Freescale Semiconductor
应用笔记

Document Number: AN4526
Rev 1.0, March 2013

Kinetis 100 MHz 版本 1.x 到 120 MHz 移植指南

1 目的与概述

本文档详细描述了从 Kinetis 100 MHz 版本 1.x 微控制器向 Kinetis 120/150 MHz 微控制器的移植。在同一系列中的两款器件之间移植可能需要对硬件和/或软件进行一定的修改。本文档描述了可能需要的一些修改。

内容

1.1 部件编号和掩膜组信息

下表列出了本文档撰写之时在产的所有 Kinetis 100 MHz 版本 1.x 掩膜组。

表 1. Kinetis 100 MHz 版本 1.x 掩膜组

版本	掩膜组	部件编号示例
1.0	0M33Z	PK10N512VMD100
1.1	0N30D	N/A
1.2	1N30D/2N30D（功能相同）	PK10N512VMD100
1.4	4N30D	MK10DN512ZVMD10（“Z”字符：初始生产掩膜组）

下表列出了在本文档撰写之时在产的 Kinetis 120 和 150 MHz 掩膜组。

表 2. Kinetis 120 和 150 MHz 掩膜组

版本	掩膜组	部件编号示例
1.0	0N96B	PK60FN1M0VMJ12
1.1	1N96B	PK60FN1M0VMJ12
1.3	3N96B	MK60FN1M0VMJ12

本文档主要关注相同 Kinetis 系列中 100 MHz 与 120/150MHz 器件之间的移植。例如，如果您从 MK60DN512VMD10 向 PK60FN1M0VMD12 移植，本文档将很有帮助；如果您从 K10 向 K60 器件移植，本文档仍能提供一些有用信息，但是不会着重说明不同 Kinetis 系列之间的功能变化。

1.2 关于本文档

本文档描述了 Kinetis 100 MHz 与 Kinetis 120/150 MHz 器件之间的移植。120 和 150 MHz 器件是同一款器件的不同速度升级选件。除了内核频率及其导致的性能提升外，两种速度升级版本在功能或特性方面并无差异。为方便起见，本文档将这些器件称为 Kinetis 120 MHz 器件。此信息对于 150 MHz 器件同样适用，但本文不会特别说明。

本文档分为 5 个主要章节——新模块、更新模块、增强型模块、具有额外实例化的模块，以及未改变的模块。

新模块章节将快速介绍 120 MHz 器件的全新模块。100 MHz 微控制器中不包含与这些模块功能相同的模块。如果您的应用中要使用这些新模块，那么您将需要修改软件来使用这些新模块。大多数模块还需要修改硬件才能使用 (LMEM 和 FPU 除外)。

更新模块章节概述了使用新版本的更新模块。这些模块的总体功能将类似，不过需要修改软件。可能需要修改硬件以使用新特性。

增强型模块章节概述了存在细微改动的现有模块。可能需要修改软件与硬件以使用新特性。

具有额外实例化的模块章节描述了那些本身没有变化的模块，但是在微控制器中包含了更多该模块的实例。

未改变的模块章节列出了 100 MHz 与 120 MHz 器件中的相同模块。

此外，全文使用了颜色代码机制：

- 绿色：指示新增内容；
- 黄色：指示修改内容；
- 红色：指示删除内容

2 新模块

Kinetis 120 MHz 平台增加了多个 100 MHz 器件所不具备的模块。以下章节将对这些模块的特性进行概述。想要利用这些新模块的应用将需要对软件进行修改，在某些情况下还需要修改硬件才能使用这些新功能。

这些模块的可用性取决于您所使用的具体 Kinetis 器件。如需了解可用特性，请参见您所使用的 Kinetis 器件的参考手册。

2.1 NAND Flash 控制器（NFC）

NAND Flash 控制器（NFC）与标准 NAND Flash 存储器器件相连接。它由多个控制逻辑单元和一个 9 KB SRAM 缓冲器组成。NFC 提供与 8 和 16 位 NAND Flash 器件（具有 512 字节、2 KB、4 KB 和 8 KB 分页大小）的无缝连接。

2.2 DDR 存储器控制器（DDRMC）

与外部 SDRAM 连接，用于存储和检索数据。支持与 16 位或 8 位 DDR1、DDR2 或 LPDDR SDRAM 存储器的无缝连接。

2.3 USB 高速 OTG 控制器（USBHS）

USB 高速 OTG 控制器（USBHS）是符合 USB 2.0 标准的串行接口引擎，用于实现 USB 接口。其寄存器和数据结构基于 Intel 公司用于通用串行总线的增强型主机控制接口规范（EHCI）。USBHS 模块可作为 USB 总线上的主机、设备或 On-The-Go 可协商主机/设备。USBHS 控制器与处理器内核相连接。可对控制器编程以支持在固件控制下的主机或设备操作。要运行 USB HS OTG 控制器，将 USB HS 控制器通过 ULPI 接口连接到外部 ULPI PHY/收发器。

2.4 图形 LCD 控制器（LCDC）

为外部灰阶或彩色 LCD 面板提供显示数据。它支持黑白、灰阶、无源矩阵型彩色（CSTN）和有源矩阵型彩色（TFT）LCD 面板。支持的最大面板分辨率为 800x600，最高支持 24 bpp 色。

2.5 MCU DryIce

MCU DryIce 模块包含一个 32 字节安全存储器，用于在执行任何篡改检测时进行异步擦除。此外，它还可用于强制系统复位和/或使实时时钟无效。

2.6 本地存储器控制器（LMEM）

本地存储器控制器是系统 RAM 控制器（用于所有 Kinetis 器件）和系统高速缓存控制器（120 MHz Kinetis 器件的新功能）的组合。LMEM 可以使多个主外设及内核同时对系统 RAM 进行访问，并控制高速缓存，通过对指令和数据流水线提供单周期访问来提高系统性能。

2.7 浮点运算单元（FPU）

Kinetis 120 MHz 器件增加了 Cortex M4 内核的可选浮点运算单元。该单精度 FPU 符合 *IEEE* 浮点算法标准（IEEE 754）。

3 更新模块

3.1 Flash 存储器模块（FTFL 到 FTFE）

为了在 120 MHz Kinetis 器件上支持更大容量的 flash，修改了 flash 模块。100 MHz 器件使用 FTFL 模块，支持最大 512 KB 的 flash；而 120 MHz 器件使用 FTFE 模块，支持最大 1 MB 的 flash。

除了支持更大的 flash 阵列，120 MHz 器件还支持更大的 FlexRAM。100 MHz 器件具有最大 4 KB 的 FlexRAM，而 120 MHz 器件具有最大 16 KB 的 FlexRAM。

3.1.1 存储空间映射比较

FTFE 和 FTFL 使用不同的模块缩略词，但它们的存储空间映射保留一致。例如，在 100 MHz 器件上有 FTFL_FSTAT 寄存器。在 120 MHz 器件上，FTFL_FSTAT 寄存器变为 FTFE_FSTAT 寄存器。所有的寄存器地址及字段均相同，只是前面的缩略词变了。

3.1.2 软件影响

最大的变化是，当从 FTFL 模块转移到 FTFE 模块时，flash 的编程大小发生了改变。FTFL 模块使用长字编程命令(FCMD = 0x6)，一次可编程 32 位；而 FTFE 模块使用 64 位编程命令(FCMD = 0x7)，一次可编程 64 位。任何修改 flash 内容的应用都将需要修改为使用新的编程大小。

另一个变化是增加了 flash 保护区域的默认大小。例如，FTFE 和 FTFL 上均有 32 个 P-flash 保护区域，其中 P-flash 的总大小平均分成 32 个区域。对于 FTFE 器件，它具有两倍大的 P-flash，因而 P-flash 保护区域的大小也翻倍。D-flash 和 EEPROM 保护区域的大小也相应增加。

3.1.3 硬件影响

无硬件影响。

3.2 电源管理控制器（PMC）

电源管理控制器（PMC）的主要功能是：控制 POR 和 LVD 电路，以及向 MCU 提供电压和电流。对于 Kinetis 120 MHz 器件，I/O 保持控制功能现在位于 PMC 中。PMC 的功能与低漏电唤醒单元（LLWU）、系统模式控制器（SMC）和新的复位控制模块（RCM）紧密联系在一起。

3.2.1 存储空间映射比较

寄存器映射和名称保留一致。下图显示了存储空间映射中仅有的位级变化。

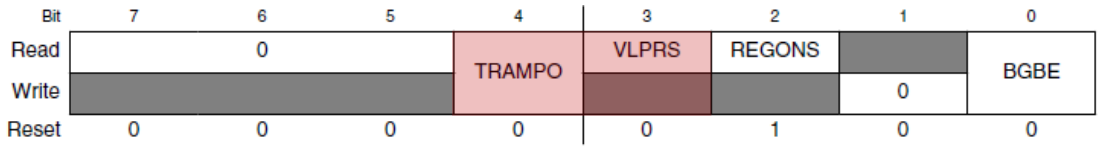


图 1. PMC_REGSC—Kinetis 100 MHz 版本 1.x

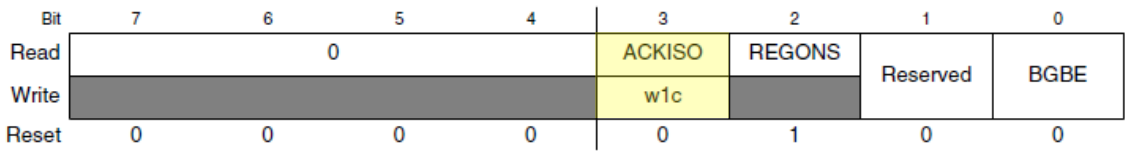


图 2. PMC_REGSC—Kinetis 120 MHz

删除的位/字段名称:

- TRAMPO
- VLPRS

改变的位/字段名称:

- ACKISO - 从 LLWU 变为 PMC

3.2.2 软件影响

PMC 寄存器名称保持一致；不过对一些重要的位进行了修改。

TRAMPO：删除了 TRAMPO 位。在 SMC 中增加了一个新位 RAM2PO，用于在 VLLS2 中控制 RAM 功耗。不过新的 RAM2PO 位具有不同的功能。

VLPRS：删除了 VLPRS 位。在进入 VLPR 模式时，软件只需检查运行稳压器状态和 SMC_PMSTAT 寄存器。当稳压器处于停止稳压或转换状态时，REGONS 位将清零，并且当 MCU 功耗模式为 VLPR 时，SMC_PMSTAT 的读数将为 04。

ACKISO：PMC 确认清除 I/O 引脚和振荡器模块上的保持状态。该位过去在 LLWU 中。从 VLLSx 模式恢复后必须清零该位，以释放 I/O 和振荡器模块上的保持状态。

3.2.3 硬件影响

带隙使能（**BGEN**）：该新位具有硬件交互影响。如果 BGEN 置位，带隙将开启。其他电路可能需要对该位进行操作，如 VREF 模块或 ADC。

ACKISO：当在低功耗模式下操作时，ACKISO 位是需要重点考虑的位。MCU 通过复位流程从 VLLSx 中恢复。在确认释放 I/O 之前，通常应初始化端口 I/O、定时器模块、通信模块和振荡器。

3.3 模式控制器版本 1 到版本 2

模式控制器 (MC) 和系统模式控制器 (SMC) 的主要功能是控制每种功耗模式的进入和退出。该模块在较新的 Kinetis 器件中有新的名称。MC 现在称为 SMC。SMC 的功能总是与电源管理控制器（PMC）、低漏电唤醒单元（LLWU）和新的复位控制模块（RCM）紧密联系在一起。

3.3.1 存储空间映射比较

在 SMC 模块中出现了 2 个新的寄存器。它们是 SMC_VLLSCTRL 和 SMC_PMSTAT。MC 中的两个 SRS 寄存器已移至 RCM。关于该寄存器的变化将在 RCM 章节中进行讨论。

Address: MC_PMPROT is 4007_E000h base + 2h offset = 4007_E002h

Bit	7	6	5	4	3	2	1	0
Read	0		AVLP	ALLS	0	AVLLS3	AVLLS2	AVLLS1
Write								
Reset	0	0	0	0	0	0	0	0

图 3. MC_PMPROT

Address: SMC_PMPROT is 4007_E000h base + 0h offset = 4007_E000h

Bit	7	6	5	4	3	2	1	0
Read	0		AVLP	0	ALLS	0	AVLLS	0
Write								
Reset	0	0	0	0	0	0	0	0

图 4. SMC_PMPROT

改变的位/字段名称:

- ALLS—移到位 3
- AVLLS3—AVLLS
- AVLLS2—AVLLS
- AVLLS1—AVLLS

Address: MC_PMCTRL is 4007_E000h base + 3h offset = 4007_E003h

Bit	7	6	5	4	3	2	1	0
Read	LPWUI	RUNM		0		LPLLSM		
Write								
Reset	0	0	0	0	0	0	0	0

图 5. MC_PMCTRL

Address: SMC_PMCTRL is 4007_E000h base + 1h offset = 4007_E001h

Bit	7	6	5	4	3	2	1	0
Read	LPWUI	RUNM		0	STOPA	STOPM		
Write								
Reset	0	0	0	0	0	0	0	0

图 6. SMC_PMCTRL

改变的位/字段名称:

- STOPA—SMC 中的新位
- LPLLSM—VLLSM，移到新寄存器 SMC_VLLSSCTRL
- STOPM—SMC 中的新位字段

Address: SMC_VLLSCTRL is 4007_E000h base + 2h offset = 4007_E002h

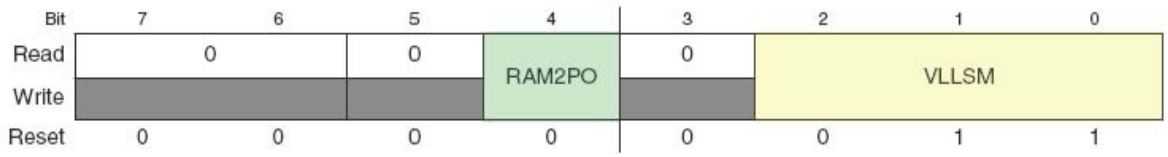


图 7. SMC_VLLCTRL

改变的位/字段名称:

- RAM2PO—SMC 中的新位
- VLLSM—SMC 中的新位字段

Address: SMC_PMSTAT is 4007_E000h base + 3h offset = 4007_E003h

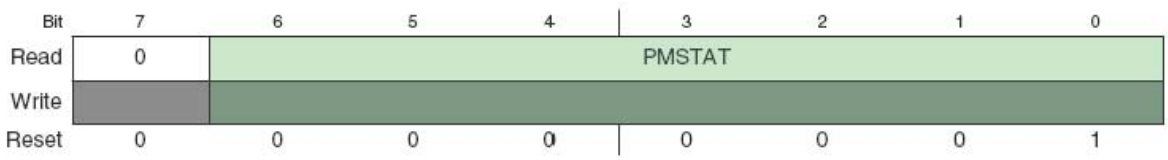


图 8. SCM_PMSTAT

改变的位/字段名称:

PMSTAT—SMC 中的新位字段

3.3.2 软件影响

通用影响：任何引用 MC 寄存器的寄存器将需要修改 RCM 和 SMC 寄存器名称。SRS 寄存器位于 RCM 中。模式控制函数的前缀从 MC 改为 SMC。

SMC_PMPROT：SMC_PMPROT 寄存器在复位寄存器类型后仍为一次性写入。进入 VLLSx 低功耗模式的保护由 SMC 中的一位控制。软件将需要在初始化代码中写入 PMPROT，从而仅允许所需的低功耗模式。寄存器的地址修改和 ALLS 位的位置修改在用于 SMC 的头文件定义中进行处理。

SMC_PMCTRL 和 **SMC_VLLSCTRL**:

SMC_PMCTRL 和 SMC_VLLSCTRL 寄存器控制模式的进入和退出。软件将需要修改写入这些寄存器的值，以实现在 MC_PMCTRL 寄存器中完成的功能。

在 SMC_PMCTRL 寄存器中读取新位 STOPA 意味着在停止进入序列期间发生了中断或复位，SMC 可以提前中止转换并返回运行模式，从而无需完全进入停止模式。 更多信息，请参见参考手册中的 SMC 功能说明。

新 STOPM 位字段的含义如下:

2–0 STOPM	Stop Mode Control When written, this field controls entry into the selected stop mode when sleep-now or sleep-on-exit mode is entered with SLEEPDEEP=1 . Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register. **NOTE:** When set to VLLSx, the VLLSM bits in the VLLSCTRL register is used to further select the particular VLLS sub-mode which will be entered. **NOTE:** 000 Normal stop (STOP) 001 Reserved 010 Very low power stop (VLPS) 011 Low leakage stop (LLS) 100 Very low leakage stop (VLLSx) 101 Reserved 110 Reseved 111 Reserved

低功耗模式进入变化：如果希望停止、VLPS 或 LLS 采用低功耗模式，必须先置位 PMPROT 位，并需要写入 STOPM。如果希望使用某种 VLLSx 模式，则需要额外写入一次 SMC_VLLSCTRL 中的 VLLSM 位。

2–0 VLLSM	VLLS Mode Control This field controls which VLLS sub-mode to enter if STOPM=VLLS. 000 Reserved 001 VLLS1 010 VLLS2 011 VLLS3 100 Reserved 101 Reserved 110 Reserved 111 Reserved

SMC_PMSTAT

读取 SMC 中的该新寄存器将给出当前的低功耗模式。

3.3.3 硬件影响

VLLS2 模式下的 **RAM** 功耗

SCM_VLLSCTRL 寄存器中的新位 RAM2PO 控制在 VLLS2 低功耗模式下 RAM 分区 2 是否上电。

目前支持 **VLPR** 和 **VLPW**

有了 SMC，现在可以支持模式 VLPR 和 VLPW。在不同版本中进入模式的方法是相同的。

在低功耗模式下调试

请注意，SMC 允许在运行、等待、VLPR 或 VLPW 下执行调试操作，方法与 MC 一致。调试器通过进入模拟停止模式来处理进入停止和 VLPS 模式的请求。关于 SMC 功能说明的更多信息，请参见参考手册。

3.4 复位控制器模块（RCM）

RCM 经过更新，但并非该模块中的所有功能均为新功能。先前位于 MC 中的 SRS 寄存器如今位于 RCM 中。SRS 寄存器也经过修改。

3.4.1 存储空间映射比较

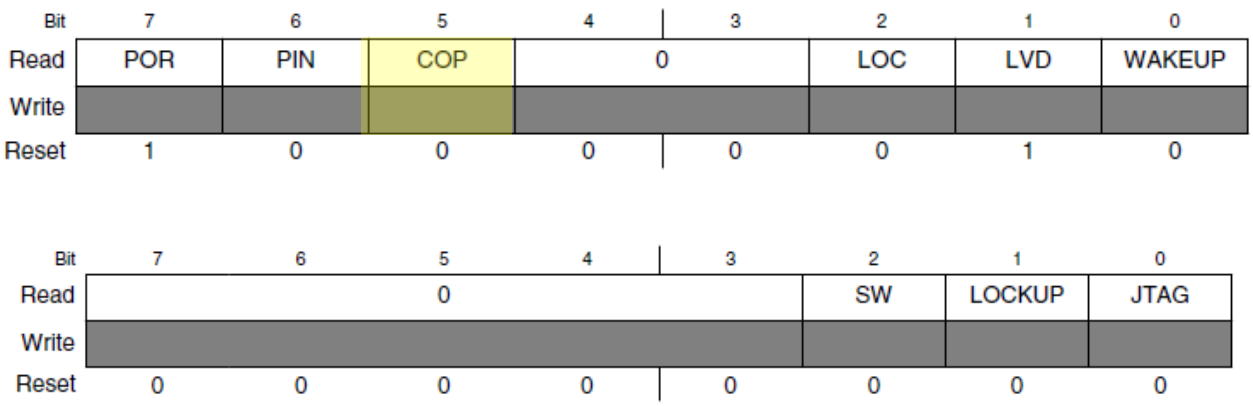


图 9. SRS 寄存器的高低位(MC_SRSx)

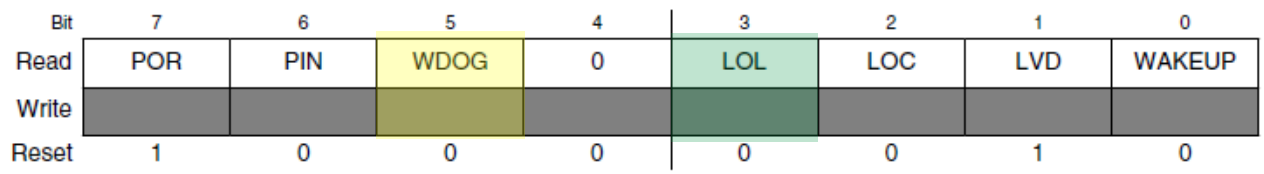


图 10. SRS 寄存器 0 (RCM_SRS0)

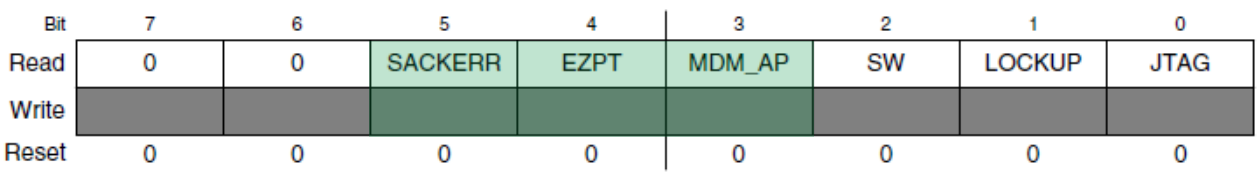


图 11. SRS 寄存器 1 (RCM_SRS1)

改变的位/字段名称:

- COP → WDOG - 改变了字段名

新的位/字段名称:

- LOL

- SACKERR
- EZPT
- MDM_AP

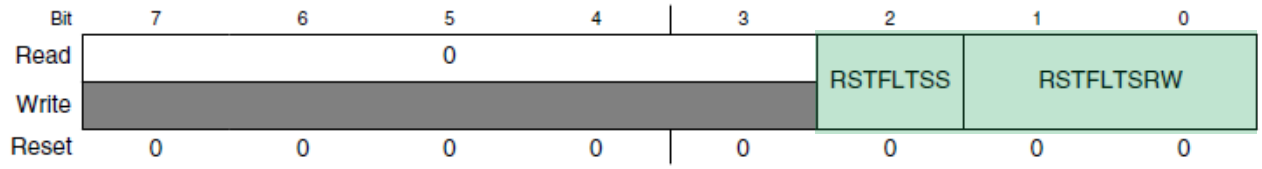


图 12. 复位引脚滤波器控制寄存器(RCM_RPFC)—新寄存器

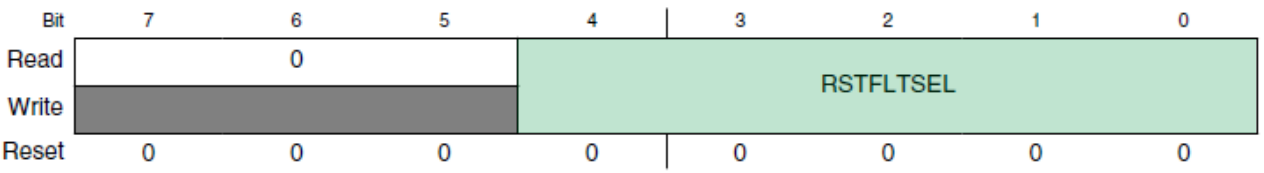


图 13. 复位引脚滤波器宽度寄存器(RCM_RFPW)—新寄存器

3.4.2 软件影响

任何对于 MC_SRSH 和 MC_SRSL 的引用都需要修改为新的寄存器名。用于解析这两个寄存器中的复位信息的软件需要加入这两个寄存器中的新位。

如果您想使用 RCM 的新数字滤波器功能，请参见参考手册中的“复位与启动”和“复位控制模块”章节。

3.4.3 硬件影响

在 RCM 中新增了四种复位源。

可通过 LPO 时钟或总线时钟, 对复位引脚输入进行数字滤波。可在 RCM_RFPW 寄存器中选择总线时钟滤波器的值。

3.5 LLWU

低漏电唤醒单元（LLWU）控制从 LLS 和 VLLSx 功耗模式退出。PMC 的功能与电源管理控制器（PMC）、系统模式控制器（SMC）和新的复位控制模块（RCM）紧密联系在一起。

3.5.1 存储空间映射比较

在 LLWU 模块中出现了 3 个新的寄存器。分别是 LLWU_FILT1、LLWU_FILT2 和 LLWU_RST 寄存器。从 LLWU 版本 2 中删除了 LLWU_CS 寄存器。

Address: LLWU_ME is 4007_C000h base + 4h offset = 4007_C004h

Bit	7	6	5	4	3	2	1	0
Read / Write	WUME7	WUME6	WUME5	WUME4	WUME3	WUME2	WUME1	WUME0
Reset	0	0	0	0	0	0	0	0

图 14. LLWU_ME 寄存器

改变的位/字段名称:

WUME7—连接到 RTC 秒使能

Address: LLWU_F3 is 4007_C000h base + 7h offset = 4007_C007h

Bit	7	6	5	4	3	2	1	0
Read	MWUF7	MWUF6	MWUF5	MWUF4	MWUF3	MWUF2	MWUF1	MWUF0
Write								
Reset	0	0	0	0	0	0	0	0

图 15. LLWU_F3 寄存器

改变的位/字段名称:

WUMF7—连接到 RTC 秒中断

RTC 秒标志与相应 LLWU 标志寄存器的位 7 相连接。

Address: LLWU_F3 is 4007_C000h base + 7h offset = 4007_C007h

Bit	7	6	5	4	3	2	1	0
Read	MWUF7	MWUF6	MWUF5	MWUF4	MWUF3	MWUF2	MWUF1	MWUF0
Write								
Reset	0	0	0	0	0	0	0	0

图 16. 删除了 LLWU_CS—版本 1

改变的位/字段名称:

- ACKISO—移动到 PMC_REGSC 寄存器中的位 3
- FLTEP—改变了滤波器功能

Address: LLWU_FILT1 is 4007_C000h base + 8h offset = 4007_C008h

Bit	7	6	5	4	3	2	1	0
Read	FILTF	FILTE		0	FILTSEL			
Write	w1c							
Reset	0	0	0	0	0	0	0	0

图 17. LLWU_FILT1—版本 2.x 中的新寄存器

Address: LLWU_FILT2 is 4007_C000h base + 9h offset = 4007_C009h

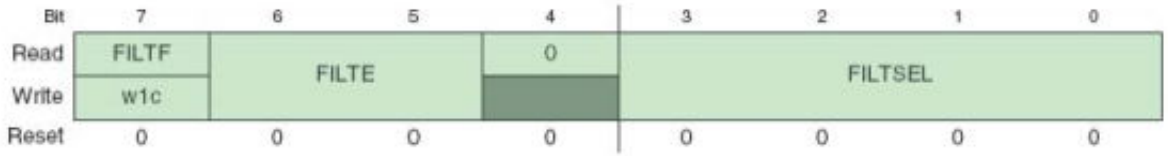


图 18. LLWU_FILT2—版本 2.x 中的新寄存器

Address: LLWU_RST is 4007_C000h base + Ah offset = 4007_C00Ah

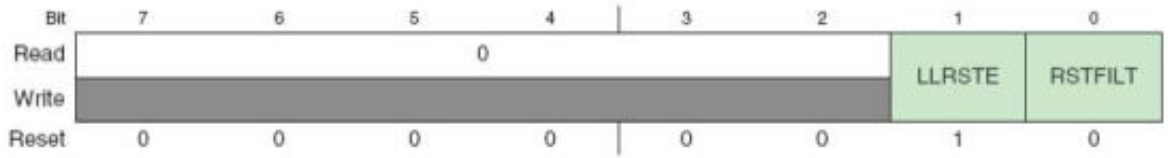


图 19. LLWU_RST—版本 2.x 中的新寄存器

当没有专用的复位引脚时，将使能 LLRSTE 和 RSTFILT 位的功能。Kinetis 部件具有专用的复位引脚；因此这些位不会改变复位引脚的操作。

3.5.2 软件影响

LLWU 模块使能和标志

除了 RTC 秒模块中断连接到 LLWU_ME 寄存器的位 7 [WUME7]之外，LLWU 版本 2 的寄存器描述没有明显变化。RTC 秒标志与相应 LLWU 标志寄存器 LLWU_F3 的位 7 [MWUF7]相连接。

LLWU 引脚和复位滤波器

寄存器 LLWU_FILT1、LLWU_FILT2 和 LLWU_RST 控制 LLWU 的滤波器功能。更多信息，请参见参考手册中的 LLWU 寄存器定义和功能说明。

移动 **ACKISO**

ACKISO 位现在位于 PMC 模块寄存器中。软件需要将“写入 1 清零”操作重定向到新寄存器。对于需要通过复位流程从 VLLSx 模式唤醒的用户，该位很重要。读 ACKISO 位指示了 I/O 管脚和系统振荡器是否处于锁存状态。请注意，不应过早清零该位。应在写入 ACKISO 之前重新初始化 I/O 和振荡器，否则可能在引脚上或振荡器启动时出现毛刺。

3.5.3 硬件影响

LLWU_M7IF → RTC 秒：在 LLWU 版本 2 中，用于模块唤醒的位 7 连接到 RTC 秒中断的输出。这允许 RTC 秒输出将 MCU 从 LLS 和 VLLSx 低功耗模式中唤醒。

LLWU 引脚滤波器功能：LLWU 模块中的引脚滤波器功能发生了改变。在原来的 LLWU 中，所有 LLWU 输入均馈入一个引脚滤波器电路，从而产生单一的唤醒标志。在新版本中，只有两个预先选择的 LLWU 输入会馈入到两个引脚滤波器电路，并且产生两个可能的滤波器引脚唤醒标志。

新版本的 LLWU 提供用于复位引脚的滤波器和引脚使能，将 MCU 从 LLS 和 VLLSx 低功耗模式中唤醒。对于像 Kinetis 这样具有专用复位引脚的器件，从 LLS 和 VLLSx 模式唤醒总是为使能，这些位不能使能或禁止复位引脚作为低功耗模式下的唤醒源。

3.6 RNG-B 到 RNG-A

Kinetis 100 MHz 1.x 版本使用随机数生成器版本 B（RNG-B），而最新的 Kinetis 系列使用随机数生成器版本 A（RNG-A）。虽然这两个版本完全不同，但是移植过程却无比简单。本章节详细说明了这两个版本的区别，说明了必须对 Kinetis 设置作出哪些修改，以确保从 RNG-B 到 RNG-A 的顺利转换。

3.6.1 RNG-A 与 RNG-B

RNG-B 类型是加密的强随机数生成器，具有三个显著特性:

- 经美国国家标准与技术研究所（NIST）批准的伪随机数生成器（http://csrc.nist.gov）
- 包含数字签名标准中定义的密钥生成算法（http://www.itl.nist.gov/fipspubs/fip186.htm）
- 集成的熵源可以为 PRNG 提供熵作为其种子

RNG-B 使用真随机数生成器模块（TRNG）向寄存器添加熵，作为伪随机数生成器的种子。以下为详细的 RNG-B 模块框图。

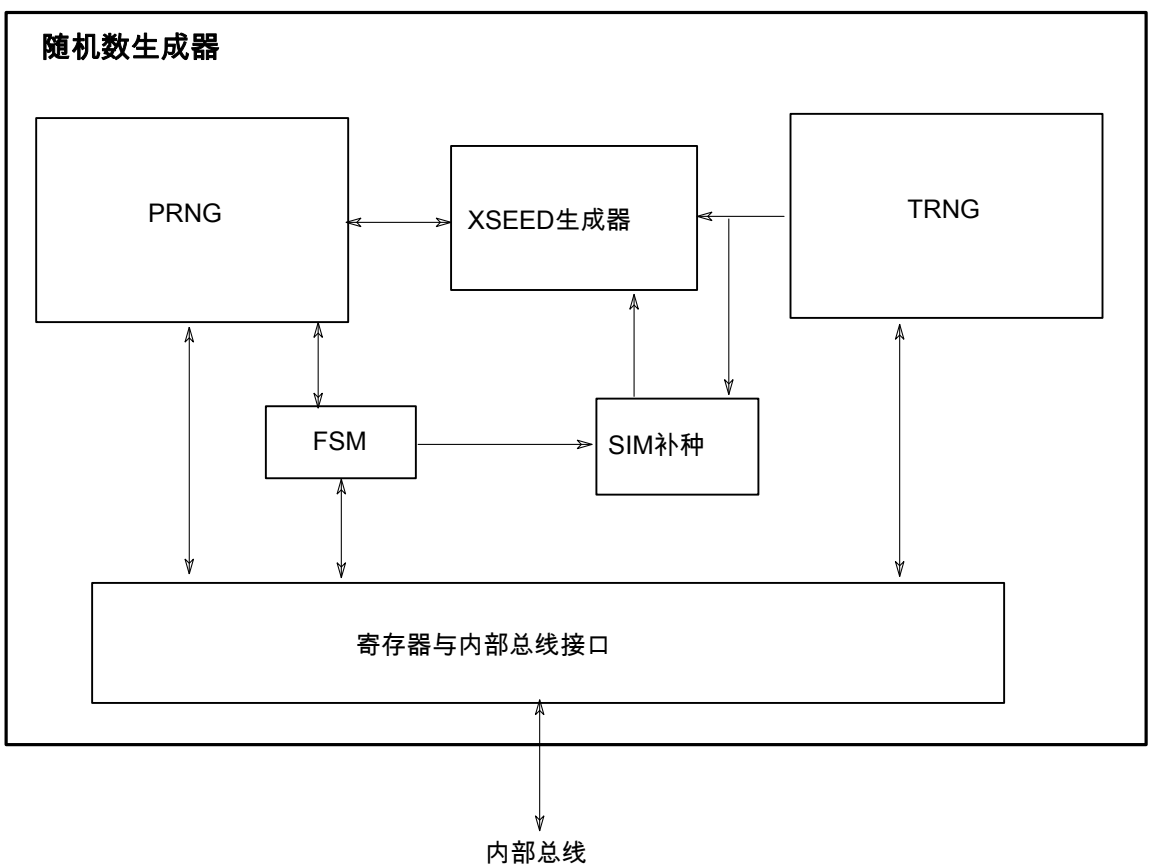


图 20. 随机数生成器

RNG-A 类型为简化的伪随机数生成器。该模块比 RNG-B 类型规模小且更简单。以下显示的是 RNG-A 结构框图。

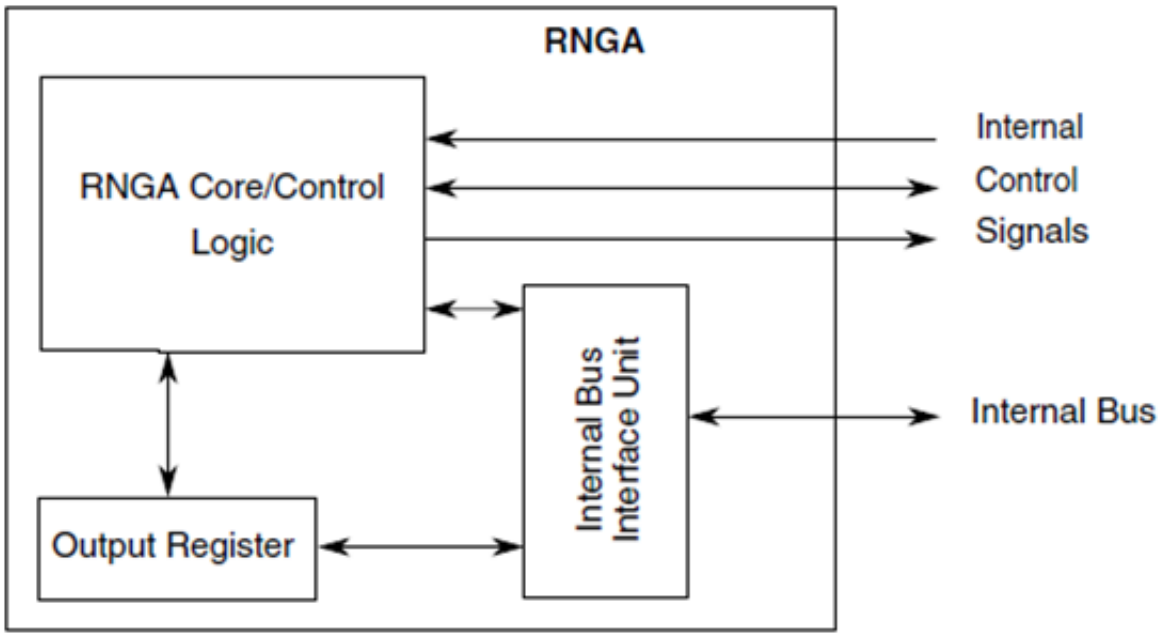


图 21. RNG-A 结构框图

目前并无已知加密技术可证明这是一种生成随机数据的安全方法。因此，强烈建议将该模块产生的随机数据作为经 NIST 批准 (基于 DES 或 SHA-1) 的伪随机数生成器的种子，该伪随机数生成器在 NIST Fips Pub 186-2 附录 3 和 NIST Fips Pub SP 800-90 中定义。

虽然这两个模块在内部区别很大，但是其操作非常相似。RNG-A 模块不允许手动提供种子。因此，这里仅比较自动操作算法。以下显示了两种不同的初始化和操作模式。

表 3. 初始化和操作算法比较

RNG-A	RNG-B（自动）	RNG-B（手动）
1. 复位/初始化 2. 写入 RNGA 控制寄存器并置位中断掩码（INTM）、高可靠（HA）和 GO 位。 3. 查询用于 RNGA 输出寄存器电平的 RNGA 状态寄存器。 4. 从 RNGA 输出寄存器读取可用的随机数据。 5. 根据需要重复步骤 3 和 4。	1. 复位/初始化 2. 写入 RNG_CR 以设置 RNGB 用于自动产生种子和所需功能。 3. 等待中断以指示第一个种子完成。 4. 查询 RNG_SR 以获取 FIFO 等级。 5. 从输出 FIFO 读取可用的随机数据。 6. 根据需要重复步骤 4 和 5。根据需要自动产生种子，并对操作透明。	1. 复位/初始化 2. 写入 RNG_CR 以设置所需功能。 3. 写入 RNG_CMD 寄存器以运行自测或种子生成。 4. 等待中断以指示所要求的操作完成。 5. 如果种子生成未完成，则重复步骤 3 和 4。 6. 查询 RNG_SR 以获取 FIFO 等级。 7. 从输出 FIFO 读取可用的随机数据。 8. 根据需要重复步骤 6 和 7，直到生成 2^{20} 个字为止。 9. 写入 RNG_CMD 以运行种子模式。 10. 重复步骤 4-9。

3.6.2 存储空间映射比较

RNG-A 和 RNG-B 模块共用 RNG-A 模块中的所有寄存器 (熵寄存器除外)。因此，移植到 RNG-A 需要您删除一些代码，并修改写入到共用寄存器中的值。

虽然它们共用寄存器，但是寄存器并不位于相同的存储器位置。因此，您必须使用 Freescale 提供的最新头文件来更新您的系统，从而使您的系统可以正常工作，这非常重要。

下图显示了存储空间映射位置的变化。对于这些寄存器的复位值修改将在具体的寄存器比较中讨论。

存储空间映射比较				
	RNG-B		RNG-A	
	位置	名称	位置	名称
控制寄存器	400A_0008	RNG_CR	400A_0000	RNG_CR
状态寄存器	400A_000C	RNG_SR	400A_0004	RNG_SR
输出寄存器	400A_0014	RNG_OUT	400A_000C	RNG_OR
熵寄存器	N/A	N/A	400A_0008	RNG_ER
命令寄存器	400A_0004	RNG_CMD	N/A	N/A
错误状态寄存器	400A_0010	RNG_ESR	N/A	N/A
版本寄存器	400A_0000	RNG_VER	N/A	N/A

除了位置变化，寄存器结构和一些复位值也发生了变化。RNG-A 和 RNG-B 的控制寄存器结构如下所示。请注意，这些寄存器结构完全不同。

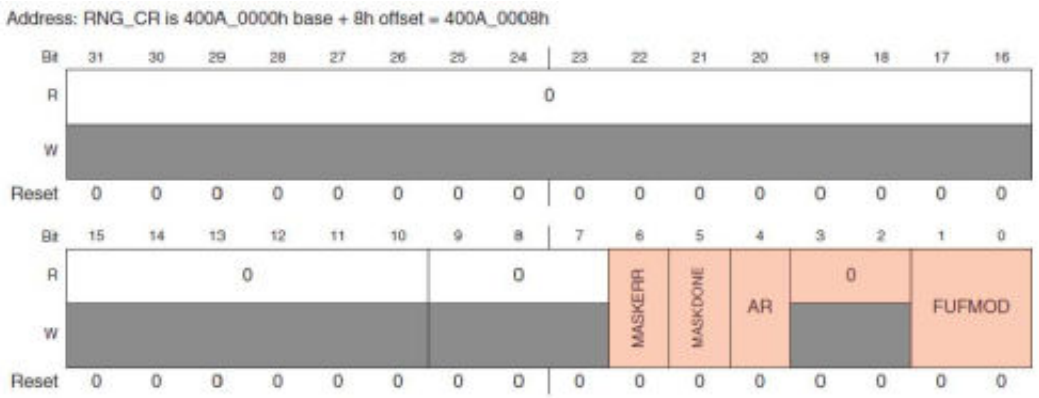


图 22. RNG-B 模块 RNG_CR

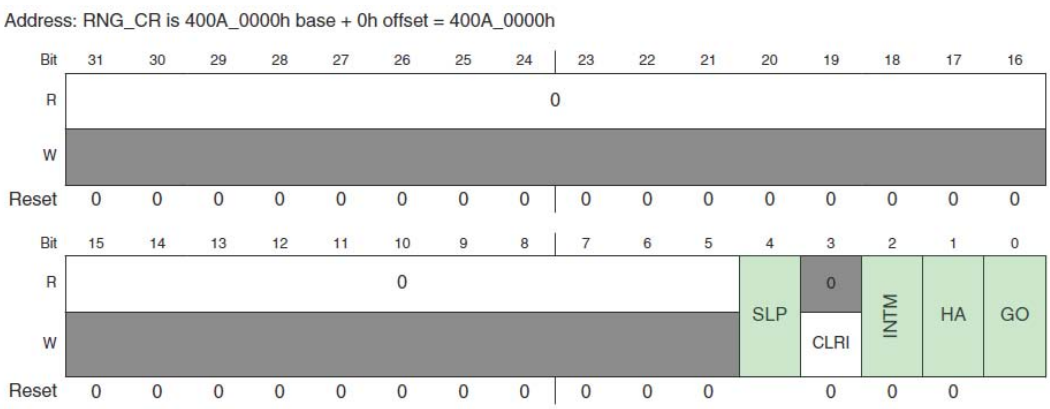


Figure 23. RNG-A module RNG_CR

图 23. RNG-A 模块 RNG_CR

删除的位字段:

- MASKERR

- MASKDONE
- AR
- FUFMOD

增加的位字段：

- SLP：置位该位使 RNG 模块进入休眠模式；清零该位将唤醒该模块。
- CLRI：置位该位以清除错误中断标志。
- INTM：置位该位以使能 RNG 中断；清零该位以禁止这些中断。
- HA：在 RNG 寄存器中使能安全违规位。当该位置位时，不允许读取 RNG 寄存器。
- GO：向 RNG 寄存器载入随机数据。

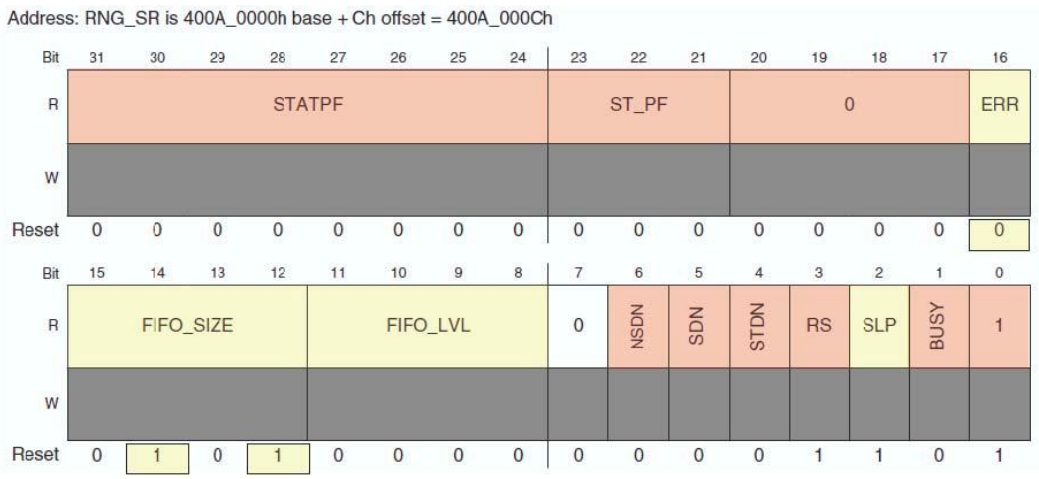


图 24. RNG-B 模块 RNG_SR

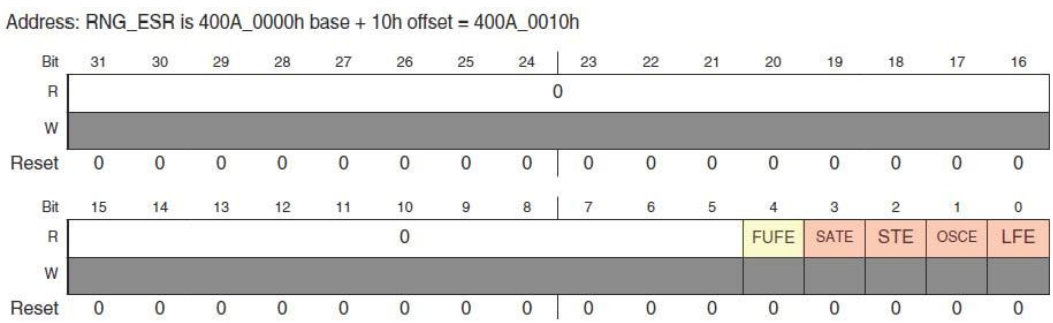


图 25. RNG-B 模块 RNG_ESR

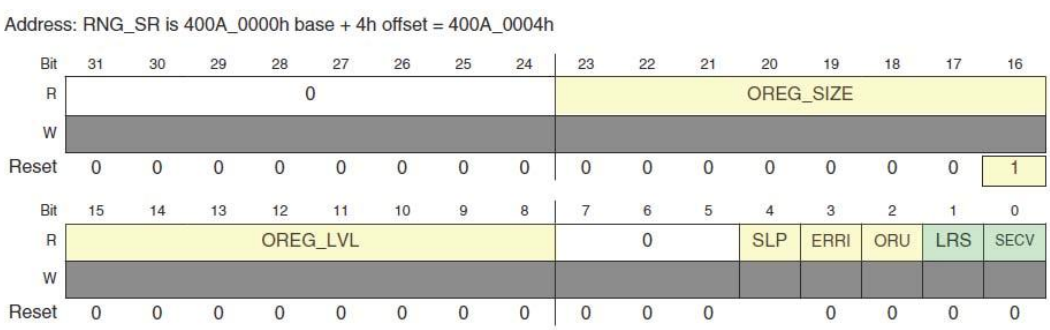


图 26. RNG-A 模块 RNG_SR

增加的位字段：

- SECV：如果置位，则指示已发生安全违规。如果清零，则未发生安全违规。
- LRS：当置位时，最后读取状态位指示在执行最后读取时输出寄存器为空（下溢情况）。

修改的位字段：

- OREG_SIZE：该 8 位整数字段用于指示输出寄存器的大小，原来为 RNG-B 类型中的 FIFO_SIZE。
- OREG_LVL：该位指示输出寄存器中可用的随机字。只可能为两个值（0b00000001 或 0b00000000）。该位原来为 RNG-B 类型中的 FIFO_LVL。
- SLP：置位时指示 RNG 模块处于休眠模式。该位从位 2（位于 RNG-B 类型中）移至位 4。
- ERRI：置位时，该位指示输出寄存器读取时为空。该位从位 16（位于 RNG-B 类型中）移至位 3。
- ORU：置位时，该位指示从最后读取状态寄存器到现在，输出寄存器读取时为空。该位从 ESR 寄存器的位 4（位于 RNG-B 类型中）移至位 2。

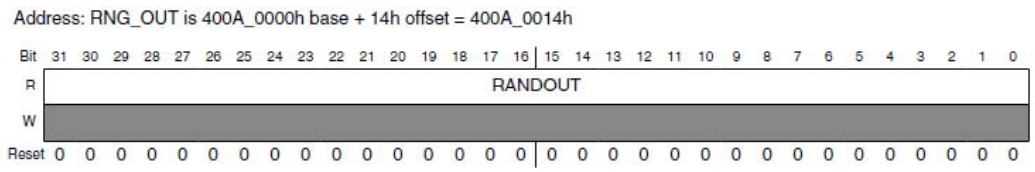


图 27. RNG-B 模块 RNG_OUT

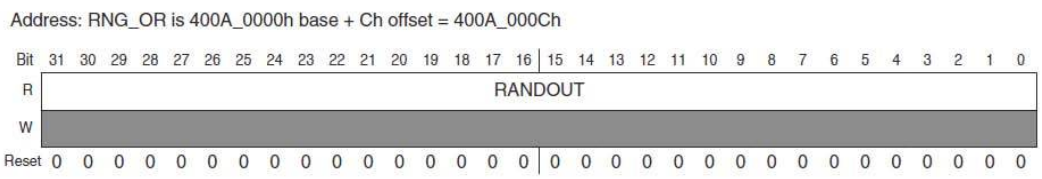


图 28. RNG-A 模块 RNG_OR

输出寄存器为简单的 32 位寄存器，在随机数生成后存放结果随机数，因而结构保持不变。寄存器重命名如下。

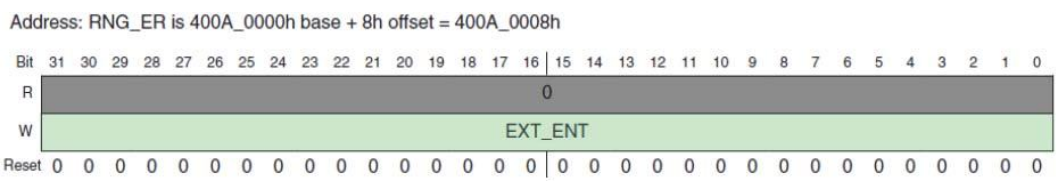


图 29. RNG-A 模块 RNG_ER

RNG-A 模块允许您向随机数生成过程添加熵。这通过向只写熵寄存器写入值实现。可用于熵寄存器的熵来源为：

- 当前时间（可能的最高精度）
- 鼠标和/或键盘动作
- 其他随机数生成器

3.6.3 软件影响

飞思卡尔 半导体将向受该移植影响的客户提供新的头文件和链接文件，使移植到新 RNG-A 模块尽可能地顺利。必须记住的是（已在前文提及），RNG-A 算法不生成加密的强随机数。如果在您当前的软件设置中，加密算法依靠加密的强随机数生成器，那么您将需要经 NIST 批准的支持软件以产生加密的强随机数，用于您当前的软件解决方案。此外，建议您使用 RNG-A 模块的熵寄存器，以向您的随机数生成过程添加额外的熵。

对于从 RNG-B 向 RNG-A 移植有两种情况：采用手动生成种子的 RNG-B 算法，或采用自动生成种子的 RNG-B 算法。以下先看一下手动生成种子的 RNG-B 算法。

3.6.4 使用手动生成种子算法时的影响

如果您在代码中使用手动生成种子，那么您将需要基于 RNG-A 的自动生成种子算法重写代码，因为 RNG-A 不提供手动生成种子选项。以下对比显示了两种算法，并强调了相似之处

RNG-B 手动算法	RNG-A 算法
复位/初始化	复位/初始化
写入 RNG_CR 以设置所需功能	写入 RNGA_CR 并置位中断掩码（INTM）、高可靠（HA）和 GO 位
写入 RNG_CMD 寄存器以运行自测或种子生成。	
等待中断以指示所要求的操作完成。	
如果种子生成未完成，则重复步骤 3-4。	
查询 RNG_SR 以获取 FIFO 等级。	查询 RNG_SR 以获取输出寄存器电平。
从输出 FIFO 读取可用的随机数据。	从输出寄存器读取可用的数据。
根据需要重复步骤 6 和 7，直到生成 2^{20} 个字为止。	
写入 RNG_CMD 以运行种子模式	
重复步骤 4-9。	重复步骤 3 和 4。

可从上述比较看出，当切换到 RNG-A 算法时，可从您的代码工程中删除步骤 3、4、5 和 9。如果您不使用 Freescale 提供的头文件和链接文件，则在写入 RNGA_CR 寄存器时，您需要更新写入 CR 寄存器的值。此外，在查询 SR 寄存器时，您将查询位 8–15（而不是 8–11）。

3.6.5 当使用自动生成种子算法时的影响

当采用自动生成种子功能时，RNG-A 和 RNG-B 之间的算法差异很小。查看以下比较表。

RNG-B 手动算法	RNG-B 手动算法
复位/初始化	复位/初始化
写入 RNG_CR 以设置所需功能	写入 RNGA_CR 并置位中断掩码（INTM）、高可靠（HA）和 GO 位
等待中断以指示第一个种子完成。	
查询 RNG_SR 以获取 FIFO 等级。	查询 RNG_SR 以获取输出寄存器电平。
从输出 FIFO 读取可用的随机数据。	从输出寄存器读取可用的数据。
重复步骤 4-9。	重复步骤 3 和 4。

如果您使用自动生成种子算法，您将只需简单地删除与 RNG-B 生成种子中断相关的代码，调整写入控制寄存器的值使其适用于 RNG-A，以及调整代码以读取正确的输出电平寄存器。为了简化读取输出寄存器，建议您插入一条定义声明，将 RNG-B 的寄存器名称转换为 RNG-A 的输出寄存器名。

```
(#define RNG_OUT    RNG_OR).
```

3.6.6 硬件影响

无硬件影响。

3.7 SSI 到 SAI

Kinetis 100 MHz 版本 1.x 器件使用 SSI 模块来提供 I2S 功能，而新的 Kinetis 器件使用 SAI 模块。虽然这两个模块的功能类似，但仍然存在许多不同。

除了从 SSI 变为 SAI 之外，120 MHz 器件还为 SAI 提供了两个实例化，而 100 MHz 器件只有一个实例化。

3.7.1 特性

虽然 SAI 模块的寄存器映射和位定义与 SSI 模块相差很大，但其主要特性仍然相同。SAI 支持具有帧同步的全双工同步串行接口，如 I2S、TDM、AC97、编解码器和 DSP 接口。然而，在从 SSI 移植到 SAI 的过程中，添加了一些特性，也删除了一些特性：

以下列出了在 SAI 模块上删除的特性：

- 没有分别禁止发送和接收帧同步的选项。一旦使能 TE 或 RE，则生成帧同步，这只能通过清零 TE 或 RE 来禁止。
- 没有在 RX 位时钟上输出过采样时钟的选项。
- 没有用于控制 SAI 是运行于 I2S 模式还是联网模式的单独的位；一切取决于您如何配置帧同步大小。
- 不再支持门控时钟模式。
- 不对 AC97 提供硬件支持，如果每帧配置为 13 或更多字，则可通过软件支持 AC97。
- 不支持单独的帧起始标志和帧最后未屏蔽字起始标志。这些标志由单个标志代替，以表示帧的起始字。
- 没有禁止 TX 或 RX FIFO 的选项。它们始终使能。
- 不支持单独的左/右 FIFO。取而代之的是，分别提供两个用于 TX 和 RX 的引脚，每个引脚具有其自己的 FIFO。
- 没有选择 MSB 对齐或 LSB 对齐的选项。
- 没有表示 FIFO 入口计数的位。取而代之的是，使用一个写入和读取 FIFO 指针以表示当前的 FIFO 入口，并且可用于辨别 FIFO 为满还是为空。

以下列出了 SAI 模块上的新特性：

- 支持停止模式操作
- 支持调试模式操作
- 支持 32 位传输
- 增加分别使能和禁止位时钟的选项
- 增加帧同步错误标志
- 除 TX 和 RX FIFO 水印触发器之外，增加 TX FIFO 空或 RX FIFO 满标志，以允许触发中断和 DMA
- 一个额外的 TX 和 RX 数据通道
- SAI 主时钟选择和分频器寄存器重定位到 SAI 本身的存储器空间。(对于 Kinetis 版本 1.x，这些位于 SIM 模块中。)

3.7.2 存储空间映射比较

表 7 提供了 SSI 和 SAI 模块的存储空间映射比较。由于存储空间映射和寄存器功能在某些情况下发生了重大改变，比较表使用了以下约定：

1. 实现相似功能的寄存器列在同一行内，例如 I2S0_TX0 对应于 I2S0_TDR0。

2. 如果 SSI 模块中某个寄存器的功能需要在不止一个寄存器上实现，则会合并该寄存器映射，从而使其对应 SAI 模块中的所有相关寄存器；例如 I2S0_TCR 的功能需要在 I2S0_TCR2、I2S0_TCR3 和 I2S0_TCR4 上实现。
3. 一些寄存器在 SSI 或 SAI 模块中没有相对应的寄存器；例如在 SAI 模块上没有用于 AC97 的硬件加速支持，在 SAI 模块上没有相关的 AC97 寄存器。在这些情况下，将在相关行内显示“N/A”，以表示无可用寄存器。
4. 对于 SSI 模块，主时钟生成需要在 SIM 模块中配置两个寄存器，而 SAI 模块将相关寄存器集成到其自身的存储器空间中。

表 7. 存储空间映射比较

	SSI			SAI	
I2S 发送数据寄存器 0	4002_F000	I2S0_TX0	SAI 发送数据寄存器 0	4002_F020	I2S0_TDR0
I2S 发送数据寄存器 1	4002_F004	I2S0_TX1	SAI 发送数据寄存器 1	4002_F024	I2S0_TDR1
I2S 接收数据寄存器 0	4002_F008	I2S0_RX0	SAI 接收数据寄存器 0	4002_F0A0	I2S0_RDR0
I2S 接收数据寄存器 1	4002_F00C	I2S0_RX1	SAI 接收数据寄存器 1	4002_F0A4	I2S0_RDR1
I2S 控制寄存器	4002_F010	I2S0_CR	SAI 发送控制寄存器	4002_F000	I2S0_TCSR
			SAI 接收控制寄存器	4002_F080	I2S0_RCSR
			SAI 发送配置 2 寄存器	4002_F008	I2S0_TCR2
			SAI 接收配置 2 寄存器	4002_F088	I2S0_RCR2
			SAI 发送配置 3 寄存器	4002_F00C	I2S0_TCR3
			SAI 接收配置 3 寄存器	4002_F08C	I2S0_RCR3
I2S 中断状态寄存器	4002_F014	I2S0_ISR	SAI 发送控制寄存器	4002_F000	I2S0_TCSR
			SAI 接收控制寄存器	4002_F080	I2S0_RCSR
I2S 中断使能寄存器	4002_F018	I2S0_IER	SAI 发送控制寄存器	4002_F000	I2S0_TCSR
			SAI 接收控制寄存器	4002_F080	I2S0_RCSR
I2S 发送配置寄存器	4002_F01C	I2S0_TCR	SAI 发送配置 2 寄存器	4002_F008	I2S0_TCR2
			SAI 发送配置 3 寄存器	4002_F00C	I2S0_TCR3
			SAI 发送配置 4 寄存器	4002_F010	I2S0_TCR4
I2S 接收配置寄存器	4002_F020	I2S0_RCR	SAI 接收配置 2 寄存器	4002_F088	I2S0_RCR2
			SAI 接收配置 3 寄存器	4002_F08C	I2S0_RCR3
			SAI 接收配置 4 寄存器	4002_F090	I2S0_RCR4

下一页继续介绍此表...

表 7. 存储空间映射比较 (继续)

	SSI			SAI	
I2S 发送时钟控制寄存器	4002_F024	I2S0_TCCR	SAI 发送配置 2 寄存器	4002_F008	I2S0_TCR2
			SAI 发送配置 4 寄存器	4002_F010	I2S0_TCR4
			SAI 发送配置 5 寄存器	4002_F014	I2S0_TCR5
I2S 接收时钟控制寄存器	4002_F028	I2S0_RCCR	SAI 接收配置 2 寄存器	4002_F088	I2S0_RCR2
			SAI 接收配置 4 寄存器	4002_F090	I2S0_RCR4
			SAI 接收配置 5 寄存器	4002_F094	I2S0_RCR5
I2S FIFO 控制/状态寄存器	4002_F02C	I2S0_FCSR	SAI 发送配置 1 寄存器	4002_F004	I2S0_TCR1
			SAI 接收配置 1 寄存器	4002_F084	I2S0_RCR1
			SAI 发送 FIFO 0 寄存器	4002_F040	I2S0_TFR0
			SAI 发送 FIFO 1 寄存器	4002_F044	I2S0_TFR1
			SAI 接收 FIFO 0 寄存器	4002_F0C0	I2S0_RFR0
			SAI 接收 FIFO 1 寄存器	4002_F0C4	I2S0_RFR1
I2S0 发送时隙掩码寄存器	4002_F048	I2S0_TMSK	SAI 发送掩码寄存器	4002_F060	I2S0_TMR
I2S0 接收时隙掩码寄存器	4002_F04C	I2S0_RMSK	SAI 接收掩码寄存器	4002_F0E0	I2S0_RMR
I2S AC97 命令数据寄存器	4002_F040	I2S0_ACDAT	N/A	N/A	N/A
I2S AC97 标签寄存器	4002_F044	I2S0_ATAG	N/A	N/A	N/A
I2S AC97 通道状态寄存器	4002_F050	I2S0_ACCST	N/A	N/A	N/A
I2S AC97 通道使能寄存器	4002_F054	I2S0_ACCEN	N/A	N/A	N/A
I2S AC97 通道禁止寄存器	4002_F058	I2S0_ACCDIS	N/A	N/A	N/A
系统选项寄存器 2	4004_8004	SIM_SOPT2	SAI MCLK 控制寄存器	4002_F100	I2S0_MCR
系统时钟分频器寄存器 2	4004_8048	SIM_CLKDIV2	SAI MCLK 分频器寄存器	4002_F014	I2S0_MDR

3.7.2.1 控制寄存器

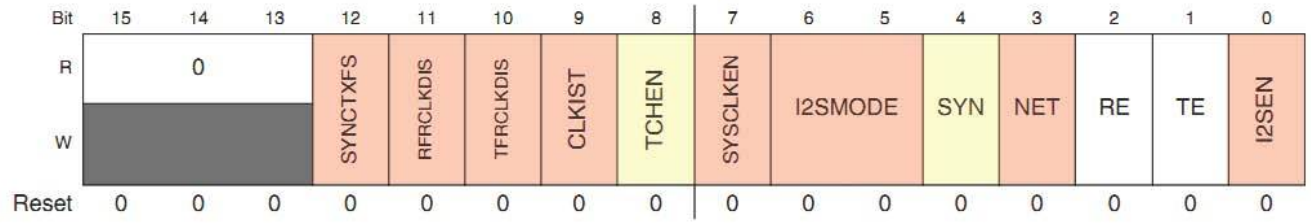


图 30. I2S0_CR SSI

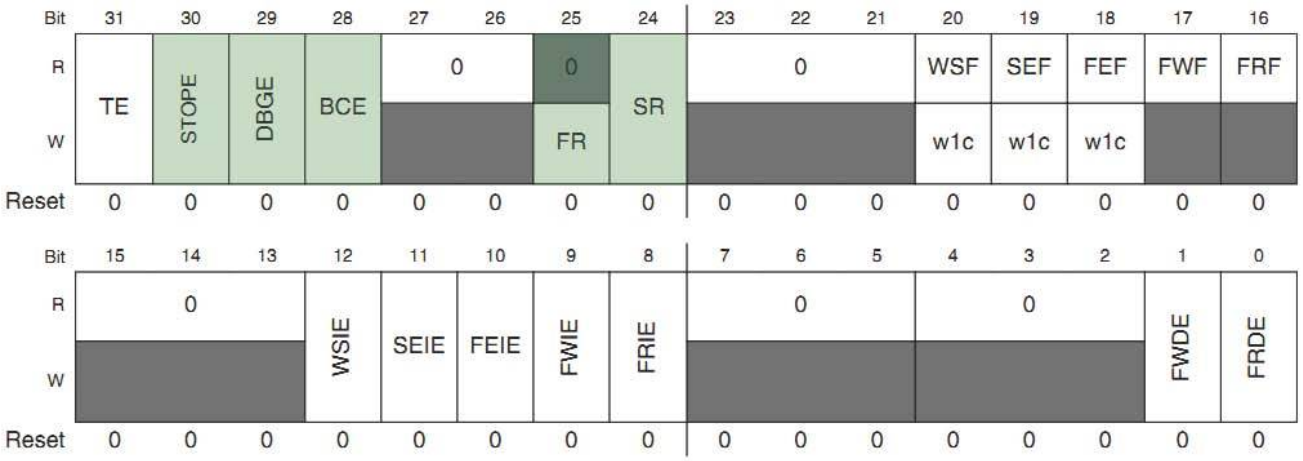


图 31. I2S0_TCSR SAI

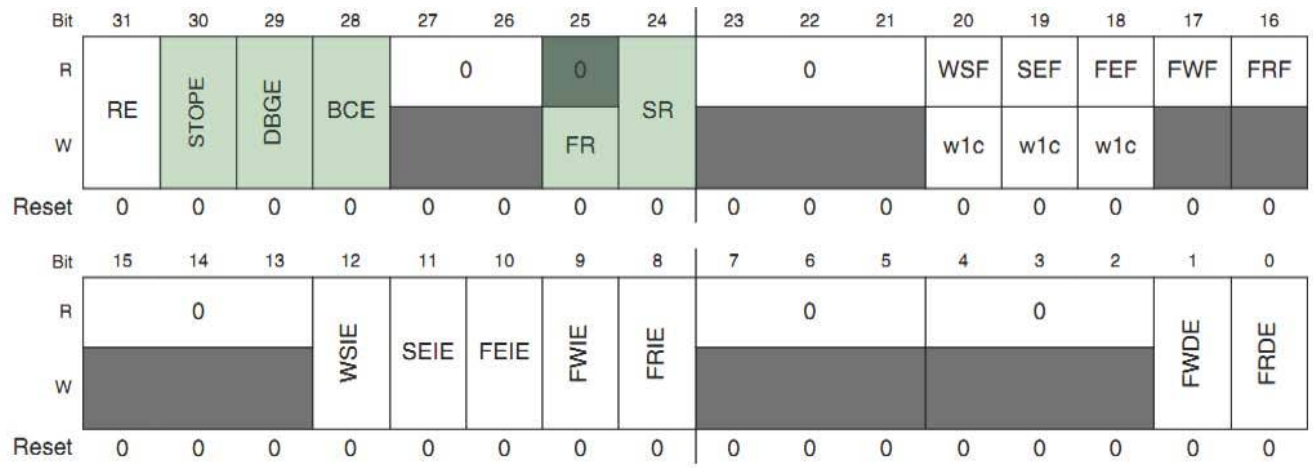


图 32. I2S0_RCSR SAI

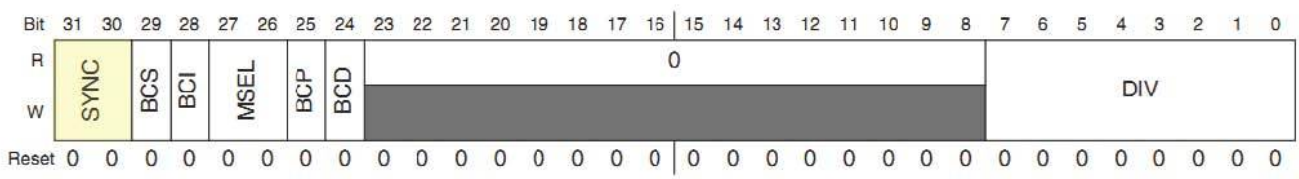


图 33. I2S0_TCR2 和 I2S0_RCR2 SAI

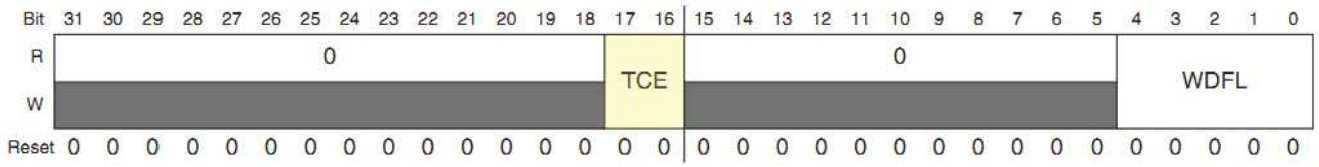


图 34. I2S0_TCR3 SAI

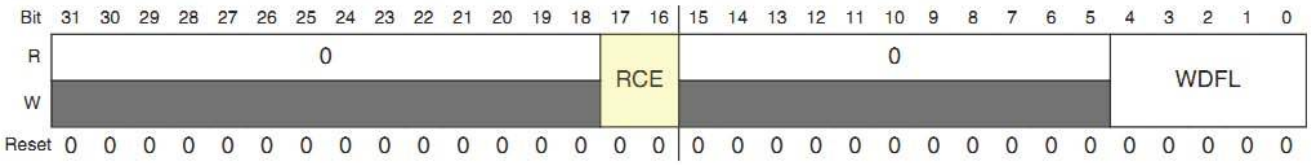


图 35. I2S0_RCR3 SAI

新增的位/字段：

- **STOPE**：停止使能
- **DBGE**：调试使能
- **FR**：FIFO 复位
- **SR**：软件复位

改变的位/字段名称：

- TCHEN → TCE 和 RCE
- SYN → SYNC

删除的位/字段：

- **SYNCTXFS**：CR[TE]锁存，带 FS
- **RFRCLKDIS**：接收帧同步禁止
- **TFRCLKDIS**：发送帧同步禁止
- **CLKIST**：在 I2S 门控时钟模式下的时钟空闲状态
- **SYSCLKEN**：过采样时钟使能
- **I2SMODE**：I2S 模式选择
- **NET**：网络模式 I2SEN：I2S 使能

3.7.2.2 中断和状态使能寄存器

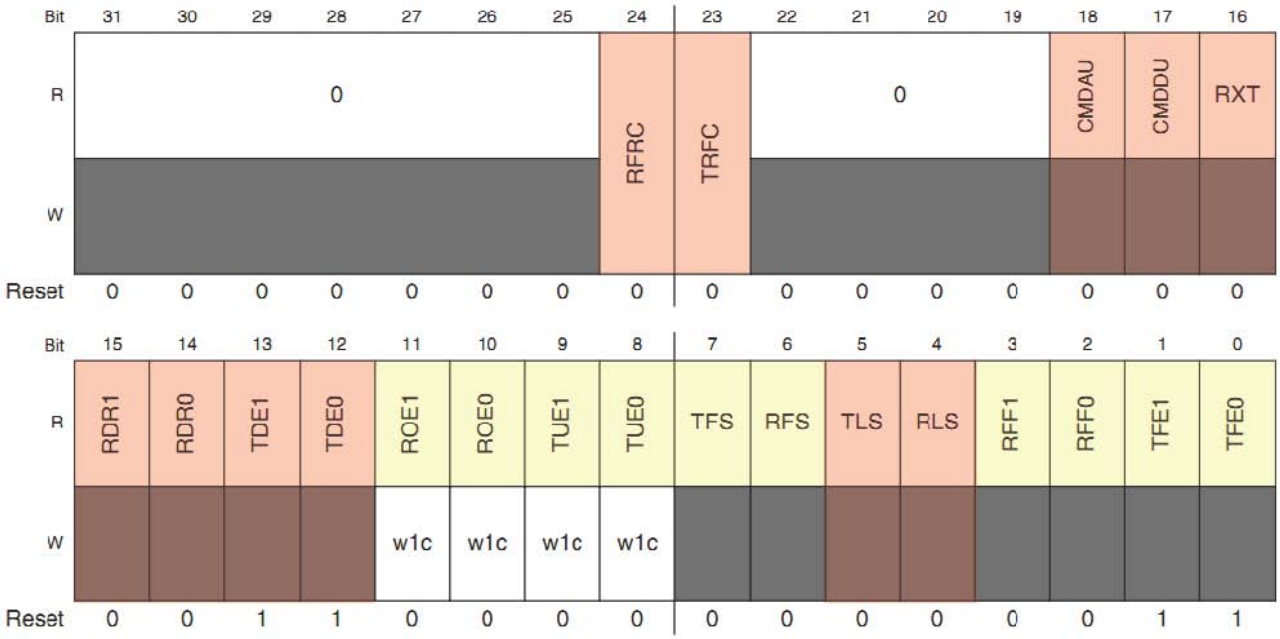


Figure 36. I2S0_ISR—SSI

图 36. I2S0_ISR—SSI

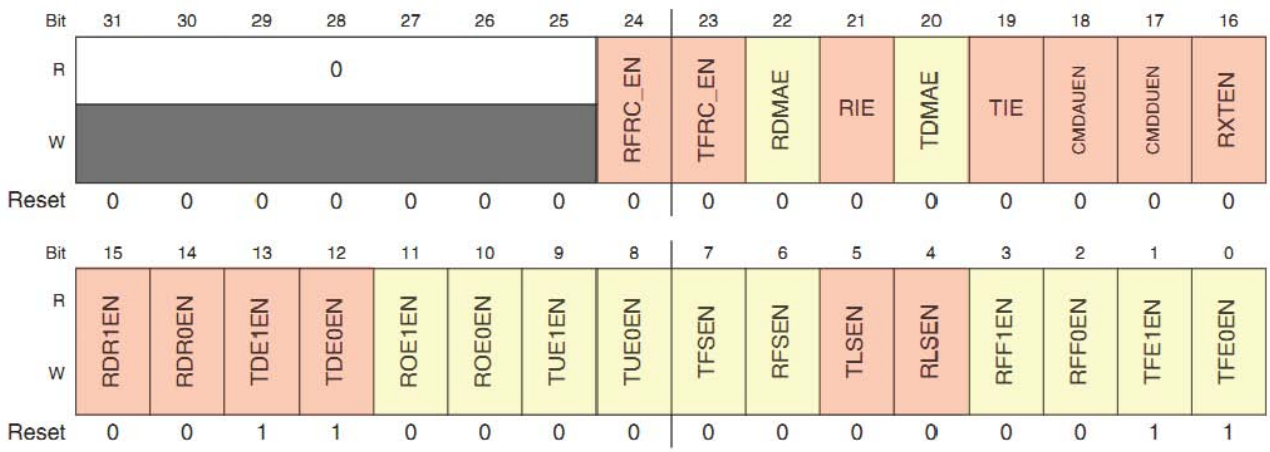


Figure 37. I2S0_IER—SSI

图 37. I2S0_IER—SSI

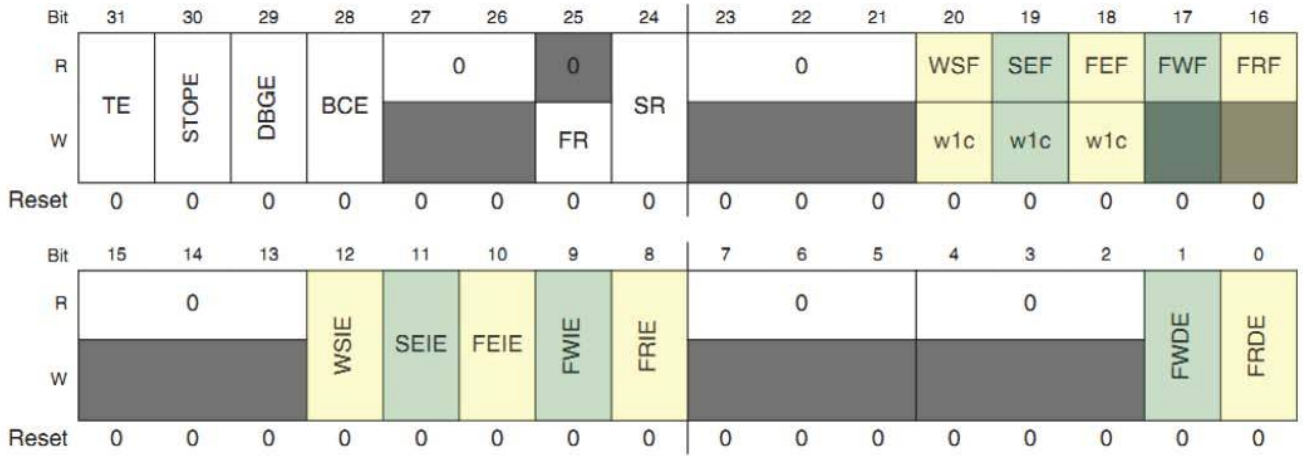


图 38. I2S0_TCSR—SAI

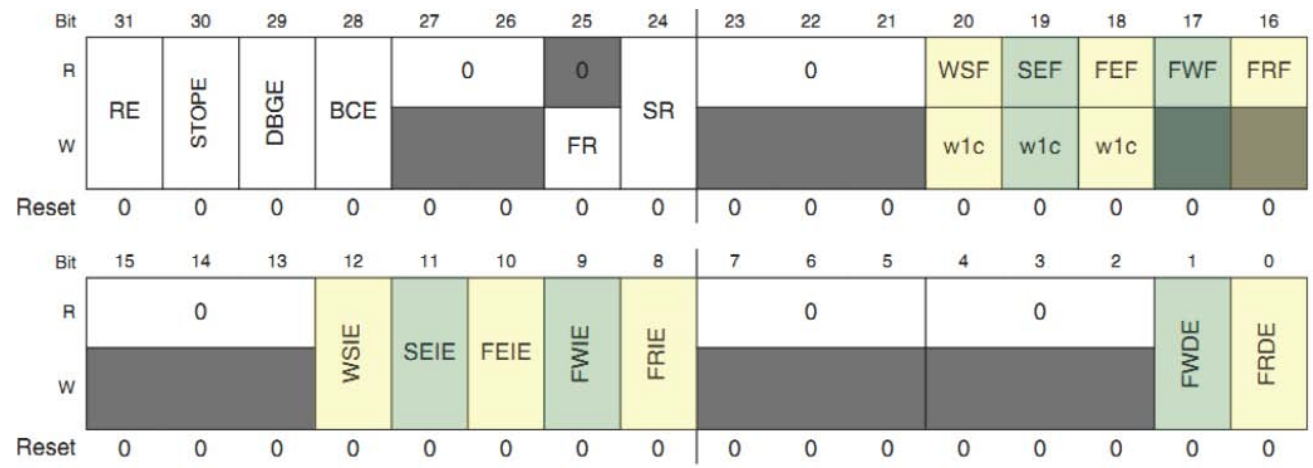


Figure 39. I2S0_RCSR—SAI

图 39. I2S0_RCSR—SAI

新增的位/字段：

- **SEF**：同步错误标志
- **FWF**：FIFO 警告标志
- **SEIE**：同步错误中断使能
- **FWIE**：FIFO 警告中断使能
- **FWDE**：FIFO 警告 DMA 使能

改变的位/字段名称：

- ROE1 和 ROE0 → RCSR[FEF]（接收 FIFO 错误，上溢）
- TUE1 和 TUE0 → TCSR[FEF]（发送 FIFO 错误，下溢）
- TFS → TCSR[WSF]（发送帧同步）
- RFS → RCSR[WSF]（接收帧同步）
- RFF1 和 RFF0 → RCSR[FRF]（接收 FIFO 请求，接收数据大于水印）
- TFE1 和 TFE0 → TCSR[FRF]（发送 FIFO 请求，发送数据小于水印）
- RDMAE → RCSR[FRDE]（接收 FIFO 请求 DMA 使能）
- TDMAE → TCSR[FRDE]（发送 FIFO 请求 DMA 使能）
- ROE1EN 和 ROE0EN → RCSR[FEIE]
- TUE1EN 和 TUE0EN → TCSR[FEIE]
- TFSEN → TCSR[WSIE]

- RFSEN → RCSR[WSIE]
- RFF1EN 和 RFF0EN → RCSR[FRIE]
- TFE1EN 和 TFE0EN → TCSR[FRIE]

删除的位/字段：

- **TFRC** 和 **RFRC**：发送和接收帧完成
- **TLS** 和 **RLS**：发送和接收最后的时隙
- **RDR1** 和 **RDR0**：接收数据就绪
- **TDE1** 和 **TDE0**：发送数据为空
- **CMDAU**：命令地址寄存器已更新
- **CMDDU**：命令数据寄存器已更新
- **RXT**：接收标签已更新

相关中断使能位：

- TFRCEN 和 RFRCEN，
- TLSEN 和 RLSEN，
- RDR1EN 和 RDR0EN，
- TDE1EN 和
- TDE0E、CMDAUEN、CMDDUEN、RXTEN、RIE 和 TIE

3.7.2.3 发送和接收配置寄存器

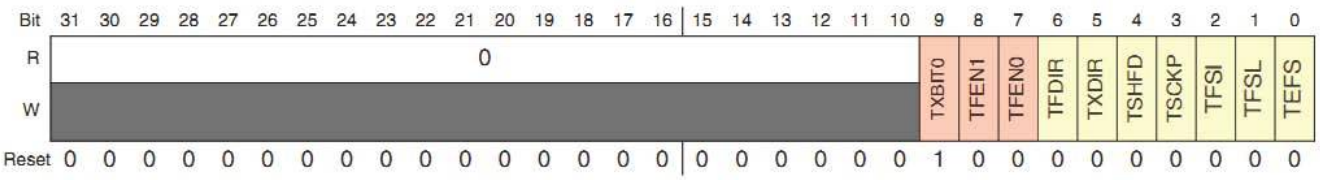


图 40. I2S0_TCR—SSI

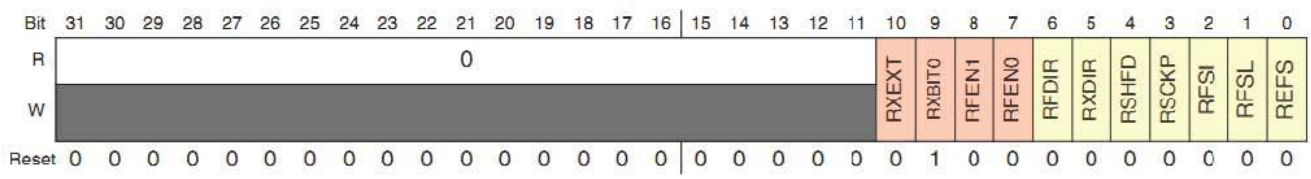


图 41. I2S0_RCR—SSI

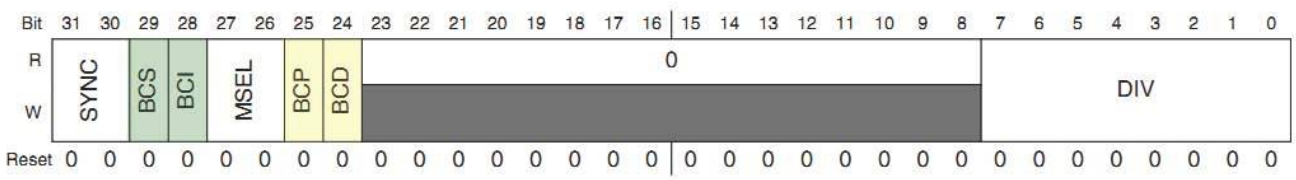


图 42. I2S0_TCR2 和 I2S0_RCR2—SAI

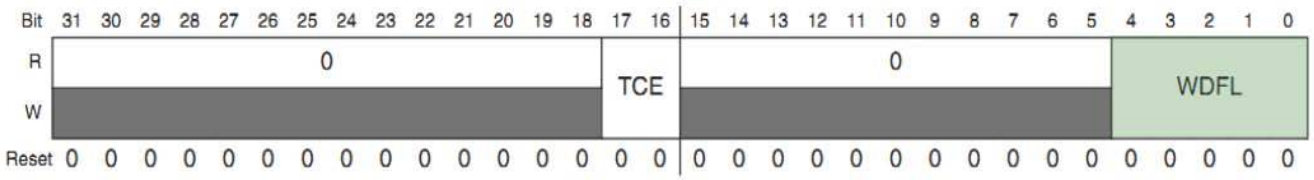


图 43. I2S0_TCR3—SAI

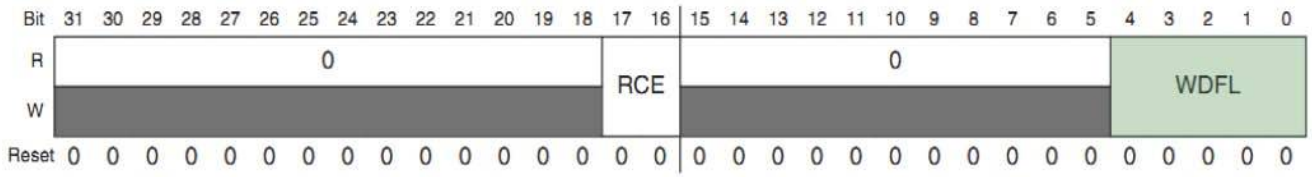


图 44. I2S0_RCR3—SAI

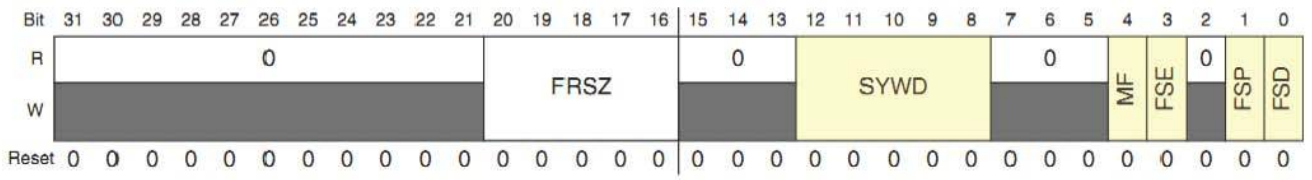


图 45. I2S0_TCR4 和 I2S0_RCR4—SAI

新增的位/字段:

- **BCS**：位时钟交换
- **BCI**：位时钟输入
- **WDFL**：配置字标志置位的起始字

改变的位/字段名称:

- TSCKP 和 RSCKP → BCP（位时钟极性）
- TXDIR 和 RXDIR → BCD（位时钟方向）
- TFSL 和 RFSL → SYWD（帧同步长度）
- TSHFD 和 RSHFD → MF（先发送 MSB 或 LSB）
- TEFS 和 REFS → FSE（帧同步过早）
- TFSI 和 RFSI → FSP（帧同步极性）
- TFDIR 和 RFDIR → FSD（帧同步方向）

删除的位/字段:

- **TXBIT0** 和 **RXBIT0**：MSB 对齐或 LSB 对齐
- **TFEN0** 和 **TFEN1**：发送 FIFO 使能
- **RXEXT**：接收符号扩展
- **RFEN0** 和 **RFEN1**：接收 FIFO 使能

3.7.2.4 发送和接收时钟配置寄存器

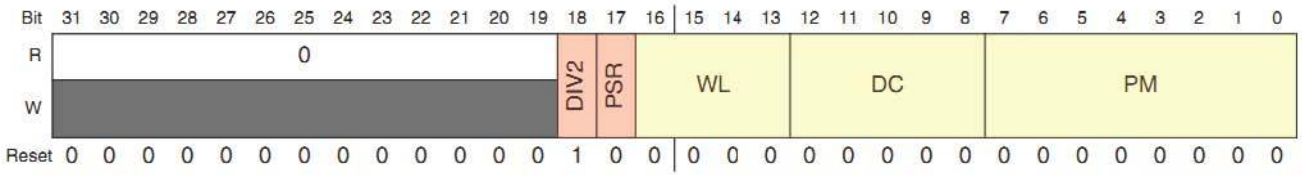


图 46. I2S0_TCCR 和 I2S0_RCCR—SSI

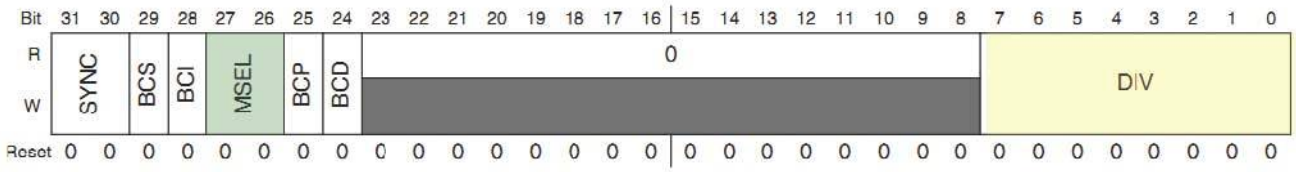


图 47. I2S0_TCR2 和 I2S0_RCR2—SAI

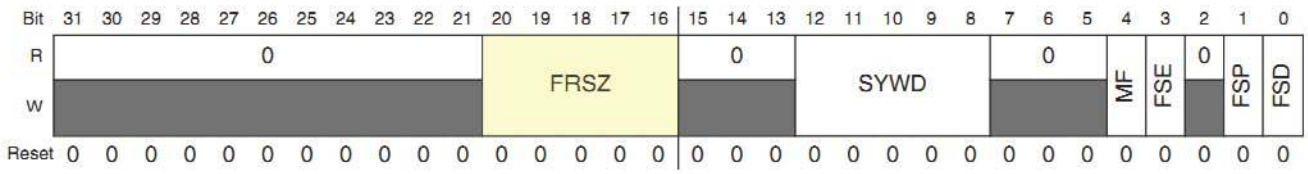


图 48. I2S0_TCR4 和 I2S0_RCR4—SAI

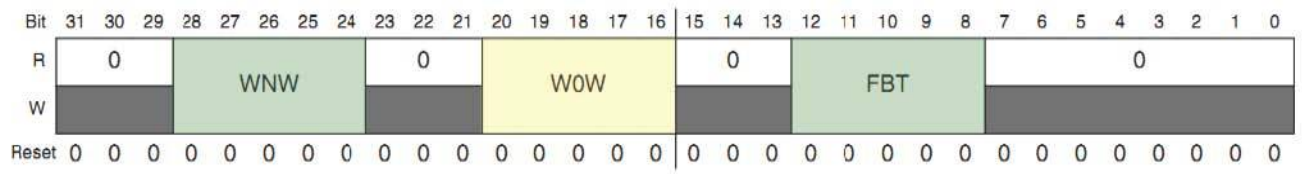


图 49. I2S0_TCR5 和 I2S0_RCR5—SAI

新增的位/字段:

- MSEL：在分频为位时钟前的主时钟选择
- WNW：配置每个字的位数（帧中的首个字除外）
- FBT：配置帧中的每个字中首个发送位的位索引

改变的位/字段名称:

- DC → FRSZ（每个帧中的字数）
- WL → W0W（每个字中的位数）
- PM → DIV（位时钟的预分频器）

删除的位/字段:

- DIV2：是否进行 2 分频
- PSR：预分频比为 8

3.7.2.5 FIFO 控制和状态寄存器

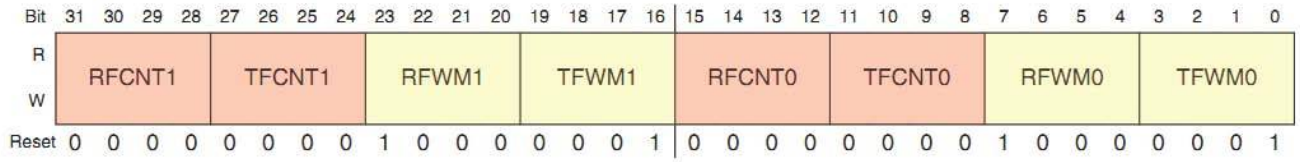


图 50. I2S0_FCSR—SSI

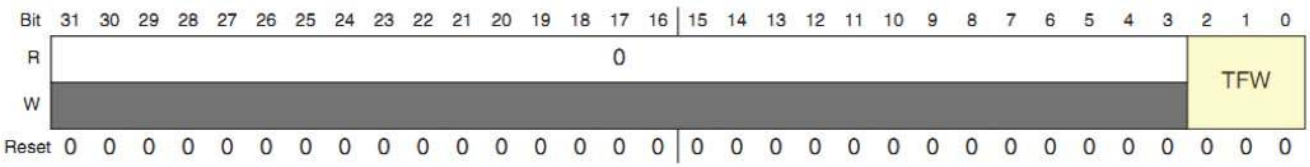


图 51. I2S0_TCR1—SAI

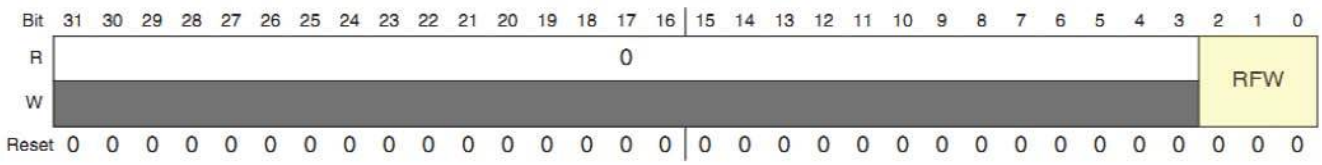


图 52. I2S0_RCR1—SAI

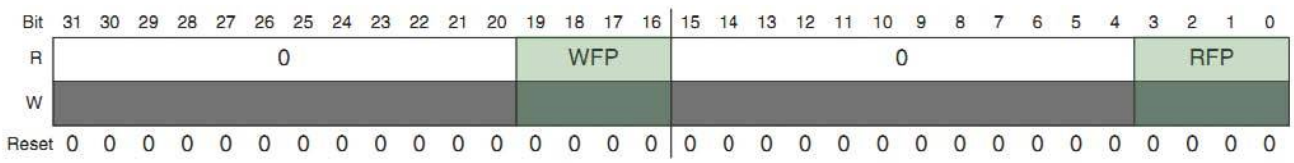


图 53. I2S0_TFR0、I2S0_TFR1、I2S0_RFR0 和 I2S0_RFR1—SAI

新增的位/字段:

- **WFP**: 写入 FIFO 指针
- **RFP**: 读取 FIFO 指针

改变的位/字段名称:

- RFWM0 和 RFWM1 → RFW（接收 FIFO 水印）
- TFWM0 和 TFWM1 → TFW（发送 FIFO 水印）

删除的位/字段:

- RFCNT0 和 RFCNT1: 接收 FIFO 计数器
- TFCNT0 和 TFCNT1: 发送 FIFO 计数器

3.7.2.6 主时钟生成寄存器

图 54. SIM_SOPT2—SSI

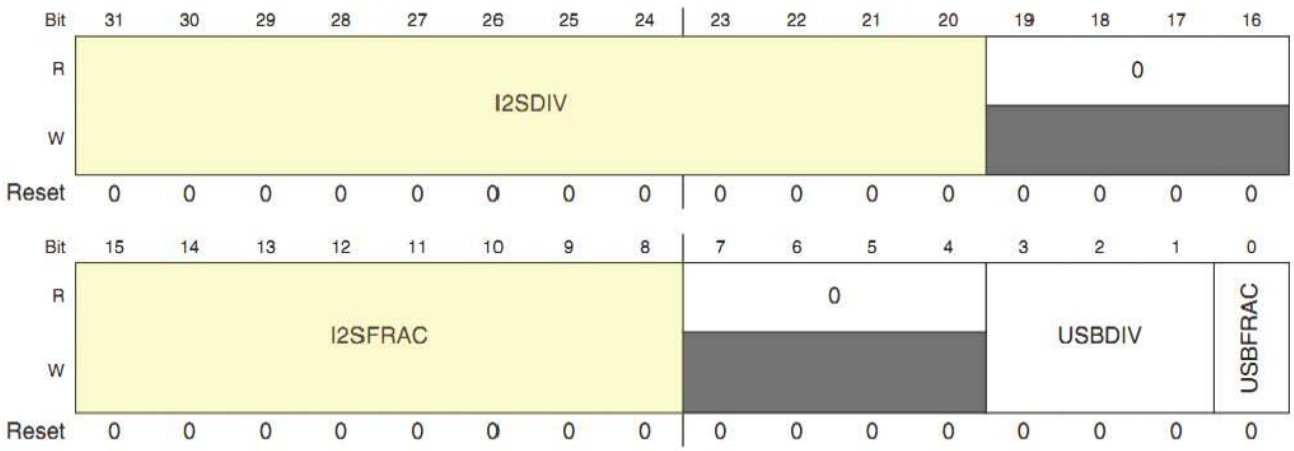


图 55. SIM_CLKDIV2—SSI

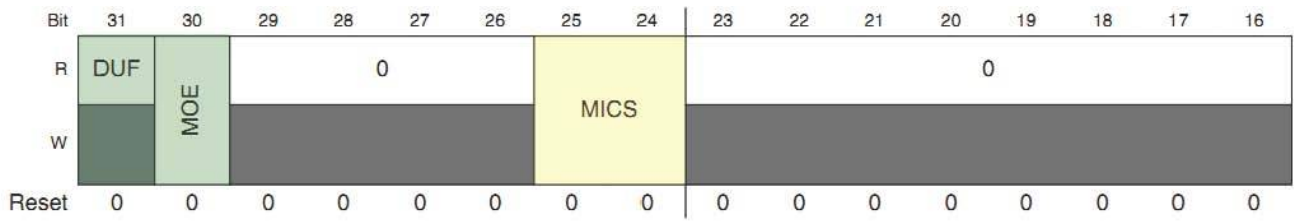


图 56. I2S0_MCR—SAI

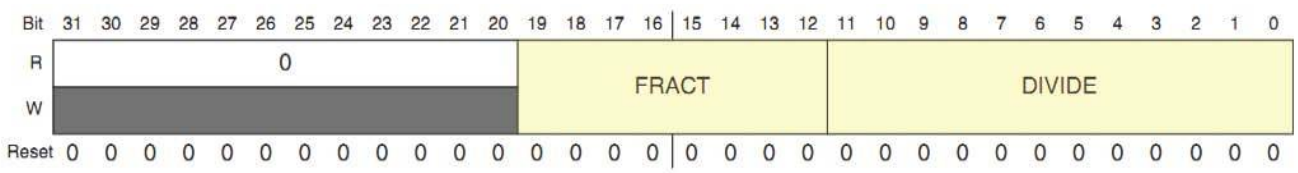


图 57. I2S0_MDR—SAI

新增的位/字段:

- DUF：分频更新标志
- MOE：主时钟输出使能

改变的位/字段名称:

- I2SSRC → MICS（主时钟源选择）
- I2SFRAC → FRACT（时钟分频比小数）

3.7.3 软件影响

对于 SAI 模块，无需修改太多的软件配置。位时钟与帧同步生成、其方向和极性选择、帧大小，以及字大小设定均与 SSI 模块相同。只有位命名不同。

要通过 TX FIFO 空或 RX FIFO 满条件来触发 DMA，可以方便地将 DMA 控制器编程为自动传输数据，这只需简单地将总传输大小编程为与 TX 和 RX FIFO 深度一致即可实现。如果要将 FIFO 水印条件用于触发 DMA，注意将 DMA 控制器编程为总传输大小< (FIFO 深度 – FIFO 水印)；否则，TX FIFO 将接收超过其容量的数据，或者 RX FIFO 将发送超出其接收的数据。

另外需要注意的是，不再有任何 FIFO 计数信息；取而代之的是写入和读取 FIFO 指针，因此实际的 FIFO 条目需要在您的代码中计算。此外，您还需要包含逻辑以确定 FIFO 为空或满，并且当 FIFO 为满时停止写入数据，或当 FIFO 为空时停止读取数据。

3.7.4 硬件影响

从 SSI 移植到 SAI 并无太多硬件改变。只是多了一个数据通道用于发送和接收，因此现在的数据通道有 I2S0_TXD0 和 I2S0_TXD1，以及 I2S0_RXD0 和 I2S0_RXD1。使用通道 0 或 1 进行发送或接收数据并无区别。由于每个通道自身具备 FIFO，因此每个通道可以发送或接收相同或不同的数据，并且可以同时进行且共用同一个帧同步和位时钟。

3.8 TSI 版本 1 到 TSI 版本 2

本章节描述了 TSI 模块的版本 1 和版本 2 之间的具体差异。TSI 模块在新版本中进行了简化，从而简化了配置。振荡器范围保持不变，但是可配置值的数量减半。取消了增量电压配置；现在使用一个常数增量电压。在每个通道上去除了用于当电容增加时产生中断的阈值寄存器；作为例外，仅保留一个寄存器用于低功耗唤醒通道。与该变化相关，也去除了用于所有这些阈值的状态寄存器；现在阈值仅用于唤醒。以下章节详细说明了针对这些差异需要在软件中进行的调整。

3.8.1 存储空间映射比较

	Kinetis 版本 1.x 100 MHz (TSI 1.0)		Kinetis 100 MHz 版本 2.x120 MHz72MHz50MHz (TSI 2.0)	
	位置	名称	位置	名称
通用控制和状态寄存器	0x4004_5000	TSI0_GENCS	0x4004_5000	TSI0_GENCS
扫描控制寄存器	4004_5004	TSI0_SCANC	4004_5004	TSI0_SCANC
引脚使能寄存器	4004_5008	TSI0_PEN	4004_5008	TSI0_PEN
状态寄存器	4004_500C	TSI0_STATUS	N/A	N/A
计数寄存器	4004_5100–4004_511C	TSI0_CNTRn	4004_5100–4004_511C	TSI0_CNTRn
通道阈值寄存器	4004_5120–4004_515C	TSI0_THRESHLDn	4004_5120	TSI0_THRESHOLD
唤醒计数器	N/A	N/A	4004_500C	TSI_WUCNTR

- **GENCS**：无变化，但是超出范围设定仅在低功耗模式下有效。

- **SCANC**：删除了 **CAPTRM**（内部电容现在固定为 1 pF），**REFCHRG** 和 **EXTCHRG** 从 5 位缩小为 4 位。改变包括将可用的电流值数量从 32 降低到 16。电流范围为 2 µA 至 32 µA，增量为 2。删除了 **DELVOL**，增量电压为固定值。改变了 AMCLKS 时钟源：总线时钟参考改为 **LPOCLK**。还删除了 **AMCLKSDIV**，因为它仅在总线时钟作为参考时有用。除了分频时钟源，现在还提供一个更慢、功耗更低的时钟源。这降低了模块的总功耗。如果需要较慢的时钟源，使用 **LPOCLOCK**，并用 **AMPSC** 进行调节。如果需要较快的时钟源，使用 **MCGIRCLK** 或 **OSCERCLK**。

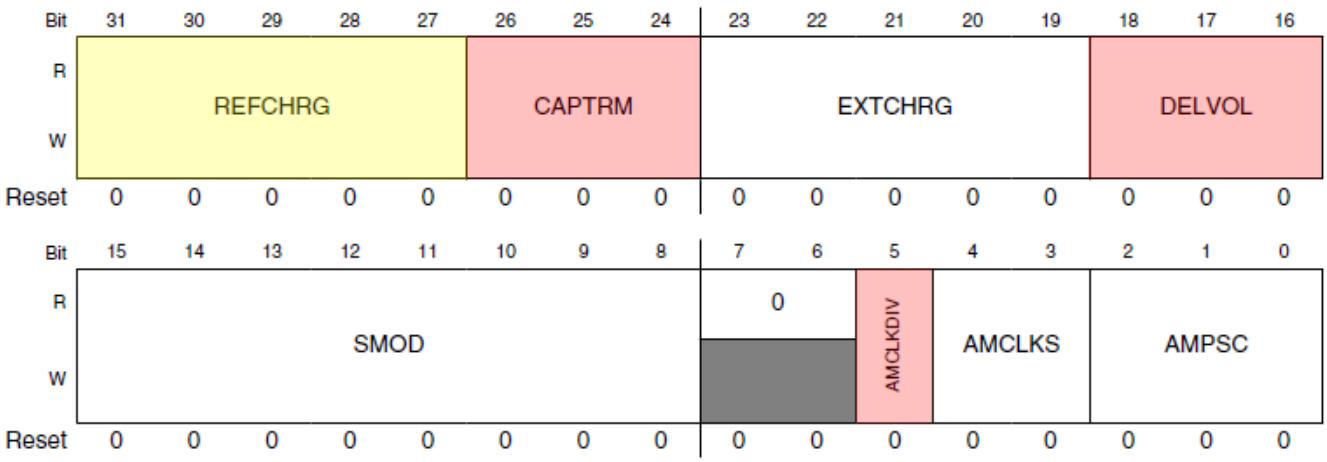


图 58. TSI_SCANC—TSI 版本 1

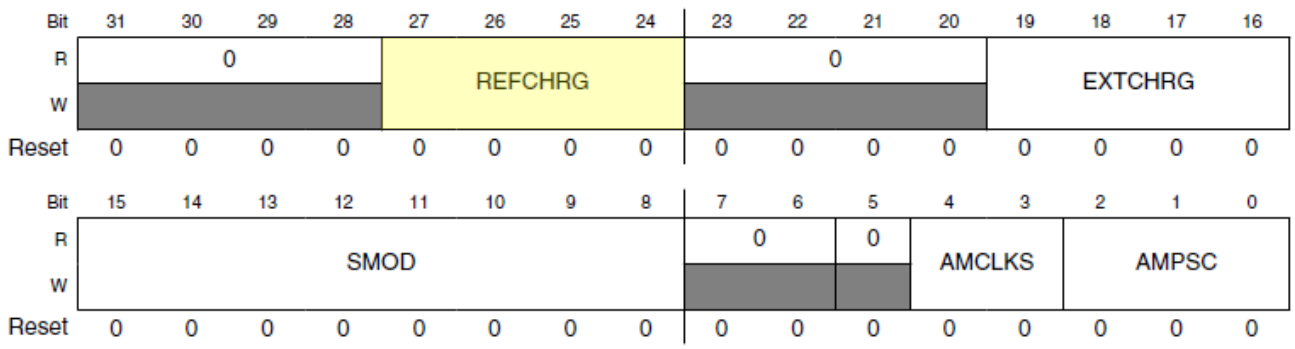
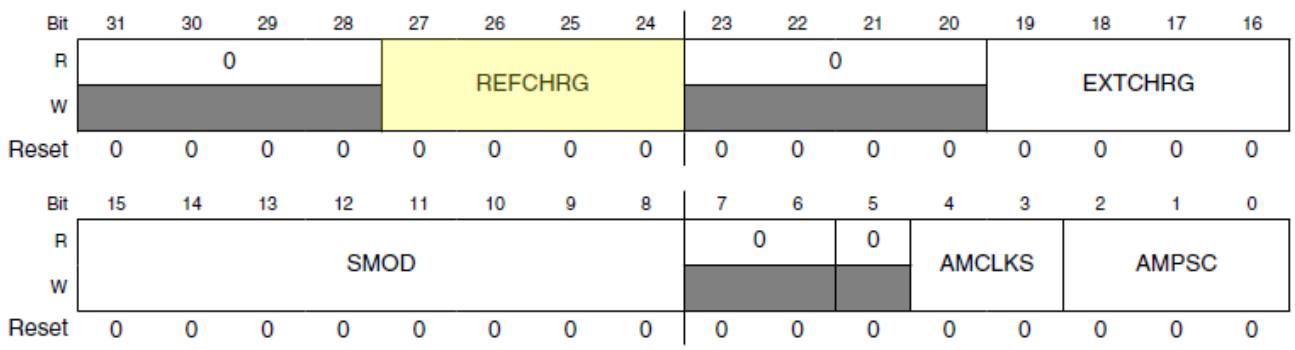


图 59. TSI_SCANC—TSI 版本 2

- **TSI_PEN**：无变化
- **TSI_WUCNTR**：新寄存器，存放用于低功耗电极唤醒的计数器值。



- **TSI_STATUS**：删除

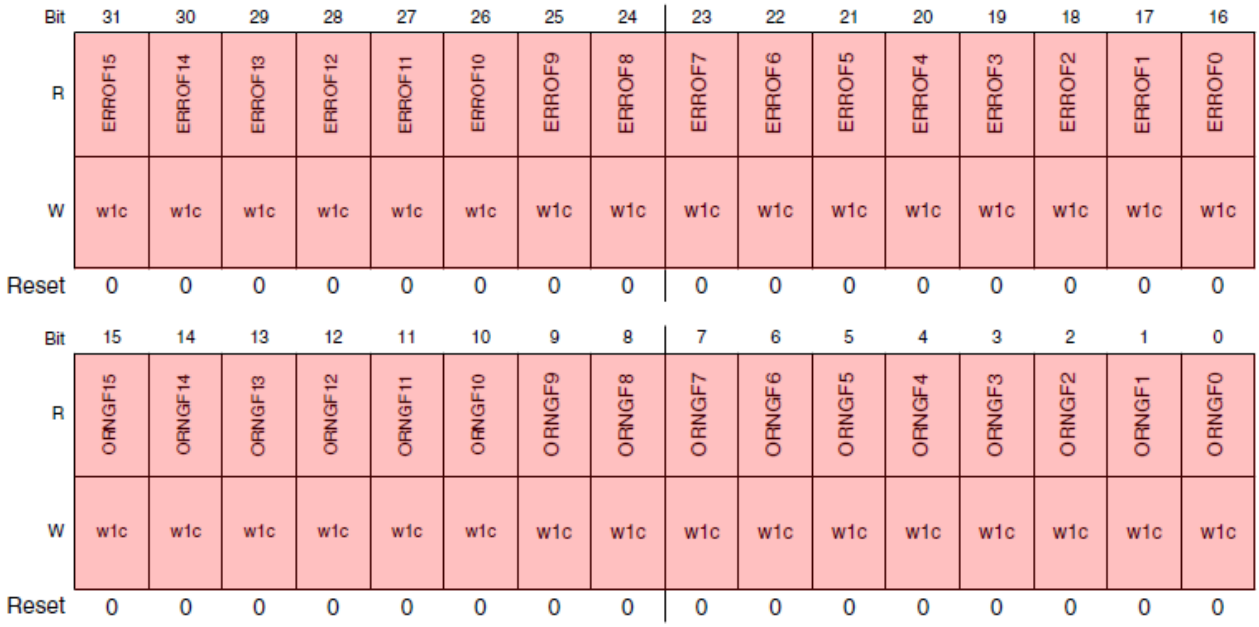


图 60. TSI_STATUS（完全删除）

- **TSI_CNTR**：无变化
- **TSI_THRESHLDn**：以前是每个电极一个阈值寄存器；现在只有一个阈值寄存器用于指定的唤醒电极，因为现在超出范围功能只在低功耗模式下工作。

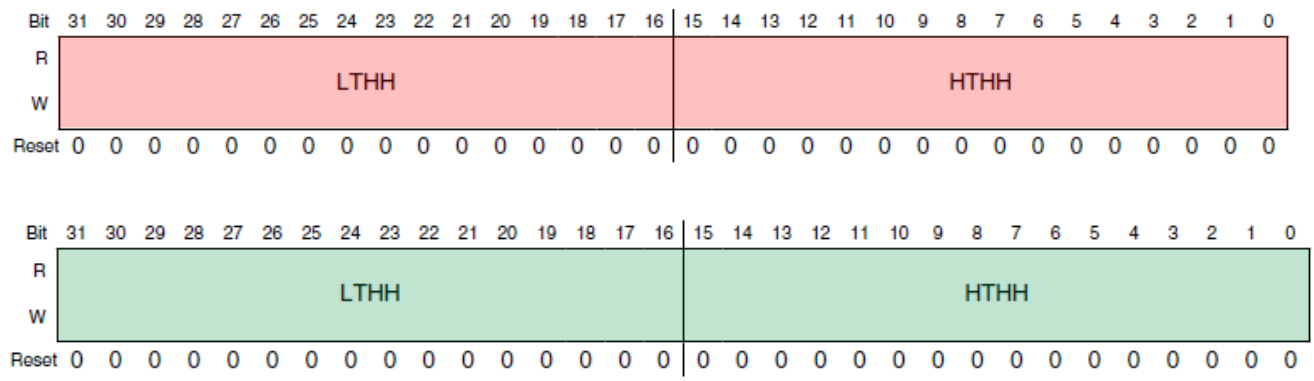


图 61. 单个 TSI_THRESHOLD 用于唤醒，以取代用于每个通道的独立寄存器

3.8.2 软件影响

- 无增量电压寄存器，因为内部与外部增量电压没有区别，不会影响频率关系。可输出的最高电压选择为固定电压以增加 EMC 耐受性。对于增量电压的改变不会影响振荡器之间的关系，而对于计数器，除了需要删除配置该寄存器所用的代码外，对软件也没有影响。
- 外部和内部振荡器的电流变小。触摸传感不需要像 TSI 版本 1 中的精细度，因而在新版本中将可选值数量下降为 16 个。需要重点注意的是，这个变化会导致 EXTCHRG 和 REFCHRG 寄存器从 5 位变为 4 位，因而有必要缩小数值。请务必注意此项修改。如果先前使用的值不再有效，可以根据方便选择相邻的较大或较小值。
- 将用于在 AMCLKS 中作为时序参考的总线时钟改为 LPOCLK，意味着需要将预分频器（AMPSC）和模数寄存器（SMOD）的时序参考调整为 LPOCLK，即 1 kHz 时钟参考。如果使用了其他两个时钟源（MCGIRCLK 或 OCSERCLK），则无需修改。

- 只有一个阈值寄存器，而不是每个计数器一个阈值寄存器。考虑到唤醒源可从 EOSF（扫描结束）或超出范围唤醒。由于超出范围只在低漏电模式（LLS 和 VLLSx）下可用，建议在这两个模式下使用 TSI 唤醒。如果需要其他模式（如停止、低功耗停止和等待等），并且需要通过 TSI 唤醒，则需要具有一个周期性唤醒源来检查 TSI 状态，这很重要。周期性唤醒可以是外部源，如 RTC 或 MTIM，也可以是 TSI 模块（通过将扫描时间配置的非常慢，如每 2 或 3 秒）。这将导致反应时间变慢，不过功耗也更低。
- 删除对 TSI_STATUS 的引用，因为已不可用。
- 删除对多个 TSI_THRESHLDn 寄存器的引用；请记住现在只有一个具有 OURGF 标志和中断的 TSI_THRESHOLD 寄存器用于唤醒。
- 如果需要唤醒后的第一个计数器值（来自配置的唤醒电极），使用 TSI_WUCNTR 读取该值。如果不需要，则等待下次扫描并直接从计数器寄存器中读取。

3.8.3 硬件影响

无硬件影响。信号处理和测量算法均相同。

4 增强型模块

4.1 系统集成模块（SIM）

4.1.1 受影响的寄存器

SIM 存储空间映射仅增加了少量的寄存器，但是对现有寄存器中的字段进行了较多修改。增加了与新模块控制相关的内容、更多的模块实例化和新功能。少量修改了一些字段功能。将一些控制功能转移到其他模块中，因此删除了与这些功能相关的 SIM 位。

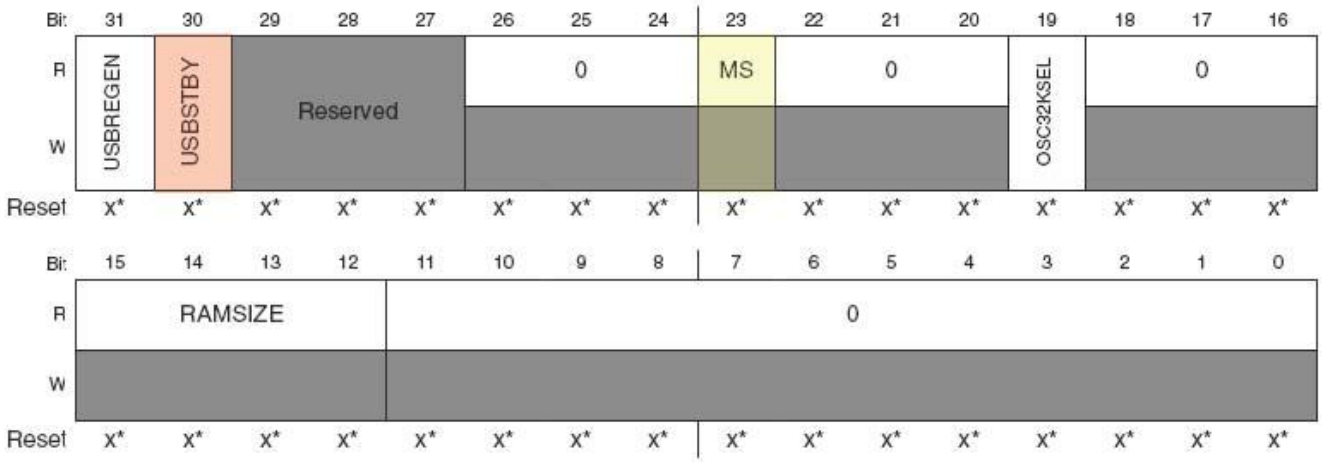
下表显示了 Kinetis 100 MHz 版本 1.x 器件中的原始 SIM 模块与 120 MHz 器件中的新 SIM 模块在存储空间映射方面的差异。以下章节提供了关于寄存器内部字段变化的信息。

表 9. 存储空间映射比较

位置	Kinetis 100 MHz 版本 1.x SIM	Kinetis 120 MHz SIM
0x40047004	N/A	SOPT1 配置寄存器(SIM_SOPT1CFG)
0x40048064	N/A	系统时钟分频器寄存器 3 (SIM_CLKDIV3)
0x40048068	N/A	系统时钟分频器寄存器 4 (SIM_CLKDIV4)
0x4004806C	N/A	其他控制寄存器(SIM_MCR)

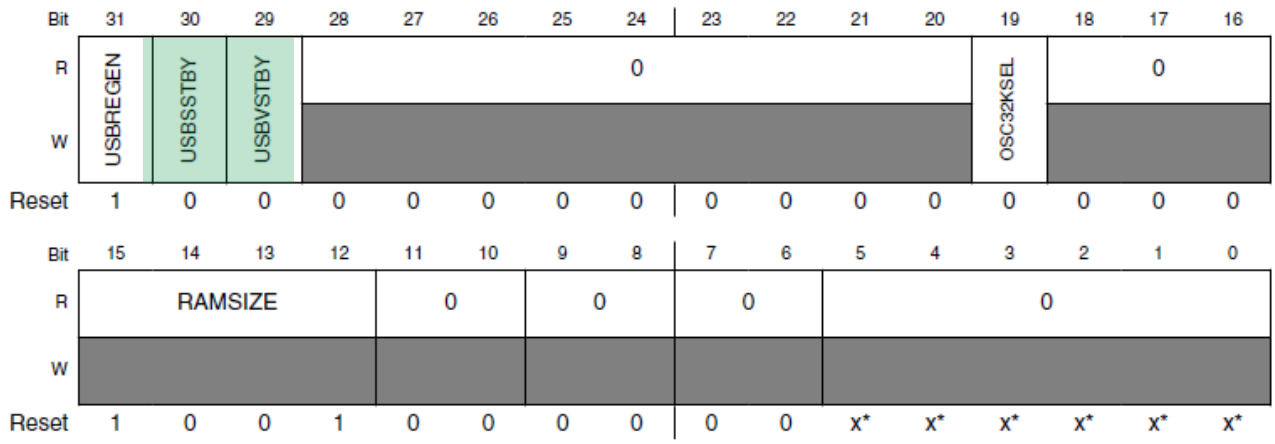
4.1.1.1 SIM SOPT 寄存器变化

以下显示了 SIM 系统选项寄存器（SOPT）发生的变化。



* Notes:
- x = Undefined at reset.

图 62. SIM_SOPT1 寄存器—Kinetis 100 MHz 版本 1.x



* Notes:
- x = Undefined at reset.

图 63. SIM_SOPT1 寄存器—Kinetis 120 MHz

新增的位/字段:

- USBSSTBY
- USBVSTBY

删除的位/字段名称:

- USBSTBY

改变的位/字段名称:

- MS → EZP_MS 移动到 RCM_MR 寄存器

Address: RCM_MR is 4007_F000h base + 7h offset = 4007_F007h

Bit	7	6	5	4	3	2	1	0
Read	0						EZP_MS	0
Write								
Reset	0	0	0	0	0	0	0	0

图 64. 模式寄存器(RCM_MR)—Kinetis 120 MHz

Address: SIM_SOPT1CFG is 4004_7000h base + 4h offset = 4004_7004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																0								0				0			
W						USSWE	UVSWE	URWE																								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

图 65. SIM_SOPT1CFG—Kinetis 120 MHz

SIM_SOPT1CFG 是一个新寄存器，增加了对于 USB 稳压器操作的控制。更多详情，请参见 USB 章节。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		SDHCSRC		0		I2SSRC		0		TIMESRC		0	USBSRC	0	PLLFLLSEL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			TRACECLKSEL	CMTUARTPAD	0	FBSL		0							MCGCLKSEL
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

图 66. SIM_SOPT2—Kinetis 100 MHz 版本 1.x

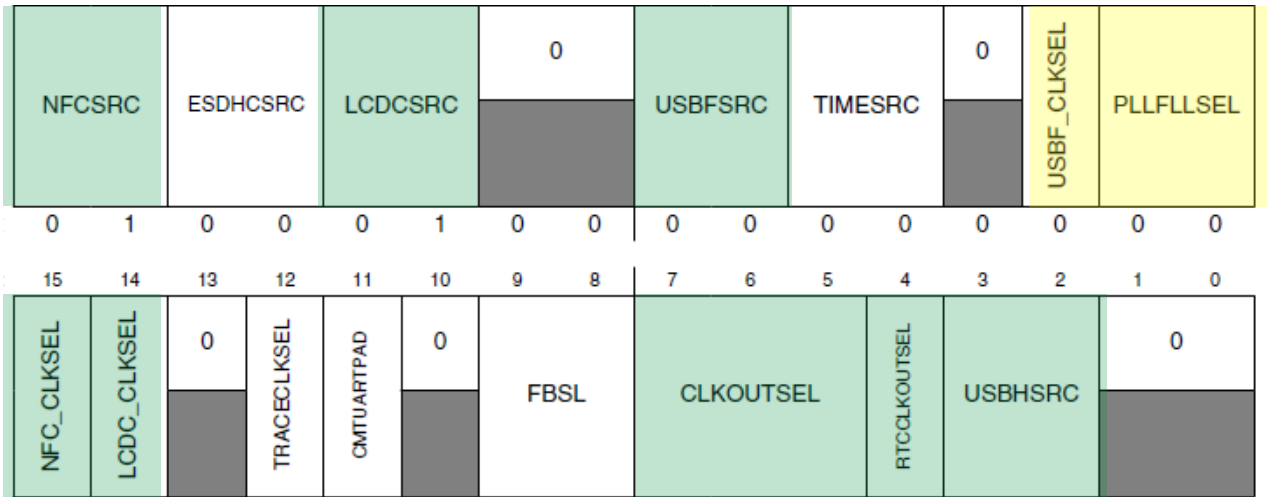


图 67. SIM_SOPT2—Kinetis 120 MHz

新增的位/字段:

- NFCSRC - 选择用于新 NFC 模块的时钟分频源
- LCDCSRC - 选择用于新 LCDC 模块的时钟分频源
- USBFSRC - 用于新 USBFS 小数分频器的时钟选择
- NFC_CLKSEL - 在 NFC 时钟分频器和 EXTAL1 时钟之间进行选择
- LCDC_CLKSEL - 在 LCDC 时钟分频器和 EXTAL1 时钟之间进行选择
- CLKOUTSEL - 选择输出到 CLKOUT 引脚上的内部时钟
- RTCCLKOUTTSEL - 选择输出到 RTC_CLKOUT 引脚上的时钟（32 kHz 或 1 Hz）
- USBHSRC - 选择用于新 USB HS 模块的时钟分频源

删除的位/字段名称:

- I2SSRC
- MCGCLKSEL

改变的位/字段名称:

- USBSRC → USBFSRC 位字段重命名，以区别 USB FS 和 USB HS
- PLLFLLSEL → PLLFLLSEL 字段扩展为 2 位，以允许选择新的内部时钟。两个版本间的编码互相兼容，因而无需修改代码以保持相同的时钟选择。

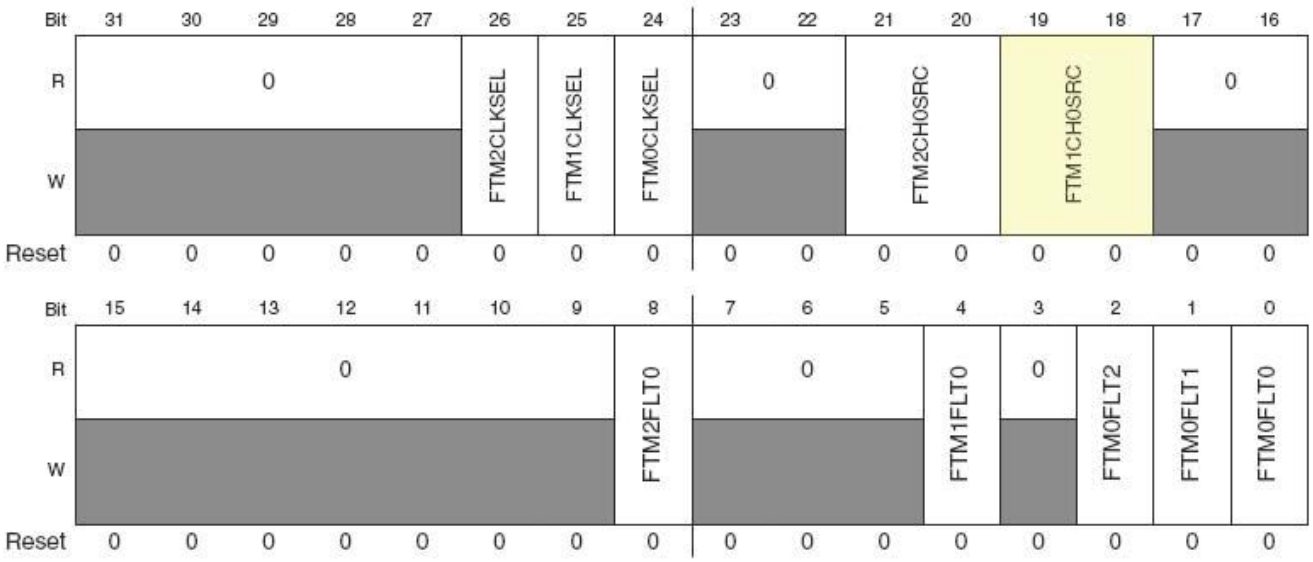


图 68. SIM_SOPT4—Kinetis 100 MHz 版本 1.x

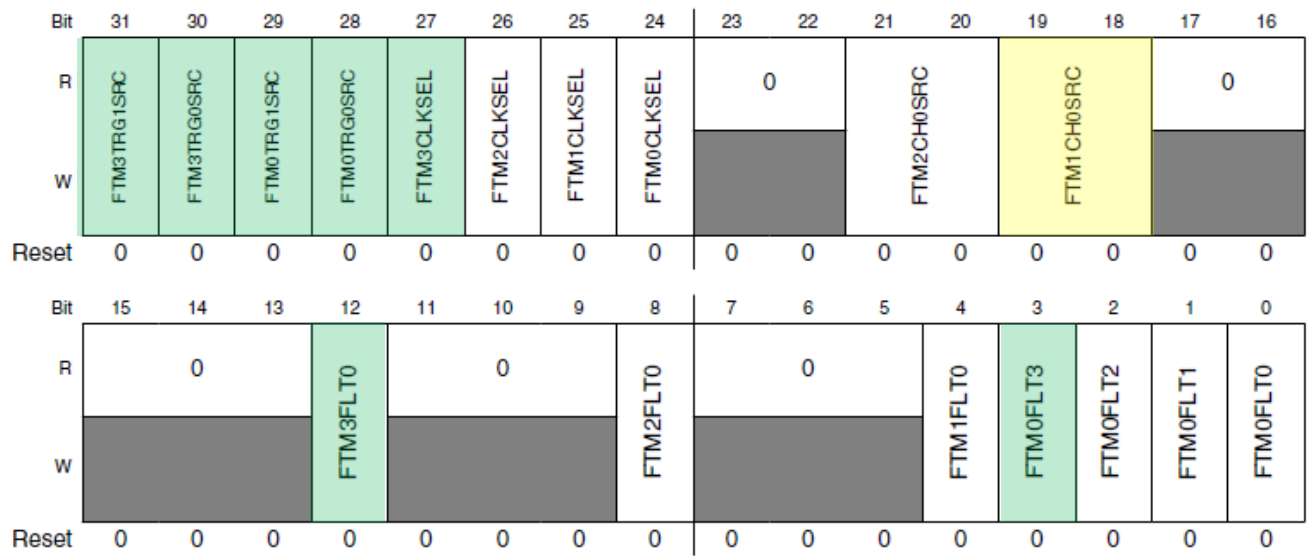


图 69. SIM_SOPT4—Kinetis 120 MHz

新增的位/字段:

- FTM3TRG1SRC、FTM3TRG0SRC、FTM0TRG1SRC 和 FTM0TRG0SRC - 新硬件触发源选择
- FTM3CLKSEL - 用于新 FTM 实例化的时钟选择
- FTM3FLT0 - 用于新 FTM 实例化的故障选择
- FTM0FLT3 - FTM0 故障 3 选择

改变的位/字段名称:

- FTM1CH0SRC → 添加用于 USB 帧起始脉冲的选项

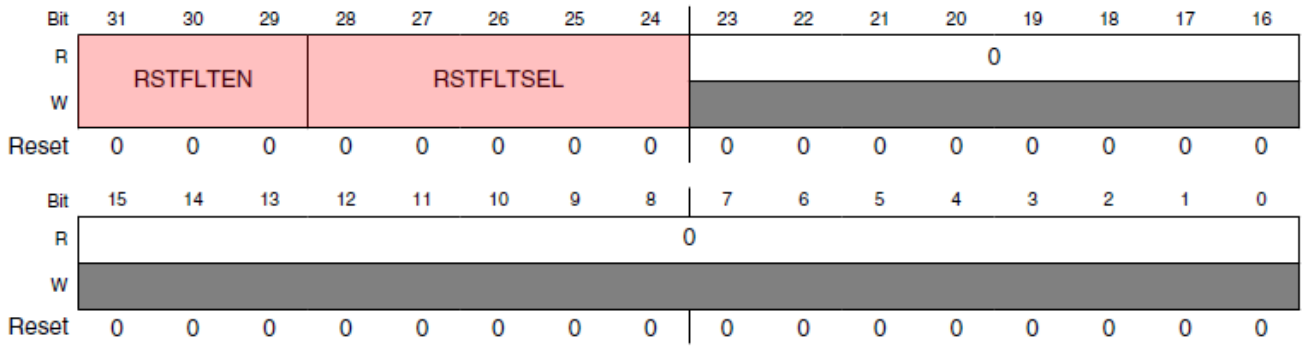


图 70. SIM_SOPT6—Kinetis 100 MHz 版本 1.x

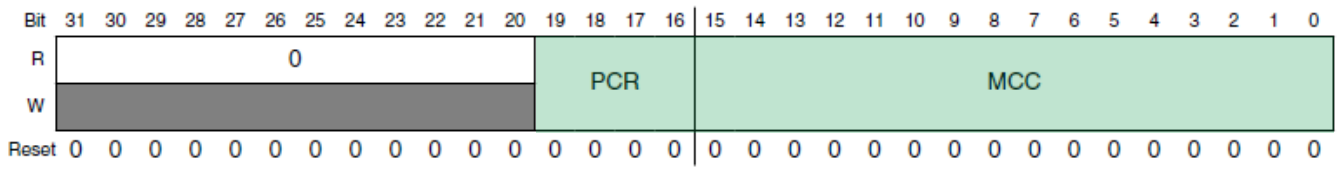


图 71. SIM_SOPT6—Kinetis 120 MHz

新增的位/字段:

- PCR 和 MCC - 这些字段控制共用引脚（NFC 和 FlexBus）的保持时间

删除的位/字段名称:

- RSTFLTEN 和 RSTFLTSEL - 复位引脚滤波器功能移至 RCM
- MCGCLKSEL

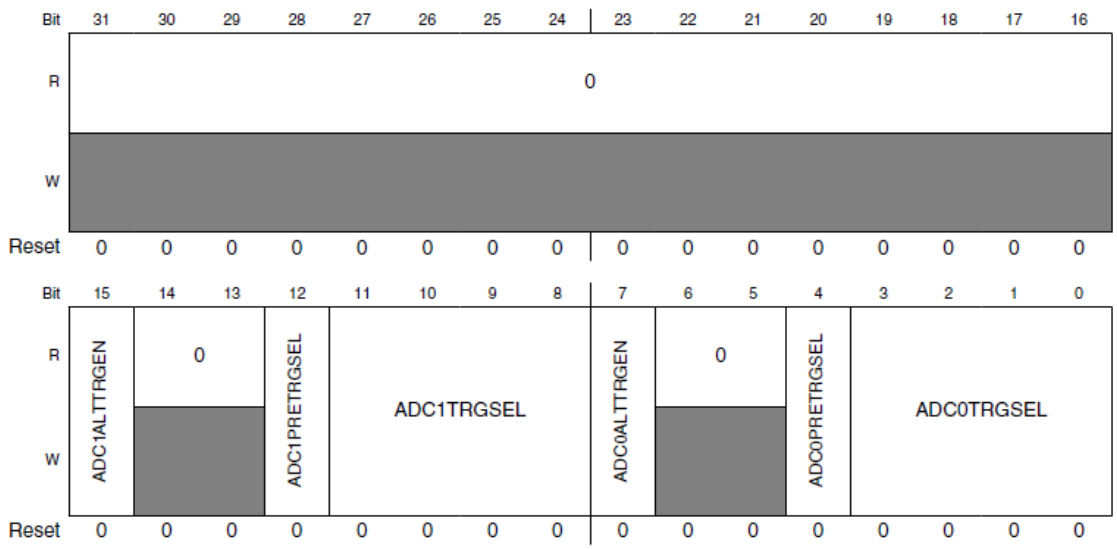


图 72. SIM_SOPT7—Kinetis 100 MHz 版本 1.x

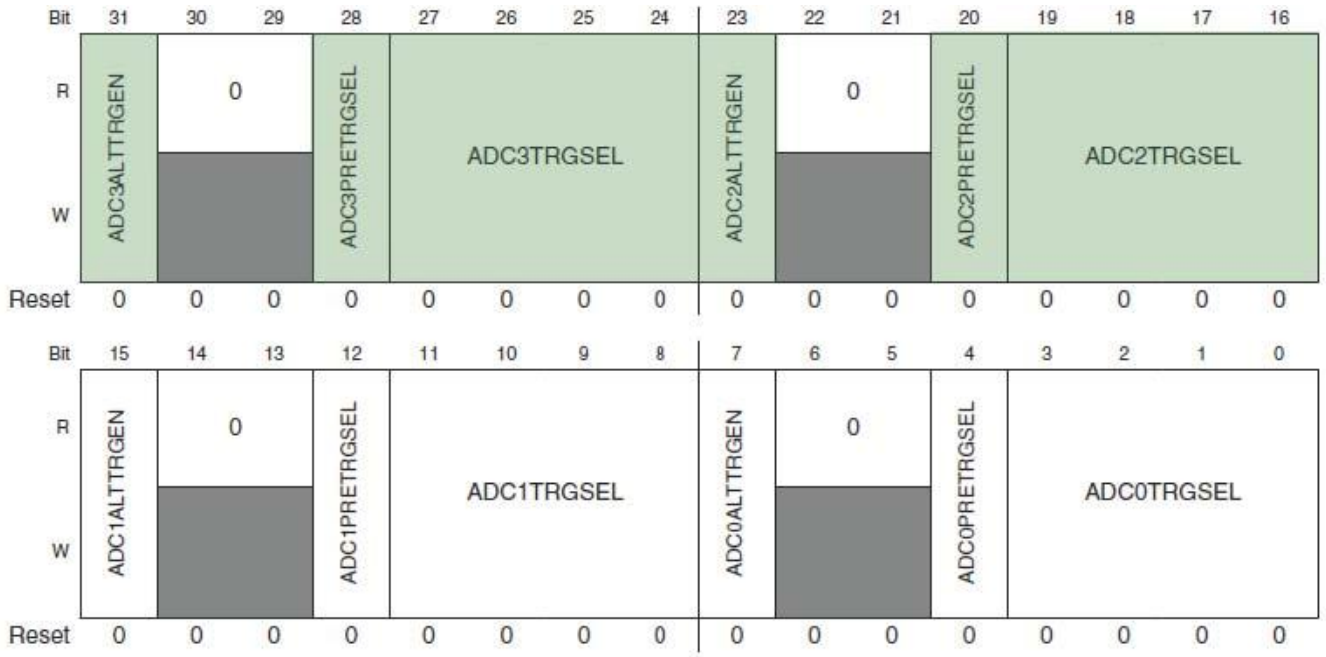


图 73. SIM_SOPT7—Kinetis 120 MHz

新增的位/字段:

- ADC3ALTTRGEN、ADC3PRETRGSEL 和 ADC3TRGSEL - 添加了用于控制新 ADC 实例化的位
- ADC2ALTTRGEN、ADC2PRETRGSEL 和 ADC2TRGSEL - 添加了用于控制新 ADC 实例化的位

4.1.1.2 SIM SCGC 寄存器变化

SIM 模块具有多个系统时钟门控寄存器。向大多数寄存器中新增了位以向新模块添加时钟门控位，Kinetis 100 MHz 版本 1.x 器件不含这些位。时钟门控寄存器向后兼容，因而只有在使用新模块和/或新模块实例化时才要求修改时钟门控。对于在 Kinetis 100 MHz 版本 1.x 器件上可用的模块，无需修改代码以使能时钟门控。

4.1.1.3 SIM CLKDIV 寄存器变化

SIM 模块还为新模块添加了时钟分频器，这些模块包括 USB HS、LCDC 和 NFC。可以在 SIM CLKDIV*n* 寄存器中找到控制这些时钟分频器的字段。这些字段与新模块相关，因而只有当使用这些新模块时才需要修改这些寄存器。

在先前已有的模块中，只有跟踪时钟在 CLKDIV 寄存器中有相关的新位。由于 CPU 频率变快，在 120 MHz Kinetis 器件中添加了跟踪时钟。默认情况下，跟踪时钟为 2 分频。

4.1.1.4 SIM 其他控制寄存器

SIM 模块中最后一个受影响的寄存器是其他控制寄存器（MCR）。该模块具有以下用途的字段:

- 跟踪时钟使能/禁止
- ULPI 时钟方向
- PDB 循环模式
- LCD 显示开始/停止
- DDR RCR 复位状态
- DDR RCR 复位请求
- DDR 管脚配置
- DDR DQS 使能
- DDR I/O 使能/禁止

- DDR 自刷新状态
- DDR 自刷新使能

4.1.2 软件影响

SIM 负责控制和配置许多模块特性，因此许多应用可能会受到 SIM 模块变化的影响。虽然一些应用可能无需任何的软件修改，SIM 仍然是最有可能需要软件更新的模块之一。实际所需的修改大部分取决于所使用的模块，以及那些模块所要求的配置和时钟。为了使用这些新模块，将需要对 SIM 时钟门控和与新模块相关的任何控制字段进行初始化。

在低功耗模式下运行具有 DDR 存储器的 Kinetis MCU 时，VLPS、VLPR、STOP、LLS 和 VLLSx 需要使用 SIM_MCR 和 DDR 存储器控制器设定，以使 DDR 控制器和存储器消耗的功耗最低。应适当设置在 SIM_MCR 寄存器中用于控制管脚和模拟电路配置的位，以在低功耗模式下使漏电流最小。外部 DDR 存储器器件也应置于 DDR 低功耗模式。关于这些模式的详细信息，请参见参考手册中的 SIM_MCR 寄存器描述和 DDR 存储器控制器章节。

4.1.3 硬件影响

SIM 不直接与外部信号相关，因而 SIM 的变化不要求修改硬件。然而，一些受 SIM 变化影响的模块可能需要更新硬件。

4.2 直接存储器访问控制器（eDMA）

4.2.1 特性

Kinetis 100 MHz 器件在 EDMA 上支持 16 个通道。由于 120 MHz 器件提高了片上集成度，支持的 DMA 通道数增加到了 32 个。

4.2.2 受影响的寄存器

为了支持新的 DMA 通道，对 eDMA 的存储空间映射做了一些修改。所有修改均为新增。之前在 100 MHz Kinetis 器件中存在的位保持不变。

下表显示了添加到 eDMA 中的新 eDMA 传输控制描述符（TCD）寄存器，用于支持通道数的增加。以下列出了增加的寄存器字段。

表 10. 存储空间映射比较

寄存器	16 通道 eDMA 位置	32 通道 eDMA 位置
DMA_TCD*n*_SADDR	0x4000_9000 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9000 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_SOFF	0x4000_9004 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9004 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_ATTR	0x4000_9006 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9006 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_NBYTES	0x4000_9008 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9008 + (32d x *n*)，其中 *n* = 0d 至 31d

下一页继续介绍此表...

表 10. 存储空间映射比较 (继续)

寄存器	16 通道 eDMA 位置	32 通道 eDMA 位置
DMA_TCD*n*_SLAST	0x4000_900C + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_900C + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_DADDR	0x4000_9010 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9010 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_DOFF	0x4000_9014 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9014 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_CITER	0x4000_9016 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9016 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_DLASTSGA	0x4000_9018 + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_9018 + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_CSR	0x4000_901C + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_901C + (32d x *n*)，其中 *n* = 0d 至 31d
DMA_TCD*n*_BITER	0x4000_901E + (32d x *n*)，其中 *n* = 0d 至 15d	0x4000_901E + (32d x *n*)，其中 *n* = 0d 至 31d

4.2.2.1 DMA_ERQ

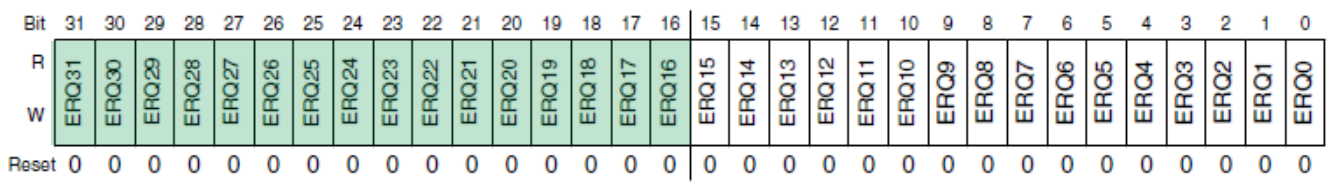


图 74. DMA_ERQ—Kinetis 120 MHz

新增的位/字段：

- ERQ31-ERQ16

4.2.2.2 DMA_EEI

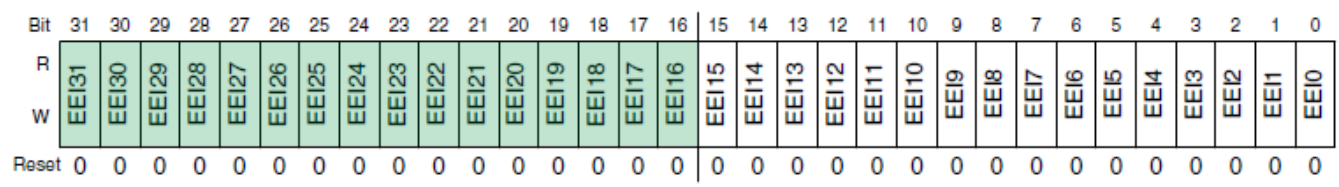
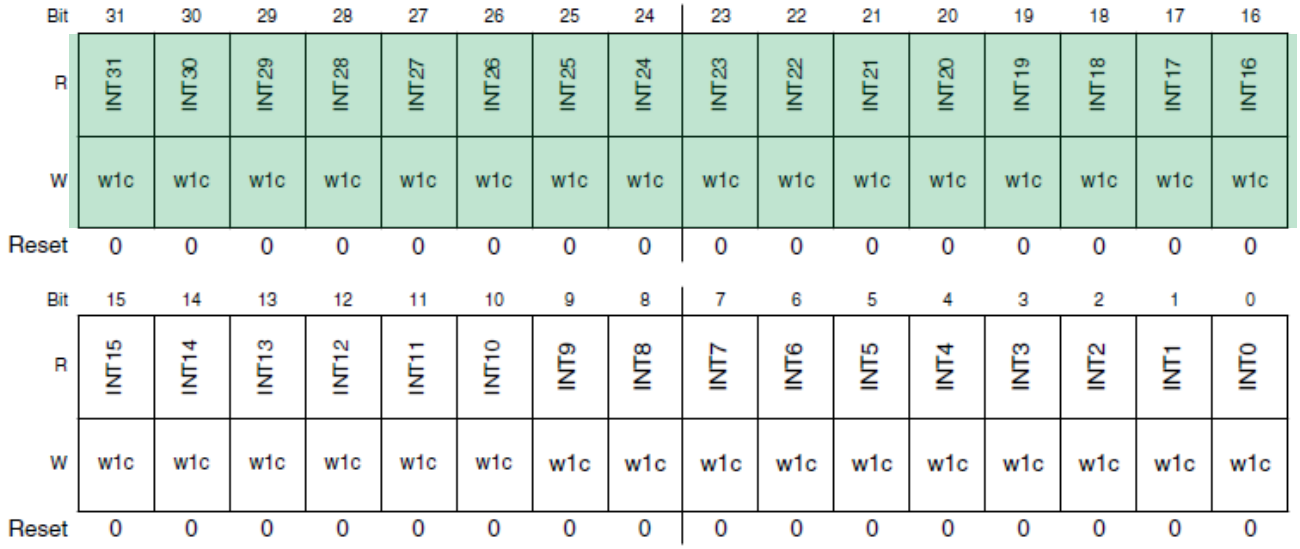


图 75. DMA_EEI—Kinetis 120 MHz

新增的位/字段：

- EEI31-EEI16

4.2.2.3 DMA_INT



Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INT31	INT30	INT29	INT28	INT27	INT26	INT25	INT24	INT23	INT22	INT21	INT20	INT19	INT18	INT17	INT16
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

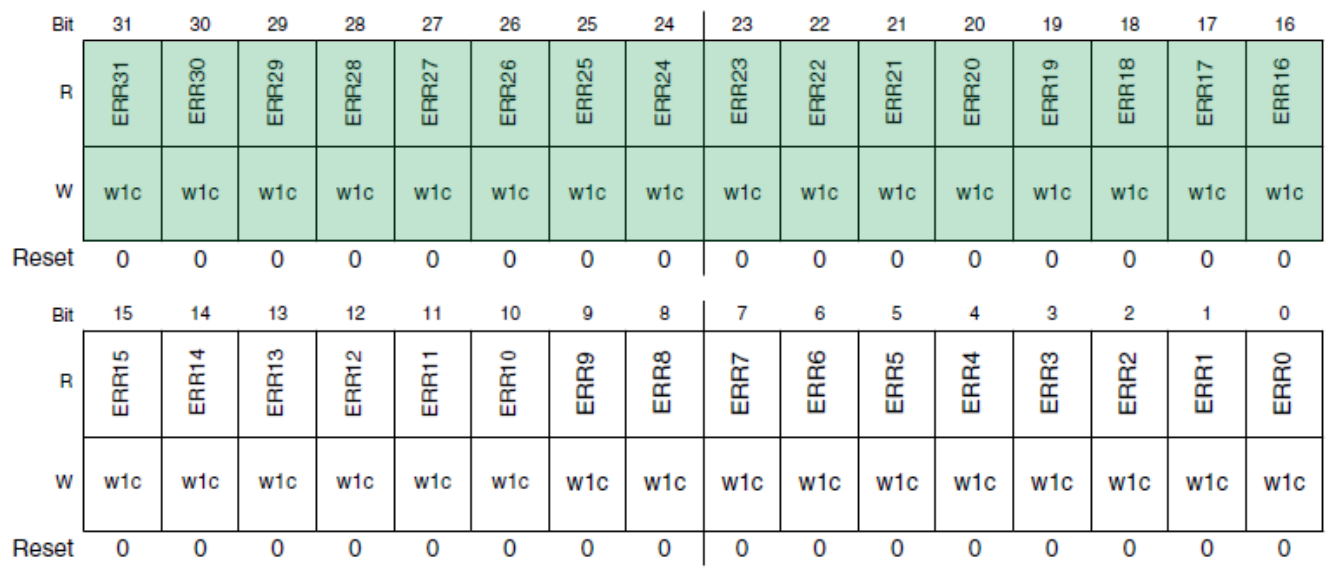
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INT15	INT14	INT13	INT12	INT11	INT10	INT9	INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1	INT0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

图 76. DMA_INT—Kinetis 120 MHz

新增的位/字段：

- INT31-INT16

4.2.2.4 DMA_ERR



Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERR31	ERR30	ERR29	ERR28	ERR27	ERR26	ERR25	ERR24	ERR23	ERR22	ERR21	ERR20	ERR19	ERR18	ERR17	ERR16
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR15	ERR14	ERR13	ERR12	ERR11	ERR10	ERR9	ERR8	ERR7	ERR6	ERR5	ERR4	ERR3	ERR2	ERR1	ERR0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

图 77. DMA_ERR—Kinetis 120 MHz

新增的位/字段：

- ERR31-ERR16

4.2.2.5 DMA_HRS

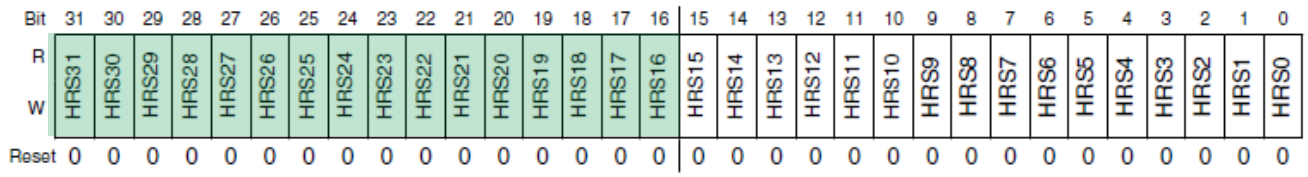


图 78. DMA_HRS—Kinetis 120 MHz

新增的位/字段:

- HRS31-HRS16

4.2.3 软件影响

现有代码将在 eDMA 上运行，无需修改。如果您希望使用新通道，将需要进行软件更新。

4.2.4 硬件影响

无硬件影响。

4.3 多功能时钟发生器（MCG）

4.3.1 特性

Kinetis 120 MHz 产品上的 MCG 进行了一系列修改。最重要的是 PLL 可支持更高的频率。此外, MCG 现在包含两个 PLL，而不是一个。

MCG 的新特性包括:

- 最大 PLL 输出频率增加到 120 MHz 或 150 MHz（最大值取决于部件编号上的速度等级）
- 新增辅助 PLL。PLL0 或 PLL1 均能够提供系统级时钟（MCGOUTCLK）。
- 只有 PLL1 可作为 DDR 模块的时钟源。
- 新增辅助 OSC
  - 任一 OSC 均可作为任一 PLL 的输入
  - 只有 OSC0 或 RTC OSC 可作为 FLL 的输入
  - 向 MCG 添加了额外的 OSC 控制输出，用于辅助 OSC（HGO1、RANGE1 和 EREFS1）控制
- PLL 输入参考范围变为 8-16 MHz
- 只有当处于特权模式下时，才能写入 MCG 寄存器。可以在任何模式下读取寄存器。

4.3.2 受影响的寄存器

MCG 新增了多个寄存器，并将 MCG_ATC 寄存器重命名为 MCG_SC。此外，更新了多个寄存器字段和位名称，以便保证其他模块使用的命名约定相符。

下表列出了存储空间映射变化：

表 11. MCG 存储空间映射比较

寄存器	位置	Kinetis 100 MHz 版本 1.x	Kinetis 120 MHz
MCG 状态和控制寄存器	0x4006_4008	MCG_ATC	MCG_SC
MCG 控制 7 寄存器	0x4006_400C	N/A	MCG_C7
MCG 控制 8 寄存器	0x4006_400D	N/A	MCG_C8
MCG 控制 10 寄存器	0x4006_400F	N/A	MCG_C10
MCG 控制 11 寄存器	0x4006_4010	N/A	MCG_C11
MCG 控制 12 寄存器	0x4006_4011	N/A	MCG_C12
MCG 状态 2 寄存器	0x4006_4012	N/A	MCG_S2

4.3.2.1 MCG_C2 寄存器

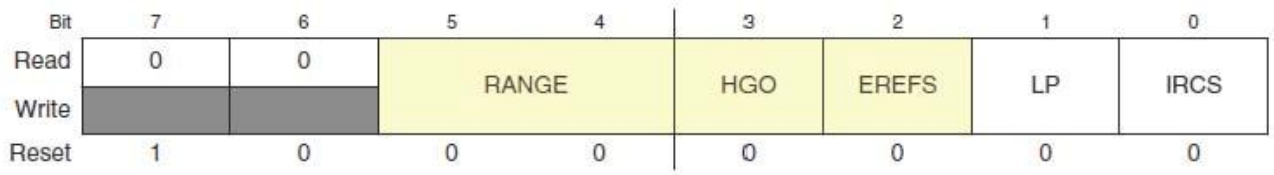


图 79. MCG_C2—Kinetis 100 MHz 版本 1.x

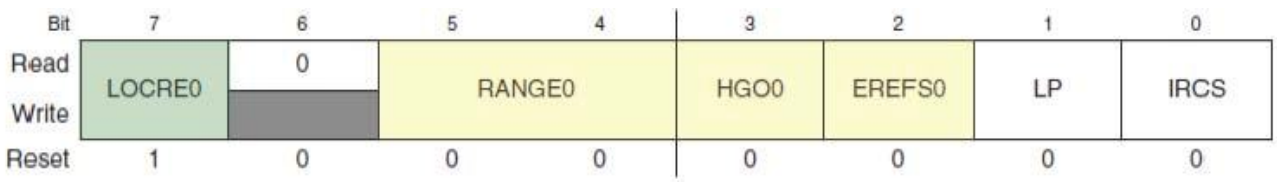


图 80. MCG_C2—Kinetis 100 MHz 版本 2.x120 MHz

新增的位/字段：

- LOCRE0

改变的位/字段名称：

- RANGE → RANGE0
- HGO → HGO0
- EREFS → EREFS0

4.3.2.2 MCG_C5 寄存器

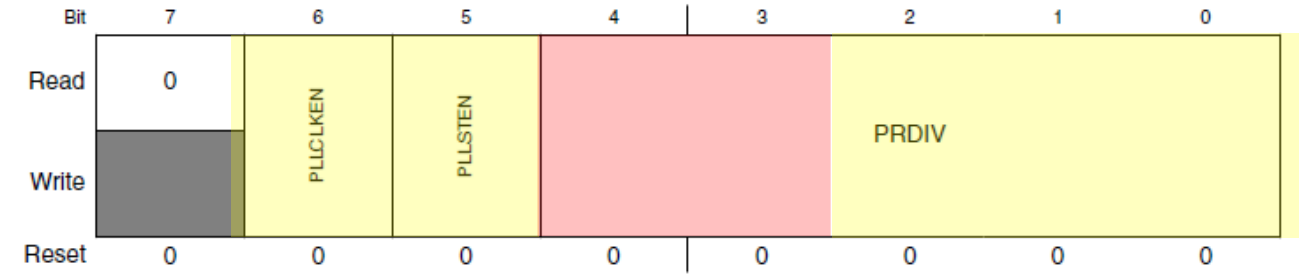


图 81. MCG_C5—Kinetis 100 MHz 版本 1.x

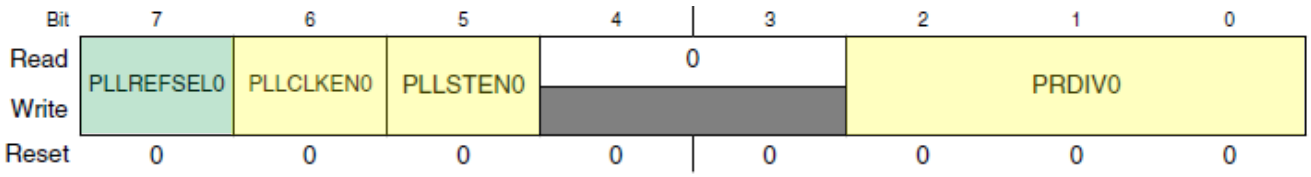


图 82. MCG_C5—Kinetis 120 MHz

新增的位/字段:

- PLLREFSEL0 - 用于在 OSC0 和 OSC1 之间选择作为 PLL0 的输入

改变的位/字段名称:

- PLLCLKEN → PLLCLKEN0
- PLLSTEN → PLLSTEN 0
- PRDIV → PRDIV 0 - 现在的有效范围是 1-8，而不是 1-25

4.3.2.3 MCG_C6

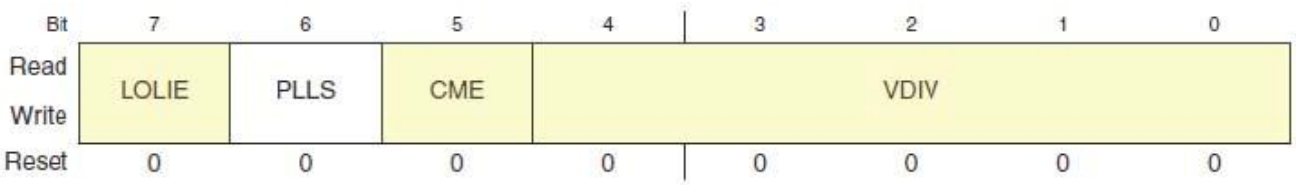


图 83. MCG_C6—Kinetis 100 MHz 版本 1.x

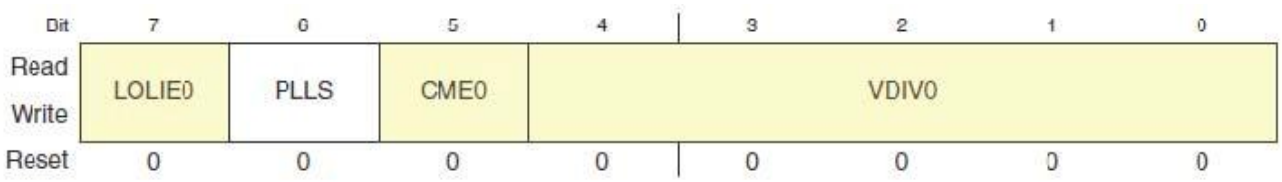


图 84. MCG_C6—Kinetis 100 MHz 版本 2.x120 MHz

改变的位/字段名称:

- LOLIE → LOLIE0
- CME → CME0
- VDIV → VDIV0 - VCO 分频器的有效范围从 24-55 改为 16-47

4.3.2.4 MCG_S 寄存器

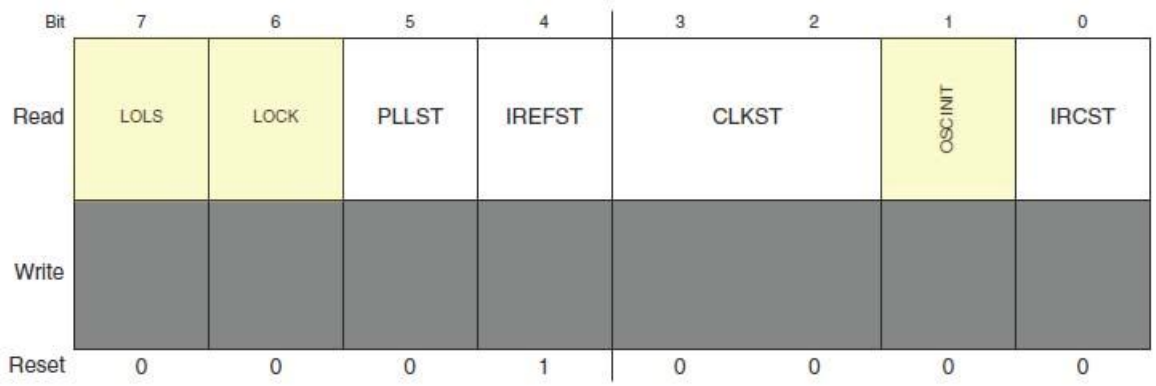


图 85. MCG_S—Kinetis 100 MHz 版本 1.x

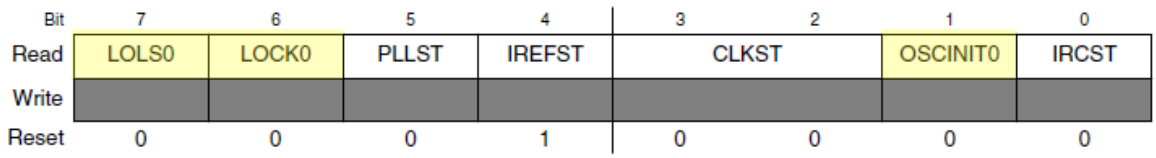


图 86. MCG_S—Kinetis 120 MHz

改变的位/字段名称:

- LOLS → LOLS0
- LOCK → LOCK0
- OSCINIT → OSCINIT0

4.3.2.5 MCG_ATC 寄存器

图 87. MCG_ATC—Kinetis 100 MHz 版本 1.x

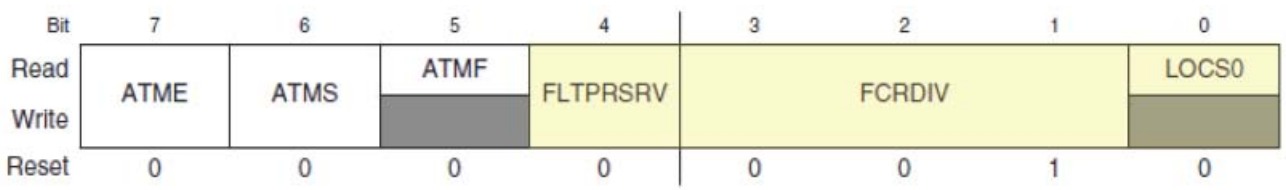


Figure 88. MCG_SC—Kinetis 120 MHz

图 88. MCG_SC—Kinetis 100 MHz 版本 2.x120 MHz

寄存器名称修改为 **MCG_SC**

新增的位/字段:

- FLTPRSRV

- FCRDIV
- LOCS0

4.3.2.6 MCG_C7 寄存器

Address: MCG_C7 is 4006_4000h base + Ch offset = 4006_400Ch

Bit	7	6	5	4	3	2	1	0
Read	0							OSCSEL
Write								
Reset	0	0	0	0	0	0	0	0

图 89. MCG_C7—Kinetis 100 MHz 版本 2.x120 MHz

新寄存器

新增的位/字段:

- OSCSEL

4.3.2.7 MCG_C8 寄存器

MCG_C8 是一个新寄存器，具有用于新 RTC 时钟监视器的控制和状态位。这是新增特性。

Bit	7	6	5	4	3	2	1	0
Read	LOCRE1	0	CME1	0				LOCS1
Write								
Reset	1	0	0	0	0	0	0	0

图 90. MCG_C8—Kinetis 120 MHz

新增的位/字段:

- LOLCRE1
- CME1
- LOCS1

4.3.2.8 MCG_C10 寄存器

MCG_C10 是一个新寄存器，用于 OSC1 控制。其大多数寄存器字段与 MCG_C2 寄存器类似。

Bit	7	6	5	4	3	2	1	0
Read	LOCRE2	0	RANGE1		HGO1	EREFS1	0	
Write								
Reset	1	0	0	0	0	0	0	0

图 91. MCG_C10—Kinetis 120 MHz

新增的位/字段:

- LOCRE2
- RANGE1
- HGO1
- EREFS1

4.3.2.9 MCG_C11 寄存器

MCG_C11 是一个新寄存器，用于 PLL1 控制。其大多数寄存器字段与 MCG_C5 寄存器（用于控制 PLL0）类似。

Bit	7	6	5	4	3	2	1	0
Read	PLLREFSEL1	PLLCLKEN1	PLLSTEN1	PLLCS	0	PRDIV1		
Write								
Reset	0	0	0	0	0	0	0	0

图 92. MCG_C11—Kinetis 120 MHz

新增的位/字段:

- PLLREFSEL1
- PLLCLKEN1
- PLLSTEN1
- PLLCS
- PRDIV1

4.3.2.10 MCG_C12 寄存器

MCG_C12 是一个新寄存器，用于 PLL1 控制。其大多数寄存器字段与 MCG_C6 寄存器（用于控制 PLL0）类似。

Bit	7	6	5	4	3	2	1	0
Read	LOLIE1	0	CME2	VDIV1				
Write								
Reset	0	0	0	0	0	0	0	0

图 93. MCG_C12—Kinetis 120 MHz

新增的位/字段:

- LOLIE1
- CME2
- VDIV1

4.3.2.11 MCG_S2 寄存器

MCG_S2 是一个新寄存器，用于 PLL1 和 OSC1 的状态。其大多数寄存器字段与 MCG_S 寄存器（用于 PLL0 和 OSC0 的状态）类似。

Bit	7	6	5	4	3	2	1	0
Read	LOLS1	LOCK1	0	PLLCST	0		OSCINIT1	LOCS2
Write								
Reset	0	0	0	0	0	0	0	0

图 94. MCG_S2—Kinetis 120 MHz

新增的位/字段:

- LOLS1
- LOCK1
- PLLCST
- OSCINIT1
- LOCS2

4.3.3 软件影响

任何使用 PLL 的系统将需要修改软件进行相关的 MCG 更新，调整为适应新的 PLL 输入范围、分频比范围和增加输出频率。还需要在软件中对寄存器和位字段的名字进行相应修改。需要修改软件，使其与新器件头文件中的新名字相匹配。

如果 RTC 作为 FLL 的参考时钟，或作为系统时钟，则不再是在 SIM_SOPT1 中选择该时钟，而是在 MCG_C7 寄存器中进行。

需添加软件以使用任意新功能。如果系统使用处理器用户模式，则配置任意 MCG 寄存器需要在授权模式下执行。

4.3.4 硬件影响

由于 PLL 输入参考范围现在为 8-16 MHz，使用低于 8 MHz 输入时钟的系统需要修改其时钟频率（如果要继续使用 PLL）。

与 RTC 振荡器选择相关的改变就是增加了 DFLL 参考时钟，从而也可以使用具有高频晶振的 OCS0 为多个片上外设提供独立的时钟（OSC0ERCLK）。

还增加了一个辅助振荡器（OSC1）。OSC1 可用作一个或全部两个 PLL 的参考时钟源，但是使用时有一定限制。在考虑使用 OSC1 作为开发板的主要时钟输入时，应考虑以下几项：

- 当 PLL 用作 MCGCLKOUT 时，OSC0 或 OSC1 均可用作 PLL 的参考时钟源。
- 只有 OSC0 或 RTC OSC 可用作 FLL 参考时钟，因此如果使用 FEE、FBE、或 BLPE 时钟模式，或者当从复位默认 FEI 模式向 PBE 模式转换（或任何由“外部”提供时钟的 MCG 模式）时，这些时钟源中的一个必须为 MCG 可用。您必须从 FBE 模式切换进入到 PEE 模式，并使用 PLL 作为 MCGCLKOUT 源。
- 如果要使用 OSC1，必须在 MCG_C10 中对其进行配置并在 OSC1_CR 中使能。
- 如果 OSC1 是仅有的外部时钟源，则可用的 MCG 时钟模式仅有 FEI、FBI 和 BLPI。

4.4 实时时钟（RTC）

4.4.1 特性

包含以下新特性：

- 在所有封装内均可提供 1 Hz 或 32 kHz 的 RTC_CLKOUT 信号

  用例/改进：原有的 1 Hz 选项可以精确产生 1 Hz 时钟，即使输入晶振频率由于板载变化而具有来自 32.768 kHz 的偏差。32.768 kHz 选项允许输入晶振频率直接通过并输出到 RTC_CLKOUT。这允许其他外部器件（需要该时钟）在权衡 RTC 振荡器时将其用作输入，并且可以降低系统 BOM 成本。

- 实现 RTC_WAKEUP 引脚用来在一些 MAPBGA 封装中提供外部警告事件信号

  用例/改进：这是有源低漏极开路引脚，当 MCU 掉电且主 RTC 中断有效时有效。请注意，该引脚可用来测量 RTC 触发输出，从而监视 MCU VDD 电平。

- 增加了 RTC 秒中断

  用例/改进：该专用中断每秒变为有效一次。请注意，这是边沿触发中断，因而没有可清除的标志。这意味着无需访问 RTC 寄存器就能执行中断，从而降低了 VBAT 功耗。秒中断可从任意低功耗模式中产生唤醒（停止/VLPS/LLS/VLLS）。例如，如果 RTC 秒信号连接到 PDB，则 PDB 触发器可接收 RTC 秒触发输入，从而强制 ADC 在运行模式下进行转换（其中 PDB 为使能）。

除了上述列出的变化，120 MHz 器件还实现了 RTC 安全版本，具有专用于创建安全系统的额外特性。这些特性是：

- 64 位单调计数器，一旦初始化后就不能为零或返回到之前任意值。
- 篡改时间秒寄存器在时间无效时进行记录。

4.4.2 受影响的寄存器

RTC 模块具有多个新寄存器。部分现有寄存器中也增加了新字段。所有现有寄存器和字段保持不变。

下表列出了新增的寄存器。以下章节提供了在现有寄存器中增加的寄存器字段的详细信息。

表 12. RTC 存储空间映射比较

寄存器	位置	Kinetis 100 MHz 版本 1.x	Kinetis 120 MHz
篡改时间秒寄存器	0x4003_D020	N/A	RTC_TTSR
单调使能寄存器	0x4003_D024	N/A	RTC_MER
单调计数器低位寄存器	0x4003_D028	N/A	RTC_MCLR
单调计数器高位寄存器	0x4003_D02C	N/A	RTC_MCHR

4.4.2.1 RTC_SR 寄存器

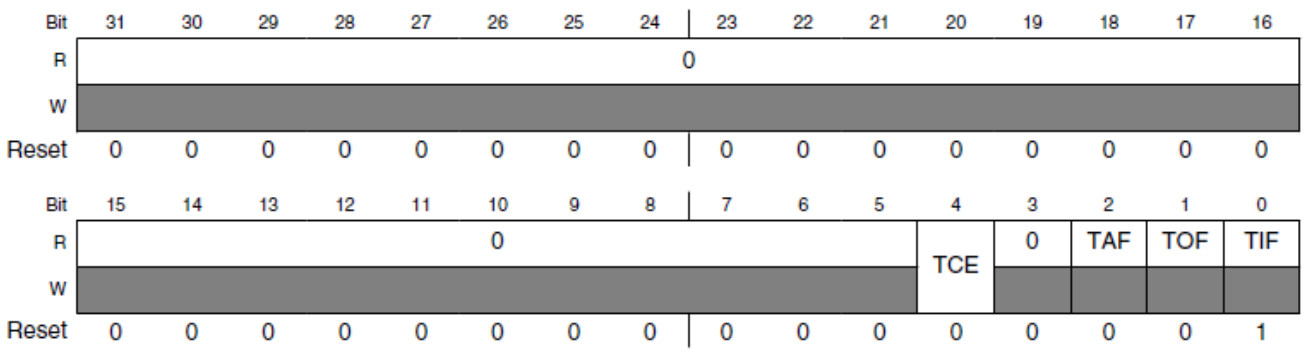


图 95. RTC_SR—Kinetis 100 MHz 版本 1.x

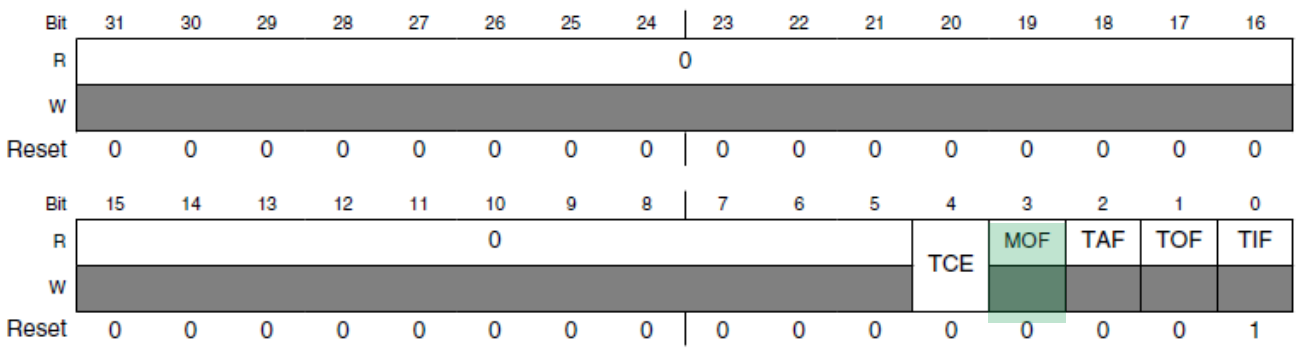


图 96. RTC_SR—Kinetis 120 MHz

新增的位/字段：

- MOF - 单调溢出标志

4.4.2.2 RTC_LR 寄存器

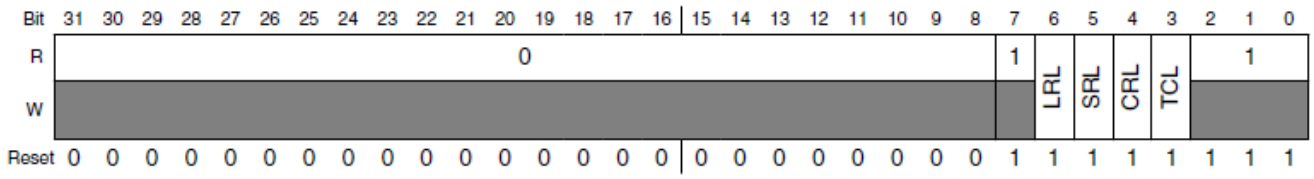


图 97. RTC_LR—Kinetis 100 MHz 版本 1.x

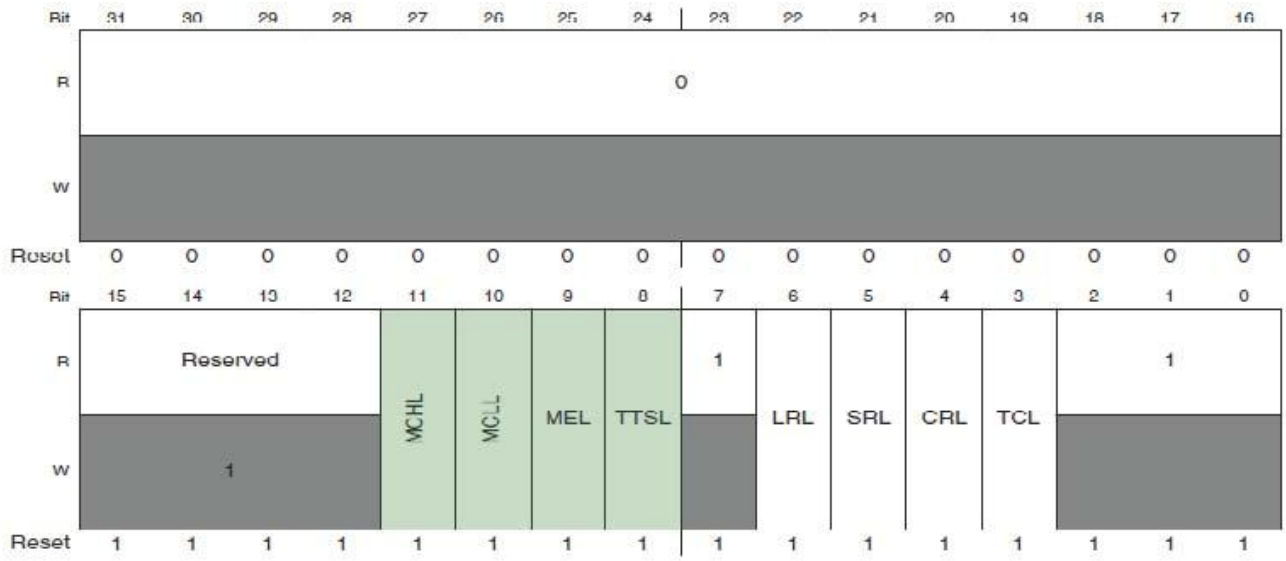


图 98. RTC_LR—Kinetis 120 MHz

新增的位/字段:

- MCHL - 单调计数器高位锁定
- MCLL - 单调计数器低位锁定
- MEL - 单调使能锁定
- TTSL - 篡改时间秒锁定

4.4.2.3 RTC_IER 寄存器

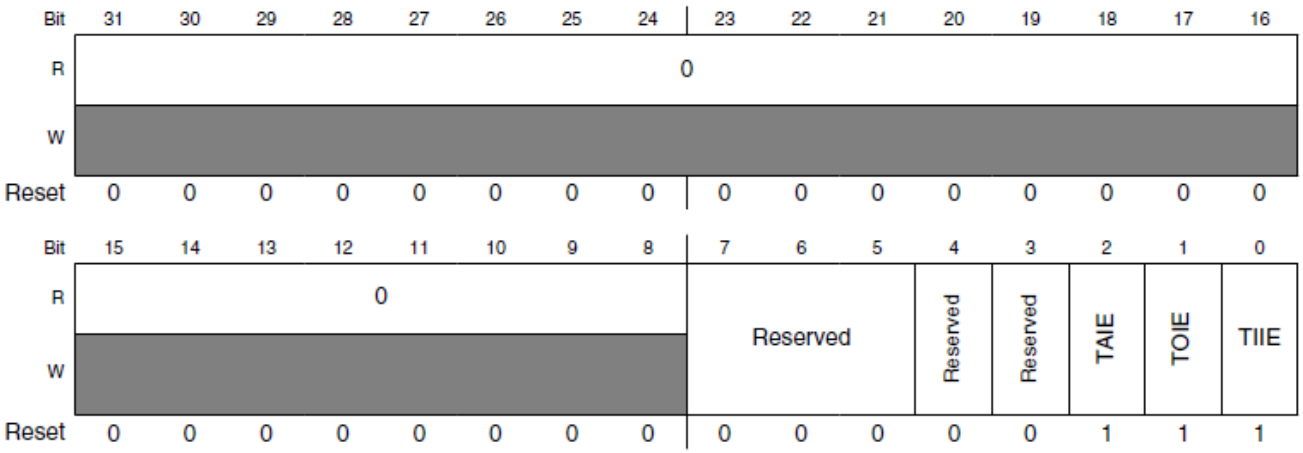


图 99. RTC_IER—Kinetis 100 MHz 版本 1.x

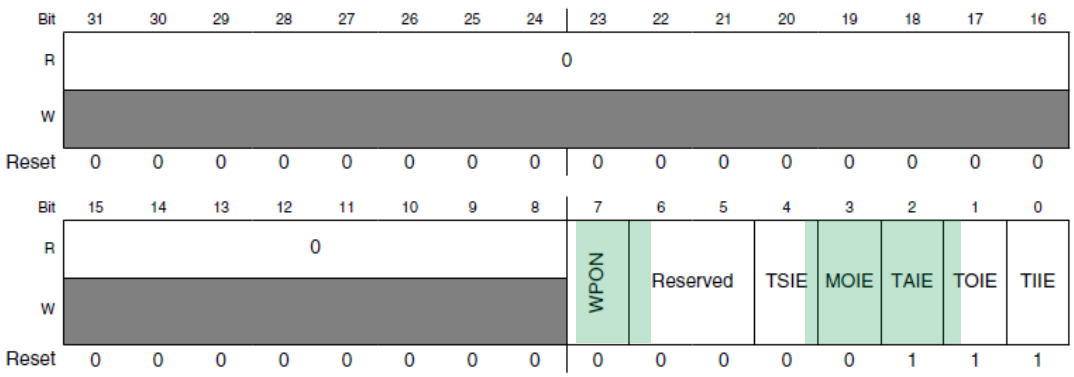


图 100. RTC_IER—Kinetis 120 MHz

新增的位/字段:

- WPON - 使能 RTC_WAKEUP 引脚有效
- TSIE - 时间秒中断使能
- MOIE - 单调计数器溢出中断使能

4.4.2.4 RTC_WAR 寄存器

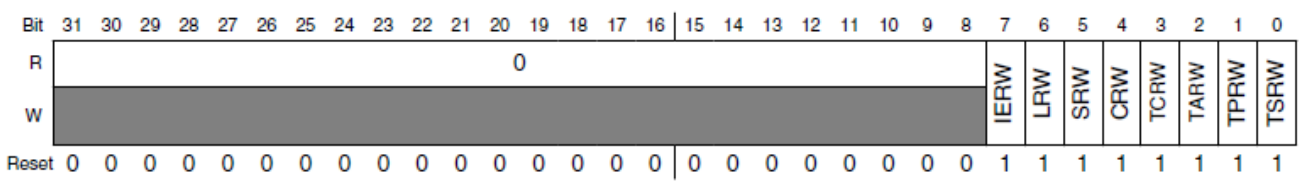


图 101. RTC_WAR—Kinetis 100 MHz 版本 1.x

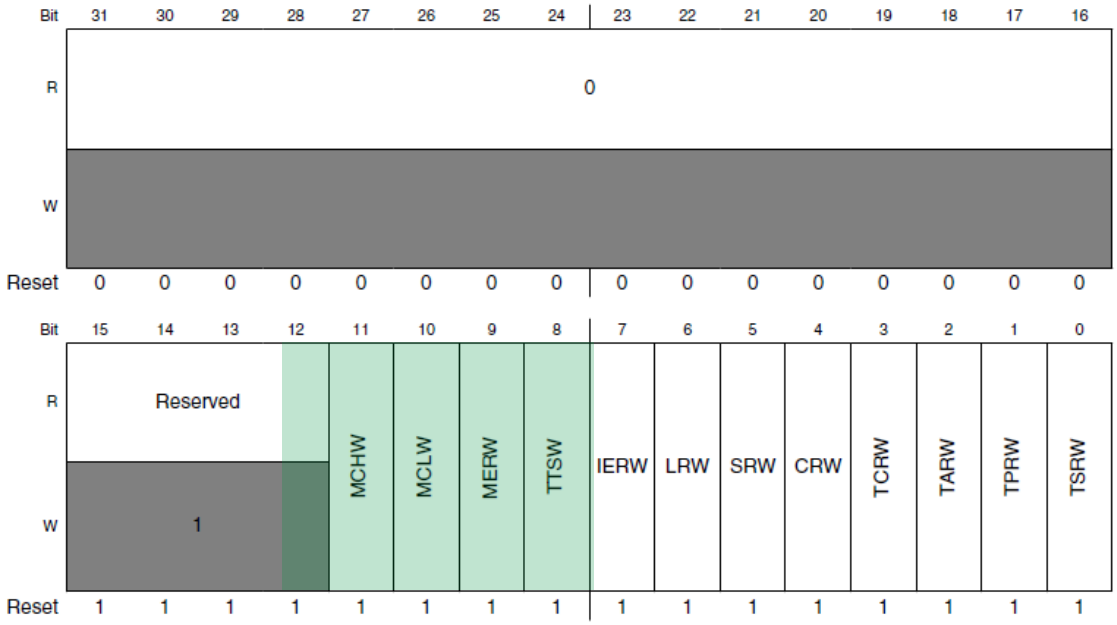


图 102. RTC_WAR—Kinetis 120 MHz

新增的位/字段:

- MCHW - 单调计数器高位写入
- MCLW - 单调计数器低位写入
- MERW - 单调使能寄存器写入
- TTSW - 篡改时间秒写入

4.4.2.5 RTC_RAR 寄存器

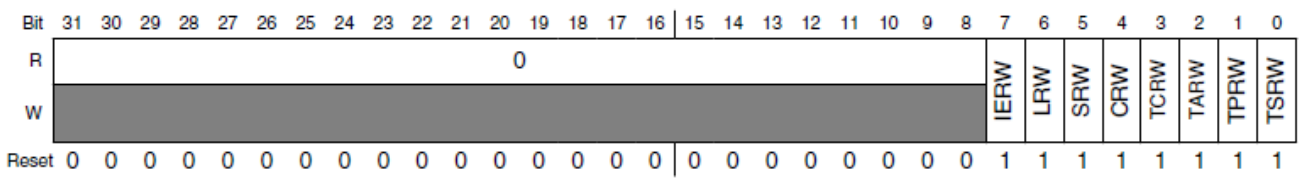


图 103. RTC_RAR—Kinetis 100 MHz 版本 1.x

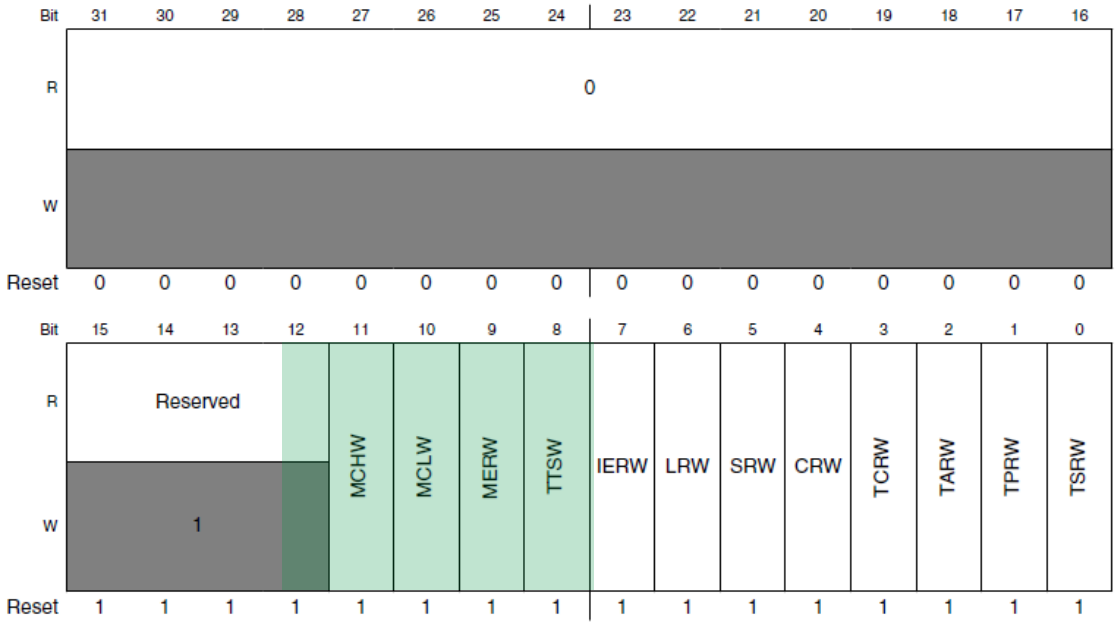


图 104. RTC_RAR—Kinetis 120 MHz

新增的位/字段:

- MCHR - 单调计数器高位读取
- MCLR - 单调计数器低位读取
- MERR - 单调使能寄存器读取
- TTSR - 篡改时间秒读取

4.4.3 软件影响

无需软件修改便能执行现有代码。要利用其它特性，可能需要进行一些软件修改。

4.4.4 硬件影响

无需修改硬件。要利用其它特性，可能需要进行一些硬件修改。

4.5 可编程延迟区块（PDB）

4.5.1 特性

PDB 增加了许多新特性。新特性包括:

- 新增了三个脉冲输出 n 延时定时器（PDB0_PO1DLY、PDB0_PO2DLY 和 PDB0_PO3DLY）。这些寄存器用于设置 CMP1、CMP2 和 CMP3 模块的采样窗口。
- 新增了两个新的 ADC 通道触发配置寄存器集（PDB0_CHnC1、PDB0_CHnS、PDB0_CHnDLY0 和 PDB0_CHnDLY1）。两个新通道可用于触发 ADC2 和 ADC3。
- 当处理器处于调试模式时，PDB 计数器寄存器(PDBx_CNT)值将暂停。

4.5.2 受影响的寄存器

PDB 新增了多个寄存器。这些寄存器均为现有寄存器的额外实例化。寄存器中的字段定义均相同，只是增加了寄存器副本以允许控制更多的通道。

下表显示了新寄存器的位置:

表 13. PDB 存储空间映射比较

寄存器	位置	Kinetis 100 MHz 版本 1.x	Kinetis 120 MHz
通道 n 控制寄存器 1	0x4003_6060	N/A	PDB0_CH2C1
通道 n 状态寄存器	0x4003_6064	N/A	PDB0_CH2S
通道 n 延时 0 寄存器	0x4003_6068	N/A	PDB0_CH2DLY0
通道 n 延时 1 寄存器	0x4003_606C	N/A	PDB0_CH2DLY1
通道 n 控制寄存器 1	0x4003_6088	N/A	PDB0_CH3C1
通道 n 状态寄存器	0x4003_608C	N/A	PDB0_CH3S
通道 n 延时 0 寄存器	0x4003_6090	N/A	PDB0_CH3DLY0
通道 n 延时 1 寄存器	0x4003_6094	N/A	PDB0_CH3DLY1
脉冲输出 n 延时寄存器	0x4003_6198	N/A	PDB0_PO1DLY
脉冲输出 n 延时寄存器	0x4003_619C	N/A	PDB0_PO2DLY
脉冲输出 n 延时寄存器	0x4003_61A0	N/A	PDB0_PO3DLY

4.5.3 软件影响

增加了 PDB0_PO1DLY 和 PDB0_PO2DLY 寄存器，使您可以为 CMP0、CMP1 和 CMP2 输出使用不同的窗口周期。如果 CMP0、CMP1 和 CMP2 配置为窗口模式，则需要修改软件来配置合适的 PDB0_PO*n*DLY 寄存器以便配置窗口，而不是将 DB0_PO1DLY 寄存器用于所有比较器。

如果新 PDB 通道要用于触发 ADC2 和 ADC3，则也需要修改软件。

4.5.4 硬件影响

无硬件影响。

4.6 灵活定时器模块（FTM）

4.6.1 特性

Kinetis 120 MHz 器件增加了具有 8 个通道的第四个 FTM 实例化（FTM3）。

FTM 模块增加了新的硬件同步触发功能，允许通过与 FTM1 或 FTM2 的匹配来触发 FTM0 和/或 FTM3。该新特性使连接到 16 位 FlexTimer 成为可能，从而允许您增大 PWM 周期，而具有比使用定时器溢出标志（TOF）来连接两个模块更短的死区时间。这些硬件触发器配置为使用 SIM_SOPT4 寄存器中的新位。关于更多信息，请参见 SIM 章节。

针对 FTM1 通道 0，也可以选择使用新的 USB 帧起始脉冲作为输入捕捉触发信号。该位可用于将 MCU 时钟与 USB 时钟同步，这对一些 USB 音频应用很有用。通过 SIM_SOPT4[FTM1CH0SRC]位控制与 USB 帧起始的同步。关于更多信息，请参见 SIM 和 USB 章节。

4.6.2 受影响的寄存器

新增一组 FTM 寄存器用于新实例化 (FTM3)。新 FTM 特性全部通过修改 SIM 寄存器进行处理；FTM 寄存器和字段保持不变。

4.6.3 软件影响

无需软件修改便能执行现有代码。要利用其它特性或新的 FTM3 模块，可能需要进行一些软件修改。

4.6.4 硬件影响

无硬件影响。

4.7 低功耗定时器（LPTMR）

4.7.1 特性

更新了 LPTMR，在读取 LPTMR_CNR 值时始终确保数据有效。

4.7.2 受影响的寄存器

LPTMR_CNR

4.7.3 软件影响

每次读取 LPTMR 计数器寄存器时，软件必须先将任意值写入 LPTMR 计数器寄存器。这将 LPTMR 计数器寄存器中的当前值同步并寄存到一个临时寄存器中。在每次读取 LPTMR 计数器寄存器时，将返回临时寄存器中的内容。

4.7.4 硬件影响

无硬件影响。

4.8 通用异步接收器/传输器（UART）

4.8.1 特性

新版本的 UART 在 UART0 上增加了对 CEA709.1-B (LON)协议的支持。该协议通常用于建立自动化和家庭联网应用。除了支持 CEA709.1-B 外，UART 实例化的特性并未改变。

4.8.2 受影响的寄存器

向 UART0 存储空间映射增加了多个新寄存器，用于支持新的 CEA709.1-B (LON)特性。该新特性及其控制均为新增的存储空间映射，因而之前存在的寄存器和位保持不变。

下表显示了原有 UART 和增强型版本（包含 LON 功能）之间的存储空间映射区别。

表 14. 存储空间映射比较

位置	原有的 UART	增强型 UART
0x4006A021	N/A	UART CEA709.1-B 控制寄存器 6 (UART0_C6)
0x4006A022	N/A	UART CEA709.1-B 数据包周期时间计数器高位(UART0_PCTH)
0x4006A023	N/A	UART CEA709.1-B 数据包周期时间计数器低位(UART0_PCTL)
0x4006A024	N/A	UART CEA709.1-B Beta1 定时器 (UART0_B1T)
0x4006A025	N/A	UART CEA709.1-B 辅助延时定时器高位 (UART0_SDTH)
0x4006A026	N/A	UART CEA709.1-B 辅助延时定时器低位 (UART0_SDTL)
0x4006A027	N/A	UART CEA709.1-B 报头(UART0_PRE)
0x4006A028	N/A	UART CEA709.1-B 发送数据包长度 (UART0_TPL)
0x4006A029	N/A	UART CEA709.1-B 中断使能寄存器 (UART0_IE)
0x4006A02A	N/A	UART CEA709.1-B WBASE (UART0_WB)
0x4006A02B	N/A	UART CEA709.1-B 状态寄存器 (UART0_S3)
0x4006A02C	N/A	UART CEA709.1-B 状态寄存器 (UART0_S4)
0x4006A02D	N/A	UART CEA709.1-B 接收数据包长度 (UART0_RPL)
0x4006A02E	N/A	UART CEA709.1-B 接收报头长度 (UART0_RPREL)
0x4006A02F	N/A	UART CEA709.1-B 冲突脉冲宽度 (UART0_CPW)

下一页继续介绍此表...

表 14. 存储空间映射比较 (继续)

位置	原有的 UART	增强型 UART
0x4006A030	N/A	UART CEA709.1-B 接收不确定时间 (UART0_RIDT)
0x4006A031	N/A	UART CEA709.1-B 发送不确定时间 (UART0_TIDT)

4.8.3 软件影响

如果您的系统无需使用新的 CEA709.1-B 特性，则无需软件修改。

4.8.4 硬件影响

如果您的系统无需使用新的 CEA709.1-B 特性，则无需硬件修改。

4.9 通用串行总线（USB）

4.9.1 特性

包含以下新特性:

- USB 稳压器待机模式保护机制
- USB_SOF_OUT 引脚上的帧起始（SOF）输出
- USB 外设在主机模式下新增可调节帧寄存器

4.9.1.1 USB 稳压器待机模式

USB 稳压器输出 Vout33 的功能之一就是作为 MCU 的主电源。

使能待机模式时，稳压器限制最大电流负载为 1 mA。Kinetis 100 MHz 版本 1.x 器件允许在 MCU 处于运行模式时配置 USB VREG 进入待机模式，这意味着 MCU 没有足够的电流用于执行指令。

新的 Kinetis 器件具有保护机制，可控制何时将 USB 稳压器置于待机模式。

4.9.1.2 帧起始输出引脚

增加了新信号可以反映 SOF 引脚的值，可作为类似启动事件用于同步目的（音频和数据记录器等）。

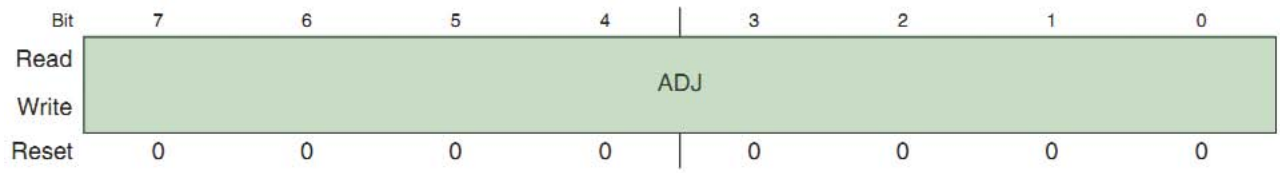
4.9.1.3 主机模式下的可调节帧

SOF 通常产生 12,000 个 12 MHz 时钟周期。该新特性可在–128 至+127 范围内进行调节，以补偿 USB 48 MHz 时钟的不精确性。

4.9.2 受影响的寄存器

可通过新的 SIM_SOPT1CFG 寄存器和 SIM_SOPT1 寄存器 (更新了位和位字段以支持新特性) 控制 USB 稳压器的功能。更多详情，请参见 SIM 章节。

USB 具有一个新寄存器：USBx_USBFRMADJUST。



USBx_USBFRMADJUST field descriptions

图 105. USBx USBFRMADJUST

4.9.3 软件影响

4.9.3.1 USB 稳压器待机模式

在 SIM_SOPT1 寄存器中增加了两个新位用于控制 USB VREG 待机模式。

USBSSTBY—在停止、VLPS、LLS 和 VLLS 模式下，USB 稳压器处于待机模式。

USBVSTBY—在 VLPR 和 VLPW 模式下，USB 稳压器处于待机模式。

在 SIM_SOPT1(USBSSTB、USBVSTB 和 USBREGEN 位)中使能 USB 稳压器设定之前，需要在新的 SIM_SOPT1CFG 寄存器中允许写入。

4.9.3.2 主机模式下的可调节帧

向新的 USBx_USBFRMADJUST 寄存器写入 2 的补码以调节 USB 帧的周期。

4.9.4 硬件影响

无需修改硬件即可兼容之前版本。

4.9.4.1 帧起始输出引脚

本文档的引脚复用章节提供关于该引脚的更多信息。

4.10 比较器

4.10.1 新增和修改的特性

修改 **1**

增加了一个 12 位 DAC1 输出，作为 CMP0 输入通道 4（IN4）输入的一部分。

下表显示了在 Kinetis 100 MHz 版本 2.x120 MHz72 MHz50 MHz 芯片中，CMP0 输入通道 4 同时与 CMP0_IN4 外部引脚和 12 位 DAC1 输出连接。当选择 CMP0 通道 4 作为输入时，只能使能两个信号中的一个，以防止信号冲突。

CMP0 输入	Kinetis 100 MHz 版本 1.x	Kinetis 100 MHz 版本 2.x120 MHz72 MHz50 MHz
IN4	CMP0_IN4	12b DAC1 参考/CMP0_IN4

修改 **2**

CMP0_IN4 输入重定位到另一个 MCU 引脚。

本文档的引脚复用章节显示了 CMP0_IN4 输入的新位置移动到 K40/K60 144 MAPBGA Kinetis 器件的引脚 L4，或是 K40/K60 LQFP Kinetis 器件的引脚 39。对于不同的 Kinetis 系列器件和封装类型，该新位置可能不同。关于 CMP0_IN4 输入的新引脚确切位置，请参见相应的 Kinetis 参考手册，并查看信号复用和引脚分配表。

修改 **3**

除了上述的功能修改外，Kinetis 120 MHz 器件还增加了一个额外的比较器实例化。120 MHz 器件最多具有 4 个比较器，而 100 MHz 器件最多为 3 个比较器。

4.10.2 受影响的寄存器

从 CMP 寄存器中删除了多个寄存器字段。

4.10.2.1 CMPx_SCR

Bit	7	6	5	4	3	2	1	0
Read	0	DMAEN	SMELB	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

图 106. CMPx_SCR—Kinetis 100 MHz 版本 1.x

Bit	7	6	5	4	3	2	1	0
Read	0	DMAEN	0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

图 107. CMPx_SCR—Kinetis 120 MHz100 MHz 版本 2.x72 MHz50 MHz

删除的位/字段名称:

- SMELB

4.10.2.2 CMPx_MUXCR

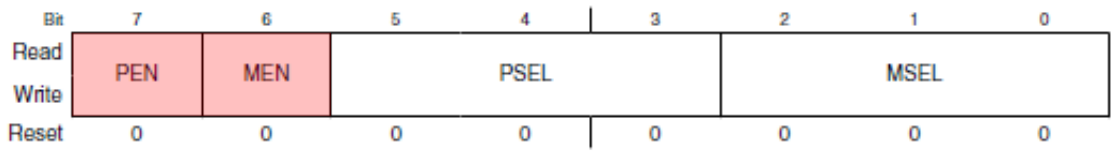


图 108. CMPx_MUXCR—Kinetis 100 MHz 版本 1.x

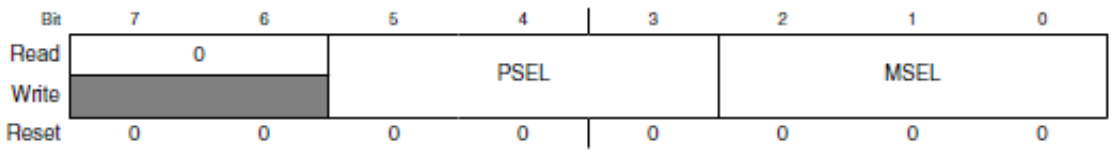


图 109. CMPx_MUXCR—Kinetis 120 MHz100 MHz 版本 2.x72 MHz50 MHz

删除的位/字段名称:

- PEN
- MEN

4.10.3 软件影响

修改 **1**

CMP0 IN4 输入通道与 12 位 DAC1 参考和 CMP0_IN4 引脚复用。当选择 CMP0 IN4 时，应该只使能两个信号中的一个。例如，如果 CMP0 IN4 用于 CMP0_IN4 引脚，要确认通过设置 DAC1_C0[DACEN] = 0 来禁止 12 位 DAC1 输出。如果 CMP0 IN4 用于 12 位 DAC1 输出，则应断开驱动 CMP0_IN4 引脚的外部源。

修改 **2**

在新的 Kinetis 器件中，PORTC_PCR_10[MUX] = 0 不再选择 CMP0_IN4 作为输入。新的 CMP0_IN4 输入重定位到一个模拟专用引脚，该引脚上的所有复用功能均为模拟功能，不作为数字 GPIO。这样的仅模拟引脚无需通过引脚控制寄存器指定引脚功能。然而，建议在同一个引脚上，应只使能一种复用模拟功能。

4.10.4 硬件影响

如果使用 CMP0_IN4 信号，则需要修改硬件以将 CMP0_IN4 输入重新布线到新引脚上。当使用 CMP0_IN4 引脚时，不应使用同一引脚上的其他复用模拟功能，并将其禁用。

4.11 周期中断定时器（PIT）

4.11.1 特性

包含以下新特性:

- 通过链模式位 PIT_TCTRLn[CHN]使外设与定时器通道级联的能力。

  用例/改进：提供了将一个定时器链接到前一个定时器的方法。例如，如果对于通道 2 该位置位，则定时器 2 链接到定时器 1，从而创建一个 64 位计数器。

4.11.2 受影响的寄存器

虽然整个 PIT 存储空间映射没有变化，但是向定时器控制寄存器中添加了一个新的寄存器位字段。

受影响的寄存器汇总如下。

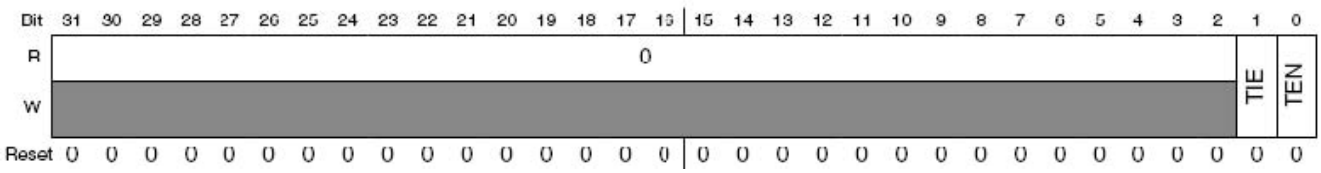


图 110. PIT_TCTRLn 寄存器 - 原有

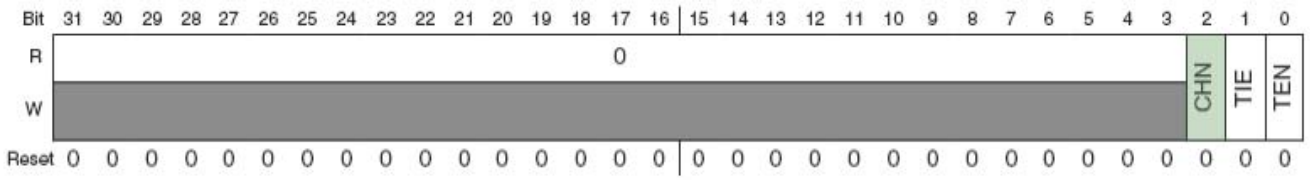


图 111. PIT_TCTRLn 寄存器 - 增强型

新增的位/字段： CHN

4.11.3 软件影响

对于新版本 PIT，无需软件修改便能执行现有代码。要利用其它特性，可能需要进行一些软件修改。

4.11.4 硬件影响

无硬件影响。

4.12 以太网 MAC（ENET）

4.12.1 特性

包含以下新特性：

1. 兼容 AMD 魔术包（Magic Packet）检测

   用例/改进：如果帧由一个同步数据流（6 个连续的 0xFF 字节）构成，并且之后跟随一串要唤醒节点的 6 个连续单播 MAC 地址，则使用先前版本的 ENET 解码魔术包。这 42 个字节之间不能存在任何内容，并可以放在以太网帧的任何位置。然后，可以将其放置在以太网、UDP、IP、TCP 或任何其他协议上。

   新版本兼容 AMD 魔术包检测，这需要一个由同步数据流（6 个连续的 0xFF 字节）形成的帧，之后跟随一串要唤醒节点的 16 个连续单播 MAC 地址。这 102 个字节可以放在以太网帧的任何位置。

2. 增强型接收缓冲区描述符新增 VLAN 优先级位

用例/改进：当使用增强型缓冲区描述符时，RxBD[VPCP]（VLAN 优先级代码点）在 3 个位中检测 VLAN 的帧优先级（只有当 RxBD[L]位置位时有效）。该值可用于为流量（例如，声音、视频或数据）提供不同优先级，VLAN 帧的最高优先级为 7。

3. 增加了缓冲区描述符字节顺序选项

用例/改进：当写入缓冲区描述符字段时，无论是增强型还是传统模式，早期和新版本 MAC-NET 均需要通过软件将字节顺序从大端字节顺序（寄存器字节顺序）转换为小端字节顺序。当读取缓冲区描述符字段时，也需要将小端字节顺序转换为大端字节顺序。

然而，ENET_ECR[DBSWP]允许在 ARM Cortex-M4 本地小端字节顺序中管理缓冲区描述符，在硬件中执行字节顺序转换。

4.12.2 受影响的寄存器

ENET 增加了两个新寄存器字段，ENET_ECR[DBSWP]和增强型 RxBD[VPCP]，并修改了 ENET_ECR[MAGICEN]的功能。

没有存储空间映射修改，下表列出了提到的寄存器以供参考。

存储空间映射比较				
	原有的 ENET		增强型 ENET	
	位置	名称	位置	名称
以太网控制寄存器	400C_0024	ENET_ECR	400C_0024	ENET_ECR
增强型接收缓冲区描述符	N/A	增强型 RxBD	N/A	增强型 RxBD

4.12.2.1 ENET_ECR 寄存器

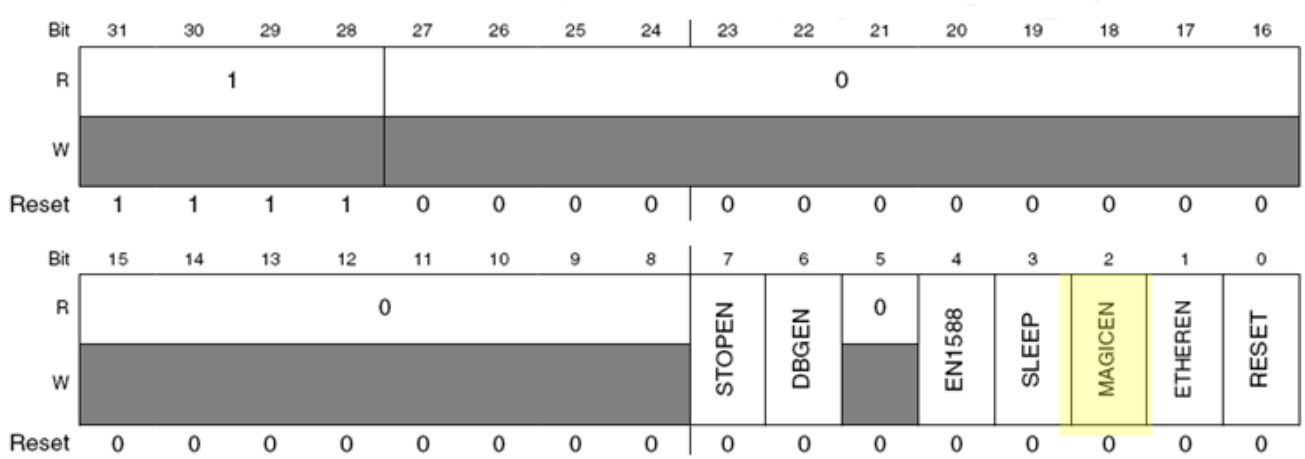


图 112. ENET_ECR - 原有

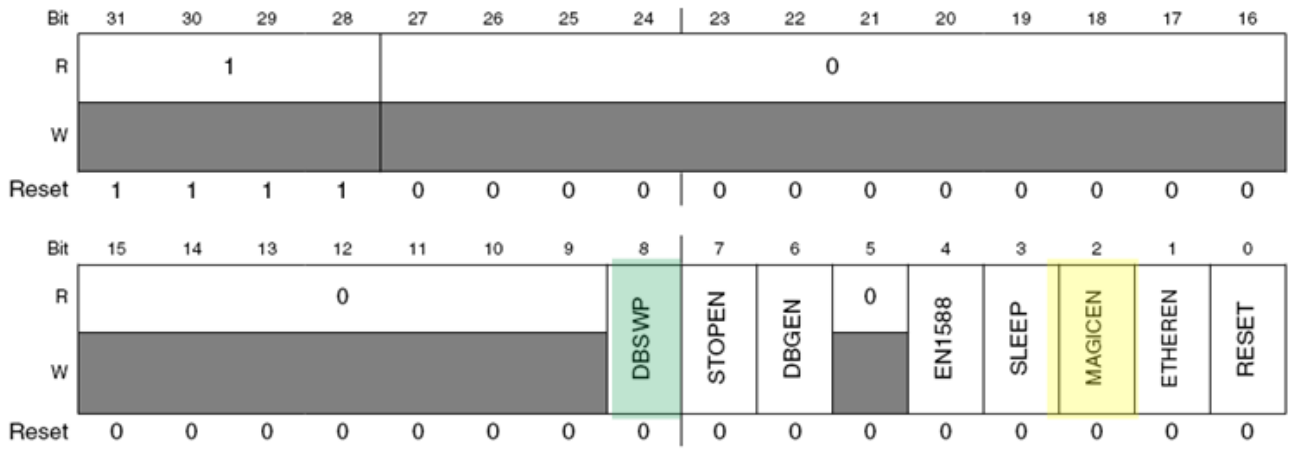


图 113. ENET_ECR - 增强型

新增的位/字段： DBSWP

新增的位/字段： MAGICEN

4.12.2.2 增强型接收缓冲区描述符（RxBD）

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset + 0	E	RO1	W	RO2	L	—	—	M	BC	MC	LG	NO	—	CR	OV	TR
Offset + 2	Data length															
Offset + 4	Rx data buffer pointer - A[31:16]															
Offset + 6	Rx data buffer pointer - A[15:0]															
Offset + 8	ME	—	—	—	—	PE	CE	UC	INT	—	—	—	—	—	—	—
Offset + A	—	—	—	—	—	—	—	—	—	—	ICE	PCR	—	VLAN	IPV6	FRAG
Offset + C	Header length					—	—	—	Protocol type							
Offset + E	Payload checksum															
Offset + 10	BDU	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 12	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 14	1588 timestamp [31:16]															
Offset + 16	1588 timestamp [15:0]															
Offset + 18	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1A	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

图 114. RxBD - 原有

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Offset + 0	E	RO1	W	RO2	L	—	—	M	BC	MC	LG	NO	—	CR	OV	TR
Offset + 2	Data length															
Offset + 4	Rx data buffer pointer - A[31:16]															
Offset + 6	Rx data buffer pointer - A[15:0]															
Offset + 8	ME	—	—	—	—	PE	CE	UC	INT	—	—	—	—	—	—	—
Offset + A	VPCP			—	—	—	—	—	—	—	ICE	PCR	—	VLAN	IPV6	FRAG
Offset + C	Header length					—	—	—	Protocol type							
Offset + E	Payload checksum															
Offset + 10	BDU	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 12	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 14	1588 timestamp [31:16]															
Offset + 16	1588 timestamp [15:0]															
Offset + 18	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1A	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1C	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Offset + 1E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

图 115. RxBD - 增强型

新增的位/字段： VPCP

4.12.3 软件影响

ENET 的默认配置/操作没有改变。所有新特性均兼容 ENET，并对应用进行标准或分载处理，将其分配给硬件。三个特性中的两个为向后兼容，可以使用之前版本，无需在软件中升级。在增强型 ENET 中，新增的缓冲区描述符(BD)字节顺序默认为关闭，并且相应位在原来的 ENET 中为保留。只有当显式使能时才需要修改。

- 兼容 AMD 魔术包（Magic Packet）检测

  新版本兼容 AMD 魔术包（Magic Packet）检测。这需要一个由同步数据流（6 个连续的 0xFF 字节）形成的帧，之后跟随一串要唤醒节点的 16 个连续单播 MAC 地址。这 102 个字节可以放在以太网帧的任何位置。六个连续单播 MAC 地址不再用于唤醒 MCU。

- 向增强型接收缓冲区描述符添加了 VLAN 优先级位

  应用不再需要解析 VLAN 数据包来获得帧优先级。可从增强型 RxBD[VPCP]字段获取。请注意，必须使能增强型缓冲区描述符模式。

- 增加了缓冲区描述符字节顺序选项
  在实现缓冲区描述符字节顺序特性之前，可以利用以下 C 语言结构中的传统缓冲区标识符。

```
typedef struct
{
   uint16_t status;        /* control and status */
```

```
    uint16_t length;        /* transfer length */
    uint8_t *data;          /* buffer address */
} NBUF;
```

所有的结构元素必须以大端字节顺序写入；当访问的结构元素用来说明缓冲区描述符时，需要在软件中进行 16 位（状态和长度）和 32 位（数据指针）字节顺序转换。这也适用于读访问。

对于增强型 ENET，如果使能了缓冲描述符字节顺序特性，则以下 C 语言结构用于说明缓冲区描述符。

```
typedef struct
{
    uint16_t length;        /* transfer length */
    uint16_t status;        /* control and status */
    uint8_t *data;          /* buffer address */
} NBUF;
```

然后，结构的所有元素均能以本地小端字节顺序访问。

请注意长度和状态元素之间的变化，因为硬件字节顺序以 32 位访问实现，而不是 16 位访问。

为了简单起见，本示例中使用了传统缓冲区描述符，而不是增强型 BD，但其均适用于这两种模式。

4.12.4 硬件影响

无需修改硬件即可兼容之前版本。

4.13 参考电压（VREF）

4.13.1 特性

VREF 模块的特性没有改变。

4.13.2 受影响的寄存器

VREF 模块的存储空间映射和字段保持不变。唯一改变的是 VREF_SC[MODE_LV]寄存器的编码。

下表显示了对 MODE_LV 字段编码的修改。

表 17. VREF_SC[MODE_LV]编码

MODE_LV	Kinetis 100 MHz 版本 1.x	Kinetis 120 MHz
00	仅带隙开启	仅带隙开启
01	保留	使能紧凑型稳压缓冲区
10	使能紧凑型稳压缓冲区	保留
11	保留	保留

4.13.3 软件影响

使用 VREF 模块的所有应用将需要修改以识别 VREF_SC[MODE_LV]字段的编码。

4.13.4 硬件影响

无硬件影响。

4.14 其他控制模块（MCM）

4.14.1 特性

MCM 模块特性不变，但是修改了一个寄存器名称。

4.14.2 受影响的寄存器

修改了 MCM_SRAMAP 寄存器。寄存器中的字段不变。 下表显示了寄存器命名的变化。

表 18. MCM 存储空间映射比较

寄存器	位置	Kinetis 100 MHz 版本 1.x	Kinetis 100 MHz 版本 1.x120 MHz72 MHz50 MHz
MCM 控制寄存器	0xE008_000C	MCM_SRAMAP	MCM_CR

4.14.2.1 MCM_CR

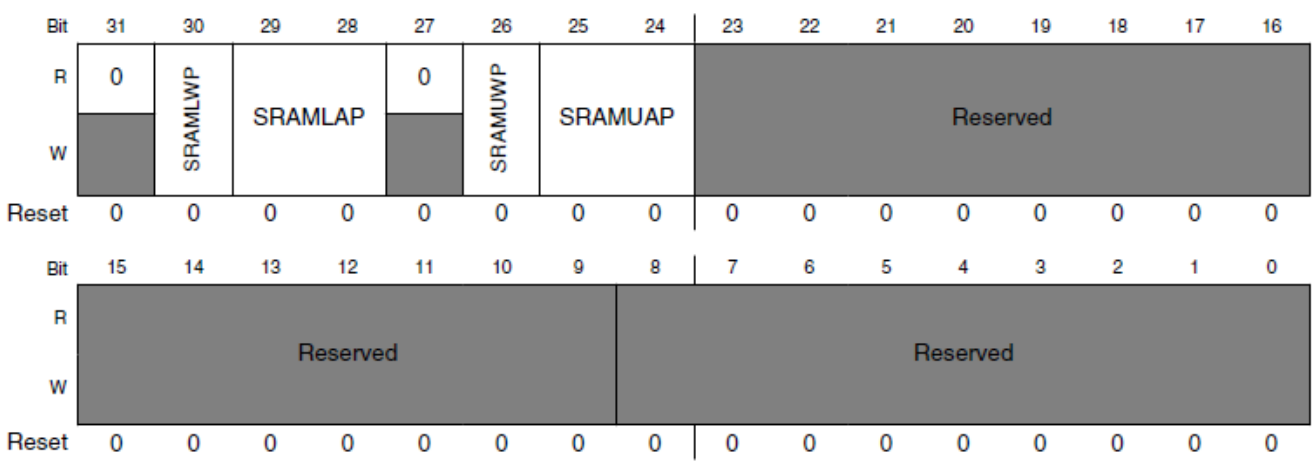


图 116. MCM_SRAMP—Kinetis 100 MHz 版本 1.x

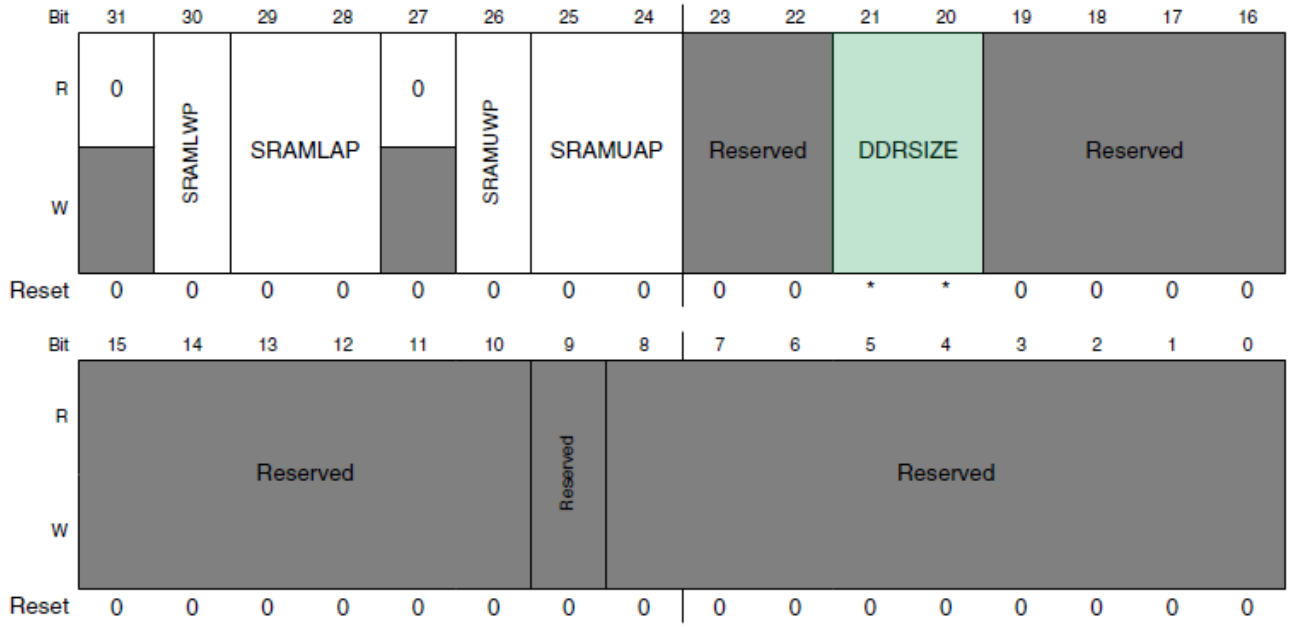


图 117. MCM_CR—Kinetis 120 MHz

增加的位/字段:

- DDRSIZE

4.14.3 软件影响

无需软件修改便能执行现有代码。如果使用更新的头文件，则任何对 MCM_SRAMAP 的引用将需要修改为 MCM_CR。如果使用 DDR，则应适当地编程 MCM_CR[DDR_SIZE]字段。

4.14.4 硬件影响

无硬件影响。

4.15 外部看门狗监控器（EWM）

4.15.1 特性

EWM 模块增加了当 $\overline{\text{EWM\_OUT}}$ 信号电平有效时产生中断请求的能力。

4.15.2 受影响的寄存器

EWM 增加了一个位用于使能新中断特性。

4.15.2.1 EWM_CTRL

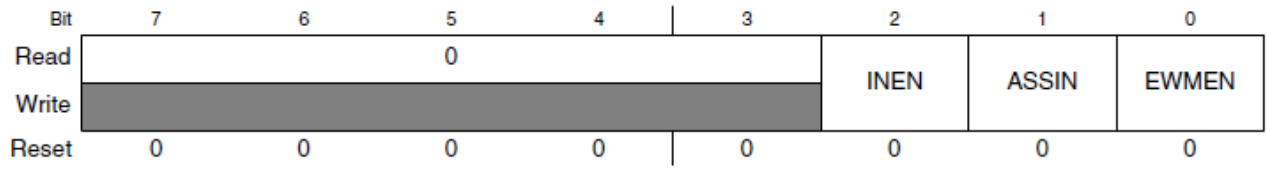


图 118. EWM_CTRL—Kinetis 100 MHz 版本 1.x

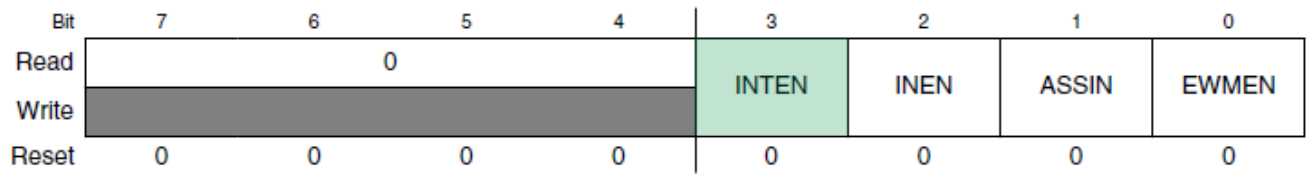


图 119. EWM_CTRL—Kinetis 100 MHz 版本 2.x120 MHz72 MHz50 MHz

增加的位/字段:

- INTEN

4.15.3 软件影响

无需软件修改便能执行现有代码。若要使用新特性，则需要修改软件。

4.15.4 硬件影响

无硬件影响。

4.16 看门狗定时器（WDOG）

4.16.1 特性

WDOG 模块的特性不变，但对寄存器做了修改。

4.16.2 受影响的寄存器

在 WDOG 模块中，删除了一个寄存器位。在之前版本中，WDOG_STCTRLH 中的 STNDBYEN 和 STOPEN 位均用于控制 STOP 模式下的看门狗操作。为了避免混淆，删除了冗余的 STNDBYEN 位。现在 STNDBYEN 位是始终为 1 的保留位。

4.16.2.1 WDOG_STCTRLH

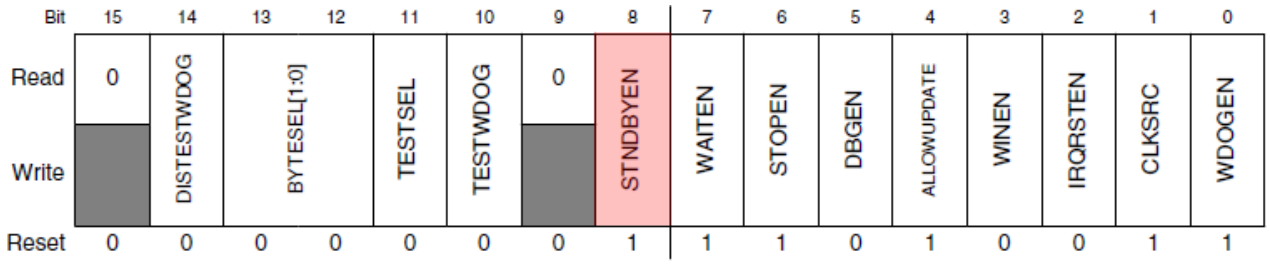


图 120. WDOG_STCTRLH—Kinetis 100 MHz 版本 1.x

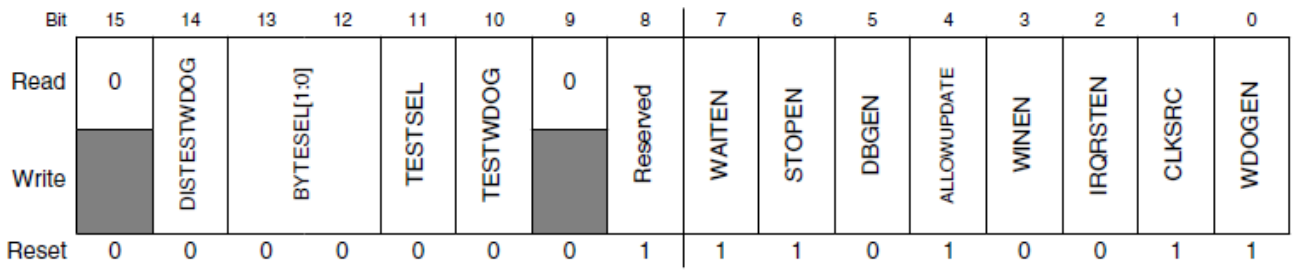


图 121. WDOG_STCTRLH—Kinetis 100 MHz 版本 2.x120 MHz72 MHz50 MHz

删除的位/字段:

- STNDBYEN

4.16.3 软件影响

从寄存器和 Freescale 提供的头文件中删除了 WDOG_STCTRLH[STNDBYEN]位。如果在软件中存在任何对该位的引用，应将其删除。

4.16.4 硬件影响

无硬件影响。

5 具有新实例化的模块

5.1 直接存储器访问多路复用器（DMAMUX）

由于增加了 DMA 通道数（从 16 增加到 32），添加了一个辅助 DMA 多路复用器。辅助 DMA 多路复用器从 DMA 通道 16-31 中选择 DMA 请求输入。

DMAMUX0 上的所有 DMA 源保持不变，但是为了尽可能实现灵活性，在 DMAMUX1 上复用了一些 DMA 源。

5.2 振荡器（OSC）

增加了一个辅助振荡器（OSC1）。新 OSC 引脚功能与 PTE24 和 PTE25 复用，因此如果使用了辅助 OSC，将会影响使用 PTE24 和 PTE25 功能的硬件。

OSC1 可用作 MCG 输入，因此增加辅助 OSC 会影响 MCG 模块。关于更多信息，请参见 MCG 章节。

5.3 模数转换器（ADC）

ADC 和 PGA 的数量从 2 个增加到了 4 个。

ADC 数量的改变会影响 PDB 模块，因为需要更多的通道用于新的 ADC 实例化。关于更多信息，请参见 PDB 章节。

SIM 模块还增加了用于配置和控制新 ADC 模块的位。关于更多信息，请参见 SIM 章节。

5.4 通用输入/输出（GPIO）

Kinetis 120 MHz 器件可提供比 100 MHz 器件更大的封装（256BGA 封装选项）。由于 120 MHz 器件增加了 GPIO 端口（PORTF），从而增加了引脚数。

6 未改变的模块

下表列出了没有任何重大功能变化的模块

模块	备注
NMI	
XBS	增加了从端口和主端口数量，但是功能保持不变。
MPU	
EzPort	
MMCAU	
FlexBus	
CRC	
DAC	
CMT	
USBDCD	
FlexCAN	
DSPI	
I2C	
SDHC	

7 附录

7.1 引脚复用

下表显示了 Kinetis 100 MHz 版本 1.x 器件与 Kinetis 120 MHz 器件之间在相同封装下的引脚复用区别。一些 Kinetis 120 MHz 器件还提供 256 MAPBGA 封装。本章节的重点在于支持用户更轻松地在两个版本之间重复利用硬件，因此并未提及新增的封装和引脚，新开发板的设计需要使用更大的封装。

本表格使用以下约定：

- (+)—从 Kinetis 100 MHz 版本 1.x 到 Kinetis 120 MHz 增加的内容
- (-)—从 Kinetis 100 MHz 版本 1.x 到 Kinetis 120 MHz 删除的内容
- 阴影区域—从 Kinetis 100 MHz 版本 1.x 到 Kinetis 120 MHz 修改的内容

K60 144 MAPBGA	K60 144 LQFP	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
L5	–	(+) RTC_WAKEUP_b							
D3	1								(+) RTC_CLKOUT
D2	2								(+) SPI1_SIN
E4	4								(+) SPI1_SOUT
E2	8							(+) FTM3_CH0	
E1	9							(-) I2S0_CLKIN (+) FTM3_CH1	(+) USB_SOF_OUT
F4	10					I2S0_RXD→I2S0_RXD0		(+) FTM3_CH2	
F3	11	(+) ADC2_SE16		(+) I2S0_RXD1				(+) FTM3_CH3	
F2	12	(+) ADC2_SE17		(+) I2S0_TXD1				(+) FTM3_CH4	
F1	13					I2S0_TXD→I2S0_TXD0		(+) FTM3_CH5	

下一页继续介绍此表...

G4	14	(+) ADC3_SE 16						(+) FTM3_CH 6	
G3	15	(+) ADC3_SE 17						(+) FTM3_CH 7	
J1	23	ADC0_DP 1 → PGA2_DP/ ADC2_DP 0/ ADC3_DP 3/ ADC0_DP 1							
J2	24	ADC0_DM 1 → PGA2_DM/ ADC2_DM 0/ ADC3_DM 3/ ADC0_DM 1							
K1	25	ADC1_DP 1 → PGA3_DP/ ADC3_DP 0/ ADC2_DP 3/ ADC1_DP 1							
K2	26	ADC1_DM 1 → PGA3_DM/ ADC3_DM 0/ ADC2_DM 3/ ADC1_DM 1							
L4	39	DAC1_OU T/ CMP2_IN3 / ADC1_SE 23 → DAC1_OU T/ CMP0_IN4 / CMP2_IN3 / ADC1_SE 23							

下一页继续介绍此表...

M4	45	ADC0_SE17 → ADC0_SE17/ EXTAL1				(+) I2S1_TX_FS			(+) I2S1_RXD1
K5	46	ADC0_SE18 → ADC0_SE18/XTAL1				(+) I2S1_TX_BCLK			(+) I2S1_TXD1
K4	47	(+) ADC3_SE5b		(+) ENET_1588_CLKIN		(-) ENET_1588_CLKIN (+) I2S1_TXD0			
J4	48	(+) ADC3_SE4b				(+) I2S1_MCLK			
H4	49	(+) ADC3_SE7a							
J5	50			UART0_CTS_b → UART0_CTS_b/ UART0_COL_b					
M8	55			(+) USB_CLKIN				I2S0_RX_BCLK → I2S0_TX_BCLK	
J7	58	(+) ADC3_SE6a		(+) ULPI_CLK		(+) I2S1_RXD0	(+) CLKOUT		
J8	59			(+) ULPI_DIR		(+) I2S1_RX_BCLK			
K8	60			(+) ULPI_NXT		(+) I2S1_RX_FS			
L8	61	(+) ADC3_SE5a		(+) ULPI_STP					
M9	62	(+) ADC3_SE4a		(+) ULPI_DATA0					
L9	63	(+) ADC3_SE15		(+) ULPI_DATA1					
K9	64							I2S0_TXD → I2S0_TXD0	

下一页继续介绍此表...

L10	66	(+) CMP3_IN0						I2S0_TX_BCLK → I2S0_RX_BCLK	(+) I2S0_TXD1
L11	67	(+) CMP3_IN1						I2S0_RXD → I2S0_RXD0	
K10	68	(+) CMP3_IN2			UART0_CTS_b →UART0_CTS_b/ UART0_COL_b				(+) I2S0_RXD1
K11	69								(-) I2S0_CLKIN
M12	72	EXTAL → EXTAL0							
M11	73	XTAL → XTAL0						LPT0ALT1 → LPTMR0_ALT1	
K12	75	(+) CMP3_IN4		(+) ULPI_DATA2					
J12	76	(+) CMP3_IN5		(+) ULPI_DATA3					
J11	77	(+) ADC2_SE15		(+) ULPI_DATA4					
J10	78	(+) ADC2_SE14		(+) ULPI_DATA5					
H12	79	(+) ADC2_SE13		(+) ULPI_DATA6					
H11	80	(+) ADC2_SE12		(+) ULPI_DATA7					
H10	81	ADC0_SE8/ ADC1_SE8/ TSI0_CH0 → ADC0_SE8/ ADC1_SE8/ ADC2_SE8/							

下一页继续介绍此表...

		ADC3_SE8/ TSI0_CH0							
H9	82	ADC0_SE9/ ADC1_SE9/ TSI0_CH6 → ADC0_SE9/ ADC1_SE9/ ADC2_SE9/ ADC3_SE9/ TSI0_CH6							
G11	84				UART0_CTS_b → UART0_CTS_b/ UART0_COL_b				
E12	91					(+) I2S1_TX_BCLK			
E11	92					(+) I2S1_TX_FS			
E10	95					(+) I2S1_TXD0			
E9	96					(+) I2S1_TXD1			
D10	99	(+) ADC2_SE4a					FB_AD31 → FB_AD31/ NFC_DATA15		
D9	100	(+) ADC2_SE5a					FB_AD30 → FB_AD310 NFC_DATA14		
C12	101						FB_AD29 → FB_AD29/ NFC_DATA13		

下一页继续介绍此表...

C11	102						FB_AD28 → FB_AD28/ NFC_DAT A12	(+) CMP3_OU T1	
B12	103					(-) I2S0_TXD	FB_AD14 → FB_AD14/ NFC_DAT A11	(+) I2S0_TXD 1	
B11	104						FB_AD13 → FB_AD13/ NFC_DAT A10	(+) I2S0_TXD 0	
A12	105						FB_AD12 → FB_AD12/ NFC_DAT A9	(+) I2S0_TX_F S	
A11	106						FB_CLKO UT → CLKOUT	(+) I2S0_TX_ BCLK	
A9	109						FB_AD11 → FB_AD11/ NFC_DAT A8		(+) I2S1_TX_ BCLK
D8	110				(+) LPTMR0_ ALT2	LPTMR0_ ALT2 → I2S0_RXD 0	FB_AD10 → FB_AD10/ NFC_DAT A7		(+) I2S1_TX_F S
C8	111					(+) I2S0_RX_ BCLK	FB_AD9 → FB_AD9/ NFC_DAT A6	(+) I2S0_MCL K	
B8	112				(+) USB_SOF _OUT	(+) I2S0_RX_ FS	FB_AD8 → FB_AD8/ NFC_DAT A5		
A8	113				(-) I2S0_MCL K (+) FTM3_CH 4	I2S0_CLKI N → I2S0_MCL K	FB_AD7 → FB_AD7/ NFC_DAT A4		
D7	114				(+) FTM3_CH 5		FB_AD6 → FB_AD6/ NFC_DAT A3		
C7	115	(-) CMP0_IN4			(+) FTM3_CH 6		FB_AD5 → FB_AD5/ NFC_DAT A2	(+) I2S1_MCL K	

下一页继续介绍此表...

B7	116				(+) FTM3_CH 7	I2S0_RXD → I2S0_RXD 1	FB_RW_b → FB_RW_b/ NFC_WE		
A6	123							(+) NFC_RB	
D5	124							(+) NFC_CE0 _b	
C5	125							(+) NFC_CE1 _b	
A5	127					(+) FTM3_CH 0		(+) I2S1_RXD 1	
D4	128					(+) FTM3_CH 1		(+) I2S1_RXD 0	
C4	129					(+) FTM3_CH 2		(+) I2S1_RX_ FS	
B4	130					(+) FTM3_CH 3		(+) I2S1_RX_ BCLK	
A4	131						FB_AD2 → FB_AD2/ NFC_DAT A1		
A3	132				UART0_C TS_b → UART0_C TS_b/ UART0_C OL_b		FB_AD1 → FB_AD1/ NFC_DAT A0		
C9	137							FB_A16 → FB_A16/ NFC_CLE	
B9	138							FB_A17 → FB_A17/ NFC_ALE	
B3	139							FB_A18 → FB_A18/ NFC_RE	
B1	141				(+) FTM3_FLT 0				

7.2 存储空间映射

7.2.1 存储空间映射—Kinetis 100 MHz 版本 1.x

System 32-bit Address Range	Destination Slave	Access
0x0000_0000–0x0FFF_FFFF	Program flash and read-only data (Includes exception vectors in first 1024 bytes)	All masters
0x1000_0000–0x13FF_FFFF	• For MK60DN256ZVLQ10: Reserved • For MK60DX256ZVLQ10: FlexNVM • For MK60DN512ZVLQ10: Reserved • For MK60DN256ZVMD10: Reserved • For MK60DX256ZVMD10: FlexNVM • For MK60DN512ZVMD10: Reserved	All masters
0x1400_0000–0x17FF_FFFF	For devices with FlexNVM: FlexRAM For devices with program flash only: Programming acceleration RAM	All masters
0x1800_0000–0x1FFF_FFFF	SRAM_L: Lower SRAM (ICODE/DCODE)	All masters
0x2000_0000–0x200F_FFFF	SRAM_U: Upper SRAM bitband region	All masters
0x2010_0000–0x21FF_FFFF	Reserved	–
0x2200_0000–0x23FF_FFFF	Aliased to SRAM_U bitband	Cortex-M4 core only
0x2400_0000–0x3FFF_FFFF	Reserved	–
0x4000_0000–0x4007_FFFF	Bitband region for peripheral bridge 0 (AIPS-Lite0)	Cortex-M4 core & DMA/EzPort
0x4008_0000–0x400F_EFFF	Bitband region for peripheral bridge 1 (AIPS-Lite1)	Cortex-M4 core & DMA/EzPort
0x400F_F000–0x400F_FFFF	Bitband region for general purpose input/output (GPIO)	Cortex-M4 core & DMA/EzPort
0x4010_0000–0x41FF_FFFF	Reserved	–
0x4200_0000–0x43FF_FFFF	Aliased to peripheral bridge (AIPS-Lite) and general purpose input/output (GPIO) bitband	Cortex-M4 core only
0x4400_0000–0x5FFF_FFFF	Reserved	–
0x6000_0000–0x7FFF_FFFF	FlexBus (External Memory - Write-back)	All masters
0x8000_0000–0x9FFF_FFFF	FlexBus (External Memory - Write-through)	All masters
0xA000_0000–0xDFFF_FFFF	FlexBus (External Peripheral - Not executable)	All masters
0xE000_0000–0xE00F_FFFF	Private peripherals	Cortex-M4 core only
0xE010_0000–0xFFFF_FFFF	Reserved	–

7.2.2 存储空间映射—Kinetis 120 MHz

System 32-bit Address Range	Destination Slave	Access	Slave Port
0x0000_0000–0x07FF_FFFF	Program flash and read-only data (Includes exception vectors in first 1024 bytes)	All masters	S0
0x0800_0000–0x0FFF_FFFF	DRAM Controller (Aliased Area)	Cortex-M4 core (M0) only	S5
0x1000_0000–0x13FF_FFFF	FlexNVM	All masters	S0
0x1000_0000–0x13FF_FFFF	Reserved	–	–
0x1400_0000–0x17FF_FFFF	For devices with FlexNVM: FlexRAM	All masters	S0
0x1400_0000–0x17FF_FFFF	For devices with program flash only: Programming acceleration RAM	–	S0
0x1800_0000–0x1BFF_FFFF	FlexBus (Aliased Area). 0x1800_0000-0x1BFF_FFFF are mapped to the same access space of 0x9800_0000-0x9BFF_FFFF.	Cortex-M4 core (M0) only	–
0x1C00_0000–0x1FFF_FFFF	SRAM_L: Lower SRAM (ICODE/DCODE)	All masters	–
0x2000_0000–0x200F_FFFF	SRAM_U: Upper SRAM bitband region	All masters	–
0x2010_0000–0x21FF_FFFF	Reserved	–	–
0x2200_0000–0x23FF_FFFF	Aliased to TCMU SRAM bitband	Cortex-M4 core only	–
0x2400_0000–0x3FFF_FFFF	Reserved	–	–
0x4000_0000–0x4007_FFFF	Bitband region for AIPS0	Cortex-M4 core & DMA/EzPort	S2
0x4008_0000–0x400F_EFFF	Bitband region for AIPS1	Cortex-M4 core & DMA/EzPort	S3
0x400F_F000–0x400F_FFFF	Bitband region for GPIO	Cortex-M4 core & DMA/EzPort	S3
0x4010_0000–0x41FF_FFFF	Reserved	–	–
0x4200_0000–0x43FF_FFFF	Aliased to AIPS and GPIO bitband	Cortex-M4 core only	–
0x4400_0000–0x5FFF_FFFF	Reserved	–	–
0x6000_0000–0x6FFF_FFFF	Flexbus (External memory - Write-back)	All masters	S4
0x7000_0000–0x7FFF_FFFF	DRAM Controller - Write-back	Cortex-M4 core (M1), eDMA(M2)	S5, S6, and S7
0x7000_0000–0x7FFF_FFFF	DRAM Controller	LCD (M4)	S5, S6, and S7
0x7000_0000–0x7FFF_FFFF	DRAM Controller	LCD(M5), eSDHC/NFC (M3), ENET(M7), USB (M6)	S5
0x8000_0000–0x8FFF_FFFF	DRAM Controller - Write-through	Cortex-M4 core (M1), eDMA(M2)	
0x8000_0000–0x8FFF_FFFF	DRAM Controller	LCD (M4)	S5, S6, and S7
0x8000_0000–0x8FFF_FFFF	DRAM Controller	LCD (M5), eSDHC/NFC (M3), ENET(M7), USB (M6)	S5, S6, and S7
0x9000_0000–0x9FFF_FFFF	FlexBus (External memory - Write-through)	All masters	S4
0xA000_0000–0xDFFF_FFFF	FlexBus (External peripheral - not executable)	All masters	S4
0xE000_0000–0xE00F_FFFF	Private Peripherals	Cortex-M4 core only	–
0xE010_0000–0xFFFF_FFFF	Reserved	–	–

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

本文档中的信息仅供系统和软件实施方使用 Freescale 产品。 本文并未明示或者暗示授予利用本文档信息进行设计或者加工集成电路的版权许可。 Freescale 保留对此处任何产品进行更改的权利，恕不另行通知。

Freescale 对其产品在任何特定用途方面的适用性不做任何担保、表示或保证，也不承担因为应用程序或者使用产品或电路所产生的任何责任，明确拒绝承担包括但不局限于后果性的或附带性的损害在内的所有责任。 Freescale 的数据表和/或规格中所提供的“典型”参数在不同应用中可能并且确实不同，实际性能会随时间而有所变化。 所有运行参数，包括“经典值”在内，必须经由客户的技术专家对每个客户的应用程序进行验证。 Freescale 未转让与其专利权及其他权利相关的许可。 Freescale 销售产品时遵循以下网址中包含的标准销售条款和条件：freescale.com/SalesTermsandConditions.

Document Number AN4526
Revision 1.0, March 2013