
CCM LPM 模式	等候	等候	停止	-
Arm Core (PDM7)	WFI	WFI	掉电	关
L1 Cache	开	开	掉电	关
FlexRAM (PDRET)	开	开	开	关
FlexRAM (PDRAM0)	开	开	掉电	关
FlexRAM (PDRAM1)	开/关	开/关	掉电	关
VDD_SOC_IN Voltage	1.15 V	0.95 V	0.925 V	关
SYS PLL	开	掉电	掉电	关
Other PLL	掉电	掉电	掉电	关
24MHz XTAL OSC	开	关	关	关
24MHz RC OSC	关	开	关	关
LDO 2P5	开	关	关	关
LDO 1P1	开	关	关	关
WEAK 2P5	关	开	关	关
WEAK 1P1	关	开	关	关
Band gap	开	关	关	关
Low Power Band gap	开	开	开	关
AHB Clock	33 MHz	12 MHz	关	关
IPG Clock	33 MHz	12 MHz	关	关
PER Clock	33 MHz	12 MHz	关	关
Module Clocks	根据需要开启	根据需要开启	关	关
RTC32K	开	开	开	开

5.2. 低功耗模式进入和退出顺序

在 i.MX RT 上，芯片可以进入每个低功耗模式并退出回到运行模式

图 1. 低功耗模式进入和退出序列



5.2.1. 进入system idle模式

System idle模式是一种简单的模式，大多数模块的默认设置不需要更改。

5.2.1.1. 怎样进入

要进入system idle模式，应首先更改核频率，然后关闭未使用的PLL。最后，核电压也需要做相应的改变。 在执行 WFI 指令之前，需要设置所使用模块的 DOZE 位。.

假设核心频率降低到132MHz。 如果核心频率从24MHz切换到132MHz，应该先提高核电压。

5.2.1.2. 如何退出

退出system idle模式也很简单。 首先调整核电压，然后启用 PLLs。

5.2.2. 进入 low power idle模式

5.2.2.1. 如何进入

要进入low power idle模式，请参阅下图。


设置时钟模式
• 将 CLPCR[LPM] 变为 WAIT
将 DC-变为-DC
• 切换DCDC 以便能使用内部 OSC
掉电 USBPHY
• 掉电 USBPHY
改变 PLL设置
• 更改 Arm PLL 以便于使用 24 MHz OSC
• 禁止使用其他所有 PLLs
• 参考 5.3.1和 5.3.2
启用弱 LDO
• 启用弱2P5 和 1P1 LDO
• 关闭常规 2P5 和1P1 LDO
• 参考 5.3.5
切换到 RC-OSC
• 切换到 24 MHz RC OSC
• 关闭外部 24 Mhz OSC
调整电压
• 调整核电压为 0.95 V
禁用常规带隙
• 禁用常规带隙并启用低功率带隙
• 参考 5.3.7
执行 WFI命令
• 设置外设进入DOZE模式，参考5.3.4
• 执行 WFI

5.2.2.2. 如何退出

要退出low power idle模式，请参考下图。

5.2.3. 进入Suspend模式

在Suspend模式下，当将 CLPCR[LPM] 设置为 STOP 时，系统将关闭 PLL，无需调整频率。

5.2.3.1. 如何进入


设置时钟模式
• 切换 CLPCR[LPM] 至 STOP
关闭 FlexRAM
• 根据APP要求关闭FlexRAM
• 参考5.3.6
掉电USBPHY
• 掉电USBPHY
调整电压
• 调整核电压为 0.95V
切换DCDC
• 切换DCDC以便使用内部OSC
发送掉电 CPU 请求
• 发送掉电 CPU 请求
执行 WFI命令
• 设置外设进入DOZE模式，参考 5.3.4
• 执行 WFI

5.2.3.2. 如何退出

5.2.4. SNVS模式

5.2.4.1. 如何进入

可以选择通过硬件或者软件这两种方式来进入SNVS模式。若要从软件

方式进入 SNVS 模式， 将 SNVS_LPCR[TOP] 设置为 1.

示例代码:

```
SNVS->LPCR |= SNVS_LPCR_TOP_MASK;
while (1) /* Shutdown */
    {
    }
```

若要从硬件方式进入SNVS模式， 长按开/关按钮（EVK板上的SW2）

5.2.4.2. 如何退出

长按开/关按钮（EVK 板上的 SW2）或使用唤醒引脚。

该芯片支持使用 WAKEUP 引脚 (GPIO5_IO00 ALT5) 来使主 SoC 上电从而退出 SNVS 模式。 要使用它，请按照以下提示操作：

- SNVS 必须处于 Dumb PMIC 模式（默认，必须是对于芯片上的 DCDC)
- 配置 IOMUXC_SNVS ，选择 WAKEUP 引脚 ALT5将其复用到 GPIO5_IO00
- 配置 GPIO5 ICR ，让它能被低电平或高电平触发
- 设置 GPIO5 IMR bit0 ，使能 GPIO5_IO00 中断
- 通过SW或ON/OFF按钮进入SNVS模式, 然后芯片上的DCDC将会被关闭, PMIC_ON_REQ引脚将变成低电平状态以此作为信号通知外部
- 将 WAKEUP 引脚置位至少两个 32 kHz 周期（高电平或低电平取决于 GPIO5 ICR 配置），以便 SNVS 模块对 GPIO5 中断进行采样
- SNVS 将 PMIC_ON_REQ 引脚置为高电平，片上 DCDC 开始攀升
- Soc的上电流程(SNVS模式除外)将开始执行(ROM boot,…)

5.3. 低功耗设计要点

5.3.1. 如何改变核频率

当核的频率需要改变时，核的电压也需要改变。改变核的率和电压是有规律的。

- 提高核的率，首先是电压，然后是频率。
- 降低核的频率，先降低频率，再降低电压。

5.3.2. 如何改变 PLL 频率

改变 PLL 频率顺序如下：

- 设置BYPASS和 ENABLE 位以先将时钟信号暂设为旁路。
- 设置 PLL时钟分频器。
- 清除PLL BYPASS位。
- 通过测检查CCM_CDHIPR 寄存器来确认 CCM 操作是否完成。

示例代码：

```
/* Wait CCM operation finishes */
        while (CCM->CDHIPR != 0)
        {
        }
```

注意

当需要改变 PLL 的频率时，先改变BYPASS 时钟，然后改变 PLL 频率，最后切换回 PLL。

5.3.3. 如何更改 SPI Flash 和 SDRAM 的时钟源

通常在任何 RUN 模式下，如果使用 SPI Flash 或 SDRAM 来存储代码或数据，则不建议更改根时钟的频率。更改时钟源和频率对R/W SPI Flash 或 SDRAM 来说是很危险的。如果必须执行这些操作，代码应该在 On-Chip RAM 中运行。

5.3.4. 如何在 FreeRTOS 中启用tickless低功耗功能

FreeRTOS tickless空闲模式在空闲时段（没有执行的应用程序任务的时段）将会停止周期性tick中断，然后在tick中断重新启动时对 RTOS 时钟内核计数值进行更正调整。

停止tick中断可以使得微控制器保持深度省电状态，直到发生其他中断将其唤醒。

注意

MCU-SDK 已经提供了 fsl_tickless_systick.c和 fsl_tickless_gpt.c来实现 Tickles 低功耗功能。用户需要在 FreeRTOSConfig.h 中定义 configUSE_TICKLESS_IDLE。

5.3.4.1. 选择一个FreeRTOS 滴答计时器

通常，系统tick模块用作 FreeRTOS 中的滴答计时器。在 i.MX RT 系列上，Cortex-M7核中的系统tick模块可以支持两个定时器源，核时钟和来自 24M 振荡器并分频为 100KHz的时钟。

在 FreeRTOS 中，在tickless停止时所消耗的时间将得到补偿。tickless停止消耗的时间不应超过最大持续时间（否则 FreeRTOS 将无法知道要补偿多少时间）。计算最大持续时间计数的代码在tickless代码中的函数 vPortSetupTimerInterrupt() 中。

```
ulTimerCountsForOneTick = ( configSYSTICK_CLOCK_HZ / configTICK_RATE_HZ );
xMaximumPossibleSuppressedTicks = portMAX_32_BIT_NUMBER / ulTimerCountsForOneTick;
```

对于系统tick模块, *portMAX_32_BIT_NUMBER*is 0x00FFFFFF 和 *configTICK_RATE_HZ*正常设为 1000, 意味着单个tick持续时间为 1ms.

因此，如果选择核时钟 600MHz 作为系统tick模块时钟源， *ulTimerCountsForOneTick* 将会是 600000 ，*xMaximumPossibleSuppressedTicks* 将会是 27.962025，也就是 27.962025 ms。

27.962025 ms 对于tickless持续时间来说太短了。意思是在tickles停止时，系统需要每27.962025ms唤醒一次来补偿时间。

因此，当考虑在 FreeRTOS 应用程序中使用低功耗 tickles 时，我们最好不要使用核心时钟作为系统时钟源，而是使用 100KHz 时钟源。在这种情况下, *ulTimerCountsForOneTick* 将会是100 ，

xMaximumPossibleSuppressedTicks 将会是167772.15 ms, 差不多168s。这是一个可以接受的持续时间。

要使用 100KHz 时钟，只需在FreeRTOSConfig.h中定义 configSYSTICK_CLOCK_HZ

#define configSYSTICK_CLOCK_HZ (10000U)

5.3.4.2. 选择一个无滴答计时器

在 tickless模式中

FreeRTOS 也需要补偿消耗的时间，但我们需要选择一个不能超过 FreeRTOS 可以支持的最大时间限度的计时器。.

Systick 是此计时器的首选。 但是在 i.MX RT 中，GPC 模块中没有包含 systick 中断。 所以systick中断不能唤醒系统，只能唤醒Arm核。 我们需要选择另一个计时器作为无滴答计时器。

无滴答计时器的要求：

- 计时器。
- 被包含在GPC模块中，可唤醒系统。
- 在系统空闲、低功耗空闲和暂停模式下工作，即使在核和所有 PLL 掉电时也可运行。

GPT 模块可以使用 32K RTC（通过设置 GPT_CR[CLKSRC] = 4）作为时钟源，可以工作在系统空闲、低功耗空闲和暂停模式。

5.3.4.3. WFI指令前后的hook函数

FreeRTOS tickless 空闲模式提供了两种方法。

- vPortSuppressTicksAndSleep功能。
  在tickless模式, 系统将在空闲任务中调用 vPortSuppressTicksAndSleep 函数，并将预期的睡眠时间（以微秒为单位）作为参数传递。

用户可以实现自定义的 vPortSuppressTicksAndSleep 并且空闲任务将定期调用此函数。.

- 优点: 可以根据用户需要处理WFI指令。
  例如，在暂停模式下，内核是处于掉电状态，因此唤醒后会将内核复位。但是用户可以在复位后从 WFI处继续 运行后续代码的。为了达到此目的，用户需要设置一个还原点，以便能在复位后进行还原。
  在这种情况下，用户更容易处理客户函数中的 WFI 指令。
- 缺点: 用户函数需要计算tickles 周期并调用vTaskStepTick 来补偿消耗的时间。
  注意: 用户可以参考 fsl_tickless_systick.c 和 fsl_tickless_gpt.c 中的 vPortSuppressTicksAndSleep 函数作为例子来学习如何使用时间补偿。
- 在WFI 指令前后调用的两个hook函数， configPRE_SLEEP_PROCESSING和 configPOST_SLEEP_PROCESSING。

如果用户想在 fsl_tickless_systick.c 和 fsl_tickless_gpt.c 中重复使用 vPortSuppressTicksAndSleep，可以用 configPRE_SLEEP_PROCESSING 和 configPOST_SLEEP_PROCESSING 来更改和恢复模块以降低功耗。

- 优点: 不需要在意时间补偿。 fsl_tickless_systick.c 和 fsl_tickless_gpt.c 中的 vPortSuppressTicksAndSleep 函数将为您完成。
- 缺点: 不是很灵活。 用户不能在自己的函数中执行 WFI 指令。

在 Power Mode Switch 示例中，我们使用第二种方法来重复使用 vPortSuppressTicksAndSleep。

5.3.5. Doze和IPG STOP信号

为了降低功耗，应该将Doze和Stop位置位，并设置将各个模块在进入低功耗模式之前进入Doze of Stop状态。在示例代码中，所有支持的外设的Doze的位都被使能，但在应用程序中，开发人员应该知道需要哪些模块并使能相应模块的Doze的位

5.3.5.1. Doze模式

对于处于休眠模式的各个模块，AHB 时钟和串行时钟域将在内部被关闭，但 IPS 总线时钟不会被关闭。通过设置IOMUXC_GPR_GPR8和IOMUXC_GPR_GPR12中相应的DOZE位进入该模式，并通过清除IOMUXC_GPR_GPR8和IOMUXC_GPR_GPR12中的相应位退出该模式。

注意

Doze模式适用于所有低功耗模式。 因此，应在进入任何低功耗模式之前设置Doze位。

示例代码:

在进入低功耗模式前:

```
IOMUXC_GPR->GPR8 = 0xaaaaaaaa;
        IOMUXC_GPR->GPR12 = 0x0000000a;
```

在唤醒后:

```
IOMUXC_GPR->GPR8 = 0x00000000;
        IOMUXC_GPR->GPR12 = 0x00000000;
```

5.3.5.2. Stop 模式

当系统要求外围设备进入Stop模式时，它将等待所有事务完成并向系统返回ACK握手信号。ACK 握手信号返回后，系统可以在系统级别对 AHB 总线时钟、IPG 总线时钟和串行时钟进行门控操作。内部没有时钟门控操作。通过设置IOMUXC_GPR_GPR8 和IOMUXC_GPR_GPR12 中相应的DOZE 和STOP 位进入该模式， 系统应等待IOMUXC_GPR_GPR4 和IOMUXC_GPR_GPR7 中相应的位被置位。暂停模式从属于Stop模式。 因此在进入暂停模式之前应该设置Stop和Doze位。

在示例代码中，所有支持的外设的 STOP 和 DOZE 位都已使能，但在应用程序中，开发人员应了解需要哪些模块并使能相应模块的 STOP 和 DOZE 位。

示例代码:

在进入暂停模式之前:

```
IOMUXC_GPR->GPR4 = 0x00000011;
while ((IOMUXC_GPR->GPR4 & 0x00110000) != 0x00110000){};
IOMUXC_GPR->GPR4 = 0x000036ff;
IOMUXC_GPR->GPR7 = 0x0000ffff;
IOMUXC_GPR->GPR8 = 0xfffcffff;
IOMUXC_GPR->GPR12 = 0x0000000a;
while ((IOMUXC_GPR->GPR4 & 0x36f90000) != 0x36f90000){};
while ((IOMUXC_GPR->GPR7 & 0xffff0000) != 0xffff0000){};
```

在唤醒后：

```
IOMUXC_GPR->GPR4 = 0x00000000;
IOMUXC_GPR->GPR7 = 0x00000000;
IOMUXC_GPR->GPR8 = 0x00000000;
IOMUXC_GPR->GPR12 = 0x00000000;
```

查看上面的代码，STOP 请求首先发送到 EDMA 和 ENET。

```
IOMUXC_GPR->GPR4 = 0x00000011;
while ((IOMUXC_GPR->GPR4 & 0x00110000) != 0x00110000){};
```

原因在于EDNA和NET是两个模块，可以作为BUS上的master，发起传输。如果代码在 SPI Flash 上运行，则stop请求不应发送到 SPI Flash。

注意

确保在进入 暂停模式之前，如果相应模块的 STOP 请求被确认，系统应等待确认位被置位。否则系统可能无法进入暂停模式。

如果代码在 SDRAM 中运行并且用户想要进入暂停模式。 SDRAM 需要等待 STOP 信号才能进入自刷新模式。 在这里，如果 STOP 请求被发送到 SDRAM 并使用轮询的方法来确认是否被置位，有可能永远轮询不到。因为代码在 SDRAM 上运行，SDRAM 上的事务不能停止。在这种情况下，代码或数据应在 SPI Flash 或芯片上 RAM 中运行。

注意

根据时钟门控和 IP 模块的使能位是否设置，每个 IP 模块在发送停止信号时可能有不同的反应。

- 一些模块即使时钟门关闭也可以确认stop信号，例如 PIT。
- 当时钟门关闭时，某些模块无法确认stop信号。但是当时钟门打开时，它可以确认stop信号，例如 EDMA。.
- 当时钟门控打开且模块的使能位为ON时，某些模块可以确认stop信号。目前只有CAN属于这种类型。

5.3.6. 如何启用弱模拟 LDO

i.MX RT 上有两种类型的模拟 LDO，常规 LDO 和低功耗弱 LDO。1.1 V 和 2.5 V LDO 包括一个备用设备、自偏置、低精度的弱稳压器，可用于需要在低功耗模式下保持 1.1 V 和 2.5 V 输出电压的应用，其中主稳压器及其 相关的全局带隙参考模块被禁用。使用弱 LDO 可以降低低功耗空闲模式下的功耗。

- 启用弱模拟LDO

```
/* Enable regular 2P5 and wait it stable */
PMU->REG_2P5_SET =   PMU_REG_2P5_ENABLE_LINREG_MASK;
while ((PMU->REG_2P5 & PMU_REG_2P5_OK_VDD2P5_MASK) == 0)
  {
  }
/* Turn off weak 2P5 */
PMU->REG_2P5_CLR = PMU_REG_2P5_ENABLE_WEAK_LINREG_MASK;

/* Enable regular 1P1 and wait for stable */
PMU->REG_1P1_SET =   PMU_REG_1P1_ENABLE_LINREG_MASK;
while ((PMU->REG_1P1 & PMU_REG_1P1_OK_VDD1P1_MASK) == 0)
{
```

```
}
/* Turn off weak 1P1 */
PMU->REG_1P1_CLR = PMU_REG_1P1_ENABLE_WEAK_LINREG_MASK;
```

- 禁止使用弱模拟LDO

```
/* Enable weak 2P5 and turn off regular 2P5 */
PMU->REG_2P5 |=
PMU_REG_2P5_ENABLE_WEAK_LINREG_MASK;PMU->REG_2P5
&= ~PMU_REG_2P5_ENABLE_LINREG_MASK;
/* Enable weak 1P1 and turn off regular 1P1 */
PMU->REG_1P1 |=
PMU_REG_1P1_ENABLE_WEAK_LINREG_MASK;PMU->REG_1P1
&= ~PMU_REG_1P1_ENABLE_LINREG_MASK;
```

5.3.7. 如何禁用 OCRAM电源

i.MX RT1050 芯片有芯片上的FlexRAM，这是由 I-TCM、D-TCM 和通用芯片上 RAM (OCRAM) 划分出来的。FlexRAM 管理一个大的芯片上 RAM 阵列。FlexRAM 有16 个RAM bank，分为Bank0、FlexRAM0（Bank 1 - 7）和FlexRAM1（Bank 8 -15）。下面列出了对OCRAM 组的描述。

Table 8. 对OCRAM bank的描述

	Bank 0	Bank 1-7	Bank 8-15
别名	Bank0	FlexRAM0	FlexRAM1
电源域	SNVS	PDRAM0	PDRAM1
电源门位	N/A	GPC_CNTR[PDRAM0_PGE]	PGC_MEGA_CTRL[PCR]
电源门默认值	N/A	1 (将在低功耗模式下掉电)	0 (不会掉电)
LPM 模式开/关状态	开	可控	可控
掉电时刻	在 SNVS模式, bank0 会掉电。	GPC_CNTR[PDRAM0_PGE]被置位，并且当核执行 WFI 指令时，Bank 1-7将关闭	PGC_MEGA_CTRL[PCR]被置位，并且Bank 8-15将会被立即关闭

注意

注意当电源门位被置位时，FlexRAM0 不会立即关闭，除非核执行 WFI 指令。但 FlexRAM1 将立即关闭。

代码示例:

- 关闭 FlexRAM0 和FlexRAM1电源

```
/* Turn off FlexRAM1 */
PGC->MEGA_CTRL |= PGC_MEGA_CTRL_PCR_MASK;
/* Turn off FlexRAM0*/
GPC->CNTR |= GPC_CNTR_PDRAM0_PGE_MASK;
```

- 打开FlexRAM0 和FlexRAM1电源

```
/* Turn on FlexRAM1 */
PGC->MEGA_CTRL &= ~PGC_MEGA_CTRL_PCR_MASK;
/* Turn on FlexRAM0*/
GPC->CNTR &= ~GPC_CNTR_PDRAM0_PGE_MASK;
```

5.3.8. 带隙切换

在 i.MX RT 中，有两个带隙模块，常规带隙和低功耗带隙。 在低功耗应用中，应在进入低功耗Wait或Stop状态之前使用低功率带隙。

注意

在暂停模式下，STOP模式将在CLPCR 中被设置，这将有助于在低功耗Stop状态自动将带隙切换到低功耗带隙。

代码示例:

将带隙切换到低功耗带隙:

```
/* Switch band gap */
   PMU->MISC0_SET     = 0x00000004;
   XTALOSC24M->LOWPWR_CTRL_SET =
XTALOSC24M_LOWPWR_CTRL_LPBG_SEL_MASK;PMU->MISC0_SET        =
CCM_ANALOG_MISC0_REFTOP_PWD_MASK;
Switch band gap back to regular band gap:
/* Restore band gap */
      /* Turn on regular band gap and wait for stable */ CCM_ANALOG-
      >MISC0_CLR = CCM_ANALOG_MISC0_REFTOP_PWD_MASK;
      while ((CCM_ANALOG->MISC0 & CCM_ANALOG_MISC0_REFTOP_VBGUP_MASK) == 0)
      {
      }
      /* Low power band gap disable */
      XTALOSC24M->LOWPWR_CTRL_CLR =
      XTALOSC24M_LOWPWR_CTRL_LPBG_SEL_MASK;PMU->MISC0_CLR= 0x00000004;
```

5.3.9. 启用 DCDC 的 DCM 模式

DCDC 模块提供断续导通模式（DCM）和连续导通模式（CCM）。 它可以在运行模式下使用 CCM 模式或 DCM 模式~~下运行~~。使用哪种模式取决于负载电流的水平。通常芯片处于运行模式时采用CCM模式，芯片处于低功耗模式时采用DCM模式。如果在 运行模式下功耗较低，也可以使用 DCM 模式。DCM 模式可以减少 DCDC_IN 的电流，但 DCDC_OUT 电流不会改变。

注意

由于不连续导通模式 (DCM) 可以在低电流负载的情况下提高 DCDC 的效率，所以推荐优先选择该模式。

应在进入低功耗模式之前设置DCM 模式。 详细步骤如下:

pwd_zcd=0x0 pwd_cmp_offset=0x0
loopctrl_en_rcscale=0x4
DCM_SET_CTRL=1'b1

唤醒时不要忘记恢复到 CCM 模式。 并启用 CCM 模式:

pwd_zcd=0x1 pwd_cmp_offset=0x0
loopctrl_en_rcscale=0x3

5.3.10. 启用和禁用RBC位

在暂停模式下，RBC 位需要在执行 WFI 指令前置位，唤醒后清零。RBC 位在 CCM_CCR 寄存器中。
CCM_CLPCR 中有一个 VSTBY 位，它将电压改变为待机电压（DCDC 将进入省电模式）。在低功耗空闲模式和暂停模式下需要对该位置位。
启用 RBC_EN 和 VSTBY 将会使模拟部分在 STOP模式中进入低功耗。

启用和禁用 RBC_EN 位需要注意两点。

- 需要在~~设~~置位之前屏蔽GPC 中断。
- ~~设~~置位后需要延迟（3us）。

示例:

- 启用RBC_EN:

```
/* Mask all GPC interrupts before enabling the RBC counters to
        avoid the counter starting too early if an interrupt is alreadypending.
     */
     for (i = 0; i < LPM_GPC_IMR_NUM; i++)
     {
          gpcIMR[i] = GPC->IMR[i]; GPC-
          >IMR[i] = 0xFFFFFFFFU;
     }

     /* Enable the RBC bypass counter here to hold off the interrupts. RBC counter
       * = 32 (1ms). Minimum RBC delay should be 400us.
       */
     CCM->CCR = (CCM_CCR_RBC_EN_MASK | CCM_CCR_COSC_EN_MASK | CCM_CCR_OSCNT(0xAF));

     /* Recover all the GPC interrupts. */for (i = 0; i <
     LPM_GPC_IMR_NUM; i++)
     {
          GPC->IMR[i] = gpcIMR[i];
     }

     /* Now delay for a short while (3usec) Arm is at 528MHz at this point
       *  so a short loop should be enough. This delay is required to ensure that
       *  the RBC counter can start counting in case an interrupt is already pending
       *  or in case an interrupt arrives just as Arm is about to assert DSM_request.
       */
     for (i = 0; i < 22 * 24; i++)
     {
          __NOP();
}
```

```
Disable RBC_EN:
/* Mask all GPC interrupts before disabling the RBC counters */for (i = 0; i <
    LPM_GPC_IMR_NUM; i++)
    {
        gpcIMR[i] = GPC->IMR[i]; GPC-
        >IMR[i] = 0xFFFFFFFFU;
    }
    /* Disable the RBC bypass counter */CCM-
    >CCR &= ~CCM_CCR_RBC_EN_MASK;
    CCM->CCR &= ~CCM_CCR_REG_BYPASS_COUNT_MASK;

    /* Now delay for 2 CKIL cycles (61usec). Arm is at 528MHz at this point
     * so a short loop should be enough.
     */
    for (i = 0; i < 528 * 22; i++)
    {
        _NOP();
    }

    /* Recover all the GPC interrupts. */for (i = 0; i <
    LPM_GPC_IMR_NUM; i++)
    {
        GPC->IMR[i] = gpcIMR[i];
    }
```

5.3.11. ERR007265的解决方法

Errata ERR007265 描述了 SoC 可能无法进入低功耗模式的情况。

ERR007265: CCM: 当使用不正确的低功耗序列时，SoC 在 Arm 核执行 WFI 之前进入低功耗模式。

软件解决方法:

1) 软件应通过设置 IOMUX_GPR1_GINT 来触发 GPR_IRQ 始终挂起。
2) 在设置 CCM 低功耗模式之前，软件应在 GPC 中取消屏蔽 GPR_IRQ。
3) 软件应在设置 CCM 低功耗模式后（设置 CCM_CLPCR 的第0和1位）屏蔽 GPR_IRQ。

5.3.12. Suspend模式复位后如何运行

5.3.12.1. 介绍

Suspend模式是一种功耗最低的低功耗模式。但随着核掉电，唤醒其实是复位唤醒。 在System idle和low power idle下，唤醒时代码可以在 WFI 指令后继续运行，这很方便，但无法获得最小的功耗。 在Suspend模式下可以获得最小的功耗。 复位后，核可以检测是否是因Suspend模式唤醒而导致的复位（通过PGC->CPU_SR[PSR]）。

如果系统想在 WFI 指令后运行代码，恢复数据可以保存在 OCRAM 上，起始地址的偏移量为 32KB。这意味着 FlexRAM0 不能在暂停模式下掉电。

5.3.12.2. SDRAM数据存储建议

在大多数应用中，用户可能希望使用 SDRAM 来存储数据。SDRAM 可以在suspend模式中自动进入自刷新模式，数据不会丢失。

警告

在进入暂停模式之前，应向 SDRAM 发送一个 STOP 请求，要求 SDRAM 进入自刷新模式。避免访问 SDRAM 以及在 OCRAM 中使用变量是很重要的。

6. 修订史

版本号	日期	实质性改变
0	11/2017	初始发行
1	01/2019	更新了低功耗解决方案的设计

7. 参考文献

1. i.MX RT 1050 Reference Manual
2. Arm Cortex M7 Reference Manual

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: AN12085
Rev.1
01/2019