恩智浦半导体
应用笔记

文档编号：AN5425
第1版，2018年5月

S32K1xx 的功耗管理

作者：恩智浦半导体

1. 介绍

设备的功耗及低功耗设计的影响是当下常见的话题。S32K1xx 系列包含内部功耗管理功能，可用于控制微控制器的电源使用并帮助实现嵌入式设计的目标。

本应用笔记讨论了如何使用功耗管理系统，提供示例并展示相应的电流测量结果。对如何使用 S32K1xx 系列每种可用功耗模式给出了建议。

目录

2. 功耗模式概览

传统嵌入式系统中典型的功耗模式是Run，Wait和Stop。ARM® Cortex™ M4 和 M0+ 功耗模式为 Run，Sleep 及 Deep Sleep。扩展功耗模式及其典型模式与 ARM Cortex M4 and M0+ 的模式关系如下图所示：

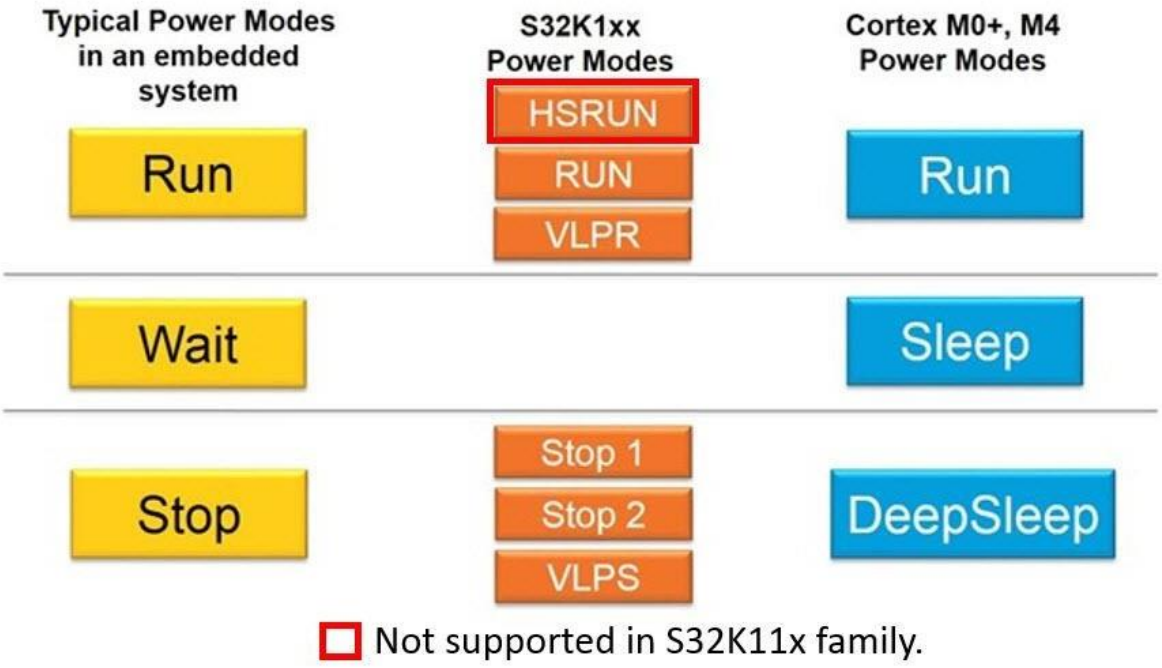

图1. 电源模式比较

2.1. ARM Cortex-M4 和 M0+ 功耗模式实现

ARM Cortex-M4 和 M0+ 内核具有三种主要操作模式：RUN、睡眠和深度睡眠。S32K1xx 芯片通过等待中断（WFI）指令切换调用Sleep和Deep Sleep模式。

图 2 展示了用于低功耗实现的 Cortex -M4 及 M0+ 架构。Sleep 和 DeepSleep状态在硬件中构建，以控制内核和中断控制器的时钟。进入Sleep时，NVIC逻辑保持激活，中断或复位可以将内核从Sleep中唤醒。进入Deep Sleep时，通过异步唤醒中断控制器（AWIC）从选定的一组源中唤醒 MCU。S32K1xx 系列参考手册中详细描述了这些唤醒源信息，相关内容请查看“内核概述”一章中的“异步唤醒中断控制器（AWIC）配置”部分。

使用退出时睡眠功能，ARM Cortex 内核可以通过另一种方式进入低功耗模式。在系统控制块中有一个称为系统控制寄存器（SCR）的寄存器，它包含几个与睡眠操作相关的控制位。SLEEPDEEP 位决定是选择睡眠模式还是深度睡眠模式。将 SLEEPONEXIT 位设置为1，处理器在完成所有异常处理程序的执行后将立即进入睡眠模式。相关更多信息，请参阅 ARM Cortex-M4 Devices Generic User Guide 和 ARM Cortex-M0+ Devices Generic User Guide。

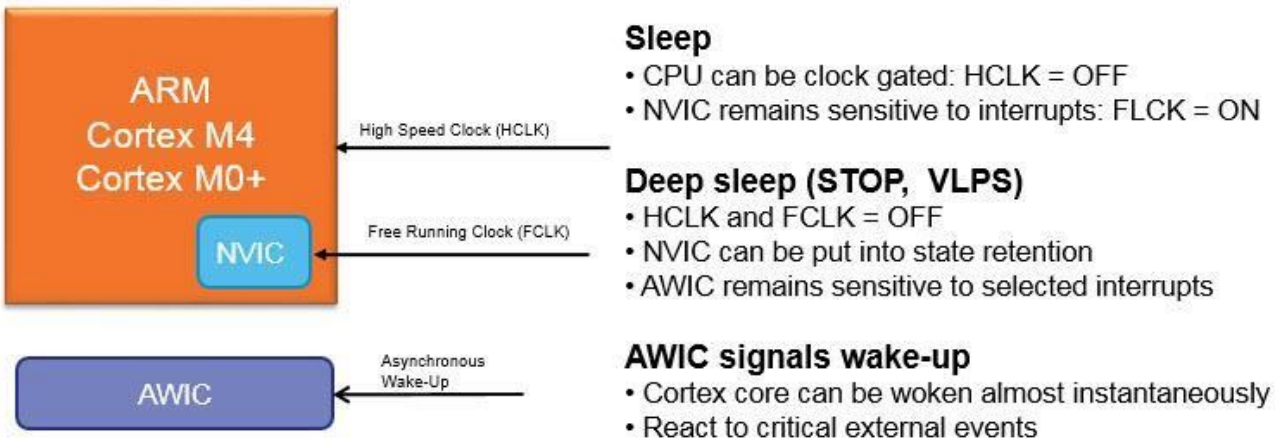

图2. ARM Cortex M4 和 M0+ 架构

3. 功耗模式描述

S32K1xx 提供多种功耗选项以允许用户针对所需功能级别优化功耗。

根据用户应用程序的要求，可以相应使用多种功耗模式，即在某些逻辑与/或存储器上提供状态保持、部分断电或完全断电。输入/输出（I/O）状态在所有功耗模式下都保持不变。有关不同功耗模式下各个模块功能的更多信息，请参阅本应用笔记 ***Modules in power modes*** 一章或 S32K1xx 的参考手册中的功率模式中的模块一章。下表比较了可用的几种功耗模式。

表 1. S32K1xx 电源模式

S32K1xx 模式	描述	内核模式	常规恢复方法
常规 RUN	Reset 后的默认模式，片上稳压器开启（内部电源完全工作）。	Run	-
高速 Run (HSRUN)[1]	芯片运行可达的最高性能状态。此模式下，与常规RUN模式相比，芯片以更高的频率RUN，但功能受限。内部时钟需要满足特定工作条件。[2]	Run	-
超低功耗 Run (VLPR)	片上稳压器处于低功耗模式，仅提供足够的功率以降低的频率运行芯片。 • Flash 内存降频访问模式（1MHz） • LVD 被关闭 • SIRC 为内核，总线及外设时钟提供低功耗4MHz时钟源。	Run	-

S32K1xx 模式	描述	内核模式	常规恢复方法
极低功耗模式（VLPS）（通过 WFI 指令）	在关闭低电压检测（LVD）操作的情况下芯片置于静态。这是保持引脚中断功能的情况下，系统的最低功耗模式。 • 部分外设时钟被停用。[3] • LPTMR，RTC 及 CMP 可以被使用。 • NVIC 被禁用。 • AWIC 用于中断唤醒。 • 内核被关闭。 • 所有 SRAM 均可操作（保留SRAM内容并维持 I/O 状态）。	Deep Sleep	中断（或复位）
Stop 1（通过 WFI 指令）	将芯片置于静态。保持 LVD 检测功能 • NVIC 被禁用。 • AWIC 用来唤醒中断。 • 部分外设时钟被停止。[3] • 内核时钟，系统时钟及总线时钟均被停止。	Deep Sleep	中断（或复位）
Stop 2（通过 WFI 指令）	将芯片置于静态。保持 LVD 检测功能 • NVIC 被禁用。 • AWIC 用来唤醒中断。 • 部分外设时钟被停止。[3] • 只有内核及系统时钟被停止，总线时钟保持活动状态。由系统时钟计时的总线主机和总线从机进入Stop模式，但由总线时钟计时的总线从机仍处于RUN模式。SCG中的时钟发生器和PMC的片上稳压器也保持在RUN模式。[4]	Deep Sleep	中断（或复位）

1. HSRUN 在 S32K11x 系列设备中不可用。
2. 内核和系统时钟必须为 112MHz 或更低，总线时钟必须设置为 56MHz 或更低，为内核时钟的整除数。Flash时钟必须设置为 28MHz 或更低，内核时钟与 Flash 时钟的比率限制为最大值 8。
3. 参阅本应用笔记 Modules in power modes 章节以获取更多信息。
4. 以下事件会触发退出 STOP 模式：复位，来自总线主机的一个异步中断（也对 STOP1 有效）和来自由总线时钟驱动的总线从机的一个异步中断（仅对 STOP2 有效）。

3.1. 功耗模式转换

下图展示了功耗模式转换。任何复位都会使芯片回到正常RUN状态。根据用户应用的需要，可以使用多种Stop模式，允许某些状态保持，逻辑与/或存储器的状态保持、部分断电或完全断电。在所有操作模式下，I/O状态以及寄存器内容都将保持不变。

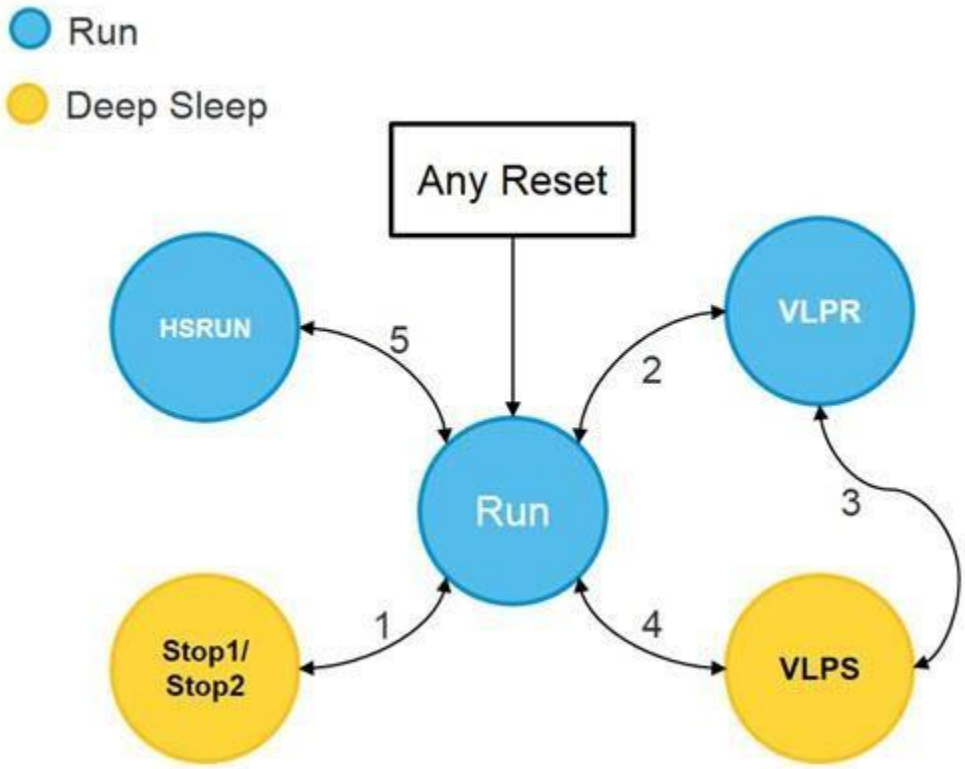

图3. 电源模式状态转换图

下表定义了上图所示的各种状态转换的触发条件。

表 2. 电源模式转换触发器

转换	从	到	模式转换触发命令/条件
1	RUN	STOP	SMC_PMCTRL[RUNM] = 00, SMC_PMCTRL[STOPM] = 000[1] 随后调用 WFI 指令。[2]
	STOP	RUN	中断[3] 或复位
2	RUN	VLPR	设置 SMC_PMPROT[AVLP] = 1, SMC_PMCTRL[RUNM] = 0b10[4]
	VLPR	RUN	设置 SMC_PMCTRL[RUNM] = 00 或复位。
3	VLPR	VLPS	SMC_PMCTRL[STOPM] = 0b010， 随后调用 WFI 指令。[2]
	VLPS	VLPR	中断。

转换	从	到	模式转换触发命令/条件
			注意：如果由RUN模式直接进入VLPS模式（转换 4），硬件会强制回退到RUN模式将不允许转换到超低速模式VLPR。
4	RUN	VLPS	SMC_PMPROT[AVLP] = 1，SMC_PMCTRL[STOPM] = 0b010，随后调用 WFI 指令。[2]
	VLPS	RUN	中断及VLPS模式直接从RUN模式或复位进入。
5	RUN	HSRUN	设置 SMC_PMPROT[AHSRUN] = 1，SMC_PMCTRL[RUNM] = 0b11。
	HSRUN	RUN	设置 SMC_PMCTRL[RUNM] = 00 或复位。

1. STOPO 寄存器必须配置为选择停止模式（STOP1 或 STOP2）。
2. 设置 SLEEPDEEP进入的 Sleep-now（WFI 指令）或 sleep-on-exit 模式，由 ARM 内核的系统控制寄存器控制。
3. 能够向设备提供异步中断的模块。
4. 内核、系统和总线时钟必须为 4MHz（最大值），Flash 时钟必须设置为 1MHz（最大值）。此外，所有异步时钟源都将被限制为 4MHz，由SCG_SRICDIV配置。

4. 低功耗模式下的时钟工作

MCU 中有多个时钟源可用。为了省电，可以通过配置 PCC 模块中外设控制寄存器的 CGC 字段来关闭大多数模块时钟。这些字段在任何复位后都会被清除，从而禁用相应模块的外设时钟。在初始化模块之前，需要设置 PCC 外设控制寄存器中的相应字段，以启用模块时钟。务必在关闭时钟之前禁用模块。

注意

禁用一个模块前，此模块的中断及 DMA 请求应先被禁用。

4.1. 系统时钟发生器（SCG）时钟

MCU 的时钟由系统时钟发生器（SCG）模块生成。图 4为 SCG 模块框图。

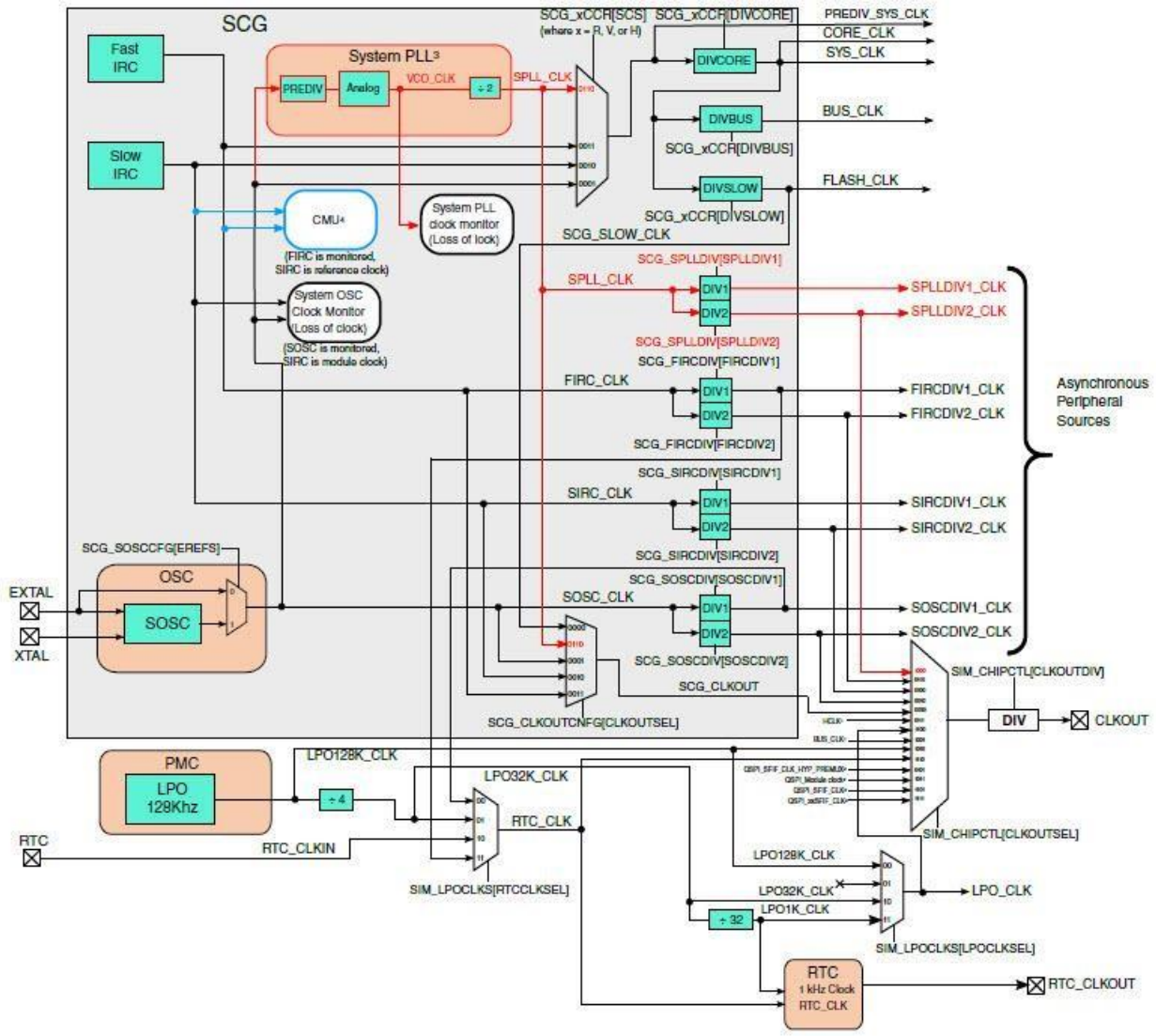

图4. 时钟图

红线标出的复用器和输入在 S32K11x 中不可用。CMU 模块（蓝线部分）仅可用于 S32K11x。

注意

系统 PLL 在 S32K11x 系列中不可用。

时钟生成电路提供多个时钟分频器和选择器，允许不同模块以该模块的特定频率计时。在 SCG 模块上可以看到四个主要时钟源（低功耗振荡器 LPO 模块除外）：快速内部参考时钟（FIRC）、慢速内部参考时钟（SIRC）、系统振荡器（SOSC）和系统 PLL（SPLL）。对于RUN模式，（HSRUN、正常RUN、VLPR）不同的源可用于向内核提供时钟信号。下图显示了用于不同功耗模式下所有可能的内核时钟源。

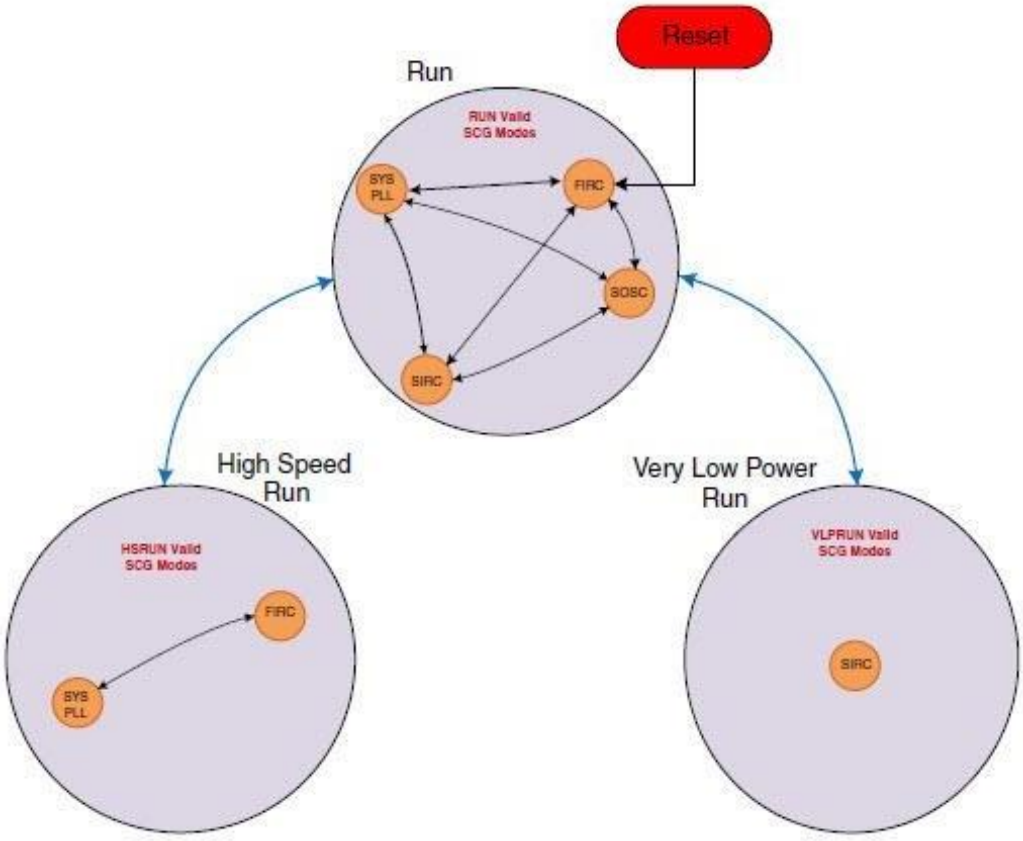

图5. SCG Core 时钟有效模式转换图

对于其他功耗模式，例如 STOP 和 VLPS，由于内核时钟被关闭，因此无需时钟源。

当进入 VLPR/VLPS 模式时，必须在 RUN 模式下由软件禁用系统 PLL 及 FIRC 后才能进行模式转换。

在切换时钟源之前，务必满足参考手册中 ***Internal clocking requirements*** 及 ***Module clocks*** 部分列出的要求。

4.2. 功耗管理控制器（PMC）

可以在电源管理控制器（PMC）模块上配置一个内部生成的典型频率为128kHz的低功耗时钟，用作在低功耗模式下RUN模块的时钟源。图4显示了低功耗振荡器（LPO）时钟源及其不同的派生。

5. 功耗模式进入/退出

当进入或退出低功耗模式时，系统必须确认一个有序的序列来确保进行安全的模式转换。

SMC 管理系统进入和退出所有的功耗模式。

5.1. HSRUN 模式进入

在 HSRUN 模式下，片上稳压器保持运行调节状态，但电压输出略有升高。在这种状态下，与正常 RUN模式相比，MCU 可以以更快的频率运行，但是当芯片处于这种模式时，任何类型的Flash命令（FTFC），包括 CSEc 命令都将不可用。

在此模式下，必须遵守以下限制：

- 禁止修改时钟门控控制位。
- 不允许进行Flash编程/擦除。

要进入 HSRUN 模式（以 112MHz SPLL 作为时钟源）：

1. 禁用 PLL. 将 FIRC 配置为RUN模式和高速RUN模式时钟源，通过将 0b0011 写入 SCG_RCCR[SCS] 和 SCG_HCCR[SCS]。
2. 设置 PMPROT[AHSRUN] 以许可高速RUN模式。
3. 将 0b11 写入 SMC_PMCTRL[RUNM] 以进入高速 RUN 模式。（系统将以 FIRC 为系统时钟源，进入高速RUN模式）。
4. 将 SPLL 重新配置为 112MHz 并启用它。
5. 通过将 SCG_HCCR[SCS] 配置为 0b0110，将时钟源切换为 PLL。

要进入 HSRUN 模式（以 80MHz SPLL 作为时钟源）：

1. 通过将 0b0110 写入 SCG_RCCR[SCS] 和 SCG_HCCR[SCS]，在RUN模式下将 SPLL 配置为 80MHz，并在 HSRUN 下使用此时钟源。
2. 配置 PMPROT[AHSRUN] 以许可 HSRUN 模式。
3. 将 0b11 写入 SMC_PMCTRL[RUNM] 以进入 HSRUN。

在增加时钟频率之前，应轮询 PMSTAT 寄存器以确定系统何时已完成进入高速 RUN 模式。下一片段代码显示了一个基本的从 RUN 到高速 RUN 的转换函数。

注意

S32K11x 系列不支持高速 RUN 模式

```
void RUN_to_HSRUN (void)
{
    /* Disable SPLL and use FIRC as MCU's source clock */
    scg_disable_spll_enable_firc();
    /* Allow high speed run mode */
    SMC->PMPROT |= SMC_PMPROT_AHSRUN_MASK;
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Move to HSRUN Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b11);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x80);
        /* Configure SPLL for 112/80 MHz */
        scg_configure_spll();
    }
}
static void scg_disable_spll_enable_firc(void)
{
    /* SCG_RCCR register only accepts 32-bit writes */
    /* Use FIRC as clock for CPU and set valid dividers */
    SCG->RCCR = (uint32_t) (SCG_RCCR_SCS(0b11) |    /* FIRC */
                            /* Core clock is 48MHz / 1 */
                            SCG_RCCR_DIVCORE(0) |
                            /* Bus clock is Core clock / 1 */
                            SCG_RCCR_DIVBUS(0) |
                            /* Flash clock is Core clock / 2 */
                            SCG_RCCR_DIVSLOW(1));
    /* SCG_HCCR register only accepts 32-bit writes */
    /* Use FIRC as clock for CPU and set valid dividers */
    SCG->HCCR = (uint32_t) (SCG_HCCR_SCS(0b11) |    /* FIRC */
                            /* Core clock is 48MHz / 1 */
                            SCG_HCCR_DIVCORE(0) |
                            /* Bus clock is Core clock / 1 */
                            SCG_HCCR_DIVBUS(0) |
                            /* Flash clock is Core clock / 2 */
                            SCG_HCCR_DIVSLOW(1));
    /* Disable PLL and clock monitors */
    SCG->SPLLCSR &= ~(SCG_SPLLCSR_SPLLEN_MASK |
                      SCG_SPLLCSR_SPLLCM_MASK);
}
```

注意

HRUN模式下，不允许Flash编程/擦除。当芯片处于此模式时，任何类型的 FTFC 命令，包括 CSE 命令（用于 CSEc 部分）均不可使用。

注意

禁止修改时钟门控制位

5.2. HSRUN 模式退出

可以通过复位或将 SMC_PMCTRL 设置为 00 以从 HSRUN 模式转换到RUN模式。由于在HSRUN模式下内核时钟可以设置为最大值（ 112MHz ）并且在RUN模式下内核时钟高达80MHz，因此可能需要在返回RUN模式之前降低核心频率。下一片段代码显示了一个基本的HSRUN到RUN转换函数。

```
void HSRUN_to_RUN (void)
{
    /* Adjust SCG settings to meet maximum frequencies value at Run mode */
    scg_configure_freq_for_RUN();
    /* Check if current mode is HSRUN mode */
    if(SMC->PMSTAT == 0x80)
    {
        /* Move to RUN Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b00);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x01);
    }
}
```

由于 HSRUN 模式允许 MCU 以最大时钟速度 RUN，务必调整 SCG 模块中的频率以满足 RUN 模式的时钟要求。可以在参考手册中的内部时钟要求部分查阅相关要求。

注意

S32K11x 系列中不支持HSRUN模式

5.3. VLPR 模式进入

在 VLPR 模式下，片上稳压器进入 Stop 模式调节状态。在这种状态下，稳压器旨在以降低的频率向 MCU 提供足够的电流。要在此模式下进一步降低功耗，需设置 PCC 寄存器中相应的时钟门控控制位禁用未使用模块的时钟。

在进入该模式前，必须满足以下条件：

- 必须禁用 SCG 中的所有时钟监视器。
- 根据允许的最大值调整时钟频率：内核，系统和总线时钟必须为 4MHz（最大值），Flash时钟必须设置为 1MHz（最大值）。此外，所有异步时钟源都将限制为 4MHz。在模式进入前，应通过软件禁用系统 PLL，系统振荡器和 FIRC。
- 模式保护必须设置为允许 VLP 模式，即 SMC_PMPROT[AVLP] 为 1。
- SMC_PMCTRL[RUNM] 必须设置为 0b10 才能进入VLPR。

下一段代码展示了一个基本的从 RUN 到 VLPR 转换的函数。

```
void RUN_to_VLPR (void)
{
    /* Disable clock monitors on SCG module */
    disable_clock_monitors();
    /* Adjust SCG settings to meet maximum frequencies value
       Disable SPLL, System Oscillator and FIRC */
    scg_configure_freq_for_VLPR();
    /* Allow very low power run mode */
    SMC->PMPROT |= SMC_PMPROT_AVLP_MASK;
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Reduce MCU power consumption in Very Low Power modes*/
        PMC->REGSC |= PMC_REGSC_BIASEN_MASK;
        /* Move to VLPR Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b10);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x04);
    }
}
```

注意

不允许Flash编程/擦除。当芯片处于此模式时，任何类型的 FTFC 命令，包括 CSE 命令（用于 CSEc 部件）均不可使用。

注意

在VLPR模式下不要增加时钟频率，因为稳压器响应缓慢且无法管理负载的快速变化。此外，请勿修改 SCG 模块或任何时钟分频器寄存器中的时钟源。PCC 中的模块时钟使能可以设置，但不能清除。

注意

为降低低功耗模式下 MCU 功耗，应设置 PMC_REGSC[BIASEN] 位。该位在低功耗模式下为内核逻辑启用源和阱偏置以优化漏电流。使用超低功耗（VLP）模式时，该位必须设置为 1。

5.4. VLPR 模式退出

要重新进入正常 RUN 模式，需清除 SMC_PMCTRL[RUNM] 位。如果需要更高的运行频率，轮询 PMSTAT 直到它被设置为 RUN，然后根据需要配置 SCG 模块。此外，复位也会导致 MCU 返回到 RUN 模式。

下一段代码显示了基本的从 VLPS 到 RUN 模式的转换函数。

```
void VLPR_to_RUN (void)
{
    /* Check if current mode is VLPR mode */
    if(SMC->PMSTAT == 0x04)
    {
        /* Move to RUN Mode*/
        SMC->PMCTRL = SMC_PMCTRL_RUNM(0b00);
        /* Wait for Transition*/
        while(SMC->PMSTAT != 0x01);
    }
}
```

5.5. STOP 模式和 VLPS 模式进入顺序

当进入 Stop/VLPS 模式时，时钟按顺序关闭，将芯片安全地置于目标低功耗状态。所有低功耗进入序列均由内核执行 WFI 指令启动。当进入低功耗模式时，芯片执行如下所示的序列：

1. CPU 时钟立即关闭
2. 请求所有非 CPU 总线主机（DMA 和 ENET，如果可用）进入 Stop 模式。
3. 在所有主机确认他们准备进入 Stop 模式后，请求所有总线从机进入 Stop 模式。
4. 在所有从机确认他们准备进入 Stop 模式后，系统和总线时钟将根据目标模式 - VLPS/STOP1/STOP2 相应关闭。请注意，在 STOP2 模式下，总线时钟不会被关闭。
5. 此外，对于 VLPS 模式，时钟发生器在 SCG 中被禁用，除非配置为启用。
6. PMC 中的片上稳压器及内部电源开关配置为满足目标低功耗模式的功耗目标。此步骤仅对 VLPS 有效，对 Stop 无效，因为在 Stop 模式下 PMC 处于运行调节状态。

注意

如果在 VLPS 模式下未启用 SIRC，则可以通过设置 PMC_REGSC[CLKBIASDIS] 位来降低功耗。使用该位时，必须确保各个时钟模块在 VLPS 模式下被禁止，否则会导致时钟模块严重故障。

5.5.1. Stop1/2 模式进入

通过在 ARM 内核的系统控制寄存器中设置 SLEEPDEEP 位的立即睡眠或退出睡眠模式，可进入 STOP1/2 模式。

SCG 模块可以配置为保持参考时钟运行状态。对于 STOP 模式，所有 SCG 时钟都可用（LPO、PLL、SIRC、FIRC 和 OSC）。有关更多详细信息，请查看 ***Modules in power modes***。

SMC_STOPCTRL[STOPO] 位选择将 MCU 发送至 STOP1 (0b01) 还是 STOP2 (0b10) 模式。下一段代码显示了基本的RUN到 STOP1/STOP2 转换函数。

```
void RUN_to_STOP (void)
{
    /* Enable SLEEPDEEP bit in the Core
     * (Allow deep sleep modes) */
    S32_SCB ->SCR|=FSL_SCB_SCR_SLEEPDEEP_MASK;
    /* Select Stop Mode */
    SMC->PMCTRL=SMC_PMCTRL_STOPM(0b00);
    /* Select which STOP mode (Stop1 or Stop2)
     * is desired (Stop1 - 0b01, Stop2 - 0b10) */
    SMC->STOPCTRL=SMC_STOPCTRL_STOPO(0b01);
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Go to deep sleep mode */
        asm("WFI");
    }
}
```

5.5.2. VLPS 模式进入

下面列出了进入 VLPS 模式的两种方式：

- 当 MCU 处于VLPR模式且PMCTRL[STOPM] = 0b010 或 0b000时，通过在系统控制寄存器中设置SLEEPDEEP位的立即睡眠Sleep-now或退出时睡眠 Sleep-on-exit 以进入VLPS模式。
- 当 MCU 处于正常RUN模式且PMCTRL[STOPM] = 0b010时，通过在Arm内核的系统控制寄存器中设置SLEEPDEEP位的Sleep-now或Sleep-on-exit以进入VLPS状态。当直接从RUN模式进入VLPS时，退出到VLPR将被硬件禁用，系统将始终退回至RUN式。

除了在 SMC_PMPROT 寄存器中允许极低功耗模式之外，转换到 VLPS 几乎与 Stop1/2 模式相同。在进入 VLPS 模式之前，务必禁用系统 PLL、系统振荡器和 FIRC[1]。下一段代码显示了基本的 RUN 到 VLPS 转换函数。

[1] 您可以阅览参考手册中**系统时钟转换**部分以获取详细信息。

```
void RUN_to_VLPS (void)
{
    /* Adjust SCG settings to meet maximum
frequencies value */
    scg_disable_pll_and_firc();
    /* Enable SLEEPDEEP bit in the Core
     * (Allow deep sleep modes) */
    S32_SCB ->SCR|=FSL_SCB_SCR_SLEEPDEEP_MASK;
    /* Allow very low power run mode */
    SMC->PMPROT |= SMC_PMPROT_AVLP_MASK;
    /* Select VLPS Mode */
    SMC->PMCTRL=SMC_PMCTRL_STOPM(0b10);
    PMC->REGSC |= PMC_REGSC_BIASEN_MASK;
    /* Check if current mode is RUN mode */
    if(SMC->PMSTAT == 0x01)
    {
        /* Go to deep sleep mode */
        asm("WFI");
    }
}
```

注意

为降低低功耗模式下的 MCU 功耗，应设置 PMC_REGSC[BIASEN] 位。该位在低功耗模式下为内核逻辑启用源和阱偏置。使用超低功耗（VLP）模式时，该位必须设置为1。

5.6. STOP 及 VLPS 模式退出顺序

复位或中断事件触发从低功耗 Stop 模式退出。然后执行以下序列以将系统恢复到 RUN 模式（RUN 或 VLPR）：

1. PMC 中的片上稳压器、时钟发生器和内部电源开关已恢复。此步骤仅对 VLPS 有效，对 STOP 无效，因为在 STOP 模式下 PMC 处于运行调节状态。
2. 系统和总线时钟对所有主机和从机启用。
3. CPU 时钟使能，CPU 开始响应此前触发低功耗 Stop 模式退出的复位或中断事件。

6. 功耗模式中各模块

参考手册中 **Module operation in available low power modes** 节列出了所有外设及其在不同低功耗模式下的可用性。

7. 软硬件注意事项

为降低 MCU 功耗，可以遵循以下指示：

7.1. 硬件注意事项

应用中所有未使用的引脚（尤其是模拟功能）应遵循一些建议以避免可能的电流消耗的增加：

- DAC 输出引脚应悬空。
- ADC 引脚应接地。

7.2. 软件注意事项

为了节省功耗，可以通过配置 PCC 模块中外设控制寄存器的 CGC 字段来关闭大多数模块的时钟。这些字段在任何复位后都会被清除，从而禁用相应模块的外设时钟。务必在关闭时钟之前先禁用对应模块。

一些外设支持打盹休眠模式。在此模式下，可使用寄存器字段在低功耗模式期间禁用外设。

7.3. 在台架上进行低功耗测量的技巧

7.3.1. 外部

在尝试复现数据手册电流规格时遇到的常见问题，请参照下列建议解决。

- ***使用数字万用表时（DMM），使用“手动范围模式”。***使用自动测量功能的数字万用表可能会导致 LVD 和 POR 复位。这在您从一种低功耗模式（如 LLS 或 VLPS）退出并返回RUN 时经常发生。数字万用表已将量程更改为微安或纳安量程，而 MCU 处于低功耗模式切换到RUN模式，突发的浪涌电流需要数字万用表更改量程。如果万用表的量程范围变化的速度不够快，MCU处于馈电状态会将 VDD 电平拉低至 LVD 或 POR 限值以下。
- ***断开调试器并对MCU重新上电。***连接 JTAG 调试器后，MCU 可能会使 MCU 中的调试器模块处于活动状态、计时并消耗功率。外部调试器硬件也可以在连接时加载 JTAG 端口的 I/O。因此，低功耗测量值将高于预期。
- ***隔离 MCU VDDs。***想要测量 MCU 的电流消耗，需移除与 MCU 同源供电的其他 IC 和网络。例如，部分 EVB 在 MCU_VDD 和地之间连接了一个电位器。3.6V 电源上的 5K 电位器可拉动 720μA。该值是很大的，考虑到 MCU 在最低功耗模式下消耗只有大约几十μA。
- ***匹配输入阻抗。***如果高速信号（快速边沿转换）的阻抗没有很好的匹配，则信号可能有“振铃效应”，导致超出器件的 VDD 电源范围。这会导致信号通过输入保护二极管向设备注入电流。对于高速输入时钟尤其如此。在最低功耗模式下，此问题会导致测量中产生负 IDD影响最终的测量值。

- ***匹配电平。***尽管 MCU 输入引脚在某些型号具有 5 V 容限，但当 MCU 进入低功耗模式时，任何高于 MCU_VDD 的输入会影响通过 MCU_VDD 的电流测量。较高的输入引脚将通过输入引脚为 MCU 反向供电，从而导致低功耗模式下的负 IDD 读数。
- ***减少 MCU 引脚负载。***当 MCU 通过输出引脚提供电流时，电源通过 MCU_VDD 提供。当您将高频信号输出到输出引脚时最为明显，比如使用外部存储器接口（例如时钟和地址/数据引脚）。

7.3.2. 内部

下面列出了最常见的问题，这些问题可能会让您无法获得数据手册中的电流最低值。

- ***看门狗未禁用，***导致复位。禁用或维护看门狗。
- ***时钟监视器未禁用，可能会导致复位。***禁用所有时钟监视器。
- ***在低功耗模式下启用晶体振荡器。***RTC 振荡器通常消耗小于500nA 的电流.
- ***CLKOUT 信号正在输出到某个引脚。***任何状态持续不断变化的引脚都会消耗功率。
- ***PMPROT 寄存器中的相应位未允许当前请求的低功耗模式。***例如，如果在 PMPROT 中没有设置 AVLP 并且执行了 WFI 指令，将不会进入 Stop 模式。
- ***在读取或写入模块之前未启用模块的时钟。***这会在 MCU 尝试进入低功耗模式之前导致复位。
- ***模块的时钟在进入低功耗模式之前已被关闭，但是此模块模式进入切换过程中仍然需要使用。***这将导致 Stop 模式确认复位。
- ***调试完成后错误注释掉对 STOP 或 SLEEP 函数的调用***会使您处于运行电流较高。
- ***唤醒操作的频率过高，***这意味着 MCU 处于RUN或 VLPR 模式的时间比处于低功耗模式的时间多。从低功耗模式到RUN模式的转换时间很快。如果 MCU 仅运行RUN 9ms，低功耗模式 1ms，则系统的平均电流将远高于 MCU 每秒仅运行 RUN 1ms 的情况。
- ***MCU 运作频率远高于完成工作所需频率。***使用 SCG 分频器限制时钟或减少时钟。显然，MCU 频率越高，MCU 的 IDD 就越高。减少时钟并降低 RUN 或等待电流。但有待权衡之处。如果可以容忍RUN模式的电流，那么尽快完成工作并立即返回低功耗模式比在 RUN 模式下运行较慢的时钟更有优势。

- ***无内部或外部上拉装置的某输入引脚悬空。***这会导致每个引脚的电流为 50-80uA。包括 JTAG 或 SWD 引脚。禁用 PORTA 上的 JTAG 引脚或正确地处理这些输入引脚。

7.4. 当前功耗测量

由于电流消耗取决于不同的因素，例如启用了哪些模块、为内核和其他外围设备供电的频率、温度条件等，MCU 的数据表包含不同功率模式的测试用例，因此，对于电流消耗测量，您可以参考 MCU 的数据手册。

8. 功耗模式使用示例

下一章节展示了基于 S32DS 2018.R1 版本的一些使用示例及其代码。

8.1. 功耗模式切换以在系统上实现 100uA（VLPS + RUN 模式）

设计系统架构时的主要关注点之一是实现最低功耗，为此，不仅定义最低功耗模式很重要，定义所涉及的功耗模式下的主要操作也很重要，例如，ADC 读数、通信传输、I/O 控制信号等。一旦功耗模式确定，其他一些重要因素如操作频率、唤醒间隔、模式之间的切换时间等也变得很非常重要。

以下用例实现了 VLPS（最低功耗模式）和 RUN 模式之间的功耗模式转换，使用 ADC 读数和和 SPI 传输等功能，实现系统功耗大约 100uA。

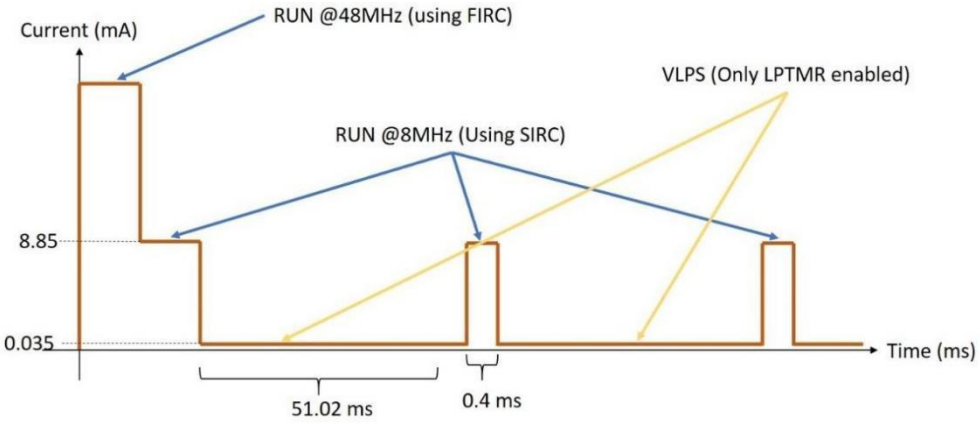

图6. 电源模式转换图

该实例是在 S32K148EVB (S32K148_PowerManagement_VLPS_RUN) 上实现的，并包含以下操作。

- 在RUN模式下：
  - 禁用 WDOG 以及其他未使用模块的时钟（例如：MPU时钟，DMA，ERM等）。
  - 禁用 SOSC 和 SPLL 时钟监视器。
  - 更改时钟配置。默认情况下，使用FIRC并且设备以 48MHz 运行，因此需要更改为 SIRC 作为时钟源（8MHz），再在执行任何功耗模式转换之前通过软件禁用 FIRC。此外，配置 SIRC_DIV2 信号，为RUN模式下的 SPI 和 AD 提供时钟源。
  - 配置 LPTMR。该定时器将于每 51.42ms 将系统从VLPS唤醒。LPTMR 使用 LPO 时钟，它在 VLPS 中保持活动状态。
  - 初始化 SPI 和 ADC 模块，转换到 VLPS 时无需禁用这些模块，因为其使用的时钟源在 VLPS 中不可用（SIRC 在 VLPS 未中启用）。请注意满足器件参考手册 ***Clock definitions*** 部分对 SIRC_DIV2 的要求，因为该信号用于计算这些 SPI 和 ADC 模块的工作频率。
  - 禁用调试器引脚以降低功耗。
  - 在切换到 VLPS 前，读取 SPI 传输及 ADC 通道。
- 在VLPS模式下：
  - 由于在此模式下除了 LPTMR 之外什么都不启用，除了等待超时 LPTMR 不会在VLPS中进行任何操作。这段时间的功耗达到~35uA。

整个系统的平均电流计算如下：

$$I_{system} = \frac{I_{RUN} \cdot t_{RUN} + I_{VLPS} \cdot t_{VLPS}}{t_{total}}$$

基于上图的值：

$$I_{system} = \frac{8\,850\ \mu A \cdot 0.4 ms + 35 \mu A \cdot 51.02\ ms}{51.42\ ms} = 103.57 uA$$

下图展示了从数字万用表的测量结果。

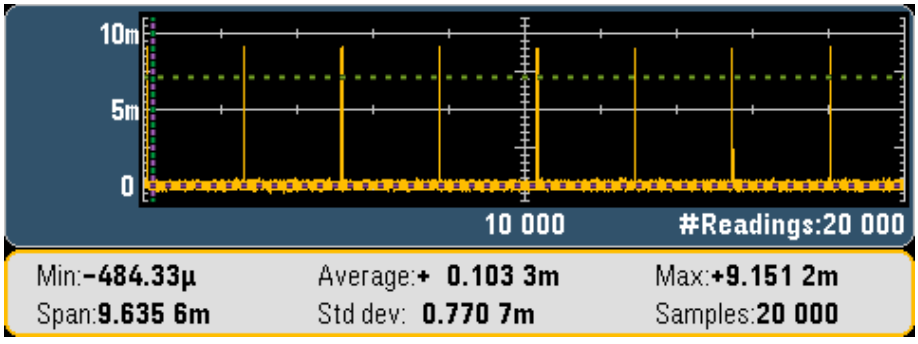

图7. 内核以 8MHz 运行时的功耗（平均为 103.3uA）

除此之外，用户可以改变运行频率、模块的设置等，以确定系统电流如何受到影响。

对于该实例，我们进行了第二次测试，将内核频率设置为 4MHz（预计最大电流可以减少，但在 RUN 中花费的时间会更长）以验证是否可以减少系统电流。下图显示了我们的结果。

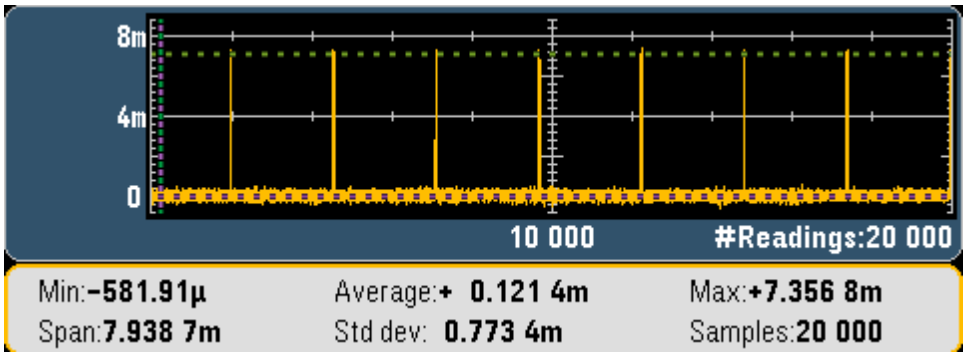

图8. 内核以 4MHz 运行时的功耗（平均为 121uA）

如上图所示，使用较低的内核频率后系统的平均电流增加，因此 RUN 中的最大电流与 RUN 花费的时间之间的关系决定了系统的总电流消耗。

$$I_{system} = \frac{7.12\ mA \cdot 0.63\ ms + 35\mu A \cdot 50.79\ ms}{51.42\ ms} = 121.8\ uA$$

综上所述，用户在设计整个系统时不仅需要考虑最低功耗模式，还需要考虑“唤醒”模式的工作频率、所用模块的配置、唤醒信号的周期性等，以实现所需的电流值。

8.2. 不同模式切换的时钟注意事项

在某些情况下，可能需要在具有不同时钟配置的功耗模式之间切换，例如切换到HSRUN模式以使用设备上的最大运行速度，切换到 RUN 模式以使用 HSRUN 中某些不支持的功能。另一种选择是，当需要低功耗时，考虑到并非所有外设在VLPR中均可启用，我们可以在RUN和VLPR之间模式切换，因此切换到 RUN 模式也是需要的。

由于各种模式的内核、总线和Flash分频器有不同的复用（SCG_RCCR用于RUN，SCG_HCCR用于HSRUN以及SCG_VCCR用于VLPR），因此在不同的功耗模式和时钟源之间切换应该很容易，但是，为了满足每种模式的要求，部分时钟仍需要考量一番。如下展示了两个实例。

8.2.1. 设备以最大速度RUN时在 HSRUN 与 RUN 切换（仅针对 S32K14x）以使用 CSEc/EEEPROM 功能

由于 HSRUN 模式下不提供编程/擦除 Flash 和 CSEc/EEEPROM 等功能，用户必须在HSRUN和RUN模式之间切换，在 HSRUN 模式下以最大速度运行设备，然后在RUN模式下按需执行某些操作（CSEc/EEEPROM）。

使用 SPLL 作为主时钟源的 112MHz HSRUN 模式到使用 FIRC 作为主时钟源的 48MHz RUN 模式的切换可以使用下表所示的不同 SCG_xCCR 寄存器设置轻松完成。

寄存器	系统时钟源 (SCS)	核心时钟分频器 (DIVCORE)	总线时钟分频器 (DIVBUS)	Flash时钟分频器 (DIVSLOW)
SCG_HCCR	6 (SPLL)	0 (Divide by 1)	1 (Divide by 2)	3 (Divide by 4)
SCG_RCCR	3 (FIRC)	0 (Divide by 1)	0 (Divide by 1)	1 (Divide by 2)

对于 HSRUN 模式，内核时钟为 112MHz，总线时钟为 56MHz，Flash时钟为 28MHz。

对于 RUN 模式，内核时钟为 48MHz，总线时钟为 48MHz，Flash时钟为 24MHz。

按配置，如果不使用异步外设时钟源（SPLL_DIV1和SPLL_DIV2），用户可以轻松的在 HSRUN 和 RUN 之间切换，否则，必须考虑其他因素。

根据参考手册中 ***Clock definitions*** 部分，SPLL_DIV1 在RUN模式下可以配置为 80MHz 或更低，在 HSRUN 模式下可以配置为 112MHz 或更低。同样，SPLL_DIV2 在 RUN 模式下可配置为 40MHz 或更低，在 HSRUN 模式下可配置为 56MHz 或更低，除此之外，每当外设时钟源或其分频器发生变化时，应禁用相应的模块。为避免这种情况，建议在满足 RUN 和 HSRUN 要求的两个信号中设置固定频率，以避免在发生功耗模式切换时禁用外设（例如，由 FTM 驱动的 PWM 信号在电源模式转换发生时，可能不需要暂停使用 SPLL_DIV1）。下图显示了每个源的最大频率，以便于 HSRUN 和 RUN 之间的功耗转换，而无需在功率模式转换之前更改分频器/时钟源。

尽管 SPLLDV1_CLK 可以被配置为 56MHz（未超过最大允许频率），但 FTM 内有要求 FTM 功能时钟不能超过 (SPLLDVI1_CLK) ¼ 功能时钟的限制。（时钟馈送 FTM 逻辑，有 SYS_CLK 驱使）。因此 SPLLDIV1 固定为 14MHz (56MHz / 4 = 14MHz)。

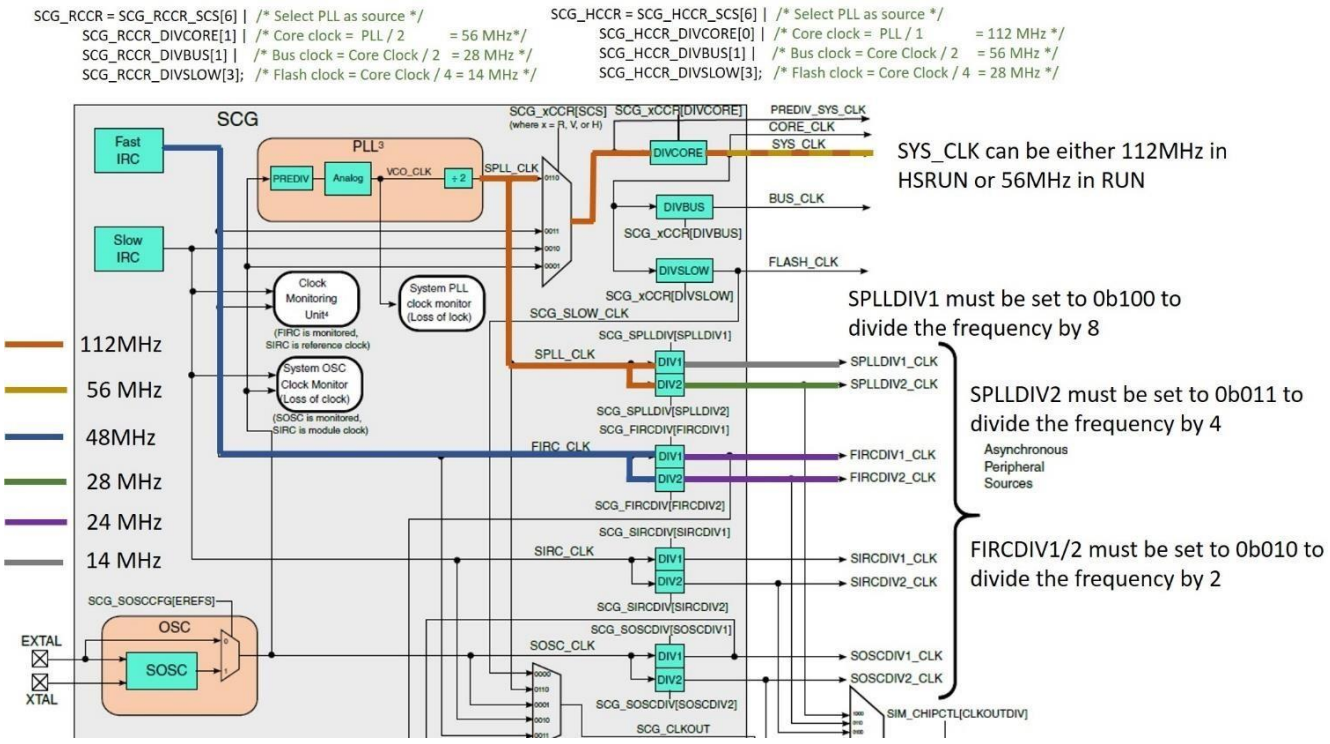

图9. HSRUN/RUN 切换方案的时钟图

对于上图，不需要为切换过程而禁用/挂起绑定到 SPLL_DIV1 和 SPLL_DIV2 的所有外设，因为在每次转换时这些信号均保持相同的值，另一方面，链接到 SYS_CLK 和 BUS_CLK 的外设可能需要在切换过程之前禁用，并在转换到所需功耗模式完成后再次启用外设。

注意

尽管 SCG_xCCR[DIVBUS] 和 SCG_xCCR[DIVSLOW] 保持不变，但 DIVCORE 的变化会影响总线和Flash时钟的频率值，因此需要在切换事件之前禁用使用这些时钟源的外设。

参考手册中的表 27-9 列出了所有外设及其可用的时钟源。确保在任何转换之前禁用所有与 SYS_CLK/BUS_CLK 相关的外设（特别是所有与时间相关的定时器/通信外设）。

注意

尽管 FIRC_DIV1 和 FIRC_DIV2 信号可以配置为 48MHz，但大多数能够使用这些信号的外设都会限制外设的允许频率，因为这由 BUS_CLK/SYS_CLK 管理。在参考手册中的 ***外设时钟摘要*** 中有相关描述。

考虑到所有这些因素后，设备可以在 HSRUN 上运行以获得高速性能，并且在需要时切换到 RUN 以执行任何 CSEc/EEEPROM 功能。这些都完成后，设备可以返回到 HSRUN。

示例项目在 S32K142EVB 上进行了测试，它实现了：

1. 配置 SPLL 时钟（SPLL_CLK 为 112MHz，SPLL_DIV1 为 14MHz 以及 SPLL_DIV2 为 28MHz），FIRC（FIRC_DIV1 和 FIRC_DIV2 为 24MHz）及 RUN 与 HSRUN 模式下的分频器。
2. 配置引脚为用作参考的 GPIO 信号，并且 配置 PTB5 为 CLKOUT 引脚，用来对外显示 SYS_CLK 信号除以 8 的时钟信号。
3. 调试控制台通过 LPUART1 模块向终端发送消息。
4. CSEc 模块配置完毕。为 ECB 操作加载 RAM 密钥。
5. FTM 设置以生成频率为 10kHz、50% 占空比的 4 个 PWM 信号。
6. LPTMR 用以每 500ms 产生一个周期性中断信号。该中断将用于切换到 RUN 模式以执行 CSEc 操作。
7. PDB0 配置为当器件使用 112MHz 时钟（HSRUN）时每 1ms 提供一次中断，当器件使用 56MHz 时钟（RUN 模式）时每 2ms 提供一次中断。PDB 的配置是固定的（两种情况的模量值都设置为 360）。
8. 转换到 HSRUN 模式。MCU 将保持在 HSRUN 状态，直到收到来自 LPTMR 的中断。
9. 中断到达时，转换到 RUN 模式。转换前，使用 SYS_CLK 和 BUS_CLK 的模块被禁用，转换完成后，这些模块再次启用（FTM 未修改，因为它使用 SPLL_DIV1）。
10. 一旦进入 RUN 模式，就会完成两次 ECB 操作。完成后，将执行到 HSRUN 的转换。同样，使用 SYS_CLK 和 BUS_CLK 的模块在转换完成之前被禁用，一旦转换完成，它们将再次启用（FTM 未修改，因为它使用 SPLL_DIV1）。

下图显示了此用例的时序图。

- 通道1（黄色信号）显示 FTM0_CH0 生成 10kHz PWM 信号输出。即使在功耗模式转换中，PWM 输出也保持不变。
- 通道2（绿色信号）显示功耗模式转换。
- 通道3（蓝色信号）连接到位于 PTB5 上的 CLKOUT 信号。CLKOUT映射为 HCLK (SYS_CLK) 的8分频时钟，因此在 HSRUN 中输出为 112MHz/8 = 14MHz，在 RUN 模式中为 56MHz/8 = 7MHz。
- 通道4（粉红色信号）连接到 PDB0 用以回调翻转引脚上的电平。可以看到信号在每次功耗模式转换时是如何改变其周期。

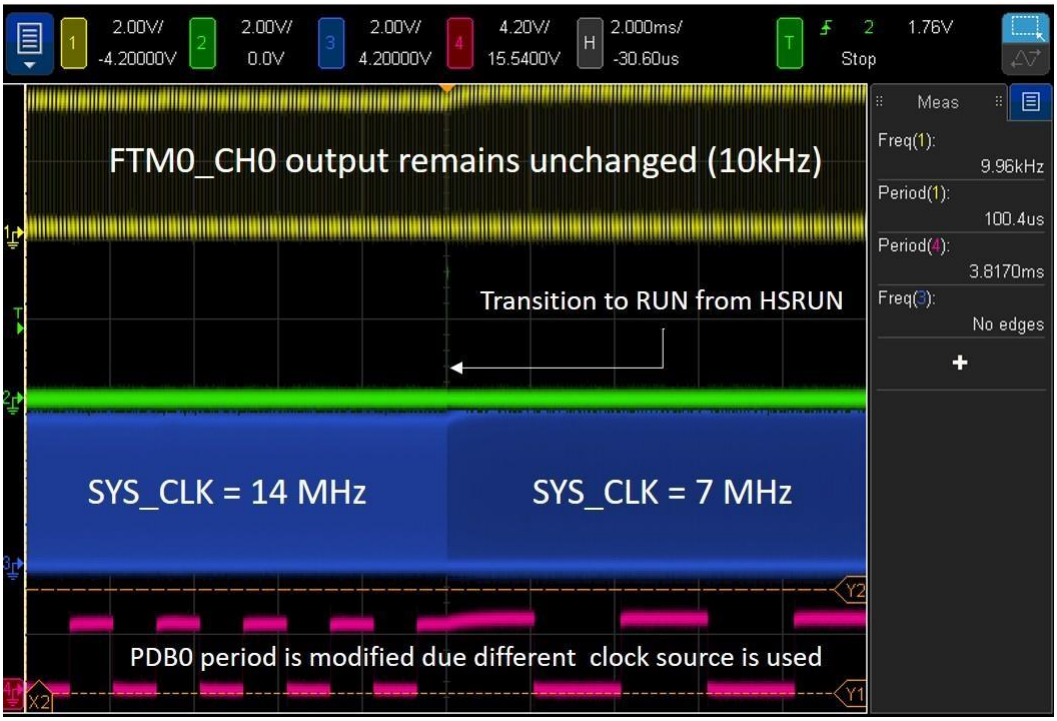

图10. HSRUN 到 RUN 转换上的时钟信号

CSEc 操作完成后，设备将再次以最大速度运行。下图显示了转换到 RUN 模式之后、期间和之前的时钟信号。

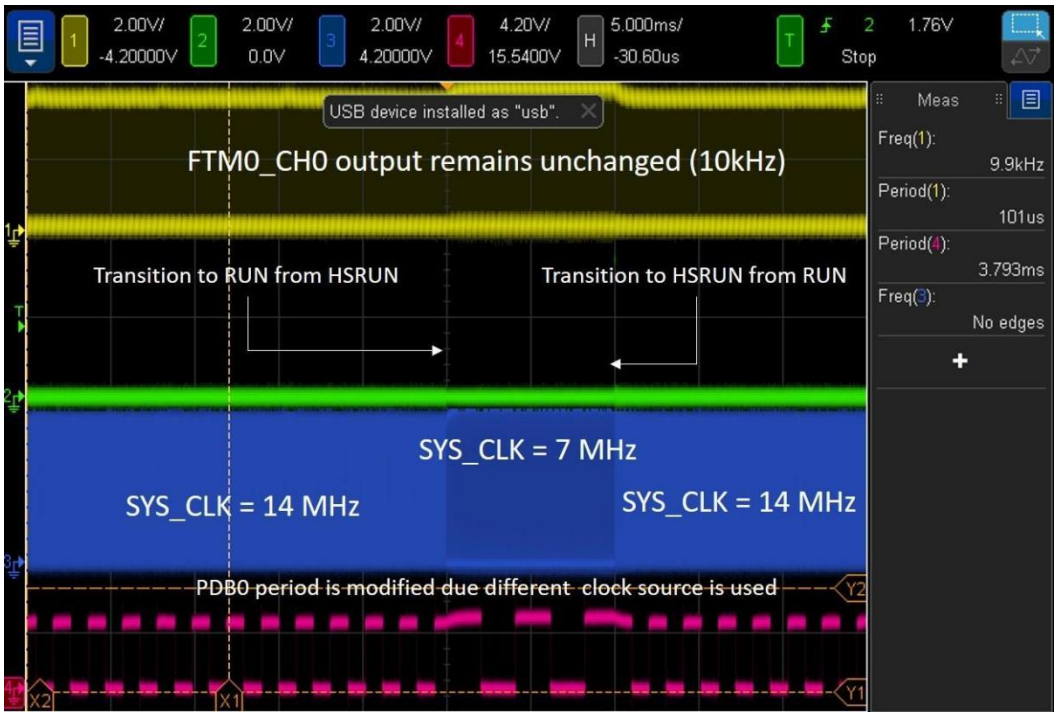

图11. 电源模式转换时的时钟信号

8.3. MCU 休眠时的传输管理：S32K11x VLPS + DMA + LPUART (LIN)

当MCU处于休眠状态时，最常见的通信传输方法是唤醒系统以发送/接收数据，然后返回休眠模式。S32K设备的一些特性允许系统在设备保持低功耗模式且MCU的时钟门控关闭（VLPS）时仍然能发送/接收数据。要实现这样功能需要使用DMA，因为它即使在VLPS模式下中仍然保持完整功能。请记住，在功耗模式进入期间发生的任何唤醒事件（例如，DMA完成）都可能导致系统挂起（勘误表 e11063）。

下一实例展示了LPUART + DMA来触发一次传输的用法，此时设备保持在VLPS，并且只有在所有数据都传输完毕后，MCU 才能被唤醒。

对于此实现，DMA 配置为将数据从SRAM移动到 LPUART_DATA 寄存器。由于需要异步操作，因此通过DMA_ERQ和DMA_EARS 启用 DMA 请求。一旦当前 TCD 完成后，硬件请求将被关闭 (DMA_TCD_CSR[DREQ] = 1) 以停止向LPUART发送更多数据并开始处理唤醒系统的DMA中断。

注意，一旦启用 DMA 请求，就会触发向 VLPS 的模式转换，如参考手册中所述，虽然转换已成功完成，但仍可能会设置 SMC_PMCTRL[VLPSA]。

同样，此方法也适用于VLPS中可用的任何其他通信模块，如 I2C，SPI 等。

示例代码是在 S32K116EVB 中创建和测试的。下图显示了 MCU 保持在 VLPS/VLPR 模式时在 LPUART_TX 引脚中看到的数据。

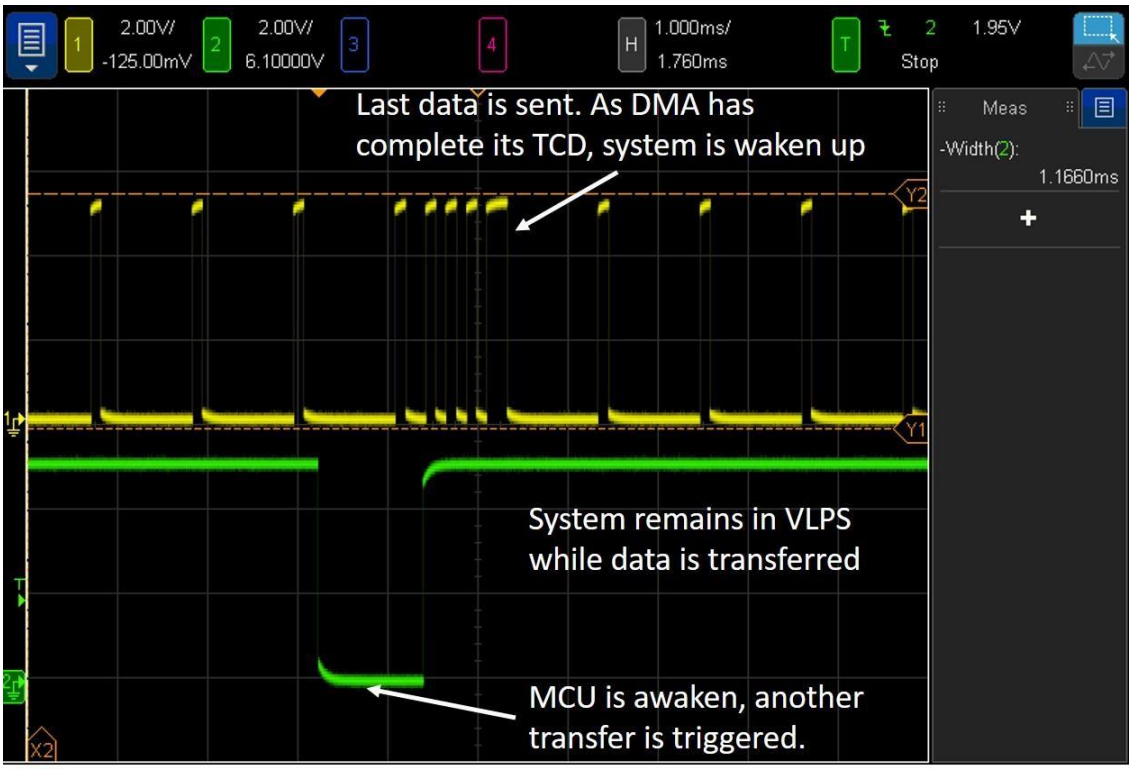

图12. MCU 保持在 VLPS 时的 LPUART 数据

因为 DMA 可以在 VLPS 保持激活状态，数据可以通过 LPUART 接口传输。一旦完成，在 TX 引脚上物理看到最后一个数据之前，MCU 被唤醒。在此实例中，LPUART 加载了相同的要传输的数据，并且过程不断重复。下图显示了此例的功耗。这里的一个重要区别是，如果 MCU 保持在 VLPS 中的同时发送更多数据，则电流消耗会降低。

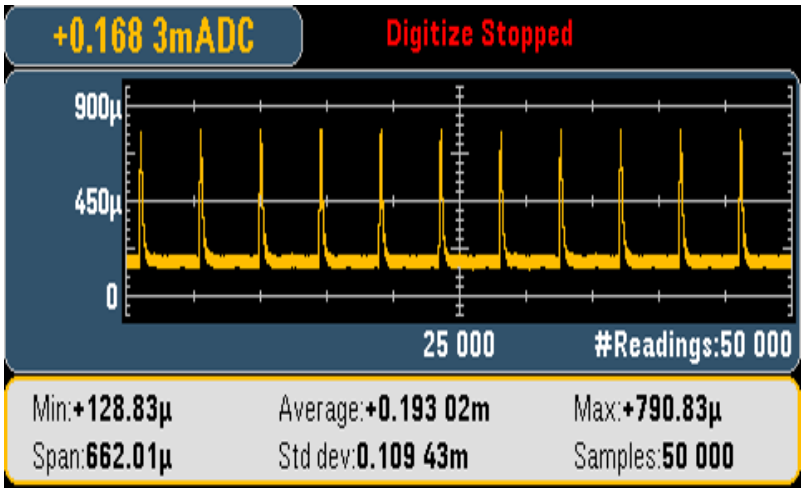

图13. MCU 保持在 VLPS 时的 LPUART 数据

9. 修订历史

修订版本号	日期	重大变更
REV 0	2017年3月	初版发布
REV 1	2018年4月	• 包含了对 S32K11x 的支持。 • 更新了 figures 4 及 5 以包含 S32K11x 参考。 • 增加了 HSRUN 和 VLPS 进入额外信息。 • 移除了 Modules in power modes 部分中的表格。 • 使用实例中增加了 Power modes use cases。

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number:AN5425
Rev. 1
05/2018