



UM10441

LPC1224/25/26/27 用户手册

修订版 2— 2011 年 9 月 19 日

用户手册

文档信息

信息	内容
关键字	LPC122x、LPC1227、LPC1226、LPC1225、LPC1224、LPC12D27、ARM Cortex-M0、微控制器
摘要	LPC122x 用户手册



修订记录

修订	日期	说明
2	20110919	LPC122x 用户手册
变更内容:		<ul style="list-style-type: none">更新了表 118 中引脚 RTCXOUT 和 RTCXIN（LQFP64 封装）的引脚配置。更新了 IOCON 寄存器表 63 到表 70 的 DRV 位。0 = 当前选定高电平模式，1 = 当前选定低电平模式。修正了 DMA 控制结构的对齐方式（章节 21.7.5 “DMA 控制”）。RTC 功能说明扩展（表 16.6）。增加了器件 LPC12D27。更新了启动逻辑 1 寄存器（表 42 到表 45）。增加了 LPC12D27 引脚配置（表 119）。
1	20110215	LPC122x 用户手册

联系信息

有关详细信息，请访问：<http://www.nxp.com>

欲咨询销售办事处地址，请发送电子邮件至：salesaddresses@nxp.com

1.1 简介

LPC122x 扩展了恩智浦 32 位 ARM 微控制器系列产品，在工厂和家庭自动化领域具有广泛的工业用途。LPC122x 得益于 ARM Cortex-M0 Thumb 指令集，相比执行一般任务的常用 8/16 位微控制器，代码密度高出 50 %。LPC122x 的另一个特色是针对 Cortex-M0 优化的 ROM 型除法库算术，其算法性能比软件型除法库高出数倍，并且具有确定性极高的周期时间和更小的闪存代码。ARM Cortex-M0 的高效性还有助于 LPC122x 在类似应用中达到较低的平均功率。

LPC122x 的 CPU 工作频率高达 45MHz。其提供的闪存选择范围广泛，从 32 kB 到 128 kB。闪存的较小 512 字节页擦除具有多种设计优点，如更精细的 EEPROM 模拟，来自任何串行接口的引导加载支持，以及减少了片上 RAM 缓冲器需求的轻松现场编程。

LPC122x 的外设补充包括一个 10 位 ADC、两个具有输出反馈环路的比较器、两个 UART、一个 SSP/SPI 接口、一个具有超快速模式功能的 I²C 总线接口、一个视窗化看门狗定时器、一个 DMA 控制器、一个 CRC 引擎、四个通用定时器、一个 32 位 RTC、一个用于波特率生成的 1% 内部振荡器和最多 55 个通用 I/O (GPIO) 引脚。

器件 LPC1227 可用作双芯片模块，将 LPC1227 与 PCF8576D LCD 驱动器相集成。

1.2 功能特点

- 处理器内核
 - ARM Cortex-M0 处理器，以高达 45 MHz（来自闪存的一等待状态）或 30 MHz（来自闪存的零等待状态）的频率运行。LPC122x 在 CoreMark CPU 性能评测测试中获得了超过 45 分的高分，相当于 1.51/MHz。
 - ARM Cortex-M0 内置可嵌套中断向量控制器 (NVIC)。
 - 串行调试接口 (SWD)。
 - 系统节拍定时器。
- 内存
 - 最多 8 kB SRAM。
 - 最高 128 KB 片内闪存编程内存。
 - 通过片内 bootloader 软件进行的在系统编程 (ISP) 和在应用编程 (IAP)。
 - 包括 ROM 型 32 位整数除法例程。
- 时钟生成单元
 - 操作范围 1 MHz 至 25 MHz 的晶体振荡器。
 - 12 MHz 内部 RC (IRC) 振荡器，调整为 1 % 精度，可选作为系统时钟。
 - PLL 使 CPU 无需高频晶振就达到最大 CPU 运行速率。可以从系统振荡器或内部 RC 振荡器运行。
 - 带分频器的时钟输出功能，可反映系统振荡器时钟、IRC 时钟、主时钟和看门狗时钟。
 - 实时时钟 (RTC)。

- 数字外设
 - 带 21 条通道的 Micro DMA 控制器。
 - CRC 引擎。
 - 两个带有小数波特率生成器和内部 FIFO 的 UART。一个带有 RS-485 和调制解调器支持的 UART 和一个带有 IrDA 的标准 UART。
 - 带 FIFO 和多协议功能的 SSP/SPI 控制器。
 - I²C 总线接口，支持完整的 I²C 总线规范和数据速率为 1 Mbit/s 的超快速模式，具有多地址识别和监控模式。I²C 总线引脚具有可编程干扰滤波器。
 - 最多 55 个通用 I/O (GPIO) 引脚，带有可编程上拉电阻、开漏模式、可编程数字输入干扰滤波器和可编程输入逆变器。
 - 所有 GPIO 引脚上都具有可编程输出驱动器。四个引脚支持高电流输出驱动器。
 - 所有 GPIO 引脚均可用作边沿和电平敏感型中断源。
 - 四个通用计数器 / 定时器，带有四个捕获输入和四个匹配输出（32 位定时器）或两个捕获输入和两个匹配输出（16 位定时器）。
 - 窗口化看门狗定时器 (WWDt)；已通过 IEC-60335 B 类认证。
- 模拟外设
 - 一个 8 通道 10 位 ADC。
 - 两个高度灵活的模拟比较器。比较器输出可编程为触发定时器匹配信号，或者可用于模拟 555 定时器特性。
- 电源
 - 三种低功耗模式：睡眠模式、深度睡眠模式和深度掉电模式。
 - 可通过使用 12 个端口引脚的启动逻辑将处理器从深度睡眠模式中唤醒。
 - 可通过 RTC 将处理器从深度掉电模式和深度睡眠模式中唤醒。
 - 使用三个独立阈值（每个用于中断和强制重置）进行掉电检测。
 - 加电重置 (POR)。
 - 集成 PMU（电源管理单元）。
- 用于标识的唯一性设备序列号。
- 3.3 V 电源。
- 提供 64 引脚和 48 引脚 LQFP 封装。
- 以双芯片模块形式提供，由 LPC1227 单芯片微控制器与 PCF8576D 通用 LCD 驱动器组合构成，采用 100 引脚封装（器件 LPC12D27FBD100/301）。¹

1. 有关 PCF8576D 操作的详细信息，请参见 [Ref. 1](#)。

1.3 订购信息

表 1. 订购信息

型号	封装		
	名称	说明	版本
LPC12D27FBD100/301	LQFP100	极薄的四侧扁平塑料封装； 100 引脚，外型尺寸 14 x 14 x 1.4 mm	sot407-1
LPC1227FBD64/301	LQFP64	LQFP64：极薄的四侧扁平塑料封装； 64 引脚，外型尺寸 10 x 10 x 1.4 mm	SOT314-2
LPC1226FBD64/301	LQFP64	LQFP64：极薄的四侧扁平塑料封装； 64 引脚，外型尺寸 10 x 10 x 1.4 mm	SOT314-2
LPC1225FBD64/321	LQFP64	LQFP64：极薄的四侧扁平塑料封装； 64 引脚，外型尺寸 10 x 10 x 1.4 mm	SOT314-2
LPC1225FBD64/301	LQFP64	LQFP64：极薄的四侧扁平塑料封装； 64 引脚，外型尺寸 10 x 10 x 1.4 mm	SOT314-2
LPC1224FBD64/121	LQFP64	LQFP64：极薄的四侧扁平塑料封装； 64 引脚，外型尺寸 10 x 10 x 1.4 mm	SOT314-2
LPC1224FBD64/101	LQFP64	LQFP64：极薄的四侧扁平塑料封装； 64 引脚，外型尺寸 10 x 10 x 1.4 mm	SOT314-2
LPC1227FBD48/301	LQFP48	LQFP48：极薄的四侧扁平塑料封装； 48 引脚，外型尺寸 7 x 7 x 1.4 mm	SOT313-2
LPC1226FBD48/301	LQFP48	LQFP48：极薄的四侧扁平塑料封装； 48 引脚，外型尺寸 7 x 7 x 1.4 mm	SOT313-2
LPC1225FBD48/321	LQFP48	LQFP48：极薄的四侧扁平塑料封装； 48 引脚，外型尺寸 7 x 7 x 1.4 mm	SOT313-2
LPC1225FBD48/301	LQFP48	LQFP48：极薄的四侧扁平塑料封装； 48 引脚，外型尺寸 7 x 7 x 1.4 mm	SOT313-2
LPC1224FBD48/121	LQFP48	LQFP48：极薄的四侧扁平塑料封装； 48 引脚，外型尺寸 7 x 7 x 1.4 mm	SOT313-2
LPC1224FBD48/101	LQFP48	LQFP48：极薄的四侧扁平塑料封装； 48 引脚，外型尺寸 7 x 7 x 1.4 mm	SOT313-2

1.3.1 器件选择一览

表 2. LPC122x 订购选项

型号	Flash	总 SRAM	UART	I ² C/ FM+	SSP	ADC 通道	GPIO	封装
LPC12D27								
LPC12D27FBD100/301	128 kB	8 kB	1	1	1	8	55	LQFP100
LPC1227								
LPC1227FBD64/301	128 kB	8 kB	2	1	1	8	55	LQFP64
LPC1227FBD48/301	128 kB	8 kB	2	1	1	8	39	LQFP48
LPC1226								
LPC1226FBD64/301	96 kB	8 kB	2	1	1	8	55	LQFP64
LPC1226FBD48/301	96 kB	8 kB	2	1	1	8	39	LQFP48
LPC1225								
LPC1225FBD64/321	80 kB	8 kB	2	1	1	8	55	LQFP64
LPC1225FBD64/301	64 kB	8 kB	2	1	1	8	55	LQFP64
LPC1225FBD48/321	80 kB	8 kB	2	1	1	8	39	LQFP48
LPC1225FBD48/301	64 kB	8 kB	2	1	1	8	39	LQFP48
LPC1224								
LPC1224FBD64/121	48 kB	4 kB	2	1	1	8	55	LQFP64
LPC1224FBD64/101	32 kB	4 kB	2	1	1	8	55	LQFP64
LPC1224FBD48/121	48 kB	4 kB	2	1	1	8	39	LQFP48
LPC1224FBD48/101	32 kB	4 kB	2	1	1	8	39	LQFP48

1.4 框图

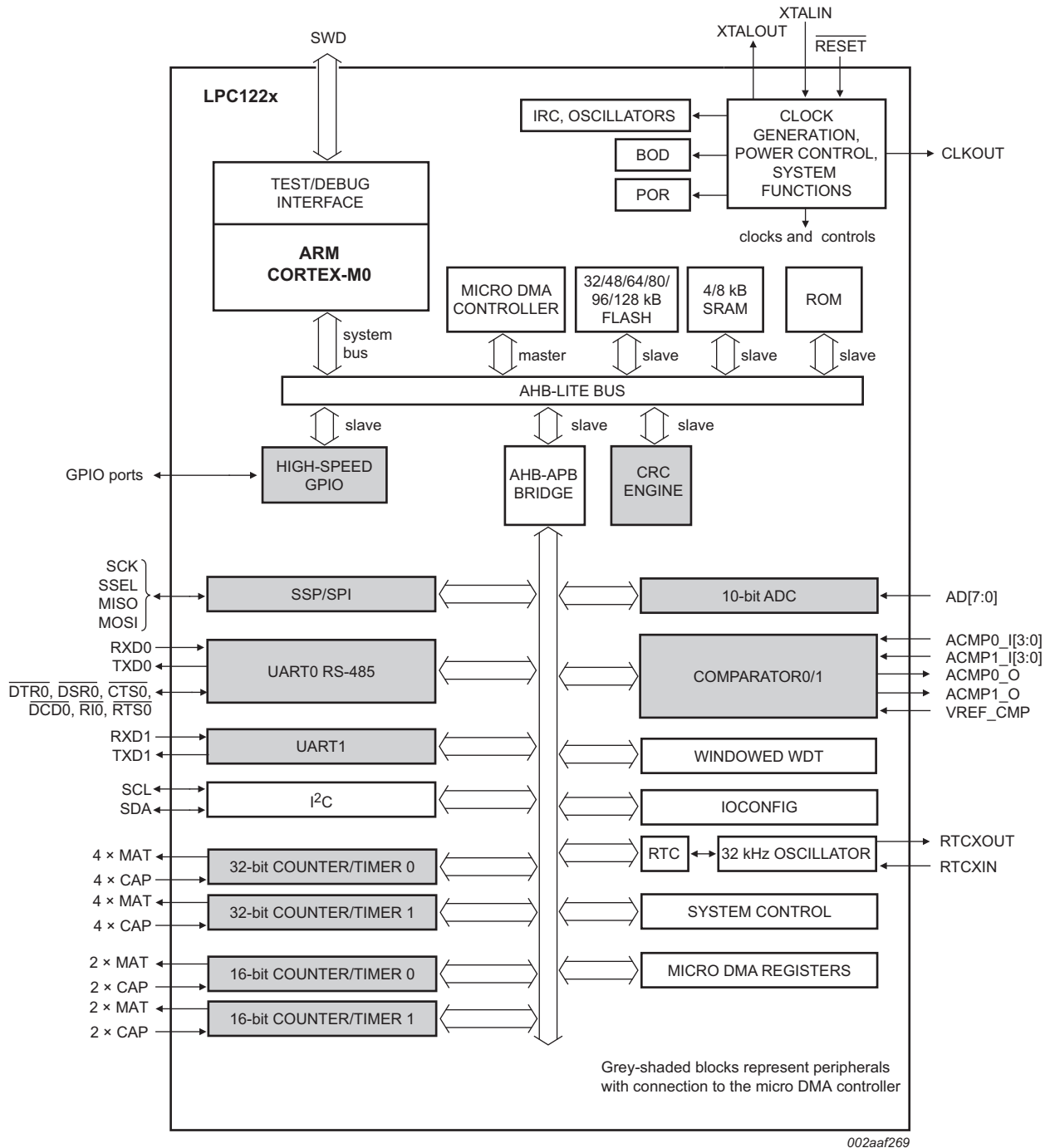


图 1. LPC122x 框图

2.1 本章导读

[表 3](#) 显示了 LPC122x 部件的存储器配置。

表 3. LPC122x 存储器配置

产品型号	闪存	总 SRAM
LPC12D27FBD100/301	128 kB	8 kB
LPC1227FBD64/301	128 kB	8 kB
LPC1227FBD48/301	128 kB	8 kB
LPC1226FBD64/301	96 kB	8 kB
LPC1226FBD48/301	96 kB	8 kB
LPC1225FBD64/321	80 kB	8 kB
LPC1225FBD64/301	64 kB	8 kB
LPC1225FBD48/321	80 kB	8 kB
LPC1225FBD48/301	64 kB	8 kB
LPC1224FBD64/121	48 kB	4 kB
LPC1224FBD64/101	32 kB	4 kB
LPC1224FBD48/121	48 kB	4 kB
LPC1224FBD48/101	32 kB	4 kB

2.2 简介

[图 2](#) 显示了 LPC122x 的存储器和外设地址空间。AHB 外设区的大小为 2 MB，可分配多达 128 个外设。

在 LPC122x 上，GPIO 端口、CRC 引擎和微 DMA 控制器均为 AHB 外设。APB 外设区的大小为 512 kB，可分配多达 32 个外设。每种类型的每一个外设空间的大小均为 16 kB，从而简化了每个外设的地址译码。

所有外设寄存器地址无论规格大小，均为 32 位字对齐。这意味着必须同时访问字和半字寄存器。例如，不能对一个字寄存器的最高字节执行单独的读或写操作。

2.3 存储器分配

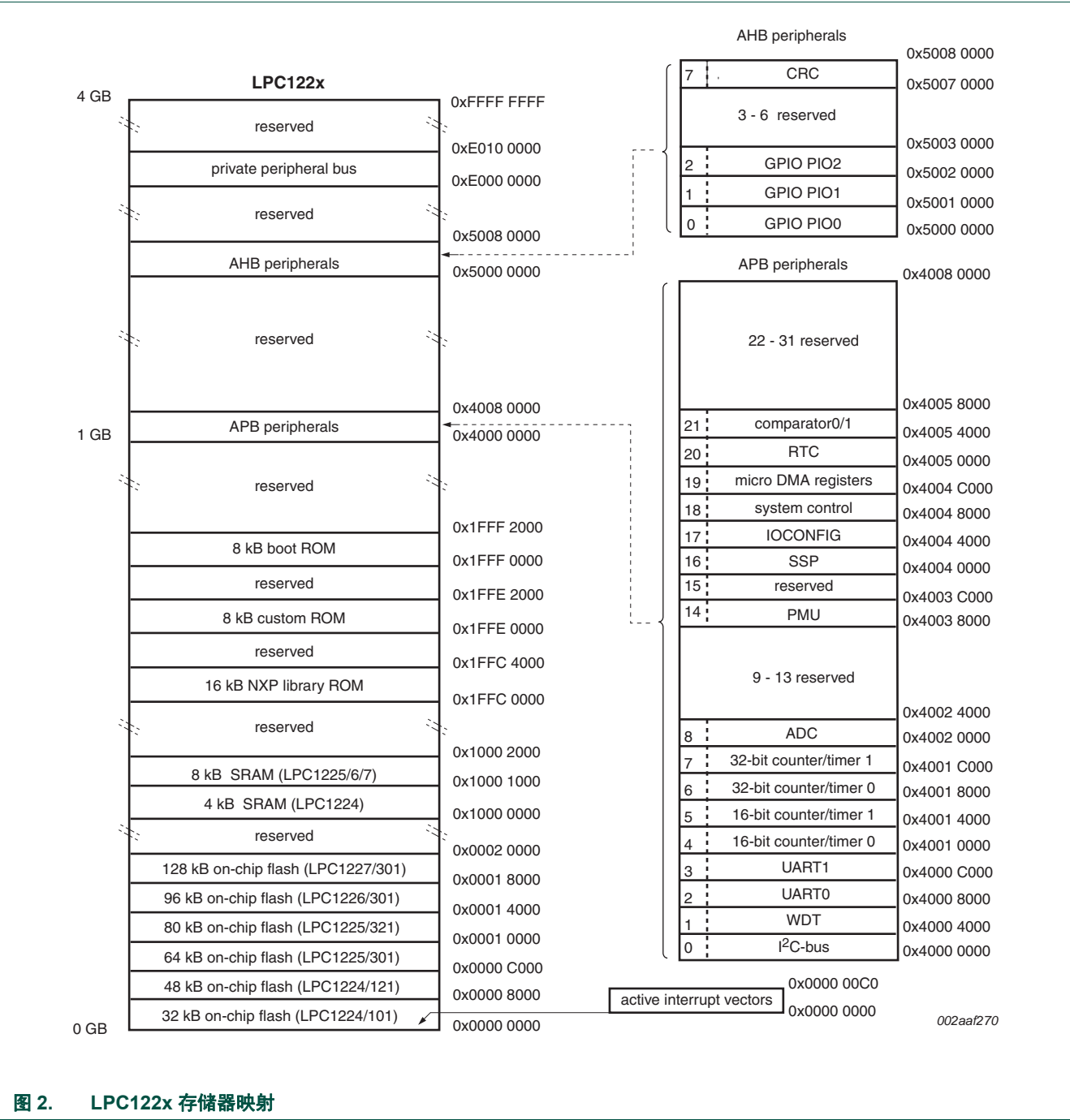


图 2. LPC122x 存储器映射

3.1 本章导读

可嵌套中断向量控制器 (NVIC) 是 ARM Cortex-M0 系统模块的一部分，且在所有 LPC122x 部件上都相同。

3.2 特性

- 可嵌套中断向量控制器是 ARM Cortex-M0 的一个重要组成部分。
- 紧耦合方式使中断延迟大大缩短。
- 可控制系统的异常及外设中断。
- 支持 32 个向量中断。
- 4 个可编程的中断优先级（带硬件优先级屏蔽功能）。
- 软件中断生成功能。
- 带可配置源的非屏蔽中断 (NMI)。

3.3 描述

可嵌套中断向量控制器 (NVIC) 是 Cortex-M0 的一个重要组成部分。它与 CPU 紧密结合，降低了中断延时，并让新进中断可以得到高效处理。

NMI 的源可以配置（参见[章节 4.5.29](#)）。可以针对 NMI 功能选择多个外设中断。

有关 NVIC 运作和寄存器描述的详情，请参阅《ARM Cortex-M0 技术参考手册》和 ARM Cortex-M0 附录（[章节 25.5.2](#)）。

3.4 中断源

[表 4](#) 列出了每种外设功能的中断源。每个外设可以有一条或多条中断线连接到中断向量控制器。每条线可代表多个中断源。除 ARM 的特定标准外，中断线的连接位置没有重要性或优先级的区别。

表 4. 中断源与中断向量控制器的连接

异常编号	功能	标志
11 至 0	启动逻辑唤醒中断	每个中断会连接到一个 PIO 输入引脚上，作为该部件处于深度睡眠模式时的唤醒引脚，中断 0 至 11 对应 PIO0_0 至 PIO0_11（见 表 37 ）。
12	I ² C	SI（状态更改）
13	CT16B0	匹配 3 至 0 捕获 3 至 0
14	CT16B1	匹配 3 至 0 捕获 3 至 0

表 4. 中断源与中断向量控制器的连接

异常编号	功能	标志
15	CT32B0	匹配 3 至 0 捕获 3 至 0
16	CT32B1	匹配 3 至 0 捕获 3 至 0
17	SSP	Tx FIFO 半空 Rx FIFO 半满 Rx 超时 Rx 溢出
18	UART0	Rx 线状态 (RLS) 发送保持寄存器空 (THRE) Rx 数据可用 (RDA) 字符超时指示 (CTI) 自动波特率结束 (ABEO) 自动波特率超时 (ABTO) 调制解调器中断
19	UART1	Rx 线状态 (RLS) 发送保持寄存器空 (THRE) Rx 数据可用 (RDA) 字符超时指示 (CTI) 自动波特率结束 (ABEO) 自动波特率超时 (ABTO)
20	比较器	比较器 0/1 中断
21	ADC	A/D 转换器转换结束
22	WDT	看门狗中断 (WDINT)
23	BOD	掉电检测
24	-	保留
25	PIO0	端口 0 的 GPIO 中断状态
26	PIO1	端口 1 的 GPIO 中断状态
27	PIO2	端口 2 的 GPIO 中断状态
28	-	保留
29	DMA	DMA 请求中断
30	RTC	RTC 中断
31	-	保留

4.1 本章导读

所有 LPC122x 部件上的系统控制模块都相同。

4.2 简介

系统配置模块可控制 LPC122x 的振荡器、启动逻辑和时钟产生。该模块中还包含用于设置 AHB 访问优先级的寄存器和用于重映射闪存、SRAM 和 ROM 内存区域的寄存器。

4.3 引脚描述

[表 5](#) 显示了与系统控制模块功能相关的引脚。

表 5. 引脚摘要

引脚名称	引脚方向	引脚描述
CLKOUT	O	Clockout 引脚
PIO0_0 至 PIO0_11	I	启动逻辑唤醒引脚端口 0

4.4 简介

有关 LPC122x 时钟产生单元 (CGU) 的概述，参见[图 3](#)。

在复位之后，LPC122x 将从内部 RC 振荡器运行，直到通过软件进行切换。这就使得系统可以在没有任何外部晶体的情况下运行，并使引导加载程序代码按照已知频率运行。

SYSAHBCLKCTRL 寄存器可开启各种外设和存储器的系统时钟。UART0/1、SSP、RTC 和 SysTick 定时器具有独立的时钟分频器，以从主时钟获得外设时钟。

看门狗时钟可以从振荡器输出或主时钟获得。

主时钟以及 IRC、系统振荡器和看门狗振荡器的时钟输出均可以直接在 CLKOUT 引脚上观察到。

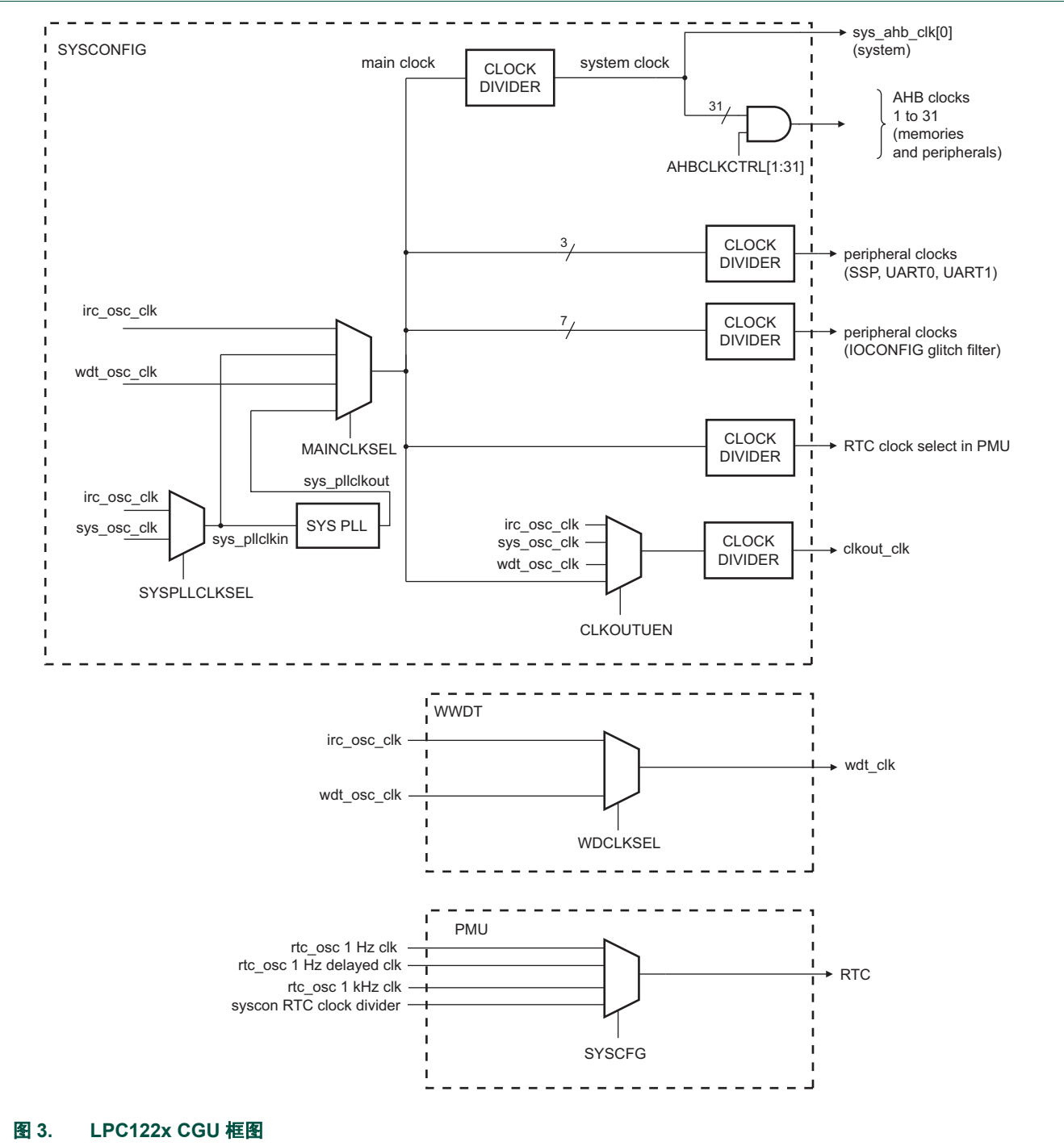


图 3. LPC122x CGU 框图

4.5 寄存器描述

所有寄存器，无论大小，都按字地址边界对齐。寄存器的详细信息都出现在每种功能的描述中。

表 6. 寄存器简介：系统控制模块（基址 0x4004 8000）

名称	访问类型	地址偏移	描述	复位值
SYSMEMREMAP	R/W	0x000	系统存储器重映射	0x0000 0000
PRESETCTRL	R/W	0x004	外设复位控制和闪存定时覆盖	0x0000 FFFF
SYSPLLCTRL	R/W	0x008	系统 PLL 控制	0x0000 0000
SYSPLLSTAT	R	0x00C	系统 PLL 状态	0x0000 0000
-	-	0x010 - 0x01C	保留	-
SYSOSCCTRL	R/W	0x020	系统振荡器控制	0x0000 0000
WDTOSCCTRL	R/W	0x024	看门狗振荡器控制	0x0000 0000
IRCCTRL	R/W	0x028	IRC 控制	0x0000 0080
-	-	0x02C	保留	-
SYSRESSTAT	R/W	0x030	系统复位状态寄存器	0x0000 0000
-	-	0x034 - 0x03C	保留	-
SYSPLLCLKSEL	R/W	0x040	系统 PLL 时钟源选择	0x0000 0000
SYSPLLCLKUEN	R/W	0x044	系统 PLL 时钟源更新使能	0x0000 0000
-	-	0x048 - 0x06C	保留	-
MAINCLKSEL	R/W	0x070	主时钟源选择	0x0000 0000
MAINCLKUEN	R/W	0x074	主时钟源更新使能	0x0000 0000
SysAHBCLKDIV	R/W	0x078	系统 AHB 时钟分频器	0x0000 0001
-	-	0x07C	保留	-
SysAHBCLKCTRL	R/W	0x080	系统 AHB 时钟控制	0xF01F FFFF
-	-	0x084 - 0x08C	保留	-
-	-	0x090	保留	-
SSPCLKDIV	R/W	0x094	SSP 时钟分频器	0x0000 0000
UART0CLKDIV	R/W	0x098	UART0 时钟分频器	0x0000 0000
UART1CLKDIV	R/W	0x09C	UART1 时钟分频器	0x0000 0000
RTCCLKDIV	R/W	0x0A0	RTC 时钟分频器	0x0000 0000
-	-	0x0A4 - 0x0DC	保留	-
CLKOUTCLKSEL	R/W	0x0E0	CLKOUT 时钟源选择	0x0000 0000
CLKOUTUEN	R/W	0x0E4	CLKOUT 时钟源更新使能	0x0000 0000
CLKOUTDIV	R/W	0x0E8	CLKOUT 时钟分频器	0x0000 0000
-	-	0x0EC - 0x0FC	保留	-
PIOPORCAP0	R	0x100	POR 捕获 PIO 状态 0	由用户决定
PIOPORCAP1	R	0x104	POR 捕获 PIO 状态 1	由用户决定
-	-	0x108 - 0x130	保留	-
IOCONFIGCLKDIV6	R/W	0x134	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 6	0x0000 0000

表 6. 寄存器简介：系统控制模块（基址 0x4004 8000）（续）

名称	访问类型	地址偏移	描述	复位值
IOCONFIGCLKDIV5	R/W	0x138	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 5	0x0000 0000
IOCONFIGCLKDIV4	R/W	0x13C	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 4	0x0000 0000
IOCONFIGCLKDIV3	R/W	0x140	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 3	0x0000 0000
IOCONFIGCLKDIV2	R/W	0x144	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 2	0x0000 0000
IOCONFIGCLKDIV1	R/W	0x148	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 1	0x0000 0000
IOCONFIGCLKDIV0	R/W	0x14C	用于可编程干扰滤波器的 IOCONFIG 模块外设时钟 0	0x0000 0000
BODCTRL	R/W	0x150	BOD 控制	未受系统复位影响
SYSTCKCAL	R/W	0x154	系统节拍计数器校准	0x0000 001F
AHBPRIO	-	0x158	AHB 优先级设置	0x0000 0004
-	-	0x15C - 0x16C	保留	-
IRQLATENCY	R/W	0x170	四分位距 (IQR) 延迟。使中断延迟和确定性之间达到平衡。	0x0000 0010
INTNMI	R/W	0x174	NMI 中断源配置控制	0x0000 003F
-	-	0x178 - 0x1FC	保留	-
STARTAPRP0	R/W	0x200	启动逻辑边沿控制寄存器 0	0x0000 0000
STARTERP0	R/W	0x204	启动逻辑信号使能寄存器 0	0x0000 0000
STARTRSRP0CLR	W	0x208	启动逻辑复位寄存器 0	0x0000 0000
STARTSRP0	R	0x20C	启动逻辑状态寄存器 0	0x0000 0000
STARTAPRP1	R/W	0x210	启动逻辑边沿控制寄存器 1；外设中断	0x0000 0000
STARTERP1	R/W	0x214	启动逻辑信号使能寄存器 1；外设中断	0x0000 0000
STARTRSRP1CLR	W	0x218	启动逻辑复位寄存器 1；外设中断	0x0000 0000
STARTSRP1	R	0x21C	启动逻辑状态寄存器 1；外设中断	0x0000 0000
-	-	0x220 - 0x22C	保留	-
PDSLEEP_CFG	R/W	0x230	深度睡眠模式掉电状态	0x0000 FFFF
PDAWAKE_CFG	R/W	0x234	从深度睡眠模式唤醒以后的掉电状态	0x0000 FDF0
PDRUN_CFG	R/W	0x238	掉电配置寄存器	0x0000 FDF0
-	-	0x23C - 0x3F0	保留	-
DEVICE_ID	R	0x3F4	器件 ID	取决于零部件

注：闪存配置模块存在于其自己的内存区域中，但与系统控制寄存器一起列出。

表 7. 寄存器简介：闪存配置（基址 0x5006 0000）

名称	访问类型	地址偏移	描述	复位值
FLASHCFG	R/W	0x028	闪存配置寄存器	0x0000 0000

4.5.1 系统存储器重映射寄存器

系统存储器重映射寄存器选择是否从 Boot ROM、闪存或 SRAM 读取 ARM 中断向量。

表 8. 系统存储器重映射寄存器（SYSMEMREMAP、地址 0x4004 8000）位描述

位	符号	值	描述	复位值
1:0	MAP		系统存储器重映射。保留值 0x3。	00
		0x0	引导加载程序模式。中断向量重映射到 Boot ROM。	
		0x1	用户 RAM 模式。中断向量重映射到静态 RAM。	
		0x2	用户闪存模式。中断向量不会被重映射，一直位于闪存中。	
31:2	-	-	保留	0x00

4.5.2 外设复位控制寄存器

该寄存器使软件可以复位单个外设。如果该寄存器中的位设置为 0，则相应外设将被复位。写 1 会取消复位。

该寄存器的位 15 会覆盖读取访问的闪存定时。

表 9. 外设复位控制寄存器（PRESETCTRL、地址 0x4004 8004）位描述

位	符号	值	描述	复位值
0	SSP_RST_N		SSP 复位控制	1
		0	SSP 复位已启用	
		1	SSP 复位已取消	
1	I2C_RST_N		I2C 复位控制	1
		0	I2C 复位已启用	
		1	I2C 复位已取消	
2	UART0_RST_N		UART0 复位控制	1
		0	UART0 复位已启用	
		1	UART0 复位已取消	
3	UART1_RST_N		UART1 复位控制	1
		0	UART1 复位已启用	
		1	UART1 复位已取消	
4	CT16B0_RST_N		16 位计数器 / 定时器 0(CT16B0) 复位控制	1
		0	CT16B0 复位已启用	
		1	CT16B0 复位已取消	
5	CT16B1_RST_N		16 位计数器 / 定时器 1(CT16B1) 复位控制	1
		0	CT16B1 复位已启用	
		1	CT16B1 复位已取消	
6	CT32B0_RST_N		32 位计数器 / 定时器 0(CT32B0) 复位控制	1
		0	CT32B0 复位已启用	
		1	CT32B0 复位已取消	

表 9. 外设复位控制寄存器 (PRESETCTRL、地址 0x4004 8004) 位描述 (续)

位	符号	值	描述	复位值
7	CT32B1_RST_N		32 位计数器 / 定时器 1(CT32B1) 复位控制	1
		0	CT32B1 复位已启用	
		1	CT32B1 复位已取消	
8	CMP_RST_N		比较器复位控制	1
		0	比较器复位已启用	
		1	比较器复位已取消	
9	CRC_RST_N		CRC 复位控制	1
		0	CRC 复位已启用	
		1	CRC 复位已取消	
10	DMA_RST_N		微型 DMA 复位控制	1
		0	DMA 复位已启用	
		1	DMA 复位已取消	
14:11	-	-	保留	-
15	FLASH_OVERRIDE		闪存读取模式。默认值为 1 次闪存读取访问。在工作频率更高时，必须使用多周期读取模式，以允许 2、3、4 或 5 次读取配置。如果选择多周期读取模式，则可以在 FLASHCFG 寄存器中配置周期数（参见表 52）。	1
		0	闪存多周期读取模式。	
		1	闪存单周期读取模式。	
31:16	-	-	保留	-

4.5.3 系统 PLL 控制寄存器

该寄存器用于连接和启用系统 PLL，并配置 PLL 倍频器和分频器的值。PLL 可从各种时钟源接收 10 MHz 到 25 MHz 的输入频率。将输入频率倍增至较高的频率，再进行分频，以提供 CPU、外设和存储器实际使用的时钟。PLL 可产生 CPU 允许的最大时钟频率。

表 10. 系统 PLL 控制寄存器 (SYSPLLCTRL、地址 0x4004 8008) 位描述

位	符号	值	描述	复位值
4:0	MSEL		反馈分频器值。分频值 M 是编程的 MSEL 值 + 1.00000: 分频比 M = 1 到 11111: 分频比 M = 32	00000
6:5	PSEL		后置分频器比率 P。分频比为 2 × P。	00
		0x0	P = 1	
		0x1	P = 2	
		0x2	P = 4	
		0x3	P = 8	
31:7	-	-	保留	0x00

4.5.4 系统 PLL 状态寄存器

该寄存器是只读寄存器，提供 PLL 锁定状态（参见[章节 4.10.1](#)）。

表 11. 系统 PLL 状态寄存器（SYSPLLSTAT、地址 0x4004 800C）位描述

位	符号	值	描述	复位值
0	LOCK		PLL 锁定状态	0
		0	PLL 未锁定	
		1	PLL 锁定	
31:1	-	-	保留	0x00

4.5.5 系统振荡器控制寄存器

该寄存器可配置系统振荡器的频率范围。

表 12. 系统振荡器控制寄存器（SYSOSCCTRL、地址 0x4004 8020）位描述

位	符号	值	描述	复位值
0	BYPASS		旁路系统振荡器	0
		0	振荡器未被旁路。	
		1	振荡器被旁路。PLL 输入 (sys_osc_clk) 直接由 XTALIN 和 XTALOUT 引脚提供。	
1	FREQRANGE		决定低功耗振荡器的频率范围。	0
		0	1 - 20 MHz 频率范围。	
		1	15 - 25 MHz 频率范围。	
31:2	-	-	保留	0x00

4.5.6 看门狗振荡器控制寄存器

该寄存器可配置看门狗振荡器。该振荡器包含模拟和数字两个部分。模拟部分含有振荡器功能，可以生成模拟时钟 (Fclkana)。通过数字部分，模拟输出时钟 (Fclkana) 可以分成所需的输出时钟频率 wdt_osc_clk。模拟输出频率 (Fclkana) 可以根据 FREQSEL 位在 500 kHz 到 3.4 MHz 之间调整。通过数字部分，使用 DIVSEL 位将 Fclkana 分成（分频器比率 = 2、4、...、64）wdt_osc_clk。

看门狗振荡器的输出时钟频率可以根据以下公式进行计算： $wdt_osc_clk = Fclkana / (2 \times (1 + DIVSEL)) = 7.8\text{ kHz} - 1.7\text{ MHz}$ （标称值）。

注：任何 FREQSEL 位的设置产生的 Fclkana 值都在所列出频率值的 ± 40% 范围内。看门狗振荡器是功耗最低的时钟源。如果需要精确计时，请使用 IRC 或系统时钟。

注：复位后，看门狗振荡器的频率未定义。使用看门狗振荡器之前，必须通过写入 WDTOSCCTRL 寄存器对看门狗振荡器频率进行编程。

表 13. 看门狗振荡器控制寄存器（WDTOSCCTRL、地址 0x4004 8024）位描述

位	符号	值	描述	复位值
4:0	DIVSEL		为 Fclkana 选择分频器。 $wdt_osc_clk = Fclkana / (2 \times (1 + DIVSEL))$ 00000: $2 \times (1 + DIVSEL) = 2$ 00001: $2 \times (1 + DIVSEL) = 4$ 到 11111: $2 \times (1 + DIVSEL) = 64$	0

表 13. 看门狗振荡器控制寄存器（WDTOSCCTRL、地址 0x4004 8024）位描述（续）

位	符号	值	描述	复位值
8:5	FREQSEL		选择看门狗振荡器模拟输出频率 (Fclkana)。	0
		0x1	0.5 MHz	
		0x2	0.8 MHz	
		0x3	1.1 MHz	
		0x4	1.4 MHz	
		0x5	1.6 MHz	
		0x6	1.8 MHz	
		0x7	2.0 MHz	
		0x8	2.2 MHz	
		0x9	2.4 MHz	
		0xA	2.6 MHz	
		0xB	2.7 MHz	
		0xC	2.9 MHz	
		0xD	3.1 MHz	
		0xE	3.2 MHz	
		0xF	3.4 MHz	
31:9	-	-	保留	-

4.5.7 内部共振晶体控制寄存器

该寄存器用于对片内 12 MHz 振荡器进行调整。该调整值为出厂预设值，在启动时由启动代码写入。

表 14. 内部共振晶体控制寄存器（IRCCTRL、地址 0x4004 8028）位描述

位	符号	描述	复位值
7:0	TRIM	调整值	0x000 0080，然后闪存将重新编程
31:9	-	保留	0x00

4.5.8 系统复位状态寄存器

SYSRSTSTAT 寄存器显示最近一个复位事件的来源。这些位可以通过给任何一个位写 “1” 清除。POR 事件可以清除该寄存器中的所有其他位，但如果取消 POR 信号后，其他复位信号（例如，EXTRST）仍然有效，则其位将设置为 “检测到”。

表 15. 系统复位状态寄存器（SYSRESSTAT、地址 0x4004 8030）位描述

位	符号	值	描述	复位值
0	POR		POR 复位状态	0
		0	未检测到 POR	
		1	检测到 POR	
1	EXTRST		复位状态	0
		0	未检测到 RESET 事件	
		1	检测到 RESET	

表 15. 系统复位状态寄存器 (SYSRESSTAT、地址 0x4004 8030) 位描述 (续)

位	符号	值	描述	复位值
2	WDT		看门狗复位状态	0
		0	未检测到 WDT 复位	
		1	检测到 WDT 复位	
3	BOD		掉电检测复位状态	0
		0	未检测到 BOD 复位	
		1	检测到 BOD 复位	
4	SYSRST		软件系统复位状态。使用 $\overline{\text{RESET}}$ 引脚，ARM 软件复位具有与硬件复位一样的效果。	0
		0	未检测到系统复位	
		1	检测到系统复位	
31:5	-	-	保留	0x00

4.5.9 系统 PLL 时钟源选择寄存器

该寄存器为系统 PLL 选择时钟源。SYSPLLCLKUEN 寄存器（参见[章节 4.5.10](#)）必须从低电平转换到高电平，才能使更新生效。

注：切换时钟源时，必须先运行两个时钟，然后才能更新时钟源。

表 16. 系统 PLL 时钟源选择寄存器 (SYSPLLCLKSEL、地址 0x4004 8040) 位描述

位	符号	值	描述	复位值
1:0	SEL		系统 PLL 时钟源	0x00
		0x0	IRC 振荡器	
		0x1	系统振荡器	
		0x2	保留	
		0x3	保留	
31:2	-	-	保留	0x00

4.5.10 系统 PLL 时钟源更新使能寄存器

写入 SYSPLLCLKSEL 寄存器后，该寄存器可以使用新输入时钟更新系统 PLL 的时钟源。必须先向 SYSPLLUEN 寄存器写入 0，再向 SYSPLLUEN 写入 1，才能使更新生效。

表 17. 系统 PLL 时钟源更新使能寄存器 (SYSPLLCLKUEN、地址 0x4004 8044) 位描述

位	符号	值	描述	复位值
0	ENA		启用系统 PLL 时钟源更新	0
		0	无变化	
		1	更新时钟源	
31:1	-	-	保留	0x00

4.5.11 主时钟源选择寄存器

该寄存器可选择主系统时钟，而所选时钟可以任意输入到系统 PLL，也可以从系统 PLL(sys_pllclkout)、看门狗或 IRC 振荡器直接输出。主系统时钟可以为内核、外设和存储器计时。

MAINCLKUEN 寄存器（参见[章节 4.5.12](#)）必须从低电平转换到高电平，才能使更新生效。

注：切换时钟源时，必须先运行两个时钟，然后才能更新时钟源。

表 18. 主时钟源选择寄存器（MAINCLKSEL、地址 0x4004 8070）位描述

位	符号	值	描述	复位值
1:0	SEL		主时钟的时钟源	00
		0x0	IRC 振荡器	
		0x1	系统 PLL 时钟输入	
		0x2	WDT 振荡器	
		0x3	系统 PLL 时钟输出	
31:2	-	-	保留	0x00

4.5.12 主时钟源更新使能寄存器

写到 MAINCLKSEL 寄存器后，该寄存器可以使用新输入时钟更新主时钟的时钟源。必须先向 MAINCLKUEN 寄存器写入 0，再向 MAINCLKUEN 写入 1，才能使更新生效。

表 19. 主时钟源更新使能寄存器（MAINCLKUEN、地址 0x4004 8074）位描述

位	符号	值	描述	复位值
0	ENA		启用主时钟源更新	0x0
		0	无变化	
		1	更新时钟源	
31:1	-	-	保留	0x00

4.5.13 系统 AHB 时钟分频寄存器

该寄存器对主时钟进行分频，以向内核、存储器和外设提供系统时钟。可以通过将 DIV 位设置为 0x0 完全关闭系统时钟。

表 20. 系统 AHB 时钟分频寄存器（SYSAHBCLKDIV、地址 0x4004 8078）位描述

位	符号	描述	复位值
7:0	DIV	系统 AHB 时钟分频器值 0：系统时钟禁用。 1：除以 1 到 255：除以 255。	1
31:8	-	保留	0x00

4.5.14 系统 AHB 时钟控制寄存器

AHBCLKCTRL 寄存器可启用单个系统和外设模块的时钟。系统时钟（sys_ahb_clk[0]，AHBCLKCTRL 寄存器的位 0）为 AHB 到 APB 桥接、AHB 矩阵、ARM Cortex-M0、SYSCON 模块和 PMU 提供时钟。无法禁用该时钟。

表 21. 系统 AHB 时钟控制寄存器（SYSAHBCLKCTRL、地址 0x4004 8080）位描述

位	符号	值	描述	复位值
0	SYS		启用 AHB 到 APB 桥接、AHB 矩阵、Cortex-M0 FCLK 1 和 HCLK、SysCon 及 PMU 的时钟。该位是只读位。	1
		0	保留	
		1	启用	

表 21. 系统 AHB 时钟控制寄存器 (SYSAHBCLKCTRL、地址 0x4004 8080) 位描述 (续)

位	符号	值	描述	复位值
1	ROM		启用 ROM 时钟。	1
		0	禁用	
		1	启用	
2	RAM		启用 RAM 时钟。	1
		0	禁用	
		1	启用	
3	FLASHREG		启用闪存控制器寄存器的时钟。	1
		0	禁用	
		1	启用	
4	FLASHARRAY		启用闪存阵列时钟。	1
		0	禁用	
		1	启用	
5	I2C		启用 I2C 时钟。	1
		0	禁用	
		1	启用	
6	CRC		启用 CRC 时钟。	1
		0	禁用	
		1	启用	
7	CT16B0		启用 16 位计数器 / 定时器 0 的时钟。	1
		0	禁用	
		1	启用	
8	CT16B1		启用 16 位计数器 / 定时器 1 的时钟。	1
		0	禁用	
		1	启用	
9	CT32B0		启用 32 位计数器 / 定时器 0 的时钟。	1
		0	禁用	
		1	启用	
10	CT32B1		启用 32 位计数器 / 定时器 1 的时钟。	1
		0	禁用	
		1	启用	
11	SSP		启用 SSP 时钟。	1
		0	禁用	
		1	启用	
12	UART0		启用 UART0 的时钟。注：启用 UART0 时钟之前必须在 IOCON 模块中将 UART0 引脚配置好。	1
		0	禁用	
		1	启用	
13	UART1		启用 UART1 的时钟。注：启用 UART1 时钟之前必须在 IOCON 模块中将 UART1 引脚配置好。	1
		0	禁用	
		1	启用	

表 21. 系统 AHB 时钟控制寄存器 (SYSAHBCLKCTRL、地址 0x4004 8080) 位描述 (续)

位	符号	值	描述	复位值
14	ADC		启用 ADC 时钟。	1
		0	禁用	
		1	启用	
15	WDT		启用 WDT 时钟。	1
		0	禁用	
		1	启用	
16	IOCON		启用 IO 配置模块的时钟。	1
		0	禁用	
		1	启用	
17	DMA		启用微型 DMA 时钟。	1
		0	禁用	
		1	启用	
18	-		保留。写为 0。	1
19	RTC		启用 RTC 时钟。 注： 如果 RTC 时钟源被禁用，则要先选择 RTC 时钟源，然后才能通过此位重新启用 RTC（参见表 58）。	1
		0	禁用	
		1	启用	
20	CMP		启用比较器的时钟。	1
		0	禁用	
		1	启用	
27:21	-	-	保留	0
28	-	-	保留。写为 0。	1
29	GPIO2		启用 GPIO 端口 2 的时钟。	1
		0	禁用	
		1	启用	
30	GPIO1		启用 GPIO 端口 1 的时钟。	1
		0	禁用	
		1	启用	
31	GPIO0		启用 GPIO 端口 0 的时钟。	1
		0	禁用	
		1	启用	

4.5.15 SSP 时钟分频寄存器

该寄存器可配置 SSP 外设时钟 SSP_PCLK。可以通过将 DIV 位设置为 0x0 关闭 SSP_PCLK。

表 22. SSP 时钟分频寄存器（SSPCLKDIV、地址 0x4004 8094）位描述

位	符号	描述	复位值
7:0	DIV	SSP0_PCLK 时钟分频器值 0：禁用 SSP0_PCLK。 1：除以 1 到 255：除以 255。	0
31:8	-	保留	0x00

4.5.16 UART0 时钟分频寄存器

该寄存器可配置 UART0 外设时钟 UART0_PCLK。可以通过将 DIV 位设置为 0x0 关闭 UART0_PCLK。

注：请注意，启用 UART0 时钟之前必须在 IOCON 模块中将 UART0 引脚配置好。

表 23. UART0 时钟分频寄存器（UART0CLKDIV、地址 0x4004 8098）位描述

位	符号	描述	复位值
7:0	DIV	UART0_PCLK 时钟分频器值 0：禁用 UART0_PCLK。 1：除以 1 到 255：除以 255。	0
31:8	-	保留	0x00

4.5.17 UART1 时钟分频寄存器

该寄存器可配置 UART1 外设时钟 UART1_PCLK。可以通过将 DIV 位设置为 0x0 关闭 UART1_PCLK。

注：请注意，启用 UART1 时钟之前必须在 IOCON 模块中将 UART1 引脚配置好。

表 24. UART1 时钟分频寄存器（UART1CLKDIV、地址 0x4004 809C）位描述

位	符号	描述	复位值
7:0	DIV	UART1_PCLK 时钟分频器值 0：禁用 UART1_PCLK。 1：除以 1 到 255：除以 255。	0
31:8	-	保留	0x00

4.5.18 RTC 时钟分频寄存器

该寄存器可配置RTC外设时钟RTC_PCLK。可以通过将DIV位设置为0x0关闭RTC_PCLK。

表 25. RTC 时钟分频寄存器（RTCCLKDIV、地址 0x4004 80A0）位描述

位	符号	描述	复位值
7:0	DIV	RTC_PCLK 时钟分频器值 0: 禁用 RTC_PCLK。 1: 除以 1 到 255: 除以 255。	0
31:8	-	保留	0x00

4.5.19 CLKOUT 时钟源选择寄存器

该寄存器可将 clkout_clk 信号配置为 CLKOUT 引脚的输出。可以针对 clkout_clk 时钟选择所有三个振荡器和主时钟。

CLKOUTCLKUEN 寄存器（参见[章节 4.5.20](#)）必须从低电平转换到高电平，才能使更新生效。

表 26. CLKOUT 时钟源选择寄存器（CLKOUTCLKSEL、地址 0x4004 80E0）位描述

位	符号	值	描述	复位值
1:0	SEL		CLKOUT 时钟源	00
		0x0	IRC 振荡器	
		0x1	系统振荡器	
		0x2	看门狗振荡器	
		0x3	主时钟	
31:2	-	-	保留	0x00

4.5.20 CLKOUT 时钟源更新使能寄存器

写入 CLKOUTCLKSEL 寄存器后，该寄存器可以使用新时钟更新 CLKOUT 引脚的时钟源。必须先向 CLKOUTUEN 寄存器写入 0，再向 CLKOUTUEN 写入 1，才能使更新在 CLKOUT 引脚输入时生效。

表 27. CLKOUT 时钟源更新使能寄存器（CLKOUTUEN、地址 0x4004 80E4）位描述

位	符号	值	描述	复位值
0	ENA		启用 CLKOUT 时钟源更新	0
		0	无变化	
		1	更新时钟源	
31:1	-	-	保留	0x00

4.5.21 CLKOUT 时钟分频寄存器

该寄存器可确定 CLKOUT 引脚上 clkout_clk 信号的分频器值。

表 28. CLKOUT 时钟分频寄存器（CLKOUTDIV、地址 0x4004 80E8）位描述

位	符号	描述	复位值
7:0	DIV	时钟输出分频器值 0：禁用 CLKOUT。 1：除以 1 到 255：除以 255。	0
31:8	-	保留	0x00

4.5.22 POR 捕获 PIO 状态寄存器 0

PIOPORCAP0 寄存器用于在上电复位时捕获端口 0、1 和 2（引脚 PIO2_0 到 PIO2_7）上的 PIO 引脚状态（高电平或低电平）。每个位代表一个 GPIO 引脚的复位状态。该寄存器是只读状态寄存器。

表 29. POR 捕获 PIO 状态寄存器 0（PIOPORCAP0、地址 0x4004 8100）位描述

位	符号	描述	复位值
31:0	PIOSTAT	原始复位状态输入 PIO0_31 到 PIO2_7	依赖于用户执行

4.5.23 POR 捕获 PIO 状态寄存器 1

PIOPORCAP1 寄存器用于在上电复位时捕获端口 2（PIO2_8 到 PIO2_11）上的 PIO 引脚状态（高电平或低电平）。每个位代表一个 PIO 引脚的复位状态。该寄存器是只读状态寄存器。

表 30. POR 捕获 PIO 状态寄存器 1（PIOPORCAP1、地址 0x4004 8104）位描述

位	符号	描述	复位值
31:0	PIOSTAT	原始复位状态输入 PIO2_8 到 PIO2_11	依赖于用户执行

4.5.24 IOCONFIG 时钟分频寄存器 0 到 6

这些寄存器可分别将 7 个外设输入时钟配置到 IOCONFIG 可编程干扰滤波器。可以通过将 DIV 位设置为 0x0 关闭时钟。

表 31. IOCONFIG 滤波器时钟分频寄存器 0 到 6（IOCONFIGCLKDIV0 到 IOCONFIGCLKDIV6、地址 4004 8014C 到 4004 80134）位描述

位	符号	描述	复位值
7:0	DIV	IOCONFIG 滤波器时钟分频器值 0：禁用 IOCONFIGCLK。 1：除以 1 到 255：除以 255。	0
31:8	-	保留	0x00

4.5.25 BOD 控制寄存器

BOD 控制寄存器可选择 3 个不同阈值，用于向 NVIC 发送 BOD 中断和强制复位。[表 32](#) 中所列的复位和中断阈值都是典型值。

BODCTRL 寄存器的复位值不会受到系统复位的影响。相反，BODCTRL 寄存器将保留其最近一次的编程值。

BOD 控制电路只能通过 POR 复位。

表 32. BOD 控制寄存器（BODCTRL、地址 0x4004 8150）位描述

位	符号	值	描述	复位值
1:0	BODRSTLEV		BOD 复位电平	-
		0x0	保留。将 00 写入这些位不会更改 BOD 复位电平。	
		0x1	1 级：能引起复位的阈值电压为 2.038 V；能使复位无效的阈值电压为 2.180 V。	
		0x2	2 级：能引起复位的阈值电压为 2.336 V；能使复位无效的阈值电压为 2.471 V。	
		0x3	3 级：能引起复位的阈值电压为 2.624 V；能使复位无效的阈值电压为 2.761 V。	
3:2	BODINTVAL		BOD 中断电平	-
		0x0	保留。将 00 写入这些位不会更改中断电平。	
		0x1	1 级：能引起中断的阈值电压为 2.248 V；能使中断无效的阈值电压为 2.385 V。	
		0x2	2 级：能引起中断的阈值电压为 2.541 V；能使中断无效的阈值电压为 2.669 V。	
		0x3	3 级：能引起中断的阈值电压为 2.828 V；能使中断无效的阈值电压为 2.929 V。	
4	BODRSTENA		BOD 复位使能	-
		0	禁用复位功能。	
		1	启用复位功能。	
31:5	-	-	保留	-

4.5.26 系统节拍计数器校准寄存器

该寄存器可确定 SYST_CALIB 寄存器的值（参见[表 254](#)）。

表 33. 系统节拍定时器校准寄存器（SYSTCKCAL、地址 0x4004 8154）位描述

位	符号	描述	复位值
25:0	CAL	系统节拍定时器校准值	0x00
31:26	-	保留	0x00

4.5.27 AHB 矩阵主优先级寄存器

AHB 矩阵主优先级寄存器可决定 AHB 主线访问总线的优先级。

表 34. AHB 矩阵主优先级寄存器（AHBPRI0、地址 0x4004 8158）位描述

位	符号	描述	复位值
1:0	M0PRIO	ARM Cortex-M0 内核的优先级	00
3:2	DMAPRIO	微型 DMA 控制器的优先级	01
5:4	-	保留	-
31:6	-	保留	-

4.5.28 中断请求 (IRQ) 延迟寄存器

IRQLATENCY 寄存器是一个 8 位寄存器，用于指定系统响应中断请求的最小允许周期数 (0-255)。该寄存器的意图在于使用户可以在中断响应时间与确定性之间选择一个平衡点。

将此参数设置为一个非常小的值（例如 0），可以保证最佳中断性能，但同时也会带来非常明显的不确定性和抖动。要求系统始终使用较大的周期数（无论是否需要）将降低不确定量，但不一定会消除。

从理论上来讲，ARM Cortex-M0 内核应始终能够在 15 个周期内处理中断请求。但是，在处理中断之前，CPU 外部系统因素、总线延迟、外设响应时间等会增加完成上一个指令所需的时间。因此，精确地指定可保证确定性的最小周期数将取决于应用程序。

该寄存器的默认设置为 0x010。

表 35. IRQ 延迟寄存器（IRQLATENCY、地址 0x4004 8170）位描述

位	符号	描述	复位值
7:0	LATENCY	8 位延迟值	0x010
31:8	-	保留	-

4.5.29 NMI 中断源配置寄存器

该寄存器可以为 ARM Cortex-M0 非屏蔽中断 (NMI) 配置源。参见[章节 25.5.2](#)。

表 36. NMI 中断源配置寄存器 (INTNMI、地址 0x4004 8174) 位描述

位	符号	描述	复位值
5:0	NMISRC	NMI 中断源选择。保留值 0x13 到 0x3E。 0x0 = I2C 中断。 0x1 = CT16B0 中断。 0x2 = CT16B1 中断。 0x3 = CT32B0 中断。 0x4 = CT32B1 中断。 0x5 = SSP 中断。 0x6 = UART0 中断。 0x7 = UART1 中断。 0x8 = 比较器中断。 0x9 = ADC 中断。 0xA = 看门狗定时器中断。 0xB = I2C 中断。 0xC = 保留。 0xD = PIO0 中断。 0xE = PIO1 中断。 0xF = PIO2 中断。 0x10 = PMU 唤醒中断。 0x11 = DMA 中断。 0x12 = RTC 中断。 0x3F = NMI 禁用。	-
31:6	-	保留	0x00

4.5.30 启动逻辑边沿控制寄存器 0

STARTAPRP0 寄存器用于控制端口 0 (PIO0_0 至 PIO0_11) 的启动逻辑输入。该寄存器为启动逻辑选择相应 PIO 输入的下降沿或上升沿来分别产生下降或上升时钟沿 (参见[章节 4.8.2](#))。

STARTAPRP0 寄存器的每一位控制一个端口输入，并连接到 NVIC 中的一个唤醒中断。STARTAPRP0 寄存器中的位 0 对应中断 0，位 1 对应中断 1，...，最多有 12 个中断。通过此启动逻辑寄存器设置，外部引脚可用于从深度睡眠模式唤醒微控制器。

注：如果相应 PIO 引脚用于从深度睡眠模式唤醒微控制器，则必须在 NVIC 中启用连接到启动逻辑输入的每个中断。

表 37. 启动逻辑边沿控制寄存器 0 (STARTAPRP0、地址 0x4004 8200) 位描述

位	符号	描述	复位值
0	APRPIO0_0	引脚 PIO0_0 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	0
1	APRPIO0_1	引脚 PIO0_1 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
2	APRPIO0_2	引脚 PIO0_2 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	

表 37. 启动逻辑边沿控制寄存器 0（STARTAPRP0、地址 0x4004 8200）位描述 *（续）*

位	符号	描述	复位值
3	APRPIO0_3	引脚 PIO0_3 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
4	APRPIO0_4	引脚 PIO0_4 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
5	APRPIO0_5	引脚 PIO0_5 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
6	APRPIO0_6	引脚 PIO0_6 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
7	APRPIO0_7	引脚 PIO0_7 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
8	APRPIO0_8	引脚 PIO0_8 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
9	APRPIO0_9	引脚 PIO0_9 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
10	APRPIO0_10	引脚 PIO0_10 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
11	APRPIO0_11	引脚 PIO0_11 启动逻辑输入 n 的边沿选择。 0 = 下降沿 1 = 上升沿	
31:12	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.31 启动逻辑信号使能寄存器 0

STARTERP0 寄存器可以启用或禁用启动逻辑中的启动信号位。位赋值与[表 37](#) 相同。

表 38. 启动逻辑信号使能寄存器 0（STARTERP0、地址 0x4004 8204）位描述

位	符号	描述	复位值
0	ERPIO0_0	启用引脚 PIO0_0 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
1	ERPIO0_1	启用引脚 PIO0_1 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
2	ERPIO0_2	启用引脚 PIO0_2 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
3	ERPIO0_3	启用引脚 PIO0_3 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
4	ERPIO0_4	启用引脚 PIO0_4 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0

表 38. 启动逻辑信号使能寄存器 0 (STARTERP0、地址 0x4004 8204) 位描述 (续)

位	符号	描述	复位值
5	ERPIO0_5	启用引脚 PIO0_5 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
6	ERPIO0_6	启用引脚 PIO0_6 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
7	ERPIO0_7	启用引脚 PIO0_7 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
8	ERPIO0_8	启用引脚 PIO0_8 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
9	ERPIO0_9	启用引脚 PIO0_9 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
10	ERPIO0_10	启用引脚 PIO0_10 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
11	ERPIO0_11	启用引脚 PIO0_11 启动逻辑输入 n 的启动信号。 0 = 禁用。 1 = 启用。	0
31:12	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.32 启动逻辑复位寄存器 0

在 STARTRSRP0CLR 寄存器中，对一个位写入 1 可复位启动逻辑状态。位赋值与表 37 相同。为了记录启动信号，启动逻辑会利用输入信号产生一个时钟边沿来实现。时钟边沿（下降沿或上升沿）可以设置从深度睡眠模式唤醒的中断。所以，启动逻辑的状态必须在它使用之前清除。

表 39. 启动逻辑复位寄存器 0 (STARTRSRP0CLR、地址 0x4004 8208) 位描述

位	符号	描述	复位值
0	RSRPIO0_0	引脚 PIO0_0 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
1	RSRPIO0_1	引脚 PIO0_1 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
2	RSRPIO0_2	引脚 PIO0_2 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
3	RSRPIO0_3	引脚 PIO0_3 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
4	RSRPIO0_4	引脚 PIO0_4 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
5	RSRPIO0_5	引脚 PIO0_5 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0

表 39. 启动逻辑复位寄存器 0 (STARTSRP0CLR、地址 0x4004 8208) 位描述 (续)

位	符号	描述	复位值
6	RSRPIO0_6	引脚 PIO0_6 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
7	RSRPIO0_7	引脚 PIO0_7 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
8	RSRPIO0_8	引脚 PIO0_8 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
9	RSRPIO0_9	引脚 PIO0_9 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
10	RSRPIO0_10	引脚 PIO0_10 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
11	RSRPIO0_11	引脚 PIO0_11 启动逻辑输入 n 的启动信号复位。 0 = 无效。 1 = 写：复位启动信号。	0
31:12	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.33 启动逻辑状态寄存器 0

该寄存器反映了已启用启动信号位的状态。位赋值与[表 37](#) 相同。每一位（如果启用）均能反映启动逻辑的状态，即给定引脚是否已接收到唤醒信号。

表 40. 启动逻辑状态寄存器 0 (STARTSRP0、地址 0x4004 820C) 位描述

位	符号	描述	复位值
0	SRPIO0_0	引脚 PIO0_0 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
1	SRPIO0_1	引脚 PIO0_1 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
2	SRPIO0_2	引脚 PIO0_2 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
3	SRPIO0_3	引脚 PIO0_3 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
4	SRPIO0_4	引脚 PIO0_4 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
5	SRPIO0_5	引脚 PIO0_5 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
6	SRPIO0_6	引脚 PIO0_6 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0

表 40. 启动逻辑状态寄存器 0 (STARTSRP0、地址 0x4004 820C) 位描述 (续)

位	符号	描述	复位值
7	SRPIO0_7	引脚 PIO0_7 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
8	SRPIO0_8	引脚 PIO0_8 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
9	SRPIO0_9	引脚 PIO0_9 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
10	SRPIO0_10	引脚 PIO0_10 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
11	SRPIO0_11	引脚 PIO0_11 启动逻辑输入 n 的启动信号状态。 0 = 未接收到启动信号。 1 = 启动信号挂起。	0
31:12	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.34 启动逻辑边沿控制寄存器 1

STARTAPRP1 寄存器用于控制连接到外设中断的启动逻辑输入。该寄存器为启动逻辑选择相应中断的下降沿或上升沿来分别产生下降或上升时钟沿。通过此启动逻辑寄存器设置，外设中断可用于从深度睡眠模式唤醒微控制器。

STARTAPRP1 寄存器的每一位控制一个外设中断（参见表 41），并对应于 NVIC 中的中断，参见表 4。

表 41. 外设中断的启动逻辑位连接

启动逻辑位	中断	描述
9:0	-	保留
10	WDT	看门狗中断 (WDINT)
11	BOD	掉电检测
17:12	-	保留
18	RTC	RTC 中断
31:19	-	保留

表 42. 启动逻辑边沿控制寄存器 1 (STARTAPRP1、地址 0x4004 8210) 位描述

位	符号	描述	复位值
9:0	-	保留。切勿对保留的寄存器位写入 1。	-
10	APRWDT	启动逻辑中断 WDT 的边沿选择。 0 = 下降沿。 1 = 上升沿。	0
11	APRBOD	启动逻辑中断 BOD 的边沿选择。 0 = 下降沿。 1 = 上升沿。	0

表 42. 启动逻辑边沿控制寄存器 1 (STARTAPRP1、地址 0x4004 8210) 位描述 (续)

位	符号	描述	复位值
17:12	-	保留。切勿对保留的寄存器位写入 1。	0
18	APRRTC	启动逻辑中断 RTC 的边沿选择。 0 = 下降沿。 1 = 上升沿。	0
31:19	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.35 启动逻辑信号使能寄存器 1

STARTERP1 寄存器可以启用或禁用启动逻辑中的启动信号位。位赋值与表 42 相同。

表 43. 启动逻辑信号使能寄存器 1 (STARTERP1、地址 0x4004 8214) 位描述

位	符号	描述	复位值
9:0	-	保留。切勿对保留的寄存器位写入 1。	-
10	ERWDT	启用启动逻辑中断 WDT 的启动信号。 0 = 已禁用。 1 = 已启用。	0
11	ERBOD	启用启动逻辑中断 BOD 的启动信号。 0 = 已禁用。 1 = 已启用。	0
17:12	-	保留。切勿对保留的寄存器位写入 1。	0
18	ERRTC	启用启动逻辑中断 RTC 的启动信号。 0 = 已禁用。 1 = 已启用。	0
31:19	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.36 启动逻辑复位寄存器 1

在 STARTRSRP1CLR 寄存器中，对一个位写入 1 可复位启动逻辑状态。位赋值与表 42 相同。为了记录启动信号，启动逻辑会利用输入信号产生一个时钟边沿来实现。时钟边沿（下降沿或上升沿）可以设置从深度睡眠模式唤醒的中断。所以，启动逻辑的状态必须在它使用之前清除。

表 44. 启动逻辑复位寄存器 1 (STARTRSRP1CLR、地址 0x4004 8218) 位描述

位	符号	描述	复位值
9:0	-	保留。切勿对保留的寄存器位写入 1。	-
10	RSRWDT	复位启动逻辑中断 WDT 的启动信号。 0 = 无效。 1 = 写：复位启动信号。	0
11	RSRBOD	复位启动逻辑中断 BOD 的启动信号。 0 = 无效。 1 = 写：复位启动信号。	0
17:12	-	保留。切勿对保留的寄存器位写入 1。	0
18	RSRRTC	复位启动逻辑中断 RTC 的启动信号。 0 = 无效。 1 = 写：复位启动信号。	0
31:19	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.37 启动逻辑状态寄存器 1

该寄存器反映了已启用启动信号的状态。位赋值与[表 42](#) 相同。

表 45. 启动逻辑信号状态寄存器 1（STARTSRP1、地址 0x4004 821C）位描述

位	符号	描述	复位值
9:0	-	保留。切勿对保留的寄存器位写入 1。	-
10	SRWDT	启动逻辑中断 WDT 的启动信号状态。 0 = 未收到启动信号。 1 = 启动信号挂起。	0
11	SRBOD	启动逻辑中断 BOD 的启动信号状态。 0 = 未收到启动信号。 1 = 启动信号挂起。	0
17:12	-	保留。切勿对保留的寄存器位写入 1。	0
18	SRRTC	启动逻辑中断 RTC 的启动信号状态。 0 = 未收到启动信号。 1 = 启动信号挂起。	0
31:19	-	保留。切勿对保留的寄存器位写入 1。	-

4.5.38 深度睡眠模式配置寄存器

该寄存器用于控制看门狗 (WD) 振荡器和 BOD 电路在器件进入深度睡眠模式后的行为。此外，WD 振荡器的行为受到 WDMODE 寄存器中 WDLOCKCLK 位的影响（[表 266](#)）。所有其他模拟模块都在深度睡眠模式下关闭。

在进入深度睡眠模式之前，并在使用[表 46](#) 中所示的 4 个值之一设置 WDT 锁定位 WDLOCKCLK 之前，必须写入该寄存器：

表 46. 当 WDLOCKCLK = 0（WD 振荡器未锁定）时，PDSLEEPCFG 寄存器的允许值

配置	WD 振荡器打开	WD 振荡器关闭
BOD 打开	PDSLEEPCFG = 0x0000 FFB7	PDSLEEPCFG = 0x0000 FFF7
BOD 关闭	PDSLEEPCFG = 0x0000 FFBF	PDSLEEPCFG = 0x0000 FFFF

表 47. 当 WDLOCKCLK = 1（WD 振荡器锁定）时，PDSLEEPCFG 寄存器的允许值

配置	WD 振荡器打开	WD 振荡器关闭
BOD 打开	PDSLEEPCFG = 0x0000 FFB7	不允许
BOD 关闭	PDSLEEPCFG = 0x0000 FFBF	不允许

注：如果未能正确地对该寄存器进行初始化和编程，则可能导致微控制器产生未定义的行为。[表 46](#) 和[表 47](#) 中所列的值是 PDSLEEPCFG 寄存器唯一允许的值。

要为深度睡眠模式选择合适的电源配置，请考虑以下内容：

- BOD：当零部件进入深度睡眠模式时，将 BOD 电路保持在启用状态可避免零部件发生低压事件。但是，BOD 电路在深度睡眠模式下会导致额外的电流损耗。
- WD 振荡器：看门狗振荡器可在深度睡眠模式下继续运行，以根据需要为看门狗定时器提供时钟，进而为唤醒事件定时（有关详细信息，请参阅[章节 4.8.3](#)）。在这种情况下，看门狗振荡器模拟输出频率必须设置为其最低的值（WDTOSCCTRL 的位 FREQSEL = 0001，参见[表 13](#)），而且在进入深度睡眠模式之前，必须禁用 SYSAHBCLKCTRL 寄存器中除定时器时钟之外的所有外设时钟（参见[表 21](#)）。

请注意，必须先运行 **WD 振荡器**，然后才能在 **WDMODE** 寄存器中设置 **WDLOCKCLK** 位。

如果正在运行，看门狗振荡器在深度睡眠模式下会导致额外的电流损耗。

注：必须始终按照说明写入该寄存器的保留位。在进入深度睡眠模式之前，该寄存器必须正确初始化。

注：该寄存器中的设置会受到 **WDT** 锁定状态的影响：如果选择看门狗振荡器作为 **WDT** 的时钟源，则如果同时在 **WDMOD** 寄存器中设置位 **5**，将会忽略对 **PDSLEEPCFG** 寄存器中位 **6** 的写入操作（参见表 266）。

表 48. 深度睡眠配置寄存器（PDSLEEPCFG、地址 0x4004 8230）位描述

位	符号	值	描述	复位值
2:0	-		保留。始终将这些位写为 1。	111
3	BOD_PD		深度睡眠模式下的 BOD 掉电控制	1
		0	上电	
		1	掉电	
5:4	-		保留。始终将这些位写为 1。	11
6	WDTOSC_PD		深度睡眠模式下的看门狗振荡器掉电控制	1
		0	上电	
		1	掉电。必须先更改为 0，再在 WDMODE 寄存器中设置 WDLOCKCLK 位。参见表 266。	
15:7	-		保留。始终将这些位写为 1。	1 1111 1111
31:16	-	-	保留	0

4.5.39 唤醒配置寄存器

当从深度睡眠模式中唤醒时，可对该寄存器中的位进行编程来表示微控制器将要进入的状态。

注：该寄存器中的设置会受到 **WDT** 锁定状态的影响：

- 如果选择看门狗振荡器作为 **WDT** 的时钟源，则如果同时在 **WDMOD** 寄存器中设置位 **5**，将会忽略对 **PDAWAKECFG** 寄存器中位 **6** 的写入操作（参见表 266）。
- 如果选择 **IRC** 作为 **WDT** 的时钟源，则如果同时在 **WDMOD** 寄存器中设置位 **5**，将会忽略对 **PDAWAKECFG** 寄存器中位 **0** 和位 **1** 的写入操作（参见表 266）。

表 49. 唤醒配置寄存器（PDAWAKECFG、地址 0x4004 8234）位描述

位	符号	值	描述	复位值
0	IRCOUT_PD		IRC 振荡器输出唤醒配置	0
		0	上电	
		1	掉电	
1	IRC_PD		IRC 振荡器掉电唤醒配置	0
		0	上电	
		1	掉电	

表 49. 唤醒配置寄存器 (PDAWAKECFG、地址 0x4004 8234) 位描述 (续)

位	符号	值	描述	复位值
2	FLASH_PD		闪存唤醒配置	0
		0	上电	
		1	掉电	
3	BOD_PD		BOD 唤醒配置	0
		0	上电	
		1	掉电	
4	ADC_PD		ADC 唤醒配置	1
		0	上电	
		1	掉电	
5	SYSOSC_PD		系统振荡器唤醒配置	1
		0	上电	
		1	掉电	
6	WDTOSC_PD		看门狗振荡器唤醒配置	1
		0	上电	
		1	掉电	
7	SYSPLL_PD		系统 PLL 唤醒配置	1
		0	上电	
		1	掉电	
8	-	-	保留	-
9	-	-	保留。	0
10	-	-	保留	-
11	-	-	保留。	1
14:12	-	-	保留	-
15	COMP_PD		比较器掉电唤醒配置	1
		0	上电	
		1	掉电	
31:16	-	-	保留	0

4.5.40 掉电配置寄存器

PDRUNCFG 寄存器中的位控制各种模拟模块的电源。微控制器运行时的任何时刻都可以写入该寄存器，而且该写入操作将会立即生效，IRC 的掉电信号除外。

为了避免在 IRC 掉电时产生干扰，IRC 时钟在无干扰时会自动关闭。因此，对于 IRC 来说，在掉电状态生效之前可能会延时。

注：该寄存器中的设置会受到 WDT 锁定状态的影响：

- 如果选择看门狗振荡器作为 WDT 的时钟源，则如果同时在 MOD 寄存器中设置位 5，将会忽略对 PDRUNCFG 寄存器中位 6 的写入操作（参见表 266）。
- 如果选择 IRC 作为 WDT 的时钟源，则如果同时在 MOD 寄存器中设置位 5，将会忽略对 PDRUNCFG 寄存器中位 0 和位 1 的写入操作（参见表 266）。

表 50. 掉电配置寄存器（PDRUNCFG、地址 0x4004 8238）位描述

位	符号	值	描述	复位值
0	IRCOUT_PD		IRC 振荡器输出掉电	0
		0	上电	
		1	掉电	
1	IRC_PD		IRC 振荡器掉电	0
		0	上电	
		1	掉电	
2	FLASH_PD		闪存掉电	0
		0	上电	
		1	掉电	
3	BOD_PD		BOD 掉电	0
		0	上电	
		1	掉电	
4	ADC_PD		ADC 掉电	1
		0	上电	
		1	掉电	
5	SYSOSC_PD		系统振荡器掉电	1
		0	上电	
		1	掉电	
6	WDTOSC_PD		看门狗振荡器掉电	1
		0	上电	
		1	掉电	
7	SYSPLL_PD		系统 PLL 掉电	1
		0	上电	
		1	掉电	
8	-	-	保留	-
9	-	-	保留	0
10	-	-	保留	-
11	-	-	保留	1
14:12	-	-	保留	-
15	COMP_PD		比较器掉电	1
		0	上电	
		1	掉电	
31:16	-	-	保留	0

4.5.41 器件 ID 寄存器

器件 ID 寄存器为只读寄存器，包含每个 LPC122x 部件的器件 ID。该寄存器也可以通过 ISP/IAP 命令读取（参见[章节 20.7.11](#)和[章节 20.8.5](#)）。

表 51. 器件 ID 寄存器 (DEVICE_ID、地址 0x4004 83F4) 位描述

位	符号	描述	复位值
31:0	DEVICEID	LPC122x 部件的器件 ID: LPC12D27FBD100/301 = 0x3670 002B LPC1227FBD64/301 = 0x3670 002B LPC1227FBD48/301 = 0x3670 002B LPC1226FBD64/301 = 0x3660 002B LPC1226FBD48/301 = 0x3660 002B LPC1225FBD64/321 = 0x3652 002B LPC1225FBD64/301 = 0x3650 002B LPC1225FBD48/321 = 0x3652 002B LPC1225FBD48/301 = 0x3650 002B LPC1224FBD64/121 = 0x3642 C02B LPC1224FBD64/101 = 0x3640 C02B LPC1224FBD48/121 = 0x3642 C02B LPC1224FBD48/101 = 0x3640 C02B	取决于零部件

4.5.42 闪存配置寄存器

根据系统时钟频率，通过在地址 0x5006 0028 写入 FLASHCFG 寄存器，为使用闪存存储器配置不同的访问时间。

注：只有将 PRESET_CTRL 寄存器（参见表 9）中的 FLASH_OVERRIDE 位设置为允许多次读取访问的 0 时，该寄存器中的设置才有效。

表 52. 闪存配置寄存器 (FLASHCFG、地址 0x5006 0028) 位描述

位	符号	值	描述	复位值
1:0	RDCFG		闪存存储器读取访问时间：	0
		0x0	2 个系统时钟周期的闪存访问时间	
		0x1	3 个系统时钟周期的闪存访问时间	
		0x2	4 个系统时钟周期的闪存访问时间	
		0x3	5 个系统时钟周期的闪存访问时间	
31:2	-		保留。	-

4.6 复位

LPC122x 有 4 个复位源：RESET 引脚、看门狗复位、上电复位 (POR) 和掉电检测 (BOD)。此外，还有一个软件复位。

RESET 引脚为施密特触发输入引脚。复位可以由任意一个复位源引起，只要工作电压达到可用电平，就会启动 IRC（可引起复位）来保持有效，直到外部复位无效为止，振荡器运行，同时闪存控制器完成初始化。

当 Cortex-M0 CPU 外部复位源（POR、BOD 复位、外部复位和看门狗复位）有效时，将发起以下进程：

- 1. IRC 启动。IRC 启动（上电时最多 6ms）以后，IRC 就可以提供稳定的时钟输出。
- 2. ROM 中的引导代码启动。引导代码可以执行引导任务，也可以跳转到闪存。
- 3. 闪存上电。闪存上电大约需要 <tbd>ms。然后，闪存初始化序列启动，这大约需要 <tbd>个周期。

当移除内部复位时，处理器将在地址 0 处开始执行，该地址最初是从引导模块映射的复位向量。这时，所有处理器和外设寄存器都已经初始化为预定值。

4.7 电源管理

LPC122x 支持多种电源控制功能。在工作模式下，当微控制器运行时，可以对所选外设的电源和时钟进行优化，从而降低功耗。此外，处理器有三种特殊的功耗降低模式：睡眠模式、深度睡眠模式和深度掉电模式。

注：在睡眠、深度睡眠或深度掉电模式下，不支持调试模式。

4.7.1 工作模式

在工作模式下，ARM Cortex-M0 内核和存储器由系统时钟计时，而外设由系统时钟或专用外设时钟计时。

微控制器复位后处于工作模式，而且默认电源配置由 PDRUNCFG 和 SYSAHBCLKCTRL 寄存器的复位值决定。在运行时可以更改电源配置。

4.7.1.1 工作模式下的电源配置

工作模式下的功耗由以下配置选项决定：

- SYSAHBCLKCTRL 寄存器控制所运行的存储器和外设（[表 21](#)）。
- 各模拟模块（PLL、振荡器、ADC、BOD 电路和闪存模块）的电源可以通过 PDRUNCFG 寄存器随时单独控制（[表 50](#)）。
- 系统时钟的时钟源可以从 IRC（默认）、系统振荡器或看门狗振荡器中选择（参见[图 3](#)和相关寄存器）。
- 系统时钟的频率可通过 SYSPLLCTRL（[表 10](#)）和 SYSAHBCLKDIV 寄存器（[表 21](#)）来选择。
- 选定的外设（UART、SSP0/1、WDT）使用单独的外设时钟及其自己的时钟分频器。外设时钟可以通过相应的时钟分频寄存器（[表 22](#) 到 [表 28](#)）来关闭。

4.7.2 睡眠模式

在睡眠模式下，ARM Cortex-M0 内核的系统时钟停止，且在复位或启用的中断出现之前都不能执行指令。

在睡眠模式下，外设功能（如果选择在 SYSAHBCLKCTRL 寄存器中计时）继续运行，并可能产生中断使处理器继续运行。睡眠模式消除了处理器自身、存储器系统及其相关控制器和内部总线的动态功耗。处理器的状态和寄存器、外设寄存器和内部 SRAM 的值都会保留，引脚的逻辑电平保持静态。

4.7.2.1 睡眠模式下的电源配置

睡眠模式下的功耗根据工作模式下相同的设置进行配置：

- 时钟继续运行。
- 系统时钟的频率与工作模式下的频率保持一致，但处理器未定时。
- 模拟和数字外设与工作模式下选择的一样。

4.7.2.2 对睡眠模式进行编程

进入睡眠模式的步骤如下:

1. PCON 寄存器中的 DPDEN 位必须设置为 0 ([表 56](#))。
2. ARM Cortex-M0 SCR 寄存器中的 SLEEPDEEP 位必须设置为 0, 参见 ([表 388](#))。
3. 使用 ARM Cortex-M0 等待中断 (WFI) 指令。

4.7.2.3 从睡眠模式唤醒

在 NVIC 启用的中断到达处理器或发生复位时, 系统将自动退出睡眠模式。因中断而唤醒之后, 微控制器将返回由 PDRUNCFG 和 SYSAHBCLKDIV 寄存器的内容定义的原始电源配置。如果发生复位, 微控制器会进入工作模式下的默认配置。

4.7.3 深度睡眠模式

在深度睡眠模式下, 处理器的系统时钟如睡眠模式中一样被禁用。在 PDSLEEPCFG 寄存器处于深度睡眠模式时, 除了 BOD 电路和看门狗振荡器之外, 所有模拟模块掉电, 必须选择或取消选择。RTC 和 RTC 振荡器可以在深度睡眠模式下运行, 除非 RTC 掉电。

深度睡眠模式消除了闪存、模拟外设使用的所有功耗以及处理器自身、存储器系统及其相关控制器和内部总线所使用的所有动态功耗。处理器的状态和寄存器、外设寄存器和内部 SRAM 的值都会保留, 引脚的逻辑电平保持静态。

4.7.3.1 深度睡眠模式下的电源配置

深度睡眠模式下的功耗由 PDSLEEPCFG ([表 48](#)) 寄存器中的深度睡眠电源配置的设置决定:

- 除了 RTC 振荡器之外, 看门狗振荡器是深度睡眠模式下唯一可用的时钟源。如果外设控制唤醒 (参见[章节 4.8.3](#)) 需要, 看门狗振荡器可在深度睡眠模式下继续运行。所有其他时钟源 (IRC 和系统振荡器) 和系统 PLL 都已关闭。看门狗振荡器模拟输出频率必须设置为其模拟时钟输出的最低值 (WDTOSCCTRL 的位 FREQSEL = 0001, 参见[表 13](#))。
- 如果应用程序需要, BOD 电路可在深度睡眠模式下继续运行。
- 如果看门狗振荡器在深度睡眠模式下运行, 则只应在 SYSAHBCLKCTRL 寄存器中启用看门狗定时器, 以最大限度降低功耗。
- RTC 和 RTC 振荡器可在深度睡眠模式下继续运行。

4.7.3.2 对深度睡眠模式进行编程

进入深度睡眠模式的步骤如下:

1. PCON 寄存器中的 DPDEN 位必须设置为 0 ([表 56](#))。
2. 选择 PDSLEEPCFG ([表 48](#)) 寄存器深度睡眠模式下的电源配置
 - a. 对于外设控制唤醒, 应确保看门狗振荡器在 PDRUNCFG 寄存器中上电, 并将时钟源切换到 MAINCLKSEL 寄存器 ([表 18](#)) 中的 WD 振荡器。
 - b. 若无外设控制唤醒, 且如果看门狗振荡器关闭, 则应确保 IRC 在 PDRUNCFG 寄存器中上电, 并将时钟源切换到 MAINCLKSEL 寄存器 ([表 18](#)) 中的 IRC。这样可以确保系统时钟无干扰关闭。
3. 在 PDAWAKECFG ([表 49](#)) 寄存器中唤醒之后选择电源配置。

4. 配置启动逻辑：
 - 如果外部引脚用于唤醒，则启用并清除启动逻辑 0 寄存器（[表 37](#) 到 [表 40](#)）中的唤醒引脚，并启用 NVIC 中的启动逻辑中断。
 - 如果使用 RTC，则启用并清除启动逻辑 1 寄存器（[表 42](#) 到 [表 45](#)）中的位 18，并启用 NVIC 中的 RTC 中断。
5. 在 SYSAHBCLKCTRL 寄存器（[表 21](#)）中，根据需要禁用所有外设，RTC 或 WDT 除外。
6. 向 ARM Cortex-M0 SCR 寄存器（[表 388](#)）中的 SLEEPDEEP 位写入 1。
7. 使用 ARM WFI 指令。

4.7.3.3 从深度睡眠模式唤醒

微控制器从深度睡眠模式唤醒的方式有以下几种：

- 外部引脚的信号。为此，引脚 PIO0_0 到 PIO0_11 可作为启动逻辑的输入启用。启动逻辑不需要任何时钟，并在 NVIC 中启用时生成中断，以从深度睡眠模式唤醒。
- RTC 匹配中断，用于自定时唤醒。必须在启动逻辑 1 模块中启用 RTC 中断。
- 从 BOD 电路复位。在这种情况下，必须在 PDSLEEPCFG 寄存器中启用 BOD 电路，而且必须在 BODCTRL 寄存器（[表 32](#)）中启用 BOD 复位。
- 从看门狗定时器复位。在这种情况下，看门狗振荡器必须在深度睡眠模式下运行（参见 PDSLEEPCFG 寄存器），而且必须在 SYSAHBCLKCTRL 寄存器中启用 WDT。
- 外部 $\overline{\text{RESET}}$ 引脚。

注：如果看门狗振荡器在深度睡眠模式下运行，则其频率决定唤醒时间，导致唤醒时间比 12 MHz IRC 的唤醒时间要长。

4.7.4 深度掉电模式

在深度掉电模式下，除了 WAKEUP 引脚之外，整个微控制器的电源和时钟都被关闭。当 WDMODE 寄存器中的 WDLOCKDP 位设置为 1 时，可以阻止微控制器进入深度掉电模式。（[表 266](#)）

如果在进入深度掉电模式之前启用 RTC，则 RTC 和 RTC 振荡器将继续在深度掉电模式下运行。如果在深度掉电模式下不需要 RTC，则禁用 RTC，以最大限度降低功耗。

在深度掉电模式下，除了少量数据可以存储在 PMU 模块的 4 个 32 位通用寄存器中之外，SRAM 和寄存器的内容将不会被保留。

在深度掉电模式下，除了 WAKEUP 引脚之外，所有功能引脚都是三态的。

4.7.4.1 深度掉电模式下的电源配置

深度掉电模式没有配置选项。除了 RTC 和 RTC 振荡器之外，所有时钟、内核和所有外设都已掉电。只有 WAKEUP 引脚和备份寄存器上电。低功耗 RTC 和 RTC 振荡器可以继续运行（这是默认设置）。

4.7.4.2 对深度掉电模式进行编程

注：如果 WDMODE 寄存器（[表 266](#)）中的 WDLOCKDP 位设置为 0，则微控制器只能进入深度掉电模式。如果 WDLOCKDP = 1，则微控制器必须在进入深度掉电模式之前复位。

进入深度掉电模式的步骤如下：

1. 向 PCON 寄存器（参见[表 56](#)）中的 DPDEN 位写入 1。
2. 存储将要保留在通用寄存器（[表 57](#)）中的数据。
3. 向 ARM Cortex-M0 SCR 寄存器（[表 388](#)）中的 SLEEPDEEP 位写入 1。
4. 在进入深度掉电模式之前，通过将 PDRUNCFG 寄存器中的位 IRCOUT_PD 和 IRC_PD 设置为 0，确保 IRC 上电。
5. 使用 ARM WFI 指令。

注：在进入深度掉电模式之前，必须从外部将 WAKEUP 引脚上拉到高电平。

4.7.4.3 从深度掉电模式唤醒

从深度掉电模式唤醒会引起微控制器复位（参见[章节 4.6](#)）。但是，RTC 寄存器和备份寄存器的内容得以保留。LPC122x 从深度掉电模式唤醒有以下两种方式（参见[章节 4.9](#)）：

- 将 WAKEUP 引脚下拉到低电平，从深度掉电唤醒 LPC122x。WAKEUP 引脚电平从高到低转换的最小脉冲宽度为 50 ns。
- RTC 中断（如果在 RTCIMSC 寄存器中没有被屏蔽，请参见[表 261](#)）从深度掉电模式唤醒 LPC122x。

注：在深度掉电模式下， $\overline{\text{RESET}}$ 引脚没有任何功能。

4.8 深度睡眠模式详细信息

4.8.1 IRC 振荡器

IRC 是 LPC122x 中唯一可以无干扰关闭的振荡器。因此，建议用户在微控制器进入深度睡眠模式之前将时钟源切换到 IRC。

4.8.2 使用外部引脚从深度睡眠模式唤醒（启动逻辑 0）

在启动逻辑表示 ARM 内核的中断时退出深度睡眠模式。端口引脚 PIO0_0 到 PIO0_11 连接至启动逻辑并作为唤醒引脚。用户必须对每个输入的启动逻辑 0 寄存器进行编程，以便为相应的唤醒事件设置合适的边沿极性。另外，必须在 NVIC 中启用对应于每个输入的中断。NVIC 中的中断 0 到 11 对应于 12 个 PIO 引脚（参见[章节 4.5.30](#)）。

启动逻辑不需要时钟运行，因为它使用已启用引脚的输入信号生成时钟沿。因此，在使用前应清除启动逻辑信号（参见[表 39](#)）。

启动逻辑也可以用于工作模式，使用 LPC122x 的输入引脚提供向量中断。

4.8.3 使用 RTC 从深度睡眠模式唤醒（启动逻辑 1）

RTC 由独立的 RTC 振荡器定时，并继续在深度睡眠模式下运行。RTC 中断在内部连接至启动逻辑 1 模块中的位 18，并作为唤醒中断。用户必须对 RTC 中断的启动逻辑 1 寄存器进行编程，以启用 RTC 唤醒，并选择上升沿配置。另外，必须在 NVIC 中启用 RTC 中断。

4.9 深度掉电模式详细信息

4.9.1 使用 WAKEUP 引脚从深度掉电模式唤醒

将 WAKEUP 引脚下拉到低电平，会从深度掉电唤醒 LPC122x，并且微控制器将完成整个复位进程（[章节 4.6](#)）。WAKEUP 引脚电平从高到低转换的最小脉冲宽度为 50 ns。

使用 WAKEUP 引脚从深度掉电模式唤醒微控制器的步骤如下：

1. 在零部件处于深度掉电模式时，通过在 WAKEUP 引脚上以至少 50 ns 的脉冲长度从外部进行从高到低的电平转换，生成唤醒信号。
 - PMU 将会开启片内调压器。当内核电压达到上电复位 (POR) 的跳变点时，系统就会复位，微控制器也将重新启动。
 - 除了 GPREG0 到 GPREG3 之外，所有寄存器都将处于复位状态。
 - 如果启用 RTC，则 RTC 寄存器的内容将会被保留。
2. 启动微控制器之后，可以读取 PCON 寄存器（[表 57](#)）中的深度掉电标志，以证明复位是由深度掉电模式的唤醒事件引起的。
3. 清除 PCON 寄存器（[表 55](#)）中的深度掉电标志。
4. （可选）读取通用寄存器（[表 57](#) 和 [表 58](#)）中储存的数据。
5. 为下一个深度掉电循环设置 PMU。

4.9.2 使用 RTC 从深度掉电模式唤醒

RTC 中断从深度掉电模式唤醒微控制器的方式与 WAKEUP 引脚采用的方式相同。在对时钟数进行编程之后，创建 RTC 中断时产生唤醒信号。

若要使用 RTC 唤醒中断，必须根据以下步骤对 RTC 进行配置：

- 使用 RTC 定时器匹配值（[表 258](#)）对 RTC 匹配寄存器进行编程。
- 清除 RTC 中断屏蔽寄存器（[表 261](#)）中的中断屏蔽。
- RTC 的时钟源必须是 RTC 振荡器（[表 58](#)）。

使用 RTC 从深度掉电模式唤醒的步骤如下：

1. 通过设置 RTC 匹配寄存器生成自定时 RTC 唤醒中断。
 - PMU 将会开启片内调压器。当内核电压达到上电复位 (POR) 的跳变点时，系统就会复位，微控制器也将重新启动。
 - 除了 GPREG0 到 GPREG3 之外，所有寄存器都将处于复位状态。
 - RTC 寄存器的内容将会被保留。
2. 启动微控制器之后，可以读取 PCON 寄存器（[表 57](#)）中的深度掉电标志，以证明复位是由深度掉电模式的唤醒事件引起的。
3. 清除 PCON 寄存器（[表 55](#)）中的深度掉电标志。
4. 根据需要读取通用寄存器的内容。
5. 读取 RTC 计数。
6. （可选）清除 RTC ICR 寄存器中的 RTC 中断和 NVIC 中的挂起中断。
7. （可选）读取通用寄存器（[表 57](#) 和 [表 58](#)）中储存的数据。

8. 为下一个深度掉电循环设置 PMU。

4.10 系统 PLL 的功能描述

LPC122x 利用系统 PLL 为内核和外设创建时钟。

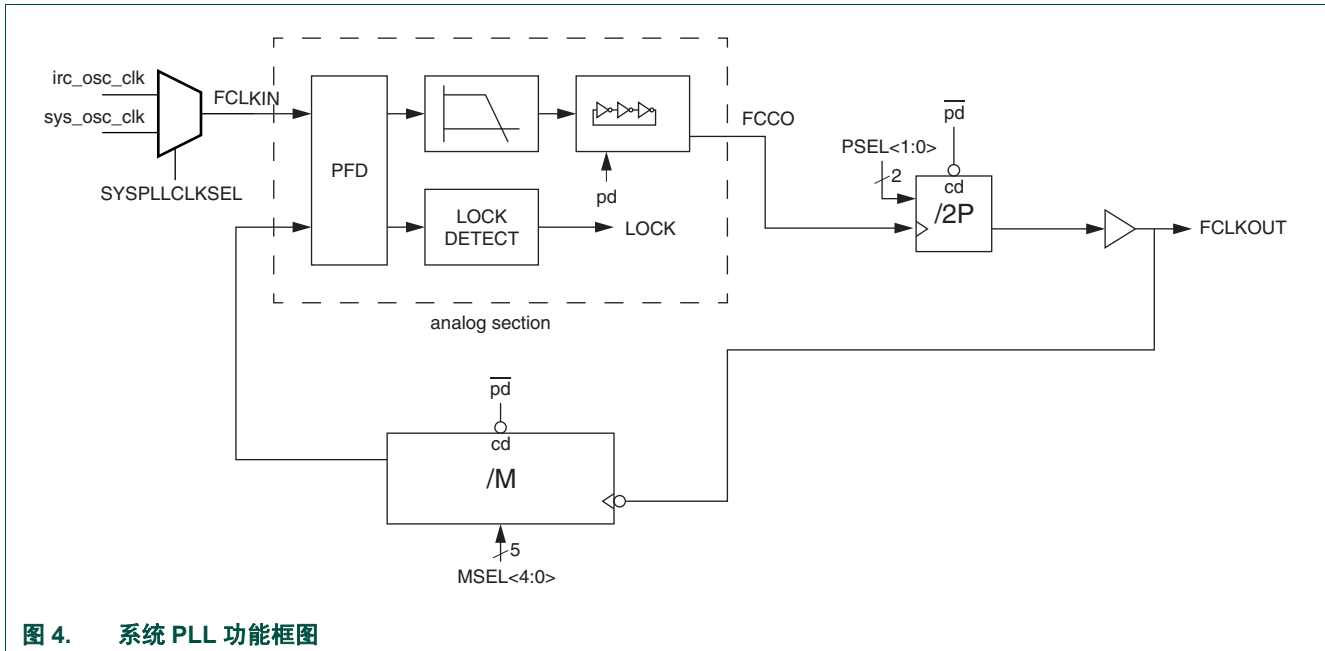


图 4. 系统 PLL 功能框图

此 PLL 的功能框图如图 4 所示。输入频率的范围从 10 MHz 到 25 MHz。输入时钟直接馈入相位频率检测器 (PFD)。该模块会比较其输入的相位和频率，并在相位和 / 或频率不匹配时生成控制信号。环形滤波器过滤掉这些控制信号并驱动电流控制振荡器 (CCO)，从而生成主时钟以及可选的两个额外相位。CCO 的频率范围从 156 MHz 到 320 MHz。这些时钟通过可编程的后置分频器除以 $2 \times P$ 来创建输出时钟，或者直接发送至输出。然后主输出时钟通过可编程的反馈分频器除以 M ，生成反馈时钟。当 PLL 锁定到输入时钟发出信号时，相位频率检测器的输出信号也由锁定检测器监控。

注：必须选择 P 和 M 的分频器值，以使 PLL 输出时钟的频率 FCLKOUT 小于 100 MHz。

4.10.1 锁定检测器

锁定检测器测量输入和反馈时钟上升沿之间的相位差异。只有该差异在超过 8 个连续输入时钟周期后都小于所谓的“锁定标准”时，锁定输出才从低电平转换到高电平。单个过大的相位差会立即使计数器复位，并造成锁定信号下降（如果为高电平）。要求连续 8 个相位测量值都低于某一特定数字，可确保锁定检测器直到输入和反馈时钟的相位和频率都排列好之后才会指示锁定。这样便能有效防止错误的锁定显示，从而确保无干扰的锁定信号。

4.10.2 掉电控制

为了减少在不需要 PLL 时钟时的功耗，集成了掉电模式。可通过在掉电配置寄存器中将 SYSPLL_PD 位设置为 1 来启用该模式（表 50）。在这种模式下，将会关闭内部电流参考，振荡器和相位频率检测器将停止，并且分频器进入复位状态。而在掉电模式下，锁定输出将会降低，表示 PLL 未锁定。当通过设置 SYSPLL_PD 位为 0 结束掉电模式时，PLL 将恢复其正常工作，并且一旦输入时钟重新获得锁定，锁定信号将变为高电平。

4.10.3 分频器比率编程

后置分频器

后置分频器的分频率由 PSEL 位控制。分频率为 PSEL 位选择的 P 值的两倍，如表 10 所示。这样能确保输出时钟的占空比为 50%。

反馈分频器

反馈分频器的分频率由 MSEL 位控制。PLL 输出时钟和输入时钟之间的分频率等于 MSEL 位十进制数值加 1，如表 10 中所规定的。

更改分频器值

不推荐在 PLL 运行时更改分频器比率。由于无法将 MSEL 和 PSEL 值的变更同步到分频器，因此计数器可能会读到未定义的值，从而造成输出时钟频率不必要地升高或下降。更改分频器设置的建议做法是使 PLL 掉电，调整分频器设置，然后再让 PLL 重新启动。

4.10.4 频率选择

PLL 频率的计算公式使用下列参数（另请参见图 3）：

表 53. PLL 频率参数

参数	系统 PLL
FCLKIN	SYSPLLCLKSEL 多路复用器中 sys_pllclk (系统 PLL 的输入时钟) 的频率 (参见章节 4.5.9)。
FCCO	电流控制振荡器 (CCO) 的频率； 156 到 320 MHz。
FCLKOUT	sys_pllclkout 的频率。FCLKOUT 必须小于 100 MHz。
P	系统 PLL 的后置分频器比率； SYSPLLCTRL 中的 PSEL 位 (参见章节 4.5.3)。
M	系统 PLL 的反馈分频寄存器； SYSPLLCTRL 中的 MSEL 位 (参见章节 4.5.3)。

4.10.4.1 正常模式

在正常模式下启用后置分频器，占空比时钟为 50%，频率关系如下：

(1)

$$F_{clkout} = M \times F_{clkkin} = (FCCO)/(2 \times P)$$

要选择合适的 M 值和 P 值，建议执行以下步骤：

- 1. 指定输入时钟频率 Fclkkin。
- 2. 计算 M 值以获得所需的输出频率 Fclkout， $M = F_{clkout} / F_{clkkin}$ 。
- 3. 找出一个值，使得 $FCCO = 2 \times P \times F_{clkout}$ 。
- 4. 检查所有的频率和分频器值是否符合表 10 中指定的限值。
- 5. 确保 FCLKOUT < 100 MHz。

表 54. PLL 配置示例

PLL 输入时钟 sys_pllclkin (Fclkin)	主时钟 (Fclkout)	MSEL 位 表 10	M 分频器值	PSEL 位 表 10	P 分频器 值	FCCO 频率	FLASH_ OVERRIDE 位 表 9	SysAHBC LKDIV 表 20
12 MHz	24 MHz	00001 (二进制)	2	10 (二进制)	4	192 MHz	1	1
12 MHz	36 MHz	00010 (二进制)	3	10 (二进制)	4	288 MHz	0	1

5.1 本章导读

电源管理单元 (PMU) 在所有 LPC122x 部件上都相同。

5.2 简介

PMU 控制深度掉电模式。PMU 中的四个通用寄存器可用于在深度掉电模式下保存数据。PMU 寄存器在深度睡眠和深度掉电模式下可保持其状态。

5.3 寄存器描述

表 55. 寄存器简介：PMU（基址 0x4003 8000）

名称	访问类型	地址偏移	描述	复位值
PCON	R/W	0x000	电源控制寄存器	0x0
GPREG0	R/W	0x004	通用寄存器 0	0x0
GPREG1	R/W	0x008	通用寄存器 1	0x0
GPREG2	R/W	0x00C	通用寄存器 2	0x0
GPREG3	R/W	0x010	通用寄存器 3	0x0
SYSCFG	R/W	0x014	系统配置寄存器 (RTC 时钟控制和 WAKEUP 引脚迟滞)。	0x0

5.3.1 电源控制寄存器

电源控制寄存器选择是否进入其中一种 ARM Cortex-M0 控制的掉电模式（睡眠模式或深度睡眠模式）或深度掉电模式，并分别提供睡眠或深度睡眠模式及深度掉电模式的标志。有关如何进入掉电模式的详情，请参阅[章节 4.7](#)。

表 56. 电源控制寄存器（PCON，地址 0x4003 8000）位描述

位	符号	值	描述	复位值
0	-	-	保留。此位不能写 1。	0x0
1	DPDEN		深度掉电模式使能	0
		0	ARM WFI 将进入睡眠或深度睡眠模式（ARM Cortex-M0 内核的时钟关闭）。	
		1	如果 WDLOCKDP = 0，则 ARM WFI 将进入深度掉电模式（ARM Cortex-M0 内核掉电）（请参阅 表 266 “看门狗模式寄存器（MOD - 0x4000 4000）位描述” ）。	
7:2	-	-	保留。此位不能写 1。	0x0
8	SLEEPFLAG		睡眠模式标志	0
		0	读：不会进入掉电模式。LPC122x 处于运行模式。 写：无效。	
		1	读：进入睡眠 / 深度睡眠或深度掉电模式。 写：写入 1 会将 SLEEPFLAG 位清除为 0。	
10:9	-	-	保留。此位不能写 1。	0x0

表 56. 电源控制寄存器 (PCON, 地址 0x4003 8000) 位描述 (续)

位	符号	值	描述	复位值
11	DPDFLAG		深度掉电标志	0x0
		0	读: 不会进入深度掉电模式。 写: 无效。	0x0
		1	读: 进入深度掉电模式。 写: 清除深度掉电标志。	0x0
31:12	-	-	保留。此位不能写 1。	0x0

5.3.2 通用寄存器 0 至 3

在深度掉电模式下，如果 V_{DD(3V3)} 引脚仍通电，但芯片已进入深度掉电模式，通用寄存器可保留数据。只有在完全移除芯片上的所有电源后进行依源启动，才会复位通用寄存器。

表 57. 通用寄存器 0 至 3 (GPREG0 - GPREG3, 地址 0x4003 8004 至 0x4003 8010) 位描述

位	符号	描述	复位值
31:0	GPDATA	深度掉电模式下保留数据。	0x0

5.3.3 系统配置寄存器

此寄存器控制 RTC 的时钟输入和 WAKEUP 引脚的迟滞。

32 kHz RTC 振荡器中有三个时钟可供选择：1 Hz 时钟（默认值）、延迟的 1 Hz 时钟及 1 kHz 时钟。此外，可以选择由 RTC 时钟分频器分频的主时钟派生的外围设备 RTC 时钟作为 RTC 时钟源。

注: 必须先选择 RTC 时钟源，才能在 SYSAHBCLKCTRL 寄存器（见[表 21](#)）中启用 RTC。当 RTC 正在运行时，不得更改时钟源。

注: 如果引脚 V_{DD(3V3)} 上的外加电压降低低于 < 待定 > V，为了能将芯片从深度掉电模式中唤醒，必须禁用 WAKEUP 输入引脚的迟滞。

表 58. 系统配置寄存器 (SYSCFG, 地址 0x4003 8014) 位描述

位	符号	值	描述	复位值
9:0	-	-	保留。此位不能写 1。	0x0
10	WAKEUPHYS		WAKEUP 引脚迟滞使能	0x0
		0	禁用 WAKUP 引脚迟滞。	
		1	启用 WAKEUP 引脚迟滞。	
14:11	RTCCLK		RTC 时钟源选择（X = 无需考虑）。 0000 = 1 Hz 时钟 0001 = 延迟的 1 Hz 时钟 101X = 1 kHz 时钟 01XX = RTC PCLK (RTC 时钟分频器值分频的主时钟)	0000
31:15	-		保留。此位不能写 1。	0x0

6.1 本章导读

每个引脚都分配有一个 IOCON 寄存器。用于不可用引脚的 IOCON 寄存器被保留（见[表 59](#)）。控制 GPIO 端口 2 引脚的 IOCON 寄存器仅在 64 引脚封装上提供。

表 59. 可用的 GPIO 引脚 /IOCON 寄存器

封装	GPIO0	GPIO1	GPIO2	总 GPIO 引脚 / IOCON 寄存器
LQFP48	PIO0_0 到 PIO0_31	PIO1_0 到 PIO1_6	-	39
LQFP64	PIO0_0 到 PIO0_31	PIO1_0 到 PIO1_6	PIO2_0 到 PIO2_15	55
LQFP100 [1]	PIO0_0 到 PIO0_31	PIO1_0 到 PIO1_6	PIO2_0 到 PIO2_15	55

[1] 器件 LPC12D27。

6.2 特性

I/O 配置寄存器控制焊盘的电气特性。可编程下列特性：

- 引脚功能
- 引脚模式：内部上拉电阻使能 / 禁能
- 引脚驱动
- 托管 ADC 输入的焊盘的模拟输入或数字模式
- 托管 I²C 总线接口的焊盘的 I²C 总线模式或 GPIO 模式
- I²C 总线引脚的真正可编程的开漏模式

6.3 简介

6.3.1 引脚功能

IOCON 寄存器中的 FUNC 位可设为 GPIO (FUNC = 000) 或外设功能。如果将引脚配置为 GPIO 引脚，则 DIR 寄存器决定引脚是配置为输入还是输出（见[表 129](#)）。对于任何外设功能，会根据引脚的功能自动控制引脚方向。GPIO_{On}DIR 寄存器对外设功能无效。

6.3.2 引脚模式

IOCON 寄存器的 MODE 位允许为每个引脚使能或禁能片内上拉电阻。默认情况下，除 I²C 总线引脚 PIO0_10 和 PIO0_11 外 - 这两个引脚没有可编程的上拉电阻，所有引脚的上拉电阻都被使能。

6.3.3 引脚驱动

对于每个正常驱动引脚，可以选择两种电平的输出驱动，即低电平模式和高电平模式。有四个引脚（PIO0_27、PIO0_28、PIO0_29、PIO0_12）被指定为高电平驱动引脚，带高电平模式和低电平模式输出驱动。有关详情，请参见 LPC122x 数据表。

6.3.4 开漏模式

可为所有数字 I/O 引脚使能开漏模式。除引脚 PIO0_10 和 PIO0_11 外，该模式不是真正的开漏模式。输入不能上拉至超过 V_{DD(IO)}。

6.3.5 A/D 模式

在 A/D 模式下，数字接收器断开以获得准确的输入电压用于模数转换。该模式可用于控制能用作 ADC 输入的引脚的那些 IOCON 寄存器。如果选择 A/D 模式，引脚模式设置无效。

6.3.6 I²C 总线模式

I²C 总线引脚 PIO0_10 和 PIO0_11 可编程为支持真正的开漏模式，不管选择了 I²C 功能还是其它数字功能。如果选择了 I²C 功能，则支持所有三个 I²C 模式，即标准模式、快速模式和超快速模式。可为所有功能配置一个数字干扰滤波器。引脚 PIO0_10 和 PIO0_11 独立于被编程的功能作为高电流接收器驱动程序 (20 mA) 工作。

6.3.7 可编程干扰滤波器

所有 PIO 引脚都配有可编程数字干扰滤波器。滤波器在一个可选择的时间段内抑制输入脉冲，这个时间段可短于一个、两个或三个滤波器时钟周期 (S_MODE = 1、2 或 3)。滤波器时钟可从 7 个外设时钟 PCLK0 ~ 6 中选择，这些时钟使用 IOCONFIGCLKDIV0 ~ 6 (见 [表 31](#)) 寄存器从主时钟 (见 [图 3](#)) 推导得出。也可以完全绕过滤波器。

满足以下条件时，T_{pulse} 期间内任意极性的输入脉冲将被抑制：
 $T_{pulse} < T_{PCLKn} \times S_MODE$

长一个滤波器时钟周期的输入脉冲也可能被抑制：

$$T_{pulse} = T_{PCLKn} \times (S_MODE + 1)$$

注：滤波效果通过以下过程实现：要求输入信号在滤波器时钟内 (S_MODE + 1) 个连续边沿保持稳定然后才传递到芯片。使能滤波器会导致延迟将信号发往内部逻辑，仅当某个应用特别要求时才应这样做。对于高速或时间关键功能，例如定时器捕获输入或 SSP 功能，请确保绕过该滤波器。

如果必须最大限度缩短输入信号的延时，请选择更快的 PCLK 和更高的采样模式 (S_MODE)，以最大限度减少潜在额外时钟周期的影响。

如果必须最大限度降低对噪音尖峰的敏感性，应选择更慢的 PCLK 和更低的采样模式。

6.4 寄存器描述

[表 60](#) 显示了 IOCONFIG 寄存器。每个多路复用引脚都与一个寄存器关联，该寄存器用于对引脚的功能和电气特性编程。寄存器名称来自复位后引脚的默认功能。请注意：有的引脚复位为除 GPIO 之外的功能。相应寄存器有一个反映引脚复位后功能的前缀：串行线调试功能 (SWDIO 或 SWCLK) 或保留功能 (R)。

表 60. 寄存器简介：I/O 配置模块 (基址 0x4004 4000)

名称	访问类型	地址偏移	描述	复位值	参考
-	R/W	0x000	保留。	-	-
-	R/W	0x004	保留。	-	-
PIO0_19	R/W	0x008	配置引脚 PIO0_19/ACMP0_I0/CT32B0_1。	0x0000 0090	表 63

表 60. 寄存器简介: I/O 配置模块 (基址 0x4004 4000) (续)

名称	访问类型	地址偏移	描述	复位值	参考
PIO0_20	R/W	0x00C	配置引脚 PIO0_20/ACMP0_I1/CT32B0_2。	0x0000 0090	表 64
PIO0_21	R/W	0x010	配置引脚 PIO0_21/ACMP0_I2/CT32B0_3。	0x0000 0090	表 65
PIO0_22	R/W	0x014	配置引脚 PIO0_22/ACMP0_I3。	0x0000 0090	表 66
PIO0_23	R/W	0x018	配置引脚 PIO0_23/ACMP1_I0/CT32B1_0。	0x0000 0090	表 67
PIO0_24	R/W	0x01C	配置引脚 PIO0_24/ACMP1_I1/CT32B1_1。	0x0000 0090	表 68
SWDIO_PIO0_25	R/W	0x020	配置引脚 SWDIO/ACMP1_I2/ CT32B1_2/PIO0_25。	0x0000 0090	表 69
SWCLK_PIO0_26	R/W	0x024	配置引脚 SWCLK/PIO0_26/ACMP1_I3/ CT32B1_3/PIO0_26	0x0000 0090	表 70
PIO0_27	R/W	0x028	配置引脚 PIO0_27/ACMP0_O。	0x0000 0090	表 71
PIO2_12	R/W	0x02C	配置引脚 PIO2_12/RXD1。	0x0000 0090	表 72
PIO2_13	R/W	0x030	配置引脚 PIO2_13/TXD1。	0x0000 0090	表 73
PIO2_14	R/W	0x034	配置引脚 PIO2_14。	0x0000 0090	表 74
PIO2_15	R/W	0x038	配置引脚 PIO2_15。	0x0000 0090	表 75
PIO0_28	R/W	0x03C	配置引脚 PIO0_28/ACMP1_O/CT16B0_0。	0x0000 0090	表 76
PIO0_29	R/W	0x040	配置引脚 PIO0_29/ROSC/CT16B0_1。	0x0000 0090	表 77
PIO0_0	R/W	0x044	配置引脚 PIO0_0/RTS0。	0x0000 0090	表 78
PIO0_1	R/W	0x048	配置引脚 PIO0_1CT32B0_0/RXD0。	0x0000 0090	表 79
PIO0_2	R/W	0x04C	配置引脚 PIO0_2/TXD0/CT32B0_1。	0x0000 0090	表 80
-	R/W	0x050	保留	-	-
PIO0_3	R/W	0x054	配置引脚 PIO0_3/DTR0/CT32B0_2。	0x0000 0090	表 81
PIO0_4	R/W	0x058	配置引脚 PIO0_4/DSR0/CT32B0_3。	0x0000 0090	表 82
PIO0_5	R/W	0x05C	配置引脚 PIO0_5。	0x0000 0090	表 83
PIO0_6	R/W	0x060	配置引脚 PIO0_6/RI0/CT32B1_0。	0x0000 0090	表 84
PIO0_7	R/W	0x064	配置引脚 PIO0_7CTS0/CT32B1_1。	0x0000 0090	表 85
PIO0_8	R/W	0x068	配置引脚 PIO0_8/RXD1/CT32B1_2。	0x0000 0090	表 86
PIO0_9	R/W	0x06C	配置引脚 PIO0_9/TXD1/CT32B1_3。	0x0000 0090	表 87
PIO2_0	R/W	0x070	配置引脚 PIO2_0/CT16B0_0/RTS0。	0x0000 0090	表 88
PIO2_1	R/W	0x074	配置引脚 PIO2_1/CT16B0_1/RXD0。	0x0000 0090	表 89
PIO2_2	R/W	0x078	配置引脚 PIO2_2/CT16B1_0/TXD0。	0x0000 0090	表 90
PIO2_3	R/W	0x07C	配置引脚 PIO2_3/CT16B1_1/DTR0。	0x0000 0090	表 91
PIO2_4	R/W	0x080	配置引脚 PIO2_4/CT32B0_0/CTS0。	0x0000 0090	表 92
PIO2_5	R/W	0x084	配置引脚 PIO2_5/CT32B0_1/DCD0。	0x0000 0090	表 93
PIO2_6	R/W	0x088	配置引脚 PIO2_6/CT32B0_2/RI0。	0x0000 0090	表 94
PIO2_7	R/W	0x08C	配置引脚 PIO2_7/CT32B0_3/DSR0。	0x0000 0090	表 95
PIO0_10	R/W	0x090	配置引脚 PIO0_10/SCL。	0x0000 0080	表 96
PIO0_11	R/W	0x094	配置引脚 PIO0_11/SDA/CT16B0_0。	0x0000 0080	表 97
PIO0_12	R/W	0x098	配置引脚 PIO0_12/CLKOUT/CT16B0_1。	0x0000 0090	表 98
RESET_PIO0_13	R/W	0x09C	配置引脚 RESET/PIO0_13。	0x0000 0090	表 99
PIO0_14	R/W	0x0A0	配置引脚 PIO0_14/SSP_CLK。	0x0000 0090	表 100
PIO0_15	R/W	0x0A4	配置引脚 PIO0_15/SSP_SSEL/CT16B1_0。	0x0000 0090	表 101

表 60. 寄存器简介: I/O 配置模块 (基址 0x4004 4000) (续)

名称	访问类型	地址偏移	描述	复位值	参考
PIO0_16	R/W	0x0A8	配置引脚 PIO0_16/SSP_MISO/CT16B1_1。	0x0000 0090	表 102
PIO0_17	R/W	0x0AC	配置引脚 PIO0_17/SSP_MOSI。	0x0000 0090	表 103
PIO0_18	R/W	0x0B0	配置引脚 PIO0_18/SWCLK/CT32B0_0。	0x0000 0090	表 104
R_PIO0_30	R/W	0x0B4	配置引脚 R/PIO0_30/AD0。	0x0000 0090	表 105
R_PIO0_31	R/W	0x0B8	配置引脚 R/PIO0_31/AD1。	0x0000 0090	表 106
R_PIO1_0	R/W	0x0BC	配置引脚 R/PIO1_0/AD2。	0x0000 0090	表 107
R_PIO1_1	R/W	0x0C0	配置引脚 R/PIO1_1/AD3。	0x0000 0090	表 108
PIO1_2	R/W	0x0C4	配置引脚 PIO1_2/SWDIO/AD4。	0x0000 0090	表 109
PIO1_3	R/W	0x0C8	配置引脚 PIO1_3/AD5/WAKEUP。	0x0000 0090	表 110
PIO1_4	R/W	0x0CC	配置引脚 PIO1_4/AD6	0x0000 0090	表 111
PIO1_5	R/W	0x0D0	配置引脚 PIO1_5/AD7/CT16B1_0。	0x0000 0090	表 112
PIO1_6	R/W	0x0D4	配置引脚 PIO1_6/CT16B1_1。	0x0000 0090	表 113
-	-	0x0D8	保留。	-	-
-	-	0x0DC	保留。	-	-
PIO2_8	R/W	0x0E0	配置引脚 PIO2_8/CT32B1_0。	0x0000 0090	表 114
PIO2_9	R/W	0x0E4	配置引脚 PIO2_9/CT32B1_1。	0x0000 0090	表 115
PIO2_10	R/W	0x0E8	配置引脚 PIO2_10/CT32B1_2/TXD1。	0x0000 0090	表 116
PIO2_11	R/W	0x0EC	配置引脚 PIO2_11/CT32B1_3/RXD1。	0x0000 0090	表 117

6.4.1 引脚配置寄存器

[表 61](#) 显示了所有 IOCON 寄存器的位分配 (PIO0_10 和 PIO0_11 除外)。[表 62](#) 显示了引脚 PIO0_10 和 PIO0_11 的位分配, 这两个引脚用于配置真正开漏模式以符合完全 I²C 总线规格。

表 61. IOCON 寄存器位分配 (I²C 引脚除外)

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		000	选择功能 0 (默认)	
		001	选择功能 1。	
		010	选择功能 2。	
		011	选择功能 3。	
		100	选择功能 4。	
		101	选择功能 5。	
		110	选择功能 6。	
		111	保留。	
3	-		保留	0
4	MODE		选择功能模式 (片内上拉电阻控制)。	1
		0	无效 (未使能上拉电阻)。	
		1	已使能上拉电阻。	
5	-		保留。	0

表 61. IOCON 寄存器位分配 (I²C 引脚除外) (续)

位	符号	值	描述	复位值
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	禁能。	
		1	已使能开漏模式。 注：这不是真正的开漏模式。输入不能上拉至超过 V _{DD(IO)} 。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	采样 1 个滤波器时钟周期。不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	采样 2 个滤波器时钟周期。不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	采样 3 个滤波器时钟周期。不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。值 0x7 保留。（见表 31）。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

表 62. IOCON 寄存器位分配 (I²C 总线引脚 PIO0_10 和 PIO0_11)

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		000	选择功能 0（默认）	
		001	选择功能 1。	
		010	选择功能 2。	
		011	选择功能 3。	
		100	选择功能 4。	
		101	选择功能 5。	
		110	选择功能 6。	
		111	保留。	
5:3	-		保留。	000
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
9:7	-	-	保留。	001
10	TOD		真正开漏模式。	0
		0	禁能。	
		1	真正已使能开漏模式。	
12:11	S_MODE		采样模式	00
		00	绕过输入滤波器。	
		01	采样 1 个滤波器时钟周期。不足 1 个滤波器时钟的输入脉冲将被抑制。	
		10	采样 2 个滤波器时钟周期。不足 2 个滤波器时钟的输入脉冲将被抑制。	
		11	采样 3 个滤波器时钟周期。不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。 (见 表 31)	000
		000	IOCONFIGCLKDIV0。	
		001	IOCONFIGCLKDIV1。	
		010	IOCONFIGCLKDIV2。	
		011	IOCONFIGCLKDIV3。	
		100	IOCONFIGCLKDIV4。	
		101	IOCONFIGCLKDIV5。	
		110	IOCONFIGCLKDIV6。	
		111	保留。	
31:16	-	-	保留。	0

6.4.2 PIO0_19 寄存器

表 63. PIO0_19 寄存器（PIO0_19，地址 0x4004 4008）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_19。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP0_I0。	
		0x3	选择功能 CT32B0_CAP1。	
		0x4	选择功能 CT32B0_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.3 PIO0_20 寄存器

表 64. PIO0_20 寄存器（PIO0_20，地址 0x4004 400C）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_20。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP0_I1。	
		0x3	选择功能 CT32B0_CAP2。	
		0x4	选择功能 CT32B0_MAT2。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.4 PIO0_21 寄存器

表 65. PIO0_21 寄存器（PIO0_21，地址 0x4004 4010）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_21。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP0_I2。	
		0x3	选择功能 CT32B0_CAP3。	
		0x4	选择功能 CT32B0_MAT3。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.5 PIO0_22 寄存器

表 66. PIO0_22 寄存器（PIO0_22，地址 0x4004 4014）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_22。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP0_I3。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.6 PIO0_23 寄存器

表 67. PIO0_23 寄存器（PIO0_23，地址 0x4004 4018）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_23。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP1_I0。	
		0x3	选择功能 CT32B1_CAP0。	
		0x4	选择功能 CT32B1_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.7 PIO0_24 寄存器

表 68. PIO0_24 寄存器（PIO0_24，地址 0x4004 401C）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_24。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP1_I1。	
		0x3	选择功能 CT32B1_CAP1。	
		0x4	选择功能 CT32B1_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.8 SWDIO_PIO0_25 寄存器

表 69. SWDIO_PIO0_25 寄存器（SWDIO_PIO0_25，地址 0x4004 4020）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 SWDIO。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP1_I2。	
		0x3	选择功能 CT32B1_CAP2。	
		0x4	选择功能 CT32B1_MAT2。	
		0x5	保留。	
		0x6	选择功能 PIO0_25。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	

表 69. SWDIO_PIO0_25 寄存器 (SWDIO_PIO0_25, 地址 0x4004 4020) 位描述 (续)

位	符号	值	描述	复位值
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.9 SWCLK_PIO0_26 寄存器

表 70. SWCLK_PIO0_26 寄存器 (SWCLK_PIO0_26, 地址 0x4004 4024) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 SWCLK。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP1_I3。	
		0x3	选择功能 CT32B1_CAP3。	
		0x4	选择功能 CT32B1_MAT3。	
		0x5	保留。	
		0x6	选择功能 PIO0_26。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择高模式电流。	
		1	已选择低模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 70. SWCLK_PIO0_26 寄存器 (SWCLK_PIO0_26, 地址 0x4004 4024) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.10 PIO0_27 寄存器

表 71. PIO0_27 寄存器 (PIO0_27, 地址 0x4004 4028) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_26。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP0_O。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（高电流驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 71. PIO0_27 寄存器 (PIO0_27, 地址 0x4004 4028) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.11 PIO2_12 寄存器

表 72. PIO2_12 寄存器 (PIO2_12, 地址 0x4004 402C) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_12。	
		0x1	保留。不使用。	
		0x2	保留。	
		0x3	选择功能 RXD1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 72. PIO2_12 寄存器 (PIO2_12, 地址 0x4004 402C) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.12 PIO2_13 寄存器

表 73. PIO2_13 寄存器 (PIO2_13, 地址 0x4004 4030) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_13。	
		0x1	保留。不使用。	
		0x2	保留。	
		0x3	选择功能 TXD1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 73. PIO2_13 寄存器 (PIO2_13, 地址 0x4004 4030) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.13 PIO2_14 寄存器

表 74. PIO2_14 寄存器 (PIO2_14, 地址 0x4004 4034) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_14。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 74. PIO2_14 寄存器 (PIO2_14, 地址 0x4004 4034) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
31:16	-	-	保留。	0

6.4.14 PIO2_15 寄存器

表 75. PIO2_15 寄存器 (PIO2_15, 地址 0x4004 4038) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_15。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 75. PIO2_15 寄存器 (PIO2_15, 地址 0x4004 4038) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.15 PIO0_28 寄存器

表 76. PIO0_28 寄存器 (PIO0_28, 地址 0x4004 403C) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_28。	
		0x1	保留。不使用。	
		0x2	选择功能 ACMP1_O。	
		0x3	选择功能 CT16B0_CAP0。	
		0x4	选择功能 CT16B0_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（高电流驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 76. PIO0_28 寄存器 (PIO0_28, 地址 0x4004 403C) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
31:16	-	-	保留。	0

6.4.16 PIO0_29 寄存器

表 77. PIO0_29 寄存器 (PIO0_29, 地址 0x4004 4040) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_29。	
		0x1	保留。不使用。	
		0x2	选择功能 ROSC。	
		0x3	选择功能 CT16B0_CAP1。	
		0x4	选择功能 CT16B0_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（高电流驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 77. PIO0_29 寄存器 (PIO0_29, 地址 0x4004 4040) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
31:16	-	-	保留。	0

6.4.17 PIO0_0 寄存器

表 78. PIO0_0 寄存器 (PIO0_0, 地址 0x4004 4044) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_0。	
		0x1	保留。不使用。	
		0x2	选择功能 RTS0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁用开漏模式。	
		1	已使能开漏模式。	

表 78. PIO0_0 寄存器 (PIO0_0, 地址 0x4004 4044) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.18 PIO0_1 寄存器

表 79. PIO0_1 寄存器 (PIO0_1, 地址 0x4004 4048) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_1。	
		0x1	保留。不使用。	
		0x2	选择功能 RXD0。	
		0x3	选择功能 CT32B0_CAP0。	
		0x4	选择功能 CT32B0_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 79. PIO0_1 寄存器 (PIO0_1, 地址 0x4004 4048) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.19 PIO0_2 寄存器

表 80. PIO0_2 寄存器 (PIO0_2, 地址 0x4004 404C) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_2。	
		0x1	保留。不使用。	
		0x2	选择功能 TXD0。	
		0x3	选择功能 CT32B0_CAP1。	
		0x4	选择功能 CT32B0_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 80. PIO0_2 寄存器 (PIO0_2, 地址 0x4004 404C) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.20 PIO0_3 寄存器

表 81. PIO0_3 寄存器 (PIO0_3, 地址 0x4004 4054) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_3。	
		0x1	保留。不使用。	
		0x2	选择功能 DTR0。	
		0x3	选择功能 CT32B0_CAP2。	
		0x4	选择功能 CT32B0_MAT2。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 81. PIO0_3 寄存器 (PIO0_3, 地址 0x4004 4054) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
31:16	-	-	保留。	0

6.4.21 PIO0_4 寄存器

表 82. PIO0_4 寄存器 (PIO0_4, 地址 0x4004 4058) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_4。	
		0x1	保留。不使用。	
		0x2	选择功能 DSR0。	
		0x3	选择功能 CT32B0_CAP3。	
		0x4	选择功能 CT32B0_MAT3。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 82. PIO0_4 寄存器 (PIO0_4, 地址 0x4004 4058) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.22 PIO0_5 寄存器

表 83. PIO0_5 寄存器 (PIO0_5, 地址 0x4004 405C) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_5。	
		0x1	保留。不使用。	
		0x2	选择功能 DCD0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁用开漏模式。	
		1	已使能开漏模式。	

表 83. PIO0_5 寄存器 (PIO0_5, 地址 0x4004 405C) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.23 PIO0_6 寄存器

表 84. PIO0_6 寄存器 (PIO0_6, 地址 0x4004 4060) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_6。	
		0x1	保留。不使用。	
		0x2	选择功能 RI0。	
		0x3	选择功能 CT32B1_CAP0。	
		0x4	选择功能 CT32B1_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 84. PIO0_6 寄存器 (PIO0_6, 地址 0x4004 4060) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
31:16	-	-	保留。	0

6.4.24 PIO0_7 寄存器

表 85. PIO0_7 寄存器 (PIO0_7, 地址 0x4004 4064) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_7。	
		0x1	保留。不使用。	
		0x2	选择功能 CTS0。	
		0x3	选择功能 CT32B1_CAP1。	
		0x4	选择功能 CT32B1_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 85. PIO0_7 寄存器 (PIO0_7, 地址 0x4004 4064) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.25 PIO0_8 寄存器

表 86. PIO0_8 寄存器 (PIO0_8, 地址 0x4004 4068) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_8。	
		0x1	保留。不使用。	
		0x2	选择功能 RXD1。	
		0x3	选择功能 CT32B1_CAP2。	
		0x4	选择功能 CT32B1_MAT2。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 86. PIO0_8 寄存器 (PIO0_8, 地址 0x4004 4068) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.26 PIO0_9 寄存器

表 87. PIO0_9 寄存器 (PIO0_9, 地址 0x4004 406C) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_9。	
		0x1	保留。不使用。	
		0x2	选择功能 TXD1。	
		0x3	选择功能 CT32B1_CAP3。	
		0x4	选择功能 CT32B1_MAT3。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 87. PIO0_9 寄存器 (PIO0_9, 地址 0x4004 406C) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.27 PIO2_0 寄存器

表 88. PIO2_0 寄存器 (PIO2_0, 地址 0x4004 4070) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_0。	
		0x2	保留。不使用。	
		0x3	选择功能 CT16B0_CAP0。	
		0x4	选择功能 CT16B0_MAT0。	
		0x5	选择功能 RTS0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 88. PIO2_0 寄存器（PIO2_0，地址 0x4004 4070）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.28 PIO2_1 寄存器

表 89. PIO2_1 寄存器（PIO2_1，地址 0x4004 4074）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_1。	
		0x1	保留。不使用。	
		0x2	选择功能 CT16B0_CAP1。	
		0x3	选择功能 CT16B0_MAT1。	
		0x4	选择功能 RXD0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 89. PIO2_1 寄存器（PIO2_1，地址 0x4004 4074）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.29 PIO2_2 寄存器

表 90. PIO2_2 寄存器（PIO2_2，地址 0x4004 4078）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_2。	
		0x1	保留。不使用。	
		0x2	选择功能 CT16B1_CAP0。	
		0x3	选择功能 CT16B1_MAT0。	
		0x4	选择功能 TXD0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 90. PIO2_2 寄存器（PIO2_2，地址 0x4004 4078）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.30 PIO2_3 寄存器

表 91. PIO2_3 寄存器（PIO2_3，地址 0x4004 407C）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_3。	
		0x1	保留。不使用。	
		0x2	选择功能 CT16B1_CAP1。	
		0x3	选择功能 CT16B1_MAT1。	
		0x4	选择功能 DTR0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 91. PIO2_3 寄存器（PIO2_3，地址 0x4004 407C）位描述（续）

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.31 PIO2_4 寄存器

表 92. PIO2_4 寄存器（PIO2_4，地址 0x4004 4080）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_4。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B0_CAP0。	
		0x3	选择功能 CT32B0_MAT0。	
		0x4	选择功能 CTS0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 92. PIO2_4 寄存器 (PIO2_4, 地址 0x4004 4080) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.32 PIO2_5 寄存器

表 93. PIO2_5 寄存器 (PIO2_5, 地址 0x4004 4084) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_5。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B0_CAP1。	
		0x3	选择功能 CT32B0_MAT1。	
		0x4	选择功能 RI0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 93. PIO2_5 寄存器（PIO2_5，地址 0x4004 4084）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.33 PIO2_6 寄存器

表 94. PIO2_6 寄存器（PIO2_6，地址 0x4004 4088）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_6。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B0_CAP2。	
		0x3	选择功能 CT32B0_MAT2。	
		0x4	选择功能 DCD0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 94. PIO2_6 寄存器（PIO2_6，地址 0x4004 4088）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.34 PIO2_7 寄存器

表 95. PIO2_7 寄存器（PIO2_7，地址 0x4004 407C）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_7。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B0_CAP3。	
		0x3	选择功能 CT32B0_MAT3。	
		0x4	选择功能 DSR0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 95. PIO2_7 寄存器（PIO2_7，地址 0x4004 407C）位描述（续）

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.35 PIO0_10 寄存器

表 96. PIO0_10 寄存器（PIO0_10，地址 0x4004 4090）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_10。	
		0x1	保留。不使用。	
		0x2	选择 I2C 总线功能 SCL。	
5:3	-		保留。	000
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
9:7	-	-	保留。	001
10	TOD		真正开漏模式。	0
		0	禁能。	
		1	真正已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	

表 96. PIO0_10 寄存器 (PIO0_10, 地址 0x4004 4090) 位描述 (续)

位	符号	值	描述	复位值
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
		0x0	保留。	
31:16	-	-	保留。	0

6.4.36 PIO0_11 寄存器

表 97. PIO0_11 寄存器 (PIO0_11, 地址 0x4004 4094) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_11。	
		0x1	保留。不使用。	
		0x2	选择 I2C 总线功能 SDA。	
		0x3	选择功能 CT16B0_CAP0。	
		0x4	选择功能 CT16B0_MAT0。	
5:3	-		保留。	000
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
9:7	-	-	保留。	001
10	TOD		真正开漏模式。	0
		0	禁能。	
		1	真正已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	

表 97. PIO0_11 寄存器 (PIO0_11, 地址 0x4004 4094) 位描述 (续)

位	符号	值	描述	复位值
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
		0x0	保留。	
31:16	-	-	保留。	0

6.4.37 PIO0_12 寄存器

表 98. PIO0_12 寄存器 (PIO0_12, 地址 0x4004 4098) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_12。	
		0x1	保留。不使用。	
		0x2	选择功能 CLKOUT。	
		0x3	选择功能 CT16B0_CAP1。	
		0x4	选择功能 CT16B0_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（高电流驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 98. PIO0_12 寄存器 (PIO0_12，地址 0x4004 4098) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.38 RESET_PIO0_13 寄存器

表 99. RESET_PIO0_13 寄存器 (RESET_PIO0_13，地址 0x4004 409C) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 RESET。	
		0x1	选择功能 PIO0_13。	
		0x2	保留。不使用。	
		0x3	保留。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 99. RESET_PIO0_13 寄存器 (RESET_PIO0_13, 地址 0x4004 409C) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.39 PIO0_14 寄存器

表 100. PIO0_14 寄存器 (PIO0_14, 地址 0x4004 40A0) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_14。	
		0x1	保留。不使用。	
		0x2	选择功能 SCK。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 100. PIO0_14 寄存器（PIO0_14，地址 0x4004 40A0）位描述（续）

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.40 PIO0_15 寄存器

表 101. PIO0_15 寄存器（PIO0_15，地址 0x4004 40A4）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_15。	
		0x1	保留。不使用。	
		0x2	选择功能 SSEL。	
		0x3	选择功能 CT16B1_CAP0。	
		0x4	选择功能 CT16B1_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 101. PIO0_15 寄存器（PIO0_15，地址 0x4004 40A4）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.41 PIO0_16 寄存器

表 102. PIO0_16 寄存器（PIO0_16，地址 0x4004 40A8）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_16。	
		0x1	保留。不使用。	
		0x2	选择功能 MISO。	
		0x3	选择功能 CT16B1_CAP1。	
		0x4	选择功能 CT16B1_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 102. PIO0_16 寄存器（PIO0_16，地址 0x4004 40A8）位描述 *（续）*

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.42 PIO0_17 寄存器

表 103. PIO0_17 寄存器（PIO0_17，地址 0x4004 40AC）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_17。	
		0x1	保留。不使用。	
		0x2	选择功能 MOSI。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 103. PIO0_17 寄存器 (PIO0_17, 地址 0x4004 40AC) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.43 PIO0_18 寄存器

表 104. PIO0_18 寄存器 (PIO0_18, 地址 0x4004 40B0) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO0_18。	
		0x1	选择功能 SWCLK。	
		0x2	保留。不使用。	
		0x3	选择功能 CT32B0_CAP0。	
		0x4	选择功能 CT32B0_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	

表 104. PIO0_18 寄存器 (PIO0_18, 地址 0x4004 40B0) 位描述 (续)

位	符号	值	描述	复位值
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.44 R_PIO0_30 寄存器

表 105. R_PIO0_30 寄存器 (R_PIO0_30, 地址 0x4004 40B4) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	保留。	
		0x1	选择功能 PIO0_30。	
		0x2	保留。不使用。	
		0x3	选择功能 AD0。	
3	-		保留	0
4	MODE		选择功能模式 (片内上拉电阻控制)。	1
		0	无效 (未使能上拉电阻)。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0
9	DRV		驱动电流模式 (正常驱动引脚)。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	

表 105. R_PIO0_30 寄存器（R_PIO0_30，地址 0x4004 40B4）位描述（续）

位	符号	值	描述	复位值
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.45 R_PIO0_31 寄存器

表 106. PIO0_31 寄存器（PIO0_31，地址 0x4004 40B8）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	保留。	
		0x1	选择功能 PIO0_31。	
		0x2	保留。不使用。	
		0x3	选择功能 AD1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0

表 106. PIO0_31 寄存器（PIO0_31，地址 0x4004 40B8）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.46 R_PIO1_0 寄存器

表 107. R_PIO1_0 寄存器（R_PIO1_0，地址 0x4004 40BC）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	保留。	
		0x1	选择功能 PIO1_0。	
		0x2	选择功能 AD2。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0

表 107. R_PIO1_0 寄存器 (R_PIO1_0, 地址 0x4004 40BC) 位描述 (续)

位	符号	值	描述	复位值
9	DRV		驱动电流模式 (正常驱动引脚)。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.47 R_PIO1_1 寄存器

表 108. R_PIO1_1 寄存器 (R_PIO1_1, 地址 0x4004 40C0) 位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	保留。	
		0x1	选择功能 PIO1_0。	
		0x2	选择功能 AD3。	
3	-		保留	0
4	MODE		选择功能模式 (片内上拉电阻控制)。	1
		0	无效 (未使能上拉电阻)。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0

表 108. R_PIO1_1 寄存器（R_PIO1_1，地址 0x4004 40C0）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.48 PIO1_2 寄存器

表 109. PIO1_2 寄存器（PIO1_2，地址 0x4004 40C4）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO1_2。	
		0x1	选择功能 SWDIO。	
		0x2	选择功能 AD4。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0

表 109. PIO1_2 寄存器（PIO1_2，地址 0x4004 40C4）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.49 PIO1_3 寄存器

表 110. PIO1_3 寄存器（PIO1_3，地址 0x4004 40C8）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。如果器件处于深度掉电模式，该引脚用作 WAKEUP 引脚，而不管 FUNC 值如何。	000
		0x0	选择功能 PIO1_3。	
		0x1	选择功能 AD5。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0

表 110. PIO1_3 寄存器（PIO1_3，地址 0x4004 40C8）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.50 PIO1_4 寄存器

表 111. PIO1_4 寄存器（PIO1_4，地址 0x4004 40CC）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO1_4。	
		0x1	选择功能 AD6。	
		0x2	保留。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	
8	-		保留。	0

表 111. PIO1_4 寄存器（PIO1_4，地址 0x4004 40CC）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.51 PIO1_5 寄存器

表 112. PIO1_5 寄存器（PIO1_5，地址 0x4004 40D0）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO1_5。	
		0x1	选择功能 AD7。	
		0x2	选择功能 CT16B1_CAP0。	
		0x3	选择功能 CT16B1_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	ADMODE		模拟 / 数字模式	1
		0	模拟模式使能。	
		1	数字模式使能。	

表 112. PIO1_5 寄存器（PIO1_5，地址 0x4004 40D0）位描述 *（续）*

位	符号	值	描述	复位值
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	选择低模式驱动电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.52 PIO1_6 寄存器

表 113. PIO1_6 寄存器（PIO1_6，地址 0x4004 40D4）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO1_6。	
		0x1	选择功能 CT16B1_CAP1。	
		0x2	选择功能 CT16B1_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0

表 113. PIO1_6 寄存器（PIO1_6，地址 0x4004 40D4）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	选择低模式驱动电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.53 PIO2_8 寄存器

表 114. PIO2_8 寄存器（PIO2_8，地址 0x4004 40E0）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_8。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B1_CAP0。	
		0x3	选择功能 CT32B1_MAT0。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0

表 114. PIO2_8 寄存器（PIO2_8，地址 0x4004 40E0）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.54 PIO2_9 寄存器

表 115. PIO2_9 寄存器（PIO2_9，地址 0x4004 40E4）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_9。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B1_CAP1。	
		0x3	选择功能 CT32B1_MAT1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1
8	-		保留。	0

表 115. PIO2_9 寄存器（PIO2_9，地址 0x4004 40E4）位描述（续）

位	符号	值	描述	复位值
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高级模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.55 PIO2_10 寄存器

表 116. PIO2_10 寄存器（PIO2_10，地址 0x4004 40E8）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_10。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B1_CAP2。	
		0x3	选择功能 CT32B1_MAT2。	
		0x4	保留。	
		0x5	选择功能 TXD1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	
7	-		保留。	1

表 116. PIO2_10 寄存器（PIO2_10，地址 0x4004 40E8）位描述（续）

位	符号	值	描述	复位值
8	-		保留。	0
9	DRV		驱动电流模式（正常驱动引脚）。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

6.4.56 PIO2_11 寄存器

表 117. PIO2_11 寄存器（PIO2_11，地址 0x4004 40EC）位描述

位	符号	值	描述	复位值
2:0	FUNC		选择引脚功能。	000
		0x0	选择功能 PIO2_11。	
		0x1	保留。不使用。	
		0x2	选择功能 CT32B1_CAP3。	
		0x3	选择功能 CT32B1_MAT3。	
		0x4	保留。	
		0x5	选择功能 RXD1。	
3	-		保留	0
4	MODE		选择功能模式（片内上拉电阻控制）。	1
		0	无效（未使能上拉电阻）。	
		1	已使能上拉电阻。	
5	-		保留。	0
6	INV		反转输入	0
		0	未反转输入。	
		1	已反转输入。	

表 117. PIO2_11 寄存器 (PIO2_11, 地址 0x4004 40EC) 位描述 (续)

位	符号	值	描述	复位值
7	-		保留。	1
8	-		保留。	0
9	DRV		驱动电流模式 (正常驱动引脚)。	0
		0	已选择低模式电流。	
		1	已选择高模式电流。	
10	OD		开漏模式。	0
		0	已禁能开漏模式。	
		1	已使能开漏模式。	
12:11	S_MODE		采样模式	00
		0x0	绕过输入滤波器。	
		0x1	不足 1 个滤波器时钟的输入脉冲将被抑制。	
		0x2	不足 2 个滤波器时钟的输入脉冲将被抑制。	
		0x3	不足 3 个滤波器时钟的输入脉冲将被抑制。	
15:13	CLK_DIV		选择用于输入滤波器采样时钟的外设时钟分频器。	000
		0x0	IOCONFIGCLKDIV0。	
		0x1	IOCONFIGCLKDIV1。	
		0x2	IOCONFIGCLKDIV2。	
		0x3	IOCONFIGCLKDIV3。	
		0x4	IOCONFIGCLKDIV4。	
		0x5	IOCONFIGCLKDIV5。	
		0x6	IOCONFIGCLKDIV6。	
31:16	-	-	保留。	0

7.1 简介

除电源引脚外，所有引脚都可能具有多种功能（如表 118 所示）。可以通过 IOCONFIG 模块中引脚盪肾 OCON 寄存器选择引脚功能。多路复用功能包括计数器 / 定时器输入与输出、UART 接收、发送与控制功能以及串行线调试功能（见表 120）。

每个引脚的默认功能首先与引脚复位状态一同列出。

7.2 引脚说明

表 118. LPC122x 引脚说明

符号	引脚 LQFP48	引脚 LQFP64	启动逻辑输入	类型	复位状态 [1]	描述
PIO0_0 至 PIO0_31				I/O		Port 0 — Port 0 为 32 位 I/O 端口，可单独控制每一位的方向和功能。Port 0 引脚的运作取决于通过 IOCONFIG 寄存器模块所选的功能。
PIO0_0/RTS0	15	19	[2] 是	I/O	I; PU	PIO0_0 — 通用数字输入 / 输出引脚。 RTS0 — UART0 请求发送输出。
PIO0_1/RXD0/ CT32B0_CAP0/ CT32B0_MAT0	16	20	[2] 是	I/O	I; PU	PIO0_1 — 通用数字输入 / 输出引脚。 RXD0 — UART0 的接收器输入。 CT32B0_CAP0 — 32 位定时器 0 的捕获输入，通道 0。 CT32B0_MAT0 — 32 位定时器 0 的匹配输出，通道 0。
PIO0_2/TXD0/ CT32B0_CAP1/ CT32B0_MAT1	17	21	[2] 是	I/O	I; PU	PIO0_2 — 通用数字输入 / 输出引脚。 TXD0 — UART0 的发送器输出。 CT32B0_CAP1 — 32 位定时器 0 的捕获输入，通道 1。 CT32B0_MAT1 — 32 位定时器 0 的匹配输出，通道 1。
PIO0_3/DTR0/ CT32B0_CAP2/ CT32B0_MAT2	18	22	[2] 是	I/O	I; PU	PIO0_3 — 通用数字输入 / 输出引脚。 DTR0 — UART0 数据终端就绪输出。 CT32B0_CAP2 — 32 位定时器 0 的捕获输入，通道 2。 CT32B0_MAT2 — 32 位定时器 0 的匹配输出，通道 2。
PIO0_4/DSR0/ CT32B0_CAP3/ CT32B0_MAT3	19	23	[2] 是	I/O	I; PU	PIO0_4 — 通用数字输入 / 输出引脚。 DSR0 — UART0 数据设置就绪输入。 CT32B0_CAP3 — 32 位定时器 0 的捕获输入，通道 3。 CT32B0_MAT3 — 32 位定时器 0 的匹配输出，通道 3。
PIO0_5/DCD0	20	24	[2] 是	I/O	I; PU	PIO0_5 — 通用数字输入 / 输出引脚。 DCD0 — UART0 数据载波检测输入。

表 118. LPC122x 引脚说明 (续)

符号	引脚 LQFP48	引脚 LQFP64	启动逻辑输入	类型	复位状态 [1]	描述
PIO0_6/ <u>RI0</u> / CT32B1_CAP0/ CT32B1_MAT0	21	25	[2] 是	I/O	I; PU	PIO0_6 — 通用数字输入 / 输出引脚。
				I	-	RI0 — UART0 振铃指示器输入。
				I	-	CT32B1_CAP0 — 32 位定时器 1 的捕获输入，通道 0。
				O	-	CT32B1_MAT0 — 32 位定时器 1 的匹配输出，通道 0。
PIO0_7/ <u>CTS0</u> / CT32B1_CAP1/ CT32B1_MAT1	22	26	[2] 是	I/O	I; PU	PIO0_7 — 通用数字输入 / 输出引脚。
				I	-	CTS0 — UART0 清除发送输入。
				I	-	CT32B1_CAP1 — 32 位定时器 1 的捕获输入，通道 1。
				O	-	CT32B1_MAT1 — 32 位定时器 1 的匹配输出，通道 1。
PIO0_8/ <u>RXD1</u> / CT32B1_CAP2/ CT32B1_MAT2	23	27	[2] 是	I/O	I; PU	PIO0_8 — 通用数字输入 / 输出引脚。
				I	-	RXD1 — UART1 的接收器输入。
				I	-	CT32B1_CAP2 — 32 位定时器 1 的捕获输入，通道 2。
				O	-	CT32B1_MAT2 — 32 位定时器 1 的匹配输出，通道 2。
PIO0_9/ <u>TXD1</u> / CT32B1_CAP3/ CT32B1_MAT3	24	28	[2] 是	I/O	I; PU	PIO0_9 — 通用数字输入 / 输出引脚。
				O	-	TXD1 — UART1 的发送器输出。
				I	-	CT32B1_CAP3 — 32 位定时器 1 的捕获输入，通道 3。
				O	-	CT32B1_MAT3 — 32 位定时器 1 的匹配输出，通道 3。
PIO0_10/ <u>SCL</u>	25	37	[3] 是	I/O	I; IA	PIO0_10 — 通用数字输入 / 输出引脚。
				I/O	-	SCL — I ² C 总线时钟输入 / 输出。
PIO0_11/ <u>SDA</u> / CT16B0_CAP0/ CT16B0_MAT0	26	38	[3] 是	I/O	I; IA	PIO0_11 — 通用数字输入 / 输出引脚。
				I/O	-	SDA — I ² C 总线数据输入 / 输出。
				I	-	CT16B0_CAP0 — 16 位定时器 0 的捕获输入，通道 0。
				O	-	CT16B0_MAT0 — 16 位定时器 0 的匹配输出，通道 0。
PIO0_12/ <u>CLKOUT</u> / CT16B0_CAP1/ CT16B0_MAT1	27	39	[7] 否	I/O	I; PU	PIO0_12 — 通用数字输入 / 输出引脚。复位期间，当此引脚为低电平时，启动 ISP 命令处理程序。大电流输出驱动器。
				O	-	CLKOUT — 时钟输出引脚。
				I	-	CT16B0_CAP1 — 16 位定时器 0 的捕获输入，通道 1。
				O	-	CT16B0_MAT1 — 16 位定时器 0 的匹配输出，通道 1。
<u>RESET</u> /PIO0_13	28	40	[4] 否	I	I; PU	RESET — 外部复位输入：此引脚上的低电平将会复位器件，从而导致 I/O 端口和外围设备采用其默认状态，并且处理器从地址 0 开始执行。
				I/O	-	PIO0_13 — 通用数字输入 / 输出引脚。
PIO0_14/ <u>SCK</u>	29	41	[2] 否	I/O	I; PU	PIO0_14 — 通用数字输入 / 输出引脚。
				I/O	-	SCK — SSP/SPI 的串行时钟。
PIO0_15/ <u>SSEL</u> / CT16B1_CAP0/ CT16B1_MAT0	30	42	[2] 否	I/O	I; PU	PIO0_15 — 通用数字输入 / 输出引脚。
				I/O	-	SSEL — SSP/SPI 的从机选择。
				I	-	CT16B1_CAP0 — 16 位定时器 1 的捕获输入，通道 0。
				O	-	CT16B1_MAT0 — 16 位定时器 1 的匹配输出，通道 0。

表 118. LPC122x 引脚说明 (续)

符号	引脚 LQFP48	引脚 LQFP64	启动逻辑输入	类型	复位状态 [1]	描述
PIO0_16/MISO/ CT16B1_CAP1/ CT16B1_MAT1	31	43	[2] 否	I/O	I; PU	PIO0_16 — 通用数字输入 / 输出引脚。
				I/O	-	MISO — SSP/SPI 主机输入从机输出。
				I	-	CT16B1_CAP1 — 16 位定时器 1 的捕获输入，通道 1。
				O	-	CT16B1_MAT1 — 16 位定时器 1 的匹配输出，通道 1。
PIO0_17/MOSI	32	44	[2] 否	I/O	I; PU	PIO0_17 — 通用数字输入 / 输出引脚。
				I/O	-	MOSI — SSP/SPI 主机输出从机输入。
PIO0_18/SWCLK/ CT32B0_CAP0/ CT32B0_MAT0	33	45	[2] 否	I/O	I; PU	PIO0_18 — 通用数字输入 / 输出引脚。
				I	-	SWCLK — 串行线时钟，备用位置。
				I	-	CT32B0_CAP0 — 32 位定时器 0 的捕获输入，通道 0。
				O	-	CT32B0_MAT0 — 32 位定时器 0 的匹配输出，通道 0。
PIO0_19/ACMP0_I0/ CT32B0_CAP1/ CT32B0_MAT1	4	4	[5] 否	I/O	I; PU	PIO0_19 — 通用数字输入 / 输出引脚。
				I	-	ACMP0_I0 — 比较器 0 的输入 0。
				I	-	CT32B0_CAP1 — 32 位定时器 0 的捕获输入，通道 1。
				O	-	CT32B0_MAT1 — 32 位定时器 0 的匹配输出，通道 1。
PIO0_20/ACMP0_I1/ CT32B0_CAP2/ CT32B0_MAT2	5	5	[5] 否	I/O	I; PU	PIO0_20 — 通用数字输入 / 输出引脚。
				I	-	ACMP0_I1 — 比较器 0 的输入 1。
				I	-	CT32B0_CAP2 — 32 位定时器 0 的捕获输入，通道 2。
				O	-	CT32B0_MAT2 — 32 位定时器 0 的匹配输出，通道 2。
PIO0_21/ACMP0_I2/ CT32B0_CAP3/ CT32B0_MAT3	6	6	[5] 否	I/O	I; PU	PIO0_21 — 通用数字输入 / 输出引脚。
				I	-	ACMP0_I2 — 比较器 0 的输入 2。
				I	-	CT32B0_CAP3 — 32 位定时器 0 的捕获输入，通道 3。
				O	-	CT32B0_MAT3 — 32 位定时器 0 的匹配输出，通道 3。
PIO0_22/ACMP0_I3	7	7	[5] 否	I/O	I; PU	PIO0_22 — 通用数字输入 / 输出引脚。
				I	-	ACMP0_I3 — 比较器 0 的输入 3。
PIO0_23/ ACMP1_I0/ CT32B1_CAP0/ CT32B1_MAT0	8	8	[5] 否	I/O	I; PU	PIO0_23 — 通用数字输入 / 输出引脚。
				I	-	ACMP1_I0 — 比较器 1 的输入 0。
				I	-	CT32B1_CAP0 — 32 位定时器 1 的捕获输入，通道 0。
				O	-	CT32B1_MAT0 — 32 位定时器 1 的匹配输出，通道 0。
PIO0_24/ACMP1_I1/ CT32B1_CAP1/ CT32B1_MAT1	9	9	[5] 否	I/O	I; PU	PIO0_24 — 通用数字输入 / 输出引脚。
				I	-	ACMP1_I1 — 比较器 1 的输入 1。
				I	-	CT32B1_CAP1 — 32 位定时器 1 的捕获输入，通道 1。
				O	-	CT32B1_MAT1 — 32 位定时器 1 的匹配输出，通道 1。
SWDIO/ACMP1_I2/ CT32B1_CAP2/ CT32B1_MAT2/ PIO0_25	10	10	[5] 否	I/O	I; PU	SWDIO — 串行线调试输入 / 输出，默认位置。
				I	-	ACMP1_I2 — 比较器 1 的输入 2。
				I	-	CT32B1_CAP2 — 32 位定时器 1 的捕获输入，通道 2。
				O	-	CT32B1_MAT2 — 32 位定时器 1 的匹配输出，通道 2。
				I/O	-	PIO0_25 — 通用数字输入 / 输出引脚。

表 118. LPC122x 引脚说明 (续)

符号	引脚 LQFP48	引脚 LQFP64	启动逻辑输入	类型	复位状态 [1]	描述
SWCLK/ACMP1_I3/ CT32B1_CAP3/ CT32B1_MAT3/ PIO0_26	11	11	[5] 否	I	I; PU	SWCLK — 串行线时钟，默认位置。
				I	-	ACMP1_I3 — 比较器 1 的输入 3。
				I	-	CT32B1_CAP3 — 32 位定时器 1 的捕获输入，通道 3。
				O	-	CT32B1_MAT3 — 32 位定时器 1 的匹配输出，通道 3。
				I/O		PIO0_26 — 通用数字输入 / 输出引脚。
PIO0_27/ACMP0_O	12	12	[7] 否	I/O	I; PU	PIO0_27 — 通用数字输入 / 输出引脚 (大电流输出驱动器)。
				O	-	ACMP0_O — 比较器 0 的输出。
PIO0_28/ACMP1_O/ CT16B0_CAP0/ CT16B0_MAT0	13	17	[7] 否	I/O	I; PU	PIO0_28 — 通用数字输入 / 输出引脚 (大电流输出驱动器)。
				O	-	ACMP1_O — 比较器 1 的输出。
				I	-	CT16B0_CAP0 — 16 位定时器 0 的捕获输入，通道 0。
				O	-	CT16B0_MAT0 — 16 位定时器 0 的匹配输出，通道 0。
PIO0_29/ROSC/ CT16B0_CAP1/ CT16B0_MAT1	14	18	[7] 否	I/O	I; PU	PIO0_29 — 通用数字输入 / 输出引脚 (大电流输出驱动器)。
				I/O	-	ROSC — 针对 555 定时器应用的张弛振荡器。
				I	-	CT16B0_CAP1 — 16 位定时器 0 的捕获输入，通道 1。
				O	-	CT16B0_MAT1 — 16 位定时器 0 的匹配输出，通道 1。
R/PIO0_30/AD0	34	46	[5] 否	I	I; PU	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	-	PIO0_30 — 通用数字输入 / 输出引脚。
				I	-	AD0 — A/D 转换器，输入 0。
R/PIO0_31/AD1	35	47	[5] 否	I	I; PU	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	-	PIO0_31 — 通用数字输入 / 输出引脚。
				I	-	AD1 — A/D 转换器，输入 1。
PIO1_0 至 PIO1_6				I/O		Port 1 — Port 1 为 32 位 I/O 端口，可单独控制每一位的方向和功能。Port 1 引脚的运作取决于通过 IOCONFIG 寄存器模块所选的功能。引脚 PIO1_7 至 PIO1_31 不可用。
R/PIO1_0/AD2	36	48	[5] 否	O	I; PU	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	-	PIO1_0 — 通用数字输入 / 输出引脚。
				I	-	AD2 — A/D 转换器，输入 2。
R/PIO1_1/AD3	37	49	[5] 否	I	I; PU	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	-	PIO1_1 — 通用数字输入 / 输出引脚。
				I	-	AD3 — A/D 转换器，输入 3。
PIO1_2/SWDIO/AD4	38	50	[5] 否	I/O	I; PU	PIO1_2 — 通用数字输入 / 输出引脚。
				I/O	-	SWDIO — 串行线调试输入 / 输出，备用位置。
				I	-	AD4 — A/D 转换器，输入 4。
PIO1_3/AD5/WAKEUP	39	51	[6] 否	I/O	I; PU	PIO1_3 — 通用数字输入 / 输出引脚。
				I	-	AD5 — A/D 转换器，输入 5。
				I	-	WAKEUP — 深度睡眠模式唤醒引脚。

表 118. LPC122x 引脚说明 (续)

符号	引脚 LQFP48	引脚 LQFP64	启动逻辑输入	类型	复位状态 [1]	描述
PIO1_4/AD6	40	52 [5]	否	I/O	I; PU	PIO1_4 — 通用数字输入 / 输出引脚。 AD6 — A/D 转换器, 输入 6。
PIO1_5/AD7/ CT16B1_CAP0/ CT16B1_MAT0	41	53 [5]	否	I/O	I; PU	PIO1_5 — 通用数字输入 / 输出引脚。 AD7 — A/D 转换器, 输入 7。 CT16B1_CAP0 — 16 位定时器 1 的捕获输入, 通道 0。 CT16B1_MAT0 — 16 位定时器 1 的匹配输出, 通道 0。
PIO1_6/ CT16B1_CAP1/ CT16B1_MAT1	42	54 [2]	否	I/O	I; PU	PIO1_6 — 通用数字输入 / 输出引脚。 CT16B1_CAP1 — 16 位定时器 1 的捕获输入, 通道 1。 CT16B1_MAT1 — 16 位定时器 1 的匹配输出, 通道 1。
PIO2_0 至 PIO2_15				I/O		Port 2 — Port 2 为 32 位 I/O 端口, 可单独控制每一位的方向和功能。Port 2 引脚的运作取决于通过 IOCONFIG 寄存器模块所选的功能。引脚 PIO2_16 至 PIO2_31 不可用。
PIO2_0/ CT16B0_CAP0/ CT16B0_MAT0/ RTS0	-	29 [2]	否	I/O	I; PU	PIO2_0 — 通用数字输入 / 输出引脚。 CT16B0_CAP0 — 16 位定时器 0 的捕获输入, 通道 0。 CT16B0_MAT0 — 16 位定时器 0 的匹配输出, 通道 0。 RTS0 — UART0 请求发送输出。
PIO2_1/ CT16B0_CAP1/ CT16B0_MAT1/RXD0	-	30 [2]	否	I/O	I; PU	PIO2_1 — 通用数字输入 / 输出引脚。 CT16B0_CAP1 — 16 位定时器 0 的捕获输入, 通道 1。 CT16B0_MAT1 — 16 位定时器 0 的匹配输出, 通道 1。 RXD0 — UART0 的接收器输入。
PIO2_2/ CT16B1_CAP0/ CT16B1_MAT0/TXD0	-	31 [2]	否	I/O	I; PU	PIO2_2 — 通用数字输入 / 输出引脚。 CT16B1_CAP0 — 16 位定时器 1 的捕获输入, 通道 0。 CT16B1_MAT0 — 16 位定时器 1 的匹配输出, 通道 0。 TXD0 — UART0 的发送器输出。
PIO2_3/ CT16B1_CAP1/ CT16B1_MAT1/DTR0	-	32 [2]	否	I/O	I; PU	PIO2_3 — 通用数字输入 / 输出引脚。 CT16B1_CAP1 — 16 位定时器 1 的捕获输入, 通道 1。 CT16B1_MAT1 — 16 位定时器 1 的匹配输出, 通道 1。 DTR0 — UART0 数据终端就绪输出。
PIO2_4/ CT32B0_CAP0/ CT32B0_MAT0/CTS0	-	33 [2]	否	I/O	I; PU	PIO2_4 — 通用数字输入 / 输出引脚。 CT32B0_CAP0 — 32 位定时器 0 的捕获输入, 通道 0。 CT32B0_MAT0 — 32 位定时器 0 的匹配输出, 通道 0。 CTS0 — UART0 清除发送输入。
PIO2_5/ CT32B0_CAP1/ CT32B0_MAT1/RI0	-	34 [2]	否	I/O	I; PU	PIO2_5 — 通用数字输入 / 输出引脚。 CT32B0_CAP1 — 32 位定时器 0 的捕获输入, 通道 1。 CT32B0_MAT1 — 32 位定时器 0 的匹配输出, 通道 1。 RI0 — UART0 振铃指示器输入。

表 118. LPC122x 引脚说明 (续)

符号	引脚 LQFP48	引脚 LQFP64	启动逻辑输入	类型	复位状态 [1]	描述
PIO2_6/ CT32B0_CAP2/ CT32B0_MAT2/DCD0	-	35 [2]	否	I/O	I; PU	PIO2_6 — 通用数字输入 / 输出引脚。 CT32B0_CAP2 — 32 位定时器 0 的捕获输入, 通道 2。 CT32B0_MAT2 — 32 位定时器 0 的匹配输出, 通道 2。 DCD0 — UART0 数据载波检测输入。
PIO2_7/ CT32B0_CAP3/ CT32B0_MAT3/DSR0	-	36 [2]	否	I/O	I; PU	PIO2_7 — 通用数字输入 / 输出引脚。 CT32B0_CAP3 — 32 位定时器 0 的捕获输入, 通道 3。 CT32B0_MAT3 — 32 位定时器 0 的匹配输出, 通道 3。 DSR0 — UART0 数据设置就绪输入。
PIO2_8/ CT32B1_CAP0/ CT32B1_MAT0	-	59 [2]	否	I/O	I; PU	PIO2_8 — 通用数字输入 / 输出引脚。 CT32B1_CAP0 — 32 位定时器 1 的捕获输入, 通道 0。 CT32B1_MAT0 — 32 位定时器 1 的匹配输出, 通道 0。
PIO2_9/ CT32B1_CAP1/ CT32B1_MAT1	-	60 [2]	否	I/O	I; PU	PIO2_9 — 通用数字输入 / 输出引脚。 CT32B1_CAP1 — 32 位定时器 1 的捕获输入, 通道 1。 CT32B1_MAT1 — 32 位定时器 1 的匹配输出, 通道 1。
PIO2_10/ CT32B1_CAP2/ CT32B1_MAT2/TXD1	-	61 [2]	否	I/O	I; PU	PIO2_10 — 通用数字输入 / 输出引脚。 CT32B1_CAP2 — 32 位定时器 1 的捕获输入, 通道 2。 CT32B1_MAT2 — 32 位定时器 1 的匹配输出, 通道 2。 TXD1 — UART1 的发送器输出。
PIO2_11/ CT32B1_CAP3/ CT32B1_MAT3/RXD1	-	62 [2]	否	I/O	I; PU	PIO2_11 — 通用数字输入 / 输出引脚。 CT32B1_CAP3 — 32 位定时器 1 的捕获输入, 通道 3。 CT32B1_MAT3 — 32 位定时器 1 的匹配输出, 通道 3。 RXD1 — UART1 的接收器输入。
PIO2_12/RXD1	-	13 [2]	否	I/O	I; PU	PIO2_12 — 通用数字输入 / 输出引脚。 RXD1 — UART1 的接收器输入。
PIO2_13/TXD1	-	14 [2]	否	I/O	I; PU	PIO2_13 — 通用数字输入 / 输出引脚。 TXD1 — UART1 的发送器输出。
PIO2_14	-	15 [2]	否	I/O	I; PU	PIO2_14 — 通用数字输入 / 输出引脚。
PIO2_15	-	16 [2]	否	I/O	I; PU	PIO2_15 — 通用数字输入 / 输出引脚。
RTCXIN	46	58	-	I	-	输入至 32 kHz 振荡器电路。
RTCXOUT	45	57	-	O	-	从 32 kHz 振荡放大器输出。
XTALIN	1	1	-	I	-	输入至系统振荡器电路和内部时钟发生器电路。
XTALOUT	2	2	-	O	-	从系统振荡放大器输出。
VREF_CMP	3	3	-	I	-	比较器的基准电压。
V _{DD(I/O)}	47	63	-	I	-	输入 / 输出电源电压。
V _{DD(3V3)}	44	56	-	I	-	供给内部调节器和 ADC 的 3.3 V 电源电压。也用作 ADC 基准电压。
V _{SSIO}	48	64	-	I	-	地线。
V _{SS}	43	55	-	I	-	地线。

- [1] 复位后默认功能的引脚状态：I= 输入；O= 输出；PU= 启用内部上拉电阻；IA= 非工作，未启用上拉电阻 / 下拉电阻。
- [2] 3.3 V 容压，数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [3] I²C 总线引脚；5 V 容压；开漏；默认值：无上拉电阻 / 下拉电阻；无迟滞。
- [4] 3.3 V 容压，带 RESET 功能的数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。在深度掉电模式下，需要在该引脚上安装一个外部上拉电阻。
- [5] 3.3 V 容压，带模拟功能的数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [6] 3.3 V 容压，带模拟功能和 WAKEUP（唤醒）功能的数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [7] 3.3 V 容压，高驱动数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。

表 119. LPC12D27 LQFP100 引脚说明

符号	引脚	启动 逻辑 输入	复位 状态 [1]	类型	描述
微控制器引脚					
PIO0_0 至 PIO0_31				I/O	Port 0 — Port 0 为 32 位 I/O 端口，可单独控制每一位的方向和功能。Port 0 引脚的运作取决于通过 IOCONFIG 寄存器模块所选的功能。
PIO0_0/RTS0	6 ^[2]	是	I；PU	I/O	PIO0_0 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				O	RTS0 — UART0 请求发送输出。
PIO0_1/RXD0/ CT32B0_CAP0/ CT32B0_MAT0	7 ^[2]	是	I；PU	I/O	PIO0_1 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I	RXD0 — UART0 的接收器输入。
				I	CT32B0_CAP0 — 32 位定时器 0 的捕获输入，通道 0。
				O	CT32B0_MAT0 — 32 位定时器 0 的匹配输出，通道 0。
PIO0_2/TXD0/ CT32B0_CAP1/ CT32B0_MAT1	8 ^[2]	是	I；PU	I/O	PIO0_2 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				O	TXD0 — UART0 的发送器输出。
				I	CT32B0_CAP1 — 32 位定时器 0 的捕获输入，通道 1。
				O	CT32B0_MAT1 — 32 位定时器 0 的匹配输出，通道 1。
PIO0_3/DTR0/ CT32B0_CAP2/ CT32B0_MAT2	9 ^[2]	是	I；PU	I/O	PIO0_3 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				O	DTR0 — UART0 数据终端就绪输出。
				I	CT32B0_CAP2 — 32 位定时器 0 的捕获输入，通道 2。
				O	CT32B0_MAT2 — 32 位定时器 0 的匹配输出，通道 2。
PIO0_4/ CT32B0_CAP3/ CT32B0_MAT3	10 ^[2]	是	I；PU	I/O	PIO0_4 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I	DSR0 — UART0 数据设置就绪输入。
				I	CT32B0_CAP3 — 32 位定时器 0 的捕获输入，通道 3。
				O	CT32B0_MAT3 — 32 位定时器 0 的匹配输出，通道 3。
PIO0_5/DCD0	11 ^[2]	是	I；PU	I/O	PIO0_5 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I	DCD0 — UART0 数据载波检测输入。

表 119. LPC12D27 LQFP100 引脚说明 (续)

符号	引脚	启动 逻辑 输入	复位 状态 [1]	类型	描述
PIO0_6/ CT32B1_CAP0/ CT32B1_MAT0	12 [2]	是	I; PU	I/O	PIO0_6 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I	RI0 — UART0 振铃指示器输入。
				I	CT32B1_CAP0 — 32 位定时器 1 的捕获输入，通道 0。
				O	CT32B1_MAT0 — 32 位定时器 1 的匹配输出，通道 0。
PIO0_7/ CT32B1_CAP1/ CT32B1_MAT1	13 [2]	是	I; PU	I/O	PIO0_7 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I	CTS0 — UART0 清除发送输入。
				I	CT32B1_CAP1 — 32 位定时器 1 的捕获输入，通道 1。
				O	CT32B1_MAT1 — 32 位定时器 1 的匹配输出，通道 1。
PIO0_8/ CT32B1_CAP2/ CT32B1_MAT2	14 [2]	是	I; PU	I/O	PIO0_8 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I	RXD1 — UART1 的接收器输入。
				I	CT32B1_CAP2 — 32 位定时器 1 的捕获输入，通道 2。
				O	CT32B1_MAT2 — 32 位定时器 1 的匹配输出，通道 2。
PIO0_9/ CT32B1_CAP3/ CT32B1_MAT3	15 [2]	是	I; PU	I/O	PIO0_9 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				O	TXD1 — UART1 的发送器输出。
				I	CT32B1_CAP3 — 32 位定时器 1 的捕获输入，通道 3。
				O	CT32B1_MAT3 — 32 位定时器 1 的匹配输出，通道 3。
PIO0_10/SCL	17 [3]	是	I; IA	I/O	PIO0_10 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I/O	SCL — I ² C 总线时钟输入 / 输出。
PIO0_11/ CT16B0_CAP0/ CT16B0_MAT0	18 [3]	是	I; IA	I/O	PIO0_11 — 通用数字输入 / 输出引脚。还作为深度睡眠模式唤醒引脚。
				I/O	SDA — I ² C 总线数据输入 / 输出。
				I	CT16B0_CAP0 — 16 位定时器 0 的捕获输入，通道 0。
				O	CT16B0_MAT0 — 16 位定时器 0 的匹配输出，通道 0。
PIO0_12/ CT16B0_CAP1/ CT16B0_MAT1	19 [7]	否	I; PU	I/O	PIO0_12 — 通用数字输入 / 输出引脚。复位期间，当此引脚为低电平时，启动 ISP 命令处理程序。大电流输出驱动器。
				O	CLKOUT — 时钟输出引脚。
				I	CT16B0_CAP1 — 16 位定时器 0 的捕获输入，通道 0。
				O	CT16B0_MAT1 — 16 位定时器 0 的匹配输出，通道 1。
RESET/PIO0_13	20 [4]	否	I; PU	I	RESET — 外部复位输入：此引脚上的低电平将会复位器件，从而导致 I/O 端口和外围设备采用其默认状态，并且处理器从地址 0 开始执行。
				I/O	PIO0_13 — 通用数字输入 / 输出引脚。
PIO0_14/SCK	21 [2]	否	I; PU	I/O	PIO0_14 — 通用数字输入 / 输出引脚。
				I/O	SCK — SSP 的串行时钟。

表 119. LPC12D27 LQFP100 引脚说明 (续)

符号	引脚	启动 逻辑 输入	复位 状态 [1]	类型	描述
PIO0_15/SSEL/ CT16B1_CAP0/ CT16B1_MAT0	22 [2]	否	I; PU	I/O	PIO0_15 — 通用数字输入 / 输出引脚。
				I/O	SSEL — SSP 的从机选择。
				I	CT16B1_CAP0 — 16 位定时器 1 的捕获输入, 通道 0。
				O	CT16B1_MAT0 — 16 位定时器 1 的匹配输出, 通道 0。
PIO0_16/MISO/ CT16B1_CAP1/ CT16B1_MAT1	23 [2]	否	I; PU	I/O	PIO0_16 — 通用数字输入 / 输出引脚。
				I/O	MISO — SSP 主机输入从机输出。
				I	CT16B1_CAP1 — 16 位定时器 1 的捕获输入, 通道 1。
				O	CT16B1_MAT1 — 16 位定时器 1 的匹配输出, 通道 1。
PIO0_17/MOSI	24 [2]	否	I; PU	I/O	PIO0_17 — 通用数字输入 / 输出引脚。
				I/O	MOSI — SSP 主机输出从机输入。
PIO0_18/SWCLK/ CT32B0_CAP0/ CT32B0_MAT0	25 [2]	否	I; PU	I/O	PIO0_18 — 通用数字输入 / 输出引脚。
				I	SWCLK — 串行线时钟, 备用位置。
				I	CT32B0_CAP0 — 32 位定时器 0 的捕获输入, 通道 0。
				O	CT32B0_MAT0 — 32 位定时器 0 的匹配输出, 通道 0。
PIO0_19/ACMP0_I0/ CT32B0_CAP1/ CT32B0_MAT1	95 [5]	否	I; PU	I/O	PIO0_19 — 通用数字输入 / 输出引脚。
				I	ACMP0_I0 — 比较器 0 的输入 0。
				I	CT32B0_CAP1 — 32 位定时器 0 的捕获输入, 通道 1。
				O	CT32B0_MAT1 — 32 位定时器 0 的匹配输出, 通道 1。
PIO0_20/ACMP0_I1/ CT32B0_CAP2/ CT32B0_MAT2	96 [5]	否	I; PU	I/O	PIO0_20 — 通用数字输入 / 输出引脚。
				I	ACMP0_I1 — 比较器 0 的输入 1。
				I	CT32B0_CAP2 — 32 位定时器 0 的捕获输入, 通道 2。
				O	CT32B0_MAT2 — 32 位定时器 0 的匹配输出, 通道 2。
PIO0_21/ACMP0_I2/ CT32B0_CAP3/ CT32B0_MAT3	97 [5]	否	I; PU	I/O	PIO0_21 — 通用数字输入 / 输出引脚。
				I	ACMP0_I2 — 比较器 0 的输入 2。
				I	CT32B0_CAP3 — 32 位定时器 0 的捕获输入, 通道 3。
				O	CT32B0_MAT3 — 32 位定时器 0 的匹配输出, 通道 3。
PIO0_22/ACMP0_I3	98 [5]	否	I; PU	I/O	PIO0_22 — 通用数字输入 / 输出引脚。
				I	ACMP0_I3 — 比较器 0 的输入 3。
PIO0_23/ ACMP1_I0/ CT32B1_CAP0/ CT32B1_MAT0	99 [5]	否	I; PU	I/O	PIO0_23 — 通用数字输入 / 输出引脚。
				I	ACMP1_I0 — 比较器 1 的输入 0。
				I	CT32B1_CAP0 — 32 位定时器 1 的捕获输入, 通道 0。
				O	CT32B1_MAT0 — 32 位定时器 1 的匹配输出, 通道 0。
PIO0_24/ACMP1_I1/ CT32B1_CAP1/ CT32B1_MAT1	100 [5]	否	I; PU	I/O	PIO0_24 — 通用数字输入 / 输出引脚。
				I	ACMP1_I1 — 比较器 1 的输入 1。
				I	CT32B1_CAP1 — 32 位定时器 1 的捕获输入, 通道 1。
				O	CT32B1_MAT1 — 32 位定时器 1 的匹配输出, 通道 1。

表 119. LPC12D27 LQFP100 引脚说明 (续)

符号	引脚	启动 逻辑 输入	复位 状态 [1]	类型	描述
SWDIO/ACMP1_I2/ CT32B1_CAP2/ CT32B1_MAT2/PIO0_25	1 [5]	否	I; PU	I/O	SWDIO — 串行线调试输入 / 输出，默认位置。
				I	ACMP1_I2 — 比较器 1 的输入 2。
				I	CT32B1_CAP2 — 32 位定时器 1 的捕获输入，通道 2。
				O	CT32B1_MAT2 — 32 位定时器 1 的匹配输出，通道 2。
SWCLK/ ACMP1_I3/ CT32B1_CAP3/ CT32B1_MAT3/PIO0_26	2 [5]	否	I; PU	I/O	PIO0_25 — 通用数字输入 / 输出引脚。
				I	SWCLK — 串行线时钟，默认位置。
				I	ACMP1_I3 — 比较器 1 的输入 3。
				I	CT32B1_CAP3 — 32 位定时器 1 的捕获输入，通道 3。
PIO0_27/ACMP0_O	3 [7]	否	I; PU	O	CT32B1_MAT3 — 32 位定时器 1 的匹配输出，通道 3。
				I/O	PIO0_26 — 通用数字输入 / 输出引脚。
				I/O	PIO0_27 — 通用数字输入 / 输出引脚（大电流输出驱动器）。
				O	ACMP0_O — 比较器 0 的输出。
PIO0_28/ACMP1_O/ CT16B0_CAP0/ CT16B0_MAT0	4 [7]	否	I; PU	I/O	PIO0_28 — 通用数字输入 / 输出引脚（大电流输出驱动器）。
				O	ACMPC1_O — 比较器 1 的输出。
				I	CT16B0_CAP0 — 16 位定时器 0 的捕获输入，通道 0。
				O	CT16B0_MAT0 — 16 位定时器 0 的匹配输出，通道 0。
PIO0_29/ROSC/ CT16B0_CAP1/ CT16B0_MAT1	5 [7]	否	I; PU	I/O	PIO0_29 — 通用数字输入 / 输出引脚（大电流输出驱动器）。
				I/O	ROSC — 针对 555 定时器应用的张弛振荡器。
				I	CT16B0_CAP1 — 16 位定时器 0 的捕获输入，通道 1。
				O	CT16B0_MAT1 — 16 位定时器 0 的匹配输出，通道 1。
R/PIO0_30/AD0	26 [5]	否	I; PU	I	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	PIO0_30 — 通用数字输入 / 输出引脚。
				I	AD0 — A/D 转换器，输入 0。
R/PIO0_31/AD1	27 [5]	否	I; PU	I	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	PIO0_31 — 通用数字输入 / 输出引脚。
				I	AD1 — A/D 转换器，输入 1。
PIO1_0 至 PIO1_6				I/O	Port 1 — Port 1 为 32 位 I/O 端口，可单独控制每一位的方向和功能。Port 1 引脚的运作取决于通过 IOCONFIG 寄存器模块所选的功能。引脚 PIO1_7 至 PIO1_31 不可用。
R/PIO1_0/AD2	28 [5]	否	I; PU	O	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	PIO1_0 — 通用数字输入 / 输出引脚。
				I	AD2 — A/D 转换器，输入 2。
R/PIO1_1/AD3	80 [5]	否	I; PU	I	R — 保留。在 IOCONFIG 模块中配置用于备用功能。
				I/O	PIO1_1 — 通用数字输入 / 输出引脚。
				I	AD3 — A/D 转换器，输入 3。
PIO1_2/SWDIO/AD4	81 [5]	否	I; PU	I/O	PIO1_2 — 通用数字输入 / 输出引脚。
				I/O	SWDIO — 串行线调试输入 / 输出，备用位置。
				I	AD4 — A/D 转换器，输入 4。

表 119. LPC12D27 LQFP100 引脚说明 (续)

符号	引脚	启动 逻辑 输入	复位 状态 [1]	类型	描述
PIO1_3/AD5/WAKEUP	82 [6]	否	I; PU	I/O	PIO1_3 — 通用数字输入 / 输出引脚。
				I	AD5 — A/D 转换器, 输入 5。
				I	WAKEUP — 深度掉电模式唤醒引脚。
PIO1_4/AD6	83 [5]	否	I; PU	I/O	PIO1_4 — 通用数字输入 / 输出引脚。
				I	AD6 — A/D 转换器, 输入 6。
PIO1_5/AD7/ CT16B1_CAP0/ CT16B1_MAT0	84 [5]	否	I; PU	I/O	PIO1_5 — 通用数字输入 / 输出引脚。
				I	AD7 — A/D 转换器, 输入 7。
				I	CT16B1_CAP0 — 16 位定时器 1 的捕获输入, 通道 0。
				O	CT16B1_MAT0 — 16 位定时器 1 的匹配输出, 通道 0。
PIO1_6/CT16B1_CAP1/ CT16B1_MAT1	85 [2]	否	I; PU	I/O	PIO1_6 — 通用数字输入 / 输出引脚。
				I	CT16B1_CAP1 — 16 位定时器 1 的捕获输入, 通道 1。
				O	CT16B1_MAT1 — 16 位定时器 1 的匹配输出, 通道 1。
PIO2_0				I/O	Port 2 — Port 2 为 32 位 I/O 端口, 可单独控制每一位的方向和功能。Port 2 引脚的运作取决于通过 IOCONFIG 寄存器模块所选的功能。引脚 PIO2_1 至 PIO2_31 不可用。
PIO2_0/CT16B0_CAP0/ CT16B0_MAT0	16 [2]	否	I; PU	I/O	PIO2_0 — 通用数字输入 / 输出引脚。
				I	CT16B0_CAP0 — 16 位定时器 0 的捕获输入, 通道 0。
				O	CT16B0_MAT0 — 16 位定时器 0 的匹配输出, 通道 0。
RTCXIN	89	-	-	-	输入至 32 kHz 振荡器电路。
RTCXOUT	88	-	-	-	从 32 kHz 振荡放大器输出。
XTALIN	92	-	-	-	输入至系统振荡器电路和内部时钟发生器电路。
XTALOUT	93	-	-	-	从系统振荡放大器输出。
VREF_CMP	94	-	-	-	比较器的基准电压。
V _{DD(I/O)}	90	-	-	-	输入 / 输出电源电压。
V _{DD(3V3)}	87	-	-	-	供给内部调节器和 ADC 的 3.3 V 电源电压。也用作 ADC 基准电压。
V _{SSIO}	91	-	-	-	地线。
V _{SS}	86	-	-	-	地线。
LCD 显示器引脚					
S0	46	-	-	O	LCD 段输出。
S1	47	-	-	O	LCD 段输出。
S2	48	-	-	O	LCD 段输出。
S3	49	-	-	O	LCD 段输出。
S4	50	-	-	O	LCD 段输出。
S5	51	-	-	O	LCD 段输出。
S6	52	-	-	O	LCD 段输出。
S7	53	-	-	O	LCD 段输出。
S8	54	-	-	O	LCD 段输出。
S9	55	-	-	O	LCD 段输出。
S10	56	-	-	O	LCD 段输出。
S11	57	-	-	O	LCD 段输出。

表 119. LPC12D27 LQFP100 引脚说明 (续)

符号	引脚	启动 逻辑 输入	复位 状态 [1]	类型	描述
S12	58	-	-	O	LCD 段输出。
S13	59	-	-	O	LCD 段输出。
S14	60	-	-	O	LCD 段输出。
S15	61	-	-	O	LCD 段输出。
S16	62	-	-	O	LCD 段输出。
S17	63	-	-	O	LCD 段输出。
S18	64	-	-	O	LCD 段输出。
S19	65	-	-	O	LCD 段输出。
S20	66	-	-	O	LCD 段输出。
S21	67	-	-	O	LCD 段输出。
S22	68	-	-	O	LCD 段输出。
S23	69	-	-	O	LCD 段输出。
S24	70	-	-	O	LCD 段输出。
S25	71	-	-	O	LCD 段输出。
S26	72	-	-	O	LCD 段输出。
S27	73	-	-	O	LCD 段输出。
S28	74	-	-	O	LCD 段输出。
S29	75	-	-	O	LCD 段输出。
S30	76	-	-	O	LCD 段输出。
S31	77	-	-	O	LCD 段输出。
S32	78	-	-	O	LCD 段输出。
S33	79	-	-	O	LCD 段输出。
S34	29	-	-	O	LCD 段输出。
S35	30	-	-	O	LCD 段输出。
S36	31	-	-	O	LCD 段输出。
S37	32	-	-	O	LCD 段输出。
S38	33	-	-	O	LCD 段输出。
S39	34	-	-	O	LCD 段输出。
BP0	42	-	-	O	LCD 背板输出。
BP1	44	-	-	O	LCD 背板输出。
BP2	43	-	-	O	LCD 背板输出。
BP3	45	-	-	O	LCD 背板输出。
LCD_SDA	35	-	-	I/O	I ² C 总线串行数据输入 / 输出。
LCD_SCL	36	-	-	I/O	I ² C 总线串行时钟输入。
SYNC	37	-	-	I/O	级联同步输入 / 输出。
CLK	38	-	-	I/O	外部时钟输入 / 输出。
V _{DD}	39	-	-	-	1.8 V 至 5.5 V 电源：PCF8576D 的电源电压。
V _{SS(LCD)}	40	-	-	-	LCD 地线。
V _{LCD}	41	-	-	-	LCD 电源：LCD 电压。

- [1] 复位后默认功能的引脚状态：I= 输入；O= 输出；PU= 启用内部上拉电阻；IA= 非工作，未启用上拉电阻 / 下拉电阻。
- [2] 数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [3] I²C 总线引脚；5 V 容压；开漏；默认值：无上拉电阻 / 下拉电阻；无迟滞。
- [4] 带 RESET 功能的数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [5] 带模拟功能的数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [6] 带模拟功能和 WAKEUP（唤醒）功能的数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。
- [7] 高驱动数字 I/O 引脚；默认值：启用上拉电阻，无迟滞。

7.3 引脚复用

为了启用外设功能，应找到相应的端口引脚或者选择复用功能中的一个引脚，通过编程该端口引脚的 IOCONFIG 寄存器启用某项功能。主要的 SWD 和 RESET 功能是引脚复位后的默认功能，所有其他数字引脚的默认功能是 GPIO。

表 120. 引脚复用

外设	功能	类型	端口引脚:		
模拟比较器	ROSC	I/O	PIO0_29	-	-
	ACMP0_I0	I	PIO0_19	-	-
	ACMP0_I1	I	PIO0_20	-	-
	ACMP0_I2	I	PIO0_21	-	-
	ACMP0_I3	I	PIO0_22	-	-
	ACMP0_O	O	PIO0_27	-	-
	ACMP1_I0	I	PIO0_23	-	-
	ACMP1_I1	I	PIO0_24	-	-
	ACMP1_I2	I	PIO0_25	-	-
	ACMP1_I3	I	PIO0_26	-	-
ADC	ACMP1_O	O	PIO0_28	-	-
	AD0	I	PIO0_30	-	-
	AD1	I	PIO0_31	-	-
	AD2	I	PIO1_0	-	-
	AD3	I	PIO1_1	-	-
	AD4	I	PIO1_2	-	-
	AD5	I	PIO1_3	-	-
	AD6	I	PIO1_4	-	-
CT16B0	AD7	I	PIO1_5	-	-
	CT16B0_CAP0	I	PIO0_11	PIO0_28	PIO2_0
	CT16B0_CAP1	I	PIO0_12	PIO0_29	PIO2_1
	CT16B0_MAT0	O	PIO0_11	PIO0_28	PIO2_0
CT16B1	CT16B0_MAT1	O	PIO0_12	PIO0_29	PIO2_1
	CT16B1_CAP0	I	PIO0_15	PIO1_5	PIO2_2
	CT16B1_CAP1	I	PIO0_16	PIO1_6	PIO2_3
	CT16B1_MAT0	O	PIO0_15	PIO1_5	PIO2_2
	CT16B1_MAT1	O	PIO0_16	PIO1_6	PIO2_3

表 120. 引脚复用

外设	功能	类型	端口引脚:		
CT32B0	CT32B0_CAP0	I	PIO0_1	PIO0_18	PIO2_4
	CT32B0_CAP1	I	PIO0_2	PIO0_19	PIO2_5
	CT32B0_CAP2	I	PIO0_3	PIO0_20	PIO2_6
	CT32B0_CAP3	I	PIO0_4	PIO0_21	PIO2_7
	CT32B0_MAT0	O	PIO0_1	PIO0_18	PIO2_4
	CT32B0_MAT1	O	PIO0_2	PIO0_19	PIO2_5
	CT32B0_MAT2	O	PIO0_3	PIO0_20	PIO2_6
	CT32B0_MAT3	O	PIO0_4	PIO0_21	PIO2_7
CT32B1	CT32B1_CAP0	I	PIO0_6	PIO0_23	PIO2_8
	CT32B1_CAP1	I	PIO0_7	PIO0_24	PIO2_9
	CT32B1_CAP2	I	PIO0_8	PIO0_25	PIO2_10
	CT32B1_CAP3	I	PIO0_9	PIO0_26	PIO2_11
	CT32B1_MAT0	O	PIO0_6	PIO0_23	PIO2_8
	CT32B1_MAT1	O	PIO0_7	PIO0_24	PIO2_9
	CT32B1_MAT2	O	PIO0_8	PIO0_25	PIO2_10
	CT32B1_MAT3	O	PIO0_9	PIO0_26	PIO2_11
UART0	RXD0	I	PIO0_1	PIO2_1	-
	TXD0	O	PIO0_2	PIO2_2	-
	CTS0	I	PIO0_7	PIO2_4	-
	DCD0	I	PIO0_5	PIO2_6	-
	DSR0	I	PIO0_4	PIO2_7	-
	DTR0	O	PIO0_3	PIO2_3	-
	RI0	I	PIO0_6	PIO2_5	-
	RTS0	O	PIO0_0	PIO2_0	-
UART1	RXD1	I	PIO0_8	PIO2_11	PIO2_12
	TXD1	O	PIO0_9	PIO2_10	PIO2_13
SSP/SPI	SCK	I/O	PIO0_14	-	-
	MISO	I/O	PIO0_16	-	-
	MOSI	I/O	PIO0_17	-	-
	SSEL	I/O	PIO0_15	-	-
I2C	SCL	I/O	PIO0_10	-	-
	SDA	I/O	PIO0_11	-	-
SWD	SWCLK ^[1]	I	PIO0_18	PIO0_26	-
	SWDIO ^[1]	I/O	PIO0_25	PIO1_2	-
复位	RESET	I	PIO0_13	-	-
Clockout 引脚	CLKOUT	O	PIO0_12	-	-

[1] 复位后，默认具有 SWD 功能的引脚是 PIO0_26 和 PIO0_25。

8.1 本章导读

每个引脚都在其中一个 GPIO 寄存器中分配有一个位。用于不可用引脚的 GPIO 寄存器被保留（见[表 121](#)）。GPIO 端口 2 寄存器仅在 64 引脚封装上提供。

表 121. 可用的 GPIO 引脚 / 端口

端口	引脚	使用的 GPIO 寄存器位	LQFP48	LQFP64	LQFP100 ^[1]
GPIO0	PIO0_0 到 PIO0_31	31:0	是	是	是
GPIO1	PIO1_0 到 PIO1_6	6:0	是	是	是
GPIO2	PIO2_0 到 PIO2_15	15:0	否	是	是

[1] 用于器件 LPC12D27。

8.2 简介

8.2.1 特性

- 数字端口可以由软件配置为输入 / 输出。
- 端口引脚的读写数据操作是可屏蔽的。
- 位级别的设置和清零寄存器允许使用单个指令集或清除一个端口中的任意引脚数。
- 位级别倒相寄存器允许对一个端口中任意数量引脚的输出倒相。
- 每个单独端口引脚均可用作外部中断输入。
- 可在单个下降沿或上升沿以及两个边沿上同时配置中断。
- 可对单独的中断级别编程。
- 复位后所有 GPIO 引脚都被配置为输入（上拉电阻器使能，见[章节 6.3.2](#)）。

8.3 寄存器描述

所有 GPIO 寄存器都为 32 位宽。MASK 寄存器可屏蔽 PIN、OUT、SET、CLR 和 NOT 寄存器上的所有操作。寄存器 DIR 到 IC 不受 MASK 寄存器中设置的位的影响。

GPIO 寄存器中的每个位代表一个 GPIO 引脚。每个端口的引脚配置决定使用或保留哪些引脚：

- 端口 0：所有 GPIO0 寄存器使用位 0 到 31。
- 端口 1：所有 GPIO1 寄存器使用位 0 到 6。位 7 到 31 保留。
- 端口 2：所有 GPIO2 寄存器使用位 0 到 15。位 16 到 31 保留。

表 122. 寄存器简介: GPIO（基址端口 0: 0x5000 0000；端口 1: 0x5001 0000；端口 2: 0x5002 0000）

名称	访问类型	地址偏移	描述	复位值
MASK	R/W	0x000	引脚值屏蔽寄存器。影响 PIN、OUT、SET、CLR 和 NOT 寄存器上的操作。	0x0000 0000
PIN	R	0x004	引脚值寄存器。	取决于配置
OUT	R/W	0x008	引脚输出值寄存器。	0x0000 0000
SET	W	0x00C	引脚输出值设置寄存器。	不适用
CLR	W	0x010	引脚输出值清除寄存器。	不适用
NOT	W	0x014	引脚输出值倒相寄存器。	0x0000 0000
DIR	R/W	0x020	数据方向寄存器。	0x0000 0000
IS	R/W	0x024	中断感应寄存器。	0x0000 0000
IBE	R/W	0x028	中断双边沿寄存器。	0x0000 0000
IEV	R/W	0x02C	中断事件寄存器。	0x0000 0000
IE	R/W	0x030	中断屏蔽寄存器。	0x0000 0000
RIS	R	0x034	原始中断状态寄存器。	0x0000 0000
MIS	R	0x038	屏蔽中断状态寄存器。	0x0000 0000
IC	W	0x03C	中断清除寄存器。	0x0000 0000
-	-	0x040	保留。	0x0000 0000

8.3.1 GPIO 屏蔽寄存器

该寄存器屏蔽对下列被屏蔽寄存器的读和 / 或写访问: PIN、OUT、SET、CLR 和 NOT。只有 MASK 寄存器中被置 0 的位才使能被屏蔽寄存器中相应位的更改或读值操作。

将任何屏蔽位置 0 允许通过对引脚的 OUT、SET、CLR 和 NOT 寄存器的写操作来更改引脚输出。引脚的当前状态可从 PIN 寄存器读取，OUT 寄存器的当前值也可被读取。

将任何屏蔽位置 1 将使对引脚的 OUT、SET、CLR 和 NOT 寄存器的写操作对引脚输出电平无效。读操作返回 0，无论引脚的电平或 OUT 寄存器的值如何。

表 123. GPIO 屏蔽寄存器 (MASK - 地址 0x5000 0000 (GPIO0)、0x5001 0000 (GPIO1)、0x5002 0000 (GPIO2)) 的位描述

位	符号	描述	复位值	访问类型
31:0	MASK	GPIO 引脚 PION_x 访问控制。 0 = 不屏蔽读 / 写。 1 = 屏蔽读 / 写。	0x0	R/W

8.3.2 GPIO 引脚值寄存器

该寄存器提供配置用于执行数字功能的端口引脚的当前逻辑状态。对该寄存器的读操作将返回引脚的逻辑值，不管引脚是配置为输入还是输出，也不管它是配置为 GPIO 还是任何其它适用的备用数字功能。例如，某个特定端口引脚可能具有 GPIO 输入、GPIO 输出以及计数器 / 定时器匹配输出和捕获输入作为可选功能。通过 PIN 寄存器，不管其配置如何，都可读出引脚的当前逻辑状态，如可读出捕获输入状态。

但有个例外：如果选择了引脚的模拟功能（如适用），则不能读取引脚状态，因为将引脚选作 ADC 输入会断开引脚的数字功能。在这种情况下，PIN 寄存器中读出的引脚值无效。

请注意：读操作可通过 MASK 位屏蔽。对被屏蔽位的读操作总是返回 0，不管引脚的当前电平如何。

表 124. GPIO 引脚值寄存器 (PIN - 地址 0x5000 0004 (GPIO0)、0x5001 0004 (GPIO1) ; 0x5002 0004 (GPIO2)) 的位描述

位	符号	描述	复位值	访问类型
31:0	PIN	GPIO 引脚 PION_x 值。 0 = 数字引脚为低电平。 1 = 数字引脚为高电平。	0x0	R

8.3.3 GPIO 引脚输出寄存器

向该寄存器写 0 或 1 将在相应端口引脚产生低电平或高电平。如果端口引脚配置为 GPIO 输出，将设为此值。对于所有其他配置（输入、非 GPIO 功能），OUT 寄存器位的值对引脚输出电平无效。写操作被 MASK 寄存器屏蔽。

读取该寄存器将返回 GPIO 输出寄存器的内容，不管数字引脚配置和方向如何。读操作被 MASK 寄存器屏蔽。

SET、CLR 和 NOT 寄存器对 OUT 寄存器执行写操作，以允许按位对单个端口引脚进行设置、清除和取反。端口输出状态只由 OUT 寄存器的内容决定。

表 125. GPIO 引脚输出寄存器 (OUT - 地址 0x5000 0008 (GPIO0)、0x5001 0008 (GPIO1)、0x5002 0008 (GPIO2)) 的位描述

位	符号	描述	复位值	访问类型
31:0	OUT	GPIO 引脚 PION_x 输出值。 0 = 写：将 GPIO 输出引脚设为低电平。读：GPIO 输出值为低电平。 1 = 写：将 GPIO 输出引脚设为高电平。读：GPIO 输出值为高电平。	0x00	R/W

8.3.4 GPIO 引脚输出设置寄存器

该寄存器用于在通过 DIR 寄存器（见[表 129](#)）配置为 GPIO 输出并通过相应 IOCONFIG 寄存器（见[表 60](#)）配置为 GPIO 的端口引脚处产生高电平输出。写 1 会将相应端口引脚设为高电平。写 0 对 GPIO 输出电平无效。如果引脚未配置为 GPIO 和输出，SET 寄存器对引脚电平无效。

该寄存器为只写寄存器。请注意：对 SET 寄存器的写操作可通过 MASK 寄存器屏蔽。

表 126. GPIO 引脚输出设置寄存器 (SET - 地址 0x5000 000C (GPIO0)、0x5001 000C (GPIO1)、0x5002 000C (GPIO2)) 的位描述

位	符号	描述	复位值	访问类型
31:0	SET	设置 GPIO 引脚 PION_x 输出值。 0 = 对 GPIO 输出电平无效。 1 = 将 GPIO 输出设为高电平。	0x00	W

8.3.5 GPIO 引脚输出清除寄存器

该寄存器用于在通过 DIR 寄存器（见[表 129](#)）配置为 GPIO 输出并通过相应 IOCONFIG 寄存器（见[表 60](#)）配置为 GPIO 的端口引脚处产生低电平输出。写 1 会将相应端口引脚设为低电平。写 0 对 GPIO 输出电平无效。如果引脚未配置为 GPIO 和输出，CLR 寄存器对引脚电平无效。

该寄存器为只写寄存器。请注意：对 CLR 寄存器的写操作可通过 MASK 寄存器屏蔽。

表 127. GPIO 引脚输出清除寄存器（CLR - 地址 0x5000 0010 (GPIO0)、0x5000 1010 (GPIO1)、0x5002 0010 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	CLEAR	清除 GPIO 引脚 PION_x 输出值。 0 = 对 GPIO 输出电平无效。 1 = 将 GPIO 输出设为低电平。	0x00	W

8.3.6 GPIO NOT 寄存器

该寄存器用于将通过 DIR 寄存器（见[表 129](#)）配置为 GPIO 输出并通过相应 IOCONFIG 寄存器（见[表 60](#)）配置为 GPIO 的端口引脚的输出电平倒相。写 1 对相应端口引脚倒相。写 0 对 GPIO 输出电平无效。如果引脚未配置为 GPIO 和输出，NOT 寄存器对引脚电平无效。

该寄存器为只写寄存器。请注意：对 NOT 寄存器的写操作可通过 MASK 寄存器屏蔽。

表 128. GPIO NOT 寄存器（NOT - 地址 0x5000 0014 (GPIO0)、0x5001 0014 (GPIO1)、0x5002 0014 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	NOT	对 GPIO 引脚 PION_x 输出值倒相。 0 = 对 GPIO 输出电平无效。 1 = GPIO 输出取与其当前值相反的值。	0x00	-

8.3.7 GPIO 数据方向寄存器

表 129. GPIO 数据方向寄存器（DIR - 地址 0x5000 0020 (GPIO0)、0x5001 0020 (GPIO1)、0x5002 0020 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	IO	选择 GPIO 引脚 PION_x 作为输入或输出。 0 = 将引脚 PION_x 配置为输入。 1 = 将引脚 PION_x 配置为输出。	0x00	R/W

8.3.8 GPIO 中断感应寄存器

表 130. GPIO 中断感应寄存器（IS - 地址 0x5000 0024 (GPIO0)、0x5001 0024 (GPIO1)、0x5002 0024 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	ISENSE	选择引脚 PION_x 上的中断为电平敏感或边沿敏感。 0 = 将引脚 PION_x 上的中断配置为边沿敏感。1 = 将引脚 PION_x 上的中断配置为电平敏感。	0x00	R/W

8.3.9 GPIO 中断双边沿感应寄存器

表 131. GPIO 中断双边沿感应寄存器（IBE - 地址 0x5000 0028 (GPIO0)、0x5001 0028 (GPIO1)、0x5002 0028 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	IBE	选择引脚 PION_x 上的中断在双边沿上触发。 0 = 引脚 PION_x 上的中断通过寄存器 IEV 控制。 1 = 引脚 PION_x 的两个边沿都会触发中断。	0x00	R/W

8.3.10 GPIO 中断事件寄存器

Table 132. GPIO 中断事件寄存器（IEV - 地址 0x5000 002C (GPIO0)、0x5001 002C (GPIO1)、0x5002 002C (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	IEV	选择引脚 PION_x 上的中断是在上升沿还是下降沿触发。 0 = 视 IS 寄存器中的设置而定，引脚 PION_x 的下降沿或低电平触发中断。 1 = 视 IS 寄存器中的设置而定，引脚 PION_x 的上升沿或高电平触发中断。	0x00	R/W

8.3.11 GPIO 中断屏蔽寄存器

如果 IE 寄存器中的位设为高，对应的引脚就会触发各自的中断和配套的 INTR 线。清除该位会禁止对应引脚的中断触发。

表 133. GPIO 中断屏蔽寄存器（IE - 地址 0x5000 0030、0x5001 0030 (GPIO1)、0x5002 0030 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	MASK	选择引脚 PION_x 上要屏蔽的中断。 0 = 屏蔽引脚 PION_x 上的中断。 1 = 不屏蔽引脚 PION_x 上的中断。	0x00	R/W

8.3.12 GPIO 原始中断状态寄存器

IRS 寄存器的位读数为高时反映了对应引脚上的原始（屏蔽之前）中断状态，表示在触发 IE 之前所有的要求都满足。位读数为 0 时表示对应的输入引脚还未启动中断。该寄存器为只读。

表 134. GPIO 原始中断状态寄存器（RIS - 地址 0x5000 0034 (GPIO0)、0x5001 0034 (GPIO1)、0x5002 0034 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	RAWST	原始中断状态。 0 = 引脚 PION_x 上无中断。 1 = PION_x 满足中断要求。	0x00	R

8.3.13 GPIO 屏蔽中断状态寄存器

MIS 寄存器中的位读为高反映了输入线触发中断的状态。读出为低则表示对应的输入引脚没有中断产生，或者中断被屏蔽。MIS 是屏蔽后的中断状态。该寄存器为只读。

表 135. GPIO 屏蔽中断状态寄存器（MIS - 地址 0x5000 0038 (GPIO0)、0x5001 0038 (GPIO1)、0x5002 0038 (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	MASK	选择引脚 PION_x 上要屏蔽的中断。 0 = 引脚 PION_x 上无中断或中断被屏蔽。 1 = PION_x 上有中断。	0x00	R

8.3.14 GPIO 中断清除寄存器

注：GPIO 和 NVIC 模块之间的同步器会造成 2 个时钟的延时。建议在清除中断边沿检测逻辑之后、退出中断服务程序之前增加 2 个 NOP。

表 136. GPIO 中断清除寄存器（IC - 地址 0x5000 003C、0x5001 003C (GPIO1)、0x5002 003C (GPIO2)）的位描述

位	符号	描述	复位值	访问类型
31:0	CLR	选择引脚 PION_x 上要清除的中断。清除中断边沿检测逻辑。 0 = 无效。 1 = 清除 PION_x 的边沿检测逻辑。	0x00	W

9.1 本章导读

UART0 在所有 LPC122x 部件上都可用。

9.2 基本配置

UART0 模块的时钟和电源由以下寄存器控制：

1. SYSAHBCLKCTRL 寄存器（参见表 21）。
2. 在 UART0 时钟分频寄存器中启用的 UART0_PCLK（参见表 23）。UART 波特率生成器将使用该时钟。

注：在启用 UART0 时钟之前，必须在相应的 IOCON 寄存器中对 UART0 引脚进行配置。

UART0_PCLK 可以在 UART0CLKDIV 寄存器（参见表 23）中禁用，而 UART 模块可以通过系统 AHB 时钟控制寄存器位 12（参见表 21）禁用，以降低功耗。

9.3 特性

- 16 字节接收和发送 FIFO。
- 寄存器存储单元符合 '550 行业标准。
- 接收器 FIFO 触发点可为 1、4、8 和 14 字节。
- 内置波特率生成器。
- UART 允许实施软件或硬件流控制。
- 支持 RS-485/EIA-485 9 位模式和输出启用。
- 调制解调器控制。

9.4 引脚描述

表 137. UART0 引脚描述

引脚	类型	描述
RXD0	输入	串行输入。串行接收数据。
TXD0	输出	串行输出。串行发送数据。
RTS0	输出	请求发送。RS-485 方向控制引脚。
DTR0	输出	数据终端就绪。
DSR0	输入	数据设置就绪。
CTS0	输入	清除发送。
DCD0	输入	数据载波检测。
RI0	输入	振铃指示器。

9.5 寄存器描述

UART 所包含的寄存器，其结构如表 138 所示。除数锁存器访问位 (DLAB) 包含在 LCR[7] 中，可实现对除数锁存器的访问。

表 138. 寄存器简介：UART0（基址：0x4000 8000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
RBR	RO	0x000	接收器缓冲寄存器。包含下一个要读取的已接收字符。(DLAB=0)	不适用
THR	WO	0x000	发送保持寄存器。在此写入下一个要发送的字符。(DLAB=0)	不适用
DLL	R/W	0x000	除数锁存器 LSB。波特率除数值的最低有效字节。整个除数用于从小数比率分频器生成波特率。(DLAB = 1)	0x01
DLM	R/W	0x004	除数锁存器 MSB。波特率除数值的最高有效字节。整个除数用于从小数比率分频器生成波特率。(DLAB = 1)	0x00
IER	R/W	0x004	中断使能寄存器。包含 7 个潜在 UART 中断的各个中断使能位。(DLAB=0)	0x00
IIR	RO	0x008	中断 ID 寄存器。识别挂起的中断。	0x01
FCR	WO	0x008	FIFO 控制寄存器。控制 UART FIFO 的使用和模式。	0x00
LCR	R/W	0x00C	线路控制寄存器。包含帧格式控制和中断生成控制。	0x00
MCR	R/W	0x010	调制解调器控制寄存器	0x00
LSR	RO	0x014	线路状态寄存器。包含发送和接收状态的标志（包括线路故障）。	0x60
MSR	RO	0x018	调制解调器状态寄存器	0x00
SCR	R/W	0x01C	高速暂时寄存器。供软件使用的 8 位临时存储空间。	0x00
ACR	R/W	0x020	自动波特率控制寄存器。包含自动波特率功能的控件。	0x00
-	-	0x024	保留	-
FDR	R/W	0x028	小数分频寄存器。生成波特率分频器的时钟输入。	0x10
-	-	0x02C	保留	-
TER	R/W	0x030	发送使能寄存器。关闭 UART 发送器，以配合软件流控制使用。	0x80
-	-	0x034 - 0x048	保留	-
RS485CTRL	R/W	0x04C	RS-485/EIA-485 控件。包含 RS-485/EIA-485 模式各个方面的配置控件。	0x00
ADRMATCH	R/W	0x050	RS-485/EIA-485 地址匹配。包含 RS-485/EIA-485 模式的地址匹配值。	0x00
RS485DLY	R/W	0x054	RS-485/EIA-485 方向控制延迟。	0x00
FIFOLVL	RO	0x058	FIFO 电平寄存器。提供发送和接收 FIFO 的当前填充电平。	0x00

[1] 复位值仅反映已使用位中保存的数据，它不包括保留位的内容。

9.5.1 UART 接收器缓冲寄存器（当 DLAB = 0 时，只读）

RBR 是 UART RX FIFO 的最高字节。RX FIFO 的最高字节包含最早接收到的字符，并可通过总线接口进行读取。LSB（位 0）表示易铃缆接收的数据位。如果接收到的字符少于 8 位，则未使用的 MSB 用 0 填充。

如果要访问 RBR，LCR 中的除数锁存器访问位 (DLAB) 必须为 0。RBR 始终为只读。

由于奇偶错误 (PE)、帧错误 (FE) 和间隔中断 (BI) 位（参见表 150）与 RBR FIFO 顶部的字节（即下次从 RBR 读取时要读取的字节）相对应，因此，要正确提取有效的接收字节对其状态位，应先读取 LSR 寄存器的内容，然后再读取 RBR 中的字节。

表 139. UART 接收器缓冲寄存器（RBR - 地址 0x4000 8000，当 DLAB = 0 时，只读）位描述

位	符号	描述	复位值
7:0	RBR	UART 接收器缓冲寄存器包含了 UART RX FIFO 中最早接收到的字节。	未定义
31:8	-	保留	-

9.5.2 UART 发送器保持寄存器（当 DLAB = 0 时，只写）

THR 是 UART TX FIFO 的最高字节。最高字节是 TX FIFO 中的最新字符，可通过总线接口进行写入。LSB 代表第一个将要发送的位。

如果要访问 THR，LCR 中的除数锁存器访问位 (DLAB) 必须为 0。THR 始终为只写。

表 140. UART 发送器保持寄存器（THR - 地址 0x4000 8000，当 DLAB = 0 时，只写）位描述

位	符号	描述	复位值
7:0	THR	写入 UART 发送保持寄存器会使数据保存到 UART 发送 FIFO 中。当字节达到 FIFO 的底部并且发送器可用时，字节就会被发送。	不适用
31:8	-	保留	-

9.5.3 UART 除数锁存器 LSB 和 MSB 寄存器（当 DLAB = 1 时）

UART 除数锁存器是 UART 波特率生成器的一部分，并保存使用的值，它与小数分频器一同使用，对 UART_PCLK 时钟进行分频，以产生波特率时钟，波特率时钟必须是所需波特率的 16 倍。DLL 和 DLM 寄存器一起构成了一个 16 位除数，其中 DLL 包含了除数的低 8 位，而 DLM 包含了除数的高 8 位。0x0000 值会被作为 0x0001 值处理，因为除数不能为 0。如果要访问 UART 除数锁存器，LCR 中的除数锁存器访问位 (DLAB) 必须为 1。有关如何为 DLL 和 DLM 选择正确值的详细信息，请参阅[章节 9.5.13](#)。

表 141. UART 除数锁存器 LSB 寄存器（DLL - 地址 0x4000 8000，当 DLAB = 1 时）位描述

位	符号	描述	复位值
7:0	DLLSB	UART 除数锁存器 LSB 寄存器与 DLM 寄存器一起决定 UART 的波特率。	0x01
31:8	-	保留	-

表 142. UART 除数锁存器 MSB 寄存器（DLM - 地址 0x4000 8004，当 DLAB = 1 时）位描述

位	符号	描述	复位值
7:0	DLMSB	UART 除数锁存器 MSB 寄存器与 DLL 寄存器一起决定 UART 的波特率。	0x00
31:8	-	保留	-

9.5.4 UART 中断使能寄存器（当 DLAB = 0 时）

IER 用于启用 4 个 UART 中断源。

表 143. UART 中断使能寄存器（IER - 地址 0x4000 8004，当 DLAB = 0 时）位描述

位	符号	值	描述	复位值
0	RBRIE		RBR 中断使能。启用 UART 的接收数据可用中断。它还控制着字符接收超时中断。	0
		0	禁用 RDA 中断。	
		1	启用 RDA 中断。	
1	THREIE		THRE 中断使能。启用 UART 的 THRE 中断。该中断的状态可从 LSR[5] 中读取。	0
		0	禁用 THRE 中断。	
		1	启用 THRE 中断。	
2	RXLIE		RX 线中断启用。启用 UART RX 线状态中断。该中断的状态可从 LSR[4:1] 中读取。	0
		0	禁用 RX 线状态中断。	
		1	启用 RX 线状态中断。	
3	-	-	保留	-
6:4	-		保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	不适用
7	-	-	保留	0
8	ABEOINTEN		启用自动波特率结束中断。	0
		0	禁用自动波特率结束中断。	
		1	启用自动波特率结束中断。	
9	ABTOINTEN		启用自动波特率超时中断。	0
		0	禁用自动波特率超时中断。	
		1	启用自动波特率超时中断。	
31:10	-		保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	不适用

9.5.5 UART 中断识别寄存器

IIR 提供状态代码用于指示挂起中断的优先级和中断源。在访问 IIR 的过程中，中断被冻结。如果在访问 IIR 的过程中产生中断，该中断会被记录，以用于下次的 IIR 访问。

表 144. UART 中断识别寄存器（IIR - 地址 0x4004 8008，只读）位描述

位	符号	值	描述	复位值
0	INTSTATUS		中断状态。注意：IIR[0] 为低电平有效。挂起的中断可通过评估 IIR[3:1] 来确定。	1
		0	至少有一个中断挂起。	
		1	没有挂起的中断。	

表 144. UART 中断识别寄存器（IIR - 地址 0x4004 8008，只读）位描述（续）

位	符号	值	描述	复位值
3:1	INTID		中断识别。IER[3:1] 可识别对应于 UART RX FIFO 的中断。下面未列出的 IER[3:1] 的其他组合均为保留值（100、101、111）。	0
		0x3	1 - 接收线状态 (RLS)。	
		0x2	2a - 接收数据可用 (RDA)。	
		0x6	2b - 字符超时指示 (CTI)。	
		0x1	3 - THRE 中断。	
		0x0	4 - 调制解调器中断。	
5:4	-		保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用
7:6	FIFOENA		这些位相当于 FCR[0]。	0
8	ABEOINT		自动波特率结束中断。如果自动波特率成功完成，且中断启用，则为真。	0
9	ABTOINT		自动波特率超时中断。如果自动波特率超时，且中断启用，则为真。	0
31:10	-		保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用

位 IIR[9:8] 由自动波特率功能设置，用于发布超时信号或自动波特率结束条件信号。自动波特率中断条件可以通过设置自动波特率控制寄存器中相应的 Clear 位来清除。

如果 IntStatus 位为 1，则表示没有中断被挂起，且 IntId 位将为 0。如果 IntStatus 为 0，表示有一个非自动波特率中断被挂起，此时 IntId 位会确定中断的类型和处理方式，如表 145 中所述。如果指定 IIR[3:0] 的状态，中断处理程序就能确定中断原因以及如何清除有效的中断。在退出中断服务程序之前，必须读取 IIR 来清除中断。

UART RLS 中断 (IIR[3:1] = 011) 为最高优先级中断，每当在 UART RX 输入端发生下面 4 个错误状况中的任意一个时，都可以进行设置：溢出错误 (OE)、奇偶检验错误 (PE)、成帧错误 (FE) 和间隔中断 (BI)。设置中断的 UART RX 错误状况可通过 LSR[4:1] 查看。在读取 LSR 时，该中断将被清除。

UART RDA 中断 (IIR[3:1] = 010) 与 CTI 中断 (IIR[3:1] = 110) 均为第二优先级。当 UART RX FIFO 达到 FCR7:6 中定义的触发电平时，RDA 即被激活；当 UART RX FIFO 深度低于触发电平时，RDA 即被复位。当 RDA 中断激活时，CPU 可读取触发电平定义的数据块。

CTI 中断 (IIR[3:1] = 110) 为第二优先级中断，在 UART RX FIFO 含有至少一个字符并且在接收到 3.5 到 4.5 个字符的时间内没有发生任何 UART RX FIFO 操作时，可对该中断进行设置。任何 UART RX FIFO 操作（UART RSR 的读取或写入）都可以清除该中断。当接收到的信息不是触发电平大小的倍数时，该中断将会刷新 UART RBR。例如：如果外设想要发送一个长度为 105 个字符的消息，而触发电平为 10 个字符，那么前 100 个字符将使 CPU 接收 10 个 RDA 中断，而剩下的 5 个字符使 CPU 收到 1 到 5 个 CTI 中断（取决于服务程序）。

表 145. UART 中断处理

IIR[3:0] 值 [1]	优先级	中断类型	中断源	中断复位
0001	-	无	无	-
0110	最高	RX 线状态/ 错误	OE [2] 、PE [2] 、FE [2] 或 BI [2]	LSR 读操作 [2]
0100	第二	RX 数据可用	RX 数据可用或达到 FIFO 触发电平 (FCR0=1)	RBR 读操作 [3] 或 UART FIFO 低于触发电平
1100	第二	字符 超时 指示	RX FIFO 中至少有一个字符，并且在一段时间内没有字符输入或移出，该时间的长短取决于 FIFO 中的字符数以及在 3.5 到 4.5 字符的时间内的触发值。 实际时间为： $[(\text{字长度}) \times 7 - 2] \times 8 + [(\text{触发电平} - \text{字符数}) \times 8 + 1] \text{ RCLK}$	RBR 读操作 [3]
0010	第三	THRE	THRE [2]	IIR 读操作 [4] (如果是中断源) 或 THR 写操作

- [1] “0000”、“0011”、“0101”、“0111”、“1000”、“1001”、“1010”、“1011”、“1101”、“1110”、“1111” 均为保留值。
- [2] 有关详情，请参阅[章节 9.5.9 “UART 线路状态寄存器”](#)
- [3] 有关详情，请参阅[章节 9.5.1 “UART 接收器缓冲寄存器（当 DLAB = 0 时，只读）”](#)
- [4] 有关详情，请参阅[章节 9.5.5 “UART 中断识别寄存器”](#)和[章节 9.5.2 “UART 发送器保持寄存器（当 DLAB = 0 时，只写）”](#)

UART THRE 中断 (IIR[3:1] = 001) 为第三优先级中断，当 UART THR FIFO 为空，且满足特定的初始化条件时，该中断激活。这些初始化条件是为了让 UART THR FIFO 有机会填入数据，以免在系统启动时产生许多 THRE 中断。当 THRE = 1 时，且在上次的 THRE = 1 事件后，THR 中没有出现至少两个字符时，这些初始化条件就会实现一个字符减去停止位的延时。当没有解码和服务的 THRE 中断时，该延时为 CPU 提供了写数据到 THR 的时间。当 UART THR FIFO 中曾经同时出现两个或更多字符，而当前的 THR 为空时，THRE 中断就会立即被设置。当发生 THR 写操作或 IIR 读操作，且 THRE 为最高优先级中断 (IIR[3:1] = 001) 时，THRE 中断复位。

9.5.6 UART FIFO 控制寄存器

FCR 控制 UART RX 和 TX FIFO 的操作。

表 146. UART FIFO 控制寄存器（FCR - 地址 0x4000 8008，只写）位描述

位	符号	值	描述	复位值
0	FIFOEN		FIFO 启用	0
		0	UART FIFO 被禁用。禁止在应用中使用。	
		1	高电平有效，启用对 UART RX、TX FIFO 和 FCR[7:1] 的访问。该位必须进行设置，以实现正确的 UART 操作。该位的任何变化都会自动清除 UART FIFO。	
1	RXFIFO RS		RX FIFO 复位	0
		0	对两个 UART FIFO 均无影响。	
		1	写逻辑 1 到 FCR[1] 将会清除 UART RX FIFO 中的所有字节，复位指针逻辑。该位可以自动清零。	

表 146. UART FIFO 控制寄存器（FCR - 地址 0x4000 8008，只写）位描述 *（续）*

位	符号	值	描述	复位值
2	TXFIFORS		TX FIFO 复位	0
		0	对两个 UART FIFO 均无影响。	
		1	写逻辑 1 到 FCR[2] 将会清除 UART TX FIFO 中的所有字节，复位指针逻辑。该位可以自动清零。	
3	DMAMODE		DMA 模式选择。当对 FIFO 使能位（该寄存器的位 0）进行设置时，该位选择 DMA 模式。参见 章节 9.5.6.1 。	0
		0	DMA 未使用。	
		1	DMA 模式已启用。	
5:4	-		保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	不适用
7:6	RXTL		RX 触发电平。这两个位决定了接收器 UART FIFO 在激活中断前必须写入的字符数量。	0
		0x0	触发电平 0（1 个字符或 0x01）。	
		0x1	触发电平 1（4 个字符或 0x04）。	
		0x2	触发电平 2（8 个字符或 0x08）。	
		0x3	触发电平 3（14 个字符或 0x0E）。	
31:8	-	-	保留	-

9.5.6.1 DMA 操作

通过使用微型 DMA，用户可选择操作 UART 的发送和 / 或接收。DMA 模式由 FCR 寄存器中的 DMA 模式选择位决定。只有通过 FCR 寄存器中的 FIFO 使能位将 FIFO 启用时，该位才会有效。

9.5.6.1.1 UART 接收器 DMA

在 DMA 模式中，当接收器 FIFO 的电平等于或大于触发电平时，或者在发生字符超时的情况下，接收器 DMA 请求就会生效。请参考上文对 RX 触发电平的描述。接收器 DMA 请求由 DMA 控制器清除。

9.5.6.1.2 UART 发送器 DMA

在 DMA 模式中，当发送器 FIFO 变为未滿时，发送器 DMA 请求就会生效。发送器 DMA 请求由 DMA 控制器清除。

9.5.7 UART 线路控制寄存器

LCR 决定要发送或接收的数据字符的格式。

表 147. UART 线路控制寄存器（LCR - 地址 0x4000 800C）位描述

位	符号	值	描述	复位值
1:0	WLS		字长度选择	0
		0x0	5 位字符长度。	
		0x1	6 位字符长度。	
		0x2	7 位字符长度。	
		0x3	8 位字符长度。	

表 147. UART 线路控制寄存器（LCR - 地址 0x4000 800C）位描述（续）

位	符号	值	描述	复位值
2	SBS		停止位选择	0
		0	1 个停止位。	
		1	2 个停止位（如果 LCR[1:0]=00，则为 1.5 个停止位）。	
3	PE		奇偶检验使能	0
		0	禁用奇偶检验的生成和检查。	
		1	启用奇偶检验的生成和检查。	
5:4	PST		奇偶检验选择	0
		0x0	奇检验。1s 内发送字符数和附加奇偶检验位为奇数。	
		0x1	偶检验。1s 内发送字符数和附加奇偶检验位为偶数。	
		0x2	强制 1 奇偶检验。	
		0x3	强制 0 奇偶校验。	
6	BC		间隔控制	0
		0	禁用间隔传输。	
		1	启用间隔传输。当 LCR[6] 为高电平有效时，输出引脚 UART TXD 强制为逻辑 0。	
7	DLAB		除数锁存器访问位 (DLAB)	0
		0	禁用对除数锁存器的访问。	
		1	启用对除数锁存器的访问。	
31:8	-	-	保留	-

9.5.8 UART 调制解调器控制寄存器

MCR 可启用调制解调器回送模式，并控制调制解调器输出信号。

表 148. UART 调制解调器控制寄存器（MCR - 地址 0x4000 8010）位描述

位	符号	值	描述	复位值
0	DTRCTRL		调制解调器输出引脚 $\overline{\text{DTR}}$ 的源。当调制解调器回送模式激活时，该位读为 0。	0
1	RTSCTRL		调制解调器输出引脚 $\overline{\text{RTS}}$ 的源。当调制解调器回送模式激活时，该位读为 0。	0
3:2	-		保留，用户软件不应对保留位写入 1。从保留位读取的值未定义。	0
4	LMS		回送模式选择。调制解调器回送模式提供了执行诊断回送测试的机制。发送器的串行数据从内部连接至接收器的串行输入。输入引脚 RXD 对回送无任何影响，而输出引脚 TXD 保持在标记状态。 <u>DSR</u> 、CTS、DCD 和 RI 引脚均被忽略。从外部看来， <u>DTR</u> 和 <u>RTS</u> 被设置为无效。从内部看来，MSR 的高 4 位由 MCR 的低 4 位驱动。这样在回送模式下，写 MCR 的低 4 位就有可能生成调制解调器状态中断。	0
		0	禁用调制解调器回送模式。	
		1	启用调制解调器回送模式。	
5	-		保留，用户软件不应对保留位写入 1。从保留位读取的值未定义。	0
6	RTSEN		RTS 启用	0
		0	禁用自动 RTS 流控制。	
		1	启用自动 RTS 流控制。	

表 148. UART 调制解调器控制寄存器（MCR - 地址 0x4000 8010）位描述（续）

位	符号	值	描述	复位值
7	CTSEN		CTS 启用	0
		0	禁用自动 CTS 流控制。	
		1	启用自动 CTS 流控制。	
31:8	-	-	保留	-

9.5.8.1 自动流控制

如果自动 RTS 模式启用，则 UART 接收器 FIFO 硬件可控制 UART 的 $\overline{\text{RTS}}$ 输出。如果自动 CTS 模式启用，则只有在 CTS 输入信号有效时，UART TSR 硬件才启动发送。

9.5.8.1.1 自动 RTS

通过设置 RTSen 位可以启用自动 RTS 功能。自动 RTS 数据流控制在 RBR 模块中产生，并链接至已编程好的接收器 FIFO 触发电平。如果自动 RTS 启用，则按照以下方式对数据流进行控制：

当接收器 FIFO 的电平达到已编程好的触发电平时， $\overline{\text{RTS}}$ 失效（变为高电平值）。发送 UART 在达到触发电平后，可能会发送一个额外字节（假设发送 UART 有额外字节要发送），因为它可能直到开始发送额外字节后才知道 $\overline{\text{RTS}}$ 失效。一旦接收器 FIFO 达到先前的触发电平， $\overline{\text{RTS}}$ 就会自动重新生效（变为低电平值）。发送 UART 的 $\overline{\text{RTS}}$ 信号重新生效后可继续发送数据。

如果自动 RTS 模式被禁用，则 RTSen 位可控制 UART 的 $\overline{\text{RTS}}$ 输出。如果自动 RTS 模式启用，则硬件可控制 RTS 输出，而且可以将 RTS 的实际值复制到 UART 的 RTS 控制位。如果自动 RTS 启用，则只有软件可对 RTS 控制位的值进行只读操作。

示例：假设 UART 在类型 ‘550 模式下操作，将 FCR 中的触发电平设置为 0x2，那么如果自动 RTS 启用，接收 FIFO 只要包含 8 个字节（[136 页表 146](#)），UART 就会使 $\overline{\text{RTS}}$ 输出无效。只要接收 FIFO 达到先前的触发电平， $\overline{\text{RTS}}$ 输出就会重新生效：4 字节。

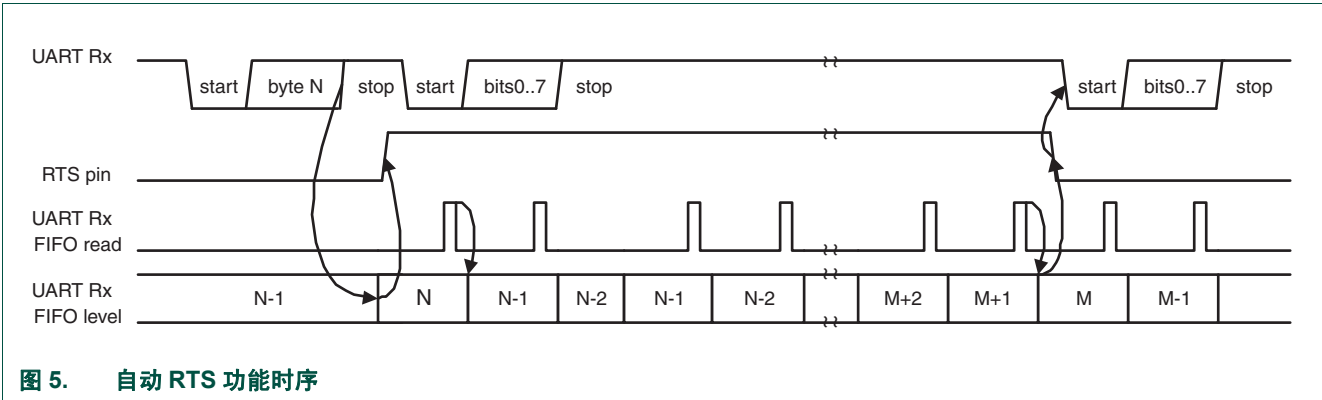


图 5. 自动 RTS 功能时序

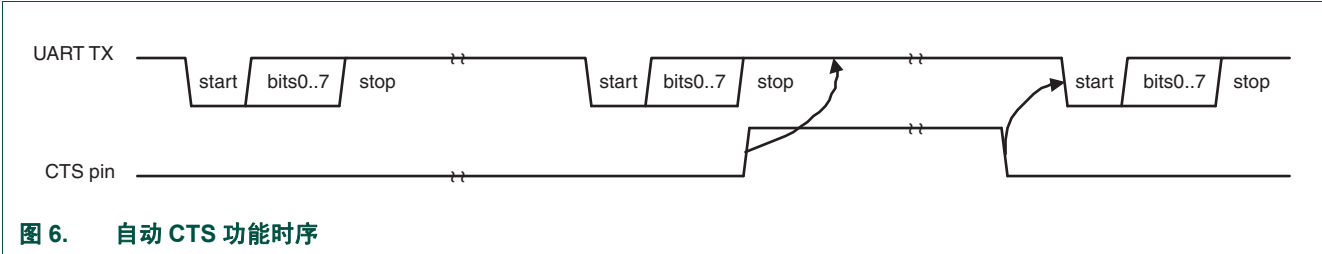
9.5.8.1.2 自动 CTS

通过设置 CTSEN 位可以启用自动 CTS 功能。如果自动 CTS 启用，则 TSR 模块中的发送器电路将在发送下一个数据字节之前检查 CTS 输入。当 CTS 有效（低电平）时，发送器发送下一个字节。为了让发送器停止发送后面的字节，必须在当前正在发送的最后一个停止位发送一半以前释放 CTS。在自动 CTS 模式下，CTS 信号的变化不会触发调制解调器状态中断，除非对 CTS 中断使能位进行设置，不过将会设置 MSR 中的 Delta CTS 位。[表 149](#) 列出了生成调制解调器状态中断的条件。

表 149. 调制解调器状态中断生成

启用调制解调器状态中断 (ER[3])	CTSen (MCR[7])	CTS 中断使能 (IER[7])	Delta CTS (MSR[0])	Delta DCD 或后沿 RI 或 Delta DSR (MSR[3] 或 MSR[2] 或 MSR[1])	调制解调器状态中断
0	x	x	x	x	否
1	0	x	0	0	否
1	0	x	1	x	是
1	0	x	x	1	是
1	1	0	x	0	否
1	1	0	x	1	是
1	1	1	0	0	否
1	1	1	1	x	是
1	1	1	x	1	是

自动 CTS 功能可减少主机系统的中断。当流控制启用时， $\overline{\text{CTS}}$ 状态的改变不会触发主机中断，因为器件会自动控制其发送器。若不使用自动 CTS，则发送器会将发送 FIFO 中存在的所有数据都发送出去，从而导致接收器发生溢出错误。图 6 展示了自动 CTS 功能时序。



开始发送第一个字符时， $\overline{\text{CTS}}$ 信号生效。一旦处理中的数据传输结束，发送就会停止。一旦 CTS 无效（高电平），UART 就会不断地发送 1 位。一旦 CTS 变为无效，传输就会恢复且发送起始位，接着是下一个字符的数据位。

9.5.9 UART 线路状态寄存器

LSR 为只读寄存器，提供 UART TX 和 RX 模块的状态信息。

表 150. UART 线路状态寄存器（LSR - 地址 0x4000 8014，只读）位描述

位	符号	值	描述	复位值
0	RDR		接收器数据就绪：当 RBR 包含未读字符时，LSR[0] 会被设置；当 UART RBR FIFO 为空时，LSR[0] 会被清零。	0
		0	RBR 为空。	
		1	RBR 包含有效数据。	
1	OE		溢出错误。溢出错误条件在错误发生后立即设置。LSR 读操作会将 LSR[1] 清零。当 UART RSR 有了新的字符组合，且 UART RBR FIFO 已满时，LSR[1] 会被设置。在这种情况下，UART RBR FIFO 不会被覆盖，而 UART RSR 中的字符将会丢失。	0
		0	溢出错误状态无效。	
		1	溢出错误状态有效。	

表 150. UART 线路状态寄存器（LSR - 地址 0x4000 8014，只读）位描述（续）

位	符号	值	描述	复位值
2	PE		奇偶检验错误。当已接收字符的奇偶检验位处于错误状态时，就会发生奇偶检验错误。LSR 读操作会将 LSR[2] 清零。奇偶检验错误检测时间取决于 FCR[0]。 注： 奇偶检验错误与 UART RBR FIFO 顶部的字符相关。	0
		0	奇偶检验错误状态无效。	
		1	奇偶检验错误状态有效。	
3	FE		成帧错误。当已接收字符的停止位为逻辑 0 时，会发生成帧错误。LSR 读操作会将 LSR[3] 清零。成帧错误检测时间取决于 FCR0。当检测到有成帧错误时，RX 会尝试与数据重新同步，并假设错误的停止位实际是一个超前的起始位。但是，即使没有出现成帧错误，也无法假设下一个接收到的字节就是正确的。 注： 成帧错误与 UART RBR FIFO 顶部的字符相关。	0
		0	成帧错误状态无效。	
		1	成帧错误状态有效。	
4	BI		间隔中断。在发送整个字符（起始、数据、奇偶检验、停止）的过程中，如果 RXD1 保持在间隔状态（全“0”），则会发生间隔中断。一旦检测到间隔条件，接收器会立即进入空闲状态，直到 RXD1 进入标记状态（全“1”）。LSR 读操作会将该状态位清零。间隔检测的时间取决于 FCR[0]。 注： 间隔中断与 UART RBR FIFO 顶部的字符相关。	0
		0	间隔中断状态无效。	
		1	间隔中断状态有效。	
5	THRE		发送器保持寄存器为空。当检测到 UART THR 为空时，THRE 就会立即被设置；写 THR 时，THRE 就会清零。	1
		0	THR 包含有效数据。	
		1	THR 为空。	
6	TEMT		发送器为空。当 THR 和 TSR 同时为空时，TEMT 就会被设置；TSR 或 THR 任意一个包含有效数据时，TEMT 就会被清零。	1
		0	THR 和 / 或 TSR 包含有效数据。	
		1	THR 和 TSR 为空。	
7	RXFE		RX FIFO 错误。当一个带有 RX 错误（如：帧错误、奇偶错误或间隔中断）的字符载入到 RBR 时，LSR[7] 就会被设置。当 LSR 寄存器被读取，且 UART FIFO 中没有后续错误时，此位会被清除。	0
		0	RBR 不包含 UART RX 错误或 FCR[0]=0。	
		1	UART RBR 包含至少一个 UART RX 错误。	
31:8	-	-	保留	-

9.5.10 UART 调制解调器状态寄存器

MSR 为只读寄存器，提供调制解调器输入信号的状态信息。MSR 读操作会清除 MSR[3:0]。请注意，调制解调器信号不会直接影响 UART 操作。但有助于通过软件执行调制解调器信号操作。

表 151. UART 调制解调器状态寄存器（MSR - 地址 0x4000 8018）位描述

位	符号	值	描述	复位值
0	DCTS		Delta CTS。 当输入 CTS 的状态改变时，该位被设置。读 MSR 会清除该位。	0
		0	没有检测到调制解调器输入 CTS 上的状态变化。	
		1	检测到调制解调器输入 CTS 上的状态变化。	
1	DDSR		Delta DSR。 当输入 DSR 的状态改变时，该位即被设置。MSR 读操作会清除该位。	0
		0	没有检测到调制解调器输入 DSR 上的状态变化。	
		1	检测到调制解调器输入 DSR 上的状态变化。	
2	TERI		后沿 RI。 当输入 RI 上低电平到高电平转换时，该位即被设置。MSR 读操作会清除该位。	0
		0	没有检测到调制解调器输入 RI 上的状态变化。	
		1	检测到 RI 上低电平到高电平的转换。	
3	DDCD		Delta DCD。当输入 DCD 的状态改变时，该位即被设置。MSR 读操作会清除该位。	0
		0	没有检测到调制解调器输入 DCD 上的状态变化。	
		1	检测到调制解调器输入 DCD 上的状态变化。	
4	CTS		清除发送状态。输入信号 CTS 的补码。在调制解调器回送模式下，该位连接到 MCR[1]。	0
5	DSR		数据集就绪状态。输入信号 DSR 的补码。在调制解调器回送模式下，该位连接到 MCR[0]。	0
6	RI		振铃指示器状态。输入 RI 的补码。在调制解调器回送模式下，该位连接到 MCR[2]。	0
7	DCD		数据载波检测状态。输入 DCD 的补码。在调制解调器回送模式下，该位连接到 MCR[3]。	0
31:8	-	-	保留，从保留位读取的值未定义。	不适用

9.5.11 UART 高速暂时寄存器

SCR 不影响 UART 操作。用户可自由对该寄存器进行读和 / 或写操作。不提供中断接口向主机指示 SCR 所发生的读或写操作。

表 152. UART 高速暂时寄存器（SCR - 地址 0x4000 801C）位描述

位	符号	描述	复位值
7:0	PAD	一个可读、可写的字节。	0x00
31:8	-	保留	-

9.5.12 UART 自动波特率控制寄存器

在用户测量波特率生成器的输入时钟 / 数据速率期间，整个测量过程就是由 UART 自动波特率控制寄存器（ACR）进行控制的。用户可自由地对该寄存器进行读写操作。

表 153. 自动波特率控制寄存器（ACR - 地址 0x4000 8020）位描述

位	符号	值	描述	复位值
0	START		起始位。在自动波特率功能结束后，系统会清除此位。	0
		0	自动波特率停止（自动波特率未运行）。	
		1	自动波特率启动（自动波特率正在运行）。自动波特率运行位。在自动波特率功能结束后，系统会清除此位。	
1	MODE		自动波特率模式选择位。	0
		0	模式 0。	
		1	模式 1。	
2	AUTORESTART		启动模式。	0
		0	不重新启动	
		1	如果超时则重新启动（计数器会在下一个 UART RX 下降沿重新启动）	
7:3	-		保留，用户软件不应对此保留位写入 1。从保留位读取的值未定义。	0
8	ABEOINTCLR		自动波特率中断结束清除位（仅可写访问）	0
		0	写 0 无影响。	
		1	写 1 会清除 IIR 中相应的中断。	
9	ABTOINTCLR		自动波特率超时中断清除位（仅可写访问）	0
		0	写 0 无影响。	
		1	写 1 会清除 IIR 中相应的中断。	
31:10	-		保留，用户软件不应对此保留位写入 1。从保留位读取的值未定义。	0

9.5.12.1 自动波特率

UART 自动波特率功能可用于测量基于“AT”协议（Hayes 命令）的输入波特率。如果启用了自动波特率功能，那么此功能将测量接收数据流的位时间，并根据这个结果来设置除数锁存器寄存器 DLM 和 DLL。

自动波特率可通过设置 ACR 起始位来启动。自动波特率可通过清零 ACR 起始位来停止。自动波特率一旦结束，起始位将自动清零，并且读取该位将会返回自动波特率的状态（挂起 / 完成）。

可通过 ACR 模式位来选择所提供的两种自动波特率测量模式。在模式 0 下，波特率是通过对 UART RX 管脚上两个连续的下降沿进行测量（起始位的下降沿和最低有效位的下降沿）来得到的。而在模式 1 下，波特率则是通过测量 UART RX 引脚上的下降沿和后续的上升沿之间的时间（起始位的长度）来得到的。

如果出现超时（速率测量计数器溢出），ACR AutoRestart 位可用于自动重启波特率测量。如果该位被置位，速率测量将会在 UART RX 引脚的下一个下降沿重新启动。

自动波特率功能会产生两种中断：

- 若中断使能，则 IIR ABTOInt 中断将置位（IER ABTOIntEn 置位且自动波特率测量寄存器溢出）。
- 若中断使能，则 IIR ABEOInt 中断将置位（IER ABEOIntEn 置位且自动波特率已经成功完成）。

自动波特率中断必须通过置位相应的 ACR ABTOIntClr 位和 ABEOIntEn 位来清零。

在自动波特率期间，小数波特率发生器通常被禁用（即 $DIVADDVAL = 0$ ）。但是，如果小数波特率发生器被启用（即 $DIVADDVAL > 0$ ），那么它将影响 UART RX 引脚波特率的测量，但 FDR 寄存器的值在速率测量后不会被修改。此外，当使用自动波特率时，任何对 DLM 和 DLL 寄存器的写操作都必须在写 ACR 寄存器之前完成。UART 支持的最小和最大波特率受 UART_PCLK、数据的位数、停止位和奇偶检验位的影响。

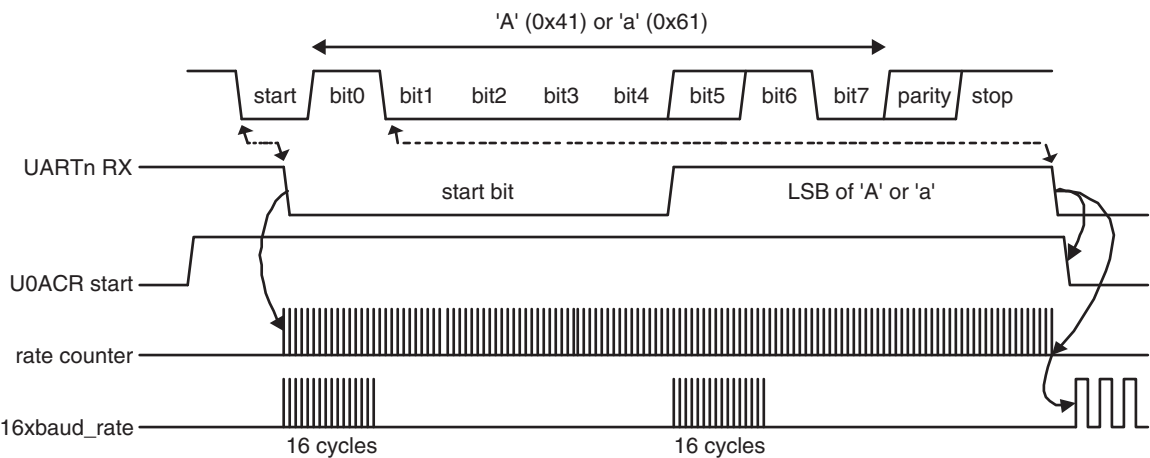
(2)

$$ratemin = \frac{2 \times PCLK}{16 \times 2^{15}} \leq UART_{baudrate} \leq \frac{PCLK}{16 \times (2 + databits + paritybits + stopbits)} = ratemax$$

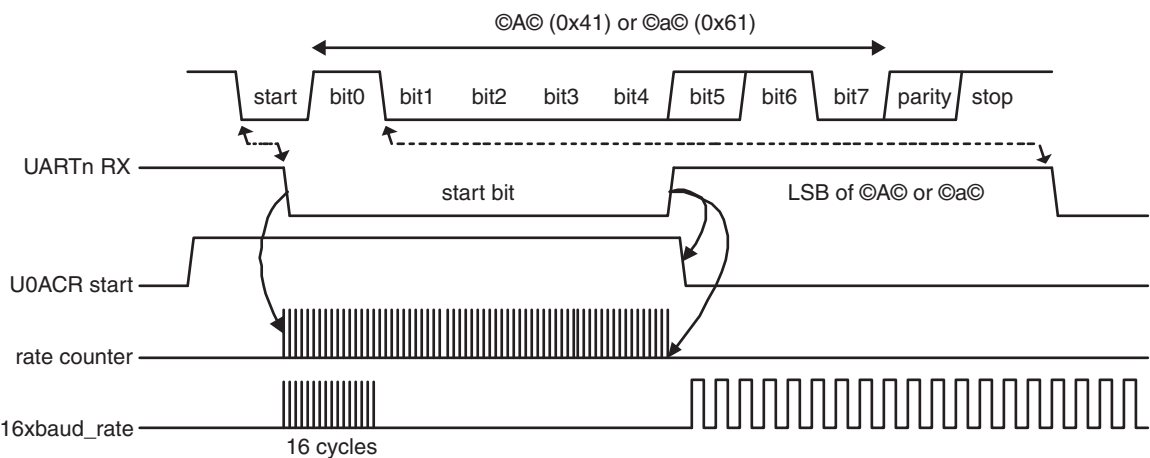
9.5.12.2 自动波特率模式

当软件执行“AT”命令时，它采用期望的字符格式对 UART 进行配置，并置位 ACR 起始位。用户不必关心除数锁存器 DLM 和 DLL 的初始值。由于字母“A”或“a”ASCII 编码（“A”= 0x41，“a”= 0x61）的关系，UART RX 引脚所检测的起始位以及期望字符的 LSB 是由两个下降沿来限定的。当 ACR 起始位被置位时，自动波特率协议将执行以下步骤：

1. ACR 起始位一旦置位，波特率测量计数器立即复位，同时 UART RSR 复位。RSR 波特率切换为最高速率。
2. UART RX 引脚上的下降沿触发起始位的开始。波特率测量计数器将开始对 UART_PCLK 周期（可选择被小数波特率发生器预分频）进行计数。
3. 在接收起始位的过程中，RSR 波特率输入端会产生 16 个脉冲，脉冲频率和（小数波特率发生器预分频）UART 输入时钟相同，这样，保证了起始位存放在 RSR 中。
4. 在接收起始位的过程中（以及模式 0 下的字符 LSB），速率计数器将随着被预分频的 UART 输入时钟 (UART_PCLK) 递增。
5. 如果是模式 0，那么速率计数器将在 UART RX 引脚的下一个下降沿停止。如果是模式 1，那么速率计数器将在 UART RX 引脚的下一个上升沿停止。
6. 速率计数器的值被载入到 DLM/DLL 中，并且波特率将会切换到正常操作模式。在设置完 DLM/DLL 后，如果自动波特率结束中断 IIR ABEOInt 被启用，IIR ABEOInt 将会被置位。接着，RSR 继续接收“A/a”字符剩下的其它位。



a. 模式 0（起始位和 LSB 均用于自动波特率）



b. 模式 1（仅起始位用于自动波特率）

图 7. 自动波特率 a) 模式 0 和 b) 模式 1 的波形图

9.5.13 UART 小数分频寄存器

UART 小数分频寄存器 (FDR) 控制波特率生成器的时钟预分频器，并且用户可自由对该寄存器进行读写操作。该预分频器使用 APB 时钟并根据指定的小数要求产生输出时钟。

注意事项：如果小数分频器有效 (DIVADDVAL > 0)，且 DLM = 0，则 DLL 寄存器的值必须大于或等于 3。

表 154. UART 小数分频寄存器（FDR - 地址 0x4000 8028）位描述

位	功能	描述	复位值
3:0	DIVADDVAL	波特率发生器预分频器除数值。如果该字段为 0，则小数波特率生成器将不会影响 UARTn 的波特率。	0
7:4	MULVAL	波特率预分频器乘数值。不管是否使用小数波特率发生器，为了让 UARTn 正常运作，该字段必须大于或等于 1。	1
31:8	-	保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	0

该寄存器控制波特率发生器的时钟预分频器。寄存器的复位值会让 UART 的小数功能保持在禁用状态，从而确保 UART 在软件和硬件方面能够与不具备该特性的 UART 完全兼容。

UART 波特率可以按下列公式计算 ($n = 1$):

(3)

$$UART_{baudrate} = \frac{PCLK}{16 \times (256 \times DLM + DLL) \times \left(1 + \frac{DivAddVal}{MulVal}\right)}$$

其中，UART_PCLK 为外设时钟，DLM 和 DLL 为标准 UART 波特率分频寄存器，而 DIVADDVAL 和 MULVAL 为 UART 小数波特率发生器的特定参数。

MULVAL 和 DIVADDVAL 的值应符合以下条件：

1. $1 \leq MULVAL \leq 15$
2. $0 \leq DIVADDVAL \leq 14$
3. $DIVADDVAL < MULVAL$

FDR 的值在发送 / 接收数据的过程中不应进行更改，否则可能会导致数据丢失或损坏。

如果 FDR 寄存器值不满足上述两个要求，那么小数分频器输出则为未定义。如果 DIVADDVAL 为 0，那么小数分频器将被禁用，并且不会对时钟进行分频。

9.5.13.1 波特率计算

UART 可以与小数分频器一起工作，也可以不使用小数分频器。在实际应用中，使用几种不同的小数分频器设置很可能会获得期望的波特率。下面的算法描绘了查找一组 DLM、DLL、MULVAL 以及 DIVADDVAL 值的方法。这组参数产生的波特率与期望值的相对误差小于 1.1%。

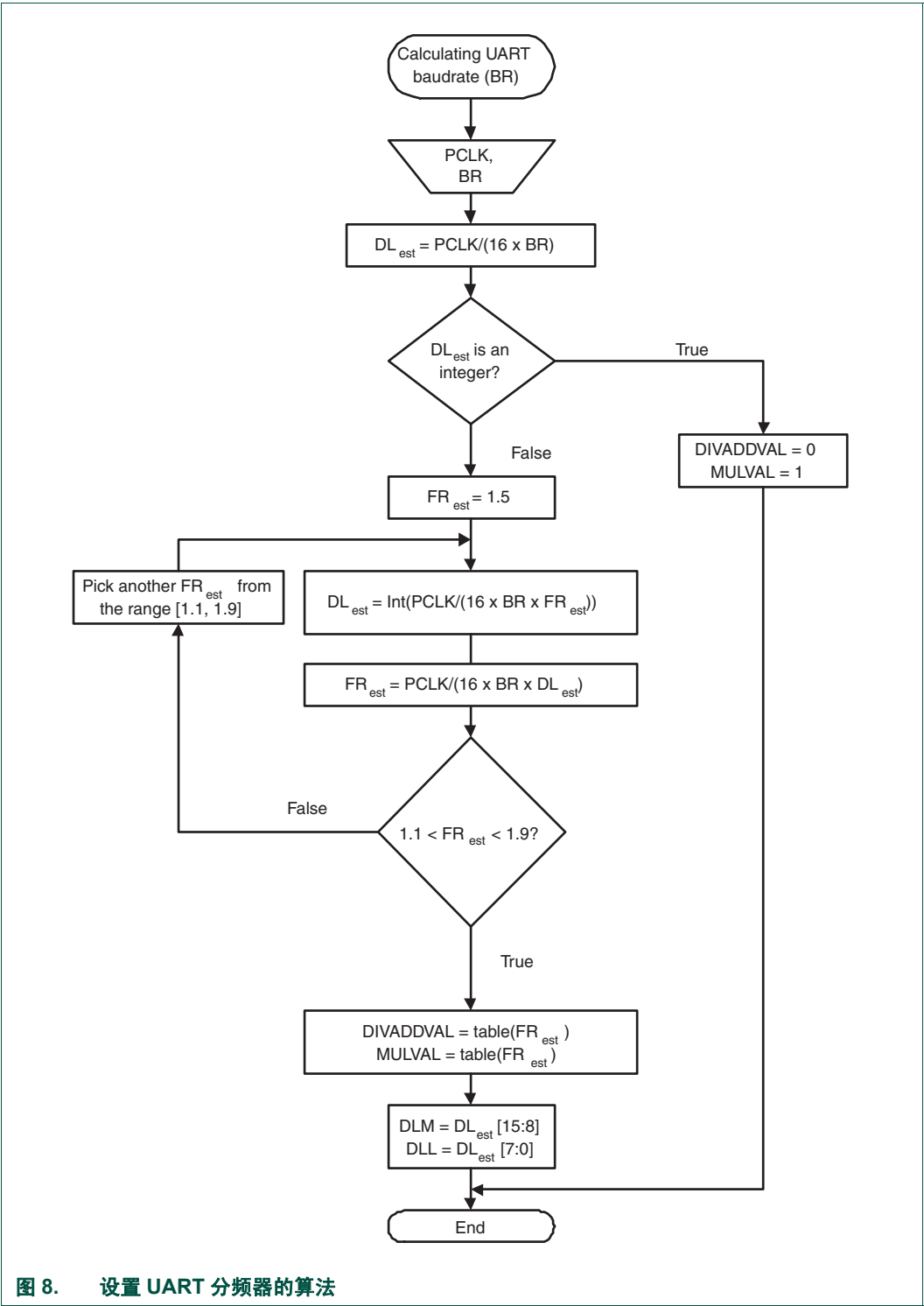


图 8. 设置 UART 分频器的算法

表 155. 小数分频器设置查找表

FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal
1.000	0/1	1.250	1/4	1.500	1/2	1.750	3/4
1.067	1/15	1.267	4/15	1.533	8/15	1.769	10/13
1.071	1/14	1.273	3/11	1.538	7/13	1.778	7/9
1.077	1/13	1.286	2/7	1.545	6/11	1.786	11/14
1.083	1/12	1.300	3/10	1.556	5/9	1.800	4/5
1.091	1/11	1.308	4/13	1.571	4/7	1.818	9/11
1.100	1/10	1.333	1/3	1.583	7/12	1.833	5/6
1.111	1/9	1.357	5/14	1.600	3/5	1.846	11/13
1.125	1/8	1.364	4/11	1.615	8/13	1.857	6/7
1.133	2/15	1.375	3/8	1.625	5/8	1.867	13/15
1.143	1/7	1.385	5/13	1.636	7/11	1.875	7/8
1.154	2/13	1.400	2/5	1.643	9/14	1.889	8/9
1.167	1/6	1.417	5/12	1.667	2/3	1.900	9/10
1.182	2/11	1.429	3/7	1.692	9/13	1.909	10/11
1.200	1/5	1.444	4/9	1.700	7/10	1.917	11/12
1.214	3/14	1.455	5/11	1.714	5/7	1.923	12/13
1.222	2/9	1.462	6/13	1.727	8/11	1.929	13/14
1.231	3/13	1.467	7/15	1.733	11/15	1.933	14/15

9.5.13.1.1 示例 1: UART_PCLK = 14.7456 MHz, BR = 9600

根据所提供的算法， $DL_{est} = PCLK / (16 \times BR) = 14.7456 \text{ MHz} / (16 \times 9600) = 96$ 。因为这里的 DL_{est} 是一个整数，所以 $DIVADDVAL = 0$ ， $MULVAL = 1$ ， $DLM = 0$ ，且 $DLL = 96$ 。

9.5.13.1.2 示例 2: UART_PCLK = 12 MHz, BR = 115200

根据所提供的算法， $DL_{est} = PCLK / (16 \times BR) = 12 \text{ MHz} / (16 \times 115200) = 6.51$ 。该算式中的 DL_{est} 并不是整数，因此下一步就要对 FR 参数进行估算。使用 $FR_{est} = 1.5$ 进行首次估算，得到新的 $DL_{est} = 4$ ，然后使用 $FR_{est} = 1.628$ 再进行计算。由于 $FR_{est} = 1.628$ 是在 1.1 到 1.9 的指定范围之内，因此 $DIVADDVAL$ 和 $MULVAL$ 的值可通过附带的查找表获得。

在查找表表 155 中，最接近 $FR_{est} = 1.628$ 的值为 $FR = 1.625$ 。也就是说 $DIVADDVAL = 5$ 而 $MULVAL = 8$ 。

基于这些查找结果，建议 UART 设置为： $DLM = 0$ ， $DLL = 4$ ， $DIVADDVAL = 5$ 和 $MULVAL = 8$ 。根据方程 3，UART 的波特率为 115384。该速率与原来指定的 115200 之间存在 0.16% 的相对误差。

9.5.14 UART 发送使能寄存器 (TER - 0x4000 8030)

除了配备完整的硬件流控制（上述的自动 CTS 和自动 RTS 机制）之外，TER 还可以实现软件流控制。当 $TXEn = 1$ 时，只要数据可用，UART 发送器就会一直发送数据。一旦 $TXEn$ 变为 0，UART 就会停止数据传输。

虽然表 156 描述了如何利用 $TXEn$ 位来实现硬件流控制，但我们强烈建议用户采用 UART 硬件所实现的自动流控制特性处理硬件流控制，并限制 $TXEn$ 对软件流控制的范围。

TER 可实现软件和硬件流控制。当 TXEn = 1 时，只要数据可用，UART 发送器就会一直发送数据。一旦 TXEn 变为 0，UART 就会停止数据传输。

表 156 描述了如何利用 TXEn 位来实现软件流控制。

表 156. UART 发送使能寄存器（TER - 地址 0x4000 8030）位描述

位	符号	描述	复位值
6:0	-	保留，用户软件不应对保留位写入 1。从保留位读取的值未定义。	不适用
7	TXEN	该位为 1 时（复位后），一旦先前的数据都被发送后，写入 THR 1 的数据就会在 TXD 引脚上输出。如果在发送某字符时该位被清零，那么在将该字符发送完毕后就不再发送字符，直到该位再次被置位。也就是说，该位为 0 时会阻止字符从 THR 或 TX FIFO 传输到发送移位寄存器。当检测到硬件信号握手 TX-permit 信号 (CTS) 变为假时，或者在接收到 XOFF 字符 (DC3) 时，软件通过执行软件握手可以将该位清零。当检测到 TX-permit 信号变为真时，或者在接收到 XON (DC1) 字符时，软件可以将该位重新置位。	1
31:8	-	保留	-

9.5.15 UART RS485 控制寄存器

RS485CTRL 寄存器控制 UART 在 RS-485/EIA-485 模式下的配置。

表 157. UART RS485 控制寄存器（RS485CTRL - 地址 0x4000 804C）位描述

位	符号	值	描述	复位值
0	NMMEN		NMM 启用。	0
		0	RS-485/EIA-485 普通多点模式 (NMM) 已禁用。	
		1	RS-485/EIA-485 普通多点模式 (NMM) 已启用。在该模式下，当接收的字节导致 UART 设置奇偶检验错误并生成中断时，对地址进行检测。	
1	RXDIS		接收器启用。	0
		0	接收器已启用。	
		1	接收器已禁用。	
2	AADEN		AAD 启用。	0
		0	自动地址检测 (AAD) 已禁用。	
		1	自动地址检测 (AAD) 已启用。	
3	SEL		选择方向控制引脚	0
		0	如果方向控制已启用（位 DCTRL = 1），则引脚 RTS 用于方向控制。	
		1	如果方向控制已启用（位 DCTRL = 1），则引脚 DTR 用于方向控制。	
4	DCTRL		自动方向控制启用。	0
		0	禁用自动方向控制。	
		1	启用自动方向控制。	

表 157. UART RS485 控制寄存器（RS485CTRL - 地址 0x4000 804C）位描述（续）

位	符号	值	描述	复位值
5	OINV		极性控制。该位完全改变了 $\overline{\text{RTS}}$ （或 $\overline{\text{DTR}}$ ）引脚上方向控制信号的极性。	0
		0	当发送器有数据要发送时，方向控制引脚会被驱动为逻辑 0。在最后一个数据位被发送后，该位就会被驱动为逻辑 1。	
		1	当发送器有数据要发送时，方向控制引脚会被驱动为逻辑 1。在最后一个数据位被发送后，该位就会被驱动为逻辑 0。	
31:6	-	-	保留，用户软件不应对保留位写入 1。从保留位读取的值未定义。	不适用

9.5.16 UART RS-485 地址匹配寄存器

RS485ADRMATCH 寄存器包含 RS-485/EIA-485 模式的地址匹配值。

表 158. UART RS-485 地址匹配寄存器（RS485ADRMATCH - 地址 0x4000 8050）位描述

位	符号	描述	复位值
7:0	ADRMATCH	包含地址匹配值。	0x00
31:8	-	保留	-

9.5.17 UART1 RS-485 延迟值寄存器

用户可通过对 8 位 RS485DLY 寄存器编程来设置：最后一个停止位离开 TXFIFO 到使 $\overline{\text{RTS}}$ （或 $\overline{\text{DTR}}$ ）无效之间的延迟。这段延迟时间是在波特率时钟周期之内的。延迟可编程设置为 0-255 个位时间。

表 159. UART RS-485 延迟值寄存器（RS485DLY - 地址 0x4000 8054）位描述

位	符号	描述	复位值
7:0	DLY	包含方向控制（RTS 或 DTR）延迟值。该寄存器与一个 8 位计数器配合使用。	0x00
31:8	-	保留，用户软件不应对保留位写入 1。从保留位读取的值未定义。	不适用

9.5.18 RS-485/EIA-485 模式的操作

RS-485/EIA-485 特性允许 UART 配置为可寻址的从机。可寻址从机是由众多受控于单一主机中的一个。

UART 主机发送器通过检查奇偶检验位（第 9 位）是否设置“1”，来识别地址字符。对于数据字符，奇偶检验位是“0”。

每一个 UART 从机接收器都会被分配到一个不同的地址。从机可编程设置为人工或自动的方式拒绝不是它们自己地址的数据。

9.5.18.1 RS-485/EIA-485 普通多点模式 (NMM)

通过将 RS485CTRL 位设置为 0，来启用该模式。在该模式下，当接收的字节导致 UART 设置奇偶检验错误并生成中断时，对地址进行检测。

如果接收器已禁用（RS485CTRL 位 1 =“1”），任何接收到的数据字节都会被忽略且不会存储到 RXFIFO 中。当检测到地址字节（奇偶检验位 =“1”）时，它会被放置在 RXFIFO 中，同时生成一个 RX 数据就绪中断。然后，处理器可以读取地址字节，决定是否允许接收器接收后续数据。

当接收器被允许（RS485CTRL 位 1 =“0”）时，所有接收到的字节（无论是数据还是地址）都将被接受并存储到 RXFIFO 中。当接收到一个地址字符时，将会产生一个奇偶检验错误中断，由处理器决定是否禁用接收器。

9.5.18.2 RS-485/EIA-485 自动地址检测 (AAD) 模式

当同时对 RS485CTRL 寄存器位 0（9 位模式使能）和位 2（AAD 模式使能）进行设置时，UART 处于自动地址检测模式。

在该模式下，接收器将会把它接收到的任何地址字节（奇偶校验 =“1”）与 RS485ADRMATCH 寄存器中编程设置的 8 位值进行比较。

若接收器被禁用（RS485CTRL 位 1 =“1”），如果接收到的字节是数据字节或是与 RS485ADRMATCH 值不匹配的地址字节，则都会被丢弃。

若检测到一个匹配的地址字符，它会带着奇偶检验位一起被压入 RXFIFO 中，同时接收器会被自动启用（RS485CTRL 位 1 会被硬件清除）。接收器也会生成一个 RX 数据就绪中断。

当接收器被启用（RS485CTRL 位 1 =“0”）时，所有接收到的字节都将被接受并存储到 RXFIFO 中，直到收到一个与 RS485ADRMATCH 值不匹配的地址字节。当这种情况发生时，接收器会自动被硬件禁用（RS485CTRL 位 1 将被设置），收到的不匹配地址字符将不会被存储在 RXFIFO 中。

9.5.18.3 RS-485/EIA-485 自动方向控制

RS485/EIA-485 模式允许发送器自动控制 DIR 引脚的状态，它可以作为方向控制的输出信号。

通过设置 RS485CTRL 位 4 =“1”可启用这个特性。

如果已启用方向控制，当 RS485CTRL 位 3 =“0”时会使用 $\overline{\text{RTS}}$ 引脚，当 RS485CTRL 位 3 =“1”时，则使用 DTR 引脚。

当启用自动方向控制时，在 CPU 写数据到 TXFIFO 时被选中的引脚会生效（驱动为低电平）。当数据的最后一位被发送出去之后，引脚变为无效（驱动为高电平）。参见 RS485CTRL 寄存器的位 4 和位 5。

除了回送模式之外，RS485CTRL 位 4 优于所有其他控制方向引脚的机制。

9.5.18.4 RS485/EIA-485 驱动器延迟时间

驱动器延迟时间是指最后一个停止位离开 TXFIFO 到 $\overline{\text{RTS}}$ 失效这一段时间。这段延迟时间可在 8 位 RS485DLY 寄存器中进行编程设置。这段延迟时间是在波特率时钟周期之内的。延迟可编程设置为 0-255 个位时间。

9.5.18.5 RS485/EIA-485 输出反转

引脚 $\overline{\text{RTS}}$ （或 $\overline{\text{DTR}}$ ）方向控制信号的极性可通过对寄存器 RS485CTRL 第 5 位编程来反转。如果该位被设置，则当发送器中有数据正在等待被发送时，方向控制引脚会被驱动为逻辑 1。当数据的最后一位已被发送之后，方向控制引脚会被驱动为逻辑 0。

9.5.19 UART FIFO 电平寄存器

FIFOLVL 寄存器是一个只读寄存器，允许软件读取 FIFO 的当前电平状态。发送和接收 FIFO 的电平均呈现在该寄存器中。

表 160. UART FIFO 电平寄存器（FIFOLVL - 地址 0x4000 8058，只读）位描述

位	符号	描述	复位值
3:0	RXFIFILVL	反映 UART 接收器 FIFO 的当前电平。 0 = 空， 0xF = FIFO 满。	0x00
7:4	-	保留。从保留位读取的值未定义。	不适用
11:8	TXFIFOLVL	反映 UART 发送器 FIFO 的当前电平。 0 = 空， 0xF = FIFO 满。	0x00
31:12	-	保留。从保留位读取的值未定义。	不适用

9.6 架构

UART 的架构如下面的功能框图所示。

APB 接口提供 CPU 或主机与 UART 之间的通信链路。

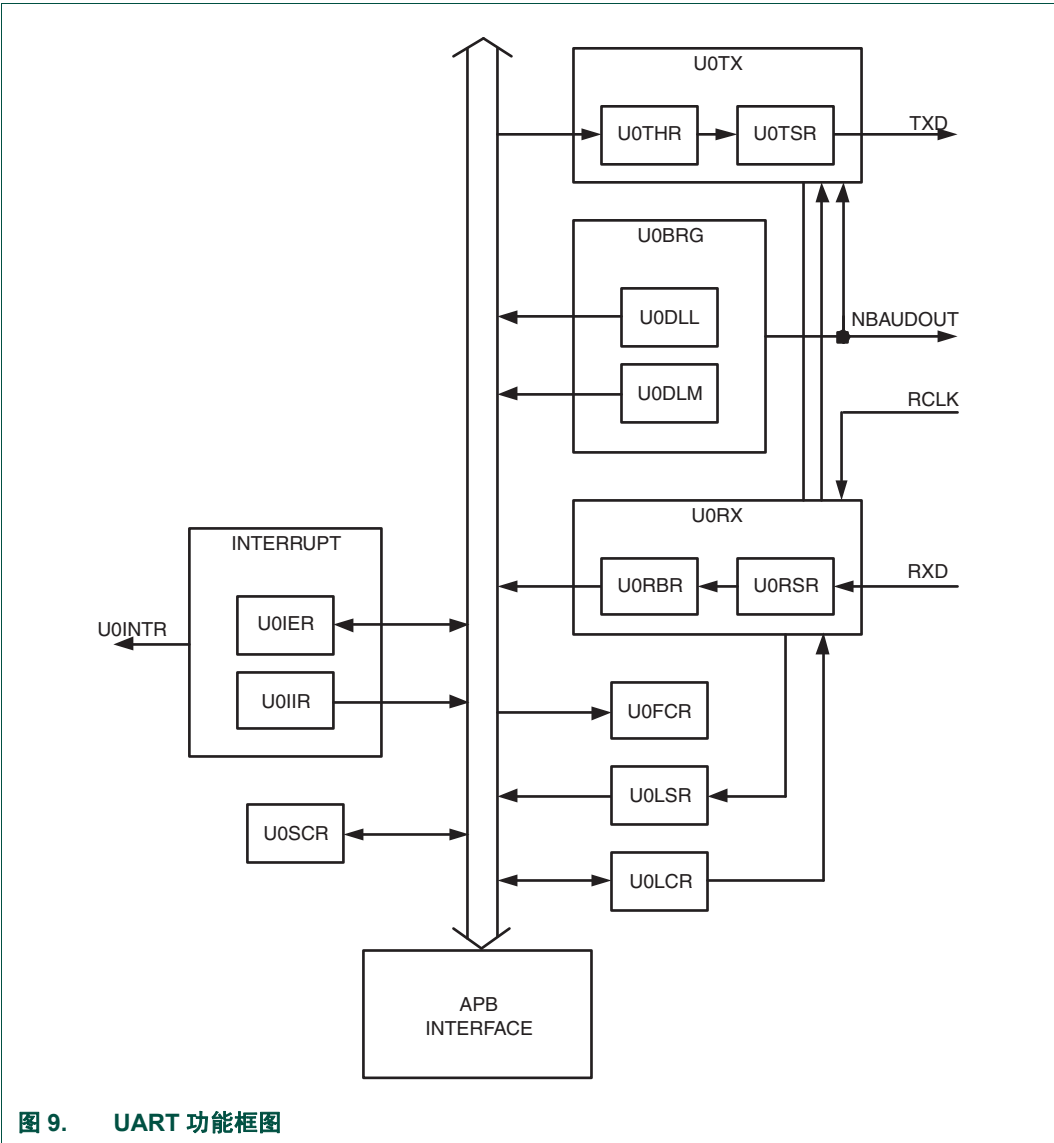
UART 接收器模块 RX 监视串行输入线 RXD 的有效输入。UART RX 移位寄存器 (RSR) 通过 RXD 接受有效的字符。当 RSR 中接收到一个有效字符后，它将该字符传送到 UART RX 缓冲寄存器 FIFO 中，等待 CPU 或主机通过通用主机接口进行访问。

UART 发送器模块 TX 接受 CPU 或主机写入的数据并将数据缓冲到 UART TX 保持寄存器 FIFO(THR) 中。UART TX 移位寄存器 (TSR) 读取存储在 THR 中的数据，并将数据通过串行输出引脚 TXD1 发送。

UART 波特率发生器模块 BRG 产生 UART TX 模块所使用的定时使能。BRG 时钟输入源为 UART_PCLK。主时钟按照 DLL 和 DLM 寄存器中所指定的除数进行分频。分频的时钟为 16 倍过采样时钟 NBAUDOUT。

中断接口包含寄存器 IER 和 IIR。中断接口接收若干个由 TX 和 RX 模块发出的单时钟宽度的使能信号。

TX 和 RX 的状态信息保存在 LSR 中。TX 和 RX 的控制信息保存在 LCR 中。



10.1 本章导读

UART1 在所有 LPC122x 部件上都可用。

10.2 基本配置

UART1 模块的时钟和电源由以下寄存器控制：

1. SYSAHBCLKCTRL 寄存器（参见[表 21](#)）。
2. UART1 时钟分频器寄存器中启用的 UART1_PCLK（参见[表 24](#)）。UART 波特率生成器将使用该时钟。

注：在启用 UART1 时钟之前，必须在相应的 IOCON 寄存器中对 UART1 引脚进行配置。

UART1_PCLK 可以在 UART1CLKDIV 寄存器（参见[表 24](#)）中禁用，而 UART 模块可以通过系统 AHB 时钟控制寄存器位 13（参见[表 21](#)）禁用，以降低功耗。

10.3 特性

- 16 字节的接收与发送 FIFO。
- 寄存器位置符合 '550 工业标准。
- 接收器 FIFO 触发点可为 1、4、8 和 14 字节。
- 内置波特率发生器。
- UART 允许实施软件或硬件流控制。
- IrDA 模式支持红外通信。

10.4 引脚说明

表 161. UART1 引脚说明

引脚	类型	描述
RXD1	输入	串行输入。串行接收数据。
TXD1	输出	串行输出。串行发送数据。

10.5 寄存器描述

UART 所包含的寄存器结构如[表 162](#)所示。除数锁存器访问位 (DLAB) 包含在 LCR[7] 中，可实现对除数锁存器的访问。

表 162. 寄存器简介：UART0（基址：0x4000 C000）

名称	访问类型	地址偏移	描述	复位值 ^[1]	备注
RBR	RO	0x000	接收器缓冲寄存器。包含下一个要读取的已接收字符。	不适用	当 DLAB=0
THR	WO	0x000	发送保持寄存器。在此写入下一个要发送的字符。	不适用	当 DLAB=0
DLL	R/W	0x000	除数锁存器 LSB。波特率除数值的最低有效字节。整个除数用于从小数比率分频器生成波特率。	0x01	当 DLAB=1
DLM	R/W	0x004	除数锁存器 MSB。波特率除数值的最高有效字节。整个除数用于从小数比率分频器生成波特率。	0x00	当 DLAB=1
IER	R/W	0x004	中断使能寄存器。包含 7 个潜在 UART 中断的各个中断使能位。	0x00	当 DLAB=0
IIR	RO	0x008	中断 ID 寄存器。识别挂起的中断。	0x01	-
FCR	WO	0x008	FIFO 控制寄存器。控制 UART FIFO 的使用和模式。	0x00	-
LCR	R/W	0x00C	线路控制寄存器。包含帧格式控制和中断生成控制。	0x00	-
-	-	0x010	保留	0x00	-
LSR	RO	0x014	线路状态寄存器。包含发送和接收状态的标志（包括线路故障）。	0x60	-
-	-	0x018	保留	0x00	-
SCR	R/W	0x01C	高速暂时寄存器。供软件使用的 8 位临时存储空间。	0x00	-
ACR	R/W	0x020	自动波特率控制寄存器。包含自动波特率功能的控件。	0x00	-
ICR	R/W	0x024	IrDA 控制寄存器。启用和配置 IrDA 模式。	0x00	-
FDR	R/W	0x028	小数分频寄存器。生成波特率分频器的时钟输入。	0x10	-
-	-	0x02C	保留	-	-
TER	R/W	0x030	发送使能寄存器。关闭 UART 发送器，以配合软件流控制使用。	0x80	-
-	-	0x034 - 0x054	保留	-	-
FIFOLVL	RO	0x058	FIFO 电平寄存器。提供发送和接收 FIFO 的当前填充电平。	0x00	-

[1] 复位值仅反映已使用位中保存的数据，它不包括保留位的内容。

10.5.1 UART 接收器缓冲寄存器（当 DLAB = 0 时，只读）

RBR 是 UART RX FIFO 的最高字节。RX FIFO 的最高字节包含最早接收到的字符，并可通过总线接口进行读取。LSB（位 0）表示易铃缆接收的数据位。如果接收到的字符少于 8 位，则未使用的 MSB 用 0 填充。

如果要访问 RBR，LCR 中的除数锁存器访问位 (DLAB) 必须为 0。RBR 始终为只读。

由于奇偶错误 (PE)、帧错误 (FE) 和间隔中断 (BI) 位（参见表 172）与 RBR FIFO 顶部的字节（即下次从 RBR 读取时要读取的字节）相对应，因此，要正确提取有效的接收字节对其状态位，应先读取 LSR 寄存器的内容，然后再读取 RBR 中的字节。

表 163. UART 接收器缓冲寄存器（RBR - 地址 0x4000 C000，当 DLAB = 0 时，只读）位描述

位	符号	描述	复位值
7:0	RBR	UART 接收器缓冲寄存器包含了 UART RX FIFO 中最早接收到的字节。	未定义
31:8	-	保留	-

10.5.2 UART 发送器保持寄存器（当 DLAB = 0 时，只写）

THR 是 UART TX FIFO 的最高字节。最高字节是 TX FIFO 中的最新字符，可通过总线接口进行写入。LSB 代表第一个将要发送的位。

如果要访问 THR，LCR 中的除数锁存器访问位 (DLAB) 必须为 0。THR 始终为只写。

表 164. UART 发送器保持寄存器（THR - 地址 0x4000 C000，当 DLAB = 0 时，只写）位描述

位	符号	描述	复位值
7:0	THR	写入 UART 发送保持寄存器会使数据保存到 UART 发送 FIFO 中。当字节达到 FIFO 的底部并且发送器可用时，字节就会被发送。	不适用
31:8	-	保留	-

10.5.3 UART 除数锁存器 LSB 和 MSB 寄存器（当 DLAB = 1 时）

UART 除数锁存器是 UART 波特率生成器的一部分，并保存使用的值，它与小数分频器一同使用，对 UART_PCLK 时钟进行分频，以产生波特率时钟，波特率时钟必须是所需波特率的 16 倍。DLL 和 DLM 寄存器一起构成了一个 16 位除数，其中 DLL 包含了除数的低 8 位，而 DLM 包含了除数的高 8 位。0x0000 值会被作为 0x0001 值处理，因为除数不能为 0。如果要访问 UART 除数锁存器，LCR 中的除数锁存器访问位 (DLAB) 必须为 1。有关如何为 DLL 和 DLM 选择正确值的详细信息，请参阅[章节 10.5.12](#)。

表 165. UART 除数锁存器 LSB 寄存器（DLL - 地址 0x4000 C000，当 DLAB = 1 时）位描述

位	符号	描述	复位值
7:0	DLLSB	UART 除数锁存器 LSB 寄存器与 DLM 寄存器一起决定 UART 的波特率。	0x01
31:8	-	保留	-

表 166. UART 除数锁存器 MSB 寄存器（DLM - 地址 0x4000 C004，当 DLAB = 1 时）位描述

位	符号	描述	复位值
7:0	DLMSB	UART 除数锁存器 MSB 寄存器与 DLL 寄存器一起决定 UART 的波特率。	0x00
31:8	-	保留	-

10.5.4 UART 中断使能寄存器（当 DLAB = 0 时）

IER 用于启用 4 个 UART 中断源。

表 167. UART 中断使能寄存器（IER - 地址 0x4000 C004，当 DLAB = 0 时）位描述

位	符号	值	描述	复位值
0	RBRIE		RBR 中断使能。启用 UART 的接收数据可用中断。它还控制着字符接收超时中断。	0
		0	禁用 RDA 中断。	
		1	启用 RDA 中断。	
1	THREIE		THRE 中断使能。启用 UART 的 THRE 中断。该中断的状态可从 LSR[5] 中读取。	0
		0	禁用 THRE 中断。	
		1	启用 THRE 中断。	
2	RXIE		RX 线中断使能。启用 UART RX 线状态中断。该中断的状态可从 LSR[4:1] 中读取。	0
		0	禁用 RX 线状态中断。	
		1	启用 RX 线状态中断。	
3	-	-	保留	-
6:4	-		保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用
7	-	-	保留	0
8	ABEOINTEN		启用自动波特率结束中断。	0
		0	禁用自动波特率结束中断。	
		1	启用自动波特率结束中断。	
9	ABTOINTEN		启用自动波特率超时中断。	0
		0	禁用自动波特率超时中断。	
		1	启用自动波特率超时中断。	
31:10	-		保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用

10.5.5 UART 中断识别寄存器

IIR 提供状态代码用于指示挂起中断的优先级和中断源。在访问 IIR 的过程中，中断被冻结。如果在访问 IIR 的过程中产生中断，该中断会被记录，以用于下次的 IIR 访问。

表 168. UART 中断识别寄存器（IIR - 地址 0x4004 C008，只读）位描述

位	符号	值	描述	复位值
0	INTSTATUS		中断状态。注意：IIR[0] 为低电平有效。挂起的中断可通过评估 IIR[3:1] 来确定。	1
		0	至少有一个中断挂起。	
		1	没有挂起的中断。	
3:1	INTID		中断识别。IER[3:1] 可识别对应于 UART RX FIFO 的中断。下面未列出的 IER[3:1] 的其他组合均为保留值（000、100、101、111）。	0
		0x3	1 - 接收线状态 (RLS)。	
		0x2	2a - 接收数据可用 (RDA)。	
		0x6	2b - 字符超时指示 (CTI)。	
		0x1	3 - THRE 中断。	
5:4	-		保留，用户软件不应为保留位写入 1。从保留位读取的值未定义。	不适用
7:6	FIFOEN		FIFO 使能。这些位相当于 FCR[0]。	0
8	ABEOINT		自动波特率结束中断。如果自动波特率成功完成，且中断启用，则为真。	0
9	ABTOINT		自动波特率超时中断。如果自动波特率超时，且中断启用，则为真。	0
31:10	-		保留，用户软件不应为保留位写入 1。从保留位读取的值未定义。	不适用

位 IIR[9:8] 由自动波特率功能设置，用于发布超时信号或自动波特率结束条件信号。自动波特率中断条件可以通过设置自动波特率控制寄存器中相应的 Clear 位来清除。

如果 IntStatus 位为 1，则表示没有中断被挂起，且 IntId 位将为 0。如果 IntStatus 为 0，表示有一个非自动波特率中断被挂起，此时 IntId 位会确定中断的类型和处理方式，如表 169 中所述。如果指定 IIR[3:0] 的状态，中断处理程序就能确定中断原因以及如何清除有效的中断。在退出中断服务程序之前，必须读取 IIR 来清除中断。

UART RLS 中断 (IIR[3:1] = 011) 为最高优先级中断，每当在 UART RX 输入端发生下面 4 个错误状况中的任意一个时，都可以进行设置：溢出错误 (OE)、奇偶检验错误 (PE)、成帧错误 (FE) 和间隔中断 (BI)。设置中断的 UART RX 错误状况可通过 LSR[4:1] 查看。在读取 LSR 时，该中断将被清除。

UART RDA 中断 (IIR[3:1] = 010) 与 CTI 中断 (IIR[3:1] = 110) 均为第二优先级。当 UART RX FIFO 达到 FCR7:6 中定义的触发电平时，RDA 即被激活；当 UART RX FIFO 深度低于触发电平时，RDA 即被复位。当 RDA 中断激活时，CPU 可读取触发电平定义的数据块。

CTI 中断 (IIR[3:1] = 110) 为第二优先级中断，在 UART RX FIFO 含有至少一个字符并且在接收到 3.5 到 4.5 个字符的时间内没有发生任何 UART RX FIFO 操作时，可对该中断进行设置。任何 UART RX FIFO 操作（UART RSR 的读取或写入）都可以清除该中断。当接收到的信息不是触发电平大小的倍数时，该中断将会刷新 UART RBR。例如：如果外设想要发送一个长度为 105 个字符的消息，而触发电平为 10 个字符，那么前 100 个字符将使 CPU 接收 10 个 RDA 中断，而剩下的 5 个字符使 CPU 收到 1 到 5 个 CTI 中断（取决于服务程序）。

表 169. UART 中断处理

IIR[3:0] 值 [1]	优先级	中断类型	中断源	中断复位
0001	-	无	无	-
0110	最高	RX 线状态 / 错误	OE [2] 、PE [2] 、FE [2] 、BI [2]	LSR 读操作 [2]
0100	第二	RX 数据可用	RX 数据可用或达到 FIFO 触发电平 (FCR0=1)	RBR 读操作 [3] 或 UART FIFO 低于触发电平
1100	第二	字符超时指示	RX FIFO 中至少有一个字符，并且在一段时间内没有字符输入或移出，该时间的长短取决于 FIFO 中的字符数以及在 3.5 到 4.5 字符的时间内的触发值。 实际时间为： $[(\text{字长度}) \times 7 - 2] \times 8 + [(\text{触发电平} - \text{字符数}) \times 8 + 1] \text{ RCLK}$	RBR 读操作 [3]
0010	第三	THRE	THRE [2]	IIR 读操作 [4] (如果为中断源) 或 THR 写操作

[1] “0000”、“0011”、“0101”、“0111”、“1000”、“1001”、“1010”、“1011”、“1101”、“1110”、“1111” 均为保留值。
[2] 有关详情，请参阅[章节 10.5.8 “UART 线路状态寄存器”](#)
[3] 有关详情，请参阅[章节 10.5.1 “UART 接收器缓冲寄存器（当 DLAB = 0 时，只读）”](#)
[4] 有关详情，请参阅[章节 10.5.5 “UART 中断识别寄存器”](#)和[章节 10.5.2 “UART 发送器保持寄存器（当 DLAB = 0 时，只写）”](#)

UART THRE 中断 (IIR[3:1] = 001) 为第三优先级中断，当 UART THR FIFO 为空，且满足特定的初始化条件时，该中断激活。这些初始化条件是为了让 UART THR FIFO 有机会填入数据，以免在系统启动时产生许多 THRE 中断。当 THRE = 1 时，且在上次的 THRE = 1 事件后，THR 中没有出现至少两个字符时，这些初始化条件就会实现一个字符减去停止位的延时。当没有解码和服务的 THRE 中断时，该延时为 CPU 提供了写数据到 THR 的时间。当 UART THR FIFO 中曾经同时出现两个或更多字符，而当前的 THR 为空时，THRE 中断就会立即被设置。当发生 THR 写操作或 IIR 读操作，且 THRE 为最高优先级中断 (IIR[3:1] = 001) 时，THRE 中断复位。

10.5.6 UART FIFO 控制寄存器

FCR 控制 UART RX 和 TX FIFO 的操作。

表 170. UART FIFO 控制寄存器（FCR - 地址 0x4000 C008，只写）位描述

位	符号	值	描述	复位值
0	FIFOEN		FIFO 使能	0
		0	UART FIFO 被禁用。禁止在应用中使用。	
		1	高电平有效，启用对 UART RX、TX FIFO 和 FCR[7:1] 的访问。该位必须进行设置，以实现正确的 UART 操作。该位的任何变化都会自动清除 UART FIFO。	
1	RXFIFO RES		RX FIFO 复位	0
		0	对两个 UART FIFO 均无影响。	
		1	写逻辑 1 到 FCR[1] 将会清除 UART RX FIFO 中的所有字节，复位指针逻辑。该位可以自动清零。	

表 170. UART FIFO 控制寄存器（FCR - 地址 0x4000 C008，只写）位描述

位	符号	值	描述	复位值
2	TXFIFO RES		TX FIFO 复位	0
		0	对两个 UART FIFO 均无影响。	
		1	写逻辑 1 到 FCR[2] 将会清除 UART TX FIFO 中的所有字节，复位指针逻辑。该位可以自动清零。	
3	DMAMO DE		DMA 模式选择。当对 FIFO 使能位（该寄存器的位 0）进行设置 0 时，该位选择 DMA 模式。参见 章节 10.5.6.1 。	0
		0	DMA 未使用。	
		1	DMA 模式已启用。	
5:4	-		保留，用户软件不应对保留位写入 1。从保留位读取的值未定义。	不适用
7:6	RXTL		RX 触发电平。这两个位决定了接收器 UART FIFO 在激活中断前必须写入的字符数量。	0
		0x0	触发电平 0（1 个字符或 0x01）。	
		0x1	触发电平 1（4 个字符或 0x04）。	
		0x2	触发电平 2（8 个字符或 0x08）。	
		0x3	触发电平 3（14 个字符或 0x0E）。	
31:8	-	-	保留	-

10.5.6.1 DMA 操作

通过使用微型 DMA，用户可选择操作 UART 的发送和 / 或接收。DMA 模式由 FCR 寄存器中的 DMA 模式选择位决定。只有通过 FCR 寄存器中的 FIFO 使能位将 FIFO 启用时，该位才会有效。

10.5.6.1.1 UART 接收器 DMA

在 DMA 模式中，当接收器 FIFO 的电平等于或大于触发电平时，或者在发生字符超时的情况下，接收器 DMA 请求就会生效。请参考上文对 RX 触发电平的描述。接收器 DMA 请求由 DMA 控制器清除。

10.5.6.1.2 UART 发送器 DMA

在 DMA 模式中，当发送器 FIFO 变为未滿时，发送器 DMA 请求就会生效。发送器 DMA 请求由 DMA 控制器清除。

10.5.7 UART 线路控制寄存器

LCR 决定要发送或接收的数据字符的格式。

表 171. UART 线路控制寄存器（LCR - 地址 0x4000 C00C）位描述

位	符号	值	描述	复位值
1:0	WLS		字长度选择	0
		0x0	5 位字符长度。	
		0x1	6 位字符长度。	
		0x2	7 位字符长度。	
		0x3	8 位字符长度。	
2	SBS		停止位选择	0
		0	1 个停止位。	
		1	2 个停止位（如果 LCR[1:0]=00，则为 1.5 个停止位）。	

表 171. UART 线路控制寄存器（LCR - 地址 0x4000 C00C）位描述（续）

位	符号	值	描述	复位值
3	PE		奇偶检验使能	0
		0	禁用奇偶检验的生成和检查。	
		1	启用奇偶检验的生成和检查。	
5:4	PS		奇偶检验选择	0
		0x0	奇检验。1s 内发送字符数和附加奇偶检验位为奇数。	
		0x1	偶检验。1s 内发送字符数和附加奇偶检验位为偶数。	
		0x2	强制 1 奇偶检验。	
		0x3	强制 0 奇偶检验。	
6	BC		间隔控制	0
		0	禁用间隔传输。	
		1	启用间隔传输。当 LCR[6] 为高电平有效时，输出引脚 UART TXD 强制为逻辑 0。	
7	DLAB		除数锁存器访问位 (DLAB)	0
		0	禁用对除数锁存器的访问。	
		1	启用对除数锁存器的访问。	
31:8	-	-	保留	-

10.5.8 UART 线路状态寄存器

LSR 为只读寄存器，提供 UART TX 和 RX 模块的状态信息。

表 172. UART 线路状态寄存器（LSR - 地址 0x4000 C014，只读）位描述

位	符号	值	描述	复位值
0	RDR		接收器数据就绪。当 RBR 包含未读字符时，LSR[0] 会被设置；当 UART RBR FIFO 为空时，LSR[0] 会被清零。	0
		0	RBR 为空。	
		1	RBR 包含有效数据。	
1	OE		溢出错误。溢出错误条件在错误发生后立即设置。LSR 读操作会将 LSR[1] 清零。当 UART RSR 有了新的字符组合，且 UART RBR FIFO 已满时，LSR[1] 会被设置。在这种情况下，UART RBR FIFO 不会被覆盖，而 UART RSR 中的字符将会丢失。	0
		0	溢出错误状态无效。	
		1	溢出错误状态有效。	
2	PE		奇偶检验错误。当已接收字符的奇偶检验位处于错误状态时，就会发生奇偶检验错误。LSR 读操作会将 LSR[2] 清零。奇偶检验错误检测时间取决于 FCR[0]。 注：奇偶检验错误与 UART RBR FIFO 顶部的字符相关。	0
		0	奇偶检验错误状态无效。	
		1	奇偶检验错误状态有效。	

表 172. UART 线路状态寄存器（LSR - 地址 0x4000 C014，只读）位描述（续）

位	符号	值	描述	复位值
3	FE		成帧错误。当已接收字符的停止位为逻辑 0 时，会发生成帧错误。LSR 读操作会将 LSR[3] 清零。成帧错误检测时间取决于 FCR0。当检测到有成帧错误时，RX 会尝试与数据重新同步，并假设错误的停止位实际是一个超前的起始位。但是，即使没有出现成帧错误，也无法假设下一个接收到的字节就是正确的。 注：成帧错误与 UART RBR FIFO 顶部的字符相关。	0
		0	成帧错误状态无效。	
		1	成帧错误状态有效。	
4	BI		间隔中断。在发送整个字符（起始、数据、奇偶检验、停止）的过程中，如果 RXD1 保持在间隔状态（全“0”），则会发生间隔中断。一旦检测到间隔条件，接收器会立即进入空闲状态，直到 RXD1 进入标记状态（全“1”）。LSR 读操作会将该状态位清零。间隔检测的时间取决于 FCR[0]。 注：间隔中断与 UART RBR FIFO 顶部的字符相关。	0
		0	间隔中断状态无效。	
		1	间隔中断状态有效。	
5	THRE		发送器保持寄存器为空。当检测到 UART THR 为空时，THRE 就会立即被设置；写 THR 时，THRE 就会清零。	1
		0	THR 包含有效数据。	
		1	THR 为空。	
6	TEMT		发送器为空。当 THR 和 TSR 同时为空时，TEMT 就会被设置；TSR 或 THR 任意一个包含有效数据时，TEMT 就会被清零。	1
		0	THR 和 / 或 TSR 包含有效数据。	
		1	THR 和 TSR 为空。	
7	RXFE		RX FIFO 错误。当一个带有 RX 错误（如：帧错误、奇偶错误或间隔中断）的字符载入到 RBR 时，LSR[7] 就会被设置。当 LSR 寄存器被读取，且 UART FIFO 中没有后续错误时，此位会被清除。	0
		0	RBR 不包含 UART RX 错误或 FCR[0]=0。	
		1	UART RBR 包含至少一个 UART RX 错误。	
31:8	-	-	保留	-

10.5.9 UART 高速暂时寄存器

SCR 不影响 UART 操作。用户可自由对该寄存器进行读和 / 或写操作。不提供中断接口向主机指示 SCR 所发生的读或写操作。

表 173. UART 高速暂时寄存器（SCR - 地址 0x4000 C01C）位描述

位	符号	描述	复位值
7:0	焊盘	一个可读、可写的字节。	0x00
31:8	-	保留	-

10.5.10 UART 自动波特率控制寄存器

在用户测量波特率生成器的输入时钟 / 数据速率期间，整个测量过程就是由 UART 自动波特率控制寄存器（ACR）进行控制的。用户可自由地对该寄存器进行读写操作。

表 174. 自动波特率控制寄存器（ACR - 地址 0x4000 C020）位描述

位	符号	值	描述	复位值
0	START		在自动波特率功能结束后，系统会清除此位。	0
		0	自动波特率停止（自动波特率未运行）。	
		1	自动波特率启动（自动波特率正在运行）。自动波特率运行位。在自动波特率功能结束后，系统会清除此位。	
1	MODE		自动波特率模式选择位。	0
		0	模式 0。	
		1	模式 1。	
2	AUTORESTART	0	不重新启动	0
		1	如果超时则重新启动（计数器会在下一个 UART RX 下降沿重新启动）	
7:3	-		保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	0
8	ABEOINTCLR		自动波特率中断结束清除位（仅可写访问）	0
		0	写 0 无影响。	
		1	写 1 会清除 IIR 中相应的中断。	
9	ABTOINTCLR		自动波特率超时中断清除位（仅可写访问）	0
		0	写 0 无影响。	
		1	写 1 会清除 IIR 中相应的中断。	
31:10	-	不适用	保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	0

10.5.10.1 自动波特率

UART 自动波特率功能可用于测量基于“AT”协议（Hayes 命令）的输入波特率。如果启用了自动波特率功能，那么此功能将测量接收数据流的位时间，并根据这个结果来设置除数锁存器寄存器 DLM 和 DLL。

自动波特率可通过设置 ACR 起始位来启动。自动波特率可通过清零 ACR 起始位来停止。自动波特率一旦结束，起始位将自动清零，并且读取该位将会返回自动波特率的状态（挂起 / 完成）。

可通过 ACR 模式位来选择所提供的两种自动波特率测量模式。在模式 0 下，波特率是通过在对 UART RX 管脚上两个连续的下降沿进行测量（起始位的下降沿和最低有效位的下降沿）来得到的。而在模式 1 下，波特率则是通过测量 UART RX 引脚上的下降沿和后续的上升沿之间的时间（起始位的长度）来得到的。

如果出现超时（速率测量计数器溢出），ACR AutoRestart 位可用于自动重启波特率测量。如果该位被置位，速率测量将会在 UART RX 引脚的下一个下降沿重新启动。

自动波特率功能会产生两种中断：

- 若中断使能，则 IIR ABTOInt 中断将置位（IER ABTOIntEn 置位且自动波特率测量寄存器溢出）。
- 若中断使能，则 IIR ABEOInt 中断将置位（IER ABEOIntEn 置位且自动波特率已经成功完成）。

自动波特率中断必须通过置位相应的 ACR ABTOIntClr 位和 ABEOIntEn 位来清零。

在自动波特率期间，小数波特率发生器通常被禁用（即 $DIVADDVAL = 0$ ）。但是，如果小数波特率发生器被启用（即 $DIVADDVAL > 0$ ），那么它将影响 UART RX 引脚波特率的测量，但 FDR 寄存器的值在速率测量后不会被修改。此外，当使用自动波特率时，任何对 DLM 和 DLL 寄存器的写操作都必须在写 ACR 寄存器之前完成。UART 支持的最小和最大波特率受 UART_PCLK、数据的位数、停止位和奇偶检验位的影响。

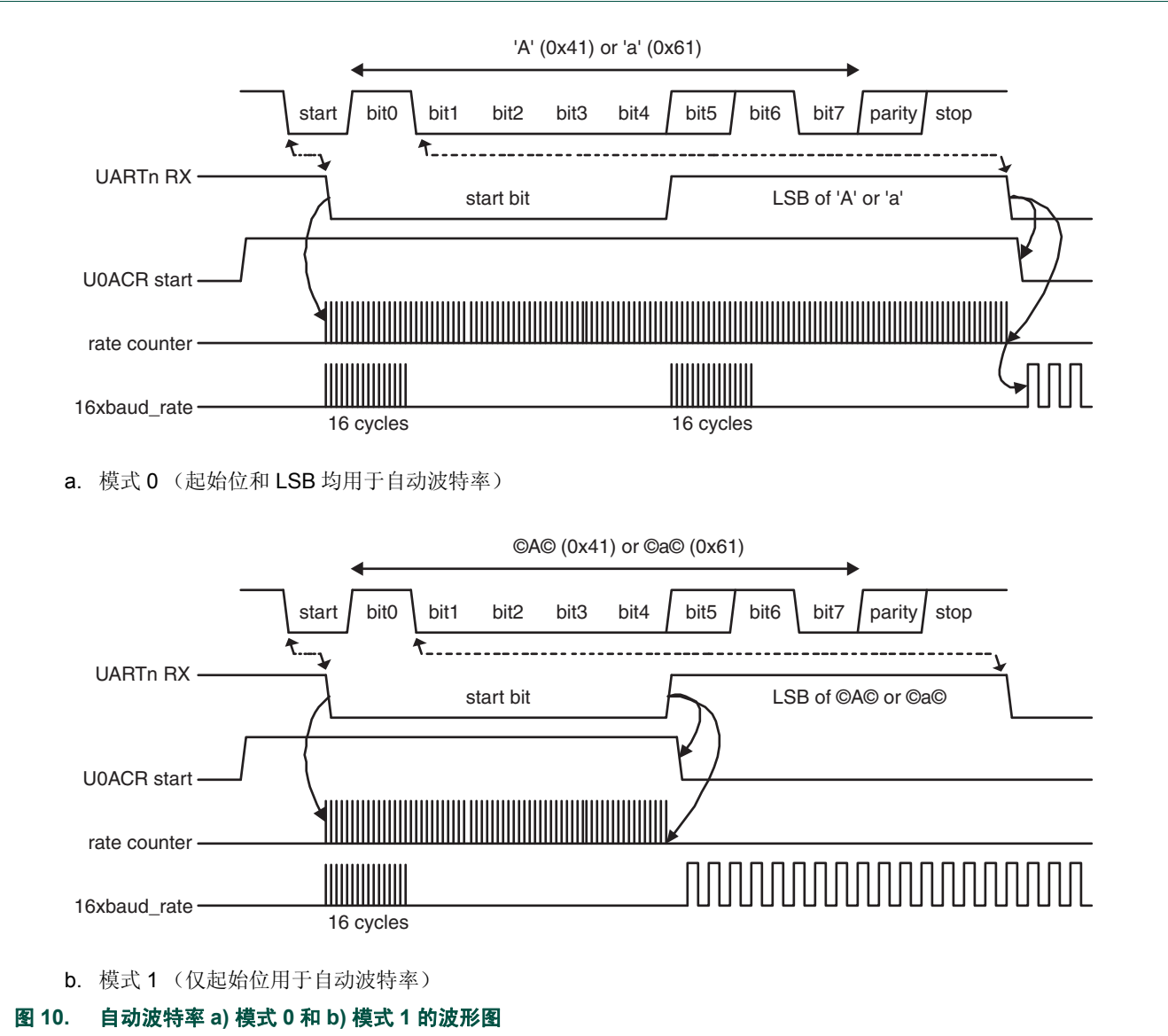
(4)

$$ratemin = \frac{2 \times PCLK}{16 \times 2^{15}} \leq UART_{baudrate} \leq \frac{PCLK}{16 \times (2 + databits + paritybits + stopbits)} = ratemax$$

10.5.10.2 自动波特率模式

当软件执行“AT”命令时，它采用期望的字符格式对 UART 进行配置，并置位 ACR 起始位。用户不必关心除数锁存器 DLM 和 DLL 的初始值。由于字母“A”或“a”ASCII 编码（“A”= 0x41，“a”= 0x61）的关系，UART RX 引脚所检测的起始位以及期望字符的 LSB 是由两个下降沿来限定的。当 ACR 起始位被置位时，自动波特率协议将执行以下步骤：

1. ACR 起始位一旦置位，波特率测量计数器立即复位，同时 UART RSR 复位。RSR 波特率切换为最高速率。
2. UART RX 引脚上的下降沿触发起始位的开始。波特率测量计数器将开始对 UART_PCLK 周期（可选择被小数波特率发生器预分频）进行计数。
3. 在接收起始位的过程中，RSR 波特率输入端会产生 16 个脉冲，脉冲频率和（小数波特率发生器预分频）UART 输入时钟相同，这样，保证了起始位存放在 RSR 中。
4. 在接收起始位的过程中（以及模式 0 下的字符 LSB），速率计数器将随着被预分频的 UART 输入时钟 (UART_PCLK) 递增。
5. 如果是模式 0，那么速率计数器将在 UART RX 引脚的下一个下降沿停止。如果是模式 1，那么速率计数器将在 UART RX 引脚的下一个上升沿停止。
6. 速率计数器的值被载入到 DLM/DLL 中，并且波特率将会切换到正常操作模式。在设置完 DLM/DLL 后，如果自动波特率结束中断 IIR ABEOInt 被启用，IIR ABEOInt 将会被置位。接着，RSR 继续接收“A/a”字符剩下的其它位。



10.5.11 UART IrDA 控制寄存器

IrDA 控制寄存器可在每个 UART 上启用和配置 IrDA 模式。在发送或接收数据时，不应更改 ICR 的值，否则会造成数据丢失或损坏。

表 175. UART IrDA 控制寄存器（ICR - 0x4000 C024）位描述

位	符号	值	描述	复位值
0	IRDAEN		IrDA 模式使能。	0
		0	UARTn 上的 IrDA 模式禁用，UARTn 用作标准的 UART。	
		1	UARTn 上的 IrDA 模式启用。	
1	IRDAINV		IrDA 输入极性。	0
		0	串行输入不会反转。	
		1	串行输入反转。此位对串行输出无效。	

表 175. UART IrDA 控制寄存器（ICR - 0x4000 C024）位描述（续）

位	符号	值	描述	复位值
2	FIXPULSEEN		IrDA 固定脉冲宽度模式。	0
		0	IrDA 固定脉冲宽度模式禁用。	
		1	IrDA 固定脉冲宽度模式启用。	
5:3	PULSEDIV		当 FixPulseEn = 1 时，对脉冲进行配置。	0
		0x0	$2 \times T_{PCLK}$	
		0x1	$4 \times T_{PCLK}$	
		0x2	$8 \times T_{PCLK}$	
		0x3	$16 \times T_{PCLK}$	
		0x4	$32 \times T_{PCLK}$	
		0x5	$64 \times T_{PCLK}$	
		0x6	$128 \times T_{PCLK}$	
		0x7	$256 \times T_{PCLK}$	
31:6	-	-	保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	0

如果在 IrDA 模式下使用了固定脉冲宽度模式（IrDAEn = 1 且 FixPulseEn = 1），ICR 中的 PulseDiv 位用于选择脉冲宽度。用户必须对这些位的值进行设置，以便得到的脉冲宽度至少为 1.63 μs。[表 176](#) 显示了可能的脉冲宽度。

表 176. IrDA 脉冲宽度

FixPulseEn	PulseDiv	IrDA 发送器脉冲宽度 (μs)
0	x	3 / (16 波特率)
1	0	$2 \times T_{PCLK}$
1	1	$4 \times T_{PCLK}$
1	2	$8 \times T_{PCLK}$
1	3	$16 \times T_{PCLK}$
1	4	$32 \times T_{PCLK}$
1	5	$64 \times T_{PCLK}$
1	6	$128 \times T_{PCLK}$
1	7	$256 \times T_{PCLK}$

10.5.12 UART 小数分频寄存器

UART 小数分频寄存器 (FDR) 控制波特率生成器的时钟预分频器，并且用户可自由对该寄存器进行读写操作。该预分频器使用 APB 时钟并根据指定的小数要求产生输出时钟。

注意事项：如果小数分频器有效 (DIVADDVAL > 0)，且 DLM = 0，则 DLL 寄存器的值必须大于或等于 3。

表 177. UART 小数分频寄存器（FDR - 地址 0x4000 C028）位描述

位	符号	描述	复位值
3:0	DIVADDVAL	波特率发生器预分频器除数值。如果该字段为 0，则小数波特率生成器将不会影响 UARTn 的波特率。	0
7:4	MULVAL	波特率预分频器乘数值。不管是否使用小数波特率发生器，为了让 UARTn 正常运作，该字段必须大于或等于 1。	1
31:8	-	保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	0

该寄存器控制波特率发生器的时钟预分频器。寄存器的复位值会让 UART 的小数功能保持在禁用状态，从而确保 UART 在软件和硬件方面能够与不具备该特性的 UART 完全兼容。

UART 波特率可以按下列公式计算 (n = 1):

(5)

$$UART_{baudrate} = \frac{PCLK}{16 \times (256 \times UIDLM + UIDLL) \times \left(1 + \frac{DivAddVal}{MulVal}\right)}$$

其中，UART_PCLK 为外设时钟，DLM 和 DLL 为标准 UART 波特率分频寄存器，而 DIVADDVAL 和 MULVAL 为 UART 小数波特率发生器的特定参数。

MULVAL 和 DIVADDVAL 的值应符合以下条件：

- 1. 1 ≤ MULVAL ≤ 15
- 2. 0 ≤ DIVADDVAL < 15
- 3. DIVADDVAL < MULVAL

FDR 的值在发送 / 接收数据的过程中不应进行更改，否则可能会导致数据丢失或损坏。

如果 FDR 寄存器值不满足上述两个要求，那么小数分频器输出则为未定义。如果 DIVADDVAL 为 0，那么小数分频器将被禁用，并且不会对时钟进行分频。

10.5.12.1 波特率计算

UART 可以与小数分频器一起工作，也可以不使用小数分频器。在实际应用中，使用几种不同的小数分频器设置很可能会获得期望的波特率。下面的算法描绘了查找一组 DLM、DLL、MULVAL 以及 DIVADDVAL 值的方法。这组参数产生的波特率与期望值的相对误差小于 1.1%。

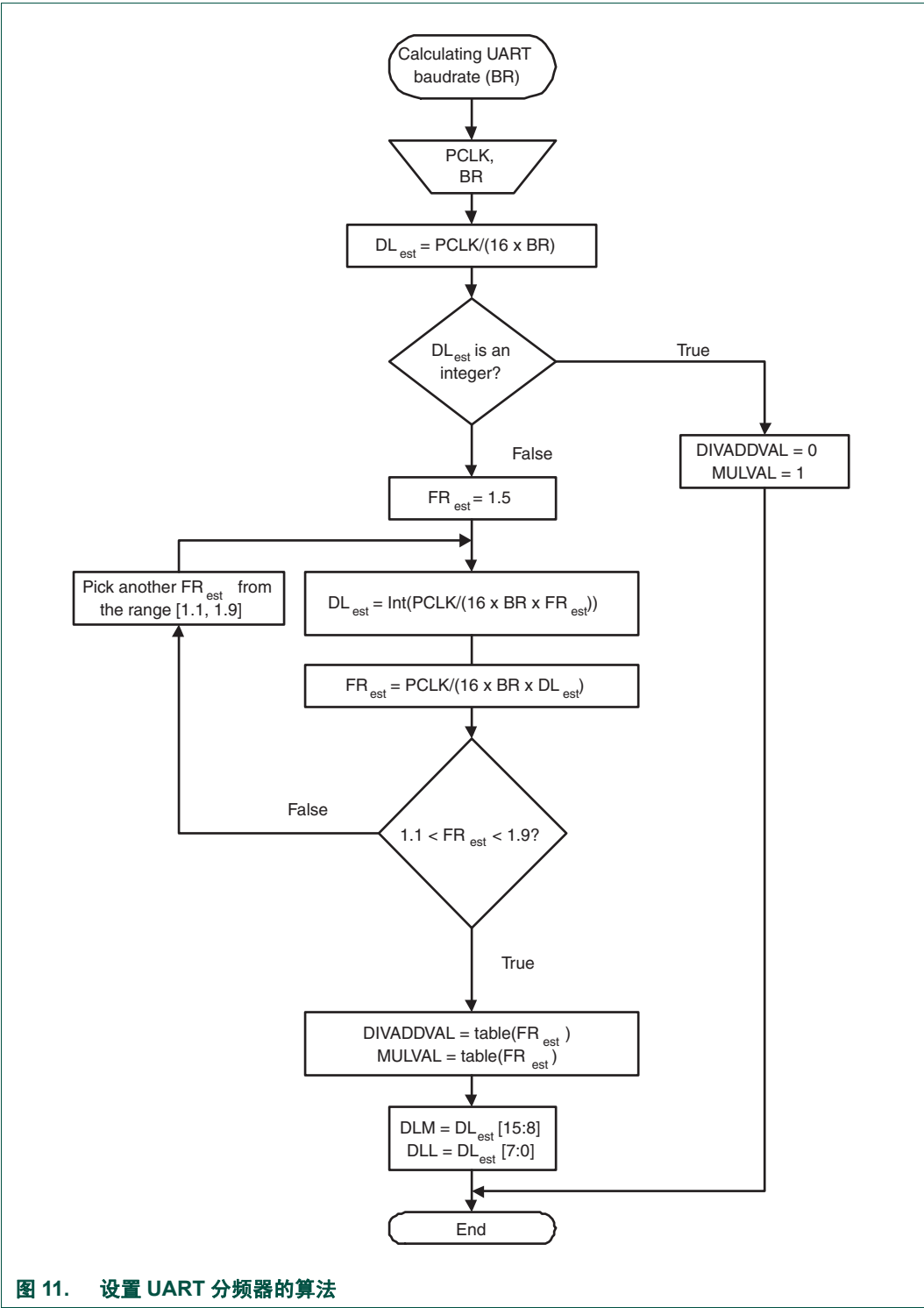


图 11. 设置 UART 分频器的算法

表 178. 小数分频器设置查找表

FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal
1.000	0/1	1.250	1/4	1.500	1/2	1.750	3/4
1.067	1/15	1.267	4/15	1.533	8/15	1.769	10/13
1.071	1/14	1.273	3/11	1.538	7/13	1.778	7/9
1.077	1/13	1.286	2/7	1.545	6/11	1.786	11/14
1.083	1/12	1.300	3/10	1.556	5/9	1.800	4/5
1.091	1/11	1.308	4/13	1.571	4/7	1.818	9/11
1.100	1/10	1.333	1/3	1.583	7/12	1.833	5/6
1.111	1/9	1.357	5/14	1.600	3/5	1.846	11/13
1.125	1/8	1.364	4/11	1.615	8/13	1.857	6/7
1.133	2/15	1.375	3/8	1.625	5/8	1.867	13/15
1.143	1/7	1.385	5/13	1.636	7/11	1.875	7/8
1.154	2/13	1.400	2/5	1.643	9/14	1.889	8/9
1.167	1/6	1.417	5/12	1.667	2/3	1.900	9/10
1.182	2/11	1.429	3/7	1.692	9/13	1.909	10/11
1.200	1/5	1.444	4/9	1.700	7/10	1.917	11/12
1.214	3/14	1.455	5/11	1.714	5/7	1.923	12/13
1.222	2/9	1.462	6/13	1.727	8/11	1.929	13/14
1.231	3/13	1.467	7/15	1.733	11/15	1.933	14/15

10.5.12.1.1 示例 1: UART_PCLK = 14.7456 MHz， BR = 9600

根据所提供的算法， $DL_{est} = PCLK / (16 \times BR) = 14.7456 \text{ MHz} / (16 \times 9600) = 96$ 。因为这里的 DL_{est} 是一个整数，所以 $DIVADDVAL = 0$ ， $MULVAL = 1$ ， $DLM = 0$ ，且 $DLL = 96$ 。

10.5.12.1.2 示例 2: UART_PCLK = 12 MHz， BR = 115200

根据所提供的算法， $DL_{est} = PCLK / (16 \times BR) = 12 \text{ MHz} / (16 \times 115200) = 6.51$ 。该算式中的 DL_{est} 并不是整数，因此下一步就要对 FR 参数进行估算。使用 $FR_{est} = 1.5$ 进行首次估算，得到新的 $DL_{est} = 4$ ，然后使用 $FR_{est} = 1.628$ 再进行计算。由于 $FRest = 1.628$ 是在 1.1 到 1.9 的指定范围之内，因此 $DIVADDVAL$ 和 $MULVAL$ 的值可通过附带的查找表获得。

在查找表表 178 中，最接近 $FRest = 1.628$ 的值为 $FR = 1.625$ 。也就是说 $DIVADDVAL = 5$ 而 $MULVAL = 8$ 。

基于这些查找结果，建议 UART 设置为： $DLM = 0$ ， $DLL = 4$ ， $DIVADDVAL = 5$ 和 $MULVAL = 8$ 。根据方程 5，UART 的波特率为 115384。该速率与原来指定的 115200 之间存在 0.16% 的相对误差。

10.5.13 UART 发送使能寄存器

除了配备完整的硬件流控制（上述的自动 CTS 和自动 RTS 机制）之外，TER 还可以实现软件流控制。当 $TXEn = 1$ 时，只要数据可用，UART 发送器就会一直发送数据。一旦 $TXEn$ 变为 0，UART 就会停止数据传输。

虽然表 179 描述了如何利用 $TXEn$ 位来实现硬件流控制，但我们强烈建议用户采用 UART 硬件所实现的自动流控制特性处理硬件流控制，并限制 $TXEn$ 对软件流控制的范围。

TER 可实现软件和硬件流控制。当 TXEn = 1 时，只要数据可用，UART 发送器就会一直发送数据。一旦 TXEn 变为 0，UART 就会停止数据传输。

表 179 描述了如何利用 TXEn 位来实现软件流控制。

表 179. UART 发送使能寄存器（TER - 地址 0x4000 C030）位描述

位	符号	描述	复位值
6:0	-	保留，用户软件不应保留位写入 1。从保留位读取的值未定义。	不适用
7	TXEN	该位为 1 时（复位后），一旦先前的数据都被发送后，写入 THR 1 的数据就会在 TXD 引脚上输出。如果在发送某字符时该位被清零，那么在将该字符发送完毕后就不再发送字符，直到该位再次被置位。也就是说，该位为 0 时会阻止字符从 THR 或 TX FIFO 传输到发送移位寄存器。当接收到 XOFF 字符 (DC3) 时，软件可以将该位清零。当接收到 XON (DC1) 字符时，软件可以重新设置该位。	1
31:8	-	保留	-

10.5.14 UART FIFO 电平寄存器

FIFOLVL 寄存器是一个只读寄存器，允许软件读取 FIFO 的当前电平状态。发送和接收 FIFO 的电平均呈现在该寄存器中。

表 180. UART FIFO 电平寄存器（FIFOLVL - 地址 0x4000 C058，只读）位描述

位	符号	描述	复位值
3:0	RXFIFILVL	反映 UART 接收器 FIFO 的当前电平。 0 = 空，0xF = FIFO 满。	0x00
7:4	-	保留。从保留位读取的值未定义。	-
11:8	TXFIFOLVL	反映 UART 发送器 FIFO 的当前电平。 0 = 空，0xF = FIFO 满。	0x00
31:12	-	保留。从保留位读取的值未定义。	不适用

10.6 架构

UART 的架构如下面的功能框图所示。

APB 接口提供 CPU 或主机与 UART 之间的通信链路。

UART 接收器模块 RX 监视串行输入线 RXD 的有效输入。UART RX 移位寄存器 (RSR) 通过 RXD 接受有效的字符。当 RSR 中接收到一个有效字符后，它将该字符传送到 UART RX 缓冲寄存器 FIFO 中，等待 CPU 或主机通过通用主机接口进行访问。

UART 发送器模块 TX 接受 CPU 或主机写入的数据并将数据缓冲到 UART TX 保持寄存器 FIFO(THR) 中。UART TX 移位寄存器 (TSR) 读取存储在 THR 中的数据，并将数据通过串行输出引脚 TXD1 发送。

UART 波特率发生器模块 BRG 产生 UART TX 模块所使用的定时使能。BRG 时钟输入源为 UART_PCLK。主时钟按照 DLL 和 DLM 寄存器中所指定的除数进行分频。分频的时钟为 16 倍过采样时钟 NBAUDOUT。

中断接口包含寄存器 IER 和 IIR。中断接口接收若干个由 TX 和 RX 模块发出的单时钟宽度的使能信号。

TX 和 RX 的状态信息保存在 LSR 中。TX 和 RX 的控制信息保存在 LCR 中。

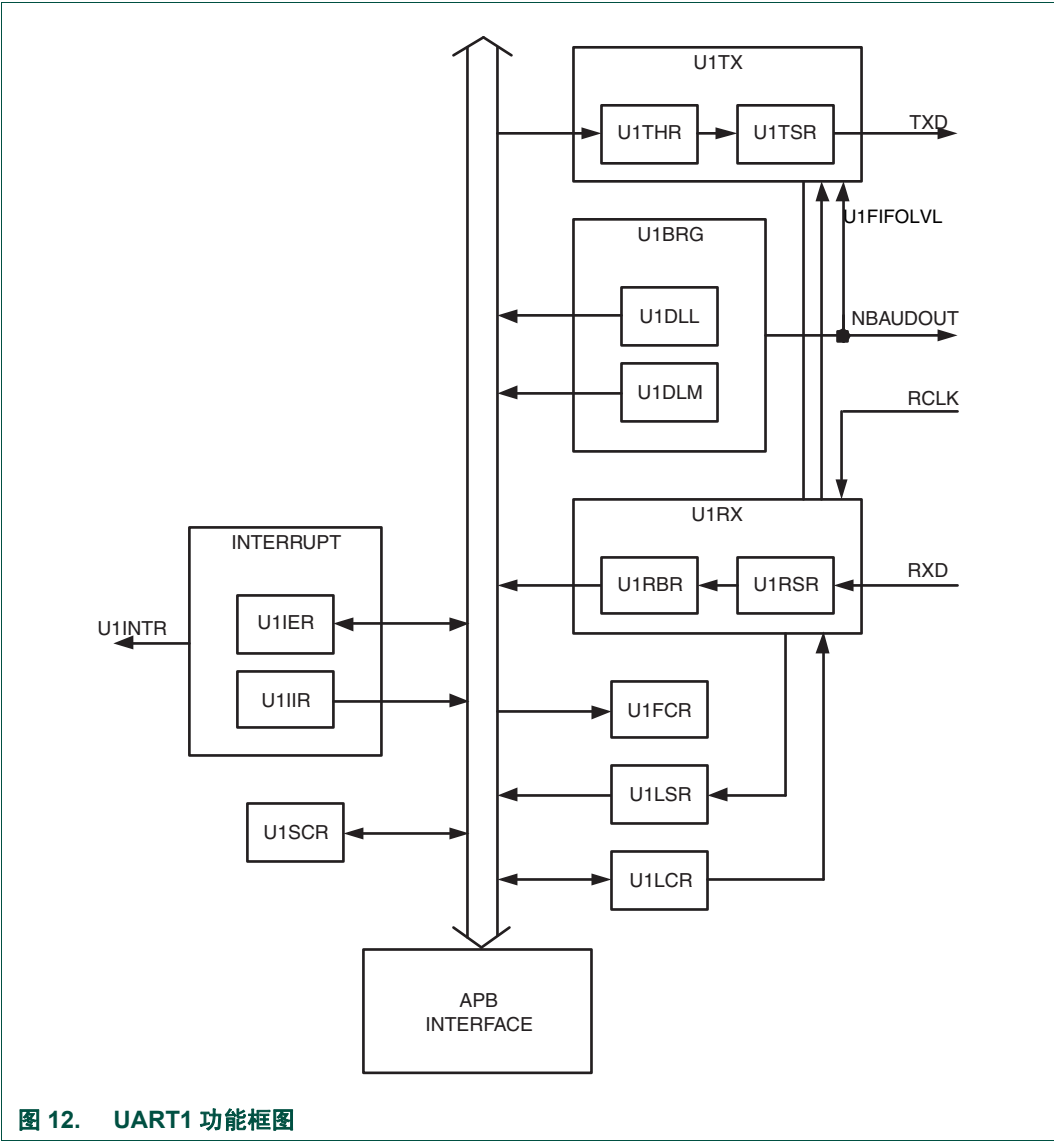


图 12. UART1 功能框图

11.1 本章导读

所有 LPC122x 器件都提供 I²C 总线控制器。

11.2 基本配置

I²C 模块 I2C_PCLK 的外设时钟由系统时钟提供，而系统时钟由 SYSAHBCLKDIV 寄存器控制（表 21）。I²C 模块可以通过系统 AHB 时钟控制寄存器位 5（表 21）禁用以降低功耗。

11.3 特性

- 标准 I²C 兼容总线接口，可配置为主机、从机或主 / 从机。
- 在同时发送的主机之间进行仲裁，从而避免总线上的串行数据的讹误。
- 可编程时钟允许调整 I²C 传输速率。
- 主机和从机之间的数据传输是双向的。
- 串行时钟同步使得比特率不同的器件能够通过一条串行总线进行通信。
- 串行时钟同步可用作一种反馈检验机制来挂起和恢复串行传输。
- 支持超快速模式。
- 可识别多达 4 个不同的从属地址。
- 监控模式可观察所有的 I²C 总线流量，而不用考虑从属地址。
- I²C 总线可用于测试和诊断。
- I²C 块包含一个带有 2 个引脚的标准 I²C 兼容总线接口。

11.4 应用

与外部 I²C 标准器件相连接，如串行 RAM、LCD、音频发生器和其它微控制器等。

11.5 描述

典型的 I²C 总线配置如图 13 所示。根据方向位的状态 (R/W)，I²C 总线上可能存在以下两种类型的数据传输方式：

- 由主发送器向从接收器传输数据。主机发送的第一个字节是从属地址。接下来是数据字节数。从机每接收一个字节后返回一个应答位。

- 由从发送器向主接收器传输数据。由主机发送第一个字节（从属地址）。然后从机返回一个应答位。接下来是由从机发送数据字节到主机。主机接收到所有字节（最后一个字节除外）后返回一个应答位。接收到最后一个字节后，主机返回“非应答”位。主机设备产生所有的串行时钟脉冲及起始和停止条件。以停止或重复起始条件结束传输。由于重复起始条件也是下一次串行传输的开始，因此不释放 I2C 总线。

I2C 接口是字节导向型，有 4 个操作模式：主发送模式、主接收模式、从发送模式及从接收模式。

I2C 接口遵循整个 I2C 规范，支持在不影响同一 I2C 总线上其它器件的情况下关闭 ARM Cortex-M0。

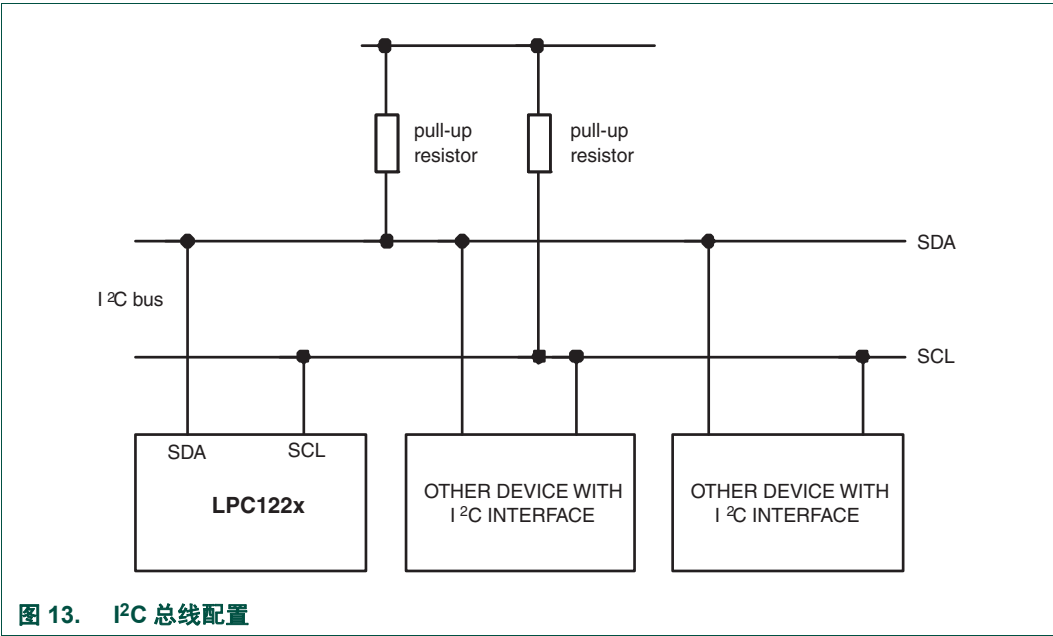


图 13. I2C 总线配置

11.5.1 I2C 超快速模式

超快速模式支持以 1 Mbit/ 秒的传输速率与恩智浦半导体现在所提供的 I2C 产品通信。

要使用超快速模式，就必须在 IOCONFIG 寄存器块正确配置 I2C 引脚，见表 96 和表 97。在超快速模式中，可选择的速率在 400 kHz 以上，高达 1MHz，见表 189。

11.6 引脚描述

表 181. I2C 总线引脚描述

引脚	类型	描述
SDA	输入 / 输出	I2C 串行数据
SCL	输入 / 输出	I2C 串行时钟

I2C 总线引脚必须通过 IOCON_PIO0_10（表 96）和 IOCON_PIO0_11 寄存器（表 97）配置，以用于标准 / 快速模式或超快速模式。在这些模式下，I2C 总线引脚为开漏输出并且完全兼容 I2C 总线规范。

11.7 寄存器描述

表 182. 寄存器简介：I²C（基址 0x4000 0000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
CONSET	R/W	0x000	I²C 控制置位寄存器。 当向该寄存器的位写 1 时，I ² C 控制寄存器中的相应位置位。写 0 时对 I ² C 控制寄存器的相应位没有影响。	0x00
STAT	RO	0x004	I²C 状态寄存器。 在 I ² C 工作期间，该寄存器提供详细的状态码，允许软件决定需要执行的下一步操作。	0xF8
DAT	R/W	0x008	I²C 数据寄存器。 在主 / 从发送模式期间，要发送的数据写入该寄存器。在主 / 从接收模式期间，可从该寄存器读出已接收的数据。	0x00
ADR0	R/W	0x00C	I²C 从属地址寄存器 0。 包含 7 位从属地址，用于从机模式下 I ² C 接口操作，不用于主机模式下。最低有效位决定从机是否对通用调用地址作出响应。	0x00
SCLH	R/W	0x010	SCH 占空比寄存器高半字。 决定 I ² C 时钟的高电平时间。	0x04
I2SCLL	R/W	0x014	SCL 占空比寄存器低半字。 决定 I ² C 时钟低电平时间。I2nSCLL 和 I2nSCLH 一起决定 I ² C 主机产生的时钟频率及从机模式下所用的时间。	0x04
CONCLR	WO	0x018	I²C 控制清零寄存器。 当向该寄存器的位写 1 时，I ² C 控制寄存器中的相应位清零。写 0 时对 I ² C 控制寄存器的相应位没有影响。	不适用
MMCTRL	R/W	0x01C	监控模式控制寄存器。	0x00
ADR1	R/W	0x020	I²C 从属地址寄存器 1。 包含 7 位从属地址，用于从机模式下 I ² C 接口操作，不用于主机模式下。最低有效位决定从机是否对通用调用地址作出响应。	0x00
ADR2	R/W	0x024	I²C 从属地址寄存器 2。 包含 7 位从属地址，用于从机模式下 I ² C 接口操作，不用于主机模式下。最低有效位决定从机是否对通用调用地址作出响应。	0x00
ADR3	R/W	0x028	I²C 从属地址寄存器 3。 包含 7 位从属地址，用于从机模式下 I ² C 接口操作，不用于主机模式下。最低有效位决定从机是否对通用调用地址作出响应。	0x00
DATA_BUFFER	RO	0x02C	数据缓冲寄存器。 每次从总线接收到 9 个位（8 位数据和 ACK 或 NACK）后，I2DAT 移位寄存器的高 8 位的内容将自动传输到 DATA_BUFFER。	0x00
MASK0	R/W	0x030	I²C 从属地址屏蔽寄存器 0。 该屏蔽寄存器与 I2ADR0 一起决定地址匹配。当与通用调用地址（‘0000000’）比较时，屏蔽寄存器不起作用。	0x00
MASK1	R/W	0x034	I²C 从属地址屏蔽寄存器 1。 该屏蔽寄存器与 I2ADR0 一起决定地址匹配。当与通用调用地址（‘0000000’）比较时，屏蔽寄存器不起作用。	0x00
MASK2	R/W	0x038	I²C 从属地址屏蔽寄存器 2。 该屏蔽寄存器与 I2ADR0 一起决定地址匹配。当与通用调用地址（‘0000000’）比较时，屏蔽寄存器不起作用。	0x00
MASK3	R/W	0x03C	I²C 从属地址屏蔽寄存器 3。 该屏蔽寄存器与 I2ADR0 一起决定地址匹配。当与通用调用地址（‘0000000’）比较时，屏蔽寄存器不起作用。	0x00

[1] 复位值只反映了使用位的值。不包括保留位的内容。

11.7.1 I²C 控制置位寄存器 (CONSET - 0x4000 0000)

I2CONSET 寄存器控制 I2CON 寄存器中位的置位，这些位控制 I²C 接口的操作。向该寄存器的位写 1 会使 I²C 控制寄存器中的相应位置位。写 0 无效。

表 183. I²C 控制置位寄存器（CONSET - 地址 0x4000 0000）位描述

位	符号	描述	复位值
1:0	-	保留。用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用
2	AA	声明应答标志。	
3	SI	I ² C 中断标志。	0
4	STO	停止标志。	0
5	STA	起始标志。	0

表 183. I²C 控制置位寄存器（CONSET - 地址 0x4000 0000）位描述（续）

位	符号	描述	复位值
6	I2EN	I ² C 接口使能。	0
7	-	保留。用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用
31: 8	-	保留	-

I2EN I²C 接口使能。当 I2EN 为 1 时，I²C 接口使能。可通过向 I2CONCLR 寄存器中的 I2ENC 位写 1 来清零 I2EN 位。当 I2EN 为 0 时，I²C 接口禁能。

当 I2EN 为“0”时，忽略 SDA 和 SCL 输入信号，I²C 块处于“不可寻址”的从状态，STO 位强制为“0”。

I2EN 不应用于暂时释放 I²C 总线，因为当 I2EN 复位时，I²C 总线状态丢失。应使用 AA 标志代替。

STA 是起始标志。该位置位时，I²C 接口进入主机模式并发送一个起始条件，如果已经处于主机模式，则发送一个重复起始条件。

当 STA 为 1 且 I²C 接口没有处于主机模式时，它将进入主机模式，校验总线并在总线空闲时产生一个起始条件。如果总线忙，则等待一个停止条件（将释放总线）并在延迟半个内部时钟发生器周期后发送一个起始条件。当 I²C 接口已经处于主机模式且已发送或接收了数据时，它会发送一个重复起始条件。STA 可在任意时间置位，包括 I²C 接口处于可寻址的从机模式时也可置位。

可通过向 I2CONCLR 寄存器中的 STAC 位写 1 来清零 STA。当 STA 为 0 时，不会产生起始条件或重复起始条件。

STA 和 STO 都置位时，如果接口处于主机模式下，则向 I²C 总线发送一个停止条件，然后再发送一个起始条件。如果 I²C 接口处于从模式，则产生内部停止条件，但不发送到总线上。

STO 是停止标志。在主机模式下，该位置位会使 I²C 接口发送一个停止条件，或在从机模式下从错误状态中恢复。当主机模式下 STO=1 时，向 I²C 总线发送停止条件。当总线检测到停止条件时，STO 自动清零。

从机模式下，置位 STO 位可从错误状态中恢复。这种情况下不向总线发送停止条件。硬件的表现就好像是接收到一个停止条件并切换到“不可寻址”的从接收模式。STO 标志由硬件自动清零。

SI 是 I²C 中断标志。当 I²C 状态改变时该位置位。但是，进入状态 F8 不会使 SI 置位，因为在那种情况下中断服务程序不起作用。

当 SI 置位时，SCL 线上的串行时钟低电平持续时间扩展，且串行传输被中止。当 SCL 为高时，它不受 SI 标志的状态影响。SI 必须通过软件复位，通过向 I2CONCLR 寄存器的 SIC 位写入 1 来实现。

AA 是应答标志位。当 AA 置 1 时，在 SCL 线的应答时钟脉冲内，出现下面的任意情况时都将返回一个应答信号（SDA 线为低电平）：

- 1. 接收到从属地址寄存器中的地址。
- 2. 当 I2ADR 中的通用调用位 (GC) 置位时，接收到通用调用地址。
- 3. 当 I²C 接口处于主接收模式时，接收到一个数据字节。
- 4. 当 I²C 接口处于可寻址的从接收模式时，接收到一个数据字节。

可通过向 I2CONCLR 寄存器中的 AAC 位写 1 来清零 AA 位。当 AA 位为 0 时，SCL 线上的应答时钟脉冲内出现下列情况时将返回一个无应答信号（SDA 为高电平）：

- 1. 当 I2C 接口处于主接收模式时，接收到一个数据字节。
- 2. 当 I2C 处于可寻址的从接收模式时，接收到一个数据字节。

11.7.2 I2C 状态寄存器 (STAT - 0x4000 0004)

每个 I2C 状态寄存器反映相应 I2C 接口的情况。I2C 状态寄存器为只读。

表 184. I2C 状态寄存器 (STAT - 0x4000 0004) 位描述

位	符号	描述	复位值
2:0	-	这些位未使用且一直为 0。	0
7:3	状态	这些位提供关于 I2C 接口的实际状态信息。	0x1F
31: 8	-	保留	-

3 个最低有效位始终为 0。作为一个字节时，状态寄存器内容表示一个状态码。有 26 种可能存在的状态码。当状态码为 0xF8 时，没有相关信息可用且 SI 位没有置位。其它所有 25 个状态码符合定义的 I2C 状态。当进入这些状态中的任一状态时，SI 位将置位。关于状态码的完整列表，参见表 200 ~ 表 203。

11.7.3 I2C 数据寄存器 (DAT - 0x4000 0008)

该寄存器包含要发送的数据或刚接收的数据。SI 位置位时，只有在该寄存器没有进行字节移位时，CPU 才可以对其进行读 / 写操作。只要 SI 位置位，I2DAT 中的数据就保持不变。I2DAT 中的数据总是从右向左移位：要发送的第一位是 MSB（位 7），接收到一个字节后，接收到数据的第一位放在 I2DAT 的 MSB 位。

表 185. I2C 数据寄存器 (DAT - 0x4000 0008) 位描述

位	符号	描述	复位值
7:0	数据	该寄存器保存已接收或将要发送的数据值。	0
31: 8	-	保留	-

11.7.4 I2C 从属地址寄存器 0 (ADR0- 0x4000 000C)

该寄存器可读 / 写，只有在 I2C 接口设置为从机模式时才可用。在主机模式下，该寄存器无效。I2ADR 的 LSB 为通用调用位。当该位置位时，可识别通用调用地址 (0x00)。

其中包含位 00x 的寄存器将被禁能且不与总线上的任意地址匹配。复位时该寄存器将清零到该禁能状态。

表 186. I2C 从属地址寄存器 0 (ADR0- 0x4000 000C) 位描述

位	符号	描述	复位值
0	GC	通用调用使能位。	0
7:1	地址	从机模式的 I2C 器件地址。	0x00
31: 8	-	保留	-

11.7.5 I²C SCL 高电平占空比寄存器和低电平占空比寄存器（SCLH - 0x4000 0010 和 I2SCLL- 0x4000 0014）

表 187. I²C SCL 高电平占空比寄存器（SCLH - 地址 0x4000 0010）位描述

位	符号	描述	复位值
15:0	SCLH	SCL 高电平周期选择	0x0004
31:16	-	保留	-

表 188. I²C SCL 低电平占空比寄存器 (SCLL - 0x4000 0014) 位描述

位	符号	描述	复位值
15:0	SCLL	SCL 低电平周期选择	0x0004
31:16	-	保留	-

11.7.5.1 选择适当的 I²C 数据速率和占空比

软件必须设定寄存器 I2SCLH 和 I2SCLL 的值以选择适当的数据速率和占空比。I2SCLH 定义了 SCL 高电平期间 I2C_PCLK 的周期数，I2SCLL 定义了 SCL 低电平期间 I2C_PCLK 的周期数。频率由下面公式得出（I2C_PCLK 是外设总线 APB 的频率）：

(6)

$$I^2C_{bitfrequency} = \frac{I2CPCLK}{I2CSCLH + I2CSCLL}$$

I2SCLL 和 I2SCLH 的值必须确保数据速率在适当的 I²C 数据速率范围内。各寄存器的值必须大于或等于 4。[表 189](#) 给出了根据 I2C_PCLK 频率和 I2SCLL 及 I2SCLH 值计算出来的 I²C 总线速率的示例。

表 189. 用于选择 I2C_PCLK 值的 I2SCLL + I2SCLH 值

I ² C 模式	I2C_PCLK (MHz)						
	6	8	10	12	16	20	30
	SCLH + SCLL						
100 kHz（标准）	60	80	100	120	160	200	300
400 kHz（快速模式）	15	20	25	30	40	50	75
1 MHz（超快速模式）	-	8	10	12	16	20	30

I2SCLL 和 I2SCLH 的值不一定要相同。软件可通过设定这两个寄存器得到 SCL 的不同占空比。例如，I²C 总线规范定义在快速模式和超快速模式 I²C 下不同值对应的 SCL 低电平时间和高电平时间。

11.7.6 I²C 控制清零寄存器 (CONCLR - 0x4000 0018)

I2CONCLR 寄存器控制对 I2CON 寄存器中的位清零，该寄存器控制 I²C 接口的操作。向该寄存器的位写 1 会使 I²C 控制寄存器中的相应位清零。写 0 无效。

表 190. I²C 控制清除寄存器 (CONCLR - 0x4000 0018) 位描述

位	符号	描述	复位值
1:0	-	保留。用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用
2	AAC	声明应答清零位。	
3	SIC	I ² C 中断清零位。	0
4	-	保留。用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用
5	STAC	START 标志清零位。	0
6	I2ENC	I ² C 接口禁能位。	0
7	-	保留。用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用
31: 8	-	保留	-

AA 是声明应答清零位。向该位写 1 可清零 I2CONSET 寄存器中的 AA 位。写 0 无效。

SIC 是 I²C 中断清零位。向该位写 1 可清零 I2CONSET 寄存器中的 SI 位。写 0 无效。

STAC 是起始标志清零位。向该位写 1 可清零 I2CONSET 寄存器中的 STA 位。写 0 无效。

I2ENC 是 I²C 接口禁能位。向该位写 1 可清零 I2CONSET 寄存器中的 I2EN 位。写 0 无效。

11.7.7 I²C 监控模式控制寄存器

该寄存器控制监控模式，监控模式允许 I²C 块在不需实际参与通信或干扰 I²C 总线的情况下监控 I²C 总线上的流量。

表 191. I²C 监控模式控制寄存器 (MMCTRL - 0x4000 001C) 位描述

位	符号	值	描述	复位值
0	MM_ENA		监控模式使能。	0
		0	监控模式禁能。	
		1	I ² C 块将进入监控模式。在该模式下，SDA 输出将被强制为高电平。这可防止 I ² C 块向 I ² C 数据总线输出任何类型的数据（包括 ACK） 根据 ENA_SCL 位状态，也可以将输出强制为高电平，以防止模块控制 I ² C 时钟线。	
1	ENA_SCL		SCL 输出使能。	0
		0	当模块处于监控模式下，清零该位则 SCL 输出将被强制为高电平。如上所述，这可防止模块控制 I ² C 时钟线。	
		1	该位置位时，I ² C 块将以与正常操作中相同的方法控制时钟线。这意味着，作为从机设备，I ² C 块可“延长”时钟线（使其为低电平），直到它有时间响应 I ² C 中断为止。 注： 当 ENA_SCL 位清零且 I ² C 不能再延迟总线时，中断响应时间就变得很重要。为了使器件在这些情况下能有更多时间对 I ² C 中断作出响应，就需要使用 DATA_BUFFER 寄存器来保存接收到的数据，保存时间为发送完一个 9 位字的时间。	

表 191. I²C 监控模式控制寄存器 (MMCTRL - 0x4000 001C) 位描述 (续)

位	符号	值	描述	复位值
2	MATCH_ALL		选择中断寄存器匹配。	0
		0	该位清零时，只有在 4 个（最多）地址寄存器（如上面所描述的）中的一个出现匹配时，才会产生中断。也就是说，模块会作为普通的从机响应，直到有地址识别。	
		1	当该位置 1 且 I ² C 处于监控模式时，可在任意接收的地址上产生中断。这将使器件监控总线上的所有通信量。	
31: 3	-	-	保留。	-

注：如果 MM_ENA 为 ‘0’（例如，如果模块没有处于监控模式），则 ENA_SCL 和 MATCH_ALL 位无效。

11.7.7.1 监控模式下的中断

当模块处于监控模式时所有中断将正常出现。这意味着检测到地址匹配时就会产生第一个中断（如果 MATCH_ALL 位置位，则接收到任意地址都会产生中断，否则只有在地址与 4 个地址寄存器中的一个匹配时才会产生中断）。

检测地址匹配后，对于从机写传输，每接收到一个字节就会产生中断，对于从机读传输，每发送完模块 “认为” 要发送的字节后产生中断。在第二种情况下，数据寄存器实际上包含了总线上其它从机发送的数据，这些从机实际上是被主机寻址的。

所有中断产生后，处理器可读数据寄存器以查看总线上实际发送的数据。

11.7.7.2 监控模式下仲裁丢失

在监控模式下，I²C 块不能响应总线主机的信息请求或发布应答，而是由总线上的其它从机作出响应。对于这一模块，这很可能会导致仲裁丢失。

软件应当意识到模块在监控模式中并且不应当对任何检测到的仲裁状态丢失做出响应。另外，模块中还可设计一个硬件以阻止一些 / 所有的仲裁丢失状态发生（如果这些状态会阻止产生想要的中断或产生不想要的中断）。是否需要此类硬件仍待定。

11.7.8 I²C 从属地址寄存器 (ADR[1, 2, 3]- 0x4000 00[20, 24, 28])

这些寄存器可读 / 写，只有在 I²C 接口设置为从机模式时才可用。在主机模式下，该寄存器无效。I2ADR 的 LSB 为通用调用位。当该位置位时，可识别通用调用地址 (0x00)。

其中包含位 00x 的寄存器将被禁能且不与总线上的任意地址匹配。复位时 4 个寄存器（包括 I2ADR0 寄存器）都要清零到该禁能状态。

表 192. I²C 从属地址寄存器 (ADR1 - 0x4000 0020、ADR2 - 0x4000 0024、ADR3 - 0x4000 0028) 位描述

位	符号	描述	复位值
0	GC	通用调用使能位。	0
7:1	地址	从机模式的 I ² C 器件地址。	0x00
31: 8	-	保留	-

11.7.9 I²C 数据缓冲寄存器 (DATA_BUFFER - 0x4000 002C)

在监控模式下，如果 ENA_SCL 没有置位，则 I²C 块就不能延长时钟（使总线延迟）。这意味着处理器读取总线接收数据内容的时间有限。如果处理器读 I2DAT 移位寄存器，则在接收数据被新数据覆写前，它通常只有一个位时间对中断作出响应。

为了使处理器有更多时间响应，将增加一个新的 8 位只读 DATA_BUFFER 寄存器。总线上每接收到 9 位（8 位数据加上 1 位 ACK 或 NACK）后，I2DAT 移位寄存器的高 8 位的内容将自动传输到 DATA_BUFFER。这意味着处理器有 9 位发送时间响应中断及在数据被覆写前读取数据。

处理器仍可直接读 I2DAT，I2DAT 的行为无论如何是不会改变的。

尽管 DATA_BUFFER 寄存器主要是用于监控模式（ENA_SCL bit = ‘0’），但它也可用于在任何操作模式下随时读取数据。

表 193. I²C 数据缓冲寄存器 (DATA_BUFFER - 0x4000 002C) 位描述

位	符号	描述	复位值
7:0	数据	该寄存器保存 I2DAT 移位寄存器中高 8 位的内容	0
31: 8	-	保留	-

11.7.10 I²C 屏蔽寄存器 (MASK[0, 1, 2, 3] - 0x4000 00[30, 34, 38, 3C])

4 个屏蔽寄存器各包含 7 个有效位（7:1）。当与屏蔽寄存器关联的 I2ADDRn 寄存器比较时，这些寄存器中的任一位置 ‘1’ 都会使接收地址的相应位自动比较。也就是说，决定地址匹配时不考虑 I2ADDRn 寄存器中被屏蔽的位。

复位时，所有屏蔽寄存器位清零。

当与通用调用地址 (“0000000”) 比较时，屏蔽寄存器无效。

屏蔽寄存器的位（31:8）和位（0）未使用且不应写入值。读这些位总返回 0。

当产生地址匹配中断时，处理器必须读数据寄存器 (DAT) 以决定实际引起匹配的接收地址。

表 194. I²C 屏蔽寄存器 (MASK0 - 0x4000 0030, MASK1 - 0x4000 0034, MASK2 - 0x4000 0038, MASK3 - 0x4000 003C) 位描述

位	符号	描述	复位值
0	-	保留。用户软件不应向保留位写入 1。这个位总是读出为 0。	0
7:1	MASK	屏蔽位。	0x00
31: 8	-	保留。用户软件不应向保留位写入 1。这些位总是读出为 0。	0

11.8 I²C 操作模式

在给定的应用中，I²C 块可作为主机、从机或同时作主机和从机。在从机模式，I²C 硬件查找其 4 个从属地址中的任何一个地址及通用调用地址。如果检测到其中一个地址，则请求中断。如果处理器想成为总线主机，则在进入主机模式前，硬件将一直等待，直到总线空闲，这样就不会中断可能存在的从机操作。如果在总线模式下丢失总线仲裁，则 I²C 块将立即切换到从机模式并在同一串行传输中检测自身的从属地址。

11.8.1 主发送模式

该模式下，数据由主机发送到从机。在进入主发送模式前，必须按表 195 所示初始化 CONSET 寄存器。I2EN 必须置 1 以使能 I²C 功能。如果 AA 位为 0，则当另一个器件为总线上的主机时，I²C 接口不会对任何地址作出应答，因此不能进入从机模式。STA、STO 和 SI 位必须为 0。通过向 CONCLR 寄存器中的 SIC 位写 1 来清零 SI 位。写从属地址后应清零 STA 位。

表 195. 用于配置主机模式的 CONSET

位	7	6	5	4	3	2	1	0
符号	-	I2EN	STA	STO	SI	AA	-	-
值	-	1	0	0	0	0	-	-

发送的第一个字节包含接收器件的从属地址（7 位）和数据方向位。在该模式下，数据方向位 (R/W) 应为 0，表示执行写操作。发送的第一个字节包含从属地址和写操作位。一次发送 8 位数据。每发送完一个字节后，接收到一个应答位。输出起始和停止条件指示串行传输的起始和结束。

转件置位 STA 位时，I²C 接口将进入主发送模式。一旦总线空闲，I²C 逻辑就会发送起始条件。发送起始条件后，SI 位置位，STAT 寄存器中的状态代码为 0x08。该状态代码引导状态服务程序，将从属地址和写操作位装入 DAT 寄存器，然后清零 SI 位。通过向 CONCLR 寄存器中的 SIC 位写入 1 清零 SI。

当已发送从属地址和 R/W 位并接收到应答位后，SI 位再次置位，此时，主机模式下可能的状态为 0x18、0x20 或 0x38，如果从机模式使能（将 AA 置 1），则可能为 0x68、0x78 或 0xB0。各状态码的相应操作见表 200 ~ 表 203。

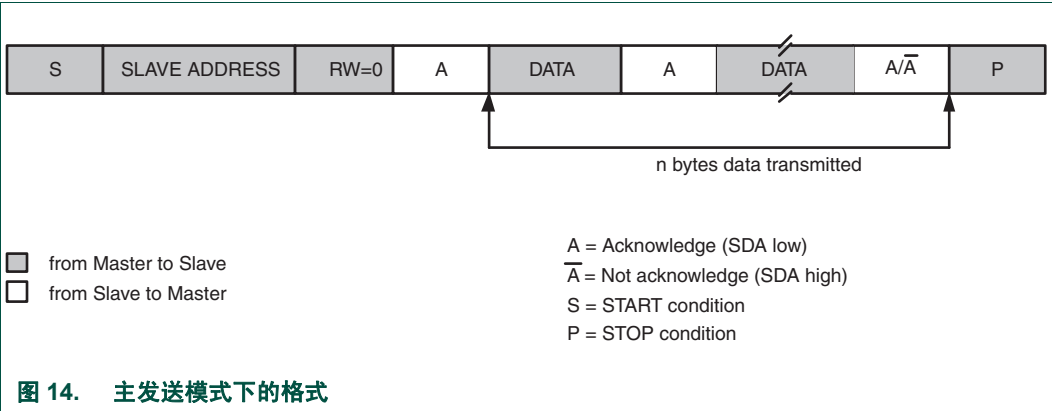


图 14. 主发送模式下的格式

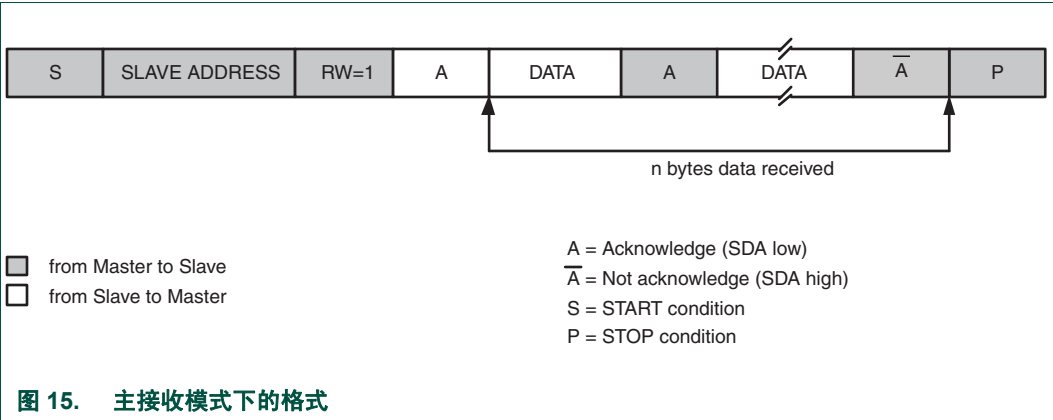
11.8.2 主接收模式

在主接收模式下，从从发送器接收数据。发起传输的方式与在主发送模式下的方式相同。发送完起始条件后，中断服务程序必须将从属地址和数据方向位装入 I²C 数据寄存器 (DAT)，然后清零 SI 位。这种情况下，数据方向位 (R/W) 应为 1 以指示读操作。

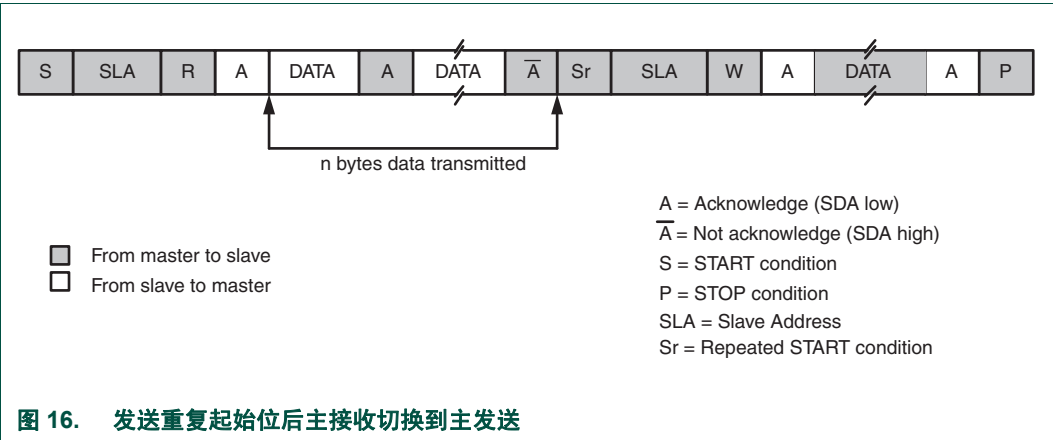
发送完从属地址和数据方向位并接收到应答位后，SI 位置位，状态寄存器将显示状态代码。对于主机模式，状态代码可能为 0x40、0x48 或 0x38。对于从机模式，状态代码可能为 0x68、0x78 或 0xB0。详细信息参见表 201。

当 LPC122x 需要应答一个接收到的字节时，需在清零 SI 位并启动读取的字节之前先将 AA 位相应置位。当 LPC122x 需要不应答一个接收到的字节时，需在清零 SI 位并启动读取的字节之前先将 AA 位清零。

请注意，最后一个收到的字节后总是跟有 LPC122x 的“非应答”位，以便主机能指示从机读取序列已完成而它需要发出停止或重复起始命令。发送“非应答”位且 SI 位置位后，LPC122x 可发送停止（STO 位置位）或重复起始条件（STA 位置位）。然后 SI 位清零以启动请求操作。



经过一个重复起始条件后，I²C 可切换到主发送模式。



11.8.3 从接收模式

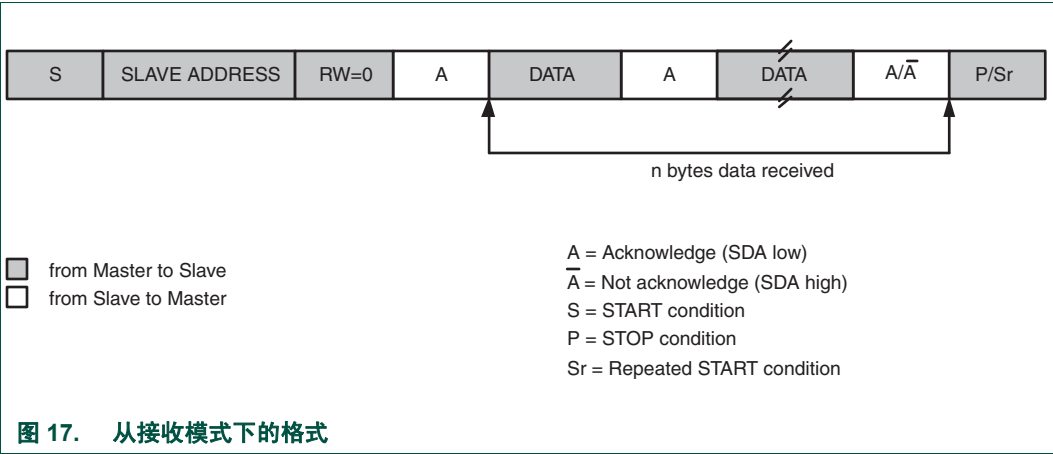
从接收模式下，从主发送器接收数据字节。要初始化从接收模式，对任一从属地址寄存器 (ADR0-3) 和从屏蔽寄存器 (MASK0-3) 进行写操作并按表 196 所示写 I²C 控制置位寄存器 (CONSET)。

表 196. 用于配置从机模式的 CONSET

位	7	6	5	4	3	2	1	0
符号	-	I2EN	STA	STO	SI	AA	-	-
值	-	1	0	0	0	1	-	-

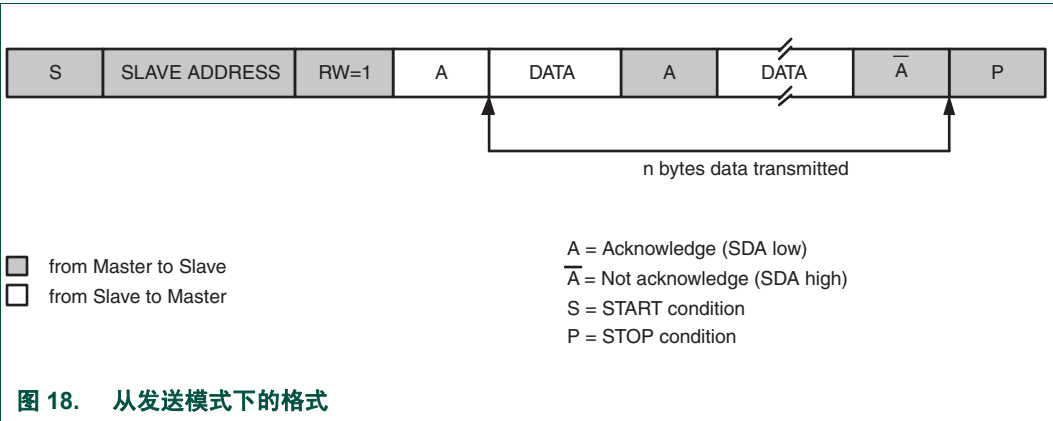
I2EN 必须置 1 以使能 I²C 功能。AA 位必须置 1 以应答其自身的从属地址或通用调用地址。STA、STO 和 SI 位置 0。

初始化 ADR 和 CONSET 后，I²C 接口开始等待，直到被其自身从属地址或后跟数据方向位的通用调用地址寻址为止。如果方向位为 0(W)，则进入从接收模式。如果方向位为 1(R)，则进入从发送模式。接收到地址和方向位后，SI 位置位，可从状态寄存器 (STAT) 读取一个有效状态代码。关于状态低码和操作请见表 202。



11.8.4 从发送模式

接收和处理第一个字节的方式与从接收模式下相同。但是，在该模式下，方向位为 1，指示读操作。通过 SDA 发送串行数据，通过 SCL 输入串行时钟。起始和停止条件分别看作串行传输的开始和结束。在特定应用中，I²C 可作为主机 / 从机。在从机模式，I²C 硬件查找自身从属地址和通用调用地址。如果检测到其中一个地址，则请求中断。如果微控制器想成为总线主机，则在进入主机模式前，硬件将一直等待，直到总线空闲，这样就不会中断可能存在的从机操作。如果在总线模式下丢失总线仲裁，则 I²C 接口将立即切换到从机模式并在同一串行传输中检测自身的从属地址。



11.9 I²C 执行和操作

图 19 所示为片内 I²C 总线接口的执行流程，下面章节将对图中各模块进行描述。

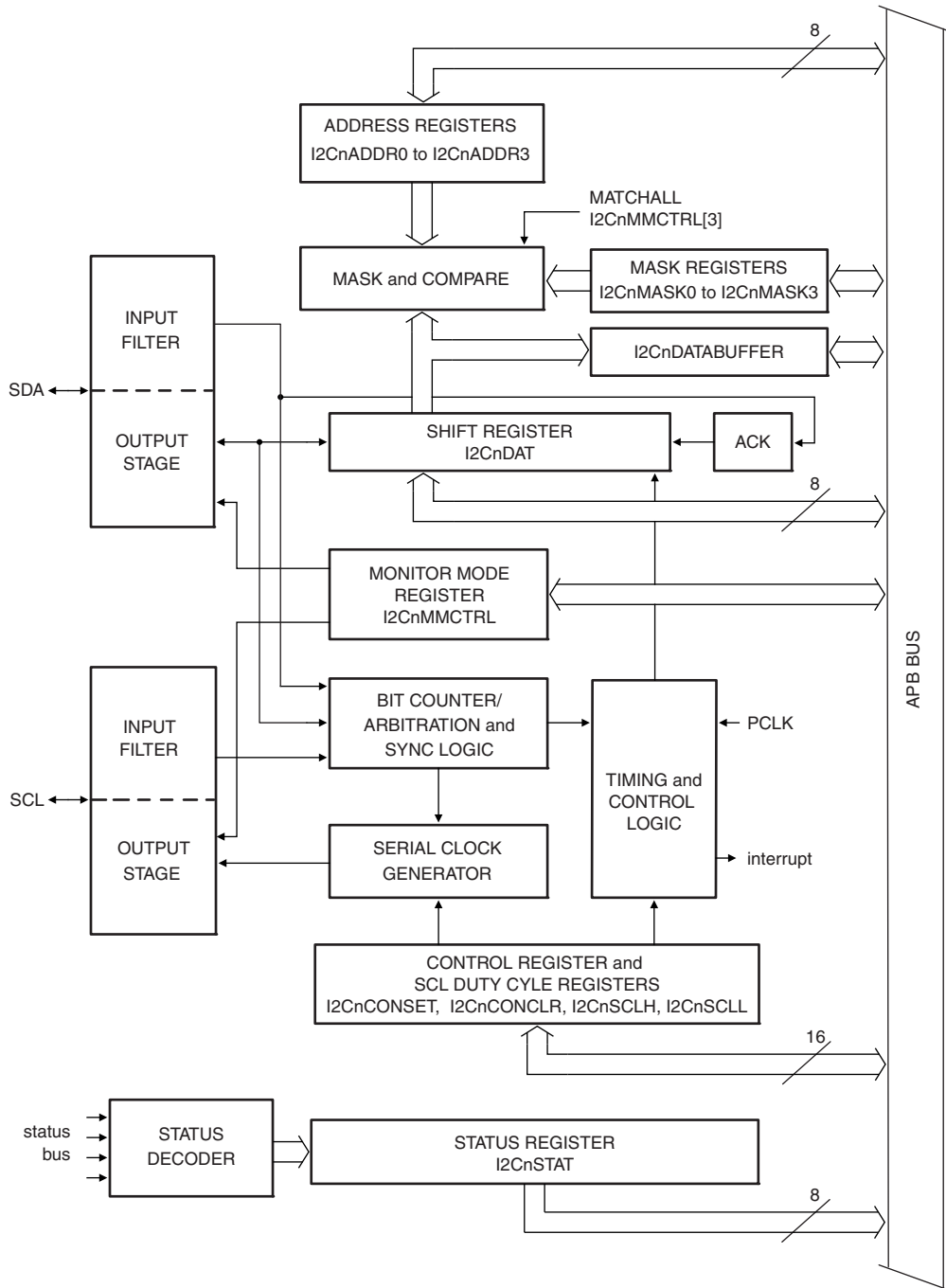


图 19. I²C 串行接口功能框图

11.9.1 输入滤波器和输出级

输入信号与内部时钟同步，小于 3 个时钟的脉冲尖峰将被滤出。

I²C 输出是一个特殊焊盘，是为符合 I²C 规范而设计的。

11.9.2 地址寄存器 ADR0 ~ ADR3

当作为从发送器或接收器时，这些寄存器可装入 7 位从属地址（7 个最高有效位），I²C 块将对这些地址作出响应。LSB (GC) 用于使能通用调用地址 (0x00) 识别。当使能多个从属地址时，接收到其“自身从属地址”后，可从 DAT 寄存器读出接收到的实际地址。

注：在本章后续部分，当提到“自身从属地址”一词时，是指地址屏蔽后四个配置的从属地址之中的任何一个。

11.9.3 地址屏蔽寄存器，MASK0 ~ MASK3

4 个屏蔽寄存器各包含 7 个有效位（7:1）。当与屏蔽寄存器相关联的 ADR_n 寄存器相比较时，这些寄存器中的任一位置“1”都会造成接收地址中的相应位自动比较。也就是说，决定地址匹配时不考虑 ADR_n 寄存器中被屏蔽的位。

当产生地址匹配中断时，处理器必须读数据寄存器 (DAT) 以决定实际引起匹配的接收地址。

11.9.4 比较器

比较器将接收到的 7 位从属地址与屏蔽后 ADR0 ~ ADR3 中配置的四个从属地址中的任何一个进行比较。它还将先收到的 8 位字节与通用调用地址 (0x00) 进行比较。如果发现匹配，则将相应状态位置位并请求中断。

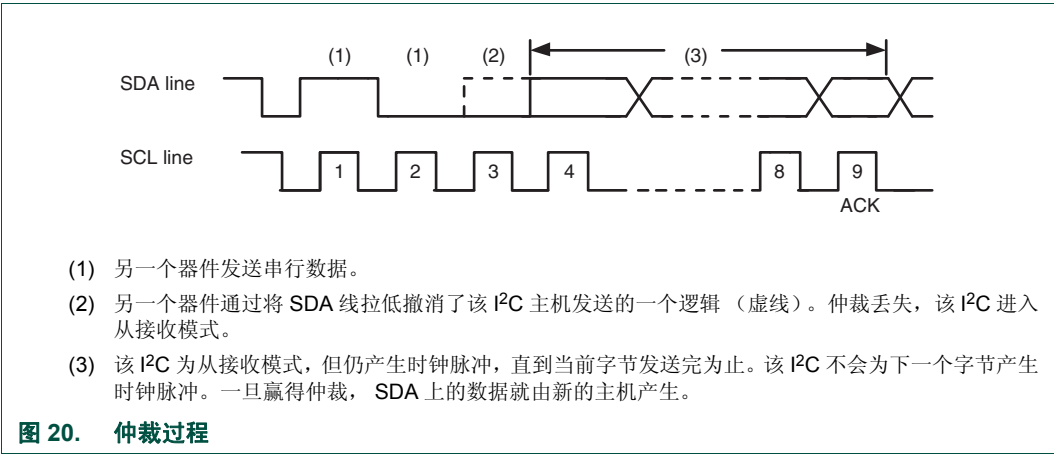
11.9.5 移位寄存器，DAT

这个 8 位寄存器包含一个要发送的串行数据字节或一个刚接收到的字节。DAT 中的数据通常是从右向左移动；要发送的第一位是 MSB（位 7），接收到一个字节后，接收到数据的第一位放在 DAT 的 MSB 位。当数据被移出时，总线上的数据同时移入；DAT 总是包含总线上出现的最后一个字节。因此，在仲裁丢失时，主发送器到从接收器的转变和 DAT 中数据的更新同时进行。

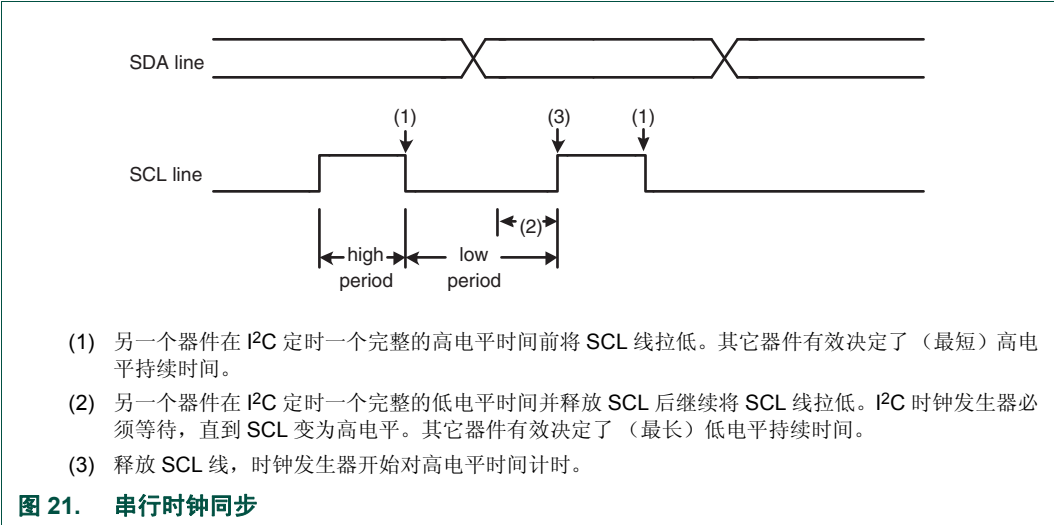
11.9.6 仲裁及同步逻辑

在主发送模式下，仲裁逻辑校验每个发送的逻辑 1 在 I²C 总线上是否真正以逻辑 1 出现。如果总线上另一个器件否定逻辑 1 并将 SDA 线拉低，则仲裁丢失，I²C 块立即由主发送器转换成从接收器。I²C 块将继续输出时钟脉冲（在 SCL 上），直到发送完当前串行字节为止。

主接收模式下也可能丢失仲裁。在该模式下，只有在 I²C 块向总线返回一个无应答：（逻辑 1）时，才会丢失仲裁。当总线上另一个器件将该信号拉低时，仲裁丢失。由于这只会发生在串行字节结束时出现，因此 I²C 块不再产生时钟脉冲。[图 20](#) 所示为仲裁过程。



同步逻辑将使串行时钟发生器与来自另一个器件的 SCL 线上的时钟脉冲同步。如果有 2 个或多个主机器件产生时钟脉冲，则高电平周期由产生最短高电平持续时间的器件决定，低电平周期由产生最长低电平持续时间的器件决定。图 21 所示为同步过程。



从机可延长低电平时间以使总线主机减速。也可以通过延长低电平时间实现握手。可在每位或一个完整字节传输后延长低电平时间。发送或接收到一个字节后，I²C 块将延长 SCL 低电平时间且已发送应答位。设置串行中断标志 (SI)，继续延长低电平时间，直到串行中断清零为止。

11.9.7 串行时钟发生器

当 I²C 块处于主发送或主接收模式时，可编程时钟脉冲发生器提供 SCL 时钟脉冲。当 I²C 块处于从机模式时，时钟脉冲发生器关闭。I²C 输出时钟频率和占空比可通过 I²C 时钟控制寄存器编程。详情请参见关于 CSCLL 和 CSCLH 寄存器的描述。除非总线与上面描述的其它 SCL 时钟源同步，否则输出时钟脉冲使用设定的占空比。

11.9.8 时序和控制

时序和控制逻辑为处理串行字节产生时序和控制信号。该逻辑块为 DAT 提供移位脉冲，可使能比较器、产生并检测起始和停止条件、接收并发送应答位、控制主 / 从机模式，还包含中断请求逻辑并监控 I²C 总线状态。

11.9.9 控制寄存器 ICONSET 和 CONCLR

I²C 控制寄存器包含用于控制以下 I²C 块功能的位：串行传输的启动和重启、串行传输的终止、位速率、地址识别及应答。

I²C 控制寄存器的内容可能读出为 CONSET。写 CONSET 可置位 I²C 控制寄存器中的相应位。反之，写 CONCLR 将清零 I²C 控制寄存器中的相应位。

11.9.10 状态解码器和状态寄存器

状态解码器读取所有内部状态位并将其压缩成 5 位代码。该代码与各 I²C 总线状态一一对应。5 位代码可用于产生向量地址，以快速处理不同的服务程序。每个服务程序处理一个特定的总线状态。如果使用 I²C 块的所有 4 种模式，则存在 26 种可能的总线状态。当串行中断标志置位（通过硬件）并保持置位（直到中断标志被软件清零为止）时，将 5 位状态码锁存到状态寄存器的 5 个最高有效位。状态寄存器的 3 个最低有效位总为 0。如果状态码用作服务程序的向量，则程序转移到 8 位地址指向的空间。大多数的服务程序不会超过 8 个字节（参见本节的软件例程）。

11.10 I²C 操作模式详解

有 4 种操作模式：

- 主发送模式
- 主接收模式
- 从接收模式
- 从发送模式

各模式下数据传输操作如[图 22](#)、[图 23](#)、[图 24](#)、[图 25](#) 和 [图 26](#) 所示。[表 197](#) 说明了介绍 I²C 操作模式的图中所使用缩写的含义。

表 197. 用于描述 I²C 操作的缩写

缩写	说明
S	起始条件
SLA	7 位从属地址
R	读数据位（SDA 为高电平）
W	写数据位（SDA 为低电平）
A	应答位（SDA 为低电平）
\overline{A}	非应答位（SDA 为高电平）
数据	8 位数据位
P	停止条件
Sr	重复起始条件

在[图 22](#) ~ [图 26](#) 中，圆圈用来指示串行中断标志何时被置位。圆圈中的数字表示 STAT 寄存器中的状态代码。每当出现这些状态代码时，必须执行服务程序来继续或结束串行传输。若串行传输被挂起，这些服务程序就不再重要，直至串行中断标志被软件清除。

当进入串行中断程序时，STAT 的状态代码用来指向跳转到的相应的服务程序。对于每个状态代码，需要的软件操作以及后面串行传输的详细情况见[表 200](#) ~ [表 204](#)。

11.10.1 主发送模式

在主发送模式中，向从接收器发送数据字节（见[图 22](#)）。在进入主发送器模式之前，CON 必须按下表进行初始化：

表 198. 用于初始化主发送模式的 CONSET

位	7	6	5	4	3	2	1	0
符号	-	I2EN	STA	STO	SI	AA	-	-
值	-	1	0	0	0	x	-	-

I²C 速率也必须在 I2SCLL 和 I2SCLH 寄存器中配置。必须将 I2EN 设置为逻辑 1 来使能 I²C 块。如果 AA 位复位，则当另一个器件正变成总线主机时，I²C 块将不会应答其自身的从属地址或通用调用地址。也就是说，如果 AA 位复位，则 I²C 接口就不能进入从机模式。STA、STO 和 SI 必须复位。

此时，可通过置位 STA 位进入主发送模式。一旦总线空闲，I²C 逻辑会立即测试 I²C 总线并产生一个起始条件。当发送起始条件时，串行中断标志 (SI) 置位，状态寄存器 (STAT) 中的状态代码为 0x08。中断服务程序利用该状态代码进入相应的状态服务程序，将从属地址和数据方向位 (SLA+W) 装入 DAT。必须先复位 CON 的 SI 位，串行传输才能继续。

当发送完从属地址和方向位且接收到一个应答位时，串行中断标志 (SI) 再次置位，STAT 中可能是一系列不同的状态代码。主机模式下为 0x18、0x20 或 0x38，从机模式（AA= 逻辑 1）为 0x68、0x78 或 0xB0。[表 200](#) 中详细介绍了每个状态代码对应的操作。在发送完重复起始条件（状态 0x10）后，I²C 块通过将 SLA+R 装入 DAT 切换到主接收模式。

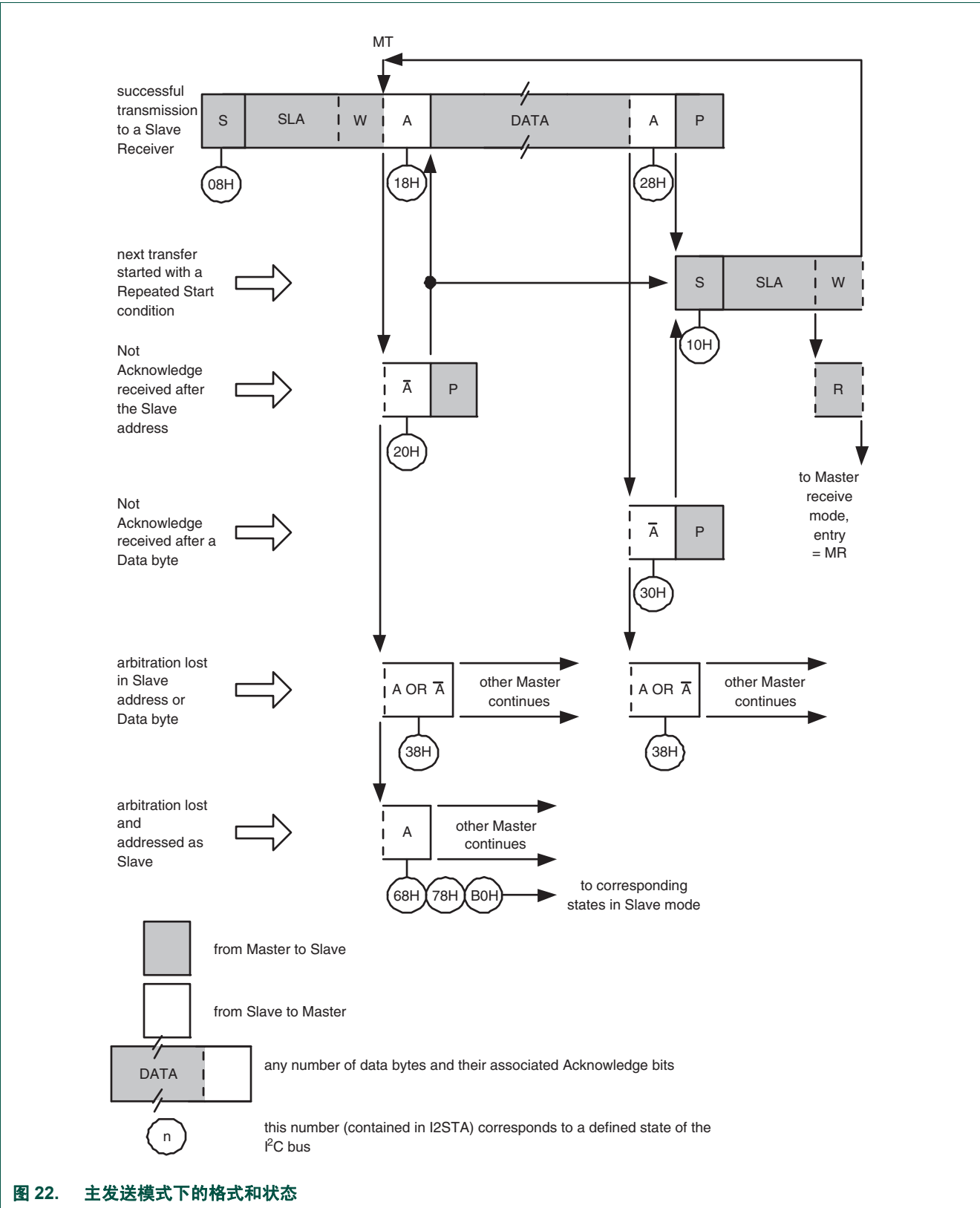


图 22. 主发送模式下的格式和状态

11.10.2 主接收模式

在主接收模式中，主机所接收的数据字节来自从发送器（见[图 23](#)）。传输按照主接收模式中的情况初始化。当发送完起始条件后，中断服务程序必须把 7 位从属地址和数据方向位 (SLA+R) 装入 DAT。必须先清零 CON 中的 SI 位，串行传输才能继续。

当发送完从属地址和数据方向位且接收到一个应答位时，串行中断标志 (SI) 再次置位，STAT 中可能是一系列不同的状态代码。主机模式下为 0x40、0x48 或 0x38，从机模式 (AA = 1) 为 0x68、0x78 或 0xB0。[表 201](#) 中详细介绍了每个状态代码对应的操作。在发送完重复起始条件（状态 0x10）后，I²C 块通过将 SLA+W 装入 DAT 切换到主发送模式。

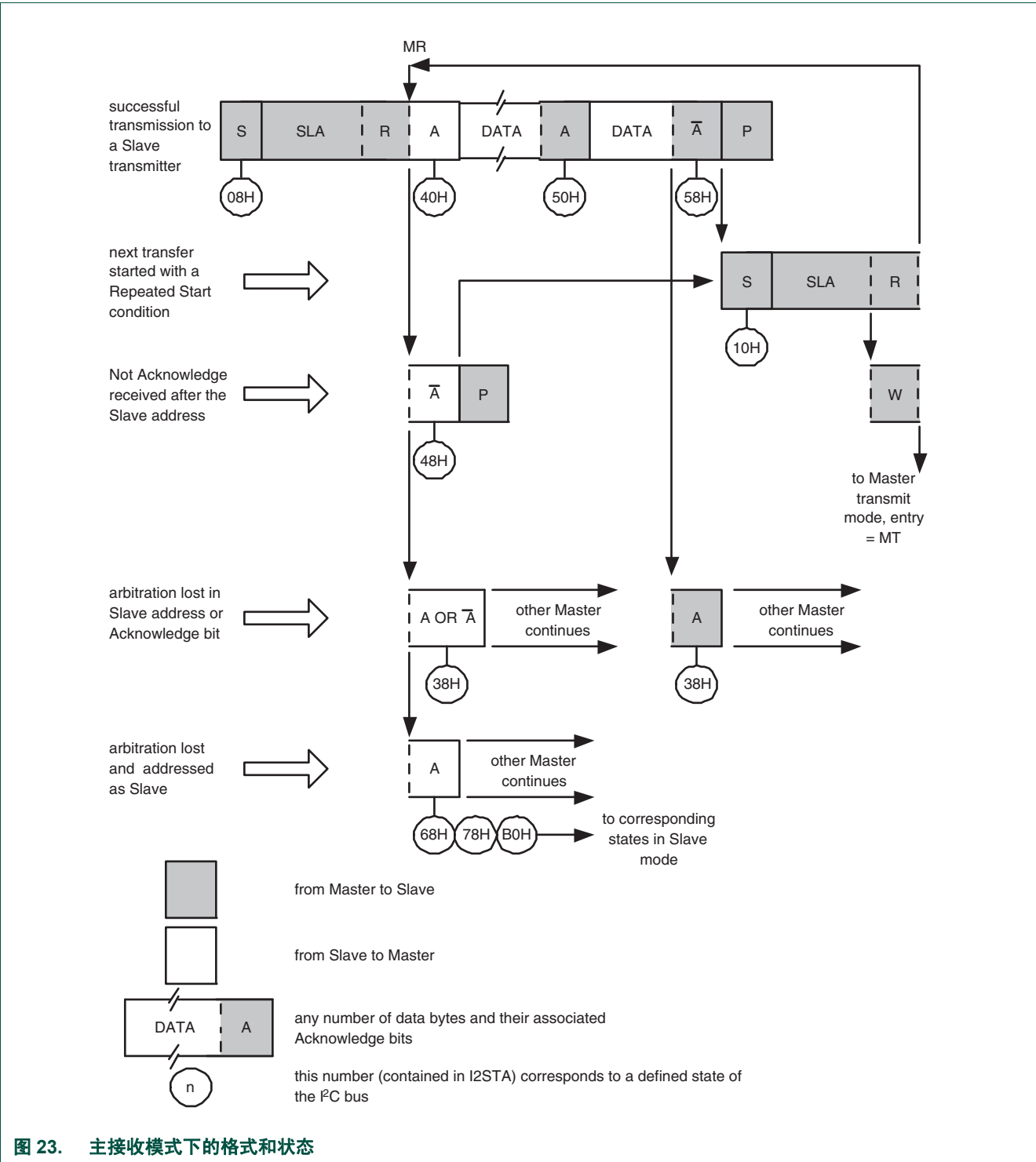


图 23. 主接收模式下的格式和状态

11.10.3 从接收模式

在从接收模式中，从机所接收的数据字节来自主发送器（见图 24）。要初始化从接收模式，必须按照下表来配置 CON 寄存器、ADR 寄存器和 MASK 寄存器。

4 个 ADR 寄存器的值与 4 个 MASK 寄存器的值决定了当使能从机功能时，I2C 块将响应哪个地址。详情请见章节 11.9.2、章节 11.9.3、章节 11.7.8 和章节 11.7.10 部分。

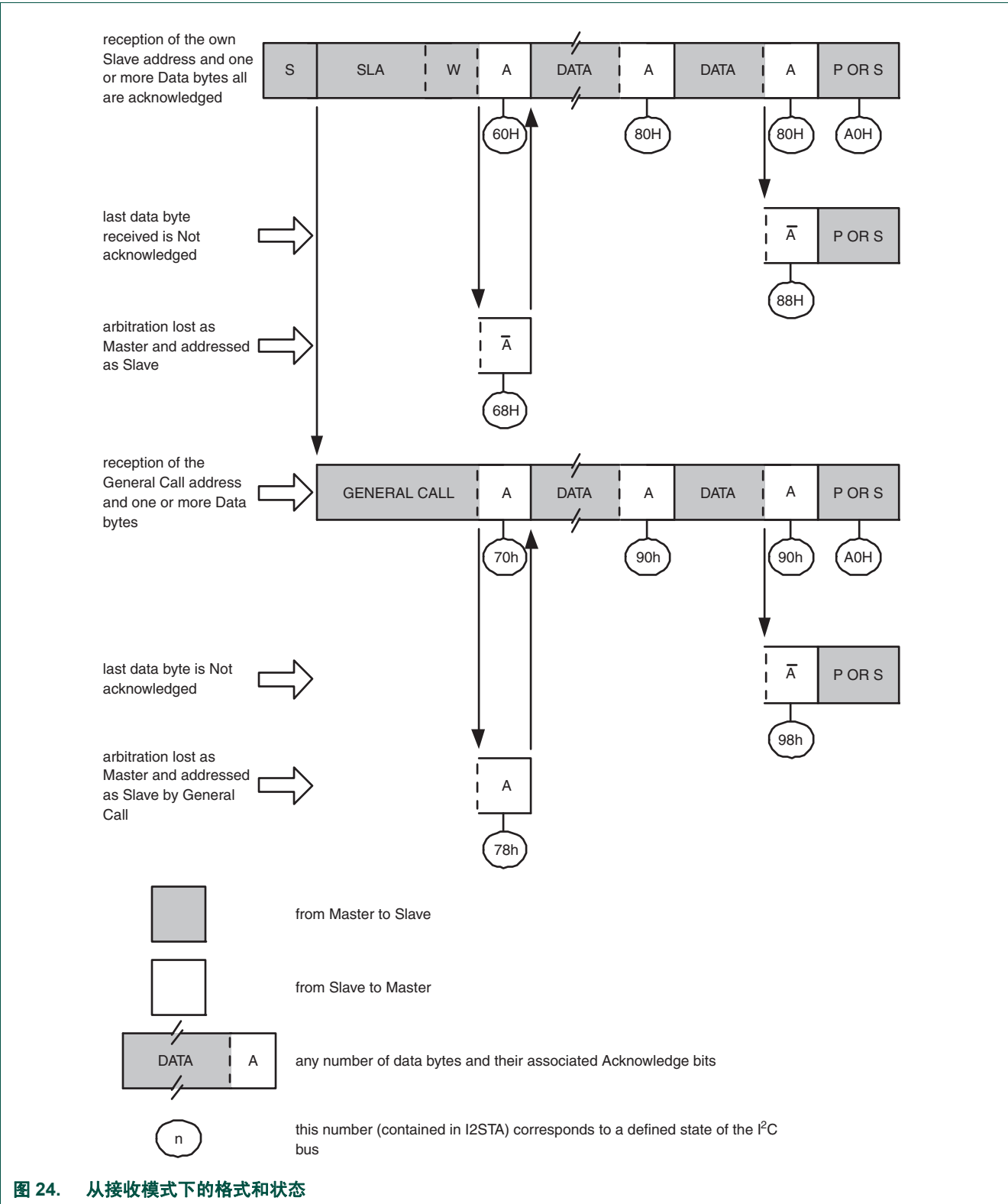
表 199. 用于初始化从接收模式的 CONSET

位	7	6	5	4	3	2	1	0
符号	-	I2EN	STA	STO	SI	AA	-	-
值	-	1	0	0	0	1	-	-

I2C 总线速率的设置不影响从机模式中的 I2C 块。必须将 I2EN 设置为逻辑 1 来使能 I2C 块。AA 位必须置位以使能 I2C 块来应答其自身从属地址或通用调用地址。STA、STO 和 SI 必须复位。

当 ADR、MASK 和 CON 寄存器完成初始化后，I2C 块一直等待，直至被自身的从属地址寻址，从属地址之后是数据方向位，该数据方向位必须为“0” (W)，以使 I2C 块在从接收模式中工作。接收完其自身的从属地址和 W 位后，串行中断标志 (SI) 置位，并且可从 STAT 中读出一个有效的状态代码。该状态代码用作状态服务程序的向量。表 202 中详细介绍了每个状态代码对应的操作。如果当 I2C 块在主机模式中时仲裁丢失，也可进入从接收模式（见状态 0x68 和 0x78）。

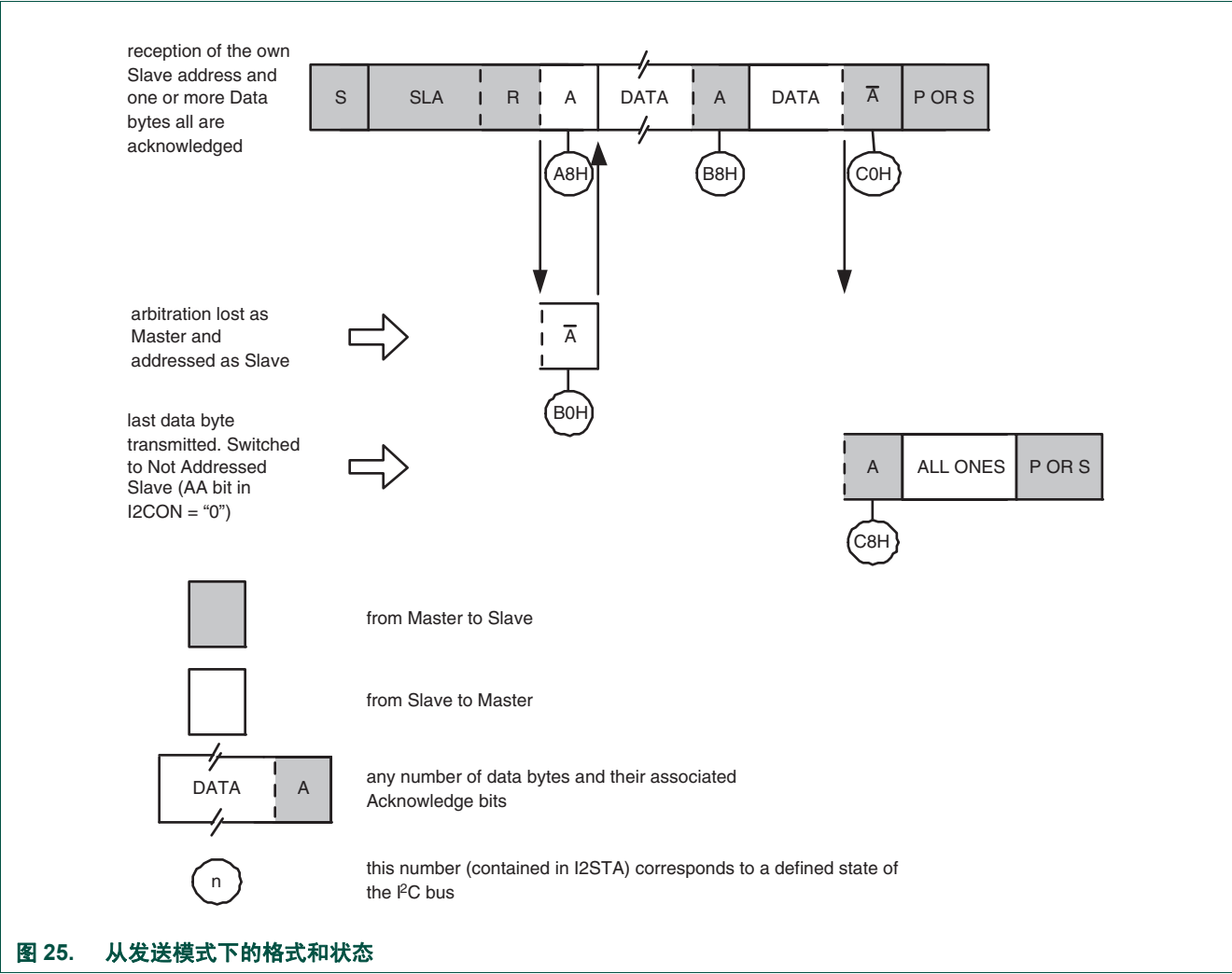
如果 AA 位在传输过程中复位，则在接收完下一个数据字节后 I2C 块将向 SDA 返回一个非应答（逻辑 1）。当 AA 复位时，I2C 块不响应其自身的从属地址或通用调用地址。但是，I2C 总线仍被监控，而且，地址识别可随时通过置位 AA 来恢复。这就意味着 AA 位可临时将 I2C 块从 I2C 总线上分离出来。



11.10.4 从发送模式

在从发送模式中，向主接收器发送数据字节（见图 25）。数据传输按照从接收模式中的情况初始化。当初始化 ADR 和 CON 后，I²C 块一直等待，直至被自身的从属地址寻址，之后是数据方向位，该数据方向位必须为“1” (R)，以便 I²C 块在从发送模式下工作。接收完其自身的从属地址和 R 位后，串行中断标志 (SI) 置位，并且可从 STAT 中读出一个有效的状态代码。该状态代码用作状态服务程序的向量，每个状态代码的对应操作详见表 203。如果当 I²C 块在主机模式中时仲裁丢失，也可进入从发送模式（见状态 0xB0）。

如果 AA 位在传输过程中复位，则 I²C 块将发送最后一个字节并进入状态 0xC0 或 0xC8。I²C 块切换到非寻址的从机模式，如果继续传输，它将忽略主接收器。因此主接收器接收所有 1 作为串行数据。当 AA 复位时，I²C 块不响应其自身的从属地址或通用调用地址。但是，I²C 总线仍被监控，而且，地址识别可随时通过置位 AA 来恢复。这就意味着 AA 位可临时将 I²C 块从 I²C 总线上分离出来。



11.10.5 详细状态表

下表显示了四种 I²C 操作模式的详细状态信息。

表 200. 主发送模式

STAT 状态代码	I ² C 总线和硬件的状态	应用程序的响应 写 / 读 DAT	写 CON				I ² C 硬件执行的下一个操作
			STA	STO	SI	AA	
0x08	已发送起始条件。	装入 SLA+W；清零 STA	X	0	0	X	将发送 SLA+W；接收 ACK 位。
0x10	已发送重复起始条件。	装入 SLA+W 或	X	0	0	X	同上。
		装入 SLA+R；清零 STA	X	0	0	X	将发送 SLA+W；I ² C 块将切换为 MST/REC 模式。
0x18	已发送 SLA+W；已接收 ACK。	装入数据字节或	0	0	0	X	将发送数据字节，接收 ACK 位。
		无 DAT 操作或	1	0	0	X	将发送重复的起始条件。
		无 DAT 操作或	0	1	0	X	将发送停止条件；STO 标志将复位。
		无 DAT 操作	1	1	0	X	将发送停止条件，然后发送起始条件；STO 标志将复位。
0x20	已发送 SLA+W；已接收非 ACK。	装入数据字节或	0	0	0	X	将发送数据字节，接收 ACK 位。
		无 DAT 操作或	1	0	0	X	将发送重复的起始条件。
		无 DAT 操作或	0	1	0	X	将发送停止条件；STO 标志将复位。
		无 DAT 操作	1	1	0	X	将发送停止条件，然后发送起始条件；STO 标志将复位。
0x28	已发送 DAT 中的数据字节；已接收 ACK。	装入数据字节或	0	0	0	X	将发送数据字节，接收 ACK 位。
		无 DAT 操作或	1	0	0	X	将发送重复的起始条件。
		无 DAT 操作或	0	1	0	X	将发送停止条件；STO 标志将复位。
		无 DAT 操作	1	1	0	X	将发送停止条件，然后发送起始条件；STO 标志将复位。
0x30	已发送 DAT 中的数据字节；已接收非 ACK。	装入数据字节或	0	0	0	X	将发送数据字节，接收 ACK 位。
		无 DAT 操作或	1	0	0	X	将发送重复的起始条件。
		无 DAT 操作或	0	1	0	X	将发送停止条件；STO 标志将复位。
		无 DAT 操作	1	1	0	X	将发送停止条件，然后发送起始条件；STO 标志将复位。
0x38	在 SLA+R/W 或数据字节中丢失仲裁。	无 DAT 操作或	0	0	0	X	I ² C 总线将被释放；进入不可寻址从机模式。
		无 DAT 操作	1	0	0	X	当总线空闲时发送起始条件。

表 201. 主接收模式

STAT 状态代码	I ² C 总线和硬件的状态	应用软件的响应 写 / 读 DAT	写 CON				I ² C 硬件执行的下一个操作
			STA	STO	SI	AA	
0x08	已发送起始条件。	装入 SLA+R	X	0	0	X	将发送 SLA+R；接收 ACK 位。
0x10	已发送重复起始条件。	装入 SLA+R 或	X	0	0	X	同上。
		装入 SLA+W	X	0	0	X	将发送 SLA+W；I ² C 块将切换为 MST/TRX 模式。
0x38	在非 ACK 位中丢失仲裁。	无 DAT 操作或	0	0	0	X	I ² C 总线将被释放；I ² C 块进入从机模式。
		无 DAT 操作	1	0	0	X	当总线空闲时发送起始条件。
0x40	已发送 SLA+R；已接收 ACK。	无 DAT 操作或	0	0	0	0	将接收数据字节，返回非 ACK 位。
		无 DAT 操作	0	0	0	1	将接收数据字节，返回 ACK 位。
0x48	已发送 SLA+R；已接收非 ACK。	无 DAT 操作或	1	0	0	X	将发送重复的起始条件。
		无 DAT 操作或	0	1	0	X	将发送停止条件；STO 标志将复位。
		无 DAT 操作	1	1	0	X	将发送停止条件，然后发送起始条件；STO 标志将复位。
0x50	已接收数据字节，已返回 ACK。	读取数据字节或	0	0	0	0	将接收数据字节，返回非 ACK 位。
		读取数据字节	0	0	0	1	将接收数据字节，返回 ACK 位。
0x58	已接收数据字节，已返回非 ACK。	读取数据字节或	1	0	0	X	将发送重复的起始条件。
		读取数据字节或	0	1	0	X	将发送停止条件；STO 标志将复位。
		读取数据字节	1	1	0	X	将发送停止条件，然后发送起始条件；STO 标志将复位。

表 202. 从接收模式

STAT 状态代码	I ² C 总线和硬件的状态	应用软件的响应				I ² C 硬件执行的下一个操作		
		写 / 读 DAT	写 CON					
			STA	STO	SI	AA		
0x60	已接收自身的 SLA+W，已返回 ACK。	无 DAT 操作或	X	0	0	0	将接收数据字节，返回非 ACK 位。	
		无 DAT 操作	X	0	0	1	将接收数据字节，返回 ACK 位。	
0x68	主控器时在 SLA+R/W 中丢失仲裁；已接收自身 SLA+W，已返回 ACK。	无 DAT 操作或	X	0	0	0	将接收数据字节，返回非 ACK 位。	
		无 DAT 操作	X	0	0	1	将接收数据字节，返回 ACK 位。	
0x70	已接收通用调用地址 (0x00)；已返回 ACK。	无 DAT 操作或	X	0	0	0	将接收数据字节，返回非 ACK 位。	
		无 DAT 操作	X	0	0	1	将接收数据字节，返回 ACK 位。	
0x78	主控器时在 SLA+R/W 中丢失仲裁；已接收通用调用地址，已返回 ACK。	无 DAT 操作或	X	0	0	0	将接收数据字节，返回非 ACK 位。	
		无 DAT 操作	X	0	0	1	将接收数据字节，返回 ACK 位。	
0x80	前一次寻址使用自身从属地址；已接收数据字节；已返回 ACK。	读取数据字节或	X	0	0	0	将接收数据字节，返回非 ACK 位。	
		读取数据字节	X	0	0	1	将接收数据字节，返回 ACK 位。	
0x88	前一次寻址使用自身SLA；已接收数据字节；已返回非 ACK。	读取数据字节或	0	0	0	0	切换到不可寻址 SLV 模式；不识别自身 SLA 或通用调用地址。	
		读取数据字节或	0	0	0	1	切换到不可寻址 SLV 模式；识别自身 SLA；如果 ADR[0] = 逻辑 1，将识别通用调用地址。	
		读取数据字节或	1	0	0	0	切换到不可寻址 SLV 模式；不识别自身 SLA 或通用调用地址。当总线空闲时发送起始条件。	
		读取数据字节	1	0	0	1	切换到不可寻址 SLV 模式；识别自身 SLA；如果 ADR[0] = 逻辑 1，将识别通用调用地址；当总线空闲后发送起始条件。	
0x90	前一次寻址使用通用调用；已接收数据字节；已返回 ACK。	读取数据字节或	X	0	0	0	将接收数据字节，返回非 ACK 位。	
		读取数据字节	X	0	0	1	将接收数据字节，返回 ACK 位。	
0x98	前一次寻址使用通用调用；已接收数据字节；已返回非 ACK。	读取数据字节或	0	0	0	0	切换到不可寻址 SLV 模式；不识别自身 SLA 或通用调用地址。	
		读取数据字节或	0	0	0	1	切换到不可寻址 SLV 模式；识别自身 SLA；如果 ADR[0] = 逻辑 1，将识别通用调用地址。	
		读取数据字节或	1	0	0	0	切换到不可寻址 SLV 模式；不识别自身 SLA 或通用调用地址。当总线空闲时发送起始条件。	
		读取数据字节	1	0	0	1	切换到不可寻址 SLV 模式；识别自身 SLA；如果 ADR[0] = 逻辑 1，将识别通用调用地址；当总线空闲后发送起始条件。	
0xA0	当使用从接收或从发送模式静态寻址时，接收到停止条件或重复的起始条件。	无 STDAT 操作或	0	0	0	0	切换到不可寻址 SLV 模式；不识别自身 SLA 或通用调用地址。	
		无 STDAT 操作或	0	0	0	1	切换到不可寻址 SLV 模式；识别自身 SLA；如果 ADR[0] = 逻辑 1，将识别通用调用地址。	
		无 STDAT 操作或	1	0	0	0	切换到不可寻址 SLV 模式；不识别自身 SLA 或通用调用地址。当总线空闲时发送起始条件。	
		无 STDAT 操作	1	0	0	1	切换到不可寻址 SLV 模式；识别自身 SLA；如果 ADR[0] = 逻辑 1，将识别通用调用地址；当总线空闲后发送起始条件。	

表 203. 从发送模式

STAT 状态代码	I ² C 总线和硬件的状态	应用软件的响应		I ² C 硬件执行的下一个操作			
		写 / 读 DAT	写 CON	STA	STO	SI	AA
0xA8	已接收自身的 SLA+R, 已返回 ACK。	装入数据字节或	X	0	0	0	将发送最后一个数据字节, 接收 ACK 位。
		装入数据字节	X	0	0	1	将发送数据字节, 接收 ACK 位。
0xB0	主控器时在 SLA+R/W 中丢失仲裁; 已接收自身 SLA+R, 已返回 ACK。	装入数据字节或	X	0	0	0	将发送最后一个数据字节, 接收 ACK 位。
		装入数据字节	X	0	0	1	将发送数据字节, 接收 ACK 位。
0xB8	已发送 DAT 中的数据字节; 已接收 ACK。	装入数据字节或	X	0	0	0	将发送最后一个数据字节, 接收 ACK 位。
		装入数据字节	X	0	0	1	将发送数据字节, 接收 ACK 位。
0xC0	已发送 DAT 中的数据字节; 已接收非 ACK。	无 DAT 操作或	0	0	0	0	切换到不可寻址 SLV 模式; 不识别自身 SLA 或通用调用地址。
		无 DAT 操作或	0	0	0	1	切换到不可寻址 SLV 模式; 识别自身 SLA; 如果 ADR[0] = 逻辑 1, 将识别通用调用地址。
		无 DAT 操作或	1	0	0	0	切换到不可寻址 SLV 模式; 不识别自身 SLA 或通用调用地址。当总线空闲时发送起始条件。
		无 DAT 操作	1	0	0	1	切换到不可寻址 SLV 模式; 识别自身 SLA; 如果 ADR[0] = 逻辑 1, 将识别通用调用地址; 当总线空闲后发送起始条件。
0xC8	已发送 DAT 中的最后一个数据字节 (AA = 0); 已接收 ACK。	无 DAT 操作或	0	0	0	0	切换到不可寻址 SLV 模式; 不识别自身 SLA 或通用调用地址。
		无 DAT 操作或	0	0	0	1	切换到不可寻址 SLV 模式; 识别自身 SLA; 如果 ADR[0] = 逻辑 1, 将识别通用调用地址。
		无 DAT 操作或	1	0	0	0	切换到不可寻址 SLV 模式; 不识别自身 SLA 或通用调用地址。当总线空闲时发送起始条件。
		无 DAT 操作	1	0	0	01	切换到不可寻址 SLV 模式; 识别自身 SLA; 如果 ADR.0 = 逻辑 1, 将识别通用调用地址; 当总线空闲后发送起始条件。

11.10.6 其它状态

还有两种 STAT 代码与已定义的 I²C 硬件状态不对应（见表 204）。下面对这两种代码进行讨论。

11.10.6.1 STAT = 0xF8

这个状态码表示没有任何可用的相关信息，因为串行中断标志 SI 还没有置位。这种情况在其它状态和 I²C 块还未开始执行串行传输之间出现。

11.10.6.2 STAT = 0x00

该状态代码表示在 I²C 串行传输过程中出现了总线错误。当格式帧的非法位置上出现了起始或停止条件时总线错误产生。这些非法位置是指在串行传输过程中的地址字节、数据字节或应答位。当外部干扰影响到内部 I²C 块信号时也会产生总线错误。总线错误出现时 SI 置位。要从总线错误中恢复，STO 标志必须置位，SI 必须被清除。这使得 I²C 块进入曳茄爸返绕从机模式（已定义的状态）并清除 STO 标志（CON 中的其它位不受影响）。SDA 和 SCL 线被释放（不发送停止条件）。

表 204. 其它状态

状态代码 (CSTAT)	I ² C 总线和硬件的状态	应用软件的响应		I ² C 硬件执行的下一个操作			
		写 / 读 DAT	写 CON	STA	STO	SI	AA
0xF8	无可用的相关状态信息； SI = 0。	无 DAT 操作	无 CON 操作				
0x00	由于非法起始或停止条件的出现，在 MST 或选择的从机模式中出现总线错误。当外部干扰使 I ² C 块进入未定义的状态时也出现 0x00 状态。	无 DAT 操作	0	1	0	X	

11.10.7 某些特殊情况

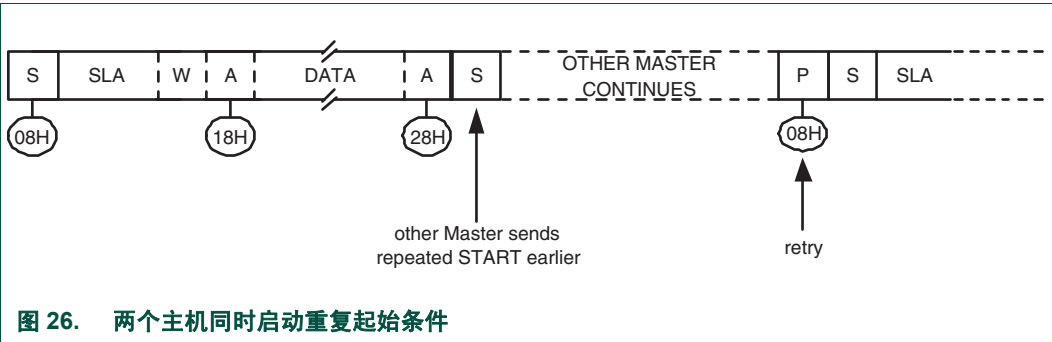
I²C 硬件可以处理串行传输过程中出现的以下几种特殊情况：

- 两个主机同时启动重复起始条件
- 仲裁丢失后的数据传输
- 强制访问 I²C 总线
- SCL 或 SDA 低电平妨碍 I²C 总线的操作
- 总线错误

11.10.7.1 两个主机同时启动重复起始条件

在主发送模式或主接收模式下可以产生重复起始条件。如果此时另一个主机同时产生重复起始条件，就出现特殊情况（见图 26）。在出现这种情况之前，任何一个主机都不会丢失仲裁，因为它们发送的数据相同。

如果 I²C 硬件在产生重复起始条件之前在 I²C 总线上检测到重复起始条件，则它将释放总线，并且不产生中断请求。如果另一个主机通过产生停止条件来释放总线，则 I²C 块将发送一个正常的起始条件（状态 0x08），并开始重新进行完整的串行数据传输。



11.10.7.2 仲裁丢失后的数据传输

在主发送模式和主接收模式中仲裁可能会丢失（见图 20）。STAT 寄存器中的状态代码可表示仲裁丢失，代码有：0x38、0x68、0x78 和 0xB0（见图 22 和图 23）。

如果 CON 中的 STA 标志由服务这些状态的程序置位, 则当总线再次空闲时, 会发送一个起始条件 (状态 0x08), 并且不受 CPU 的影响, 开始重新尝试完整的串行数据传输。

11.10.7.3 强制访问 I²C 总线

在某些应用中, 不可控制源可能会造成总线挂起。在这种情况下, 干扰、总线的暂时中断或 SDA 和 SCL 之间的暂时短路都会导致总线挂起。

如果不可控制源产生了一个多余的起始条件或屏蔽了一个停止条件, 则 I²C 总线一直保持忙碌状态。如果 STA 标志置位且在相应的时间内未访问总线, 那么 I² 总线有可能会被强制访问。这可通过在 STA 标志仍被设置时置位 STO 标志来实现。不发送停止条件。I²C 硬件的操作就好像是接收到停止条件一样, 可以发送起始条件。STO 标志通过硬件清零 (图 27)。

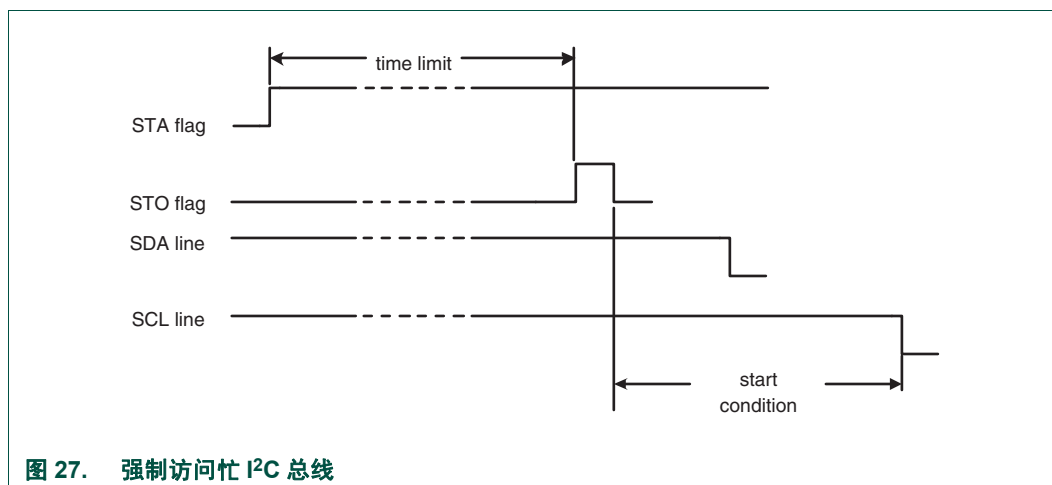
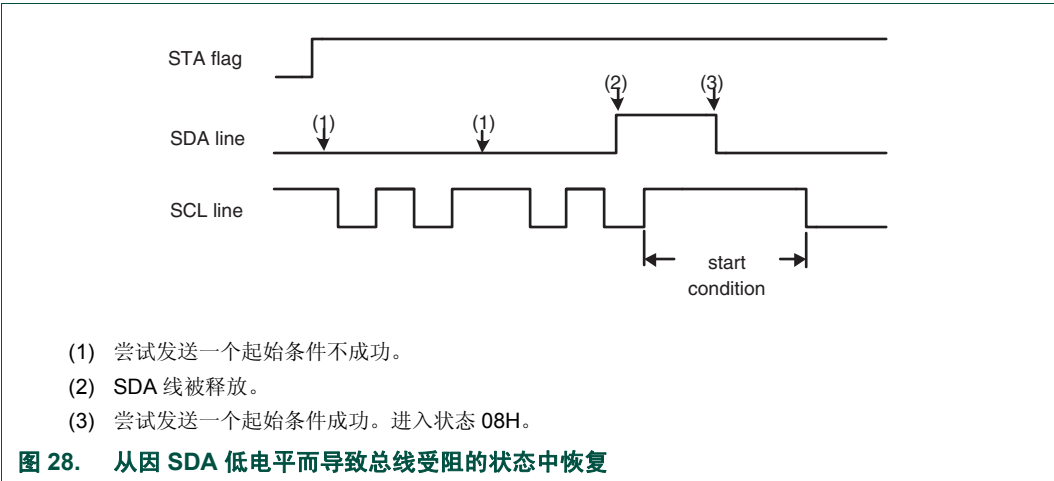


图 27. 强制访问忙 I²C 总线

11.10.7.4 I²C 总线的操作被 SCL 或 SDA 低电平妨碍

如果 SDA 或 SCL 线被总线上任何一个器件拉低, I²C 总线可能挂起。如果 SCL 线被总线上的器件妨碍 (拉低), 不能继续串行传输, 这时必须通拉低 SCL 总线的器件来处理。

一般来说, SDA 线可能会被总线上另一个不和当前总线主机同步的器件拉低, 不同步的原因可能是丢失了一个时钟周期或是以噪声脉冲作为时钟信号。在这种情况下, 可通过向 SCL 发送另外的时钟脉冲来处理, 见图 28。虽然 I²C 接口没有专门用来检测总线挂起的定时器, 但可以用系统的其它定时器来完成。当检测到总线挂起时, 软件会强制给 SCL (要求最多 9 个) 时钟信号, 直至 SDA 被造成问题的器件释放。此时, 从机可能还是不同步, 所以还要发送一个起始条件以确保所有 I²C 外设同步。



11.10.7.5 总线错误

当格式帧的非法位置上出现起始或停止条件时总线错误产生。这些非法位置是指在串行传输过程中的地址字节、数据位或应答位。

仅当 I²C 硬件作为主机或被寻址的从机进行串行传输时，它才对总线错误有反应。检测到总线错误时，I²C 块会立即切换成非寻址的从机模式，并释放 SDA 和 SCL 线，设置中断标志，并将 0x00 装入状态寄存器。该状态代码可用作状态服务程序的向量，尝试再次终止串行传输或从错误状态中恢复，如表 204 所示。

11.10.8 I²C 状态服务程序

本节将介绍不同 I²C 状态服务程序都必须执行的操作。它们包括：

- 复位后 I²C 块的初始化。
- I²C 中断服务
- 支持 4 种 I²C 操作模式的 26 种状态服务程序。

11.10.8.1 初始化

在初始化示例中，I²C 块可在主机模式和从机模式中使能。对于每种模式，缓冲区可用于发送和接收数据。初始化程序将执行以下功能：

- 向 ADR 寄存器和 MASK 寄存器装入用于配置器件自身从属地址的值和通用调用位 (GC)
- 置位 I²C 中断使能位和中断优先级位
- 通过同时设置 CON 寄存器中的 EN 和 AA 位来使能从机模式，通过装载 SCLH 和 SCLL 寄存器来定义串行时钟频率（主机模式）。主机程序必须从主程序开始执行。

这时，I²C 硬件开始在 I²C 总线上检查自身的从属地址和通用调用。一旦检测到通用调用或自身从属地址，则请求中断且把相应的状态信息装入 STAT。

11.10.8.2 I²C 中断服务

当进入 I²C 中断时，STAT 含有一个状态代码，可识别要执行的 26 个状态服务中的其中一个。

11.10.8.3 状态服务程序

每个状态程序都是 I²C 中断程序的组成部分，分别用来处理 26 种状态。

11.10.8.4 配合实际应用的状态服务

状态服务示例演示了响应 26 个 I²C 状态代码必须执行的典型操作。如果 4 种 I²C 操作模式中有一种或几种没被用到，则模式的相关的状态服务可被忽略，只要小心处理，那些状态就不会出现。

在应用中，可能需要在 I²C 操作过程中执行一些超时处理，来限制无效总线或丢失服务程序。

11.11 软件示例

11.11.1 初始化程序

将 I²C 接口初始化用作从机和 / 或主机的例子。

1. 将自身的从属地址装入 ADR，使能通用调用识别（如果需要的话）。
2. 使能 I²C 中断。
3. 向寄存器 CONSET 写入 0x44 来置位 EN 和 AA 位，并使能从机功能。对于主机功能，可向寄存器 CONSET 写入 0x40。

11.11.2 启动主机发送功能

通过建立缓冲区、指针和数据计数、然后启动起始条件来执行主发送操作。

1. 初始化主机数据计数器。
2. 建立数据将被发送到的从属地址，并且添加写位。
3. 向 CONSET 写入 0x20 来置位 STA 位。
4. 在主发送缓冲区内建立要发送的数据。
5. 初始化主机数据计数器来匹配正在发送的信息长度。
6. 退出

11.11.3 启动主机接收功能

通过建立缓冲区、指针和数据计数、然后启动起始条件来执行主接收操作。

1. 初始化主机数据计数器。
2. 建立数据将被发送到的从属地址，并且添加读位。
3. 向 CONSET 写入 0x20 来置位 STA 位。
4. 建立主接收缓冲区。
5. 初始化主机数据计数器来匹配接收到的信息长度。
6. 退出

11.11.4 I²C 中断程序

确定 I²C 的状态和处理该状态的状态程序。

1. 从 STA 中读出 I²C 的状态。
2. 使用状态值跳转到 26 个可能状态程序中的一个。

11.11.5 无指定模式的状态

11.11.5.1 状态：0x00

总线错误。进入不可寻址的从机模式并释放总线。

1. 向 CONSET 写入 0x14 来置位 STO 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

11.11.5.2 主机状态

状态 08 和 10 适用于主发送模式和主接收模式。R/W 位决定了下一个状态是在主发送模式中还是在主接收模式中。

11.11.5.3 状态：0x08

已发送起始条件。即将发送从属地址 + R/W 位和接收 ACK 位。

1. 向 DAT 写入从属地址和 R/W 位。
2. 向 CONSET 写入 0x04 来置位 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 建立主发送模式数据缓冲区。
5. 建立主接收模式数据缓冲区。
6. 初始化主机数据计数器。
7. 退出

11.11.5.4 状态：0x10

已发送重复起始条件。即将发送从属地址 + R/W 位和接收 ACK 位。

1. 向 DAT 写入从属地址和 R/W 位。
2. 向 CONSET 写入 0x04 来置位 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 建立主发送模式数据缓冲区。
5. 建立主接收模式数据缓冲区。
6. 初始化主机数据计数器。
7. 退出

11.11.6 主发送状态

11.11.6.1 状态: 0x18

之前状态为 8 或 10, 已发送从属地址和写操作位, 并接收了 ACK。即将发送第一个数据字节和接收 ACK 位。

1. 将主发送缓冲区的第一个数据字节装入 DAT。
2. 向 CONSET 写入 0x04 来置位 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 主发送缓冲区指针加 1。
5. 退出

11.11.6.2 状态: 0x20

已发送从属地址和写操作位并接收了非 ACK。即将发送停止条件。

1. 向 CONSET 写入 0x14 来置位 STO 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

11.11.6.3 状态: 0x28

已发送数据并接收了 ACK。如果发送的数据是最后一个数据字节则发送一个停止条件, 否则发送下一个数据字节。

1. 主机数据计数器减 1, 如果发送的不是最后一个数据字节就跳至第 5 步。
2. 向 CONSET 写入 0x14 来置位 STO 和 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 退出
5. 将主发送缓冲区的下一个数据字节装入 DAT。
6. 向 CONSET 写入 0x04 来置位 AA 位。
7. 向 CONCLR 写入 0x08 来清除 SI 标志。
8. 主发送缓冲区指针加 1
9. 退出

11.11.6.4 状态: 0x30

已发送数据并接收到非应答。即将发送停止条件。

1. 向 CONSET 写入 0x14 来置位 STO 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

11.11.6.5 状态: 0x38

仲裁已在发送从属地址和写操作位或数据的过程中丢失。总线已被释放且进入非寻址的从机模式。当总线再次空闲时将发送一个新的起始条件。

1. 向 CONSET 写入 0x24 来置位 STA 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

11.11.7 主接收状态

11.11.7.1 状态: 0x40

之前状态为 08 或 10。已发送从属地址和读操作位，并接收了 ACK。将接收数据和返回 ACK。

1. 向 CONSET 写入 0x04 来置位 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

11.11.7.2 状态: 0x48

已发送从属地址和读操作位并接收了非 ACK。即将发送停止条件。

1. 向 CONSET 写入 0x14 来置位 STO 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

11.11.7.3 状态: 0x50

已接收数据，已返回 ACK。将从 DAT 读取数据。将接收其它的数据。如果这是最后一个数据字节，则返回非应答，否则返回 ACK。

1. 读取 DAT 中的数据字节，存放到主机接收缓冲区。
2. 主机数据计数器减 1，如果发送的不是最后一个数据字节就跳至第 5 步。
3. 向 CONCLR 写入 0x0C 来清除 SI 标志和 AA 位。
4. 退出
5. 向 CONSET 写入 0x04 来置位 AA 位。
6. 向 CONCLR 写入 0x08 来清除 SI 标志。
7. 主接收缓冲区指针加 1
8. 退出

11.11.7.4 状态: 0x58

已接收数据，已返回非 ACK。将从 DAT 读取数据。即将发送停止条件。

1. 读取 DAT 中的数据字节，存放到主机接收缓冲区。
2. 向 CONSET 写入 0x14 来置位 STO 和 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 退出

11.11.8 从接收状态

11.11.8.1 状态: 0x60

已接收到自身从属地址和写操作位, 已返回 ACK。将接收数据和返回 ACK。

1. 向 CONSET 写入 0x04 来置位 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 建立从接收模式数据缓冲区。
4. 初始化从机数据计数器。
5. 退出

11.11.8.2 状态: 0x68

用作总线主机时仲裁已在传输从属地址和 R/W 位时丢失。已接收到自身从属地址和写操作位, 已返回 ACK。将接收数据和返回 ACK。STA 设为再次释放总线后重启主机模式。

1. 向 CONSET 写入 0x24 来置位 STA 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 建立从接收模式数据缓冲区。
4. 初始化从机数据计数器。
5. 退出。

11.11.8.3 状态: 0x70

已接收到通用调用和返回 ACK。将接收数据和返回 ACK。

1. 向 CONSET 写入 0x04 来置位 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 建立从接收模式数据缓冲区。
4. 初始化从机数据计数器。
5. 退出

11.11.8.4 状态: 0x78

用作总线主机时仲裁已在传输从属地址和 R/W 位时丢失。已接收到通用调用和返回 ACK。将接收数据和返回 ACK。STA 设为再次释放总线后重启主机模式。

1. 向 CONSET 写入 0x24 来置位 STA 和 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 建立从接收模式数据缓冲区。
4. 初始化从机数据计数器。
5. 退出

11.11.8.5 状态: 0x80

之前寻址自身从属地址。已接收到数据并返回 **ACK**。将读取其它数据。

1. 读取 **DAT** 中的数据字节, 存放从机接收缓冲区。
2. 从机数据计数器减 1, 如果发送的不是最后一个数据字节就跳至第 5 步。
3. 向 **CONCLR** 写入 0x0C 来清除 **SI** 标志和 **AA** 位。
4. 退出。
5. 向 **CONSET** 写入 0x04 来置位 **AA** 位。
6. 向 **CONCLR** 写入 0x08 来清除 **SI** 标志。
7. 从接收缓冲区指针加 1。
8. 退出

11.11.8.6 状态: 0x88

之前寻址自身从属地址。已接收到数据并返回非 **ACK**。不会保存接收到的数据。进入非寻址的从机模式。

1. 向 **CONSET** 写入 0x04 来置位 **AA** 位。
2. 向 **CONCLR** 写入 0x08 来清除 **SI** 标志。
3. 退出

11.11.8.7 状态: 0x90

之前寻址通用调用地址。已接收数据, 已返回 **ACK**。将保存接收到的数据。只接收第一个数据字节并返回 **ACK**。接收其它数据字节后返回非应答。

1. 读取 **DAT** 中的数据字节, 存放从机接收缓冲区。
2. 向 **CONCLR** 写入 0x0C 来清除 **SI** 标志和 **AA** 位。
3. 退出

11.11.8.8 状态: 0x98

之前寻址通用调用地址。已接收数据, 已返回非 **ACK**。不会保存接收到的数据。进入非寻址的从机模式。

1. 向 **CONSET** 写入 0x04 来置位 **AA** 位。
2. 向 **CONCLR** 写入 0x08 来清除 **SI** 标志。
3. 退出

11.11.8.9 状态: 0xA0

已接收停止条件或重复起始条件, 但仍作为从机寻址。不会保存数据。进入非寻址的从机模式。

1. 向 **CONSET** 写入 0x04 来置位 **AA** 位。
2. 向 **CONCLR** 写入 0x08 来清除 **SI** 标志。
3. 退出

11.11.9 从发送状态

11.11.9.1 状态：0xA8

已接收到自身从属地址和读操作位，已返回 ACK。将发送数据和接收 ACK 位。

1. 将从机发送缓冲区的第一个数据字节装入 DAT。
2. 向 CONSET 写入 0x04 来置位 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 建立从发送模式数据缓冲区。
5. 从发送缓冲区指针加 1。
6. 退出

11.11.9.2 状态：0xB0

用作总线主机时，在传输从属地址和 R/W 位时丢失仲裁。已接收到自身从属地址和读操作位，已返回 ACK。将发送数据和接收 ACK 位。STA 设为再次释放总线后重启主机模式。

1. 将从机发送缓冲区的第一个数据字节装入 DAT。
2. 向 CONSET 写入 0x24 来置位 STA 和 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 建立从发送模式数据缓冲区。
5. 从发送缓冲区指针加 1。
6. 退出

11.11.9.3 状态：0xB8

已发送数据并接收了 ACK。将发送数据和接收 ACK 位。

1. 将从机发送缓冲区的数据字节装入 DAT。
2. 向 CONSET 写入 0x04 来置位 AA 位。
3. 向 CONCLR 写入 0x08 来清除 SI 标志。
4. 从发送缓冲区指针加 1。
5. 退出

11.11.9.4 状态：0xC0

已发送数据并接收到非应答。进入非寻址的从机模式。

1. 向 CONSET 写入 0x04 来置位 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出。

11.11.9.5 状态：0xC8

已发送最后一个数据字节并接收到 ACK。进入非寻址的从机模式。

1. 向 CONSET 写入 0x04 来置位 AA 位。
2. 向 CONCLR 写入 0x08 来清除 SI 标志。
3. 退出

12.1 本章导读

SSP 控制器在所有 LPC122x 部件上都可用。

12.2 基本配置

SSP 的时钟和电源由以下各项控制：

1. SYSAHBCLKCTRL 寄存器（见[表 21](#)）。
2. SSP 时钟分频器寄存器中启用的 SSP_PCLK（见[表 22](#)）。
此时钟供 SSP 预分频器使用。

SSP_PCLK 时钟可以在 SSP 寄存器（见[表 22](#)）中禁用。而 SSP 模块可以通过系统 AHB 时钟控制寄存器位 11（见[表 21](#)）禁用以降低功耗。

12.3 特性

- 兼容摩托罗拉 SPI、4 线德州仪器 SSI 和国家半导体 Microwire 总线
- 同步串行通信
- 主机或从机操作
- 同时适用于发送与接收的 8 帧 FIFO
- 4 位至 16 位帧

12.4 描述

SSP 是同步串行端口 (SSP) 控制器，可控制 SPI、4 线 SSI 或 Microwire 总线的操作。它可以与总线上的多个主机和从机进行交互。在指定数据传输中，总线上只有一个主机和一个从机进行通信。数据传输原则上为全双工方式，4 位到 16 位数据帧由主机发送到从机或由从机发送到主机。实际上，通常情况下只有一个方向上的数据流包含有意义的数字。

12.5 引脚说明

表 205. SSP 引脚说明

引脚名称	类型	接口引脚名称 / 功能			引脚说明
		SPI	SSI	Microwire	
SCK	I/O	SCK	CLK	SK	串行时钟。 SCK/CLK/SK 是用于同步数据传输的时钟信号。由主机驱动，从机接收。当使用 SPI 接口时，可将时钟编程为高电平有效或低电平有效，否则，它一直是高电平有效。SCK 只在数据传输期间跳变。在其它时间，SSP 接口使其保持非工作状态或不驱动它（使其处于高阻抗状态）。
SSEL	I/O	SSEL	FS	CS	帧同步 / 从机选择。 当 SSP 接口为总线主机时，它在串行数据发起前将该信号驱动到工作状态，再在发送数据后将信号释放到非工作状态。该信号为高电平有效还是低电平有效取决于所选择的总线和模式。当 SSP 接口为总线从机时，该信号根据使用的协议限定从主机发出的数据。 当只有一个总线主机和一个总线从机时，来自主机的帧同步或从机选择信号可直接连接到从机的相应输入。当总线上有多个从机时，通常必需进一步限制其帧选择 / 从机选择输入，以避免多个从机对传输作出响应。
MISO	I/O	MISO	DR(M) DX(S)	SI(M) SO(S)	主机输入从机输出。 MISO 信号将串行数据由从机传输到主机。当 SSP 是从机时，从该信号上输出串行数据。当 SSP 为主机时，它记录从该信号发出的串行数据。当 SSP 为从机，且未被 FS/SSEL 选择时，它不会驱动该信号（使其处于高阻抗状态）。
MOSI	I/O	MOSI	DX(M) DR(S)	SO(M) SI(S)	主机输出从机输入。 MOSI 信号将串行数据从主机传输到从机。当 SSP 为主机时，从该信号上输出串行数据。当 SSP 为从机时，它记录从该信号发出的串行数据。

12.6 寄存器描述

SSP 控制器的寄存器地址如[表 206](#) 所示。

表 206. 寄存器简介：SSP（基址 0x4004 0000）

名称	访问类型	地址偏移	描述	复位值 [1]
CR0	R/W	0x000	控制寄存器 0。选择串行时钟速率、总线类型和数据大小。	0
CR1	R/W	0x004	控制寄存器 1。选择主机 / 从机及其他模式。	0
DR	R/W	0x008	数据寄存器。写满发送 FIFO，读空接收 FIFO。	0
SR	RO	0x00C	状态寄存器	0x0000 0003
CPSR	R/W	0x010	时钟预分频寄存器	0
IMSC	R/W	0x014	中断屏蔽设置和清除寄存器	0
RIS	RO	0x018	原始中断状态寄存器	-

表 206. 寄存器简介：SSP（基址 0x4004 0000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
MIS	RO	0x01C	屏蔽中断状态寄存器	0x0000 0008
ICR	WO	0x020	SSPICR 中断清除寄存器	不适用
DMACR	R/W	0x024	DMA 控制寄存器	0

[1] 复位值仅反映已使用位中保存的数据，它不包括保留位的内容。

12.6.1 SSP 控制寄存器 0

此寄存器控制 SSP 控制器的基本操作。

表 207. SSP 控制寄存器 0（CR0 - 地址 0x4004 0000）位描述

位	符号	值	描述	复位值
3:0	DSS		数据大小选择。该字段控制每帧中传输的位数。不支持且不使 用值 0000-0010。	0000
		0x3	4 位传输	
		0x4	5 位传输	
		0x5	6 位传输	
		0x6	7 位传输	
		0x7	8 位传输	
		0x8	9 位传输	
		0x9	10 位传输	
		0xA	11 位传输	
		0xB	12 位传输	
		0xC	13 位传输	
		0xD	14 位传输	
		0xE	15 位传输	
		0xF	16 位传输	
5:4	FRF		帧格式。	00
		0x0	SPI	
		0x1	德州仪器	
		0x2	Microwire	
		0x3	不支持且不使用该组合。	
6	CPOL		时钟输出极性。该位只用于 SPI 模式。	0
		0	SSP 控制器使帧之间的总线时钟保持为低电平。	
		1	SSP 控制器使帧之间的总线时钟保持为高电平。	

表 207. SSP 控制寄存器 0（CR0 - 地址 0x4004 0000）位描述（续）

位	符号	值	描述	复位值
7	CPHA		时钟输出相位。该位只用于 SPI 模式。	0
		0	SSP 控制器在帧传输的第一次时钟跃迁时捕获串行数据，也就是说跃迁远离时钟线的帧间状态。	
		1	SSP 控制器在帧传输的第二次时钟跃迁时捕获串行数据，也就是说跃迁回到时钟线的帧间状态。	
15:8	SCR		串行时钟速率。总线上传送的每一个位对应的预分频器输出时钟数减 1。假设 CPDVSRR 为预分频器，APB 时钟 PCLK 计时预分频器，则位频率为 $PCLK/(CPDVSRR \times [SCR+1])$ 。	0x00
31:16	-		保留。	-

12.6.2 SSP 控制寄存器 1

此寄存器控制 SSP 控制器操作的某些方面。

表 208. SSP 控制寄存器 1（CR1 - 地址 0x4004 0004）位描述

位	符号	值	描述	复位值
0	LBM		环回模式。	0
		0	正常操作时。	
		1	串行输入取自串行输出（MOSI 或 MISO），而不是串行输入引脚（分别为 MISO 或 MOSI）。	
1	SSE		SSP 使能。	0
		0	禁用 SSP 控制器。	
		1	SSP 控制器可与串行总线上的其他设备相互通信。设置此位前，软件应向其他 SSP 寄存器和中断控制器寄存器写入适当的控制信息。	
2	MS		主机 / 从机模式。只有在 SSE 位为 0 时，才能对该位执行写入操作。	0
		0	SSP 控制器作为总线上的主机，驱动 SCLK、MOSI 和 SSEL 线并接收 MISO 线。	
		1	SSP 控制器作为总线上的从机，驱动 MISO 线并接收 SCLK、MOSI 及 SSEL 线。	
3	SOD		从机输出禁用。只有在从机模式下才与此位有关（MS = 1）。如果值为 1，则禁止此 SSP 控制器驱动发送数据线（MISO）。	0
31:4	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.6.3 SSP 数据寄存器

软件可向该寄存器写入要发送的数据，并读取已接收的数据。

表 209. SSP 数据寄存器（DR - 地址 0x4004 0008）位描述

位	符号	描述	复位值
15:0	DATA	写： 当状态寄存器中的 TNF 位为 1（指示 Tx FIFO 未滿）时，软件就可以将要在日后帧中发送的数据写入该寄存器。如果 Tx FIFO 先前为空且总线上的 SSP 控制器空闲，则将立即开始发送数据。否则，一旦先前所有的数据都已发送（和接收），即会发送写入该寄存器的数据。如果数据长度小于 16 位，则软件必须使写入该寄存器的数据向右对齐。 读： 只要状态寄存器中的 RNE 位为 1（指示 Rx FIFO 未空），软件就可以从该寄存器读取数据。当软件读取该寄存器时，SSP 控制器返回 Rx FIFO 中最早接收到的帧数据。如果数据长度小于 16 位，则使此字段的数据向右对齐，更高阶位用 0 填充。	0x0000
31:16	-	保留	-

12.6.4 SSP 状态寄存器

该只读寄存器反映 SSP 控制器的当前状态。

表 210. SSP 状态寄存器（SR - 地址 0x4004 000C）位描述

位	符号	描述	复位值
0	TFE	发送 FIFO 为空。如果发送 FIFO 为空，则该位为 1；反之为 0。	1
1	TNF	发送 FIFO 未滿。如果 Tx FIFO 已滿，则该位为 0；反之为 1。	1
2	RNE	接收 FIFO 未空。如果接收 FIFO 为空，则该位为 0；反之为 1。	0
3	RFF	接收 FIFO 滿。如果接收 FIFO 已滿，则该位为 1；反之为 0。	0
4	BSY	忙。如果 SSP 控制器空闲，则该位为 0；或如果当前正在发送 / 接收一个帧和 / 或 Tx FIFO 未空，则该位为 1。	0
31:5	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.6.5 SSP 时钟预分频寄存器

该寄存器控制预分频器的分频因数，预分频器对 SSP 外设时钟 SSP_PCLK 进行分频以产生预分频器时钟，而预分频器时钟被 CR0 中的 SCR 因数分频以确定位时钟。

表 211. SSP 时钟预分频寄存器（CPSR - 地址 0x4004 0010）位描述

位	符号	描述	复位值
7:0	CPSDVSR	此偶数值介于 2 至 254 之间，SSP_PCLK 通过该值进行分频以产生预分频器输出时钟。位 0 始终读取为 0。	0
31:8	-	保留	-

注意事项：必须适当地对 CPSR 值进行初始化，否则 SSP 控制器不能正确发送数据。

在从机模式下，主机提供的 SSP 时钟速率不能超过[表 22](#)中所选 SSP 外设时钟的 1/12。CPSR 寄存器的内容与此无关。

在主机模式下， $CPSDVSR_{min} = 2$ 或更大的值（只能为偶数）。

12.6.6 SSP 中断屏蔽设置 / 清除寄存器 (IMSC - 0x4004 0014)

该寄存器控制是否启用 SSP 控制器中 4 个可能的中断条件。请注意，ARM 所使用的术语 masked 与普通计算机术语中的意思相反，“masked”在普通计算机术语中的意思是禁用，而 ARM 使用的 masked 一词指启用。为了避免混淆，我们将不使用 masked 一词。

表 212. SSP 中断屏蔽设置 / 清除寄存器（IMSC - 地址 0x4004 0014）位描述

位	符号	描述	复位值
0	RORIM	出现接收溢出（即当 Rx FIFO 已满且另一个帧完全接收）时，软件应 将此位设置为启用中断。ARM 规范指明，出现这种情况时，前面的帧 数据会被新的帧数据覆盖。	0
1	RTIM	出现接收超时条件时，软件应将此位设置为启用中断。当 Rx FIFO 未 空且在“超时期限”内没有读取任何数据时，将会出现接收超时。 从机和主机模式的超时期限都相同，并由 SSP 位率决定： 在 PCLK/(CPSDVSR × [SCR+1]) 时为 32 位。	0
2	RXIM	Rx FIFO 至少有一半为满时，软件应将此位设置为使能中断。	0
3	TXIM	Tx FIFO 至少有一半为空时，软件应将此位设置为启用中断。	0
31:4	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.6.7 SSP 原始中断状态寄存器

不管 IMSC 中是否启用中断，只要出现有效的中断条件，该只读寄存器便会在相应的位置包
含 1。

表 213. SSP 原始中断状态寄存器（RIS - 地址 0x4004 0018）位描述

位	符号	描述	复位值
0	RORRIS	如果在 Rx FIFO 已满时完整接收到另一个帧，则该位为 1。ARM 规范 指明，出现这种情况时，前面的帧数据会被新的帧数据覆盖。	0
1	RTRIS	如果 Rx FIFO 不为空，且在“超时期限”内未读取，则该位为 1。 从机和主机模式的超时期限都相同，并由 SSP 位率决定： 在 PCLK/(CPSDVSR × [SCR+1]) 时为 32 位。	0
2	RXRIS	如果 Rx FIFO 至少有一半为满时，则该位为 1。	0
3	TXRIS	如果 Tx FIFO 至少有一半为空时，则该位为 1。	1
31:4	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.6.8 SSP 屏蔽中断状态寄存器

该寄存器是一个只读寄存器，当中断条件出现且相应的中断在 IMSC 中启用时，该寄存器
中对应的位为 1。当出现 SSP 中断时，中断服务例程会读取此寄存器以确定中断的原因。

表 214. SSP 屏蔽中断状态寄存器（MIS - 地址 0x4004 001C）位描述

位	符号	描述	复位值
0	RORMIS	如果在 Rx FIFO 已满时完整接收另一个帧，则此位为 1，且此中断启 用。	0
1	RTMIS	如果 Rx FIFO 不为空并在“超时期限”内未读取，则此位为 1，且 此中断启用。从机和主机模式的超时期限都相同，并由 SSP 位率决定 在 PCLK/(CPSDVSR × [SCR+1]) 时为 32 位。	0
2	RXMIS	如果 Rx FIFO 至少有一半为满，则此位为 1，且此中断启用。	0
3	TXMIS	如果 Tx FIFO 至少有一半为空，则此位为 1，且此中断启用。	0
31:4	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.6.9 SSP 中断清除寄存器

软件可向该只写寄存器写入 1 个或多个 1 以清除 SSP 控制器中相应的中断条件。请注意，其他两个中断条件可通过写入或读取相应的 FIFO 清除，或通过清除 IMSC 中的相应位将其禁用。

表 215. SSP 中断清除寄存器（ICR - 地址 0x4004 0020）位描述

位	符号	描述	复位值
0	RORIC	向该位写入 1 可清除 Rx FIFO 为满时接收帧中断。	不适用
1	RTIC	向该位写入 1 可清除 Rx FIFO 不为空且在超时期限内未读取中断。 从机和主机模式的超时期限都相同，并由 SSP 位率决定： 在 PCLK/(CPSDVSR × [SCR+1]) 时为 32 位。	不适用
31:2	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.6.10 SSP DMA 控制寄存器

DMACR 寄存器为 DMA 控制寄存器。它是读 / 写寄存器。

表 216. SSP DMA 控制寄存器（DMACR - 地址 0x4004 0024）位描述

位	符号	值	描述	复位值
0	RXDMAE		接收 DMA 使能	0
		0	接收 DMA 禁用。	
		1	接收 FIFO 的 DMA 启用。	
1	TXDMAE		发送 DMA 使能	0
		0	发送 DMA 禁用。	
		1	发送 FIFO 的 DMA 启用。	
31:2	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

12.7 功能说明

12.7.1 德州仪器同步串行帧格式

图 29 显示了 SSP 模块支持的 4 线德州仪器同步串行帧格式。

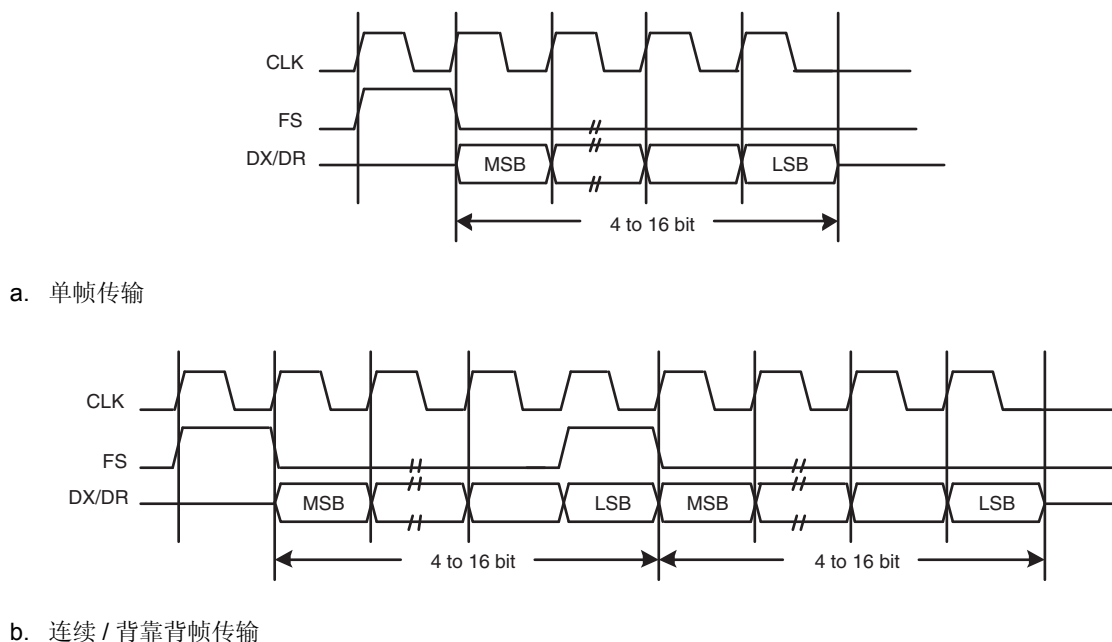


图 29. 德州仪器同步串行帧格式: a) 单帧和 b) 连续 / 背靠背 2 帧传输

对于在该模式下配置为主机的设备，CLK 和 FS 被强制为低电平，且只要 SSP 空闲，发送数据线 DX 便会处于 3 态模式。一旦发送 FIFO 的底端含有数据，FS 就会变为高电平，并持续一个 CLK 周期。要发送的值也会从发送 FIFO 传输到发送逻辑的串行移位寄存器。在下一个 CLK 上升沿上，4 位到 16 位数据帧的 MSB 输出到 DX 引脚。同样，接收数据的 MSB 由片外串行从器件传送到 DR 引脚。

在每个 CLK 的下降沿，SSP 和片外串行从器件将各个数据位放入其串行移位器。LSB 被锁存后，在 CLK 的第一个上升沿，接收的数据从串行移位器传输到接收 FIFO。

12.7.2 SPI 帧格式

SPI 接口是 4 线接口，其中 SSEL 信号用作从机选择。SPI 格式的主要特性是 SCK 信号的非工作状态和相位可通过对 SSPCR0 控制寄存器内的 CPOL 和 CPHA 位编程设定。

12.7.2.1 时钟极性 (CPOL) 及相位 (CPHA) 控制

当 CPOL 时钟极性控制位为低电平时，它会在 SCK 引脚产生一个稳定状态的低电平值。如果 CPOL 时钟极性控制位为高电平，则在没有传输数据时，它会在 CLK 引脚上产生一个稳态高电平值。

CPHA 控制位选择捕获数据及允许数据更改状态的时钟沿。无论在第一个数据捕获沿之前允许还是不允许时钟跃迁，都会对传输的第一位产生极大影响。当 CPHA 相位控制位为低电平时，则在第一次出现时钟沿跃迁时捕获数据。如果 CPHA 时钟相位控制位为高电平，则在第二次出现时钟沿跃迁时捕获数据。

12.7.2.2 CPOL=0, CPHA=0 时的 SPI 格式

CPOL = 0 且 CPHA = 0 时 SPI 格式的单帧和连续帧传输信号序列如图 30 所示。

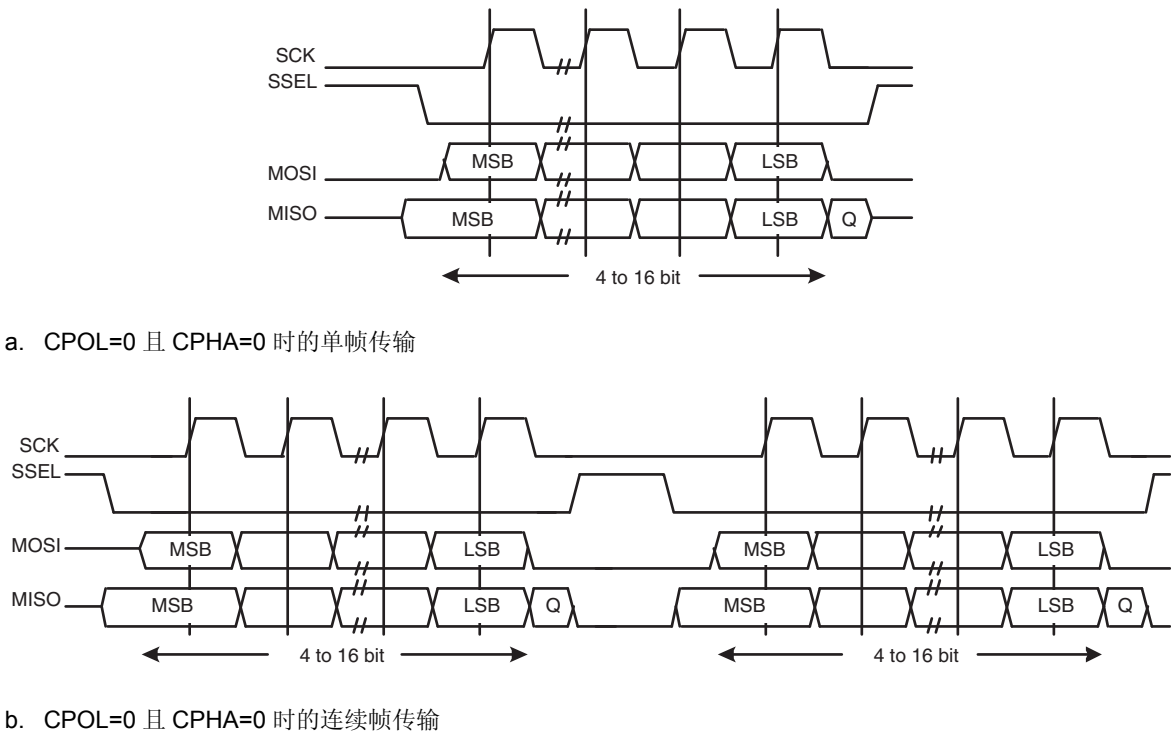


图 30. CPOL=0 且 CPHA=0 时的 SPI 帧格式 (a) 单帧和 b) 连续帧传输)

该配置中，在空闲期间：

- CLK 信号被强制为低电平。
- SSEL 被强制为高电平。
- 发送 MOSI/MISO 垫片处于高阻态。

如果启用 SSP，且在发送 FIFO 中存在有效数据，则 SSEL 主机信号被驱动为低电平，指示数据传输开始。这样，从机数据就可以进入到主机的主 MISO 输入线中。启用主机的主 MOSI。

1/2 个 SCK 周期后，有效的主机数据将被传输到 MOSI 引脚。由于主机和从机数据均已设定，因此再过 1/2 个 SCK 周期，SCK 主时钟引脚就会变为高电平。

此时，在 SCK 信号的上升沿捕获数据，并在 SCK 信号的下降沿传播数据。

发送单个字时，在传输完数据字的所有位后，在捕获到最后一位后的一个 SCK 周期内，SSEL 线将返回到其空闲高电平状态。

但是，在进行连续的背靠背传输时，传输各数据字之间的 SSEL 信号必须为高电平。这是因为如果 CPHA 位为逻辑 0，从机选择引脚会冻结串行外设寄存器中的数据并且不允许改变数据。因此，主机设备必须拉高各数据传输之间的从器件的 SSEL 引脚，以启用串行外设数据写入操作。连续传输完成后，SSEL 引脚在捕获到最后一位后的一个 SCK 周期内将返回至空闲状态。

12.7.2.3 CPOL=0，CPHA=1 时的 SPI 格式

CPOL = 0 且 CPHA = 1 时 SPI 格式的传输信号序列如图 31 所示，包含单帧和连续帧传输。

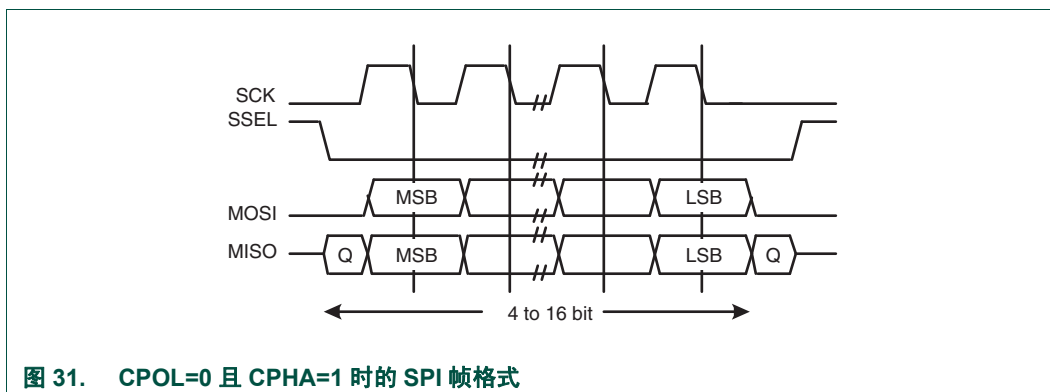


图 31. CPOL=0 且 CPHA=1 时的 SPI 帧格式

该配置中，在空闲期间：

- CLK 信号被强制为低电平。
- SSEL 被强制为高电平。
- 发送 MOSI/MISO 垫片处于高阻态。

如果启用 SSP，且在发送 FIFO 中存在有效数据，则 SSEL 主机信号被驱动为低电平，指示数据传输开始。启用主机的 MOSI 引脚。再过 1/2 个 SCK 周期后，主机和从机的有效数据都将在其各自的传输线上启用。同时，SCK 在上升沿跃迁时启用。

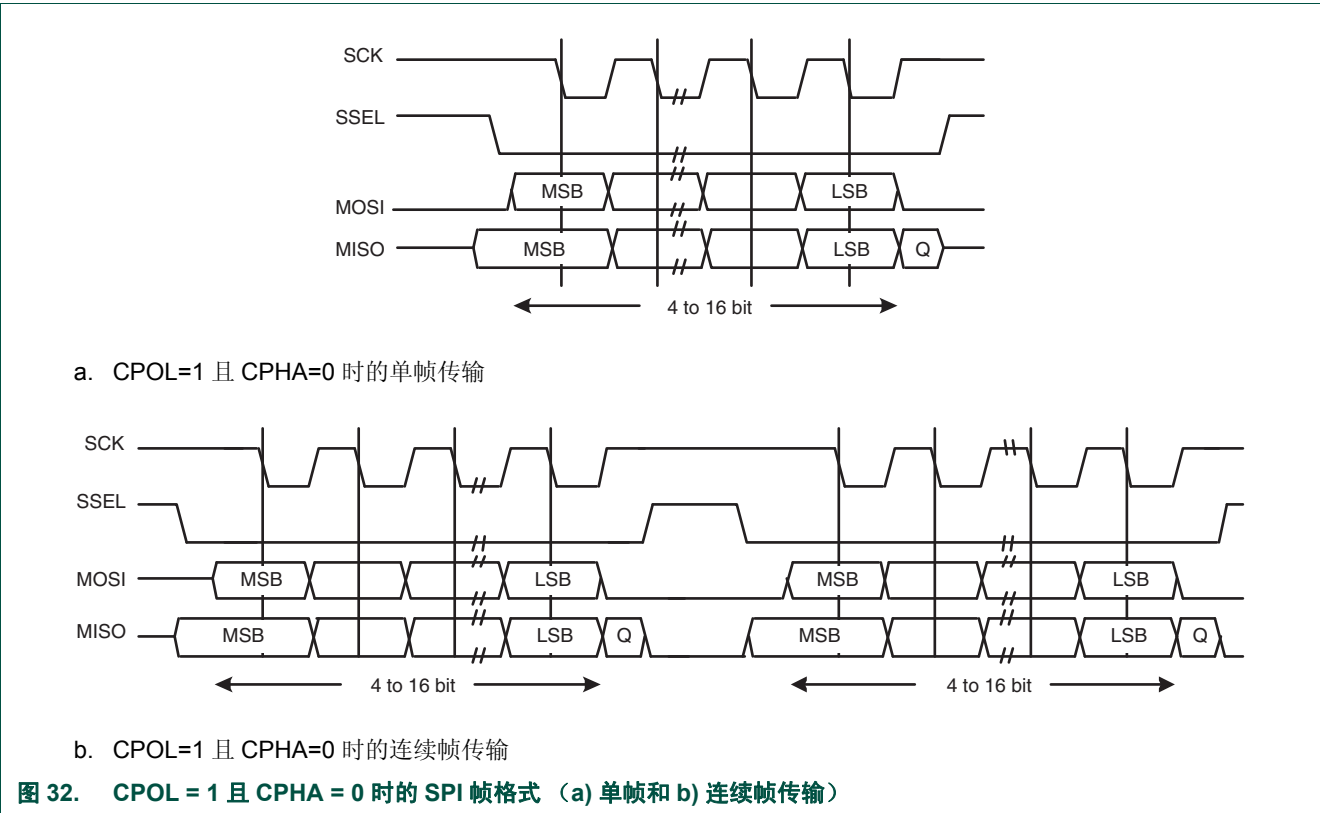
然后，在 SCK 信号的下降沿捕获数据，并在 SCK 信号的上升沿传播数据。

发送单个字时，在传输完所有位后，在捕获到最后一位后的一个 SCK 周期内，SSEL 线返回其空闲高电平状态。

当进行连续的背靠背传输时，连续的数据字之间 SSEL 引脚保持为低电平，终止条件与发送单个字时相同。

12.7.2.4 CPOL = 1 且 CPHA = 0 时的 SPI 格式

CPOL=1 且 CPHA=0 时，SPI 格式的单帧和连续帧传输信号序列如[图 32](#)所示。



该配置中，在空闲期间：

- CLK 信号被强制为高电平。
- SSEL 被强制为高电平。
- 发送 MOSI/MISO 垫片处于高阻态。

如果启用 SSP，且在发送 FIFO 中存在有效数据，则 SSEL 主机信号被驱动为低电平，指示数据传输开始，这会使从机数据立即传输到主机的 MISO 线上。启用主机的 MOSI 引脚。

1/2 个 SCK 周期后，有效的主机数据将被传输到 MOSI 线。由于主机和从机数据均已设定，因此再过 1/2 个 SCK 周期，SCK 主时钟引脚就会变为低电平。这意味着，在 SCK 信号的下降沿捕获数据并在 SCK 信号的上升沿传播数据。

发送单个字时，在传输完所有位后，在捕获最后一位后的一个 SCK 周期内，SSEL 线将返回到其空闲高电平状态。

但是，在进行连续的背靠背传输时，传输各数据字之间的 SSEL 信号必须为高电平。这是因为如果 CPHA 位为逻辑 0，从机选择引脚会冻结串行外设寄存器中的数据并且不允许改变数据。因此，主机设备必须拉高各数据传输之间的从器件的 SSEL 引脚，以启用串行外设数据写入操作。连续传输完成后，SSEL 引脚在捕获到最后一位后的一个 SCK 周期内将返回至空闲状态。

12.7.2.5 CPOL = 1 且 CPHA = 1 时的 SPI 格式

CPOL = 1 且 CPHA = 1 时 SPI 格式的传输信号序列如图 33 所示，包含单帧和连续帧传输。

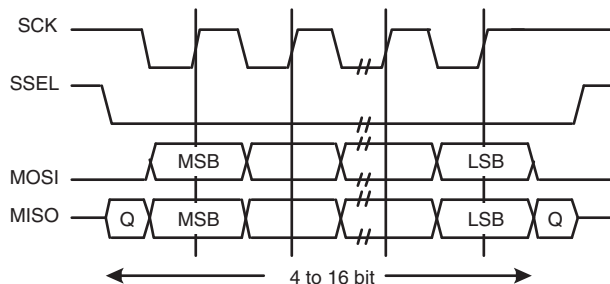


图 33. CPOL = 1 且 CPHA = 1 时的 SPI 帧格式

该配置中，在空闲期间：

- CLK 信号被强制为高电平。
- SSEL 被强制为高电平。
- 发送 MOSI/MISO 垫片处于高阻态。

如果启用 SSP，且在发送 FIFO 中存在有效数据，则 SSEL 主机信号被驱动为低电平，指示数据传输开始。启用主机的 MOSI。再过 1/2 个 SCK 周期后，主机和从机数据都将在其各自的传输线上启用。同时，SCK 在下降沿跃迁时启用。然后，在 SCK 信号的上升沿捕获数据，并在 SCK 信号的下降沿传播数据。

发送单个字时，在传输完所有位后，在捕获最后一位后的一个 SCK 周期内，SSEL 线返回到其空闲高电平状态。对于连续的背靠背传输，SSEL 引脚保持处于其有效的低电平状态，直到捕获最后一个字的最后一位为止，然后返回到其空闲状态（如上所述）。通常，当进行连续的背靠背传输时，连续的数据字之间 SSEL 引脚保持为低电平，终止条件与传送单个字时相同。

12.7.3 半导体 Microwire 帧格式

图 34 显示了单帧的 Microwire 帧格式。图 35 显示了进行背靠背帧传输时的 Microwire 帧格式。

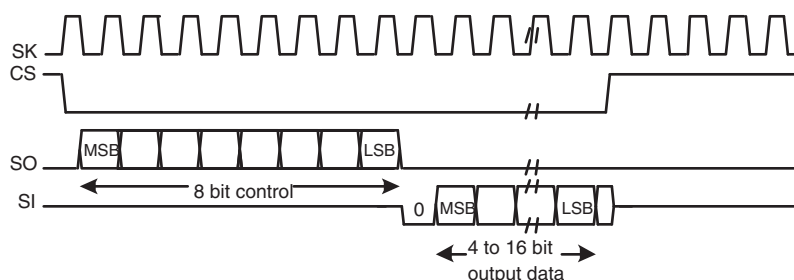


图 34. Microwire 帧格式（单帧传输）

Microwire 格式与 SPI 格式非常相似，都是使用主 - 从消息传递技术，但它采用的传输方式是半双工方式而不是全双工方式。每次串行传输均从一个 8 位控制字开始，由 SSP 发送到片外从器件。在此发送过程中，SSP 不会接收输入的数据。发送完消息后，片外从机对其进行解码，然后在 8 位控制消息的最后一位发送结束后再等待一个串行时钟，以所需的数据进行响应。返回的数据长度为 4 至 16 位，使整个帧长度范围为 13 至 25 位。

该配置中，在空闲期间：

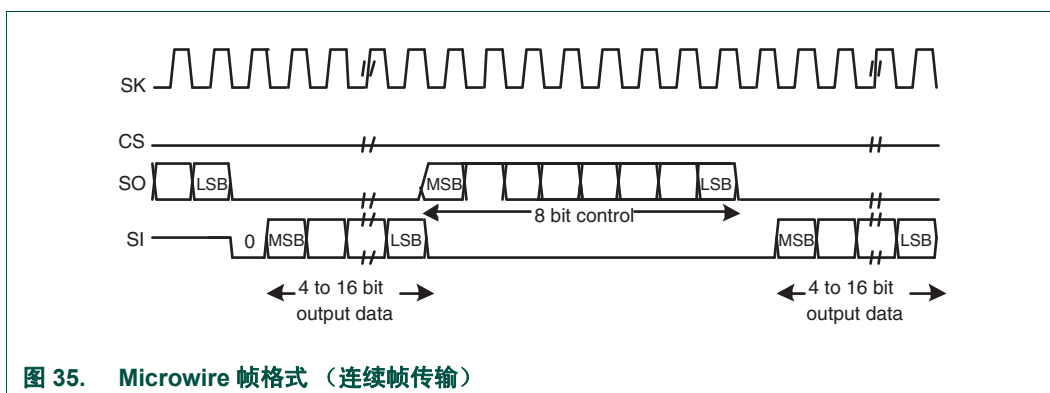
- SK 信号被强制为低电平。
- CS 被强制为高电平。
- 发送数据线 SO 被任意强制为低电平。

通过向发送 FIFO 写入控制字节来触发传送。CS 下降沿会使包含在发送 FIFO 底端的值传输到发送逻辑的串行移位寄存器，并使 8 位控制帧的 MSB 输出到 SO 引脚。CS 在帧传输期间保持低电平。SI 引脚在此传输过程中保持三态。

片外串行从器件在每个 SK 的上升沿将各控制位锁存到串行移位器中。当从器件将最后一位锁存后，控制字节会在一个时钟等待状态期间被译码，而从机则通过将数据传回至 SSP 来进行响应。每个位在 SK 的下降沿被驱动到 SI 线。SSP 依次在 SK 的上升沿锁存每个位。对于单帧传输，在帧的末端，在最后一位已锁存到接收串行移位器后的一个时钟周期内，CS 信号会被置为高电平，这会导致数据被传输到接收 FIFO。

注：LSB 被接收移位器锁存后或当 CS 引脚变为高电平时，片外从器件的接收线在 SK 下降沿呈现三态。

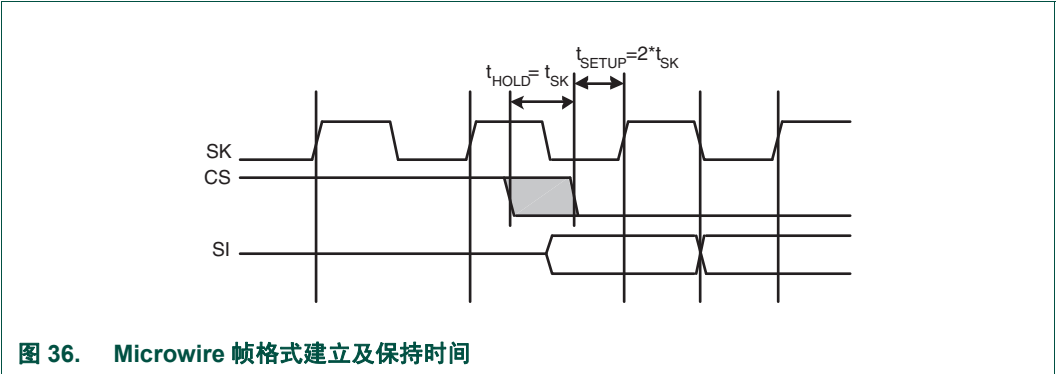
对于连续帧传输，数据传输开始和结束的方式与单帧传输相同。然而，CS 线持续有效（保持低电平），且数据以背靠背方式进行传输。当前数据帧的 LSB 被接收后，下一帧的控制字节立即发送。在帧数据的 LSB 被锁存到 SSP 后，每个收到的值将在 SK 的下降沿传输到接收移位器。



12.7.3.1 Microwire 模式下 CS 相对于 SK 的建立和保持时间要求

在 Microwire 模式下，CS 变为低电平后，SSP 从机在 SK 上升沿时对接收数据的第一位进行采样。主机驱动 SK 自由运行时必须确保 CS 信号相对于 SK 上升沿有充足的建立和保持时间。

图 36 描绘了这些建立和保持时间的要求。相对于 SK 上升沿（SSP 从机在该上升沿对接收数据的第一位进行采样），CS 的建立时间必须至少为 SK（SSP 在 SK 上运行）周期的 2 倍。相对于该边沿之前的 SK 上升沿，CS 必须保持至少一个 SK 周期。



13.1 本章导读

所有 LPC122x 部件都提供 CT16B0/1。

13.2 基本配置

16 位定时器计数器模块的外设时钟由系统时钟提供，系统时钟由 SYSAHBCLKDIV 寄存器控制（[表 20](#)）。而 CT16B0/1 模块可以通过系统 AHB 时钟控制寄存器位 7 和 8（[表 21](#)）禁能以降低功耗。

13.3 特性

- 两个带有可编程 16 位预分频器的 16 位计数器 / 定时器。
- 计数器 / 定时器操作
- 两个 16 位捕获通道，可在输入信号跃迁时快速捕获定时器值。捕获事件也可能产生一个中断。
- 可配置定时器和预分频器在指定捕获事件清零。此特性通过在输入脉冲前沿清零定时器并捕获定时器在后沿的值，方便进行脉冲宽度测量。
- 四个 16 位匹配寄存器允许：
 - 连续操作，可选择在匹配时产生中断。
 - 在匹配时停止定时器，可选择产生中断。
 - 在匹配时复位定时器，可选择产生中断。
- 两个对应于匹配寄存器的外部输出，具备以下功能：
 - 匹配时设置低电平。
 - 匹配时设置高电平。
 - 匹配时切换。
 - 匹配时不执行任何操作。
- 对于各定时器，最多 4 个匹配寄存器可配置为 PWM，允许使用最多 2 个匹配输出作为单独边沿控制的 PWM 输出。
- 最多有两个匹配寄存器可用来产生定时 DMA 请求。

13.4 应用程序

- 时间间隔定时器，用于对内部事件进行计数
- 脉冲宽度解调器（经捕获输入）
- 自由运行的定时器
- 脉冲宽度调制器（经匹配输出）

13.5 描述

每个计数器 / 定时器都设计用来计算外设时钟 (PCLK) 或外部供电时钟的周期，并且可根据 4 个匹配寄存器的值在指定定时器值处产生中断或执行其他操作。每个计数器 / 定时器还包括 1 个捕获输入，用来在输入信号跃迁时捕获定时器值，同时可根据需要产生一个中断。

在 PWM 模式下，可使用其中 2 个匹配寄存器向匹配输出引脚提供单边沿控制的 PWM 输出。建议使用不用于输出的匹配寄存器来控制 PWM 周期长度。

注：16 位计数器 / 定时器 0 (CT16B0) 和 16 位计数器 / 定时器 1 (CT16B1) 除外设基址不同外，其他功能相似。

13.6 引脚描述

表 217 简要总结了每个计数器 / 定时器相关的引脚。

表 217. 计数器 / 定时器引脚描述

引脚	类型	描述
CT16B0_CAP[1:0] CT16B1_CAP[1:0]	输入	捕获信号： 可以配置捕获引脚上的跃迁，用计数器 / 定时器中的值载入捕获寄存器，并且可以有选择性地产生中断。 计数器 / 定时器模块可选择捕获信号作为时钟源来代替 PCLK 衍生时钟。有关更多详情，请参阅 章节 13.7.11 。
CT16B0_MAT[1:0] CT16B1_MAT[1:0]	输出	CT16B0/1 的外部匹配输出： 当 CT16B0/1 (MR1:0) 的匹配寄存器与定时器计数器 (TC) 相等时，该输出可以进行切换、转入低电平、转入高电平或不执行任何操作。外部匹配寄存器 (EMR) 和 PWM 控制寄存器 (PWMCON) 控制该输出的功能。

另外，两个比较器的电平和边沿输出内部连接到每个 16 位计数器 / 定时器的剩余捕获通道 2 和 3(见[表 218](#))。有关比较器输出的详情，请参阅[章节 18.7.5](#)。

表 218. 比较器到 16 位计数器 / 定时器的连接

比较器输出	定时器捕获输入
比较器 0，电平输出	CT16B0_CAP2
比较器 0，边沿输出	CT16B0_CAP3
比较器 1，电平输出	CT16B1_CAP2
比较器 1，边沿输出	CT16B1_CAP3

13.7 寄存器描述

16 位计数器 / 定时器 0 包含的寄存器如[表 219](#) 所示，16 位计数器 / 定时器 1 包含的寄存器如[表 220](#) 所示。详细描述如下。

表 219. 寄存器简介：16 位计数器 / 定时器 0 CT16B0（基址 0x4001 0000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
IR	R/W	0x000	中断寄存器 (IR)。可向 IR 写入相应值来清除中断。可以通过读 IR 来识别 8 个中断源中哪个中断源正在被挂起。	0
TCR	R/W	0x004	定时器控制寄存器 (TCR)。TCR 用于控制定时器计数器功能。定时器计数器可通过 TCR 来禁能或复位。	0
TC	R/W	0x008	定时器计数器 (TC)。16 位 TC 每隔 PR + 1 个 PCLK 周期递增一次。通过 TCR 控制 TC。	0
PR	R/W	0x00C	预分频寄存器 (PR)。当预分频计数器（见下）与该值相等时，下个时钟 TC 加 1，PC 清零。	0
PC	R/W	0x010	预分频计数器 (PC)。16 位 PC 是一个计数器，它会增加到与 PR 中存放的值相等。当达到 PR 的值时，TC 递增，PC 清零。可通过总线接口来观察和控制 PC。	0
MCR	R/W	0x014	匹配控制寄存器 (MCR)。MCR 用于控制在匹配出现时是否产生中断及出现匹配时 TC 是否复位。	0
MR0	R/W	0x018	匹配寄存器 0(MR0)。MR0 可通过 MCR 使能，当 MR0 与 TC 匹配时复位 TC，停止 TC 和 PC，和 / 或产生中断。	0
MR1	R/W	0x01C	匹配寄存器 1(MR1)。见 MR0 描述。	0
MR2	R/W	0x020	匹配寄存器 2(MR2)。见 MR0 描述。	0
MR3	R/W	0x024	匹配寄存器 3(MR3)。见 MR0 描述。	0
CCR	R/W	0x028	捕获控制寄存器 (CCR)。CCR 控制捕获时使用捕获输入的哪个边沿来加载捕获寄存器，以及在捕获时是否产生中断。	0
CR0	RO	0x02C	捕获寄存器 0(CR0)。当 CT16B0_CAP0 输入上产生事件时，CR0 载入 TC 值。	0
CR1	RO	0x030	捕获寄存器 1(CR1)。当 CT16B0_CAP1 输入上产生事件时，CR1 载入 TC 值。	0
CR2	RO	0x034	捕获寄存器 3(CR3)。当比较器输入上产生事件时，CR2 载入 TC 值。	0
CR3	RO	0x038	捕获寄存器 3(CR3)。当比较器输入上产生事件时，CR3 载入 TC 值。	0
EMR	R/W	0x03C	外部匹配寄存器 (EMR)。EMR 控制匹配功能及外部匹配引脚 CT16B0_MAT[1:0] 和 CT16B1_MAT[1:0]。	0
-	-	0x040 - 0x06C	保留	-
CTCR	R/W	0x070	计数控制寄存器 (CTCR)。CTCR 选择在定时器模式还是在计数器模式下工作，在计数器模式下选择计数的信号和边沿。	0
PWMC	R/W	0x074	PWM 控制寄存器 (PWMCON)。PWMCON 使能 PWM 模式，用于外部匹配引脚 CT16B0_MAT[1:0] 和 CT16B1_MAT[1:0]。	0

[1] 复位值只反映了使用位的值。不包括保留位的内容。

表 220. 寄存器简介：16 位计数器 / 定时器 1 CT16B1（基址 0x4001 4000）

名称	访问类型	地址	描述	复位值 ^[1]
IR	R/W	0x000	中断寄存器 (IR)。可向 IR 写入相应值来清除中断。可以通过读 IR 来识别 8 个中断源中哪个中断源正在被挂起。	0
TCR	R/W	0x004	定时器控制寄存器 (TCR)。TCR 用于控制定时器计数器功能。定时器计数器可通过 TCR 来禁能或复位。	0
TC	R/W	0x008	定时器计数器(TC)。16位TC每隔PR+1个PCLK周期递增一次。通过TCR控制TC。	0
PR	R/W	0x00C	预分频寄存器 (PR)。当预分频计数器（见下）与该值相等时，下个时钟 TC 加 1，PC 清零。	0
PC	R/W	0x010	预分频计数器 (PC)。16 位 PC 是一个计数器，它会增加到与 PR 中存放的值相等。当达到 PR 的值时，TC 递增，PC 清零。可通过总线接口来观察和控制 PC。	0
MCR	R/W	0x014	匹配控制寄存器 (MCR)。MCR 用于控制在匹配出现时是否产生中断及出现匹配时 TC 是否复位。	0
MR0	R/W	0x018	匹配寄存器 0(MR0)。MR0 可通过 MCR 使能，当 MR0 与 TC 匹配时复位 TC，停止 TC 和 PC，和 / 或产生中断。	0
MR1	R/W	0x01C	匹配寄存器 1(MR1)。见 MR0 描述。	0
MR2	R/W	0x020	匹配寄存器 2(MR2)。见 MR0 描述。	0
MR3	R/W	0x024	匹配寄存器 3(MR3)。见 MR0 描述。	0
CCR	R/W	0x028	捕获控制寄存器 (CCR)。CCR 控制捕获时使用捕获输入的哪个边沿来加载捕获寄存器，以及在捕获时是否产生中断。	0
CR0	RO	0x02C	捕获寄存器 0(CR0)。当 CT16B1_CAP0 输入上产生事件时，CR0 载入 TC 值。	0
CR1	RO	0x030	捕获寄存器 1(CR1)。当 CT16B1_CAP1 输入上产生事件时，CR1 载入 TC 值。	0
CR2	RO	0x034	捕获寄存器 3(CR3)。当比较器输入上产生事件时，CR2 载入 TC 值。	0
CR3	RO	0x038	捕获寄存器 3(CR3)。当比较器输入上产生事件时，CR3 载入 TC 值。	0
EMR	R/W	0x03C	外部匹配寄存器 (EMR)。EMR 控制匹配功能及外部匹配引脚 CT16B0_MAT[1:0] 和 CT16B1_MAT[1:0]。	0
-	-	0x040 - 0x06C	保留	-
CTCR	R/W	0x070	计数控制寄存器 (CTCR)。CTCR 选择在定时器模式还是在计数器模式下工作，在计数器模式下选择计数的信号和边沿。	0
PWMC	R/W	0x074	PWM 控制寄存器 (PWMCON)。PWMCON 使能 PWM 模式，用于外部匹配引脚 CT16B0_MAT[1:0] 和 CT16B1_MAT[1:0]。	0

[1] 复位值只反映了使用位的值。不包括保留位的内容。

13.7.1 中断寄存器

中断寄存器包含 4 个用于匹配中断的位及 2 个用于捕获中断的位。如果有中断产生，IR 中的相应位为高电平。否则，该位为低电平。向对应的 IR 位写逻辑 1 会使中断复位。写 0 无效。清除定时器匹配中断会同时清除任何相应的 DMA 请求。

表 221. 中断寄存器 (IR, 地址 0x4001 0000 (CT16B0) 和 0x4001 4000 (CT16B1)) 位描述

位	符号	描述	复位值
0	MR0INT	匹配通道 0 的中断标志。	0
1	MR1INT	匹配通道 1 的中断标志。	0
2	MR2INT	匹配通道 2 的中断标志。	0
3	MR3INT	匹配通道 3 的中断标志。	0
4	CR0INT	捕获通道 0 事件的中断标志。	0
5	CR1INT	捕获通道 1 事件的中断标志。	0
6	CR2INT	捕获通道 2 事件的中断标志。	0
7	CR3INT	捕获通道 3 事件的中断标志。	0
31:8	-	保留	-

13.7.2 定时器控制寄存器

定时器控制寄存器 (TCR) 用于控制计数器 / 定时器的操作。

表 222. 定时器控制寄存器 (TCR, 地址 0x4001 0004 (CT16B0) 和 0x4001 4004 (CT16B1)) 位描述

位	符号	值	描述	复位值
0	CEN		计数器使能。	0
		0	计数器被禁能。	
		1	定时器 / 计数器和预分频计数器使能计数。	
1	CRST		计数器复位。	0
		0	不执行任何操作。	
		1	定时器计数器和预分频计数器在 PCLK 的下一个上升沿同步复位。计数器在 TCR[1] 恢复为 0 之前保持复位状态。	
31:2	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	

13.7.3 定时器计数器

当预分频器计数器达到其最终计数时，16 位定时器计数器会递增计数。如果 TC 在到达计数器上限之前没有复位，它将一直计数到 0x0000 FFFF 然后翻转到 0x0000 0000。该事件不会产生中断，如果需要，可使用匹配寄存器检测溢出。

表 223. 定时器计数器寄存器 (TC, 地址 0x4001 0008 (CT16B0) 和 0x4001 4008 (CT16B1)) 位描述

位	符号	描述	复位值
15:0	TC	定时器计数器值。	0
31:16	-	保留。	-

13.7.4 预分频寄存器

16 位预分频寄存器指定了预分频计数器的最大计数值。

表 224. 预分频寄存器（PR，地址 0x4001 000C (CT16B0) 和 0x4001 400C (CT16B1)）位描述

位	符号	描述	复位值
15:0	PCVAL	预分频值。	0
31:16	-	保留。	-

13.7.5 预分频计数器寄存器

16 位预分频计数器用某个常量来控制 PCLK 的分频，再使其输入到定时器计数器。它所控制的是定时器分辨率与最大时间之间的关系，从而能防止定时器溢流。预分频计数器会在每个 PCLK 时钟上递增计数。当预分频计数器的计数达到预分频寄存器中存储的值时，定时器计数器将递增计数，并且在下一个 PCLK 时钟上进行预分频计数器复位。这将使得 TC 当 PR = 0 时在每个 PCLK 上递增计数，当 PR = 1 时，在每 2 个 PCLK 上递增计数，依次类推。

表 225. 预分频计数器寄存器（PC，地址 0x4001 0010 (CT16B0) 和 0x4001 4010 (CT16B1)）位描述

位	符号	描述	复位值
15:0	PC	预分频计数器值。	0
31:16	-	保留。	-

13.7.6 匹配控制寄存器

匹配控制寄存器用于控制当其中一个匹配寄存器的值与定时器计数器的值匹配时应执行的操作。匹配控制寄存器各位的功能如表 226 所示。

表 226. 匹配控制寄存器（MCR，地址 0x4001 0014 (CT16B0) 和 0x4001 4014 (CT16B1)）位描述

位	符号	值	描述	复位值
0	MR0I		MR0 上的中断：当 MR0 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
1	MR0R		MR0 上的复位：MR0 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
2	MR0S		MR0 上的停止：MR0 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
3	MR1I		MR1 上的中断：当 MR1 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
4	MR1R		MR1 上的复位：MR1 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
5	MR1S		MR1 上的停止：MR1 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	

表 226. 匹配控制寄存器 (MCR, 地址 0x4001 0014 (CT16B0) 和 0x4001 4014 (CT16B1)) 位描述 (续)

位	符号	值	描述	复位值
6	MR2I		MR2 上的中断：当 MR2 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
7	MR2R		MR2 上的复位：MR2 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
8	MR2S		MR2 上的停止：MR2 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
9	MR3I		MR3 上的中断：当 MR3 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
10	MR3R		MR3 上的复位：MR3 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
11	MR3S		MR3 上的停止：MR3 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
31:12	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

13.7.7 匹配寄存器

匹配寄存器值会不断地与定时器计数器值进行比较。当两个值相等时，自动触发相应操作。这些操作包括产生中断、复位定时器计数器或停止定时器。所有操作均由 MCR 寄存器中的设置控制。

表 227. 匹配寄存器 (MR0 到 3, 地址 0x4001 0018 到 24 (CT16B0) 以及 0x4001 4018 到 24 (CT16B1)) 位描述

位	符号	描述	复位值
15:0	MATCH	定时器计数器匹配值。	0
31:16	-	保留。	-

13.7.8 捕获控制寄存器

捕获控制寄存器用于控制当捕获事件发生时，是否将计数器 / 定时器中的值装入捕获寄存器，以及捕获事件是否产生中断。同时将上升沿位和下降沿位置位是有效配置，会使两个边沿都产生捕获事件。在下面描述中，“n”表示定时器编号，0 或 1。

在 LPC122x 中，捕获通道 2 和 3 连接到比较器的边沿和电平输出（见表 218）。在下面描述中，“n”还表示比较器编号，0 或 1。比较器 0 连接到 CT16B0，比较器 1 连接到 CT16B1。

表 228. 捕获控制寄存器（CCR，地址 0x4001 0028 (CT16B0) 和 0x4001 4028 (CT16B1)）位描述

位	符号	值 描述	复位值
0	CAP0RE	CT16Bn_CAP0 上升沿捕获：CT16Bn_CAP0 的从 0 至 1 的序列，将使 CR0 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
1	CAP0FE	CT16Bn_CAP0 下降沿捕获：CT16Bn_CAP0 上的从 1 至 0 的序列，将使 CR0 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
2	CAP0I	CT16Bn_CAP0 事件中断：CT16Bn_CAP0 事件所导致的 CR0 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
3	CAP1RE	CT16Bn_CAP1 上升沿捕获：CT16Bn_CAP1 上的从 0 至 1 的序列，将使 CR1 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
4	CAP1FE	CT16Bn_CAP1 下降沿捕获：CT16Bn_CAP1 的从 1 至 0 的序列，将使 CR1 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
5	CAP1I	CT16Bn_CAP1 事件中断：CT16Bn_CAP1 事件所导致的 CR1 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
6	CAP2RE	比较器 n 电平输出 - 上升沿捕获：比较器 n 输出上的从 0 至 1 的序列，将使 CR2 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
7	CAP2FE	比较器 n 电平输出 - 下降沿捕获：比较器 n 输出上的从 1 至 0 的序列，将使 CR2 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
8	CAP2I	比较器 n 电平输出事件中断：比较器 0 事件所导致的 CR2 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
9	CAP3RE	比较器 n 边沿输出 - 上升沿捕获：比较器 n 输出上的从 0 至 1 的序列，将使 CR3 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
10	CAP3FE	比较器 n 边沿输出 - 下降沿捕获：比较器 n 输出上的从 1 至 0 的序列，将使 CR3 载入 TC 内容。	0
		1 使能。	
		0 禁能。	
11	CAP3I	比较器 n 边沿输出事件中断：比较器 n 事件所导致的 CR3 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
31:12	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

13.7.9 捕获寄存器

各捕获寄存器与器件引脚相关联，当引脚发生特定的事件时，可将计数器 / 定时器的值装入该捕获寄存器。捕获控制寄存器中的设置决定是否使能捕获功能，及在相关引脚的上升沿、下降沿或上升沿和下降沿上是否产生捕获事件。

表 229. 捕获寄存器（CR0 到 3，地址 0x4001 002C 到 38 (CT16B0) 以及 0x4001 402C 到 38 (CT16B1)）位描述

位	符号	描述	复位值
15:0	CAP	定时器计数器捕获值。	0
31:16	-	保留。	-

13.7.10 外部匹配寄存器

外部匹配寄存器为外部匹配引脚 CT16Bn_MAT[1:0] 提供控制和状态。

每个定时器中的匹配 0 和匹配 1 的匹配事件会产生 DMA 请求。

如果匹配输出被配置为 PWM 输出，则外部匹配寄存器的功能由 PWM 规则决定（[235 页上的章节 13.7.13 “单边沿控制的 PWM 输出规则”](#)）。

表 230. 外部匹配寄存器（EMR，地址 0x4001 003C (CT16B0) 和 0x4001 403C (CT16B1)）位描述

位	符号	值	描述	复位值
0	EM0		外部匹配 0。该位反映输出 CT16B0_MAT0/CT16B1_MAT0 的状态，不管该输出是否连接到此引脚。当 TC 与 MR0 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[5:4] 控制该输出的功能。如果选用了 IOCOM 寄存器的匹配功能（0= 低电平，1= 高电平），该位就会被驱动到 CT16B0_MAT0/CT16B1_MAT0 引脚上。	0
1	EM1		外部匹配 1。该位反映输出 CT16B0_MAT1/CT16B1_MAT1 的状态，不管该输出是否连接到此引脚。当 TC 与 MR1 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[7:6] 控制该输出的功能。如果选用了 IOCOM 寄存器的匹配功能（0= 低电平，1= 高电平），该位就会被驱动到 CT16B0_MAT0/CT16B1_MAT0 引脚上。	0
2	EM2		外部匹配 2。该位反映匹配通道 2 的状态。当 TC 与 MR2 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[9:8] 控制该输出的功能。	0
3	EM3		外部匹配 3。该位反映匹配通道 3 的输出的状态。当 TC 与 MR3 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[11:10] 控制该输出的功能。	0
5:4	EMC0		外部匹配控制 0。决定外部匹配 0 的功能。这些位的编码如 表 231 所示。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT16Bi_MAT0 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT16Bi_MAT0 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
7:6	EMC1		外部匹配控制 1。决定外部匹配 1 的功能。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT16Bi_MAT1 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT16Bi_MAT1 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	

表 230. 外部匹配寄存器（EMR，地址 0x4001 003C (CT16B0) 和 0x4001 403C (CT16B1)）位描述

位	符号	值	描述	复位值
9:8	EMC2		外部匹配控制 2。决定外部匹配 2 的功能。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT16Bi_MAT2 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT16Bi_MAT2 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
11: 10	EMC3		外部匹配控制 3。决定外部匹配 3 的功能。这些位的编码如表 231 所示。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT16Bi_MAT3 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT16Bi_MAT3 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
31: 12	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

表 231. 外部匹配控制

EMR[11:10]、EMR[9:8]、EMR[7:6] 或 EMR[5:4]	功能
00	不执行任何操作。
01	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT16Bn_MATm 引脚为低电平）。
10	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT16Bn_MATm 引脚为高电平）。
11	切换相应的外部匹配位 / 输出。

13.7.10.1 DMA 操作

每个定时器的外部匹配 0 位在 0 到 1 的跃迁过程中，会生成 DMA 请求。如果要产生某种效果，则必须配置 GPDMA，并且必须将相关的定时器 DMA 请求选为 DMA 源。当定时器最初设置为生成 DMA 请求时，可能会在匹配条件产生之前使该请求变为有效。可通过让软件写“1”以清除定时器中断标志来避免第一个 DMA 请求。DMA 请求将在处理后由 DMA 控制器通过硬件自动清除。

如果通道 0 的 EMR 位被置 10 或 11（上升沿或切换），即使相应 MR 寄存器被置 0，仍将产生 DMA 请求，因为存在“零匹配”条件。要禁用任意 DMA 请求，请将通道 0 的 EMR 位置 00。

13.7.11 计数控制寄存器

计数控制寄存器 (CTCR) 用于在定时器模式和计数器模式之间进行选择，且在处于计数器模式时选择进行计数的引脚和边沿。

如果将计数器模式选为操作模式，则在 PCLK 时钟的每个上升沿上会对（CTCR 位 3:2 所选的）CAP 输入进行采样。在比较该 CAP 输入的两个连续样本后，将会确认下列四个事件之一：所选 CAP 输入电平处于上升沿、下降沿，或上升下降沿或无变化。仅当所标识的事件与 CTCR 寄存器中的位 1:0 所选的内容相匹配时，定时器计数器寄存器才会递增计数。

要有效地处理计数器的外部源时钟会有一些限制。由于 PCLK 时钟的两个连续上升沿仅用于标识 CAP 所选输入的一个边缘，因此 CAP 输入的频率不能超过 PCLK 时钟的一半。因此在这种情况下，相同的 CAP 输入上的高电平 / 低电平的持续时间不能少于 1/PCLK。

该寄存器的位 7:4 还用于使能和配置捕获 - 清除 - 定时器特性。该特性允许特定 CAP 输入的指定边沿将定时器全部清零。使用该机制在输入脉冲前沿清除定时器然后在后沿执行捕获，可以使用单捕获输入来直接测量脉冲宽度，而无需在软件中执行减法操作。

表 232. 计数控制寄存器 (CTCR, 地址 0x4001 0070 (CT16B0) 和 0x4001 4070 (CT16B1)) 位描述

位	符号	值	描述	复位值
1:0	CTM		计数器 / 定时器模式。该字段用于选择可使用哪个 PCLK 上升沿，使定时器预分频计数器 (PC) 递增计数，或清除 PC 并使定时器计数器 (TC) 递增计数。	00
		0x0	定时器模式：每个 PCLK 上升沿	
		0x1	计数器模式：TC 在位 3:2 选择的 CAP 输入的上升沿时递增。	
		0x2	计数器模式：TC 在位 3:2 选择的 CAP 输入的下降沿时递增。	
		0x3	计数器模式：TC 在位 3:2 选择的 CAP 输入的两个边沿递增。	
3:2	CIS		计数输入选择。在计数器模式下（当该寄存器中位 1:0 不为 00 时），这两位选择哪个 CAP 引脚或比较器输出被采样用于计时：	00
		0x0	CT16Bn_CAP0	
		0x1	CT16Bn_CAP1	
		0x2	比较器 n，电平输出	
		0x3	比较器 n，边沿输出	
4	ENCC		将此位置 1 可在发生位 7:5 指定的捕获 - 边沿事件时清零定时器和预分频器。	0
7:5	SELCC		当位 4 为 1 时，这两位选择哪个捕获输入边沿将导致定时器和预分频器被清零。当位 4 为低电平时，这两位无效。	
		0x0	CAP0 的上升沿清零定时器（如果位 4 被置位）	
		0x1	CAP0 的下降沿清零定时器（如果位 4 被置位）	
		0x2	CAP1 的上升沿清零定时器（如果位 4 被置位）	
		0x3	CAP1 的下降沿清零定时器（如果位 4 被置位）	
		0x4	CAP2 的上升沿清零定时器（如果位 4 被置位）	
		0x5	CAP2 的下降沿清零定时器（如果位 4 被置位）	
		0x6	CAP3 的上升沿清零定时器（如果位 4 被置位）	
		0x7	CAP3 的下降沿清零定时器（如果位 4 被置位）	
31:8	-	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	-

13.7.12 PWM 控制寄存器

PWM 控制寄存器用于将匹配输出配置为 PWM 输出。每个匹配输出均可分别设置，以决定匹配输出是作为 PWM 输出还是作为功能受外部匹配寄存器 (EMR) 控制的匹配输出。

对于定时器，CT16Bn_MAT[1:0] 输出最多可选择 3 个单边沿控制的 PWM 输出。一个附加的匹配寄存器决定 PWM 的周期长度。当任何其他匹配寄存器出现匹配时，PWM 输出置为高电平。用于设置 PWM 周期长度的匹配寄存器负责将定时器复位。当定时器复位到 0 时，所有当前配置为 PWM 输出的高电平匹配输出清零。

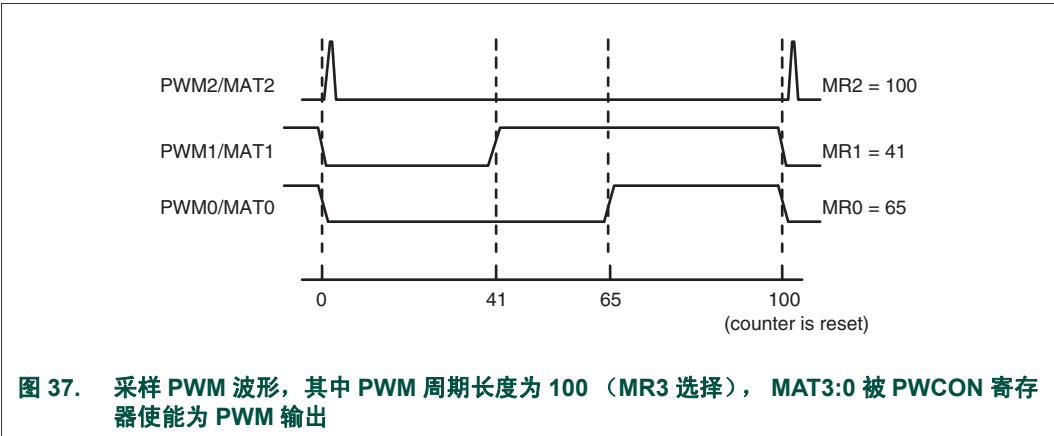
表 233. PWM 控制寄存器（PWMC，地址 0x4001 0074 和 0x4001 4074 (CT16B1)）位描述

位	符号	值	描述	复位值
0	PWMEN0		通道 0 的 PWM 模式使能。	0
		0	CT16Bi_MAT0 受 EM0 控制。	
		1	CT16Bi_MAT0 的 PWM 模式使能。	
1	PWMEN1		通道 1 的 PWM 模式使能。	0
		0	CT16Bi_MAT01 受 EM1 控制。	
		1	CT16Bi_MAT1 的 PWM 模式使能。	
2	PWMEN2		通道 2 的 PWM 模式使能。	0
		0	CT16Bi_MAT2 受 EM2 控制。	
		1	CT16Bi_MAT2 的 PWM 模式使能。	
3	PWMEN3		通道 3 的 PWM 模式使能。	0
		0	CT16Bi_MAT3 受 EM3 控制。	
		1	CT16Bi_MAT3 的 PWM 模式使能。	
31:4	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	

13.7.13 单边沿控制的 PWM 输出规则

- 1. 所有单边沿控制的 PWM 输出在 PWM 周期开始时都为低电平（定时器置为 0），除非它们的匹配值等于 0。
- 2. 每个 PWM 输出在达到其匹配值时将转入高电平。如果没有发生匹配（即匹配值大于 PWM 周期长度），则 PWM 输出将继续保持低电平。
- 3. 如果将大于 PWM 周期长度的匹配值写入到匹配寄存器，且 PWM 信号已经为高电平，则在下一个 PWM 周期开始时 PWM 信号将被清零。
- 4. 如果匹配寄存器中包含与定时器复位值（PWM 周期长度）相同的值，则在下一个时钟节拍时 PWM 输出将复位到低电平。因此，PWM 输出总是包含一个时钟节拍宽度的正脉冲，周期由 PWM 周期长度决定（即定时器重载入值）。
- 5. 如果匹配寄存器置 0，则 PWM 输出将在定时器第一次返回 0 时变为高电平，并继续保持高电平。

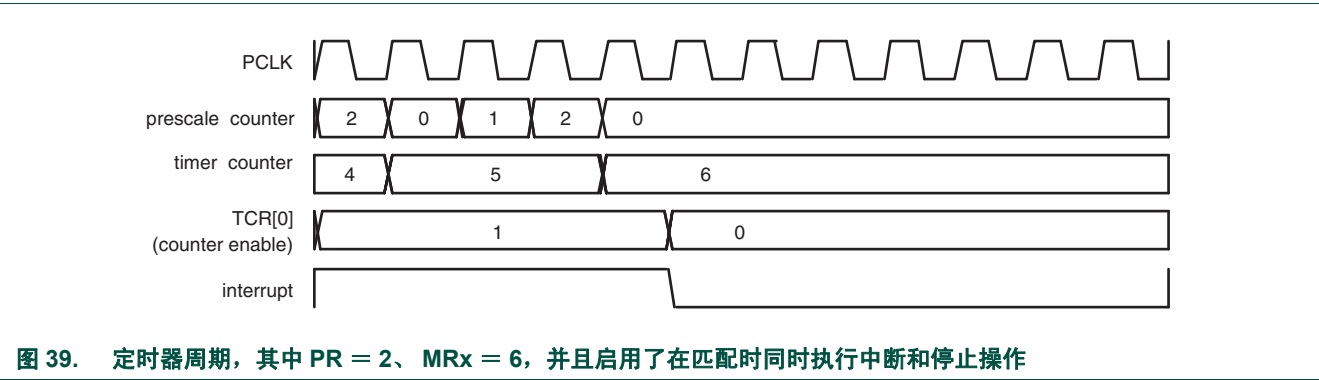
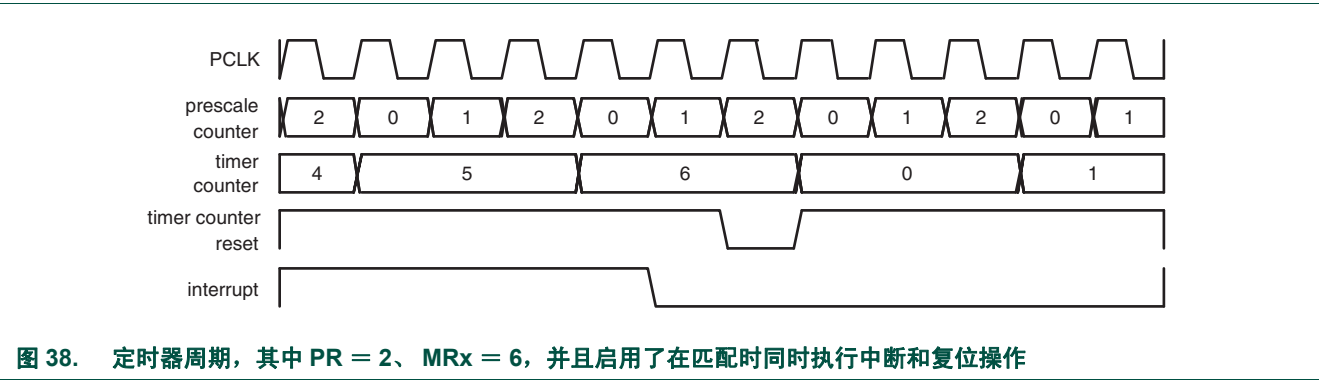
注：当选择匹配输出作为 PWM 输出来执行时，除匹配寄存器设置 PWM 周期长度外，匹配控制寄存器 MCR 中的定时器复位 (MRnR) 和定时器停止 (MRnS) 位必须置为 0。对于该寄存器，当定时器值与相应的匹配寄存器值匹配时，将 MRnR 位置 1 以使能定时器复位。



13.8 定时器操作示例

如图 38 所示，定时器配置为在匹配时复位计数并产生中断。预分频器置为 2，匹配寄存器置为 6。在发生匹配的定时器周期结束时，定时器计数复位。这样就使匹配值具有完整长度的周期。在定时器到达匹配值后的下一个时钟产生指示匹配发生的中断。

如图 39 所示，定时器配置为在匹配时停止计数并产生中断。预分频器再次置为 2，匹配寄存器置为 6。在定时器达到匹配值的下一个时钟，TCR 中的定时器使能位被清零，产生指示匹配发生的中断。



13.9 架构

计数器 / 定时器 0 和计数器 / 定时器 1 的功能框图如图 40 所示。

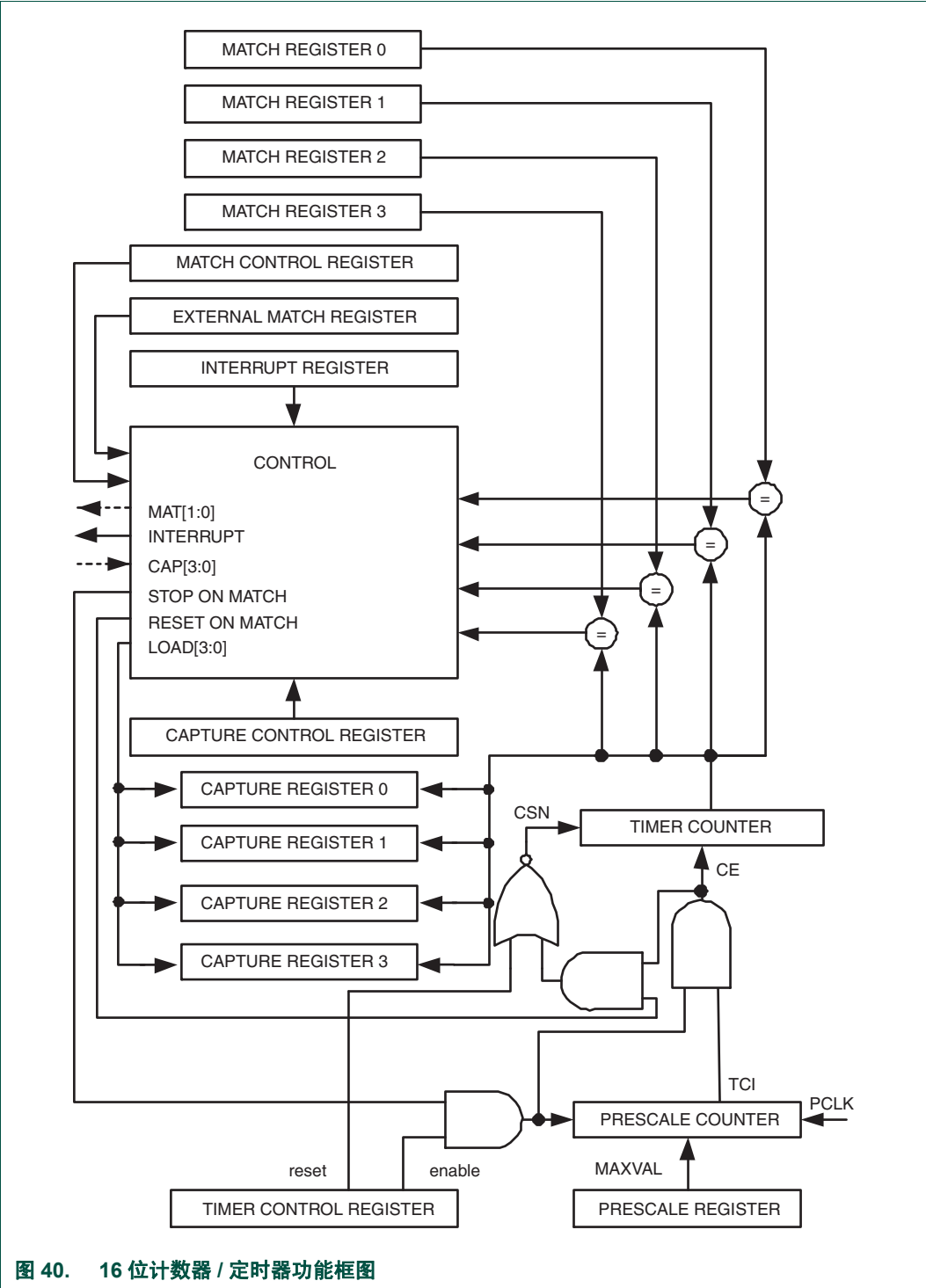


图 40. 16 位计数器 / 定时器功能框图

14.1 本章导读

所有 LPC122x 部件都提供 CT32B0/1。

14.2 基本配置

32 位定时器计数器模块的外设时钟由系统时钟提供，系统时钟由 SYSAHBCLKDIV 寄存器控制（[表 20](#)）。而 CT32B0/1 模块可以通过系统 AHB 时钟控制寄存器位 9 和 10（[表 21](#)）禁能以降低功耗。

14.3 特性

- 两个带有可编程 32 位预分频器的 32 位计数器 / 定时器。
- 计数器或定时器操作。
- 四个 32 位捕获通道，可在输入信号跃迁时快速捕获定时器值。捕获事件也可能产生一个中断。
- 可配置定时器和预分频器在指定捕获事件清零。此特性通过在输入脉冲前沿清零定时器并捕获定时器在后沿的值，方便进行脉冲宽度测量。
- 四个 32 位匹配寄存器允许：
 - 连续操作，可选择在匹配时产生中断。
 - 在匹配时停止定时器，可选择产生中断。
 - 在匹配时复位定时器，可选择产生中断。
- 四个对应于匹配寄存器的外部输出，具备以下功能：
 - 匹配时设置低电平。
 - 匹配时设置高电平。
 - 匹配时切换。
 - 匹配时不执行任何操作。
- 对于各定时器，最多 4 个匹配寄存器可配置为 PWM，允许使用最多 3 个匹配输出作为单独边沿控制的 PWM 输出。
- 最多有两个匹配寄存器可用来产生定时 DMA 请求。

14.4 应用程序

- 时间间隔定时器，用于对内部事件进行计数
- 脉冲宽度解调器（经捕获输入）
- 自由运行的定时器
- 脉冲宽度调制器（经匹配输出）

14.5 简介

每个计数器 / 定时器都设计用来计算外设时钟 (PCLK) 或外部供电时钟的周期，并且可根据 4 个匹配寄存器的值在指定定时器值处产生中断或执行其他操作。每个计数器 / 定时器还包括 1 个捕获输入，用来在输入信号跃迁时捕获定时器值，同时可根据需要产生一个中断。

在 PWM 模式下，可使用其中 3 个匹配寄存器向匹配输出引脚提供单边沿控制的 PWM 输出。而剩下的那个匹配寄存器则用于控制 PWM 周期长度。

注：32 位计数器 / 定时器 0 (CT32B0) 和 32 位计数器 / 定时器 1 (CT32B1) 除外设基址不同外，其他功能相似。

14.6 引脚描述

表 234 简要总结了每个计数器 / 定时器相关的引脚。

表 234. 计数器 / 定时器引脚描述

引脚	类型	描述
CT32B0_CAP[3:0] CT32B1_CAP[3:0]	输入	捕获信号： 可以配置捕获引脚上的跃迁，用定时器计数器中的值载入其中一个捕获寄存器，并且可以有选择性地产生一个中断。 计数器 / 定时器模块可选择捕获信号作为时钟源来代替 PCLK 衍生时钟。有关更多详情，请参阅 247 页上的章节 14.7.11 “计数控制寄存器”。
CT32B0_MAT[3:0] CT32B1_MAT[3:0]	输出	CT32B0/1 的外部匹配输出： 当匹配寄存器 MR3:0 的值与定时器计数器值 (TC) 相等时，该输出可以进行切换、转入低电平、转入高电平或不执行任何操作。外部匹配寄存器 (EMR) 和 PWM 控制寄存器 (PWMCON) 控制该输出的功能。

14.7 寄存器描述

32 位计数器 / 定时器 0 包含的寄存器如表 235 所示，32 位计数器 / 定时器 1 包含的寄存器如表 236 所示。详细描述如下。

表 235. 寄存器简介：32 位计数器 / 定时器 0 CT32B0（基址 0x4001 8000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
IR	R/W	0x000	中断寄存器 (IR)。可向 IR 写入相应值来清除中断。可以通过读 IR 来识别 8 个 0 中断源中哪个中断源正在被挂起。	0
TCR	R/W	0x004	定时器控制寄存器 (TCR)。TCR 用于控制定时器计数器功能。定时器计数器可通过 TCR 来禁能或复位。	0
TC	R/W	0x008	定时器计数器 (TC)。32 位 TC 每隔 PR + 1 个 PCLK 周期递增一次。通过 TCR 控制 TC。	0
PR	R/W	0x00C	预分频寄存器 (PR)。当预分频计数器（见下）与该值相等时，下个时钟 TC 加 1，PC 清零。	0
PC	R/W	0x010	预分频计数器 (PC)。32 位 PC 是一个计数器，它会增加到与 PR 中存放的值相等。当达到 PR 的值时，TC 递增，PC 清零。可通过总线接口来观察和控制 PC。	0
MCR	R/W	0x014	匹配控制寄存器 (MCR)。MCR 用于控制在匹配出现时是否产生中断及出现匹配时 TC 是否复位。	0
MR0	R/W	0x018	匹配寄存器 0(MR0)。MR0 可通过 MCR 使能，当 MR0 与 TC 匹配时复位 TC，停止 TC 和 PC，和 / 或产生中断。	0

表 235. 寄存器简介：32 位计数器 / 定时器 0 CT32B0（基址 0x4001 8000）（续）

名称	访问类型	地址偏移	描述	复位值 ^[1]
MR1	R/W	0x01C	匹配寄存器 1(MR1)。见 MR0 描述。	0
MR2	R/W	0x020	匹配寄存器 2(MR2)。见 MR0 描述。	0
MR3	R/W	0x024	匹配寄存器 3(MR3)。见 MR0 描述。	0
CCR	R/W	0x028	捕获控制寄存器 (CCR)。CCR 控制捕获时使用捕获输入的哪个边沿来加载捕获寄存器，以及在捕获时是否产生中断。	0
CR0	RO	0x02C	捕获寄存器 0(CR0)。当 CT32B0_CAP0 输入上产生事件时，CR0 载入 TC 值。	0
CR1	RO	0x030	捕获寄存器 1(CR1)。当 CT32B0_CAP1 输入上产生事件时，CR1 载入 TC 值。	0
CR2	RO	0x034	捕获寄存器 2(CR2)。当 CT32B0_CAP2 输入上产生事件时，CR2 载入 TC 值。	0
CR3	RO	0x038	捕获寄存器 3(CR3)。当 CT32B3_CAP3 输入上产生事件时，CR3 载入 TC 值。	0
EMR	R/W	0x03C	外部匹配寄存器(EMR)。EMR控制匹配功能及外部匹配引脚CT32Bn_MAT[3:0]。	0
-	-	0x040 - 0x06C	保留	-
CTCR	R/W	0x070	计数控制寄存器 (CTCR)。CTCR 选择在定时器模式还是在计数器模式下工作，在计数器模式下选择计数的信号和边沿。	0
PWMC	R/W	0x074	PWM 控制寄存器 (PWMCON)。PWMCON 使能 PWM 模式，用于外部匹配引脚 CT32Bn_MAT[3:0]。	0

[1] 复位值只反映了使用位的值。不包括保留位的内容。

表 236. 寄存器简介：32 位计数器 / 定时器 1 CT32B1（基址 0x4001 C000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
IR	R/W	0x000	中断寄存器 (IR)。可向 IR 写入相应值来清除中断。可以通过读 IR 来识别 8 个中断源中哪个中断源正在被挂起。	0
TCR	R/W	0x004	定时器控制寄存器 (TCR)。TCR 用于控制定时器计数器功能。定时器计数器可通过 TCR 来禁能或复位。	0
TC	R/W	0x008	定时器计数器 (TC)。32 位 TC 每隔 PR + 1 个 PCLK 周期递增一次。通过 TCR 控制 TC。	0
PR	R/W	0x00C	预分频寄存器 (PR)。当预分频计数器（见下）与该值相等时，下个时钟 TC 加 1，PC 清零。	0
PC	R/W	0x010	预分频计数器 (PC)。32 位 PC 是一个计数器，它会增加到与 PR 中存放的值相等。当达到 PR 的值时，TC 递增，PC 清零。可通过总线接口来观察和控制 PC。	0
MCR	R/W	0x014	匹配控制寄存器 (MCR)。MCR 用于控制在匹配出现时是否产生中断及出现匹配时 TC 是否复位。	0
MR0	R/W	0x018	匹配寄存器 0(MR0)。MR0 可通过 MCR 使能，当 MR0 与 TC 匹配时复位 TC，停止 TC 和 PC，和 / 或产生中断。	0
MR1	R/W	0x01C	匹配寄存器 1(MR1)。见 MR0 描述。	0

表 236. 寄存器简介：32 位计数器 / 定时器 1 CT32B1（基址 0x4001 C000）（续）

名称	访问类型	地址偏移	描述	复位值 ^[1]
MR2	R/W	0x020	匹配寄存器 2(MR2)。见 MR0 描述。	0
MR3	R/W	0x024	匹配寄存器 3(MR3)。见 MR0 描述。	0
CCR	R/W	0x028	捕获控制寄存器 (CCR)。CCR 控制捕获时使用捕获输入的哪个边沿来加载捕获寄存器，以及在捕获时是否产生中断。	0
CR0	RO	0x02C	捕获寄存器 0(CR0)。当 CT32B1_CAP0 输入上产生事件时，CR0 载入 TC 值。	0
CR1	RO	0x030	捕获寄存器 1(CR1)。当 CT32B1_CAP1 输入上产生事件时，CR1 载入 TC 值。	0
CR2	RO	0x034	捕获寄存器 2(CR2)。当 CT32B1_CAP2 输入上产生事件时，CR2 载入 TC 值。	0
CR3	RO	0x038	捕获寄存器 3(CR3)。当 CT32B1_CAP3 输入上产生事件时，CR3 载入 TC 值。	0
EMR	R/W	0x03C	外部匹配寄存器 (EMR)。EMR 控制匹配功能及外部匹配引脚 CT32Bn_MAT[3:0]。	0
-	-	0x040 - 0x06C	保留	-
CTCR	R/W	0x070	计数控制寄存器 (CTCR)。CTCR 选择在定时器模式还是在计数器模式下工作，在计数器模式下选择计数的信号和边沿。	0
PWMC	R/W	0x074	PWM 控制寄存器 (PWMCON)。PWMCON 使能 PWM 模式，用于外部匹配引脚 CT32Bn_MAT[3:0]。	0

[1] 复位值只反映了使用位的值。不包括保留位的内容。

14.7.1 中断寄存器

中断寄存器包含 4 个用于匹配中断的位及 4 个用于捕获中断的位。如果有中断产生，IR 中的相应位为高电平。否则，该位为低电平。向对应的 IR 位写逻辑 1 会使中断复位。写 0 无效。清除定时器匹配中断会同时清除任何相应的 DMA 请求。

表 237. 中断寄存器（IR，地址 0x4001 8000 (CT32B0)；和 IR，地址 0x4001 C000）位描述

位	符号	描述	复位值
0	MR0INT	匹配通道 0 的中断标志。	0
1	MR1INT	匹配通道 1 的中断标志。	0
2	MR2INT	匹配通道 2 的中断标志。	0
3	MR3INT	匹配通道 3 的中断标志。	0
4	CR0INT	捕获通道 0 事件的中断标志。	0
5	CR1INT	捕获通道 1 事件的中断标志。	0
6	CR2INT	捕获通道 2 事件的中断标志。	0
7	CR3INT	捕获通道 3 事件的中断标志。	0
31:8	-	保留	-

14.7.2 定时器控制寄存器

定时器控制寄存器 (TCR) 用于控制计数器 / 定时器的操作。

表 238. 定时器控制寄存器 (TCR, 地址 0x4001 8004 (CT32B0) 和 0x4001 C004 (CT32B1)) 位描述

位	符号	值	描述	复位值
0	CEN		计数器使能。	0
		0	计数器被禁能。	
		1	定时器 / 计数器和预分频计数器使能计数。	
1	CRST		计数器复位。	0
		0	不执行任何操作。	
		1	定时器计数器和预分频计数器在 PCLK 的下一个上升沿同步复位。计数器在 TCR[1] 恢复为 0 之前保持复位状态。	
31:2	-		保留, 用户软件不应向保留位写入 1。从保留位读取的值 未定义。	

14.7.3 定时器计数器寄存器

当预分频器计数器达到计数上限时, 32 位定时器计数器加 1。如果 TC 在到达计数器上限之前没有复位, 它将一直计数到 0xFFFF FFFF 然后翻转到 0x0000 0000。该事件不会产生中断, 如果需要, 可使用匹配寄存器检测溢出。

表 239. 定时器计数器寄存器 (TC, 地址 0x4001 8008 (CT32B0) 和 0x4001 C008 (CT32B1)) 位描述

位	符号	描述	复位值
31:0	TC	定时器计数器值。	0

14.7.4 预分频寄存器

32 位预分频寄存器指定了预分频计数器的最大计数值。

Table 240: 预分频寄存器 (PR, 地址 0x4001 800C (CT32B0) 和 0x4001 C00C (CT32B1)) 位描述

位	符号	描述	复位值
31:0	PCVAL	预分频器值。	0

14.7.5 预分频计数器寄存器

32 位预分频计数器将在 PCLK 应用于定时器计数器之前, 使用某一常数值对 PCLK 执行分频控制。它所控制的是定时器分辨率与最大时间之间的关系, 从而能防止定时器溢流。预分频计数器会在每个 PCLK 时钟上递增计数。当预分频计数器的计数达到预分频寄存器中存储的值时, 定时器计数器将递增计数, 并且在下一个 PCLK 时钟上进行预分频计数器复位。这将使得 TC 当 PR = 0 时在每个 PCLK 上递增计数, 当 PR = 1 时, 在每 2 个 PCLK 上递增计数, 依次类推。

表 241. 预分频寄存器 (PC, 地址 0x4001 8010 (CT32B0) 和 0x4001 C010 (CT32B1)) 位描述

位	符号	描述	复位值
31:0	PC	预分频计数器值。	0

14.7.6 匹配控制寄存器

匹配控制寄存器用于控制当其中一个匹配寄存器的值与定时器计数器的值匹配时应执行的操作。匹配控制寄存器各位的功能如表 242 所示。

表 242. 匹配控制寄存器（MCR，地址 0x4001 8014 (CT32B0) 和 0x4001 C014 (CT32B1)）位描述

位	符号	值	描述	复位值
0	MR0I		MR0 上的中断：当 MR0 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
1	MR0R		MR0 上的复位：MR0 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
2	MR0S		MR0 上的停止：MR0 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
3	MR1I		MR1 上的中断：当 MR1 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
4	MR1R		MR1 上的复位：MR1 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
5	MR1S		MR1 上的停止：MR1 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
6	MR2I		MR2 上的中断：当 MR2 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
7	MR2R		MR2 上的复位：MR2 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
8	MR2S		MR2 上的停止：MR2 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
9	MR3I		MR3 上的中断：当 MR3 与 TC 值匹配时产生中断。	0
		1	使能	
		0	禁能	
10	MR3R		MR3 上的复位：MR3 与 TC 匹配将使 TC 复位。	0
		1	使能	
		0	禁能	
11	MR3S		MR3 上的停止：MR3 与 TC 匹配时将使 TC 和 PC 停止，TCR[0] 置 0。	0
		1	使能	
		0	禁能	
31:12	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

14.7.7 匹配寄存器

匹配寄存器值会不断地与定时器计数器值进行比较。当两个值相等时，自动触发相应操作。这些操作包括产生中断、复位定时器计数器或停止定时器。所有操作均由 MCR 寄存器中的设置控制。

表 243. 匹配寄存器（MR0 到 3，地址 0x4001 8018 到 24 (CT32B0) 以及 0x4001 C018 到 24 (CT32B1)）位描述

位	符号	描述	复位值
31:0	MATCH	定时器计数器匹配值。	0

14.7.8 捕获控制寄存器（CCR 和 CCR）

捕获控制寄存器用于控制当捕获事件发生时，是否将定时器计数器中的值装入 4 个捕获寄存器中的一个，以及捕获事件是否产生中断。同时将上升沿位和下降沿位置位是有效配置，会使两个边沿都产生捕获事件。在下面描述中，“n”表示定时器编号，0 或 1。

表 244. 捕获控制寄存器（CCR，地址 0x4001 8028 (CT32B0) 和 0x4001 C028 (CT32B1)）位描述

位	符号	值 描述	复位值
0	CAP0RE	1 CT32Bn_CAP0 上升沿捕获：CT32Bn_CAP0 上“0”到“1”的跳变将使 TC 的内容装入 CR0。	0
		1 使能。	
		0 禁能。	
1	CAP0FE	1 CT32Bn_CAP0 下降沿捕获：CT32Bn_CAP0 上“1”到“0”的跳变将使 TC 的内容装入 CR0。	0
		1 使能。	
		0 禁能。	
2	CAP0I	CT32Bn_CAP0 事件中断：CT32Bn_CAP0 事件所导致的 CR0 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
3	CAP1RE	CT32Bn_CAP1 上升沿捕获：CT32Bn_CAP1 上“0”到“1”的跳变将使 TC 的内容装入 CR1。	0
		1 使能。	
		0 禁能。	
4	CAP1FE	CT32Bn_CAP1 下降沿捕获：CT32Bn_CAP1 上“1”到“0”的跳变将使 TC 的内容装入 CR1。	0
		1 使能。	
		0 禁能。	
5	CAP1I	CT32Bn_CAP1 事件中断：CT32Bn_CAP1 事件所导致的 CR1 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
6	CAP2RE	CT32Bn_CAP2 上升沿捕获：CT32Bn_CAP2 上“0”到“1”的跳变将使 TC 的内容装入 CR2。	0
		1 使能。	
		0 禁能。	
7	CAP2FE	CT32Bn_CAP2 下降沿捕获：CT32Bn_CAP2 上“1”到“0”的跳变将使 TC 的内容装入 CR2。	0
		1 使能。	
		0 禁能。	
8	CAP2I	CT32Bn_CAP2 事件中断：CT32Bn_CAP2 事件所导致的 CR2 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	

表 244. 捕获控制寄存器（CCR，地址 0x4001 8028 (CT32B0) 和 0x4001 C028 (CT32B1)）位描述 *（续）*

位	符号	值 描述	复位值
9	CAP3RE	CT32Bn_CAP3 上升沿捕获：CT32Bn_CAP3 上“0”到“1”的跳变将使 TC 的内容装入 CR3。	0
		1 使能。	
		0 禁能。	
10	CAP3FE	CT32Bn_CAP3 下降沿捕获：CT32Bn_CAP3 上“1”到“0”的跳变将使 TC 的内容装入 CR3。	0
		1 使能。	
		0 禁能。	
11	CAP3I	CT32Bn_CAP3 事件中断：CT32Bn_CAP3 事件所导致的 CR3 装载将产生一个中断。	0
		1 使能。	
		0 禁能。	
31:12	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

14.7.9 捕获寄存器

各捕获寄存器与器件引脚相关联，当引脚发生特定的事件时，可将定时器计数器的值装入该捕获寄存器。捕获控制寄存器中的设置决定是否使能捕获功能，及在相关引脚的上升沿、下降沿或上升沿和下降沿上是否产生捕获事件。

表 245. 捕获寄存器（CR0 到 3，地址 0x4001 802C 到 38 (CT32B0) 以及 0x4001 C02C 到 38 (CT32B1)）位描述

位	符号	描述	复位值
31:0	CAP	定时器计数器捕获值。	0

14.7.10 外部匹配寄存器

外部匹配寄存器为外部匹配引脚 CAP32Bn_MAT[3:0] 提供控制和状态。

每个定时器中的匹配 0 和匹配 1 的匹配事件会产生 DMA 请求。

如果匹配输出被配置为 PWM 输出，则外部匹配寄存器的功能由 PWM 规则决定（[249 页上的章节 14.7.13 “单边沿控制的 PWM 输出规则”](#)）。

表 246. 外部匹配寄存器（EMR，地址 0x4001 803C (CT32B0) 和 0x4001 C03C (CT32B1)）位描述

位	符号	值 描述	复位值
0	EM0	外部匹配 0。该位反映输出 CT32Bn_MAT0 的状态，不管该输出是否连接到此引脚。当 TC 与 MR0 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[5:4] 控制该输出的功能。如果选用了 IOCOM 寄存器的匹配功能（0= 低电平，1= 高电平），该位就会被驱动到 CT32B0_MAT0/CT32B1_MAT0 引脚上。	0
1	EM1	外部匹配 1。该位反映输出 CT32Bn_MAT1 的状态，不管该输出是否连接到此引脚。当 TC 与 MR1 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[7:6] 控制该输出的功能。如果选用了 IOCOM 寄存器的匹配功能（0= 低电平，1= 高电平），该位就会被驱动到 CT32B0_MAT1/CT32B1_MAT1 引脚上。	0
2	EM2	外部匹配 2。该位反映输出 CT32Bn_MAT2 的状态，不管该输出是否连接到此引脚。当 TC 与 MR2 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[9:8] 控制该输出的功能。如果选用了 IOCOM 寄存器的匹配功能（0= 低电平，1= 高电平），该位就会被驱动到 CT32B0_MAT2/CT32B1_MAT2 引脚上。	0
3	EM3	外部匹配 3。该位反映输出 CT32Bn_MAT3 的状态，不管该输出是否连接到此引脚。当 TC 与 MR3 相匹配时，该位可以进行切换、转入低电平、转入高电平或不执行任何操作。位 EMR[11:10] 控制该输出的功能。如果选用了 IOCOM 寄存器的匹配功能（0= 低电平，1= 高电平），该位就会被驱动到 CT32B3_MAT0/CT32B1_MAT3 引脚上。	0

表 246. 外部匹配寄存器（EMR，地址 0x4001 803C (CT32B0) 和 0x4001 C03C (CT32B1)）位描述（续）

位	符号	值	描述	复位值
5:4	EMC0		外部匹配控制 0。决定外部匹配 0 的功能。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT32Bi_MAT0 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT32Bi_MAT0 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
7:6	EMC1		外部匹配控制 1。决定外部匹配 1 的功能。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT32Bi_MAT1 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT32Bi_MAT1 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
9:8	EMC2		外部匹配控制 2。决定外部匹配 2 的功能。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT32Bi_MAT2 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT32Bi_MAT2 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
11:10	EMC3		外部匹配控制 3。决定外部匹配 3 的功能。	00
		0x0	不执行任何操作。	
		0x1	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT32Bi_MAT3 引脚为低电平）。	
		0x2	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT32Bi_MAT3 引脚为高电平）。	
		0x3	切换相应的外部匹配位 / 输出。	
31:12	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

表 247. 外部匹配控制

EMR[11:10]、EMR[9:8]、EMR[7:6] 或 EMR[5:4]	功能
00	不执行任何操作。
01	将相应的外部匹配位 / 输出清零（如果引脚处于输出状态，则 CT32Bn_MATm 引脚为低电平）。
10	将相应的外部匹配位 / 输出置 1（如果引脚处于输出状态，则 CT32Bn_MATm 引脚为高电平）。
11	切换相应的外部匹配位 / 输出。

14.7.10.1 DMA 操作

每个定时器的外部匹配 0 位和 1 位在 0 到 1 的跃迁过程中，会生成 DMA 请求。如果要产生某种效果，则必须配置 GPDMA，并且必须将相关的定时器 DMA 请求选为 DMA 源。当定时器最初设置为生成 DMA 请求时，可能会在匹配条件产生之前使该请求变为有效。可通过让软件写“1”以清除定时器中断标志来避免第一个 DMA 请求。DMA 请求将在处理后由 DMA 控制器通过硬件自动清除。

如果通道 0 或 1 的 EMR 位被置 10 或 11（上升沿或切换），即使相应 MR 寄存器被置 0，仍将产生 DMA 请求，因为存在“零匹配”条件。要禁能任意 DMA 请求，请将通道 0 或 1 的 EMR 位置 00。

14.7.11 计数控制寄存器

计数控制寄存器 (CTCR) 用于在定时器模式和计数器模式之间进行选择，且在处于计数器模式时选择进行计数的引脚和边沿。

如果将计数器模式选为操作模式，则在 PCLK 时钟的每个上升沿上会对（CTCR 位 3:2 所选的）CAP 输入进行采样。在比较该 CAP 输入的两个连续样本后，将会确认下列四个事件之一：所选 CAP 输入电平处于上升沿、下降沿，或上升下降沿或无变化。仅当所标识的事件与 CTCR 寄存器中的位 1:0 所选的内容相匹配时，定时器计数器寄存器才会递增计数。

要有效地处理计数器的外部源时钟会有一些限制。由于 PCLK 时钟的两个连续上升沿仅用于标识 CAP 所选输入的一个边缘，因此 CAP 输入的频率不能超过 PCLK 时钟的一半。因此在这种情况下，相同的 CAP 输入上的高电平 / 低电平的持续时间不能少于 1/PCLK。

该寄存器的位 7:4 还用于使能和配置捕获 - 清除 - 定时器特性。该特性允许特定 CAP 输入的指定边沿将定时器全部清零。使用该机制在输入脉冲前沿清除定时器然后在后沿执行捕获，可以使用单捕获输入来直接测量脉冲宽度，而无需在软件中执行减法操作。

表 248. 计数控制寄存器（CTCR，地址 0x4001 8070 (CT32B0) 和 0x4001 C070 (CT32B1)）位描述

位	符号	值	描述	复位值
1:0	CTM		计数器 / 定时器模式。该字段用于选择可使用哪个 PCLK 上升沿，使定时器预分频计数器 (PC) 递增计数，或清除 PC 并使定时器计数器 (TC) 递增计数。	00
			注：如果在 CTCR 中选择计数器模式，必须将捕获控制寄存器 (CCR) 中的位 2:0 设置为 000。	
		0x0	定时器模式：每个 PCLK 上升沿	
		0x1	计数器模式：TC 在位 3:2 选择的 CAP 输入的上升沿时递增。	
		0x2	计数器模式：TC 在位 3:2 选择的 CAP 输入的下降沿时递增。	
3:2	CIS	0x3	计数器模式：TC 在位 3:2 选择的 CAP 输入的两个边沿递增。	00
			计数输入选择。在计数器模式下（当该寄存器中位 1:0 不为 00 时），这些位将选择为时钟提供哪个 CAP 引脚或比较器输出。	
			注：如果在 CTCR 中选择计数器模式，必须将捕获控制寄存器 (CCR) 中与该输入对应的 3 位设置为 000。	
		0x0	CT32Bn_CAP0	
		0x1	CT32Bn_CAP1	
		0x2	CT32Bn_CAP2	
		0x3	CT32Bn_CAP3	

表 248. 计数控制寄存器（CTCR，地址 0x4001 8070 (CT32B0) 和 0x4001 C070 (CT32B1)）位描述（续）

位	符号	值	描述	复位值
4	ENCC		将此位置 1 可在发生位 7:5 指定的捕获 - 边沿事件时清零定时器和预分频器。	0
7:5	SEICC		当位 4 为 1 时，这两位选择哪个捕获输入边沿将导致定时器和预分频器被清零。当位 4 为低电平时，这两位无效。	
		0x0	CAP0 的上升沿清零定时器（如果位 4 被置位）	
		0x1	CAP0 的下降沿清零定时器（如果位 4 被置位）	
		0x2	CAP1 的上升沿清零定时器（如果位 4 被置位）	
		0x3	CAP1 的下降沿清零定时器（如果位 4 被置位）	
		0x4	CAP2 的上升沿清零定时器（如果位 4 被置位）	
		0x5	CAP2 的下降沿清零定时器（如果位 4 被置位）	
		0x6	CAP3 的上升沿清零定时器（如果位 4 被置位）	
		0x7	CAP3 的下降沿清零定时器（如果位 4 被置位）	
31:8	-	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	-

14.7.12 PWM 控制寄存器

PWM 控制寄存器用于将匹配输出配置为 PWM 输出。每个匹配输出均可分别设置，以决定匹配输出是作为 PWM 输出还是作为功能受外部匹配寄存器 (EMR) 控制的匹配输出。

对于定时器，MATn.2:0 输出最多可选择 3 个单边沿控制的 PWM 输出。一个附加的匹配寄存器决定 PWM 的周期长度。当任何其他匹配寄存器出现匹配时，PWM 输出置为高电平。用于设置 PWM 周期长度的匹配寄存器负责将定时器复位。当定时器复位到 0 时，所有当前配置为 PWM 输出的高电平匹配输出清零。

表 249. PWM 控制寄存器（PWMC，0x4001 8074 (CT32B0) 和 0x4001 C074 (CT32B1)）位描述

位	符号	值	描述	复位值
0	PWMEN0		通道 0 的 PWM 模式使能。	0
		0	CT32Bi_MAT0 受 EM0 控制。	
		1	CT32Bi_MAT0 的 PWM 模式使能。	
1	PWMEN1		通道 1 的 PWM 模式使能。	0
		0	CT32Bi_MAT01 受 EM1 控制。	
		1	CT32Bi_MAT1 的 PWM 模式使能。	
2	PWMEN2		通道 2 的 PWM 模式使能。	0
		0	CT32Bi_MAT2 受 EM2 控制。	
		1	CT32Bi_MAT2 的 PWM 模式使能。	
3	PWMEN3		通道 3 的 PWM 模式使能。 注： 建议使用匹配通道 3 设置 PWM 周期。	0
		0	CT32Bi_MAT3 受 EM3 控制。	
		1	CT132Bi_MAT3 的 PWM 模式使能。	
31:4	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

14.7.13 单边沿控制的 PWM 输出规则

- 1. 所有单边沿控制的 PWM 输出在 PWM 周期开始时都为低电平（定时器置为 0），除非它们的匹配值等于 0。
- 2. 每个 PWM 输出在达到其匹配值时将转入高电平。如果没有发生匹配（即匹配值大于 PWM 周期长度），则 PWM 输出将继续保持低电平。
- 3. 如果将大于 PWM 周期长度的匹配值写入到匹配寄存器，且 PWM 信号已经为高电平，则在下一个 PWM 周期开始时 PWM 信号将被清零。
- 4. 如果匹配寄存器中包含与定时器复位值（PWM 周期长度）相同的值，则在定时器达到匹配值后的下一个时钟节拍时 PWM 输出将复位到低电平。因此，PWM 输出总是包含一个时钟节拍宽度的正脉冲，周期由 PWM 周期长度决定（即定时器重载入值）。
- 5. 如果匹配寄存器置 0，则 PWM 输出将在定时器第一次返回 0 时变为高电平，并继续保持高电平。

注：当选择匹配输出作为 PWM 输出来执行时，除匹配寄存器设置 PWM 周期长度外，匹配控制寄存器 MCR 中的定时器复位 (MRnR) 和定时器停止 (MRnS) 位必须置为 0。对于该寄存器，当定时器值与相应的匹配寄存器值匹配时，将 MRnR 位置 1 以使能定时器复位。

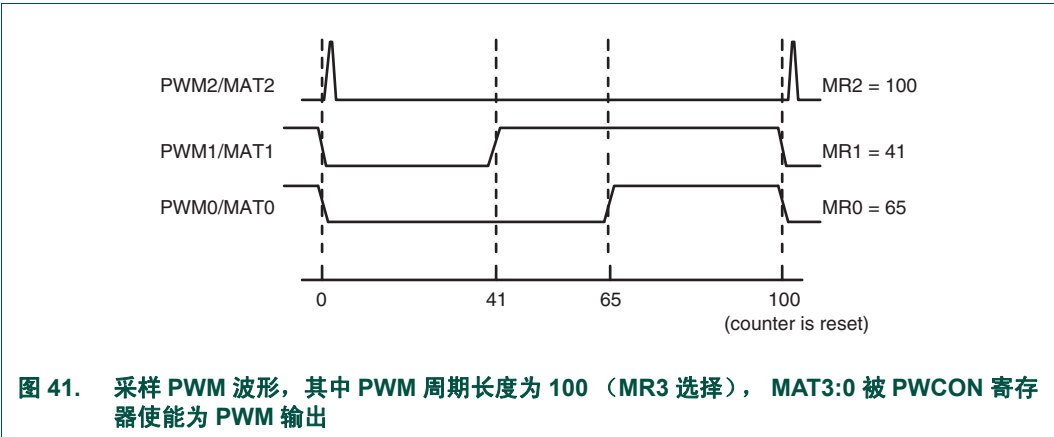
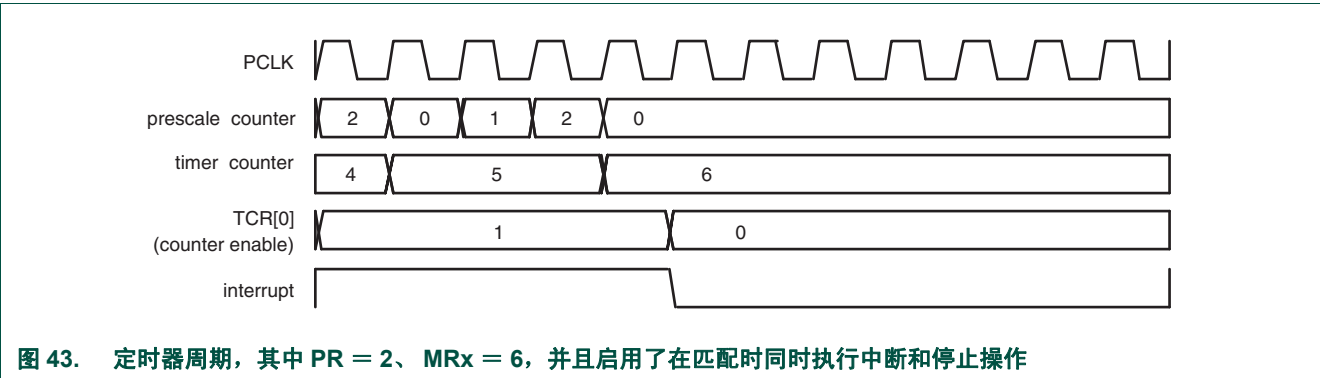
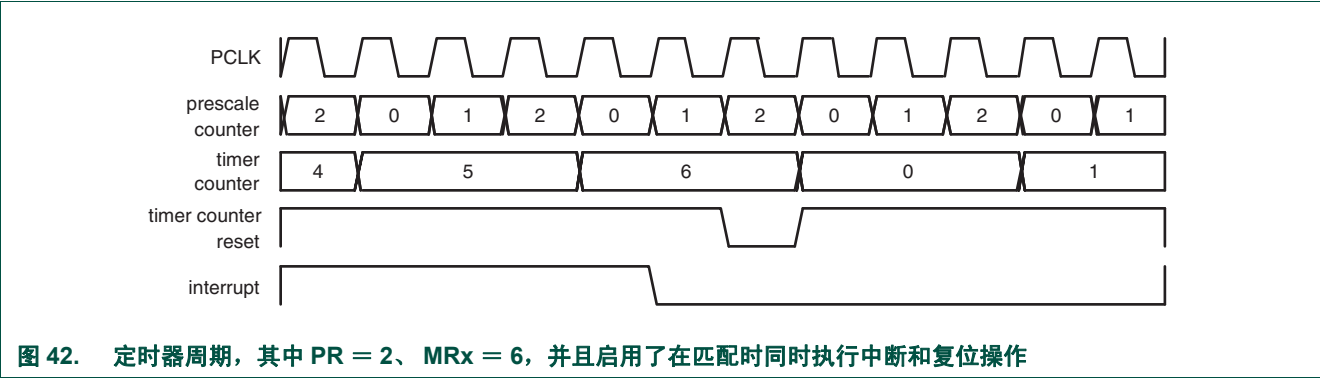


图 41. 采样 PWM 波形，其中 PWM 周期长度为 100（MR3 选择），MAT3:0 被 PWCON 寄存器使能为 PWM 输出

14.8 定时器操作示例

如图 42 所示，定时器配置为在匹配时复位计数并产生中断。预分频器置为 2，匹配寄存器置为 6。在发生匹配的定时器周期结束时，定时器计数复位。这样就使匹配值具有完整长度的周期。在定时器到达匹配值后的下一个时钟产生指示匹配发生的中断。

如图 43 所示，定时器配置为在匹配时停止计数并产生中断。预分频器再次置为 2，匹配寄存器置为 6。在定时器达到匹配值的下一个时钟，TCR 中的定时器使能位被清零，产生指示匹配发生的中断。



14.9 架构

32 位计数器 / 定时器 0 和 32 位计数器 / 定时器 1 的功能框图如[图 44](#) 所示。

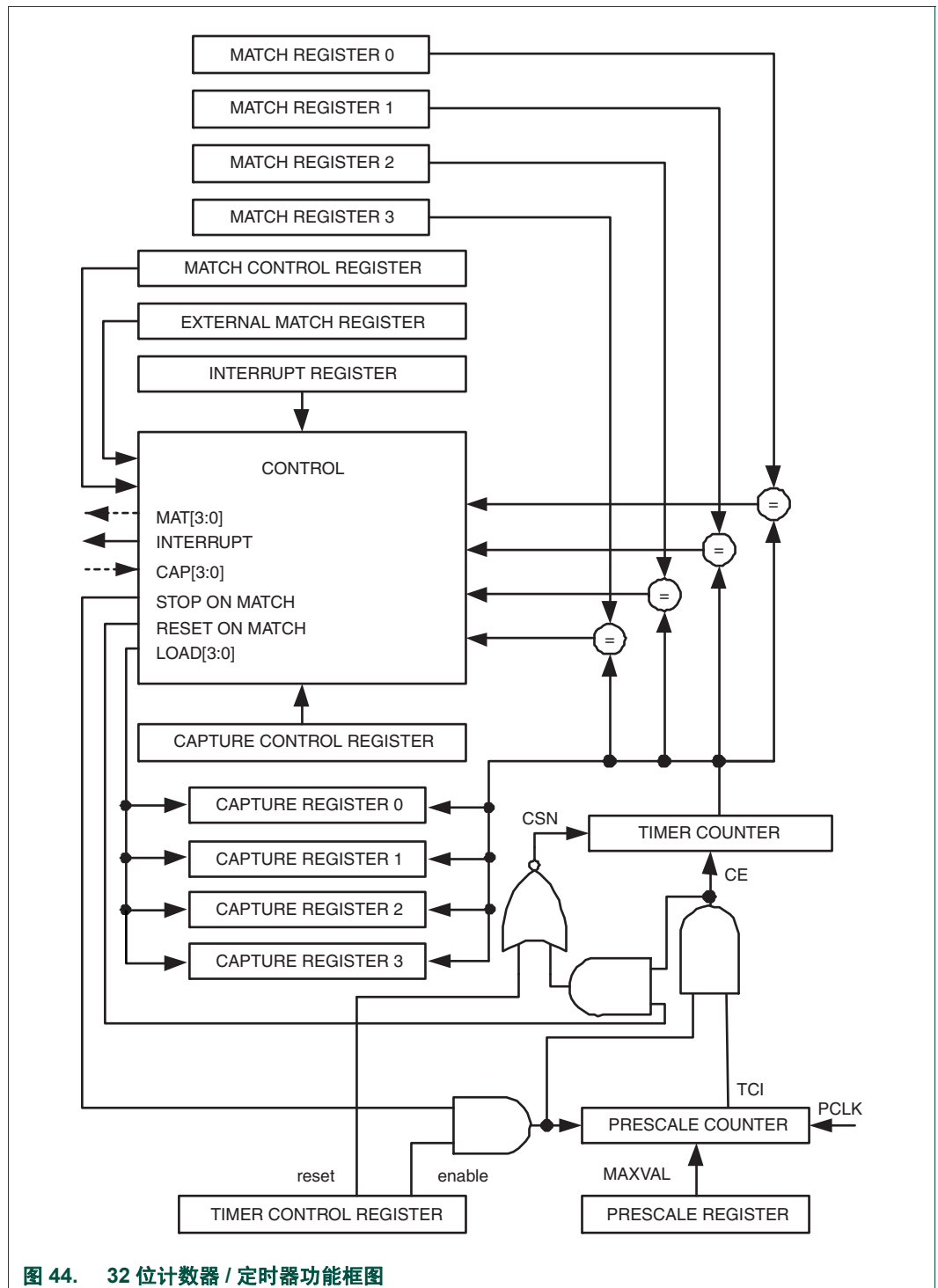


图 44. 32 位计数器 / 定时器功能框图

15.1 本章导读

系统节拍定时器（SysTick 定时器）是 ARM Cortex-M0 内核的一部分，而且对于所有 LPC122x 部件都是相同的。

15.2 基本配置

系统节拍定时器是使用以下寄存器进行配置的：

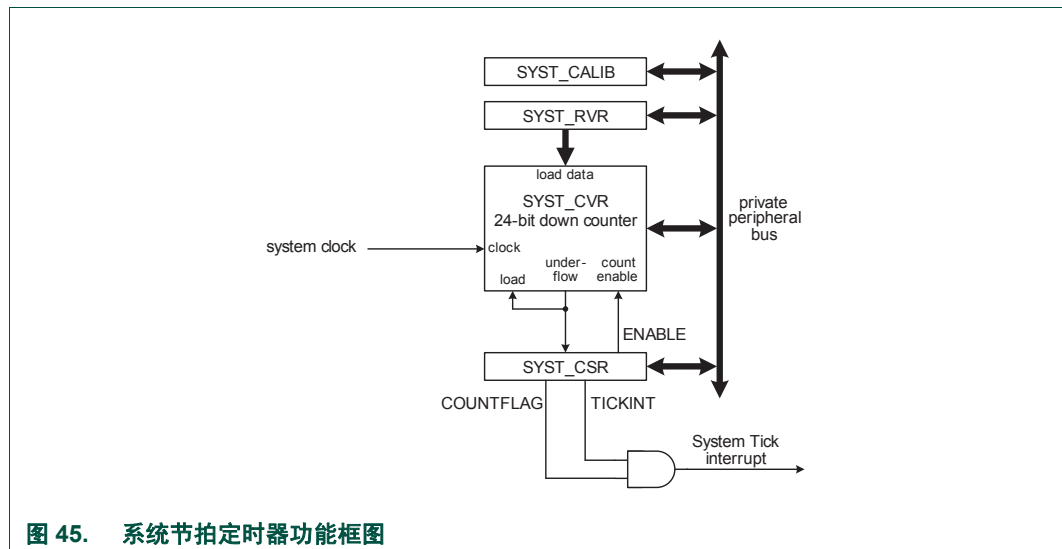
1. 引脚：系统节拍定时器不使用外部引脚。
2. 电源：系统节拍定时器通过 SysTick 控制寄存器 ([章节 15.5.1](#)) 启用。系统节拍定时器时钟固定为系统时钟频率的二分之一。

15.3 特性

- 简单的 24 位定时器。
- 使用专用的异常向量。
- 由系统时钟从内部定时。

15.4 简介

SysTick 定时器的功能框图如下面的[图 45](#)所示。



SysTick 定时器是 Cortex-M0 的主要组成部分。SysTick 定时器为操作系统或其它系统管理软件提供固定 10 毫秒的中断。

由于 SysTick 定时器是 Cortex-M0 的一部分，它为基于 Cortex-M0 的器件提供一个标准定时器，有助于软件的移植。SysTick 定时器可用于：

- 可编程设置频率的 RTOS 节拍定时器（例如 100 Hz），调用一个 SysTick 例程。
- 使用内核时钟的高速报警定时器。
- 简单的计数器。软件可使用它测量时间（如：完成任务所需时间、已使用的时间）。
- 基于丢失 / 命中期限控制的内部时钟源。控制和状态寄存器中的 COUNTFLAG 位字段可用于决定一个动作是否在设定的期限内完成，作为动态时钟管理控制环的一部分。

详情请参考 *Cortex-M0 用户指南*。

15.5 寄存器描述

systick 定时器寄存器位于 ARM Cortex-M0 专用外设总线上（参见图 65），是 ARM Cortex-M0 内核外设的一部分。有关详情，请参阅[章节 25.5.4](#)。

表 250. 寄存器简介：SysTick 定时器（基址 0xE000 E000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
SYST_CSR	R/W	0x010	系统定时器控制和状态寄存器	0x000 0000
SYST_RVR	R/W	0x014	系统定时器重载值寄存器	0
SYST_CVR	R/W	0x018	系统定时器当前值寄存器	0
SYST_CALIB	R/W	0x01C	系统定时器校准值寄存器	0x1F

[1] 复位值仅反映已使用位中保存的数据，它不包括保留位的内容。

15.5.1 系统定时器控制和状态寄存器

SYST_CSR 寄存器包含 SysTick 定时器的控制信息，并提供状态标志。该寄存器是 ARM Cortex-M0 内核系统定时器寄存器模块的一部分。有关该寄存器的位描述，请参阅[章节 25.5.4](#)。

该寄存器决定系统节拍定时器的时钟源。

表 251. SysTick 定时器控制和状态寄存器 (SYST_CSR - 0xE000 E010) 位描述

位	符号	描述	复位值
0	启用	系统节拍计数器启用。为 1 时，计数器启用。为 0 时，计数器禁用。	0
1	TICKINT	系统节拍中断启用。为 1 时，系统节拍中断启用。为 0 时，系统节拍中断禁用。启用时，在系统节拍计数器倒数到 0 时生成中断。	0
2	CLKSOURCE	保留	0
15:3	-	保留，用户软件不应对应保留位写入 1。从保留位读取的值未定义。	不适用
16	COUNTFLAG	如果自上一次对该寄存器进行读取以来 SysTick 定时器数到 0，则返回 1。	0
31:17	-	保留，用户软件不应对应保留位写入 1。从保留位读取的值未定义。	不适用

15.5.2 系统定时器重载值寄存器

SYST_RVR 寄存器被设置为 SysTick 定时器倒数到 0 时载入的值。在定时器进行初始化时，该寄存器由软件载入。如果 CPU 运行频率适合用 SYST_CALIB 值，则可对 SYST_CALIB 寄存器进行读取，并将其用作 SYST_RVR 寄存器的值。

表 252. 系统定时器重载值寄存器 (SYST_RVR - 0xE000 E014) 位描述

位	符号	描述	复位值
23:0	RELOAD	该值在系统节拍计数器倒数到 0 时载入该计数器。	0
31:24	-	保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用

15.5.3 系统定时器当前值寄存器

当软件读取 SYST_CVR 寄存器时，它将从系统节拍计数器返回当前计数。

表 253. 系统定时器当前值寄存器 (SYST_CVR - 0xE000 E018) 位描述

位	符号	描述	复位值
23:0	CURRENT	读该寄存器会返回系统节拍计数器的当前值。写任意值都可将系统节拍计数器和 STCTRL 中的 COUNTFLAG 位清零。	0
31:24	-	保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用

15.5.4 系统定时器校准值寄存器 (SYST_CALIB - 0xE000 E01C)

SYST_CALIB 寄存器的值由系统配置模块中 SYSTCKCAL 寄存器的值驱动（参见[章节 4.5.26](#)）。

表 254. 系统定时器校准值寄存器 (SYST_CALIB - 0xE000 E01C) 位描述

位	符号	值	描述	复位值
23:0	TENMS		参见 表 397 。	0x1F
29:24	-		保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用
30	SKEW		参见 表 397 。	0
31	NOREF		参见 表 397 。	0

15.6 功能说明

SysTick 定时器是一个 24 位定时器，可倒计数到 0，还可生成中断。SysTick 定时器的作用就是为每次中断之间提供一个 10 毫秒的固定时间间隔。SysTick 定时器的时钟信号可由 CPU 时钟（系统时钟，参见[图 3](#)）提供，也可由参考时钟（固定为 CPU 时钟频率的二分之一）提供。若要在特定的间隔上生成循环中断，就必须使用所需间隔的正确值初始化 SYST_RVR 寄存器。默认值保存在 SYST_CALIB 寄存器中，可通过软件进行修改。如果将 CPU 时钟设置为 <tb> MHz，则默认值为 10 毫秒中断速率。

15.7 定时器计算示例

使用系统节拍定时器需遵循以下规则：

- 1. 用重载值 RELOAD 对 SYST_RVR 寄存器进行编程，以获得所需的时间间隔。
- 2. 通过写操作将 SYST_CVR 寄存器清零。这样能确保定时器启用后可以从 SYST_RVR 值开始计数，而不是从任意值开始计数。
- 3. 取值 0x7 对 SYST_SCR 寄存器进行编程，即可启用 SysTick 定时器和 SysTick 定时器中断。

以下示例说明，如果将 LPC122x 系统时钟设置为 30 MHz，选择 SysTick 定时器重载值可以获得 10 ms 的时间间隔。

示例（系统时钟 = 30 MHz）。

系统节拍时钟 = 系统时钟 = 30 MHz。

$\text{RELOAD} = (\text{系统节拍时钟频率} \times 10 \text{ ms}) - 1 = (30 \text{ MHz} \times 10 \text{ ms}) - 1 = 300000 - 1 = 299999 = 0x000493DF。$

16.1 本章导读

实时时钟 (RTC) 在所有 LPC122x 部件上都可用。

16.2 基本配置

RTC 时钟由 PMU 中的 SYSCFG（[表 58](#)）寄存器控制，并可以从 RTC 内部振荡器或主时钟中进行选择。[表 255](#) 显示了根据 LPC122x 的电源模式提供的时钟选项（另请参见[图 46](#)）。

表 255. RTC 时钟选项

时钟	选项	参考	工作模式	睡眠模式	深度睡眠模式	深度掉电模式
RTC 内部振荡器	1 Hz（默认值）、1 kHz、延迟的 1 Hz	表 58	是	是	是	是
主时钟	看门狗振荡器	表 18	是	是	是	否
	IRC	表 18	是	是	否	否
	PLL	表 18	是	是	否	否
	输入至 PLL	表 18	是	是	否	否

注：当 RTC 正在运行时，不得更改输入时钟。

RTC 的供电方式如下：

- 只要 $V_{DD(3V3)}$ 处于有效的电平，RTC 的模拟模块便始终开启。
- 默认情况下，RTC 寄存器接口在 SYSAHBCLKCTRL 寄存器（[表 21](#)）中启用。
- 启用 RTC 可使用 CR 寄存器（[表 260](#)）创建匹配事件。

16.3 特性

- 专用的 32 kHz 超低功耗振荡器。
- 使用 1 Hz 时钟、延迟的 1 Hz 时钟、1 kHz 时钟或外设 RTC 时钟作为输入。
- 32 位计数器。
- 可编程的 32 位匹配 / 比较寄存器。
- 当计数器和匹配寄存器相同时，软件可屏蔽中断。
- RTC 可将设备从深度睡眠模式和深度掉电模式中唤醒。

16.4 简介

RTC 和专用的 32 kHz 振荡器 $V_{DD(3V3)}$ 在独立的电源域中运行，以便只要 $V_{DD(3V3)}$ 存在，便会在复位和所有低功耗模式期间保留 RTC 计数及匹配值。只要工作电压 $V_{DD(3V3)}$ 达到有效的电平，包括 RTC 内部计数器和 RTC 振荡器在内的低功耗模拟部分便会持续运行。一旦在 CR 寄存器中启用，RTC 便可以创建匹配事件和中断。有关 RTC 时钟连接，请参阅图 46。

RTC 使用 32 位计数器和 32 位匹配寄存器。软件可屏蔽中断。

有关详情，请参阅[章节 16.6](#)。

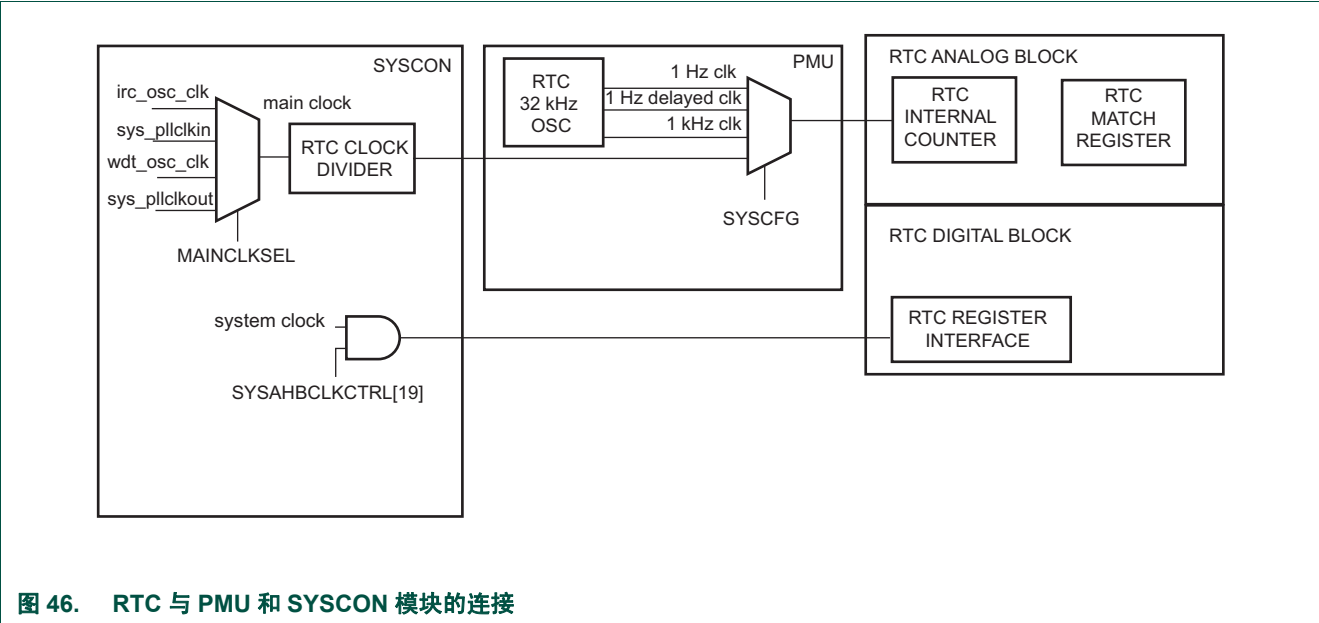


图 46. RTC 与 PMU 和 SYSCON 模块的连接

16.5 寄存器描述

表 256. 寄存器简介：RTC（基址 0x4005 0000）

名称	访问类型	地址偏移	描述	复位值
DR	R	0x000	数据寄存器	0x00
MR	R/W	0x004	匹配寄存器	0x00
LR	R/W	0x008	加载寄存器	0x00
CR	R/W	0x00C	控制寄存器	0x00
ICSC	R/W	0x010	中断控制设置 / 清除寄存器	0x00
RIS	R	0x014	原始中断状态寄存器	0x00
MIS	R	0x018	屏蔽中断状态寄存器	0x00
ICR	W	0x01C	中断清除寄存器	0x00

16.5.1 RTC 数据寄存器

此寄存器可返回 RTC 计数器的当前值。请注意，启用 RTC 后，同步时段生效；在此期间，即使内部 RTC 计数器计算正确，此寄存器也不会返回有效值（请参阅[章节 16.6.1](#)）。

表 257. RTC 数据寄存器（DR - 地址 0x4005 0000）位描述

位	符号	描述	复位值
31:0	DATA	返回当前的 RTC 值。	0x00

16.5.2 RTC 匹配寄存器

可在 RTC 初始化期间及运行定时器中随时写入 RTC 匹配寄存器。在当前 RTC 值与匹配寄存器的内容相同时，会发生匹配事件。

如果在复位生效时发生匹配事件，则除非死区时间到期后在软件中测试匹配条件，否则不会在事件上创建中断且 RTC 会错过匹配事件。有关详情，请参阅[章节 16.6.1](#)。

表 258. RTC 匹配寄存器（MR - 地址 0x4005 0004）位描述

位	符号	描述	复位值
31:0	MATCH	RTC 匹配寄存器值。	0x00

16.5.3 RTC 加载寄存器

RTC 加载寄存器可在运行时使用任何值加载或更新内部 RTC 计数器值。对加载寄存器执行写入操作会更新内部 RTC 计数器的偏移。

表 259. RTC 加载寄存器（LR - 地址 0x4005 0008）位描述

位	符号	描述	复位值
31:0	LOAD	RTC 加载寄存器值。	0x00

16.5.4 RTC 控制寄存器

此寄存器为读 / 写 (R/W) 寄存器。读取操作会返回 RTC 的状态。写入操作会启用或禁用 RTC。启用 RTC 后，对此寄存器位 0 的任何写入操作只有在系统复位后才会生效。

表 260. RTC 控制寄存器（CR - 地址 0x4005 000C）位描述

位	符号	值	描述	复位值
0	RTCSTART		启用 RTC。通过此位启用 RTC 后，对此位的任何写入操作只有在上电复位 (POR) 后才会生效。	0x0
		0	禁用 RTC。	
		1	启用 RTC。	
31:1	-		保留。未定义读取。这些位应写为零。	-

16.5.5 RTC 中断控制设置 / 清除寄存器

此寄存器为 R/W 寄存器，可控制 RTC 产生的中断屏蔽。写入操作可设置或清除屏蔽。读取此寄存器可返回 RTC 中断上屏蔽的当前值。

表 261. RTC 中断屏蔽寄存器（ICSC - 地址 0x4005 0010）位描述

位	符号	值	描述	复位值
0	RTCIC		中断控制寄存器。读取操作将返回 RTC 控制寄存器的当前值。	0x0
		0	写入 0 将屏蔽中断。	
		1	写入 1 将启用中断。	
31:1	-		保留。读取为零。请勿修改这些位。	0x0

16.5.6 RTC 中断状态寄存器

此寄存器为只读 (RO) 寄存器。读取此寄存器将提供屏蔽前相应中断的当前原始状态值。写入操作无效。

表 262. RTC 中断状态寄存器（RIS - 地址 0x4005 0014）位描述

位	符号	描述	复位值
0	RTCRIIS	原始中断事件标志寄存器。读取操作将返回原始中断事件标志的状态。	0x0
31:1	-	保留。读取为零。	0x0

16.5.7 RTC 屏蔽中断状态寄存器

此寄存器为只读 (RO) 寄存器。读取此寄存器将提供相应中断的当前屏蔽状态值。写入操作无效。

表 263. RTC 屏蔽中断状态寄存器（MIS - 地址 0x4005 0018）位描述

位	符号	描述	复位值
0	RTCMIS	屏蔽中断寄存器状态。读取操作将返回由 ICR 寄存器控制的屏蔽中断状态。	0x0
31:1	-	保留。读取为零。	0x0

16.5.8 RTC 中断清除寄存器

此寄存器为只写 (WO) 寄存器。写入 1 会清除相应的中断。写入 0 无效。

表 264. RTC 中断清除寄存器 (ICR - 地址 0x4005 001C) 位描述

位	符号	描述	复位值
0	RTCICR	原始中断事件标志清除寄存器。写入 1 将清除中断事件标志。写入 0 无效。	0x0
31:1	-	保留。写为零。	0x0

16.6 功能说明

16.6.1 RTC 启动行为

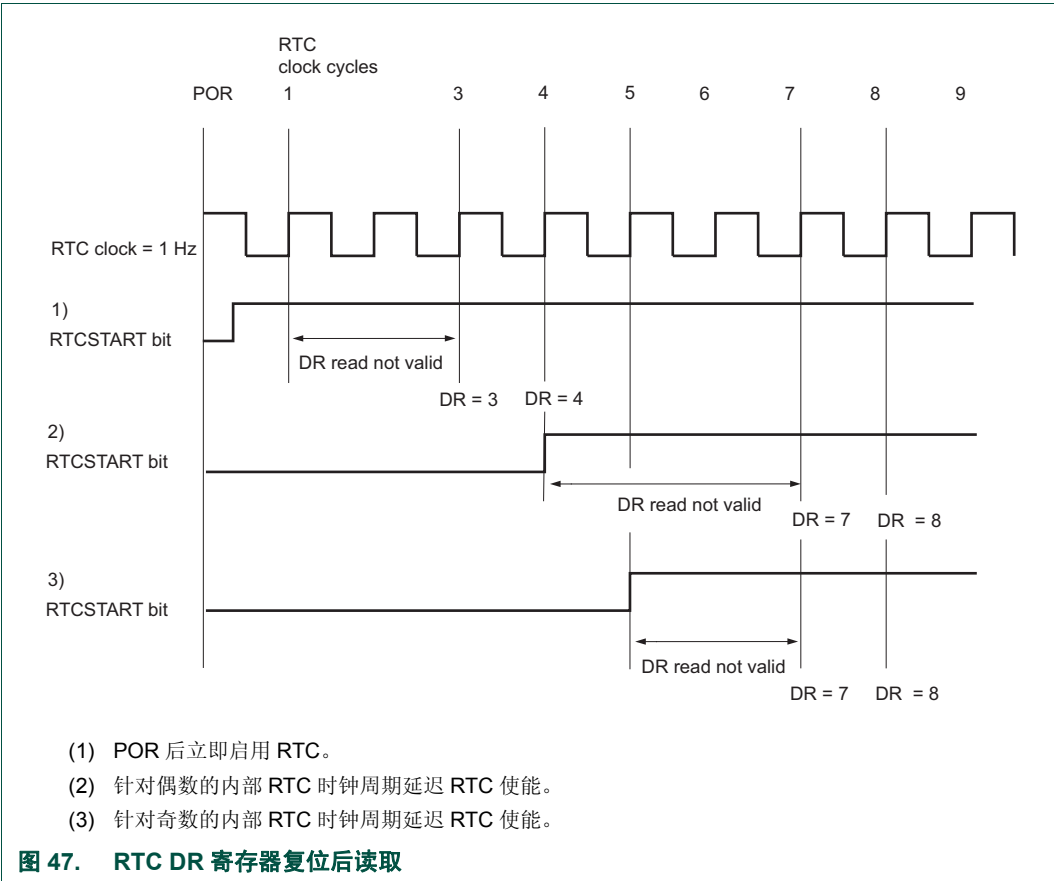
RTC 内部计数器在上电复位 (POR) 时以默认时钟速率 (1 Hz, 见表 58) 从 0 开始计数, 而 RTC 内部计数器值只能通过 POR 复位。只要芯片保持通电, RTC 便会在所有其他复位事件 (BOD、WWDT 复位、软件复位、RESET 引脚) 及深度掉电模式期间继续计数。

但在 POR 后, DR 寄存器的内容在最多三个 RTC 时钟周期的同步时段内不会从 CR 寄存器中启用 RTC 的时间开始定义。请注意, 即使 DR 寄存器没有产生有效的结果, 内部计数器也不会走慢。

如果 RTC 在内部 RTC 计数的奇数时启用, 则读取 DR 寄存器的同步时段为两个时钟周期; 如果 RTC 在内部 RTC 计数的偶数时启用, 则读取 DR 寄存器的同步时段为三个时钟周期。

图 47 显示了在 CR 寄存器中启用 RTC 与从 DR 寄存器中读取有效计数之间产生延迟的示例。

注: 建议在 POR 后 RTC 时钟的首个时钟周期期间通过软件启用并初始化 RTC (图 47 中的示例 1)。



16.6.2 RTC 匹配中断

当匹配值与加载寄存器的内容相同时，会发生匹配事件。匹配值只能在 POR 期间复位。如果 RTC 匹配与计数值相同时，RTC 处于复位状态，则不会产生中断。在此情况下，软件必须检查在释放复位后是否已发生任何匹配事件。

16.6.3 在深度睡眠或深度掉电模式下使用 RTC

RTC 可以配置为在产生 RTC 中断时，将芯片从深度睡眠或深度掉电模式中唤醒。有关详情，请参阅[章节 4.8.3](#)和[章节 4.9.2](#)。

如果在深度掉电模式下由于主时钟不可用而使用 RTC 计时，应始终选择 RTC 振荡器的其中一个输出作为 RTC 时钟。在深度睡眠模式下，RTC 时钟应为看门狗振荡器或应来自 RTC 振荡器。

注 为了在从深度掉电模式中唤醒后获取有效的 RTC 值，应先对 RTC 执行“虚拟”读取。下一次读取将包含更新的 RTC 值。

17.1 本章导读

窗口看门狗定时器 (WWDT) 在所有 LPC122x 部件上都可用。

17.2 基本配置

看门狗定时器会在复位后由 Boot ROM 自动启用，而用户必须继续喂狗序列或禁用看门狗定时器，以免造成器件意外复位。

17.3 特性

- 如果没有在可编程超时期限内重载，则产生片内复位。
- 时间周期可选，从 1,024 个看门狗时钟 ($T_{WDCLK} \times 256 \times 4$) 到超过 6,700 万个看门狗时钟 ($T_{WDCLK} \times 2^{24} \times 4$)，取 4 个看门狗时钟的位数。
- 带内置预分频器的可编程 24 位定时器。
- 安全的门狗操作。一旦启用看门狗，便需要禁用硬件复位或看门狗复位。
- 看门狗时钟 (WDCLK) 源可从内部 RC 振荡器 (IRC) 或看门狗振荡器中选择，见[表 13](#)。这为看门狗在不同功率降低条件下提供了较宽的潜在时序选择范围。为了提高可靠性，它还可以使看门狗定时器在与外部晶振及其相关元件和线路无关的内部时钟源下运行。
- 如果启用看门狗，则不正确 / 不完整的喂狗序列会产生复位 / 中断。
- 具有指示看门狗复位的标志。
- 可选窗口式操作需要在可编程的最小与最大超时期限之间发生重载
- 可在看门狗超时前的可编程时间产生可选的警告中断。
- 可以选择性保护看门狗重载值，以便只有在达到警告中断时间后，才能更改该值。
- 可以将看门狗定时器配置为在深度睡眠模式下运行。
- 具有调试模式。
- 通过 IEC-60335 Class B 认证。

17.4 描述

看门狗定时器的用途是使微控制器在进入错误状态后的一定时间内复位。当看门狗启用时，如果用户程序没有在预定时间内喂狗（或给看门狗定时器重载定时值），则将产生看门狗事件。如果加以配置，看门狗事件将导致芯片复位。

在对看门狗窗口进行编程时，提前看门狗喂狗也会视为看门狗事件。此设计可防止出现系统故障仍可喂狗的情况。例如，应用程序代码可能卡在包含看门狗喂狗的中断服务中。设置窗口，以便导致提前喂狗，从而产生看门狗事件以实现系统恢复。

看门狗包括一个固定的 4 分频的预分频器和一个 24 位计数器，计数器每个时钟周期递减。计数器递减的最小值为 0xFF。如果设置小于 0xFF 的值，则系统会将 0xFF 装入计数器。因此，看门狗的最小间隔为 $(T_{WDCLK} \times 256 \times 4)$ ，最大间隔为 $(T_{WDCLK} \times 2^{24} \times 4)$ ，两者都是 $(T_{WDCLK} \times 4)$ 的倍数。看门狗应按如下方法使用：

- 在 TC 寄存器中设置看门狗定时器固定的重装值。
- 在 MOD 寄存器中设置看门狗定时器的工作模式。
- 如果需要窗口式操作，则在 WINDOW 寄存器中设置看门狗窗口时间的值。
- 如果需要警告中断，则在 WARNINT 寄存器中设置看门狗警告中断的值。
- 向 FEED 寄存器先写入 0xAA，再写入 0x55 以启用看门狗。
- 在看门狗计数器计数到零之前必须再次喂狗，以免发生看门狗事件。如果已对窗口值编程，则还必须在看门狗计数器超过该值后喂狗。

当对看门狗定时器进行配置，以便看门狗事件将导致复位且计数器计数到零时，CPU 将复位，并从向量表中加载堆栈指针和编程计数器（与外部复位情况相同）。可检查看门狗超时标志 (WDTOF) 以决定看门狗是否已引起复位条件。WDTOF 标志必须通过软件清零。

如果看门狗定时器配置为产生警告中断，则在计数器与寄存器定义的值匹配时将出现中断。

17.5 时钟和电源控制

看门狗定时器模块使用两个时钟：PCLK 和 WDCLK。PCLK 由系统时钟生成（见图 3），供 APB 访问看门狗寄存器使用。WDCLK 由图 3 中的 wdt_clk 生成，供看门狗定时器计数使用。以下两个时钟可用作 wdt_clk 时钟的时钟源：IRC 和看门狗振荡器。时钟源在 WDCLKSEL 寄存器（表 271）中选择，但请注意，时钟源可以由软件通过 MODE 寄存器锁定。

这两个时钟域之间存在一些同步逻辑。当 MOD 和 TC 寄存器通过 APB 操作更新时，新的值将在 WDCLK 时钟域内逻辑的 3 个 WDCLK 周期内生效。当看门狗定时器处于 WDCLK 时钟周期时，同步逻辑会先锁定 WDCLK 上计数器的值，然后使其与 PCLK 同步，以作为 TV 寄存器的值供 CPU 读取。

如果没有使用看门狗振荡器，则可在 PDRUNCFG 寄存器（章节 4.5.40）中将其关闭，除非设置了 MOD 寄存器中的位 5（表 266）。为了节能，可在 AHBCLKCTRL 寄存器（表 21）将输入到看门狗寄存器模块的时钟 (PCLK) 禁用。

17.6 看门狗锁定功能

可以通过多种方式锁定看门狗定时器操作，以确保 WDT 始终处于运行状态。锁定功能可以通过一次写入相应的锁定寄存器位来启用，但只有通过芯片复位才能取消。

可应用以下锁定机制：

- 锁定 WDT 的启用 / 禁用状态，并同时锁定看门狗是否触发中断或复位（[表 266](#)）。
- 锁定时钟源的切换。此锁定机制可防止切换至已掉电的时钟源（[表 271](#)）。
- 锁定 PDRUNCFG、PDSLEEPCFG、PDAWAKECFG 寄存器中任何 WDT 时钟源的电源控制（[表 266](#)）。
- 锁定更新 WDT 重载值（[表 266](#)）。
- 锁定进入深度掉电模式（[表 266](#)）。

备注：必须谨慎使用锁定功能。

- 确保在锁定电源控制和时钟源选择前，所有三个电源配置寄存器 PDSLEEPCFG、PDRUNCFG 和 PDAWAKECFG 中所选的 WDT 时钟源已上电。
- 如果在深度睡眠模式下使用 WDT，则必须在锁定电源控制前开启看门狗振荡器。

17.7 寄存器描述

表 265. 寄存器简介：看门狗定时器（基址 0x4000 4000）

名称	访问类型	地址偏移	描述	复位值 ^[1]
MOD	R/W	0x000	看门狗模式寄存器。该寄存器包含看门狗定时器的基本模式和状态。	0x0000 0003
TC	R/W	0x004	看门狗定时器常量寄存器。该寄存器确定超时值。	0x0000 FFFF
FEED	WO	0x008	看门狗喂狗序列寄存器。依序向该寄存器写入 0xAA 和 0x55 使看门狗定时器重新载入 TC 中的值。	不适用
电视	RO	0x00C	看门狗定时器值寄存器。该寄存器读出看门狗定时器的当前值。	0xFF
CLKSEL	R/W	0x010	看门狗时钟源选择寄存器。	0
WARNINT	R/W	0x014	看门狗警告中断比较值。	0
WINDOW	R/W	0x018	看门狗窗口比较值。	0xFF FFFF

[1] 复位值仅反映使用位中保存的数据，不包含保留位的内容。

17.7.1 看门狗模式寄存器

MOD 寄存器通过 WDEN 和 RESET 位的组合可控制看门狗的操作。

在看门狗运行时可随时产生看门狗复位或中断，且看门狗复位或中断还具有工作时钟源。如果在睡眠模式中出现看门狗中断，则看门狗中断会唤醒器件。

表 266. 看门狗模式寄存器（MOD - 0x4000 4000）位描述

位	符号	值	描述	复位值
0	WDEN		看门狗启用位。通过 WDLOCKEN 位后续写入可锁定 WDEN 位。	1
		0	看门狗定时器停止。	
		1	看门狗定时器运行。看门狗定时器会在复位时自动启用，而无需有效的喂狗序列。对此位的任何后续写入操作需要有效的喂狗序列，更改才会生效。	

表 266. 看门狗模式寄存器 (MOD - 0x4000 4000) 位描述 (续)

位	符号	值	描述	复位值
1	WDRESET		看门狗复位使能位。可随时更改此位。WDRESET 位通过外部复位或看门狗定时器复位设置。通过 WDLOCKEN 位后续写入可锁定 WDRESET 位。	1
		0	看门狗超时会引起中断。	
		1	看门狗超时会引起芯片复位。	
2	WDTOF		看门狗超时标志。在看门狗超时、发生喂狗错误或当 WDPROTECT=1 且尝试向 WDTC 寄存器写入时，便会设置看门狗超时标志。通过软件向此位写入 0 可将该标志清零。如果 WDRESET=1，将引起芯片复位。	0 (仅在外部复位后)
3	WDINT		当看门狗计数器计数到 WDWARNINT 指定的值时，便会设置看门狗中断标志。此标志可通过任何复位或通过软件向此位写入 1 清零。	0
4	WDPROTECT		看门狗更新模式。此位只能设置。WDPROTECT 位一旦设置，便无法通过软件清零。WDPROTECT 位可通过外部复位或看门狗定时器复位清零。	0
		0	看门狗定时器常量值 (WDTC) 可随时更改。	
		1	只有计数器小于 WARNINT 和 WINDOW 的值后，才能更改看门狗定时器常量值 (WDTC)。	
5	WDLOCKCLK		看门狗时钟锁定位。此位可在复位时清零，且随后只能写入一次以进行设置。此位一旦设置，便只能通过芯片复位清零。	0
		0	看门狗时钟 (WDCLK) 可随时禁用。	
		1	设置此位将禁止向控制电源配置寄存器 PDRUNCFG、PDSLEEPCFG 和 PDAWAKECFG 中当前所选看门狗时钟源电源的位执行任何写入操作。电源配置寄存器中的其他位将不受影响。设置 WDLOCKCLK 位确保只要看门狗振荡器和 / 或 IRC 上电，WDT 便始终具有 WDCLK 的有效时钟源。 备注： 在设置 WDLOCKCLK 位前，用户必须在所有三个电源配置寄存器中启用看门狗振荡器或 IRC 或两者，以确保所选的时钟源在活动、睡眠或深度睡眠模式下运行。一旦 WDLOCKCLK 位设置，看门狗时钟源便无法关闭或开启。如果 WDT 将在深度睡眠模式下运行，则必须先在 PDSLEEPCFG 寄存器中启用看门狗振荡器，再设置 WDLOCKCLK 位 (见 表 48 “深度睡眠配置寄存器 (PDSLEEPCFG、地址 0x4004 8230) 位描述”)。	
6	WDLOCKDP		深度掉电禁用位。此位可在复位时清零，且随后只能写入一次以进行设置。此位一旦设置，便只能通过芯片复位清零。	0
		0	可以随时进入深度掉电模式。	
		1	PMU 中的 DPDEN 位 (见 表 56 “电源控制寄存器 (PCON，地址 0x4003 8000) 位描述”) 不能设为 1。	
7	WDLOCKEN		看门狗启用和复位锁定位。此位可在复位时清零，且随后只能写入一次以进行设置。此位一旦设置，便只能通过芯片复位清零。	0
		0	软件可随时写入 WDEN 和 WDRESET 位以启用或禁用看门狗操作。	
		1	如果此位设为 1，则将阻止 WDEN 和 WDRESET 位的任何后续写入。当设置 WDLOCK 位时，将根据 WDEN 位的状态永久禁用或启用看门狗。复位行为受影响如下： <ul style="list-style-type: none">如果设置 WDLOCKEN 位之前，看门狗启用且 WDRESET 位设为 1，则看门狗触发器会始终引起复位，且此行为无法由软件覆盖。如果设置 WDLOCKEN 位之前，看门狗启用且 WDRESET 位设为 0，则看门狗触发器会始终引起中断，且此行为无法由软件覆盖。	
31:8	-		保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	-

表 267. 看门狗工作模式选择

WDEN	WDRESET	工作模式
0	X (0 或 1)	调试 / 操作模式 (看门狗未运行)
1	0	看门狗中断模式：将产生看门狗警告中断，但不会产生看门狗复位。 当选择这种模式时，看门狗计数器计数到 WDWARNINT 指定的值时便会设置 WDINT 标志，并产生看门狗中断请求。
1	1	看门狗复位模式：启用看门狗中断和看门狗复位。 当选择这种模式时，看门狗计数器计数到 WDWARNINT 指定的值时便会设置 WDINT 标志，并产生看门狗中断请求。看门狗计数器为零将复位微控制器。 看门狗复位的其他原因包括：在计数到 WDWINDOW 的值之前看门狗喂狗或更改 WDTN 值 (如果在 MOD 寄存器中设置 WDPROTECT 位)。

17.7.2 看门狗定时器常量寄存器

TC 寄存器决定超时值。每当喂狗序列产生时，TC 的内容就会重新载入看门狗定时器。复位时，值 0x00 FFFF 会预载。写入小于 0xFF 的值会使 0xFF 载入 TC。因此，最小超时间隔为 $T_{WDCLK} \times 256 \times 4$ 。

如果 MOD 中的 WDPROTECT 位为 1，则在看门狗计数器小于 WARNINT 和 WINDOW 的值之前，尝试更改 TC 值将引起看门狗复位并设置 WDTN 标志。

表 268. 看门狗定时器常量寄存器 (TC - 0x4000 4004) 位描述

位	符号	描述	复位值
23:0	WDTN	看门狗超时间隔。	0x00 FFFF
31:24	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	-

17.7.3 看门狗喂狗寄存器

向该寄存器依序写入 0xAA 和 0x55 将使 WDTN 的值重新载入看门狗定时器。如果看门狗已通过 WDMOD 寄存器启用，则此操作也会启动看门狗。设置 WDMOD 寄存器中的 WDEN 位不足以启用看门狗。设置 WDEN 后，还必须完成一个有效的喂狗序列，看门狗才能产生复位。在此之前，看门狗将忽略喂狗错误。向 WDFEED 写入 0xAA 之后，必须紧接着写入 0x55，否则如果看门狗启用，访问任何看门狗寄存器将会立即产生复位 / 中断，并设置 WDTN 标志。在喂狗序列期间错误地访问了看门狗寄存器，将会在第二个 PCLK 周期产生复位。

在喂狗序列期间应禁用中断。如果在喂狗序列期间发生中断，则会产生中止条件。

表 269. 看门狗喂狗寄存器 (FEED - 0x4000 4008) 位描述

位	符号	描述	复位值
7:0	Feed	喂狗值应依序为 0xAA 和 0x55。	-
31:8	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	-

17.7.4 看门狗定时器值寄存器

WDTV 寄存器用于读取看门狗定时器的当前值。

当读取 24 位计数器的值时，锁定和同步过程需要 6 个 WDCLK 周期和 6 个 PCLK 周期，因此，WDTV 的值比 CPU 正在读取的定时器的实际值要“旧”。

表 270. 看门狗定时器值寄存器 (TV - 0x4000 400C) 位描述

位	符号	描述	复位值
23:0	Count	计数器定时器值。	0x00 00FF
31:24	-	保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用

17.7.5 看门狗定时器时钟源选择寄存器

该寄存器可选择看门狗定时器的时钟源。时钟源选择位可由软件通过此寄存器的位 31 锁定，以便无法修改。此外，如果看门狗振荡器和 IRC 在 PDRUNCFG 寄存器中未同时上电，则将忽略时钟源的更改。此设计可防止用户切换到不存在的时钟源。

如果 WDT 在深度睡眠模式下运行，则必须选择看门狗振荡器作为时钟源。

复位时，时钟源选择位始终处于解锁状态。

表 271. 看门狗定时器时钟源选择寄存器（CLKSEL - 地址 0x4000 4010）位描述

位	符号	值	描述	复位值
1:0	WDSEL		这些位可选择看门狗定时器的时钟源（如下所述）。 警告： 此值设置不当会导致看门狗定时器无法正常工作，从而可能对系统运行造成不利影响。如果设置此寄存器中的 WDLOCK 位，则无法修改 WDSEL 位。 注： 如果相应的时钟源在 PDRUNCFG 寄存器（ 表 50 ）中掉电，则将忽略 WDSEL 位的写入。	00
		0x0	选择内部 RC 振荡器作为看门狗时钟源（默认值）。仅限于工作模式。	
		0x1	选择看门狗振荡器作为看门狗时钟源。如果 WDT 在深度睡眠模式下运行，则必须选择此设置。	
		0x2	保留。不使用。	
		0x3	保留。不使用。	
30:2	-	-	保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用
31	WDLOCK		锁定看门狗时钟源。	0
		0	在复位时，此位设为 0。无法由软件清零。	
		1	软件可以随时将此位设为 1。一旦设置 WDLOCK，则无法修改此寄存器的位。	

17.7.6 看门狗定时器警告中断寄存器

WDWARNINT 寄存器决定将产生看门狗中断的看门狗定时器计数器值。当看门狗定时器计数器与 WDWARNINT 定义的值匹配时，将在后续 WDCLK 后产生中断。

当计数器的低 10 位与 WARNINT 的 10 位具有相同值，且计数器其余高位均为 0 时，便产生 WDWARNINT 的看门狗定时器计数器匹配。这会产生 1,023 个看门狗定时器计数（4,096 个看门狗时钟）的发生看门狗事件之前的最长中断时间。如果 WARNINT 设为 0，则将与看门狗事件同时产生中断。

表 272. 看门狗定时器警告中断寄存器（WARNINT - 0x4000 4014）位描述

位	符号	描述	复位值
9:0	WARNINT	看门狗警告中断比较值。	0
31:10	-	保留，用户软件不应应对保留位写入 1。从保留位读取的值未定义。	不适用

17.7.7 看门狗定时器窗口寄存器

WDWINDOW 寄存器决定在执行看门狗喂狗时允许的 WDTV 最大值。如果喂狗有效序列在 WDTV 达到 WDWINDOW 中的值之前完成，则将发生看门狗事件。

WDWINDOW 会复位为 WDTV 最大的可能值，因此窗口不会生效。如果 WDWINDOW 的值小于 0x100，将无法成功进行看门狗喂狗。

表 273. 看门狗定时器窗口寄存器 (WINDOW - 0x4000 4018) 位描述

位	符号	描述	复位值
23:0	WINDOW	看门狗窗口值。	0xFF FFFF
31:24	-	保留，用户软件不应向保留位写入 1。从保留位读取的值未定义。	不适用

17.8 功能框图

看门狗的功能框图如下面的图 48 所示。功能框图中未显示同步逻辑 (PCLK - WDCLK)。

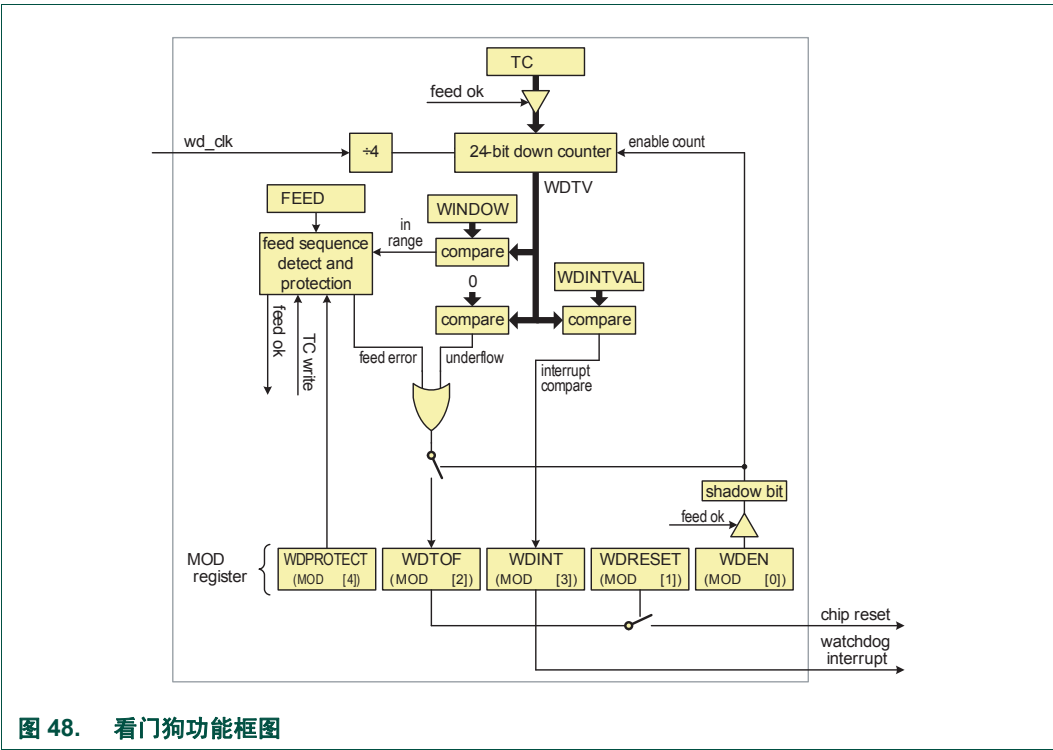
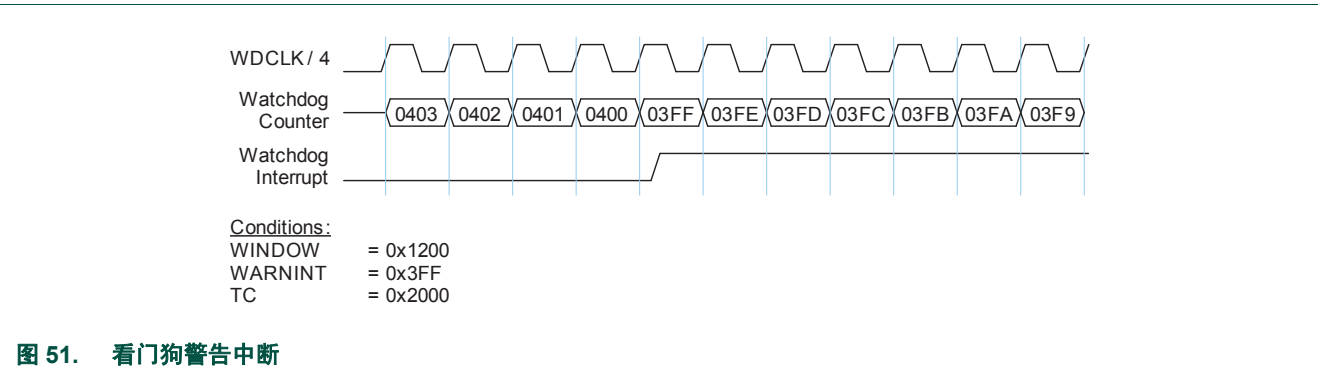
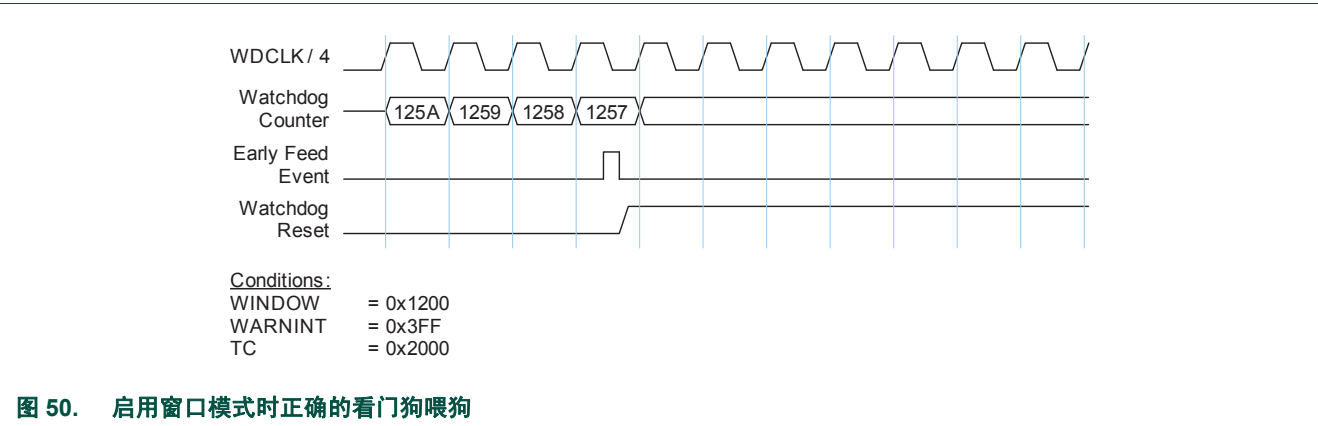
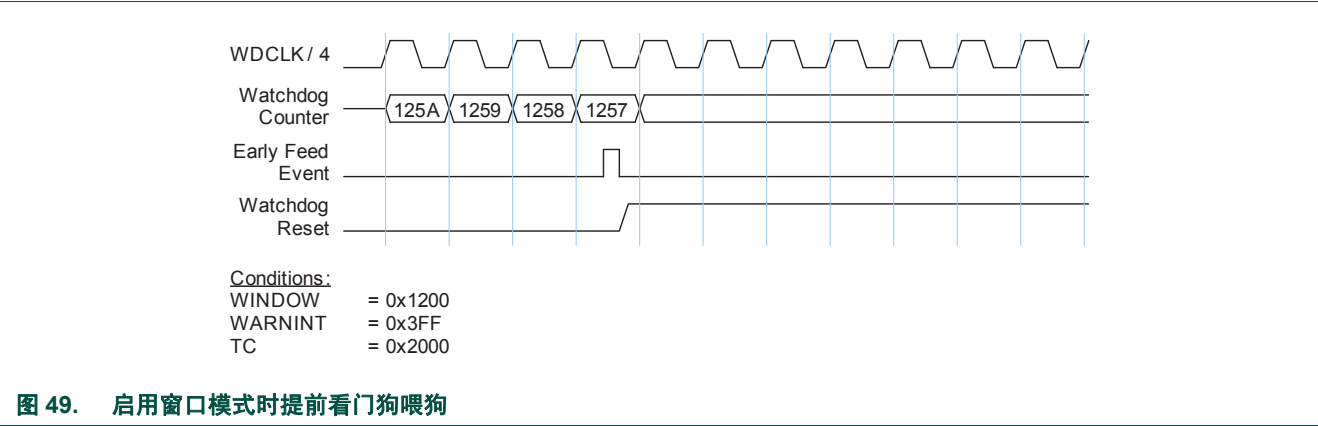


图 48. 看门狗功能框图

17.9 看门狗定时示例

下图描绘了看门狗定时器工作（如下面的图 49 所示）的几个方面。



18.1 本章导读

所有 LPC122x 部件均提供比较器。

18.2 基本配置

比较器模块的外设时钟由系统时钟提供，而系统时钟由 SYSAHBCLKDIV 寄存器控制（[表 21](#)）。可以通过系统 AHB 时钟控制寄存器位 20（[表 21](#)）和 PDRUNCFG 寄存器位 15（[表 50](#)）禁能比较器模块以降低能耗。

注：必须在 PDRUNCFG 寄存器（[表 50](#)）中使能 BOD，才能使用比较器。如果使用比较器从深度睡眠模式唤醒部件，必须通过覆盖 PDSLEEPCFG 寄存器的建议设置在该寄存器中为 BOD 和比较器供电。

18.3 特性

- 每个比较器六个可选外部源；可在正负比较器输入通道上完全配置。
- 两个比较器上均可选择 BOD 0.9 V 内部基准电压；可在正负比较器输入通道上配置。
- 两个比较器上均可选择 32 级电压阶梯的内部基准电压；可在正负比较器输入通道上配置。
- 可从外部引脚选择电压阶梯源电压，如果没有外部电源，则可从 3.3 V 电压轨选择。
- 对于只要求比较器功能的应用程序，电压阶梯可单独掉电。
- 张弛振荡器电路输出，用于反馈 555 定时器应用。
- 单独比较器中断连接到 I/O 引脚，通用中断连接到 NVIC。
- 边沿和电平比较器输出连接到两个允许在电平匹配有效时进行边沿节拍计数的定时器。

18.4 简介

片内集成的两个嵌入式比较器用于比较外部引脚电压电平和内部电压。每个比较器均可从多达六个外部引脚电压和两个内部基准电压之间选择。另外，如果两个比较器均要求相同电压，可选择四个外部输入电压以驱动两个比较器上通用的输入（见[图 52](#)）。

18.5 引脚描述

表 274. 比较器引脚描述

引脚	类型	描述
ACMP0_I[3:0]	输入	比较器 0 输入源（可为比较器 1 输入设置 ACMP0_I[0] 和 ACMP0_I[1]）
ACMP1_I[3:0]	输入	比较器 1 输入源（可为比较器 0 输入设置 ACMP1_I[0] 和 ACMP1_I[1]）
ACMP0_O	输出	比较器 0 输出
ACMP1_O	输出	比较器 1 输出
VREF_CMP	输入	32 级电压阶梯的外部基准电压源
ROSC	输出	张弛振荡器输出，适用于 555 定时器类应用

18.6 寄存器描述

表 275. 寄存器简介：比较器（基址 0x4005 4000）

名称	访问类型	地址偏移	描述	复位值
CMP	R/W	0x00	比较器控制寄存器	0
VLAD	R/W	0x04	电压阶梯寄存器	0

18.6.1 比较器控制寄存器

该寄存器使能比较器，配置中断，并控制两个比较器的输入多路复用器。

表 276. 比较器控制寄存器（CMP，地址 0x4005 4000）位描述

位	符号	值	描述	复位值
0	CMP0_EN		比较器 0 使能。	0
		0	比较器 0 被禁能。	
		1	比较器 0 被使能。	
1	CMP1_EN		比较器 1 使能。	0
		0	比较器 1 被禁能。	
		1	比较器 1 被使能。	
2	CMPIS		选择中断源。	0
		0	边沿触发。	
		1	电平触发。	
3	CMPIEV		选择边沿触发中断对高跃迁还是低跃迁有效。	0
		0	中断对下降沿有效。	
		1	中断对上升沿有效。	
4	CMPBE		选择边沿触发中断对两个边沿均有效。	0
		0	中断对两个边沿均无效。	
		1	中断对两个边沿均有效。	
5	CMPSR		选择张弛振荡器电路中两个比较器的方向。	0
6	CMPSA0		选择比较器 0 异步 / 同步输出。	0
		0	直接使用比较器输出。	
		1	比较器输出与总线时钟同步以输出到其他模块。	

表 276. 比较器控制寄存器（CMP，地址 0x4005 4000）位描述 *（续）*

位	符号	值	描述	复位值
7	CMPSA1		选择比较器 1 异步 / 同步输出。	0
		0	直接使用比较器输出。	
		1	比较器输出与总线时钟同步以输出到其他模块。	
10:8	CMP0_VP_CTRL		选择比较器 0，正压 (VP) 输入通道。	000
		0x0	电压阶梯输出	
		0x1	ACMP0_I0	
		0x2	ACMP0_I1	
		0x3	ACMP0_I2	
		0x4	ACMP0_I3	
		0x5	ACMP1_I0	
		0x6	ACMP1_I1	
		0x7	BOD 0.9 V 带隙	
13:11	CMP0_VM_CTRL		选择比较器 0，负压 (VM) 输入通道。	000
		0x0	电压阶梯输出	
		0x1	ACMP0_I0	
		0x2	ACMP0_I1	
		0x3	ACMP0_I2	
		0x4	ACMP0_I3	
		0x5	ACMP1_I0	
		0x6	ACMP1_I1	
		0x7	BOD 0.9 V 带隙	
16:14	CMP1_VP_CTRL		选择比较器 1，正压 (VP) 输入通道。	000
		0x0	电压阶梯输出	
		0x1	ACMP1_I0	
		0x2	ACMP1_I1	
		0x3	ACMP1_I2	
		0x4	ACMP1_I3	
		0x5	ACMP0_I0	
		0x6	ACMP0_I1	
		0x7	BOD 0.9 V 带隙	
19:17	CMP1_VM_CTRL		选择比较器 1，负压 (VM) 输入通道。	000
		0x0	电压阶梯输出	
		0x1	ACMP1_I0	
		0x2	ACMP1_I1	
		0x3	ACMP1_I2	
		0x4	ACMP1_I3	
		0x5	ACMP0_I0	
		0x6	ACMP0_I1	
		0x7	BOD 0.9 V 带隙	
20	INTCLR		中断清除位。将该位置 1 将清除比较器中断。	

表 276. 比较器控制寄存器（CMP，地址 0x4005 4000）位描述（续）

位	符号	值	描述	复位值
21	CMP0STAT		比较器 0 状态。该位反映比较器 0 输出状态。	
22	CMP1STAT		比较器 1 状态。该位反映比较器 1 输出状态。	
31:23	-		保留。	0

18.6.2 电压阶梯寄存器

该寄存器使能比较器基准输入的电压阶梯。基准输入 VREF_CMP 可在 32 个电平之中设置，从 V_{SS} 到 VREF_CMP = 3.3 V。

表 277. 电压阶梯寄存器（VLAD，地址 0x4005 4004）位描述

位	符号	值	描述	复位值
0	VLADEN		电压阶梯使能	0
		0	电压阶梯被禁能。	
		1	电压阶梯被使能。	
5:1	VSEL		电压阶梯值。基准电压 Vref 取决于该寄存器位 6 的设置（可以是 V _{DD(3V3)} 或引脚 VREF_CMP 上的电压）： 0000 = V _{SS} 0001 = 1 × Vref/31 0010 = 2 × Vref/31 ... 11111 = Vref	0
6	VLADREF		电压基准选择	1
		0	VREF_CMP 引脚	
		1	V _{DD(3V3)} 引脚	
31:7	-		未使用	0

18.7 功能说明

18.7.1 输入多路复用器

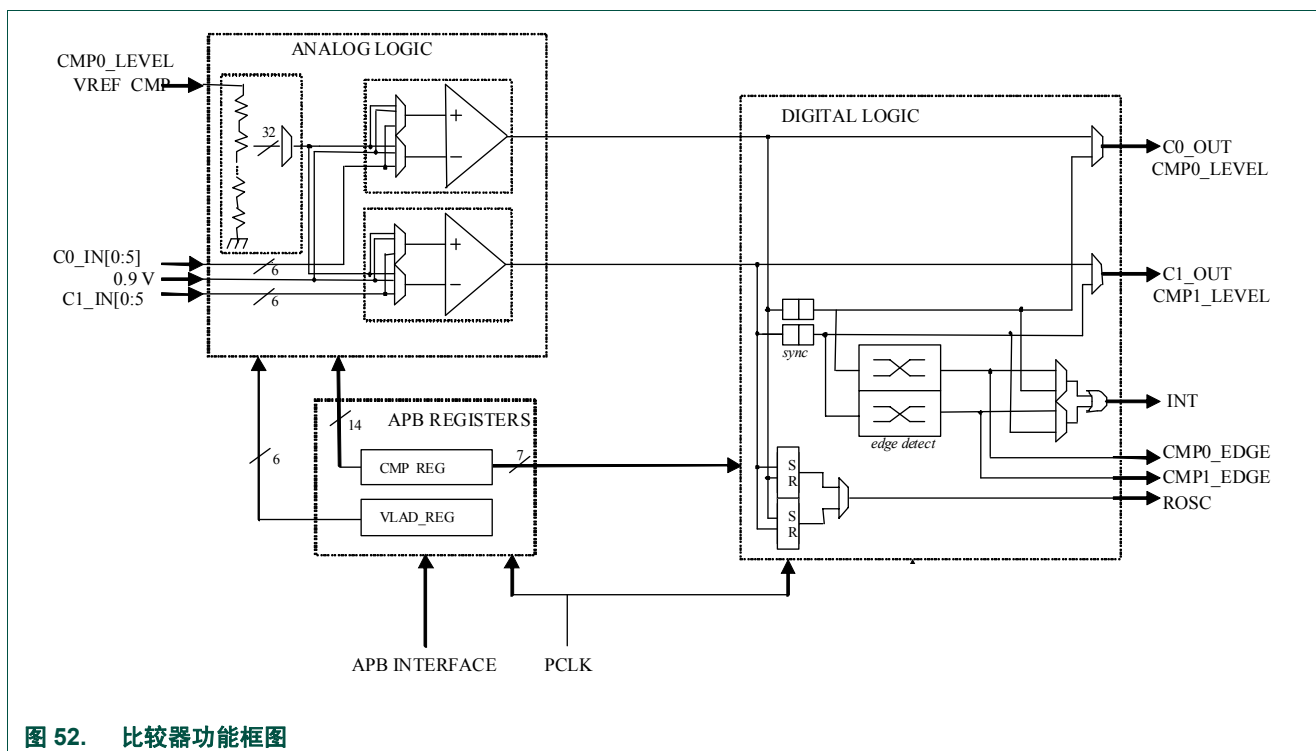


图 52. 比较器功能框图

两个比较器各有 8 个单独复用到各自正负输入的输入。每个比较器的多路复用器（正负输入）均由比较器寄存器 CMP 控制（见[图 52](#)和[表 276](#)）。

可以选择每个比较器的正负输入的位 0，其衍生自可编程电压阶梯输出。

可选择每个比较器的正负输入的位 7，其衍生自片内带隙电压。

比较器正负端的剩余 6 个比较器输入来自外部芯片 IO 引脚。但由于可用性限制，每个比较器只分配了四个 IO 引脚。每个比较器剩余的两个输入可根据需要从替代比较器输入中选择；可为 ACMP1_I4 和 ACMP1_I5 输入源分别选择 ACMP0_I0 和 ACMP0_I1，类似地，可为 ACMP0_I4 和 ACMP0_I5 输入源分别选择 ACMP1_I0 和 ACMP1_I1（见[图 53](#)）

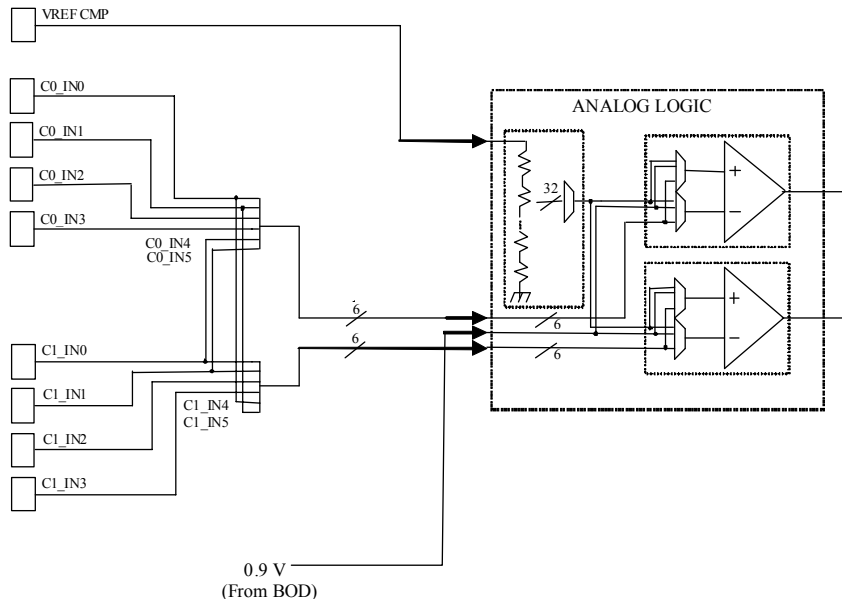


图 53. 比较器输入

18.7.2 基准电压

比较器可以使用两个基准电压：外部基准电压 **VREF_CMP** 和内部带隙电压 **BOD 0.9 V** 基准电压。外部基准电压可结合一个可编程电压阶梯使用，该电压阶梯可在 32 个电平之中设置，从 **VREF_CMP** IO 基准电压一直到 **VSS**。**VREF_CMP** 的最大电压为 3.3 V。

18.7.3 张弛振荡器

数字电路包括一个张弛振荡器，芯片的 **ROSC** 引脚可向外路由回到比较器输入，从而形成可用作 **555** 定时器的振荡。

555 支持三种工作模式：

1. 单稳态模式：在该模式下，**555** 定时器产生一个单脉冲。脉冲宽度可通过外部 **RC** 网络配置。
2. 非稳态 - 自由运行模式：在该模式下，**555** 定时器产生连续的矩形脉冲流。振荡频率由两个外部电阻器和一个电容器决定。
3. 双稳态模式或施密特触发器：如果未连接定时电容器引脚或未使用电容器，**555** 可作为触发器工作。

张弛振荡器应用包括精确定时、脉冲生成、顺序定时、延时生成、脉冲宽度生成、脉冲位置生成以及脉冲丢失检测。

18.7.4 中断

中断可选择为边沿式或电平式。如果选择电平式，则离开该模块的中断首先与外设时钟域同步，以防止异步中断路径导致 CPU 崩溃。如果选择边沿电平中断，可选择对高跃迁有效、对低跃迁有效还是对两个边沿均有效。通过 CPU 写高 CLINT 可清除中断。

比较器 0 和比较器 1 的组合“或”关系中断使用 CPU 路由到 NVIC 用于 ISR 目的。

18.7.5 比较器输出

两个比较器电平输出路由到外部引脚。电平和边沿比较器输出还内部连接到两个 16 位定时器的捕获输入。输出可选择为同步或异步模式（见[表 276](#)）：

- 对于到定时器捕获输入的內部连接，可选择同步或异步模式。
- 当异步输出路由到外部引脚时，配置好器件后，可不必定时 PCLK 来使用比较器以降低功耗。
- 当比较器配置为从深度睡眠模式唤醒部件时，必须选择异步模式。

此外，可通过比较器状态寄存器位观察每个比较器输出的状态

每个比较器的电平和边沿输出内部路由至两个 16 位定时器，如下所示：

- 16 位定时器 0 (CT16B0):
 - 捕获输入 3: 比较器 0 边沿
 - 捕获输入 2: 比较器 0 电平
- 16 位定时器 1 (CT16B1):
 - 捕获输入 3: 比较器 1 边沿
 - 捕获输入 2: 比较器 1 电平

如果选择了边沿捕获，该特性可对正边沿、负边沿或两个边沿（取决于比较器寄存器的配置）上的比较器输出跃迁进行节拍计数。如果选择了定时器电平捕获，计数器可在比较器输出处于特定状态时运行。

19.1 本章导读

所有 LPC12xx 部件均提供模数转换器 (ADC)。

19.2 基本配置

时钟和 ADC 电源由 SYSAHBCLKDIV 寄存器控制（见[表 20](#)）。可通过 AHBCLKCTRL 寄存器（[表 21](#)）中的第 14 位禁用该时钟以节省能耗。

有关如何连接到 DMA 的信息，请参见[章节 19.7.3](#)。

可通过 PDRUNCFG 寄存器在运行时使 ADC 进入掉电模式，请参见[表 50](#)。

19.3 特性

- 10 位逐次逼近型模数转换器 (ADC)。
- 在 8 个引脚间实现输入多路复用。
- 支持掉电模式。
- 测量范围：0 至 3 V。
- 257 kHz 的 10 位转换时间。
- 一个或多个输入的猝发转换模式。
- 输入引脚或定时器匹配信号过渡的可选转换。
- 每个 A/D 通道具有各自的结果寄存器以减少中断开销。

19.4 简介

系统时钟提供 A/D 转换器的基本计时。每个 ADC 都包含一个可编程分频器，以将该时钟按比例转成逐次逼近过程所需的 9 MHz（最大）时钟。一次完全准确的转换需要 35 个这样的时钟。

19.5 引脚描述

[表 278](#) 简要总结了每个 ADC 相关引脚。

表 278. ADC 引脚说明

引脚	类型	描述
AD[7:0]	输入	模拟输入。A/D 转换器元件可测量这些输入信号的电压。输入信号不得超过 V_{REF} （一般为 3.3 V）。
$V_{DD}(3V3)$	输入	V_{REF} ；基准电压。

必须通过 IOCON 寄存器选择 ADC 功能，以获得对监视引脚的准确电压读数。对于接受 ADC 输入的引脚，不可能选择数字功能而仍获得有效的 ADC 读数。只要在该引脚上选择数字功能，有一个内部电路就会将 ADC 硬件与相关引脚断开。

19.6 寄存器描述

表 279. 寄存器简介：ADC（基址 0x4002 0000）

名称	访问类型	地址偏移	描述	复位值
CR	R/W	0x000	A/D 控制寄存器。必须写入 CR 寄存器以选择操作模式，然后才会发生 A/D 转换。	0x0000 0000
GDR	R/W	0x004	A/D 全局数据寄存器。包含最近一次 A/D 转换结果。	不适用
-	-	0x008	保留	-
INTEN	R/W	0x00C	A/D 中断使能寄存器。该寄存器包含的使能位控制每个 A/D 通道的 DONE 标志是否用于产生 A/D 中断。	0x0000 0100
DR0	R/W	0x010	A/D 通道 0 数据寄存器。该寄存器包含通道 0 中最近一次完成的转换结果	不适用
DR1	R/W	0x014	A/D 通道 1 数据寄存器。该寄存器包含通道 1 中最近一次完成的转换结果。	不适用
DR2	R/W	0x018	A/D 通道 2 数据寄存器。该寄存器包含通道 2 中最近一次完成的转换结果。	不适用
DR3	R/W	0x01C	A/D 通道 3 数据寄存器。该寄存器包含通道 3 中最近一次完成的转换结果。	不适用
DR4	R/W	0x020	A/D 通道 4 数据寄存器。该寄存器包含通道 4 中最近一次完成的转换结果。	不适用
DR5	R/W	0x024	A/D 通道 5 数据寄存器。该寄存器包含通道 5 中最近一次完成的转换结果。	不适用
DR6	R/W	0x028	A/D 通道 6 数据寄存器。该寄存器包含通道 6 中最近一次完成的转换结果。	不适用
DR7	R/W	0x02C	A/D 通道 7 数据寄存器。该寄存器包含通道 7 中最近一次完成的转换结果。	不适用
STAT	RO	0x030	A/D 状态寄存器。该寄存器包含所有 A/D 通道的 DONE 和 OVERRUN 标志，以及 A/D 中断标志。	0
TRM	R/W	0x034	A/D 微调寄存器	0

19.6.1 A/D 控制寄存器

A/D 控制寄存器提供用于选择要转换的 A/D 通道、A/D 定时、A/D 模式以及 A/D 起动触发器的位。

表 280. A/D 控制寄存器（CR - 地址 0x4002 0000）位描述

位	符号	值	描述	复位值
7:0	SEL		选择要采样和转换的 AD7:0 引脚。对于 ADC，位 0 选择引脚 AD0，位 7 选择引脚 AD7。在软件控制模式下，这些位中只能有一个是 1。在硬件扫描模式下，任何包含 1 到 8 个“1”的值。所有零相当于 0x01。	0
15:8	CLKDIV		将 APB 时钟 (PCLK) 进行分频（CLKDIV 值 +1）得到 A/D 转换器的时钟，该时钟必须小于或等于 9 MHz。通常软件应编程该域中的最小值来得到 9 MHz 或稍低于 9 MHz 的时钟，但某些情况下（例如高阻抗模拟电源）可能需要更低的时钟。	0
16	BURST		猝发模式控制。	0
		0	转换由软件控制，需要 36 个时钟才能完成。	
		1	AD 转换器重复进行转换，最高可达 250 kHz，必要时将扫描 SEL 字段中的“1”选择的引脚。起动后进行的第一次转换对应于 SEL 字段中的最低有效位 1，然后是（适用情况下）更高编号的 1 位（引脚）。清零该位可终止重复转换，但是该位清零时将完成正在进行的转换。	
注意事项：当 BURST = 1 时 START 位必须为 000，否则转换无法启动。				

表 280. A/D 控制寄存器（CR - 地址 0x4002 0000）位描述

位	符号	值	描述	复位值
23:17	-		保留。这些位始终读为零。	0
26:24	START		转换启动控制。当 BURST 位为 0 时，这些位控制 A/D 转换是否启动及何时启动。	0
		0x0	不启动（PDN 清零时使用该值）。	
		0x1	立即启动转换。	
		0x2	当位 27 选择的边沿出现在 PIO0_2/SSEL/CT16B0_CAP0 时启动转换。	
		0x3	当位 27 选择的边沿出现在 PIO1_5/DIR/CT32B0_CAP0 时启动转换。	
		0x4	当位 27 选择的边沿出现在 CT32B0_MAT0 时启动转换。	
		0x5	当位 27 选择的边沿出现在 CT32B0_MAT1 时启动转换。	
		0x6	当位 27 选择的边沿出现在 CT16B0_MAT0 时启动转换。	
		0x7	当位 27 选择的边沿出现在 CT16B0_MAT1 时启动转换。	
27	EDGE		边沿控制。该位只有在 START 字段包含 010-111 时有效。	0
		1	在所选 CAP/MAT 信号的下降沿启动转换。	
		0	在所选 CAP/MAT 信号的上升沿启动转换。	
31:28	-		保留。这些位始终读为零。	0

19.6.2 A/D 全局数据寄存器

A/D 全局数据寄存器包含最近一次 A/D 转换的结果。其中包含数据、DONE 和 Overrun 标志以及与数据相关的 A/D 通道数。

表 281. A/D 全局数据寄存器（GDR - 地址 0x4002 0004）位描述

位	符号	描述	复位值
5:0	-	保留。这些位始终读为零。	0
15:6	RESULT	当 DONE 为 1 时，该字段包含的是一个二进制小数，表示的是 SEL 字段所选定的 ADn 引脚的电压除以 V _{DD(3V3)} 引脚上的电压：V/V _{REF} 。该字段为 0 表示 ADn 引脚处的电压小于、等于或接近于 V _{SS} ，而 0x3FF 表明 ADn 引脚处的电压接近于、等于或大于 V _{REF} 。	X
23:16	-	保留。这些位始终读为零。	0
26:24	CHN	这些位包含 RESULT 位的转换通道。	X
29:27	-	保留。这些位始终读为零。	0
30	OVERRUN	在猝发模式下，如果在产生 RESULT 位结果转换之前一个或多个转换结果丢失或被覆盖，则该位置 1。	0
31	DONE	A/D 转换结束时该位置 1。该位在读取该寄存器和写 ADCR 时清零。如果在转换过程中写 ADCR，则该位置位并启动新的转换。	0

19.6.3 A/D 中断使能寄存器

该寄存器用来控制转换完成时哪些 A/D 通道产生中断。例如，可能需要通过不断在某些 A/D 通道上执行转换来监视传感器。最近的转换结果可根据需要随时由应用程序读出。在这种情况下，某些 A/D 通道转换结束时都不使用中断方式。

表 282. A/D 中断使能寄存器（INTEN - 地址 0x4002 000C）位描述

位	符号	描述	复位值
7:0	ADINTEN	这些位用来控制哪些 A/D 通道在转换结束时产生中断。当位 0 为 1 时，A/D 通道 0 转换结束将产生中断，当位 1 为 1 时，A/D 通道 1 转换结束将产生中断，以此类推。	0x00
8	ADGINTEN	为 1 时，使能 ADDR 中的全局 DONE 标志产生中断。为 0 时，只有个别由 ADINTEN 7:0 使能的 A/D 通道产生中断。	1
31:9	-	保留。这些位始终读为零。	0

19.6.4 A/D 数据寄存器

A/D 转换完成时，A/D 数据寄存器保存转换结果，还包含指示转换结束及转换溢出发生的标志。

表 283. A/D 数据寄存器（DR0 到 DR7 - 地址 0x4002 0010 到 0x4002 002C）位描述

位	符号	描述	复位值
5:0	-	保留。这些位始终读为零。	0
15:6	RESULT	当 DONE 为 1 时，该字段包含的是一个二进制小数，表示的是 SEL X 字段所选定的 ADn 引脚的电压除以 V _{DD(3V3)} 引脚上的电压 V/V _{REF} 。该字段为 0 表示 ADn 引脚处的电压小于、等于或接近于 V _{SSA} ，而 0x3FF 表明 ADn 引脚处的电压接近于、等于或大于 V _{REF} 。	X
29:16	-	保留。这些位始终读为零。	0
30	OVERRUN	在猝发模式下，如果在产生 RESULT 位结果转换之前一个或多个转换结果丢失或被覆盖，则该位置 1。读取该寄存器可清除此位。	0
31	DONE	A/D 转换结束时该位置 1。读取该寄存器时会清除此位。	0

19.6.5 A/D 状态寄存器

A/D 状态寄存器允许同时检查所有 A/D 通道的状态。每个 A/D 通道的 ADDRn 寄存器中的 DONE 和 OVERRUN 标志都反映在 ADSTAT 中。在 ADSTAT 中还可以找到中断标志（所有 DONE 标志的逻辑“或”）。

表 284. A/D 状态寄存器（STAT - 地址 0x4002 0030）位描述

位	符号	描述	复位值
7:0	DONE	这些位反映了每个 A/D 通道的结果寄存器中的 DONE 状态标志。	0
15:8	OVERRUN	这些位反映了每个 A/D 通道的结果寄存器中的 OVERRRUN 状态标志。读取 ADSTAT 允许同时检查所有 A/D 通道的状态。	0
16	ADINT	该位为 A/D 中断标志。当任何一条 A/D 通道的 Done 标志置 1 且使能 A/D 产生中断（通过 ADINTEN 寄存器设置）时，该位置 1。	0
31:17	-	保留。这些位始终读为零。	0

19.6.6 A/D 微调寄存器

该寄存器将由引导代码在起动机时设置。它包含 ADC 的微调值。ADC 的偏移微调值可由用户覆盖。

表 285. A/D 微调寄存器（TRM - 地址 0x4002 4034）位描述

位	符号	描述	复位值
3:0	-	保留。这些位始终读为零。	0
7:4	ADCOFFS	ADC 操作的偏移微调位。由引导代码初始化。可由用户覆盖。	0
31:8	-	保留。这些位始终读为零。	0

19.7 操作

一旦启动 ADC 转换，便无法中断。在上一转换仍在进行期间，启动新转换或新边沿触发事件的新软件写入将被忽略。

19.7.1 硬件触发转换

如果 ADCR 中的 BURST 位为 0，且 START 字段包含 010-111，ADC 将在选定引脚或定时器匹配信号发生过渡时启动转换。选项包括在 4 个匹配信号中任意一个的特定边沿转换，或在 2 个捕获 / 匹配引脚中任意一个的特定边沿转换。边沿检测逻辑中使用选定焊盘的引脚状态或选定匹配信号 - XORed，ADCR 位 27。

19.7.2 中断

当 DONE 位为 1 时，已确认向 NVIC 发出了中断请求。软件可以使用 NVIC 中 A/D 转换器的中断使能位控制此确认是否会引起中断。读取 ADDR 时 DONE 将被取消。

19.7.3 DMA 控制

DMA 传输请求从 ADC 中断请求线路生成。要生成 DMA 传输，必须满足与生成中断相同的条件（参见[章节 19.6.3](#)和[章节 19.7.2](#)）。

注：如果使用 DMA，则必须在 NVIC 中禁用 ADC 中断。

对于 DMA 传输，传输大小可设为 DMA 通道控制结构（参见[表 347](#)）中的一个或设为已转换的 ADC 通道数。DMA 传输大小决定何时生成 DMA 中断。

20.1 本章导读

各种 Flash 配置请参见[表 286](#)。

表 286. LPC122x 的 Flash 配置

产品型号	Flash
LPC1227	
LPC12D27FBD100/301	128 kB
LPC1227FBD64/301	128 kB
LPC1227FBD48/301	128 kB
LPC1226	
LPC1226FBD64/301	96 kB
LPC1226FBD48/301	96 kB
LPC1225	
LPC1225FBD64/321	80 kB
LPC1225FBD64/301	64 kB
LPC1225FBD48/321	80 kB
LPC1225FBD48/301	64 kB
LPC1224	
LPC1224FBD64/121	48 kB
LPC1224FBD64/101	32 kB
LPC1224FBD48/121	48 kB
LPC1224FBD48/101	32 kB

20.2 特性

- 在系统编程：在系统编程 (ISP) 是使用 **bootloader** 软件和 **UART** 串行端口对片内 **Flash** 存储器的编程或重新编程。当器件位于最终用户端时，可执行此操作。
- 在应用编程：在应用编程 (IAP) 是指在终端用户应用代码的引导下，对片内 **Flash** 存储器进行擦除和写入操作。

20.3 bootloader

bootloader 控制着复位后的初始操作，并提供了对 **Flash** 存储器编程的方法。这可以是对空白器件的初始编程、对之前已编程的器件进行的擦除和重新编程，或者是通过运行的系统中的应用程序对 **Flash** 存储器进行的编程。

20.4 应用程序

bootloader 提供了在系统与在应用编程的接口，用于对片内 **Flash** 存储器进行编程。

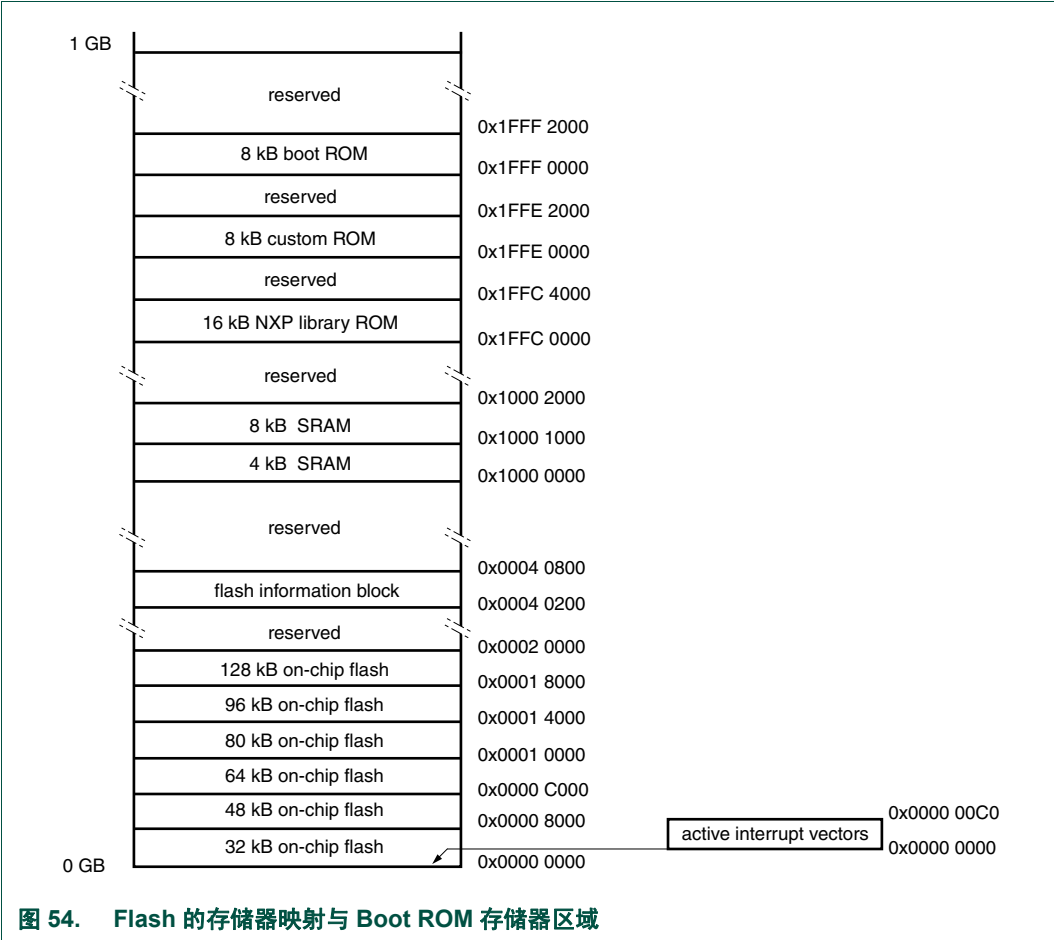
20.5 描述

每当部件上电或复位时，**bootloader** 代码即被执行一次。**bootloader** 还可以执行 ISP 命令处理程序或用户应用程序代码。复位后引脚 **PIO0_12** 的低电平被视为启动 ISP 命令处理程序的外部硬件请求。假设在 **RESET** 引脚上产生上升沿时，电源引脚处于标称的电平，那么在最多 3 ms 后，会对 **PIO0_12** 引脚信号进行采样并确定是继续用户代码还是产生 ISP 处理程序。如果对 **PIO0_12** 引脚采样的结果为低电平，同时看门狗溢出标志被置位，那么外部硬件启动 ISP 命令处理程序的请求将被忽略。如果不存在执行 ISP 命令处理程序的请求（**PIO0_12** 在复位后的采样结果为高电平），则会搜寻有效的用户程序。如果找到了有效的用户程序，那么执行控制将由此程序接管。如果未找到有效的用户程序，则会调用自动波特率程序。

用作 ISP 硬件请求的 **PIO0_12** 引脚需要特别注意。由于 **PIO0_12** 在复位后处于高阻态，用户需要提供外部硬件（上拉电阻器或其他器件）使引脚处于定义状态。否则，可能意外进入 ISP 模式。

20.5.1 复位后的存储器映射

引导模块的大小为 8 kB。引导模块在存储器区域中的起始地址为 0x1FFF 0000。**bootloader** 被设计为从该存储器区域运行，但 ISP 和 IAP 软件都会占用部分的片内 RAM。本章后面的部分描述了 RAM 的使用情况。复位后，驻留在片内 Flash 存储器的引导模块中的中断向量也变为有效。也就是说，引导模块底部的 512 个字节也将出现在起始地址为 0x0000 0000 的存储器区域中。



20.5.2 Flash 扇区

LPC122x Flash 模块分成大小为 512 字节的页。页是可擦除的最小 Flash 区域。此外，虚拟扇区用于软件兼容现有的 ISP 和 IAP 命令。Flash 模块分为 32 个虚拟扇区（见表 287）。每个虚拟扇区的大小为 4 kB。一个虚拟扇区包含 8 个 512 字节的页。擦除操作可在虚拟扇区或页边界进行。

表 287. LPC122x 的 Flash 配置

页编号 (每个扇区 8 页)	虚拟扇区 编号 (每个扇 区 4 kB)	地址范围	Flash 大小					
			32 kB	48 kB	64 kB	80 kB	96 kB	128 kB
0 - 63	0 - 7	0x0000 0000 - 0x0000 7FFF	是	是	是	是	是	是
64 - 95	8 - 11	0x0000 8000 - 0x0000 BFFF	-	是	是	是	是	是
96 - 127	12 - 15	0x0000 8000 - 0x0000 FFFF	-	-	是	是	是	是
128 - 149	16 - 19	0x0001 0000 - 0x0001 3FFF	-	-	-	是	是	是
150 - 181	20 - 23	0x0001 4000 - 0x0001 7FFF	-	-	-	-	是	是
182 - 255	24 - 31	0x0001 8000 - 0x0001 FFFF	-	-	-	-	-	是

20.5.2.1 信息块

Flash 存储器区域还包含 Flash 信息块, 起始地址为 0x0004 0200, 总大小 3 页 (见 [图 54](#))。该 Flash 区域可供用户存储变量、配置参数以及用户定义的其他数据。

CPU 在信息块地址的读访问将返回该位置包含的数据。信息块中的页可通过发出 IAP 命令 “Erase info page” (擦除信息页) 来擦除, 这个命令专用于信息块。然后可使用 IAP “Copy RAM to flash” (将 RAM 内容复制到 Flash) 命令向信息块写入新用户数据。

20.5.2.2 Flash 模块的擦除操作

ISP 和 IAP 命令允许准备一个或一个以上扇区用于擦除操作以及擦除这些扇区。

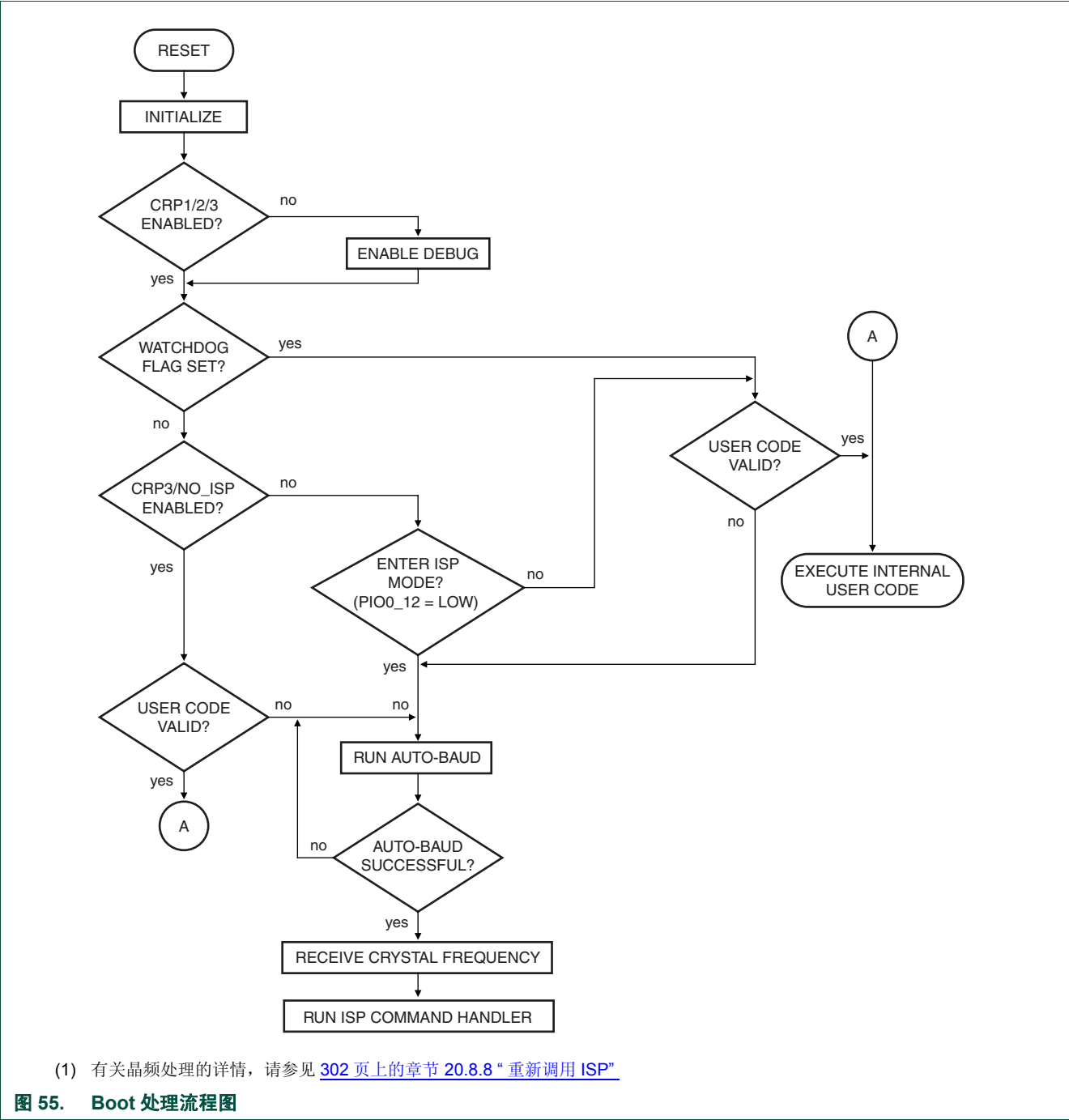
此外, IAP 命令包括一个页擦除命令, 允许擦除用户 Flash 存储器中的多个页而不论扇区边界如何。

在 LPC122x 上, 擦除时间随着要擦除的页数或扇区数的增加而增加。如果选择擦除 Flash 区域的所有扇区, ISP 和 IAP 程序将自动使用更有效的 “批量擦除” 程序, 该程序能显著缩短擦除时间。有关 Flash 定时参数的详情, 请参见 《LPC122x 数据表》。

20.5.2.3 Flash 模块的写操作

使用 ISP 和 IAP “Copy RAM to flash” (将 RAM 内容复制到 Flash) 命令的写入 Flash 操作允许用户代码写入 Flash。最小写大小为 4 字节, 相当于一个 Flash 字。总编程时间与写入的 Flash 字数成正比。但是, 一次写入一整行 32 个 Flash 字或 128 个字节更高效。“Copy RAM to flash” (将 RAM 内容复制到 Flash) 命令自动检测行对齐地址, 并使用更高效的写入程序来缩短编程时间。

20.5.3 Boot 处理流程图



(1) 有关晶频处理的详情，请参见 [302 页上的章节 20.8.8 “重新调用 ISP”](#)

图 55. Boot 处理流程图

20.5.4 有效用户代码的判定标准

有效用户代码的判定标准：保留的 Cortex-M0 异常向量位置 7（在向量表中的偏移为 0x0000 001C）应当包含表项 0~6 的校验和的 2 的补码。这可以使前 8 个表项的校验和为 0。bootloader 代码校验 Flash 扇区 0 中的前 8 个位置。如果结果为 0，则执行控制权便转移给用户代码。

如果签名无效,则自动波特率程序通过串行端口 0 与主机进行同步。主机应当发送一个同步字符 '?'(0x3F) 并等待响应。主机的串行端口设置应为 8 个数据位、1 个停止位且无奇偶校验。自动波特率程序根据其自身的频率测量接收到的同步字符的位时间并对串行端口波特率发生器进行编程。它还向主机发送一个 ASCII 字符串 (“Synchronized<CR><LF>”),作为响应,主机应当发送同一字符串 (“Synchronized<CR><LF>”)。自动波特率程序通过观察接收到的字符来验证同步。如果通过验证,则向主机发送 “OK<CR><LF>” 字符串。主机应当通过发送正在运行部分的晶频(单位为 KHz)进行响应。例如,如果运行在 10MHz,主机的响应当为 “10000<CR><LF>”。在接收到晶频后再向主机发送 “OK<CR><LF>”。如果同步验证没有通过,则自动波特率程序再次等待一个同步字符。在用户调用 ISP 的情况下要使自动波特率正确工作, CCLK 频率应当大于或等于 10 MHz。

在接收到晶频后,器件初始化并调用 ISP 命令处理程序。出于安全性的考虑,在执行 Flash 擦除 / 写操作命令和 “Go” 命令之前必须执行 “Unlock” (解锁) 命令。执行其他命令时不需要执行解锁命令。每个 ISP 会话都要执行一次 “Unlock” (解锁) 命令。解锁命令在 [290 页上的章节 20.7 “ISP 命令”](#) 中进行了说明。

20.5.5 通信协议

所有 ISP 命令都应以单个 ASCII 字符串形式发送。字符串应当以回车 (CR) 和 / 或换行 (LF) 控制字符作为终止符。多余的 <CR> 和 <LF> 将被忽略。所有 ISP 响应都是以 <CR><LF> 终止的 ASCII 字符串形式发送。数据是以 UU 编码格式发送和接收。

20.5.5.1 ISP 命令格式

“命令 参数 _0 参数 _1 ... 参数 _n<CR><LF>” “数据” (只适用于写命令)。

20.5.5.2 ISP 响应格式

“返回 _ 代码 <CR><LF> 响应 _0<CR><LF> 响应 _1<CR><LF> ... 响应 _n<CR><LF>”
“数据” (只适用于读命令)。

20.5.5.3 ISP 数据格式

数据流采用 UU 编码格式。UU 编码算法将 3 字节的二进制数据转换成 4 字节可打印的 ASCII 字符集,该编码的效率高于 Hex 格式 (Hex 格式将 1 字节的二进制数据转换成 2 字节的 ASCII Hex 数据)。发送器应当在发送 20 个 UU 编码行之后发送校验和。任何 UU 编码行的长度都不应超过 61 个字符 (字节),也就是说它可以容纳 45 个数据字节。接收器应当将其与接收字节的校验和进行比较。如果校验和匹配,接收器应当响应 “OK<CR><LF>” 来继续下一次发送。如果校验和不匹配,则接收器应当响应 “RESEND<CR><LF>”。作为响应,发送器应当重新发送字节。

UU 编码的描述可以在 wotsit 的网页上找到。

20.5.5.4 ISP 流量控制

软件 XON/XOFF 流程控制机制可防止缓冲区溢出时的数据丢失。当数据快速到达时,发送 ASCII 控制字符 DC3 (停止) 使数据流停止。发送 ASCII 控制字符 DC1 (启动) 恢复数据流。主机也应支持相同的流控制机制。

20.5.5.5 ISP 命令中止

命令可通过发送 ASCII 控制字符 “ESC” 来中止。该特性在 [章节 20.7](#) 中并没有作为一个命令。一旦接收到退出代码,ISP 命令处理程序就会等待新的命令。

20.5.5.6 ISP 过程中的中断

在任何复位后，位于 Flash 引导块内的引导块中断向量都有效。

20.5.5.7 IAP 过程中的中断

在擦除 / 写操作过程中，片内 Flash 存储器不可访问。只有当用户应用程序代码启动执行时，用户 Flash 区的中断向量才有效。在调用 Flash 擦除 / 写 IAP 之前，用户应当禁止中断或确保用户中断向量在 RAM 中有效且中断处理程序位于 RAM 中。IAP 代码不能使用或禁止中断。

20.5.5.8 ISP 命令处理程序使用的 RAM

ISP 命令使用片内地址 0x1000 017C 到 0x1000 025B 范围内的 RAM。用户可以使用该区域，但是在复位时内容可能会丢失。Flash 编程命令使用片内 RAM 最顶端的 32 字节。堆栈位于 RAM 顶端减去 32 字节。可使用的最大堆栈为 256 字节，堆栈是向下递增的。

20.5.5.9 IAP 命令处理程序使用的 RAM

Flash 编程命令使用片内 RAM 最顶端的 32 字节。用户分配的堆栈空间中可使用的最大堆栈为 128 字节，堆栈是向下递增的。

20.6 代码读保护 (CRP)

代码读保护是允许用户在系统中通过使能不同的安全级别来限制对片内 Flash 的访问和 ISP 的使用的一种机制。需要时，可通过在 Flash 地址单元 0x0000 02FC 编程特定的格式来调用 CRP。IAP 命令不受代码读保护的影响。

注意事项：CRP 所作出的任何改变只有在器件经过一个电源周期之后才会生效。

表 288. 代码读保护选项

名称	在 0x0000 02FC 处编程的格式	描述
NO_ISP	0x4E69 7370	阻止对 PIO0_12 引脚进行采样而进入 ISP 模式。PIO0_12 引脚用作其他用途。Flash 内容不受保护，可通过 SWD 接口重新对 Flash 编程。
CRP1	0x12345678	<p>通过 SWD 引脚访问芯片被禁能。该模式允许利用下列 ISP 命令和限制条件来进行部分的 Flash 更新：</p> <ul style="list-style-type: none">写入 RAM 命令不能访问在 0x1000 0300 以下的 RAM。“将 RAM 内容复制到 Flash”命令不能写入扇区 0。只有在选择擦除所有扇区时，擦除命令才能擦除扇区 0。比较命令被禁能。读取存储器命令被禁能。 <p>当需要 CRP 且要更新 Flash 字段时可使用该模式，但不能擦除所有扇区。由于在 Flash 部分更新的情况下比较命令被禁能，因此辅助加载程序应执行校验和机制来验证 Flash 的完整性。</p>
CRP2	0x87654321	<p>通过 SWD 引脚访问芯片被禁能。下列 ISP 命令被禁能：</p> <ul style="list-style-type: none">读存储器写入 RAM运行将 RAM 内容复制到 Flash比较 <p>使能 CRP2 时，ISP 擦除命令仅允许擦除所有用户扇区的内容。</p>
CRP3	0x43218765	<p>通过 SWD 引脚访问芯片被禁能。如果 Flash 扇区 0 中存在有效用户代码，则禁止通过拉低 PIO0_12 来进入 ISP。</p> <p>该模式有效禁止了通过 PIO0_12 引脚来强行进入 ISP 的行为。用户的应用程序可决定是调用 IAP 来进行 Flash 更新还是通过 UART 重新调用 ISP 命令来进行 Flash 更新。</p> <p>注意：如果选择了 CRP3，此后该器件将无法执行厂商测试。</p>

表 289. 代码读保护硬件 / 软件的相互作用

CRP 选项	用户代码是否有效	复位时 PIO0_12 引脚电平	SWD 是否使能	LPC122x 是否进入 ISP 模式	ISP 模式下 Flash 是否部分更新
否	否	x	是	是	是
否	是	高	是	否	不适用
否	是	低	是	是	是
CRP1	是	高	否	否	不适用
CRP1	是	低	否	是	是
CRP2	是	高	否	否	不适用
CRP2	是	低	否	是	否
CRP3	是	x	否	否	不适用
CRP1	否	x	否	是	是
CRP2	否	x	否	是	否
CRP3	否	x	否	是	否

表 290. 不同 CRP 等级下允许使用的 ISP 命令

ISP 命令	CRP1	CRP2	CRP3（不允许在 ISP 模式下进入）
解锁	是	是	不适用
设置波特率	是	是	不适用
回应	是	是	不适用
写入 RAM	是；只在 0x1000 0300 以上	否	不适用
读存储器	否	否	不适用
准备写操作的扇区	是	是	不适用
将RAM内容复制到Flash	是；不到扇区 0	否	不适用
运行	否	否	不适用
擦除扇区	是；只有在擦除所有扇区时才能擦除扇区 0。	是；只有所有扇区	不适用
扇区查空	否	否	不适用
读器件 ID	是	是	不适用
读 Boot 代码版本	是	是	不适用
比较	否	否	不适用
读 UID	是	是	不适用

若 CRP 模式使能，并且允许通过 ISP 访问芯片，不支持或受限制的 ISP 命令将被终止执行并同时返回代码 CODE_READ_PROTECTION_ENABLED。

20.6.1 ISP 入口保护

除了3种CRP模式外，用户可防止对PIO0_12引脚采样而进入ISP模式，从而释放PIO0_12引脚，使其用于其它方面。这称为 NO_ISP 模式。可通过对 0x0000 02FC 单元的 0x4E69 7370 编程进入 NO_ISP 模式。

在NO_ISP模式下，可通过SWD接口对Flash重新编程。在该模式下，Flash内容不受保护。

20.7 ISP 命令

下面的命令是 ISP 命令处理程序所接受的命令。每个命令都支持具体的状态代码。当接收到未定义命令时，命令处理程序会发送返回代码 INVALID_COMMAND。命令和返回代码为 ASCII 格式。

只有当接收到的 ISP 命令执行完毕时，ISP 命令处理程序才会发送 CMD_SUCCESS，这时主机才能发送新的 ISP 命令。但 “ 设置波特率 ”、 “ 写 RAM ”、 “ 读存储器 ” 和 “ 运行 ” 命令除外。

表 291. ISP 命令总结

ISP 命令	用法	描述
解锁	U < 解锁代码 >	表 292
设置波特率	B < 波特率 > < 停止位 >	表 293
回应	A < 设定 >	表 294
写入 RAM	W < 起始地址 > < 字节数 >	表 295
读存储器	W < 地址 > < 字节数 >	表 296
准备写操作的扇区	P < 起始扇区号 > < 结束扇区号 >	表 297

表 291. ISP 命令总结 (续)

ISP 命令	用法	描述
将RAM内容复制到Flash	C <Flash 地址> <RAM 地址> < 字节数>	表 298
运行	G < 地址> < 模式>	表 299
擦除扇区	E < 起始扇区号> < 结束扇区号>	表 300
扇区查空	I < 起始扇区号> < 结束扇区号>	表 301
读器件 ID	J	表 302
读 Boot 代码版本	K	表 304
比较	M < 地址 1> < 地址 2> < 字节数>	表 305
读 UID	N	表 306

20.7.1 解锁 < 解锁代码 >

表 292. ISP 解锁命令

命令	U
输入	解锁代码：23130（十进制）
返回代码	CMD_SUCCESS INVALID_CODE PARAM_ERROR
描述	该命令用于解锁 Flash 写、擦除和运行命令。
示例	“U 23130<CR><LF>” 解锁 Flash 写 / 擦除和运行命令。

20.7.2 设置波特率 < 波特率> < 停止位>

表 293. ISP 设置波特率命令

命令	B
输入	波特率：9600 19200 38400 57600 115200 停止位：1 2
返回代码	CMD_SUCCESS INVALID_BAUD_RATE INVALID_STOP_BIT PARAM_ERROR
描述	该命令用于改变波特率。新的波特率在命令处理程序发送 CMD_SUCCESS 返回代码之后生效。
示例	“B 57600 1<CR><LF>” 设置串口波特率 57600bps 和 1 个停止位。

20.7.3 回应 < 设定 >

表 294. ISP 回应命令

命令	A
输入	设定：打开 = 1 关闭 = 0
返回代码	CMD_SUCCESS PARAM_ERROR
描述	回应命令的默认设定是打开。当打开时，ISP 命令处理程序将接收到的串行数据发送回主机。
示例	“A 0<CR><LF>” 将回应关闭。

20.7.4 写 RAM < 起始地址 > < 字节数 >

主机应只在接收到 CMD_SUCCESS 返回代码后才发送数据。当发送完 20 个 UU 编码行之后主机应当发送校验和。校验和是通过把原始数据（UU 编码前）字节的值叠加起来产生的，当发送完 20 个 UU 编码行后该值会重置。任何 UU 编码行的长度不应超过 61 个字符（字节），也就是说，它可以携带 45 个数据字节。当数据少于 20 个 UU 编码行时，校验和应当按照实际发送的字节数进行计算。ISP 命令处理程序将它与接收字节的校验和进行比较。如果校验和匹配，那么 ISP 命令处理程序响应“OK<CR><LF>”来继续下一次发送。如果校验和不匹配，那么 ISP 命令处理程序响应“RESEND<CR><LF>”。作为响应，主机应当重新发送字节。

表 295. ISP 写 RAM 命令

命令	W
输入	起始地址: 要写入数据字节的 RAM 地址。该地址应当以字为边界。 字节数: 写入的字节数。计数值应当为 4 的倍数
返回代码	CMD_SUCCESS ADDR_ERROR（地址不是以字为边界） ADDR_NOT_MAPPED COUNT_ERROR（字节计数值不是 4 的倍数） PARAM_ERROR CODE_READ_PROTECTION_ENABLED
描述	该命令用于将数据下载到 RAM。数据应当为 UU 编码格式。当代码读保护使能时该命令被禁止。
示例	“W 268436224 4<CR><LF>” 向地址 0x1000 0300 写入 4 个字节数据。

20.7.5 读存储器 < 地址 > < 字节数 >

数据流之后是命令成功返回代码。发送完 20 个 UU 编码行之后发送校验和。校验和是通过把原始数据（UU 编码前）字节的值叠加起来产生的，当发送完 20 个 UU 编码行后该值会重置。任何 UU 编码行的长度不应超过 61 个字符（字节），也就是说，它可以携带 45 个数据字节。当数据少于 20 个 UU 编码行时，校验和按照实际发送的字节数进行计算。主机应当将它与接收字节的校验和进行比较。如果校验和匹配，那么主机应当响应“OK<CR><LF>”来继续下一次发送。如果校验和不匹配，主机应当响应“RESEND<CR><LF>”。作为响应，ISP 命令处理程序重新发送字节。

表 296. ISP 读存储器命令

命令	R
输入	起始地址: 被读出数据字节的地址。该地址应当以字为边界。 字节数: 读出的字节数。计数值应当为 4 的倍数。
返回代码	CMD_SUCCESS，后面是 < 实际数据（UU 编码） > ADDR_ERROR（地址不是以字为边界） ADDR_NOT_MAPPED COUNT_ERROR（字节计数值不是 4 的倍数） PARAM_ERROR CODE_READ_PROTECTION_ENABLED
描述	该命令用于读出 RAM、Flash 存储器或信息块的数据。当代码读保护使能时该命令被禁止。
示例	“R 268435456 4<CR><LF>” 从地址 0x1000 0000 读出 4 个字节数据。

20.7.6 准备写操作的扇区 < 起始扇区号 > < 结束扇区号 >

该命令使 Flash 写 / 擦除操作分成两个步骤处理。

表 297. ISP 准备写操作命令的扇区

命令	P
输入	起始扇区号 结束扇区号: 应当大于或等于起始扇区号。
返回代码	CMD_SUCCESS BUSY INVALID_SECTOR PARAM_ERROR
描述	该命令必须在执行 “ 将 RAM 内容复制到 Flash” 或 “ 擦除扇区 ” 命令之前执行。“ 将 RAM 内容复制到 Flash” 或 “ 擦除扇区 ” 命令的成功执行会使相关的扇区再次被保护。不能使用该命令准备引导块。要准备单个扇区, 可将 “ 起始 ” 和 “ 结束 ” 扇区号设置为相同值。
示例	“P 0 0<CR><LF>” 准备 Flash 扇区 0。

20.7.7 将 RAM 内容复制到 Flash<Flash 地址 > <RAM 地址 > < 字节数 >

表 298. ISP 复制命令

命令	C
输入	Flash 地址 (DST): 要写入数据字节的目标 Flash 地址。目标地址的边界应当为 Flash 字 (4 个字节)。最好在行边界 (128 个字节) 上写入, 因为这样的执行效率更高。复制的字节总数必须为 4096 个字节。 RAM 地址 (SRC): 读出数据字节的源 RAM 地址。 字节数: 写入的字节数。必须为 4 字节的倍数。
返回代码	CMD_SUCCESS SRC_ADDR_ERROR (地址不以字为边界) DST_ADDR_ERROR (地址不以字为边界, 正确的边界应以字为边界) SRC_ADDR_NOT_MAPPED DST_ADDR_NOT_MAPPED COUNT_ERROR (字节计数值不是 4 字节的倍数) SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION BUSY CMD_LOCKED PARAM_ERROR CODE_READ_PROTECTION_ENABLED
描述	该命令用于编程 Flash 存储器。“ 准备写操作的扇区 ” 命令应当在该命令之前被执行。当成功执行复制命令后, 受影响的扇区将自动再次受到保护。当代码读保护使能时该命令被禁止。
示例	“C 0 268467504 512<CR><LF>” 将 RAM 地址 0x1000 0800 开始的 512 字节复制到 Flash 地址 0。

20.7.8 运行 < 地址 > < 模式 >

表 299. ISP 运行命令

命令	G
输入	地址： 代码执行起始的 Flash 或 RAM 地址。该地址应当以字为边界。 模式： T（执行 Thumb 模式下的程序） A（执行 ARM 模式下的程序）。
返回代码	CMD_SUCCESS ADDR_ERROR ADDR_NOT_MAPPED CMD_LOCKED PARAM_ERROR CODE_READ_PROTECTION_ENABLED
描述	该命令用于执行位于 RAM 或 Flash 存储器当中的程序。一旦成功执行该命令，就有可能不再返回 ISP 命令处理程序。当代码读保护使能时该命令被禁止。
示例	“G 0 A<CR><LF>” 跳转到 ARM 模式下的地址 0x0000 0000 处

20.7.9 擦除扇区 < 起始扇区号 > < 结束扇区号 >

表 300. ISP 擦除扇区命令

命令	E
输入	起始扇区号 结束扇区号： 应当大于或等于起始扇区号。
返回代码	CMD_SUCCESS BUSY INVALID_SECTOR SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION CMD_LOCKED PARAM_ERROR CODE_READ_PROTECTION_ENABLED
描述	该命令用于擦除片内 Flash 存储器的一个或多个扇区。引导块不能使用该命令来擦除。当代码读保护使能时，该命令只允许擦除所有用户扇区的内容。
示例	“E 2 3<CR><LF>” 擦除 Flash 扇区 2 和 3。

20.7.10 扇区查空 < 起始扇区号 > < 结束扇区号 >

表 301. ISP 扇区查空命令

命令	I
输入	起始扇区号： 结束扇区号： 应当大于或等于起始扇区号。
返回代码	CMD_SUCCESS SECTOR_NOT_BLANK（后跟 < 第一个非空字的偏移量 > < 非空字的内容 >） INVALID_SECTOR PARAM_ERROR
描述	该命令用于对片内 Flash 存储器的一个或多个扇区进行查空。 对扇区 0 查空总是失败，这是由于前 512 字节映射到 ROM 引导块。
示例	“I 2 3<CR><LF>” 对 Flash 扇区 2 和 3 进行查空。

20.7.11 读器件标识号

表 302. ISP 读器件标识命令

命令	J
输入	无。
返回代码	CMD_SUCCESS 后跟 ASCII 格式的器件标识号（见表 303）。
描述	该命令用于读取器件的标识号。

表 303. LPC122x 器件标识号

器件	Hex 编码
LPC12D27FBD100/301	0x3670 002B
LPC1227FBD64/301	0x3670 002B
LPC1227FBD48/301	0x3670 002B
LPC1226FBD64/301	0x3660 002B
LPC1226FBD48/301	0x3660 002B
LPC1225FBD64/321	0x3652 002B
LPC1225FBD64/301	0x3650 002B
LPC1225FBD48/321	0x3652 002B
LPC1225FBD48/301	0x3650 002B
LPC1224FBD64/121	0x3642 C02B
LPC1224FBD64/101	0x3640 C02B
LPC1224FBD48/121	0x3642 C02B
LPC1224FBD48/101	0x3640 C02B

20.7.12 读 Boot 代码版本号

表 304. ISP 读 Boot 代码版本号命令

命令	K
输入	无
返回代码	CMD_SUCCESS 后跟 2 字节 ASCII 格式的 Boot 代码版本号。将其解释为 < 字节 1（主）>.< 字节 0（次）>。
描述	该命令用于读取 Boot 代码版本号。

20.7.13 比较 < 地址 1> < 地址 2> < 字节数 >

表 305. ISP 比较命令

命令	M
输入	地址 1 (DST): 要比较的数据字节的起始 Flash 或 RAM 地址。该地址应当以字为边界。 地址 2 (SRC): 要比较的数据字节的起始 Flash 或 RAM 地址。该地址应当以字为边界。 字节数: 待比较的字节数；计数值应当为 4 的倍数。
返回代码	CMD_SUCCESS （源和目标数据相同） COMPARE_ERROR （后跟第一个不匹配字节的地址） COUNT_ERROR （字节计数值不是 4 的倍数） ADDR_ERROR ADDR_NOT_MAPPED PARAM_ERROR
描述	该命令用来比较两个地址单元的存储器内容。 当源或目的地址包含从地址 0 起的前 512 个字节时，比较结果可能不正确。前 512 个字节重新映射到 boot ROM
示例	“M 8192 268468224 4<CR><LF>” 将 RAM 地址 0x1000 8000 开始的 4 个字节与 Flash 地址 0x2000 开始的 4 个字节进行比较。

20.7.14 读 UID

表 306. 读 UID 命令

命令	N
输入	无
返回代码	CMD_SUCCESS 后跟 E 类测试信息的 4 个 32 位字（ASCII 格式）。先发送低位地址的字。
描述	该命令用于读唯一的 ID。

20.7.15 ISP 返回代码

表 307. ISP 返回代码总览

返回代码	助记符	描述
0	CMD_SUCCESS	成功执行命令。只有成功执行了主机发出的命令后，才由 ISP 处理程序发送该代码。
1	INVALID_COMMAND	无效命令。
2	SRC_ADDR_ERROR	源地址没有以字为边界。
3	DST_ADDR_ERROR	目标地址的边界错误。
4	SRC_ADDR_NOT_MAPPED	源地址的映射不在存储器映射中。适当情况下将计数值考虑在内。
5	DST_ADDR_NOT_MAPPED	目标地址的映射不在存储器映射中。适当情况下将计数值考虑在内。
6	COUNT_ERROR	字节计数值不是 4 的倍数或是一个非法值。
7	INVALID_SECTOR	扇区号无效或结束扇区号大于起始扇区号。
8	SECTOR_NOT_BLANK	扇区非空。
9	SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION	为写操作准备扇区的命令未执行。
10	COMPARE_ERROR	源和目标数据不相等。

表 307. ISP 返回代码总览 (续)

返回代码	助记符	描述
11	BUSY	Flash 编程硬件接口忙。
12	PARAM_ERROR	参数不足或无效参数。
13	ADDR_ERROR	地址没有以字为边界。
14	ADDR_NOT_MAPPED	地址的映射不在存储器映射中。适当情况下将计数值考虑在内。
15	CMD_LOCKED	命令被锁定。
16	INVALID_CODE	解锁代码无效。
17	INVALID_BAUD_RATE	无效波特率设定。
18	INVALID_STOP_BIT	无效停止位设定。
19	CODE_READ_PROTECTION_ENABLED	代码读保护使能。

20.8 IAP 命令

对于在应用编程来说，应当通过寄存器 r0 中的字指针来调用 IAP 程序，该字指针指向含有命令代码和参数的存储器 (RAM)。IAP 命令的结果返回到寄存器 r1 所指向的结果表。用户可以把寄存器 r0 和 r1 中的指针赋予相同的值，如此便能将命令表复用来存放结果。参数表应当大到足够保存所有的结果以防结果的数目大于参数的数目。参数传递见图 56。参数和结果的数目根据 IAP 命令而有所不同。参数的最大数目为 5，传送到“将 RAM 内容复制到 Flash”命令。结果的最大数目为 4，由“读 UID”命令返回。命令处理程序在接收到一个未定义的命令时发送状态代码 INVALID_COMMAND。IAP 程序是 Thumb 代码，驻留在地址 0x1FFF 1FF0 处。

可按下面方式使用 C 调用 IAP 函数。

定义 IAP 程序的入口地址。由于 IAP 地址的第 0 位被置位，因此，当程序计数器转移到该地址时会使当前指令集变为 Thumb 指令集。

```
#define IAP_LOCATION 0x1fff1ff1
```

定义数据结构或指针，将 IAP 命令表和结果表传递给 IAP 函数：

```
unsigned long command[5];
unsigned long result[4];

或

unsigned long * command;
unsigned long * result;
command=(unsigned long *) 0x..
result= (unsigned long *) 0x..
```

定义函数类型指针，其包含 2 个参数，无返回值。需要注意的是 IAP 将函数结果和 R1 中的表格基址一同返回。

```
typedef void (*IAP)(unsigned int [],unsigned int[]);
IAP iap_entry;
```

设置函数指针：

```
iap_entry=(IAP) IAP_LOCATION;
```

可随时使用下面的语句来调用 IAP。

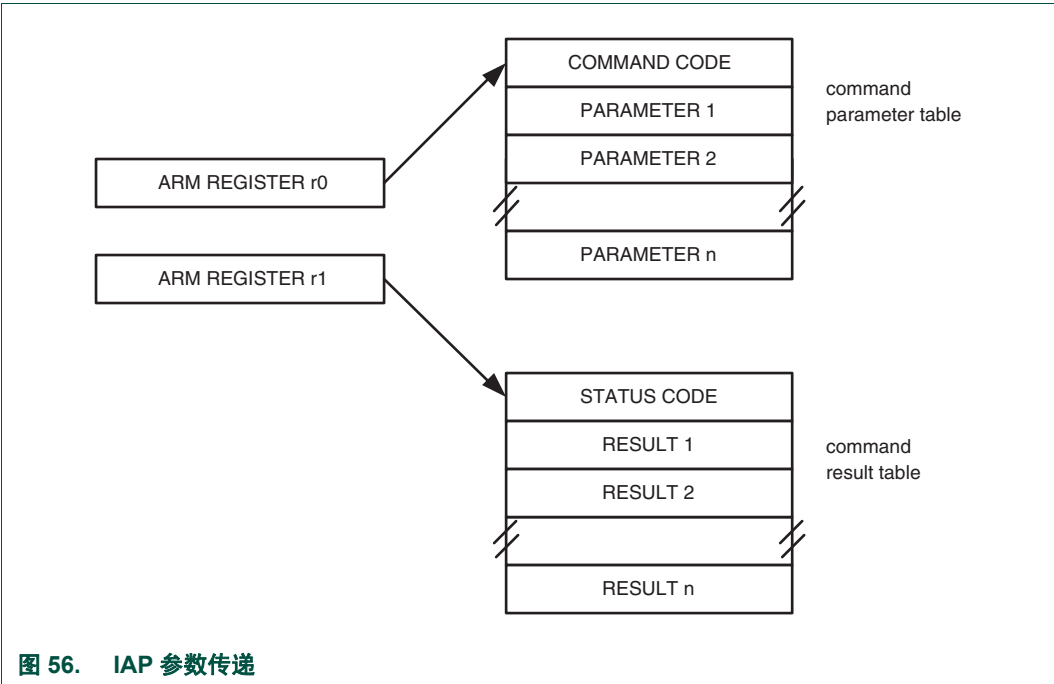
```
iap_entry (command, result);
```

根据 ARM 规范（ARM Thumb 过程调用标准 SWS ESPC 0002 A-05），可分别通过 r0、r1、r2 和 r3 寄存器来传递最多 4 个参数。另外的参数通过堆栈传递。最多 4 个参数可以分别通过 R0、R1、R2 和 R3 寄存器返回。其它的参数通过存储器间接返回。有些 IAP 调用需要多于 4 个参数。如果使用 ARM 建议的机制来传递 / 返回参数，则有可能因为不同厂商所提供的 C 编译器不同而产生问题。使用建议的参数传递机制便可降低这种风险。

在写或擦除操作过程中不能访问 Flash 存储器。那些导致 Flash 写 / 擦除操作的 IAP 命令将使用片内 RAM 顶端的 32 个字节空间来执行。如果应用程序中允许 IAP 编程，那么用户程序不应使用该空间。

表 308. IAP 命令总览

IAP 命令	命令代码（十进制）	描述
准备写操作的扇区	50	表 309
将 RAM 内容复制到 Flash	51	表 310
擦除扇区	52	表 311
扇区查空	53	表 312
读器件 ID	54	表 313
读 Boot 代码版本	55	表 314
比较	56	表 315
重新调用 ISP	57	表 316
读 UID	58	表 317
擦除页	59	表 318
擦除信息页	60	表 319



20.8.1 准备写操作的扇区

该命令使 Flash 写 / 擦除操作分成两个步骤处理。

表 309. IAP 准备写操作命令的扇区

命令	准备写操作的扇区
输入	命令代码：50（十进制） 参数 0：起始扇区号 参数 1：结束扇区号（应当大于或等于起始扇区号）。
返回代码	CMD_SUCCESS BUSY INVALID_SECTOR
结果	无
描述	该命令必须在执行“将 RAM 内容复制到 Flash”或“擦除扇区”命令之前执行。“将 RAM 内容复制到 Flash”或“擦除扇区”命令的成功执行会使相关的扇区再次被保护。不能使用该命令准备引导扇区。要准备单个扇区，可将“起始”和“结束”扇区号设置为相同值。要准备一页或多页进行擦除，请使用该命令并指定包含待擦除页的一个或多个扇区。

20.8.2 将 RAM 内容复制到 Flash

表 310. IAP 将 RAM 内容复制到 Flash 命令

命令	将 RAM 内容复制到 Flash
输入	<p>命令代码：51（十进制）</p> <p>参数 0(DST)：要写入数据字节的目标 Flash 地址。该地址应当以 4 字节为边界。目标地址的边界应当为 Flash 字（4 个字节）。最好在行边界（128 个字节）上写入，因为这样的执行效率更高。复制的字节总数必须为 4096 个字节。</p> <p>参数 1(SRC)：从中读取数据字节的源 RAM 地址。该地址应当以字为边界。</p> <p>参数 2：写入的字节数。必须为 4 字节的倍数。</p> <p>参数 3：系统时钟频率 (CCLK)（单位：kHz）。</p>
返回代码	<p>CMD_SUCCESS </p> <p>SRC_ADDR_ERROR（地址不以字为边界） </p> <p>DST_ADDR_ERROR（地址边界错误） </p> <p>SRC_ADDR_NOT_MAPPED </p> <p>DST_ADDR_NOT_MAPPED </p> <p>COUNT_ERROR（字节计数值不是 4 的倍数） </p> <p>SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION </p> <p>BUSY </p>
结果	无
描述	该命令用于编程 Flash 存储器。受影响的扇区应先通过调用“准备写操作扇区”命令准备好。当成功执行复制命令后，受影响的扇区将自动再次受到保护。

20.8.3 擦除扇区

表 311. IAP 擦除扇区命令

命令	擦除扇区
输入	<p>命令代码：52（十进制）</p> <p>参数 0：起始扇区号</p> <p>参数 1：结束扇区号（应当大于或等于起始扇区号）。</p> <p>参数 2：系统时钟频率 (CCLK)（单位：kHz）。</p>
返回代码	<p>CMD_SUCCESS </p> <p>BUSY </p> <p>SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION </p> <p>INVALID_SECTOR</p>
结果	无
描述	该命令用于擦除片内 Flash 存储器的一个或多个扇区。该命令不能擦除引导扇区。要擦除单个扇区，可将“起始”和“结束”扇区号设定为相同值。

20.8.4 扇区查空

表 312. IAP 扇区查空命令

命令	扇区查空
输入	命令代码：53（十进制） 参数 0：起始扇区号 参数 1：结束扇区号（应当大于或等于起始扇区号）。
返回代码	CMD_SUCCESS BUSY SECTOR_NOT_BLANK INVALID_SECTOR
结果	结果 0：状态代码为 SECTOR_NOT_BLANK 时第一个非空字位置的偏移量。 结果 1：非空字位置的内容。
描述	该命令用于对片内 Flash 存储器的一个或多个扇区进行查空。要查空单个扇区，可将“起始”和“结束”扇区号设定为相同值。

20.8.5 读器件标识号

表 313. IAP 读器件标识命令

命令	读器件标识号
输入	命令代码：54（十进制） 参数：无
返回代码	CMD_SUCCESS
结果	结果 0：器件标识号。
描述	该命令用于读取器件的标识号。

20.8.6 读 Boot 代码版本号

表 314. IAP 读 Boot 代码版本号命令

命令	读 Boot 代码版本号
输入	命令代码：55（十进制） 参数：无
返回代码	CMD_SUCCESS
结果	结果 0：2 字节 Boot 代码版本号（ASCII 格式）。将其解释为 < 字节 1（主）>.< 字节 0（次）>
描述	该命令用于读取 Boot 代码版本号。

20.8.7 比较 < 地址 1> < 地址 2> < 字节数 >

表 315. IAP 比较命令

命令	比较
输入	命令代码：56（十进制） 参数 0(DST): 要比较的数据字节的起始 Flash 或 RAM 地址。该地址应当以字为边界。 参数 1(SRC): 要比较的数据字节的起始 Flash 或 RAM 地址。该地址应当以字为边界。 参数 2: 待比较的字节数；计数值应当为 4 的倍数。
返回代码	CMD_SUCCESS COMPARE_ERROR COUNT_ERROR（字节计数值不是 4 的倍数） ADDR_ERROR ADDR_NOT_MAPPED
结果	结果 0: 当状态代码为 COMPARE_ERROR 时第一个不匹配字节的偏移地址。
描述	该命令用来比较两个地址单元的存储器内容。 当源或目标地址包含从地址 0 开始的前 512 字节中的任意一个地址时，比较的结果可能不正确。因为前 512 字节是可以重新映射到 RAM 中的。

20.8.8 重新调用 ISP

表 316. IAP 重新调用 ISP

命令	重新调用 ISP
输入	命令代码：57（十进制）
返回代码	无
结果	无。
描述	该命令用来调用 ISP 模式中的引导装载程序。它会映射引导向量，设定 PCLK = CCLK，配置 UART 引脚 RXD0 和 TXD0，复位计数器 / 定时器 CT32B1 和 U0FDR（见表 154）。内部 Flash 存储器中出现有效的用户程序且 PIO0_12 引脚不可访问时，可使用该指令强制进入 ISP 模式。

20.8.9 读 UID

表 317. IAP 读 UID 命令

命令	读 UID
输入	命令代码：58（十进制）
返回代码	CMD_SUCCESS
结果	结果 0: 第 1 个 32 位字（在最低地址）。 结果 1: 第 2 个 32 位字。 结果 2: 第 3 个 32 位字。 结果 3: 第 4 个 32 位字。
描述	该命令用于读唯一的 ID。

20.8.10 擦除页

表 318. IAP 擦除页命令

命令	擦除页
输入	命令代码：59（十进制） 参数 0：擦除起始页。 参数 1：擦除结束页（应大于或等于起始页） 参数 2：系统时钟频率 (CCLK)（单位：kHz）。
返回代码	CMD_SUCCESS BUSY SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION INVALID_SECTOR
结果	无
描述	该命令用于擦除片内 Flash 存储器的一页或多页。要擦除单页，可使用相同的“起始”和“结束”页号。

20.8.11 擦除信息页

表 319. IAP 擦除信息页命令

命令	擦除信息页
输入	命令代码：60（十进制） 参数 0：擦除信息块中的起始页。 参数 1：擦除信息块中的结束页（应大于或等于起始页） 参数 2：系统时钟频率 (CCLK)（单位：kHz）。
返回代码	CMD_SUCCESS BUSY SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION INVALID_SECTOR
结果	无
描述	该命令用于擦除信息块的一页或多页。信息块包含 3 页，编号分别是 0、1 和 2。要擦除单页，可使用相同的“起始”和“结束”页号。

20.8.12 IAP 状态代码

表 320. IAP 状态代码小结

状态代码	助记符	描述
0	CMD_SUCCESS	成功执行命令。
1	INVALID_COMMAND	无效命令。
2	SRC_ADDR_ERROR	源地址没有以字为边界。
3	DST_ADDR_ERROR	目标地址的边界错误。无效信息页地址。
4	SRC_ADDR_NOT_MAPPED	源地址的映射不在存储器映射中。适当情况下将计数值考虑在内。
5	DST_ADDR_NOT_MAPPED	目标地址的映射不在存储器映射中。适当情况下将计数值考虑在内。
6	COUNT_ERROR	字节计数值不是 4 的倍数或是一个非法值。
7	INVALID_SECTOR	扇区号或页号无效。

表 320. IAP 状态代码小结 (续)

状态代码	助记符	描述
8	SECTOR_NOT_BLANK	扇区非空。
9	SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION	为写操作准备扇区的命令未执行。
10	COMPARE_ERROR	源和目标数据不相同。
11	BUSY	Flash 编程硬件接口忙。

20.9 串行线调试 (SWD)Flash 编程接口

调试工具可将一部分 Flash 映像写入 RAM，然后按照正确的偏移地址重复调用 IAP 命令“将 RAM 内容复制到 Flash”。

21.1 本章导读

所有 LPC122x 部件都提供微 DMA 控制器。

21.2 简介

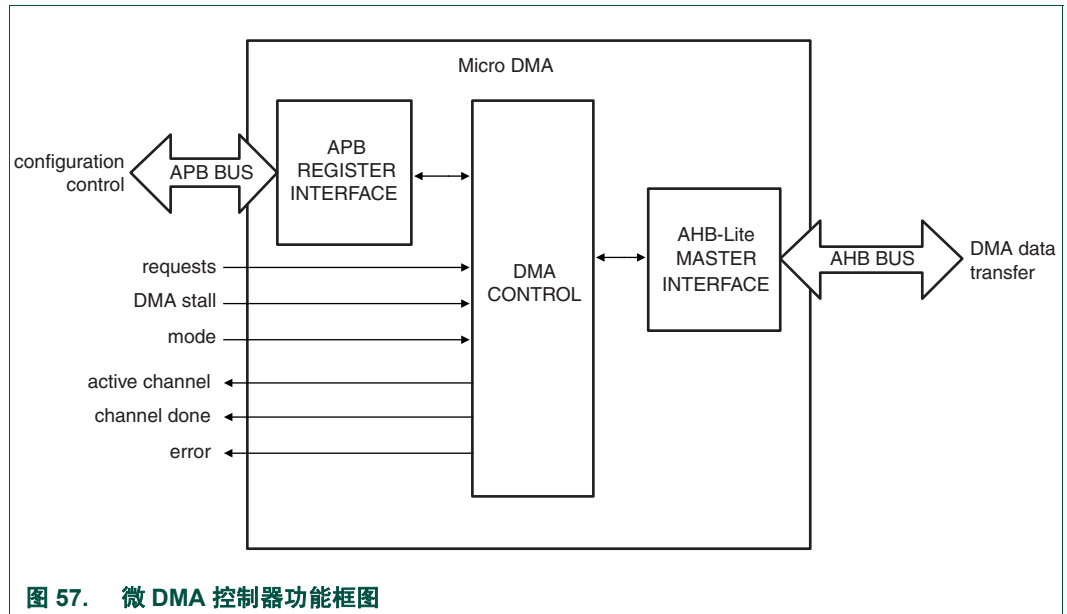
微 DMA 控制器是一个门数极低的 DMA，与 AMBA AHB-Lite 协议兼容以支持 DMA 传输。微 DMA 寄存器通过 APB 接口编程。

21.3 特性

- 单个 AHB-Lite 主机使用 32 位地址和数据总线传输数据。
- 21 个 DMA 通道，外加一个用于存储器到存储器传输的通道。
- 每个通道都带握手信号和可编程优先级。
- 每个优先级仲裁使用一个由 DMA 通道号决定的固定优先级。
- 支持存储器到存储器、存储器到外设、外设到存储器的传输。
- 支持多种 DMA 周期类型和多种 DMA 传输宽度。
- 每个 DMA 通道都能访问一个主用和一个备用通道控制数据结构。
- 通道控制数据按小端格式存储在系统内存中。
- 使用单 AHB-Lite 传输执行所有 DMA 传输。不支持猝发传输。
- 目标数据宽度与源数据宽度相等。
- 单个 DMA 周期的传输数可在 1 到 1024 之间设置。
- 传输地址增量可能大于数据宽度。
- AHB 上出现错误情况时通过单路输出进行指示。

21.4 描述

微 DMA 控制器包含 APB 寄存器接口、到 AHB 多层矩阵的 AHB-Lite 主机接口以及 DMA 控制模块（见[图 57](#)）。



DMA 控制器支持 8、16 或 32 位（字节、半字或字）的数据传输大小，可通过通道控制数据结构（见[表 347](#)）配置。源数据传输大小和目标数据传输大小必须相同。控制器访问通道控制数据结构时始终使用 32 位数据传输。

DMA 控制模块包含执行下列任务的控制逻辑：

- 仲裁接收到的请求。
- 指示哪个通道活跃。
- 通道完成时给予提示。
- AHB-Lite 接口上发生错误时进行指示。
- 允许较慢的外设中断 DMA 周期。
- 完成 DMA 周期之前等待请求清除。
- 对每个请求执行多个或单个 DMA 传输。
- 执行以下类型的 DMA 传输：
 - 存储器到存储器
 - 存储器到外设
 - 外设到存储器

注：不支持外设到外设交易，因为每个通道仅提供一个 DMA 请求接口。

21.4.1 微 DMA 控制器可访问的存储器区域

DMA 通道控制数据结构写入 SRAM 并在其中更新。存储器到存储器 DMA 传输作为软件控制的传输得到支持。

21.4.2 DMA 系统连接

微 DMA 与支持的外设之间的连接类型取决于这些外设中实施的 DMA 功能。SSP 使用单传输和传输请求，UART 和 ADC 使用允许一个或一个以上传输的传输请求。有关传输类型的更多详情，请参见[表 343](#)。[表 321](#) 显示了支持的外设使用的 DMA 通道编号。

表 321. DMA 连接

外设	DMA 通道	单 DMA 传输请求	DMA 传输请求
UART0 Tx	0	-	是
UART0 Rx	1	-	是
UART1 Tx	2	-	是
UART1 Rx	3	-	是
SSP Tx	4	是	是
SSP Rx	5	是	是
ADC	6	-	是
RTC	7	-	是
32 位定时器 0 匹配 0	8	-	是
32 位定时器 0 匹配 1	9	-	是
32 位定时器 1 匹配 0	10	-	是
32 位定时器 1 匹配 1	11	-	是
16 位定时器 0 匹配 0	12	-	是
16 位定时器 1 匹配 0	13	-	是
比较器 0	14	-	是
比较器 1	15	-	是
PIO 0	16	-	是
PIO 1	17	-	是
PIO 2	18	-	是
保留	19	-	是
保留	20	-	是
存储器到存储器	21	-	仅软件

[1] 任何未用于外设 DMA 的 DMA 通道都可用于执行软件触发的存储器到存储器传输。

DMA 中断连接到 NVIC（见[表 4](#)）。中断信号由各通道的 DMA 完成信号以及 DMA 错误情况（如使能）产生。另请参见[表 343](#)。

21.5 计时和功耗控制

至微 DMA 控制器的时钟由系统时钟提供，而系统时钟由 SYSAHBCLKDIV 寄存器控制（[表 20](#)）。微 DMA 控制器可以通过系统 AHB 时钟控制寄存器位 12（[表 21](#)）禁用以降低功耗。

21.6 寄存器描述

[表 322](#) 显示了微 DMA 寄存器映射。

表 322. 寄存器简介：微 DMA（基址 0x4004 C000）

名称	访问类型	地址偏移	描述	复位值
DMA_STATUS	RO	0x000	DMA 状态寄存器	-
DMA_CFG	WO	0x004	DMA 配置寄存器	-
CTRL_BASE_PTR	R/W	0x008	通道控制基址指针寄存器	0x0000 0000
ATL_CTRL_BASE_PTR	RO	0x00C	通道备用控制基址指针寄存器	
DMA_WAITONREQ_STATUS	RO	0x010	通道应请求等待状态寄存器	< 待定 >
CHNL_SW_REQUEST	WO	0x014	通道软件请求寄存器	-
CHNL_USEBURST_SET	R/W	0x018	通道使用猝发设置寄存器	0x0000 0000
CHNL_USEBURST_CLR	WO	0x01C	通道使用猝发清除寄存器	-
CHNL_REQ_MASK_SET	R/W	0x020	通道请求屏蔽设置寄存器	0x0000 0000
CHNL_REQ_MASK_CLR	WO	0x024	通道请求屏蔽清除寄存器	-
CHNL_ENABLE_SET	R/W	0x028	通道使能设置寄存器	0x0000 0000
CHNL_ENABLE_CLR	WO	0x02C	通道使能清除寄存器	-
CHNL_PRI_ALT_SET	R/W	0x030	通道主 - 备用设置寄存器	0x0000 0000
CHNL_PRI_ALT_CLR	WO	0x034	通道主 - 备用清除寄存器	-
CHNL_PRIORITY_SET	R/W	0x038	通道优先级设置寄存器	0x0000 0000
CHNL_PRIORITY_CLR	WO	0x03C	通道优先级清除寄存器	-
-	-	0x040 - 0x048	保留	-
ERR_CLR	R/W	0x04C	总线错误清除寄存器	0x0000 0000
-	-	0x050 - 0x07C	保留	-
CHNL_IRQ_STATUS	R/W	0x080	通道 DMA 中断状态寄存器	0x0000 0000
IRQ_ERR_ENABLE	R/W	0x084	DMA 错误中断使能寄存器	0x0000 0000
CHNL_IRQ_ENABLE	R/W	0x088	通道 DMA 中断使能寄存器	0x0000 0000

21.6.1 DMA 状态寄存器

该寄存器为只读寄存器，返回微 DMA 控制器的状态。当微 DMA 控制器处于复位状态时，该寄存器不可读。

表 323. DMA 状态寄存器（DMA_STATUS，地址 0x4004 C000）位描述

位	符号	描述	复位值
0	MASTER_EN	控制器使能状态： 0 = 控制器被禁能。 1 = 控制器被使能。	
3:1	-	保留。	-
7:4	STATE	控制状态机器的当前状态。可以是以下任一状态： 0000 = 空闲 0001 = 正在读取通道控制器数据 0010 = 正在读取源数据末端指针 0011 = 正在读取目标数据末端指针 0100 = 正在读取源数据 0101 = 正在写入目标数据 0110 = 正在等待 DMA 请求清除 0111 = 正在写入通道控制器数据 1000 = 停止 1001 = 完成 1010 = 外设分散 - 集中跃迁 1011-1111 = 未定义	
15:8	-	保留。	-
20:16	CHNLS	为 22 个通道配置了 LPC122x 微 DMA 控制器。复位值 10100 为通道数 - 1。	
31:21	-	保留。	-

21.6.2 DMA 配置寄存器

该寄存器为只写寄存器，可配置微 DMA 控制器。

CHNL_PROT_CTRLn 位（位 6:5）可配置 AHB 对通道控制数据结构的访问保护控制，为确保 DMA 控制器的正常操作应置为零。

表 324. DMA 配置寄存器（DMA_CFG，地址 0x4004 C004）位描述

位	符号	值	描述	复位值
0	MASTER_EN		控制器使能：	-
		0	禁能控制器。	
		1	使能控制器。	
4:1	-		保留。写为 0。	-
5	CHNL_PROT_CTRL1		通道保护访问控制：	-
		0	用户访问	
		1	特权访问	

表 324. DMA 配置寄存器（DMA_CFG，地址 0x4004 C004）位描述（续）

位	符号	值	描述	复位值
6	CHNL_PROT_CTRL2		通道保护缓冲控制：	-
		0	不可缓冲	
		1	可缓冲	
31:7	-		保留。写为 0。	-

21.6.3 通道控制基址指针寄存器

该寄存器为读 / 写寄存器，可配置基址指针。基址指针必须指向 LPC122x 的 SRAM 中的一个位置，因为微 DMA 控制器不提供用于存储通道控制数据结构的内部存储器。

当微 DMA 控制器处于复位状态时，该寄存器不可读。

表 325. 通道控制基址指针寄存器（CTRL_BASE_PTR，地址 0x4004 C008）位描述

位	符号	描述	复位值
7:0	-	保留。写为 0。	0x0
31:8	CTRL_BASE_PTR	至主用数据结构基址的指针。	0x0

21.6.4 通道备用控制基址指针寄存器

该寄存器为只读寄存器，返回备用数据结构的基址。

该寄存器消除了应用程序软件计算备用数据结构基址的必要（见[章节 21.7.5](#)）。

当微 DMA 控制器处于复位状态时，该寄存器不可读。

表 326. 通道备用控制基址指针寄存器（ALT_CTRL_BASE_PTR，地址 0x4004 C00C）位描述

位	符号	描述	复位值
31:0	ALT_CTRL_BASE_PTR	备用数据结构的基址	0x0

21.6.5 通道应请求等待状态寄存器

该寄存器为只读寄存器，返回通道 c (c = 0-20) 的 dma_waitonreq[c] 信号的状态。

当微 DMA 控制器处于复位状态时，该寄存器不可读。

表 327. 通道应请求等待状态寄存器（DMA_WAITONREQ_STATUS，地址 0x4004 C010）位描述

位	符号	描述	复位值
21:0	DMA_WAITONREQ_STATUS	通道 c 应请求等待状态 (c = 0-20): 位 c = 0: dma_waitonreq[c] 为低电平。 位 c = 1: dma_waitonreq[c] 为高电平。	-
31:22	-	保留。	-

21.6.6 通道软件请求寄存器

该寄存器为只写寄存器，允许通道 c (c = 0-20) 产生软件 DMA 请求。

对未实现 DMA 通道的位的写操作不会为该通道创建 DMA 请求。

表 328. 通道软件请求寄存器（CHNL_SW_REQUEST，地址 0x4004 C014）位描述

位	符号	描述	复位值
21:0	DMA_SW_REQUEST	将相应位置位以在相应 DMA 通道上产生软件 DMA 请求。 写为： 位 [c] = 0：不为通道 c 创建 DMA 请求。 位 [c] = 1：为通道 c 创建 DMA 请求。	-
31:22	-	保留。	-

21.6.7 通道使用猝发设置寄存器

该寄存器为读 / 写寄存器，可禁止通道 c (c = 0-20) 的单个 DMA 请求 (dma_sreq[c]) 输入产生请求。因此，只有 dma_req[c] 信号产生请求。

注：使用猝发状态寄存器仅适用于通道 4 和 5 (SSP)。

读取该寄存器将返回使用猝发状态。对未实现 DMA 通道的位的写操作无任何效果。

表 329. 通道使用猝发设置寄存器（CHNL_USEBURST_SET，地址 0x4004 C018）位描述

位	符号	描述	复位值
21:0	DMA_USEBURST_SET	返回通道 c (c = 0-20) 的使用猝发状态或禁止 dma_sreq[c] 产生 DMA 请求。（仅适用于通道 4 和 5。） 读为： 位 [c] = 0：DMA 通道 c 响应它在 dma_req[c] 或 dma_sreq[c] 上收到的请求。控制器执行 2 ^R ，或单总线传输。 位 [c] = 1：DMA 通道 c 不会响应它在 dma_sreq[c] 上收到的请求。控制器仅响应 dma_req[c] 请求并执行 2 ^R 传输。 写为： 位 [c] = 0：无效。使用 CHNL_USEBURST_CLR 寄存器将位 [c] 置为 0。 位 [c] = 1：禁止 dma_sreq[C] 产生 DMA 请求。控制器执行 2 ^R 传输。	0x0
31:22	-	保留。	-

21.6.8 通道使用猝发清除寄存器

该寄存器为只写寄存器，可对通道 c 使能 DMA 单请求 (c = 0-7, dma_sreq[c]) 以产生请求。对未实现 DMA 通道的位的写操作无任何效果。

注：使用猝发清除寄存器仅适用于通道 4 和 5 (SSP)。

表 330. 通道使用猝发清除寄存器（CHNL_USEBURST_CLR，地址 0x4004 C01C）位描述

位	符号	描述	复位值
21:0	CHNL_USEBURST_CLR	将相应位置位以允许 dma_sreq[c] 产生请求。（仅适用于通道 4 和 5。） 写为： 位 [c] = 0：无效。使用 chnl_useburst_set 寄存器禁止 dma_sreq[c] 产生请求。 位 [c] = 1：允许 dma_sreq[c] 产生 DMA 请求。	-
31:22	-	保留。	-

21.6.9 通道请求屏蔽设置寄存器

该寄存器为读 / 写寄存器，禁止通道 c ($c = 0-20$) 的 DMA 请求信号（`dma_req[c]` 信号）或单 DMA 请求信号 (`dma_sreq[c]`) 上的高电平产生请求。对该寄存器的读操作将返回 `dma_req[c]` 和 `dma_sreq[c]` 的请求屏蔽状态。对未实现 DMA 通道的位的写操作无任何效果。

表 331. 通道请求屏蔽设置寄存器（CHNL_REQ_MASK_SET，地址 0x4004 C020）位描述

位	符号	描述	复位值
21:0	CHNL_REQ_MASK_SET	返回 <code>dma_req[c]</code> 和 <code>dma_sreq[c]</code> 的请求屏蔽状态，或禁止相应通道产生 DMA 请求。 读为： 位 $[c] = 0$ ：对通道 c 使能外部请求。 位 $[c] = 1$ ：对通道 c 禁能外部请求。 写为： 位 $[c] = 0$ ：无效。使用 CHNL_REQ_MASK_CLR 寄存器使能 DMA 请求。 位 $[c] = 1$ ：禁止 <code>dma_req[c]</code> 和 <code>dma_sreq[c]</code> 产生 DMA 请求。	0x0
31:22	-	保留。	-

21.6.10 通道请求屏蔽清除寄存器

该寄存器为只写寄存器，对通道 c ($c = 0-20$) 允许 `dma_req[c]` 或 `dma_sreq[c]` 的高电平输出产生请求。对未实现 DMA 通道的位的写操作无任何效果。

表 332. 通道请求屏蔽清除寄存器（CHNL_REQ_MASK_CLR，地址 0x4004 C024）位描述

位	符号	描述	复位值
21:0	CHNL_REQ_MASK_CLR	将相应位置位可允许 <code>dma_req[c]</code> 和 <code>dma_sreq[c]</code> 的相应通道产生 DMA 请求。 写为： 位 $[c] = 0$ ：无效。使用 <code>chnl_req_mask_set</code> 寄存器禁止 <code>dma_req[c]</code> 和 <code>dma_sreq[c]</code> 产生请求。 位 $[c] = 1$ ：允许 <code>dma_req[c]</code> 或 <code>dma_sreq[c]</code> 产生 DMA 请求。	-
31:22	-	保留。	-

21.6.11 通道使能设置寄存器

该寄存器为读 / 写寄存器，使能 DMA 通道 c ($c = 0-20$)。对该寄存器的读操作将返回通道的使能状态。对未实现 DMA 通道的位的写操作无任何效果。

表 333. 通道使能设置寄存器（CHNL_ENABLE_SET，地址 0x4004 C028）位描述

位	符号	描述	复位值
21:0	CHNL_ENABLE_SET	返回通道的使能状态，或使能相应通道。 读为： 位 $[c] = 0$ ：通道 c 被禁能。 位 $[c] = 1$ 通道 c 被使能。 写为： 位 $[c] = 0$ ：无效。使用 CHNL_ENABLE_CLR 寄存器禁能通道。 位 $[c] = 1$ ：使能通道 c 。	0x0
31:22	-	保留。	-

21.6.12 通道使能清除寄存器

该寄存器为只写寄存器，可禁能 DMA 通道。对未实现 DMA 通道的位的写操作无任何效果。

注：控制器可在以下任一情况下通过将相应位置位来禁能某个通道：

- 控制器完成 DMA 周期。
- 控制器读取 cycle_ctrl = 000 的 channel_cfg 存储器位置。
- AHB-Lite 总线发生错误。

表 334. 通道使能清除寄存器（CHNL_ENABLE_CLR，地址 0x4004 C02C）位描述

位	符号	描述	复位值
21:0	CHNL_ENABLE_CLR	将相应位置位以禁能相应 DMA 通道。 写为： 位 [c] = 0：无效。使用 CHNL_ENABLE_SET 寄存器使能 DMA 通道。 位 [c] = 1 禁能通道 c。	-
31:22	-	保留。	-

21.6.13 通道主 - 备用设置寄存器

该寄存器为读 / 写寄存器，将 DMA 通道 c (c = 0-20) 配置为使用备用数据结构。对该寄存器的读操作将返回相应 DMA 通道正使用的数据结构的状态。对未实现 DMA 通道的位的写操作无任何效果。

注：控制器在完成以下任一任务后切换 CHNL_PRI_ALT_SET[c] 位的值：

- 主数据结构为内存分散 - 集中或外设分散 - 集中 DMA 周期指定的四个传输。有关详情，请参阅 *ARM 微 DMA 文档*。
- 主数据结构为乒乓 DMA 周期指定的所有传输。
- 备用数据结构为以下 DMA 周期类型指定的所有传输：
 - 乒乓。
 - 内存分散 - 集中。有关详情，请参阅 *ARM 微 DMA(PL230) 文档*。
 - 外设分散 - 集中。有关详情，请参阅 *ARM 微 DMA(PL230) 文档*。

表 335. 通道主用 - 备用设置寄存器（CHNL_PRI_ALT_SET，地址 0x4004 C030）位描述

位	符号	描述	复位值
21:0	CHNL_PRI_ALT_SET	返回通道控制数据结构状态，或者为相应 DMA 通道 c 选择备用数据结构。读为： 位 [c] = 0：DMA 通道 c 在使用主数据结构。 位 [c] = 1：DMA 通道 c 在使用备用数据结构。 写为： 位 [c] = 0：无效。使用 CHNL_PRI_ALT_CLR 寄存器将位 [c] 置为 0。 位 [c] = 1：为通道 c 选择备用数据结构。	0x0
31:22	-	保留。	-

21.6.14 通道主 - 备用清除寄存器

该寄存器为只写寄存器，可将 DMA 通道配置为使用主数据结构。对未实现 DMA 通道的位的写操作无任何效果。

注：控制器在完成以下任一任务后切换 CHNL_PRI_ALT_CLR[c] 位的值：

- 主数据结构为内存分散 - 集中或外设分散 - 集中 DMA 周期指定的四个传输。
- 主数据结构为乒乓 DMA 周期指定的所有传输。
- 备用数据结构为以下 DMA 周期类型指定的所有传输：
 - 乒乓。
 - 内存分散 - 集中。有关详情，请参阅 *ARM 微 DMA(PL230) 文档*。
 - 外设分散 - 集中。有关详情，请参阅 *ARM 微 DMA(PL230) 文档*。

表 336. 通道主用 - 备用清除寄存器（CHNL_PRI_ALT_CLR，地址 0x4004 C034）位描述

位	符号	描述	复位值
21:0	CHNL_PRI_ALT_CLR	将相应位置位以便为相应 DMA 通道 c 选择主数据结构。 写为： 位 [c] = 0：无效。使用 CHNL_PRI_ALT_SET 寄存器选择备用数据结构。 位 [c] = 1：为通道 c 选择主数据结构。	-
31:22	-	保留。	-

21.6.15 通道优先级设置寄存器

该寄存器为读 / 写寄存器，将 DMA c (c = 0-20) 通道配置为使用高优先级。对该寄存器的读操作将返回通道优先级屏蔽的状态。对未实现 DMA 通道的位的写操作无任何效果。

表 337. 通道优先级设置寄存器（CHNL_PRIORITY_SET，地址 0x4004 C038）位描述

位	符号	描述	复位值
21:0	CHNL_PRIORITY_SET	返回通道优先级屏蔽状态，或将通道优先级设为高。 读为： 位 [c] = 0：DMA 通道 c 在使用默认优先级。 位 [c] = 1：DMA 通道 c 在使用高优先级。 写为： 位 [c] = 0：无效。使用 CHNL_PRIORITY_CLR 寄存器将通道 c 设为默认优先级。 位 [c] = 1：通道 c 使用高优先级。	0x0
31:22	-	保留。	-

21.6.16 通道优先级清除寄存器

该寄存器为只写寄存器，可将 DMA 通道 c (c = 0-20) 配置为使用默认优先级。对未实现 DMA 通道的位的写操作无任何效果。

表 338. 通道优先级清除寄存器（CHNL_PRIORITY_CLR，地址 0x4004 C03C）位描述

位	符号	描述	复位值
21:0	CHNL_PRIORITY_CLR	将相应位置位以便为指定 DMA 通道选择默认优先级。 写为： 位 [c] = 0：无效。使用 CHNL_PRIORITY_SET 寄存器将通道 c 设为高优先级。 位 [c] = 1：通道 c 使用默认优先级。	-
31:22	-	保留。	-

21.6.17 总线错误清除寄存器

该寄存器为读 / 写寄存器，返回 dma_err 状态或将 dma_err 信号设为低电平。

注：如果 dma_err 在 AHB-Lite 总线发生错误时被撤销，则错误条件优先且 dma_err 保持有效。

表 339. 总线错误清除寄存器（ERR_CLR，地址 0x4004 C04C）位描述

位	符号	描述	复位值
0	ERR_CLR	返回 dma_error 状态或将信号设为低电平。 读为： 0 = dma_err 为低电平。 1 = dma_err 为高电平。 写为： 0 = 无效， dma_err 状态不变。 1 = 将 dma_err 设为低电平。	-
31:1	-	保留。写为 0。	0x0

21.6.18 通道 DMA 中断状态寄存器

该寄存器为读 / 写寄存器，显示每个 DMA 通道 c (c = 0-20) 的 DMA 完成中断状态。写 “1” 将清除状态位。对未实现 DMA 通道的位的写操作无任何效果。

表 340. 通道 DMA 中断状态寄存器（CHNL_IRQ_STATUS，地址 0x4004 C080）位描述

位	符号	描述	复位值
21:0	CHNL_IRQ_STAT	返回每个通道的 DMA 完成中断的状态。 读为： 位 [c] = 0：DMA 完成中断未撤销。 位 [c] = 1：通道 c 的 DMA 传输完成。 写为： 位 [c] = 0：无效。 位 [c] = 1：清除通道 c 的 DMA 完成状态。	0x0
31:22	-	保留。	-

21.6.19 DMA 错误中断使能寄存器

该寄存器为读 / 写寄存器，使能 DMA 错误中断。对未实现 DMA 通道的位的写操作无任何效果。

表 341. DMA 错误中断使能寄存器（IRQ_ERR_ENABLE，地址 0x4004 C084）位描述

位	符号	描述	复位值
0	IRQ_ERR_ENABLE	允许 DMA 错误 (dma_error) 信号创建中断。 写为： 位 0 = 0：禁能 DMA 错误中断。 位 0 = 1：使能 DMA 错误中断。	0x0
31:1	-	保留。	-

21.6.20 通道 DMA 中断使能寄存器

该寄存器为读 / 写寄存器，允许 DMA 通道 c (c = 0-20) 在完成 DMA 传输后创建中断。对未实现 DMA 通道的位的写操作无任何效果。

表 342. 通道 DMA 中断使能寄存器（CHNL_IRQ_ENABLE，地址 0x4004 C088）位描述

位	符号	描述	复位值
21:0	CHNL_IRQ_ENABLE	允许 DMA 完成 (dma_done[c]) 信号创建中断。 写为： 位 [c] = 0：禁能通道 c 的 DMA 完成中断。 位 [c] = 1：使能通道 c 的 DMA 完成中断。	0x0
31:22	-	保留。	-

21.7 功能说明

21.7.1 DMA 控制信号

表 343 中列出了用于 DMA 传输以及用于为外设到存储器传输提供握手信号的 DMA 控制信号。

表 343. DMA 控制信号

信号	名称	源 / 目标	描述
DMA 通道请求	dma_req[c]	外设 / 控制器	外设在有一个或多个需要服务的数据传输时使 dma_req[c] 有效。控制器通过使用 2 ^R DMA 传输执行 DMA 周期来服务请求，视 R_power（见表 347）设置而定可能在传输之间进行仲裁。在通道 c 的传输完成之前，dma_req[c] 信号保持高电平。然后撤销请求。所有外设要等到传输请求清除后才开始下一个传输。
DMA 单通道请求	dma_sreq[c]	外设 / 控制器	外设在有一个需要服务的数据传输时使 dma_sreq 有效。控制器通过使用单一 DMA 传输执行 DMA 周期来服务请求。在通道 c 的传输完成之前，dma_sreq[c] 信号保持高电平。然后撤销请求。所有外设要等到传输请求清除后才开始下一个传输。
DMA 通道请求控制	dma_waitonreq[c]	外设 / 控制器	完成一组 2 ^R DMA 传输后，该信号能防止控制器撤销 dma_active，直到相应 dma_req 或 dma_sreq 被取消。
DMA 停止	dma_stall[c]	外设 / 控制器	外设可使用 dma_stall 来延长当前 DMA 周期。在撤销请求方面较慢的外设，开始 DMA 传输后可能意外触发额外请求。在这种情况下，外设必须使 dma_stall 有效，直至请求可被取消。

表 343. DMA 控制信号 (续)

信号	名称	源 / 目标	描述
DMA 激活	dma_active[c]	控制器 / 外设	控制器使正在传输数据的当前激活通道的 dma_active 信号有效。不论何时，一次只能激活一个 dma_active 信号。
DMA 完成	dma_done[c]	控制器 / NVIC	控制器完成 DMA 周期后，它在一个系统时钟周期内使 dma_done 有效。对于已使能的通道，不论何时，一次都只能激活一个 dma_done 信号。DMA 完成信号将 CHNL_IRQ_STATUS 寄存器 (表 340) 中的相应位置位，如果在 CHNL_IRQ_ENABLE 寄存器 (表 342) 中使能，还会对 NVIC 产生一个中断。
总线错误	dma_err	控制器 / NVIC	这是 DMA 中断信号。如果 AHB-Lite 主机接口发生错误，控制器将使 dma_err 变为高电平有效。在用户对 ERR_CLR 寄存器 (表 339) 执行写操作之前，该信号将一直保持高电平。如果在 IRQ_ERR_ENABLE 寄存器 (表 341) 中使能，dma_err 信号可对 NVIC 产生一个中断。

21.7.2 DMA 仲裁

控制器可配置为在可编程次数的传输前后在 DMA 周期期间执行仲裁。这将缩短服务高优先级通道的延迟。

控制器使用通道控制数据结构 (见表 347) 中的四个位来配置在控制器重新仲裁之前发生的 AHB 总线传输次数。这些位被称为 R_power 位，因为值 R 被升为 2 的幂，这决定了仲裁率。例如，如果 R = 4，则仲裁率为 2⁴，这表示控制器每 16 次 DMA 传输仲裁一次。

注：不要为低优先级通道分配较大的 R_power 值，因为这将阻止控制器服务高优先级请求，直至控制器重新仲裁。

当 N > 2^R 且不是 2^R 的整数倍时，控制器始终执行 2^R 次传输序列，直至剩余的传输数 N < 2^R。控制器将在 DMA 周期结束时执行剩余的 N 次传输。

21.7.3 DMA 优先级

每个通道都可配置为使用默认优先级或高优先级，这通过设置 CHNL_PRIORITY_SET 寄存器来实现。

当控制器执行仲裁时，它将使用下列信息确定要服务的下一个通道：

- 分配给通道的优先级 (默认或高)
- 通道编号

通道编号 0 具有最高优先级，随着通道编号增大，通道优先级递减。控制器按升序通道编号服务于高优先级的所有使能通道，然后再服务于具有默认优先级的所有通道 (见表 344)。

表 344. DMA 通道优先级

通道编号	优先级设置	仲裁优先级降序
0	高	最高
1	高	第二高
...
20	高	...
0	默认	...

表 344. DMA 通道优先级 (续)

通道编号	优先级设置	仲裁优先级降序
1	默认	...
...
20	默认	最低

21.7.4 DMA 周期类型

通道控制数据结构中的 `cycle_ctrl` 位控制 DMA 控制器如何执行周期（见表 347）。

控制器使用本手册中介绍的四种周期类型：

- 无效
- 基本
- 自动请求
- 乒乓

有关其他周期类型，请参见 *ARM 微 DMA (PL230) 文档*。

对于所有周期类型，控制器在 2^R 次 DMA 传输后进行仲裁。如果将低优先级通道设为较大的 2^R 值，则将阻止其他所有通道执行 DMA 传输，直至低优先级 DMA 传输完成。因此，用户在设置 `channel_cfg` 数据结构中的 `R_power` 位时必须注意，不要让高优先级通道的延迟明显增加。

21.7.4.1 无效周期

控制器完成 DMA 周期后，会将周期类型设为无效，以防止重复同一 DMA 周期。

21.7.4.2 基本周期

在这种模式下，可将控制器配置为使用主用或备用通道控制数据结构。通道使能且控制器收到该通道的请求后，基本周期的数据流如下：

1. 控制器执行 2^R 传输。如果剩余传输次数为 0，则数据流将继续第 3 步。
2. 控制器的仲裁方式如下：
 - 如果高优先级通道请求服务，控制器将服务于该通道。
 - 如果外设或软件向控制器发出请求，控制器将继续第 1 步。
3. 控制器将该通道的 `dma_done[c]` 信号设为高电平并持续一个系统时钟周期。这对主机处理器表示 DMA 周期已完成。

21.7.4.3 自动请求周期

当控制器在此模式下工作时，控制器仅需要收到单个请求便能完成整个 DMA 周期。这允许发生较大的数据传输，而不明显增加服务高优先级请求的延迟，也不需要来自处理器或外设的多个请求。

自动请求周期一般用于存储器到存储器的请求。在这种情况下，软件在设置 DMA 控制数据结构后产生启动 2^R 个传输的请求。

在这种模式下，可将控制器配置为使用主用或备用通道控制数据结构。通道使能且控制器收到该通道的请求后，自动请求周期的数据流如下：

1. 控制器执行 2^R 传输。如果剩余传输次数为 0，则数据流将继续第 3 步。
2. 如果在 2^R 个传输后还有剩余传输，控制器将进行仲裁。如果当前通道 **c** 具有最高优先级，周期将继续第 1 步。
3. 控制器将该通道的 `dma_done[c]` 信号设为高电平并持续一个系统时钟周期。这对主机处理器表示 DMA 周期已完成。

21.7.4.4 乒乓周期

在这种模式下，控制器使用其中一个数据结构执行 DMA 周期，然后使用另一种数据结构执行 DMA 周期。控制器继续在主用和备用结构间切换，直至读取到无效的数据结构、用户将 `cycle_type` 重新设置为基本或主机处理器禁用通道。

在乒乓模式下，用户可以设置或重新设置两个通道数据结构（主用或备用）中的一个，同时将另一个通道数据结构用于活动传输。一个传输完成后，可使用预备的通道数据结构立即开始下一个传输，前提是高优先级通道没有要求服务。如果用户没有将未使用的通道控制数据结构重新设置用于传输，则周期类型将保持无效（使用该结构的最后一次传输结束时的值），乒乓周期完成。

乒乓周期可用于传输到外设或从外设传输，也可用于存储器到存储器的传输。

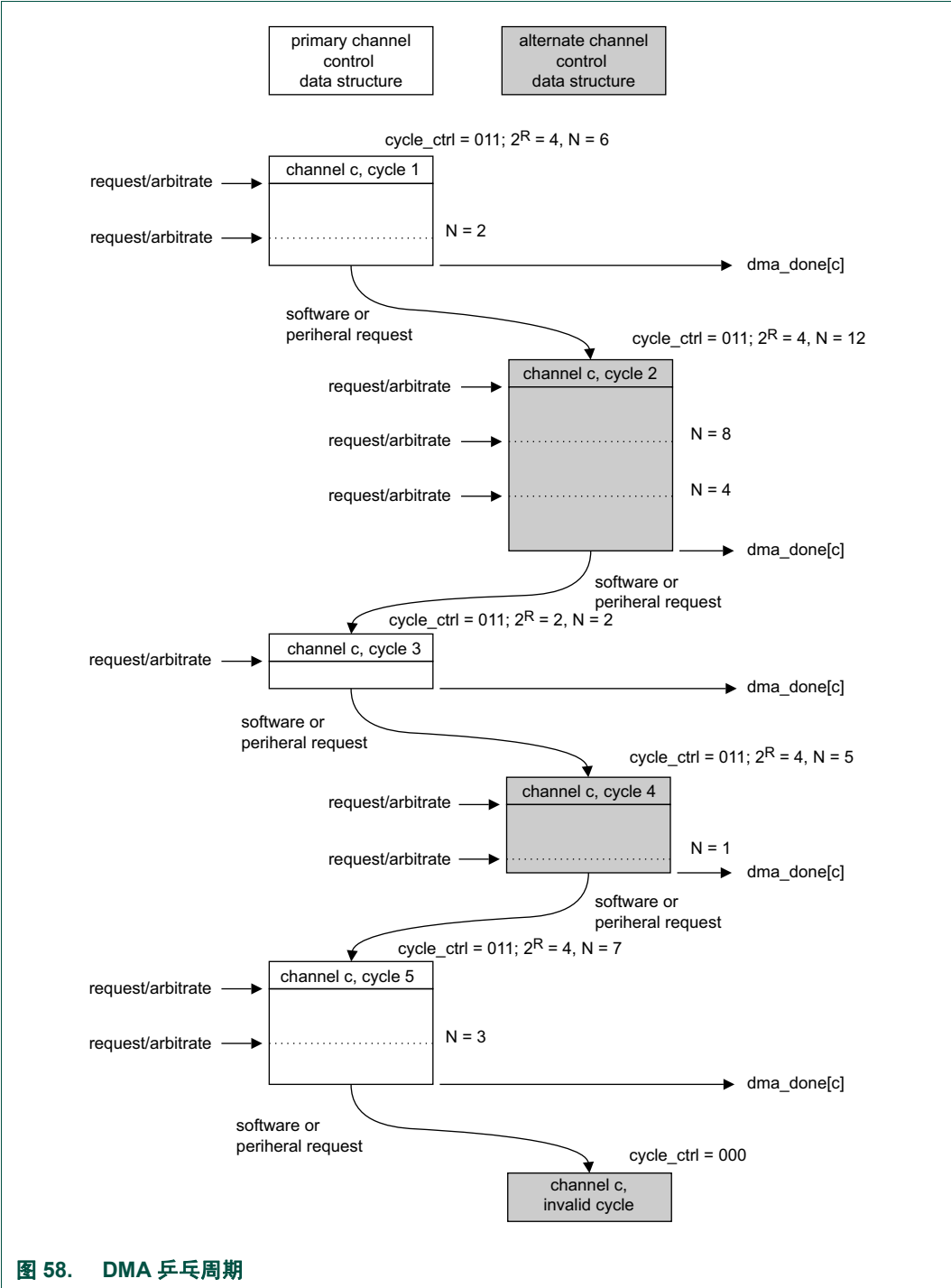


图 58. DMA 乒乓周期

21.7.5 DMA 控制

控制器使用 SRAM 来获得使能以访问两个指针和它针对每个通道所需的控制信息。通道控制信息包含在通道控制数据结构中，而 DMA 传输的源和目标地址则由源端和目标端指针来定义。

DMA 通道控制数据结构必须在 LPC122x 的 SRAM 内存中设置。数据结构必须在 1024 字节边界处对齐，以创建一个用于 21 个通道控制数据结构和 21 个备用通道控制数据结构的连续区域。主通道控制数据结构在 SRAM 中的基址必须在 0x1000 0000 到 0x1000 1F00 之间，以 0x400 为增量（见图 59）。

指向通道控制数据结构的指针必须在 CTRL_BASE_PTR 寄存器（表 325）中设置。

注：DMA 控制器不访问用户内存 (SRAM)，除非该通道被使能，且为该通道发起了传输。

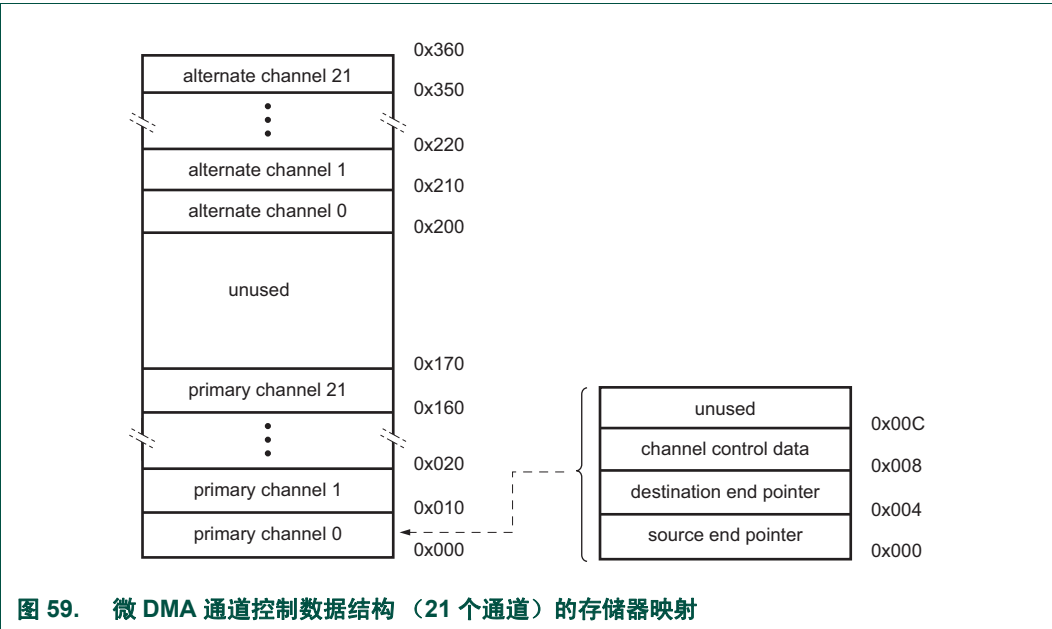


图 59. 微 DMA 通道控制数据结构（21 个通道）的存储器映射

21.7.5.1 源数据末端指针

src_data_end_ptr 存储器位置包含一个指向源数据末端地址的指针。

必须先为该存储器位置指定源数据的末端地址，控制器才能执行 DMA 传输。控制器在开始 2^R DMA 传输后，将读取该存储器位置。在 DMA 传输周期期间，控制器对末端地址进行递减计数，而且在每次仲裁之前，通道控制数据结构都将用剩余传输次数更新。

注：控制器不会对该存储器位置执行写操作。

表 345. src_data_end_ptr 的位分配

位	名称	描述
31:0	src_data_end_ptr	指向源数据末端地址的指针

21.7.5.2 目标数据末端指针

dst_data_end_ptr 存储器位置包含一个指向目标数据末端地址的指针。

必须先为该存储器位置指定目标数据的末端地址，控制器才能执行 DMA 传输。控制器在开始 2^R DMA 传输后，将读取该存储器位置，而且在每次仲裁之前，通道控制数据结构都将用剩余传输次数更新。

注：控制器不会对该存储器位置执行写操作。

表 346. dst_data_end_ptr 的位分配

位	名称	描述
31:0	dst_data_end_ptr	指向目标数据末端地址的指针

21.7.5.3 控制数据配置

对于每个 DMA 传输， channel_cfg 存储器位置都为控制器提供控制信息。

在 DMA 周期启动时，或 2^R DMA 传输开始时，控制器从 SRAM 存储器获取 channel_cfg 字。控制器执行 2^R 或 N 次传输后，将更新的 channel_cfg 字存储于 SRAM 中。

控制器不支持与 src_size 值不同的 dst_size 值。如果控制器检测到这些值不匹配，将为源和目标使用 src_size 值，而且在接下来更新 n_minus_1 字段时，也会将 dst_size 字段设为与 src_size 字段相同的值。

控制器完成 N 个传输后，它将 cycle_ctrl 字段设为 000 以指示 channel_cfg 数据无效。此时，通道配置在 SRAM 中被重写。这将防止控制器重复相同的 DMA 传输。

注：控制器在每次仲裁后更新 SRAM 中的通道控制数据结构。

表 347. channel_cfg 的位分配

位	名称	描述
2:0	cycle_ctrl	DMA 周期的操作模式。其中包括： 000: 停止。表示数据结构无效。 001: 基本。控制器必须先收到新的请求，然后再进入仲裁过程，以允许 DMA 周期完成。 010: 自动请求。控制器在仲裁过程中自动插入相应通道的请求。这意味着初始请求足以允许 DMA 周期完成。 011: 乒乓。控制器使用其中一个数据结构执行 DMA 周期。DMA 周期完成后，控制器使用另一个数据结构执行 DMA 周期。DMA 周期完成且主机处理器已更新原来的数据结构后，控制器将使用原来的数据结构执行 DMA 周期。控制器继续执行 DMA 周期，直至它读到无效的数据结构，或主机处理器将 cycle_ctrl 位改为 001 或 010。 100 - 111: 未使用。
3	next_useburst	控制当控制器在执行外设分散-集中并即将完成使用备用数据结构 DMA 周期时， chnl_useburst_set [C] 位是否置为 1。有关详情，请参阅 ARM 微 DMA (PL230) 文档。
13:4	n_minus_1	在 DMA 周期开始之前，这些位表示 DMA 周期包含的 DMA 传输总数。这些位必须根据 DMA 周期大小置位。 10 位值表示 DMA 传输次数减一。可能的值包括： 000000000 = 1 次 DMA 传输 000000001 = 2 次 DMA 传输 000000010 = 3 次 DMA 传输 000000011 = 4 次 DMA 传输 000000100 = 5 次 DMA 传输 ... 111111111 = 1024 次 DMA 传输 控制器在进入仲裁过程之前会立即更新此字段。这使得控制器能够存储对于完成 DMA 周期所必需的剩余 DMA 传输次数。

表 347. channel_cfg 的位分配 (续)

位	名称	描述
17:14	R_power	将这些位置位以控制在控制器重新仲裁之前可发生的 DMA 传输次数。 可能的仲裁率设置为： 0000: 每次完成 DMA 传输后仲裁。 0001: 完成 2 次 DMA 传输后仲裁。 0010: 完成 4 次 DMA 传输后仲裁。 0011: 完成 8 次 DMA 传输后仲裁。 0100: 完成 16 次 DMA 传输后仲裁。 0101: 完成 32 次 DMA 传输后仲裁。 0110: 完成 64 次 DMA 传输后仲裁。 0111: 完成 128 次 DMA 传输后仲裁。 1000: 完成 256 次 DMA 传输后仲裁。 1001: 完成 512 次 DMA 传输后仲裁。 1010-1111: 完成 1024 次 DMA 传输后仲裁。这表示在 DMA 传输期间不发生仲裁，因为最大传输大小为 1024。
20:18	src_prot_ctrl3:1	将这些位置位以在控制器读取源数据时控制 AHB Lite 的访问保护。 位 [20] 控制状态访问的方式如下： 0 = 访问不可缓存。 1 = 访问可缓存。 位 [19] 控制访问的方式如下： 0 = 访问不可缓冲。 1 = 访问可缓冲。 位 [18] 控制访问的方式如下： 0 = 访问无特权。 1 = 访问有特权。
23:21	dst_prot_ctrl3:1	将这些位置位以在控制器写入目标数据时控制 AHB Lite 的访问保护。为确保 DMA 控制器的正常操作，将位 21 到 23 置 0。 位 [23]: 保留。 位 [22] 控制访问的方式如下： 0 = 访问不可缓冲。 1 = 访问可缓冲。 位 [21] 控制访问的方式如下： 0 = 访问无特权。 1 = 访问有特权。
25:24	src_size	将这些位置位以匹配源数据大小： 00 = 字节 01 = 半字 10 = 字 11 = 保留

表 347. channel_cfg 的位分配 (续)

位	名称	描述
27:26	src_inc	<p>将这些位置位以控制源地址增量。地址增量取决于源数据宽度，具体如下：</p> <p>源数据宽度 = 字节 00 = 字节。 01 = 半字。 10 = 字。 11 = 无增量。地址保留设置为 src_data_end_ptr 存储器位置包含的值。</p> <p>源数据宽度 = 半字 00 = 保留。 01 = 半字。 10 = 字。 11 = 无增量。地址保留设置为 src_data_end_ptr 存储器位置包含的值。</p> <p>源数据宽度 = 字 00 = 保留。 01 = 保留。 10 = 字。 11 = 无增量。地址保留设置为 src_data_end_ptr 存储器位置包含的值。</p>
29:28	dst_size	目标数据大小。必须设为与源数据大小相同的值。
31:30	dst_inc	<p>目标地址增量。</p> <p>地址增量取决于源数据宽度，具体如下：</p> <p>源数据宽度 = 字节 00 = 字节。 01 = 半字。 10 = 字。 11 = 无增量。地址保留设置为 dst_data_end_ptr 存储器位置包含的值。</p> <p>源数据宽度 = 半字 00 = 保留。 01 = 半字。 10 = 字。 11 = 无增量。地址保留设置为 dst_data_end_ptr 存储器位置包含的值。</p> <p>源数据宽度 = 字 00 = 保留。 01 = 保留。 10 = 字。 11 = 无增量。地址保留设置为 dst_data_end_ptr 存储器位置包含的值。</p>

22.1 本章导读

所有 LPC122x 部件都提供 CRC 引擎。

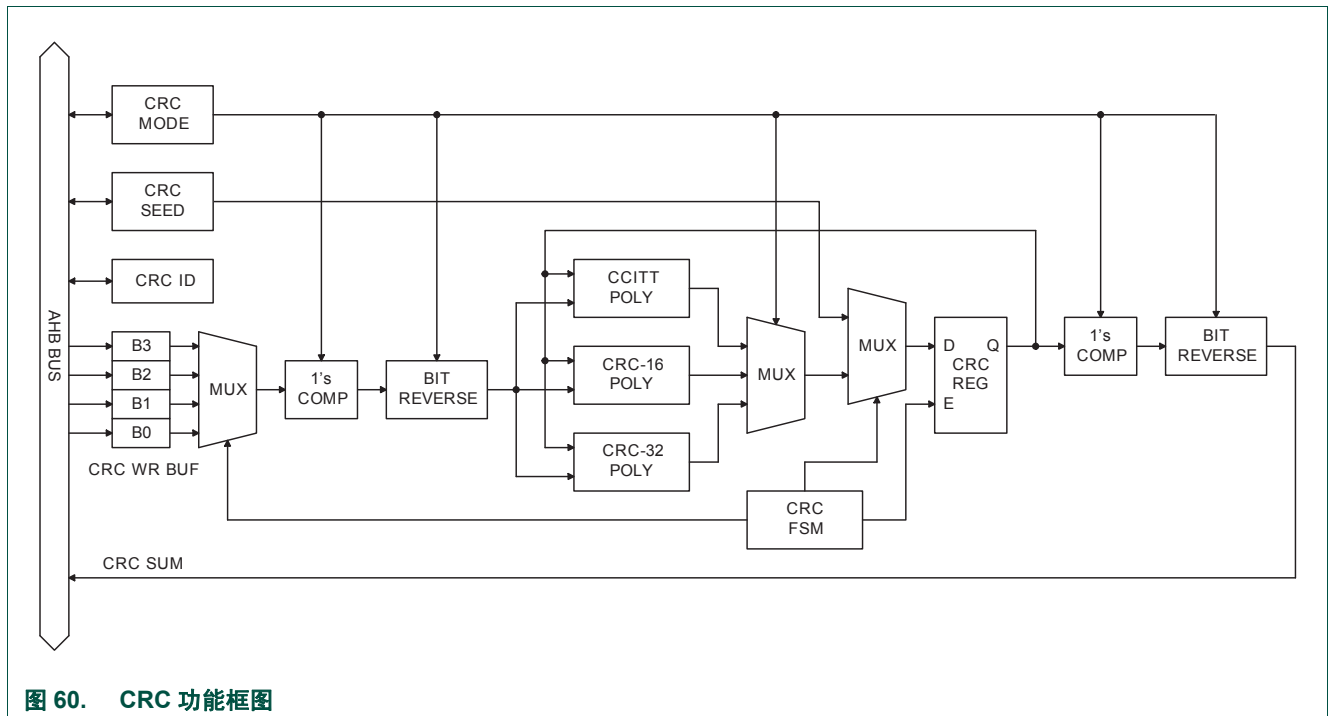
22.2 简介

带可编程多项式设置的循环冗余校验 (CRC) 生成器支持多个常用的 CRC 标准。为节约系统功耗和总线带宽，除使用 CPU 的软件 PIO 操作外，CRC 引擎还支持 DMA 传输。

22.3 特性

- 支持三个通用多项式 CRC-CCITT、CRC-16 和 CRC-32。
 - CRC-CCITT: $x^{16} + x^{12} + x^5 + 1$
 - CRC-16: $x^{16} + x^{15} + x^2 + 1$
 - CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- 针对输入数据及 CRC 和的位顺序取反和 1 的补码可编程设置。
- 可编程种子数设置。
- 支持 CPU PIO 或 DMA 背靠背传输。
- 接受每次写操作任意大小数据宽度：8、16 或 32 位。
 - 8 位写操作：1 周期操作
 - 16 位写操作：2 周期操作（8 位 x 2 周期）
 - 32 位写操作：4 周期操作（8 位 x 4 周期）

22.4 描述



22.5 寄存器描述

表 348. 寄存器简介: CRC 引擎 (基址 0x5007 0000)

名称	访问类型	地址偏移	描述	复位值
MODE	R/W	0x00	CRC 模式寄存器	0x0000 0000
SEED	R/W	0x04	CRC 种子寄存器	0x0000 FFFF
SUM	RO	0x08	CRC 校验和寄存器	0x0000 FFFF
WR_DATA	WO	0x08	CRC 数据寄存器	-

22.5.1 CRC 模式寄存器

表 349. CRC 模式寄存器 (MODE, 地址 0x5007 0000) 位描述

位	符号	描述	复位值
1:0	CRC_POLY	CRC 多项式: 1X= CRC-32 多项式 01= CRC-16 多项式 00= CRC-CCITT 多项式	00
2	BIT_RVS_WR	数据位顺序: 1= 对 CRC_WR_DATA 进行位顺序取反 （每字节） 0= 不对 CRC_WR_DATA 进行位顺序取反 （每字节）	0
3	CMPL_WR	数据补码: 1= 1 的补码 （针对 CRC_WR_DATA） 0= 不针对 CRC WR DATA 采用 1 的补码	0

表 349. CRC 模式寄存器（MODE，地址 0x5007 0000）位描述

位	符号	描述	复位值
4	BIT_RVS_SUM	CRC 和位顺序： 1= 对 CRC_SUM 进行位顺序取反 0= 不对 CRC_SUM 进行位顺序取反	0
5	CMPL_SUM	CRC 和补码： 1= 1 的补码（针对 CRC_SUM） 0= 不针对 CRC_SUM 采用 1 的补码	0
31:6	保留	读时始终为 0	0x0000000

22.5.2 CRC 种子寄存器

表 350. CRC 种子寄存器（SEED，地址 0x5007 0004）位描述

位	符号	描述	复位值
31:0	CRC_SEED	对该寄存器的写访问将按选定位顺序和 1 的补码预处理将 CRC 种子值加载到 CRC_SUM 寄存器。 注：对该寄存器的写访问将使正在进行的 CRC 计算无效。	0x0000 FFFF

22.5.3 CRC 校验和寄存器

该寄存器是只读寄存器，包含最近一次校验和。对该寄存器的读请求会被自动推迟有限个等待状态，直至结果有效，且校验和运算完成。

表 351. CRC 校验和寄存器（SUM，地址 0x5007 0008）位描述

位	符号	描述	复位值
31:0	CRC_SUM	可通过该寄存器按选定位顺序和 1 的补码后处理读取最近一次的 CRC 和。	0x0000 FFFF

22.5.4 CRC 数据寄存器

该寄存器是只写寄存器，包含将计算 CRC 和的数据块。

表 352. CRC 数据寄存器（WR_DATA，地址 0x5007 0008）位描述

位	符号	描述	复位值
31:0	CRC_WR_DATA	将使用写入该寄存器的数据按选定位顺序和 1 的补码预处理进行 CRC 计算。允许 8、16 或 32 位的任意写大小，并接受背靠背交易。	-

22.6 功能说明

下面的小节介绍了适用于每个支持的 CRC 标准的寄存器设置：

CRC-CCITT 设置

多项式 = $x^{16} + x^{12} + x^5 + 1$

种子值 = 0xFFFFF

对数据输入进行位顺序取反：否

对数据输入采用 1 的补码：否

对 CRC 和进行位顺序取反: 否

对 CRC 和采用 1 的补码: 否

CRC_MODE = 0x0000 0000

CRC_SEED = 0x0000 FFFF

CRC-16 设置

多项式 = $x^{16} + x^{15} + x^2 + 1$

种子值 = 0x0000

对数据输入进行位顺序取反: 是

对数据输入采用 1 的补码: 否

对 CRC 和进行位顺序取反: 是

对 CRC 和采用 1 的补码: 否

CRC_MODE = 0x0000 0015

CRC_SEED = 0x0000 0000

CRC-32 设置

多项式 = $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

种子值 = 0xFFFF FFFF

对数据输入进行位顺序取反: 是

对数据输入采用 1 的补码: 否

对 CRC 和进行位顺序取反: 是

对 CRC 和采用 1 的补码: 是

CRC_MODE = 0x0000 0036

CRC_SEED = 0xFFFF FFFF

23.1 本章导读

所有 LPC122x 部件都提供整数除法程序。

23.2 特性

- 性能优化的有符号 / 无符号整数除法
- 性能优化的有符号 / 无符号整数带余数除法
- 基于 ROM 的程序，可减少代码大小
- 支持最多 32 位的整数
- ROM 调用可轻松添加到 EABI 兼容函数，以使 C 语言中的“/”和“%”算符过载。

23.3 描述

整数除法程序执行算术整数除法运算，并可通过简单的 API 调用在应用程序代码中调用。

使用以下函数原型：

```
typedef struct { int quot; int rem; } idiv_return;

typedef struct { unsigned quot; unsigned rem; } udiv_return;

typedef struct{
    /* Signed integer division */
    int (*sdiv)(int numerator, int denominator);
    /* Unsigned integer division */
    unsigned(*udiv)(unsigned numerator, unsigned denominator);
    /* Signed integer division with remainder */
    idiv_return(*sdivmod)(int numerator, int denominator);
    /* Unsigned integer division with remainder */
    udiv_return (*udivmod)(unsigned numerator, unsigned denominator);
} LPC_ROM_DIV_STRUCT;

/* The value at this address is the entry to ROM Division API.
Once it is dereferenced, individual API functions can be used
*/
#define LPC_122x_DIVROM_LOC (0xFFC0000)
```

23.4 示例

23.4.1 初始化

下面列出的示例 C 代码显示了如何初始化 API 的 ROM 表指针。

```
/* Declare table pointer */
LPC_ROM_DIV_STRUCT const* pDivROM;
/* Assign pointer to table */
pDivROM = *((LPC_ROM_DIV_STRUCT**)LPC_122x_DIVROM_LOC);
```

23.4.2 有符号除法

下面列出的示例 C 代码显示了如何通过 ROM API 执行有符号整数除法。

```
/* Divide (-99) by (+6) */
int32_t result;
result = pDivROM->sidiv(-99, 6);
/* result now contains (-16) */
```

23.4.3 带余数的无符号除法

下面列出的示例 C 代码显示了如何通过 ROM API 执行带余数的无符号整数除法。

```
/* Modulus Divide (+99) by (+4) */
uidiv_return result;
result = pDivROM->uidivmod(+99, 4);
/* result.quot contains (+24) */
/* result.rem contains (+3) */
```

24.1 本章导读

所有 LPC122x 部件的调试功能都相同。

24.2 特性

- 支持 ARM 串行调试接口模式。
- 可直接对所有存储器、寄存器和外设进行调试。
- 调试阶段不需要目标资源。
- 4 个断点。4 个指令断点，可以用于重映射代码补丁的指令地址。2 个数据比较器，可用于将补丁的地址重映射到字面值。
- 2 个数据观察点，可用作跟踪触发器。

24.3 简介

ARM Cortex-M0 集成了调试功能。LPC122x 使用串行调试接口模式进行调试。

24.4 引脚说明

表 353. 串行调试接口引脚说明

引脚名称	类型	描述
SWCLK	输入	串行接口时钟。此引脚在串行调试接口模式下是调试逻辑的时钟 (SWDCLK)。
SWDIO	输入 / 输出	串行调试接口数据输入 / 输出。外部调试工具可通过 SWDIO 引脚与 ARM Cortex-M0 CPU 通信，并对其进行控制。

24.5 调试注意事项

24.5.1 调试限制

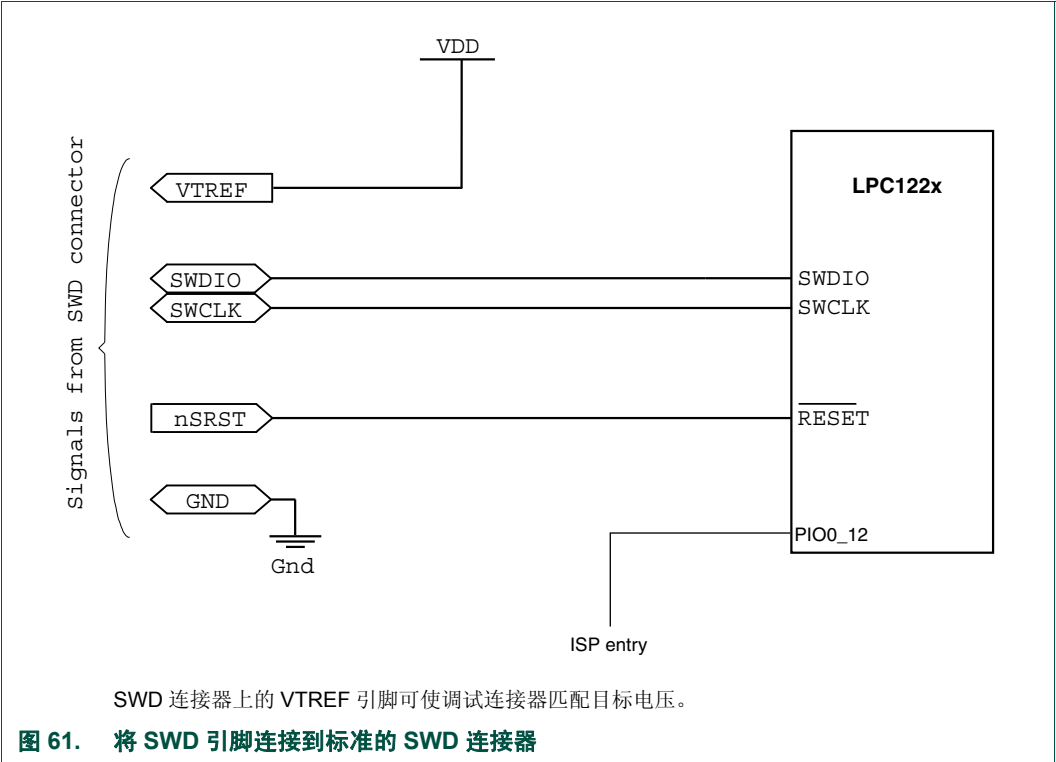
注意事项：用户应了解调试期间的某些限制。最重要的是：由于 ARM Cortex-M0 集成的限制，LPC122x 无法通过常规方式从深度睡眠模式中唤醒。建议在调试期间不要使用此模式。

另一个问题是调试模式会改变 ARM Cortex-M0 CPU 的低功耗状态，并波及到整个系统。这些差异意味着在调试过程中不应该进行功率测量，其结果会高于正常运行的情况。

在调试会话过程中，当 CPU 停止时系统节拍定时器将会自动停止，其他外设不受影响。

24.5.2 调试连接

就调试而言，需要访问 ISP 入口引脚 PIO0_1。此引脚可用于使部件从将禁用 SWD 端口的配置中恢复，如不当的 PLL 配置或重新配置 SWD 引脚作为 ADC 输入、复位后进入深度掉电模式等。此引脚可用于 GPIO 等其他功能，但在上电或复位时不应保持低电平。



25.1 简介

下面的参考材料以 ARM 《Cortex-M0 用户指南》为蓝本。只针对 LPC122x Cortex-M0 的具体实现做了细微的改动。

25.2 关于 Cortex-M0 处理器和内核外设

Cortex-M0 处理器是入门级 32 位 ARM Cortex 处理器，适用于多种嵌入式应用。该处理器包含以下特性，给开发者提供了极大的便利：

- 结构简单，容易学习和编程
- 功耗极低，运算效率高
- 出色的代码密度
- 确定、高性能的中断处理
- 向上兼容 Cortex-M 处理器系列。

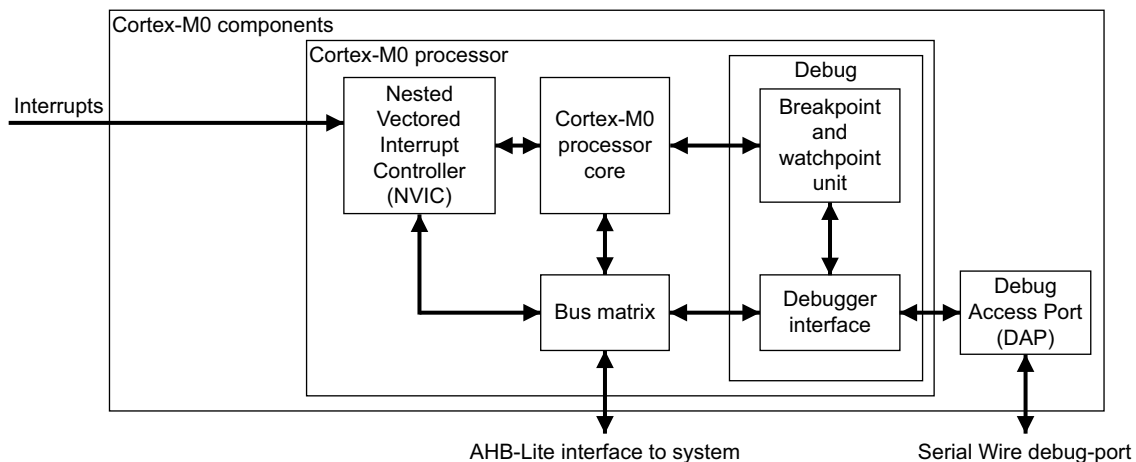


图 62. Cortex-M0 的具体实现

Cortex-M0 处理器基于一个高集成度、低功耗的 32 位处理器内核，采用了 3 级流水线冯·诺伊曼结构 (Von Neumann architecture)。通过简单、功能强大的指令集以及全面优化的设计（提供包括一个单周期乘法器在内的高端处理硬件），Cortex-M0 处理器可实现极高的能效。

Cortex-M0 处理器采用 ARMv6-M 架构，基于 16 位的 Thumb 指令集，并包含 Thumb-2 技术。提供了一个现代 32 位结构所希望的优秀性能，代码密度比其他 8 位和 16 位微控制器都要高。

Cortex-M0 处理器紧密集成了一个可配置的可嵌套中断向量控制器 (NVIC)，提供业界领先的中断性能。NVIC 具有以下功能：

- 包含一个非屏蔽中断 (NMI)。
- 提供零抖动中断选项。
- 提供四个中断优先级。

处理器内核与 NVIC 的紧密集成使得**中断服务程序 (ISR)**可以快速执行，极大地缩短了中断延迟。这是通过寄存器的硬件堆栈以及加载 - 乘和存储 - 乘操作的停止和重启来获得的。中断处理程序不需要任何汇编程序封装代码，不用消耗任何 ISR 代码。末尾连锁优化还极大地降低了一个 ISR 切换到另一个 ISR 时的开销。

为了优化低功耗设计，NVIC 还与睡眠模式相结合，提供深度睡眠功能，使整个器件迅速掉电。

25.2.1 系统级接口

Cortex-M0 处理器提供一个简单的系统级接口，使用 AMBA 技术来提供高速、低延迟的存储器访问。

25.2.2 集成的可配置调试

Cortex-M0 处理器执行一个完整的硬件调试解决方案，带有大量硬件断点和观察点选项。通过一个非常适合微控制器和其他小型封装器件的 2 针**串行调试接口 (SWD)**，提供了高系统透明度的处理器、存储器和外设执行。

25.2.3 Cortex-M0 处理器的特性小结

- 高代码密度，具有 32 位性能
- 工具和二进制代码与 Cortex-M 处理器系列向上兼容
- 集成了极低功耗的睡眠模式
- 高效的代码执行允许处理器时钟更低，或者延长睡眠模式的时间
- 单周期的 32 位硬件乘法器
- 零抖动中断处理
- 丰富的调试功能。

25.2.4 Cortex-M0 内核外设

Cortex-M0 内核外设：

NVIC — NVIC 是一个嵌入式中断控制器，支持低延迟的中断处理。

系统控制块 — **系统控制块 (SCB)** 是到处理器的编程模型接口。它提供系统实现信息及系统控制，包括配置、控制以及系统异常的报告。

系统定时器 — 系统定时器 SysTick 是一个 24 位的递减定时器。可以将其用作一个实时操作系统 (RTOS) 的节拍定时器，或者用作一个简单的计数器。

25.3 处理器

25.3.1 编程模型

本节描述了 Cortex-M0 的编程模型。除了个别内核寄存器的描述之外，本节还包含处理器模式和堆栈的相关信息。

25.3.1.1 处理器模式

处理器模式有：

线程模式 — 用于执行应用软件。处理器在退出复位后进入线程模式。

处理机模式 — 用于处理异常。处理器在完成所有异常处理后即返回到线程模式。

25.3.1.2 堆栈

处理器使用一个满递减堆栈。这就意味着堆栈指针指向堆栈存储器中的最后一个堆栈项。当处理器将一个新的项压入堆栈时，堆栈指针递减，然后将该项写入新的存储器单元。处理器执行两个堆栈，主堆栈和进程堆栈，两个堆栈有自己独立的堆栈指针副本，见[章节 25.3.1.3.2](#)。

在线程模式下，CONTROL 寄存器控制处理器使用主堆栈还是进程堆栈，见[章节 25–25.3.1.3.7](#)。在处理程序模式下，处理器始终使用主堆栈。处理器操作的选择如下：

表 354. 处理器模式和堆栈使用选项小结

处理器模式	用于执行	使用的堆栈
线程模式	应用程序	主堆栈或进程堆栈 见章节 25–25.3.1.3.7
处理机模式	异常处理程序	主堆栈

25.3.1.3 内核寄存器

处理器内核寄存器有：

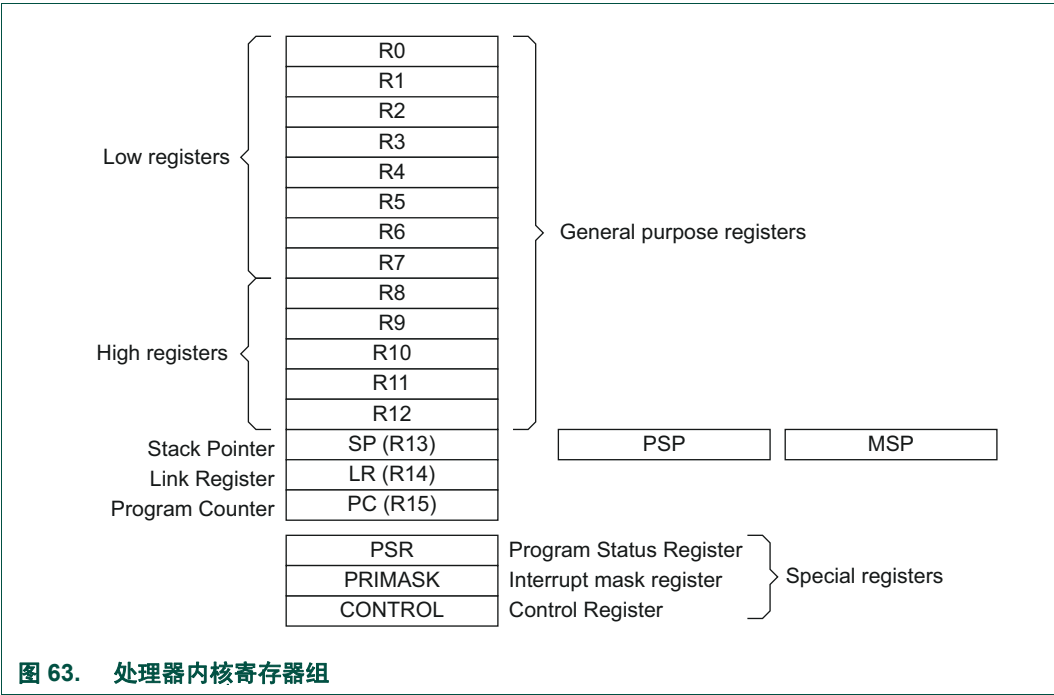


图 63. 处理器内核寄存器组

表 355. 内核寄存器组小结

名称	类型 ^[1]	复位值	描述
R0-R12	RW	不可知	章节 25–25.3.1.3.1
MSP	RW	见文中描述	章节 25–25.3.1.3.2
PSP	RW	不可知	章节 25–25.3.1.3.2
LR	RW	不可知	章节 25–25.3.1.3.3
PC	RW	见文中描述	章节 25–25.3.1.3.4
PSR	RW	不可知 ^[2]	表 25–356
APSR	RW	不可知	表 25–357
IPSR	RO	0x00000000	表 358
EPSR	RO	不可知 ^[2]	表 25–359
PRIMASK	RW	0x00000000	表 25–360
控制	RW	0x00000000	表 25–361

[1] 描述在线程模式和处理机模式下程序执行期间的访问类型。调试访问可以不同。

[2] 位 [24] 为 T 位，从复位向量的位 [0] 加载。

25.3.1.3.1 通用寄存器

R0-R12 是供数据操作使用的 32 位通用寄存器。

25.3.1.3.2 堆栈指针

堆栈指针 (SP) 是寄存器 R13。在线程模式中，CONTROL 寄存器的位 [1] 指示了要使用的堆栈指针：

- 0 = 主堆栈指针 (MSP)。这是复位值。
- 1 = 进程堆栈指针 (PSP)。

复位时，处理器将地址 0x00000000 的值加载到 MSP 中。

25.3.1.3.3 链接寄存器

链接寄存器 (LR) 是寄存器 R14。它存储子程序、函数调用以及异常的返回信息。复位时，LR 值不可知。

25.3.1.3.4 程序计数器

程序计数器 (PC) 是寄存器 **R15**。它包含当前程序地址。复位时，处理器将复位向量（地址：**0x00000004**）的值加载到 **PC**。值的位 **[0]** 复位时被加载到 **EPSR** 的 **T** 位，必须为 **1**。

25.3.1.3.5 程序状态寄存器

程序状态寄存器 (PSR) 由下列 3 种寄存器组合而成:

- 应用程序状态寄存器 (APSR)
- 中断程序状态寄存器 (IPSR)
- 执行程序状态寄存器 (EPSR)。

在 32 位的 PSR 中，这 3 个寄存器的位域分配互斥。PSR 的位分配如下所示：

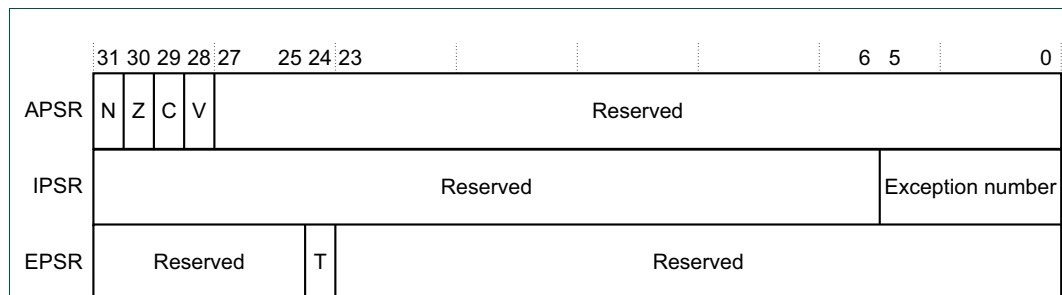


图 64. APSR、IPSR、EPSR 寄存器的位分配

这 3 个寄存器可以单独访问，也可以 2 个一组或 3 个一组进行访问，访问时，将寄存器名称作为 MSR 或 MRS 指令的一个自变量。例如：

- 使用寄存器名称 `PSR`，用 `MRS` 指令来读所有寄存器
- 使用寄存器名称 `APSR`，用 `MSR` 指令来写 `APSR`。

PSR 的组合和属性如下所示:

表 356. PSR 寄存器组合

寄存器	类型	组合
PSR	RW ^{[1][2]}	APSR、EPSR 和 IPSR
IEPSR	RO	EPSR 和 IPSR
IAPSR	RW ^[1]	APSR 和 IPSR
EAPSR	RW ^[2]	APSR 和 EPSR

[1] 处理器忽略对 IPSR 位的写操作。

[2] 读取 EPSR 位将返回零，处理器忽略对这些位的写操作。

有关如何访问程序状态寄存器的更多信息，请参考指令描述[章节 25-25.4.7.6](#) 和[章节 25-25.4.7.7](#)。

应用程序状态寄存器：APSR 包含执行完前面的指令后条件标志的当前状态。有关寄存器的属性请见[表 25–355](#) 中的寄存器小结。寄存器的位分配如下所示：

表 357. APSR 位分配

位	名称	功能
[31]	N	负值标志
[30]	Z	零值标志
[29]	C	进位或借位标志
[28]	V	溢出标志
[27:0]	-	保留

有关 APSR 的负值、零值、进位或借位以及溢出标志的更多信息，请参考[章节 25.4.4.1.4](#)。

中断程序状态寄存器：IPSR 包含当前**中断服务程序 (ISR)** 的异常编号。有关寄存器的属性请见[表 25–355](#) 中的寄存器小结。寄存器的位分配如下所示：

表 358. IPSR 的位分配

位	名称	功能
[31:6]	-	保留
[5:0]	异常编号	这是当前异常的编号： 0 = 线程模式 1 = 保留 2 = NMI 3 = HardFault 4-10 = 保留 11 = SVCall 12、13 = 保留 14 = PendSV 15 = SysTick 16 = IRQ0 . . . 47 = IRQ31 48-63 = 保留。 更多信息请见 章节 25–25.3.3.2 。

异常程序状态寄存器：EPSR 包含 Thumb 状态位。

有关寄存器的属性请见[表 25–355](#) 中的寄存器小结。寄存器的位分配如下所示：

表 359. EPSR 位分配

位	名称	功能
[31:25]	-	保留
[24]	T	Thumb 状态位
[23:0]	-	保留

应用软件尝试使用 MRS 指令直接读取 EPSR 时始终返回零。使用 MSR 指令尝试写入 EPSR 时将被忽略。故障处理程序可检查堆栈 PSR 中的 EPSR 值以确定故障原因。参见[章节 25–25.3.3.6](#)。下列情况可将 T 位清零：

- BLX、BX 和 POP{PC} 指令
- 异常返回时恢复被压入栈中的 xPSR 值
- 进入异常时向量值的位 [0]。

在 T 位为 0 时尝试执行指令将导致 HardFault 异常或锁定。更多信息请见[章节 25–25.3.4.1](#)。

可中断 - 可重启指令：可中断 - 可重启指令有 LDM 和 STM。如果在执行这两条中的其中一条指令的过程中出现中断，处理器就终止指令的执行。

在处理完中断后，处理器再从头开始重新执行指令。

25.3.1.3.6 异常屏蔽寄存器

异常屏蔽寄存器禁止处理器处理异常。当异常可能影响到实时任务或要求连续执行的代码序列时，异常就会被禁用。

可以使用 MSR 和 MRS 指令、或 CPS 指令改变 PRIMASK 的值来禁用或重新启用异常。更多信息请见[章节 25–25.4.7.6](#)、[章节 25–25.4.7.7](#) 和 [章节 25–25.4.7.2](#)。

优先级屏蔽寄存器：PRIMASK 寄存器阻止优先级可配置的所有异常被激活。有关寄存器的属性请见[表 25–355](#) 中的寄存器小结。寄存器的位分配如下所示：

表 360. PRIMASK 寄存器的位分配

位	名称	功能
[31:1]	-	保留
[0]	PRIMASK	0 = 无影响 1 = 阻止优先级可配置的所有异常被激活。

25.3.1.3.7 CONTROL 寄存器

CONTROL 寄存器控制着处理器处于线程模式时所使用的堆栈。有关寄存器的属性请见[表 25–355](#) 中的寄存器小结。寄存器的位分配如下所示：

表 361. CONTROL 寄存器的位分配

位	名称	功能
[31:2]	-	保留
[1]	有效堆栈指针	定义当前的堆栈： 0 = MSP 是当前的堆栈指针 1 = PSP 是当前的堆栈指针 在处理器模式中，这个位读出为零，写操作被忽略。
[0]	-	保留。

处理机模式始终使用 MSP，因此，在处理机模式下，处理机忽略对 CONTROL 寄存器的有效堆栈指针位执行的明确的写操作。异常进入和返回机制会将 CONTROL 寄存器更新。

在一个 OS 环境中，推荐运行在线程模式中的线程使用进程堆栈，内核和异常处理器使用主堆栈。

默认情况下，线程模式使用 MSP。要将线程模式中使用的堆栈指针切换为 PSP，请使用 MSR 指令将有效栈指针位置为 1，见[章节 25–25.4.7.6](#)。

注：当更改堆栈指针时，软件必须在 MSR 指令后立即使用一个 ISB 指令。这将确保 ISB 之后的指令执行时使用新的堆栈指针，见[章节 25–25.4.7.5](#)。

25.3.1.4 异常和中断

Cortex-M0 处理器支持中断和系统异常。处理器和**嵌套向量中断控制器 (NVIC)** 划分所有异常的优先级，并对所有异常进行处理。一个中断或异常会改变软件控制的正常流程。处理器使用处理机模式来处理除复位之外的所有异常，更多信息请见[章节 25-25.3.3.6.1](#) 和[章节 25-25.3.3.6.2](#)。

NVIC 寄存器控制中断处理。更多信息请见[章节 25-25.5.2](#)。

25.3.1.5 数据类型

处理器：

- 支持下列数据类型：
 - 32 位字
 - 16 位半字
 - 8 位字节
- 管理所有数据存储器访问都采用小端模式。指令存储器和**专用外设总线 (PPB)** 访问始终是小端模式。更多信息请见[章节 25-25.3.2.1](#)。

25.3.1.6 Cortex 微控制器软件接口标准

ARM 为编程 Cortex-M0 微控制器编程提供了 **Cortex 微控制器软件接口标准 (CMSIS)**。CMSIS 是器件驱动程序库的一个组成部分。

CMSIS 为 Cortex-M0 微控制器系统定义了：

- 一种通用方式来：
 - 访问外设寄存器
 - 定义异常向量
- 以下两项的名称：
 - 内核外设寄存器
 - 内核异常向量
- 一个 RTOS 内核的器件独立的接口。

CMSIS 包含 Cortex-M0 处理器中内核外设的地址定义和数据结构。还包含有组成 TCP/IP 堆栈和 Flash 文件系统的中间件元件的可选接口。

通过支持模板代码的重复使用以及将不同中间件厂商提供的符合 CMSIS 的软件组件组合起来，CMSIS 大大简化了整个软件开发过程。软件厂商可以扩展 CMSIS，使其包含各个厂商的外设定义以及这些外设的存取函数。

本文档包含了 CMSIS 定义的寄存器名称，并对处理器内核和内核外设相关的 CMSIS 函数进行了简单描述。

注：本文档使用 CMSIS 定义的寄存器缩略名称。在某些情况下，这些名称与其他文档中可能用到的架构缩略名称不同。

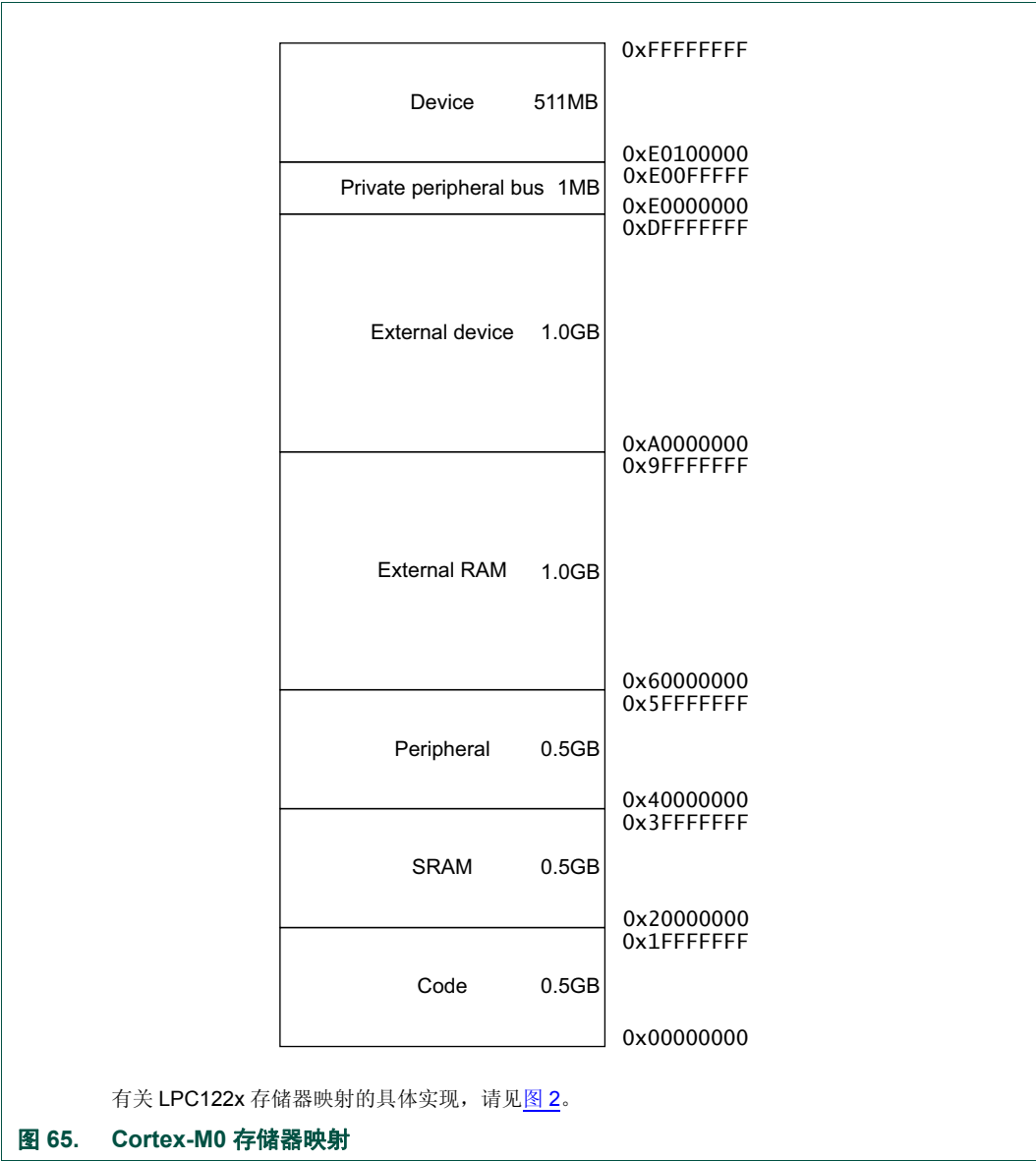
下面各节给出了有关 CMSIS 的更多信息：

- [章节 25.3.5.3 “电源管理编程提示”](#)

- [章节 25.4.2 “内部函数”](#)
- [章节 25.5.2.1 “使用 CMSIS 访问 Cortex-M0 NVIC 寄存器”](#)
- [章节 25.5.2.8.1 “NVIC 编程提示”](#)

25.3.2 存储器模型

本节描述处理器存储器映射以及存储器访问的行为。处理器有一个固定的存储器映射，提供有高达 4GB 的可寻址存储空间。存储器映射是：



处理器为内核外设寄存器保留了专用外设总线 (PPB) 地址范围空间，见[章节 25–25.2](#)。

25.3.2.1 存储区、类型和属性

存储器映射分成多个区域。每个区域有一个定义好的存储器类型，某些区域还有附加的存储器属性。存储器类型和属性决定了各个区域的访问行为。

存储器类型有：

常规存储器 — 处理器为了提高效率，可以重新对交易进行排序，或者刻意地进行读取。

Device 存储器 — 处理器保护与 Device 或强秩序存储器的其他交易相关的交易秩序。

强秩序存储器 — 处理器保护与所有其他交易相关的交易秩序。

Device 存储器和强秩序存储器的不同秩序要求意味着，存储器系统可以缓冲一个对 Device 存储器的写操作，但不得缓冲对强秩序存储器的写操作。

附加的存储器属性包括：

永不执行 (XN) — 表示处理器阻止指令访问。当执行从存储器的 XN 区提取出来的指令时，产生一个 HardFault 异常。

25.3.2.2 存储器系统的存储器访问秩序

对于大多数由明确的存储器访问指令引发的存储器访问，存储器系统都不保证访问秩序与指令的编写顺序完全一致，只要所有访问秩序的重新安排不影响指令序列的操作就行。一般情况下，如果两个存储器访问的顺序必须与两条存储器访问指令编写的顺序完全一致程序才能正确执行，软件就必须在两条存储器访问指令之间插入一条内存屏障指令，请见[章节 25-25.3.2.4](#)。

但是，存储器系统不保证 Device 存储器和强秩序存储器的一些访问秩序。对于两条存储器访问指令 A1 和 A2，如果 A1 的编写顺序在前，两条指令所引发的存储器访问顺序为：

A1 \ A2		Normal access	Device access		Strongly-ordered access
			Non-shareable	Shareable	
Normal access		-	-	-	-
Device access, non-shareable		-	<	-	<
Device access, shareable		-	-	<	<
Strongly-ordered access		-	<	<	<

图 66. 存储器秩序限制

其中：

- — 表示存储器系统不保证访问秩序。
- < — 表示观察到访问顺序与指令编写顺序一致，即， A1 总是在 A2 之前。

25.3.2.3 存储器访问行为

存储器映射中每个区域的访问行为如下：

表 362. 存储器访问行为

地址范围	存储器区域	存储器类型 ^[1]	XN ^[1]	描述
0x00000000-0x1FFFFFFF	Code	常规存储器	-	程序代码的可执行区域。也可以把数据保存到这里。
0x20000000-0x3FFFFFFF	SRAM	常规存储器	-	数据的可执行区域。也可以把代码保存到这里。
0x40000000-0x5FFFFFFF	外设	Device 存储器	XN	外部设备存储器。
0x60000000-0x9FFFFFFF	外部 RAM	常规存储器	-	数据的可执行区域。
0xA0000000-0xDFFFFFFF	外部设备	Device 存储器	XN	外部设备存储器。
0xE0000000-0xE00FFFFF	专用外设总线	强秩序存储器	XN	这个区域包括 NVIC、系统定时器和系统控制块。这个区域只能使用字访问。
0xE0100000-0xFFFFFFFF	Device 存储器	Device 存储器	XN	厂商提供的特定存储器。

[1] 更多信息请见[章节 25–25.3.2.1](#)。

Code、SRAM 和外部 RAM 区域可以保存程序。

25.3.2.4 软件的存储器访问秩序

程序流程的指令顺序不能担保相应的存储器交易顺序。这是因为：

- 为了提高效率，处理器可以将一些存储器访问的顺序重新安排，只要不影响指令序列的操作就行
- 存储器映射中的存储器或设备可能有不同的等待状态
- 某些存储器访问被缓冲，或者是刻意为之的。

[章节 25–25.3.2.2](#) 节描述了存储器系统在哪些情况下能保证存储器访问的秩序。否则，如果存储器访问的秩序十分重要，软件就必须插入一些内存屏障指令来强制保持存储器访问的秩序。处理器提供了以下内存屏障指令：

DMB — 数据存储器屏障 (DMB) 指令保证先完成未处理的存储器交易，再执行后面的存储器交易，见[章节 25–25.4.7.3](#)。

DSB — 数据同步屏障 (DSB) 指令保证先完成未处理的存储器交易，再执行后面的指令，见[章节 25–25.4.7.4](#)。

ISB — 指令同步屏障 (ISB) 保证所有已完成的存储器交易的结果都能被后面的指令辨认出来。见[章节 25–25.4.7.5](#)。

下面是内存屏障指令使用的一些例子：

向量表 — 如果程序改变了向量表中的一项，然后又使能了相应的异常，那么就在两个操作之间插入一条 DMB 指令。这就确保了，如果异常在被使能后立刻被采纳，处理器能使用新的异常向量。

自修改代码 — 如果一个程序包含自修改代码，代码修改之后在程序中立刻使用一条 ISB 指令。这就确保了后面的指令执行使用的是更新后的程序。

存储器映射切换 — 如果系统包含一个存储器映射切换机制，在切换存储器映射之后使用一条 DSB 指令。这就确保了后面的指令执行使用的是更新后的存储器映射。

对强秩序存储器（例如，系统控制块）执行的存储器访问不需要使用 DMB 指令。

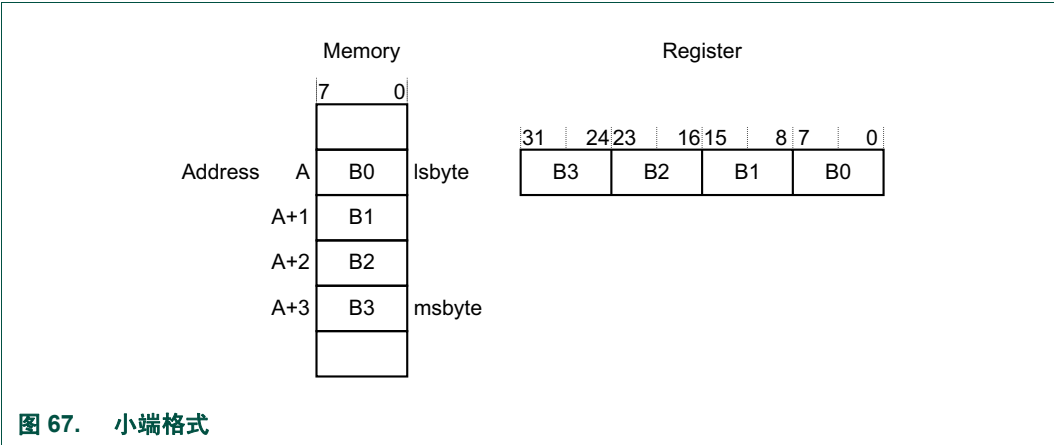
处理器保护与所有其他交易相关的交易秩序。

25.3.2.5 存储器的字节存储顺序

处理器看到的存储器是一个从零开始、编号逐次递增的字节集合。例如，字节 0-3 存放第一个保存的字，字节 4-7 存放第二个保存的字。[章节 25-25.3.2.5.1](#) 描述了数据的字在存储器中是如何存放的。

25.3.2.5.1 小端格式

在小端格式中，处理器将字的**最低有效字节 (lsbyte)** 保存在编号最小的字节中，**最高有效字节 (msbyte)** 保存在编号最大的字节中。例如：



25.3.3 异常模型

本节描述异常模型。

25.3.3.1 异常状态

每个异常都处于下面其中一种状态：

无效 — 异常无效，也未挂起。

挂起 — 异常正在等待处理器处理。

一个外设或软件的中断请求可以改变相应的挂起中断的状态。

有效 — 一个异常正在被处理器处理，但处理尚未结束。

一个异常处理程序可以中断另一个异常处理程序的执行。在这种情况下，两个异常都处于有效状态。

有效和挂起 — 异常正在被处理器处理，而且有一个来自同一个异常源的异常正在等待处理。

25.3.3.2 异常类型

异常类型有：

复位 — 复位在上电或热复位时启动。异常模型将复位当做一种特殊形式的异常来对待。当复位产生时，处理器的操作停止（可能停止在一条指令的任何一点上）。当复位撤销时，从向量表中复位项提供的地址处重新启动执行。执行在线程模式下重新启动。

NMI — 一个**不可屏蔽中断 (NMI)** 可以由外设产生，也可以由软件来触发。这是除复位之外优先级最高的异常。NMI 永远使能，优先级固定为 -2。NMI 不能：

- 被屏蔽，它的执行也不能被其他任何异常中止；
- 被除复位之外的任何异常抢占。

HardFault — HardFault 是由于在正常操作过程中或在异常处理过程中出错而出现的一个异常。HardFault 的优先级固定为 -1，表明它的优先级要高于任何优先级可配置的异常。

SVC **Call** — **管理程序调用 (SVC)** 异常是一个由 svc 指令触发的异常。在 OS 环境下，应用程序可以使用 svc 指令来访问 OS 内核函数和器件驱动程序。

PendSV — PendSV 是一个中断驱动的系统级服务请求。在 OS 环境下，当没有其他异常有效时，使用 PendSV 来进行任务切换。

SysTick — SysTick 是一个系统定时器到达零时产生的异常。软件也可以产生一个 SysTick 异常。在 OS 环境下，处理器可以将这个异常用作系统节拍。

中断 (IRQ) — 中断（或 IRQ）是外设发出的一个异常，或者是由软件请求产生的一个异常。所有中断都与指令执行不同步。在系统中，外设使用中断来与处理器通信。

表 363. 各种异常类型的特性

异常编号 [1]	IRQ 编号 [1]	异常类型	优先级	向量地址 [2]
1	-	复位	-3, 优先级最高	0x00000004
2	-14	NMI	-2	0x00000008
3	-13	HardFault	-1	0x0000000C
4-10	-	保留	-	-
11	-5	SVC	可配置 [3]	0x0000002C
12-13	-	保留	-	-
14	-2	PendSV	可配置 [3]	0x00000038
15	-1	SysTick	可配置 [3]	0x0000003C
16 和大于 16 的值	0 和大于 0 的值	中断 (IRQ)	可配置 [3]	0x00000040 以及更高的地址 [4]

[1] 为了简化软件层，CMSIS 只使用 IRQ 编号，因此，对除中断外的其他异常都使用负值。IPSR 返回异常编号，见表 25-358。

[2] 更多信息请见章节 25.3.3.4。

[3] 见章节 25-25.5.2.6。

[4] 地址值以 4 为步长，逐次递增。

对于除复位之外的异步异常，在异常被触发和处理器进入异常处理程序之间，处理器可以执行条件指令。

被特许的软件可以将表 25-363 中列出的优先级可配置的异常禁能，见章节 25-25.5.2.3。

有关 HardFault 的更多信息，请见章节 25-25.3.4。

25.3.3.3 异常处理程序

处理器使用以下处理程序来处理异常：

中断服务程序 (ISR) — 中断 IRQ0-IRQ31 是由 ISR 来处理的异常。

故障处理程序 — HardFault 是唯一一个由故障处理程序来处理的异常。

系统处理程序 — NMI、PendSV、SVCall、SysTick 和 HardFault 是由系统处理程序来处理的全部异常。

25.3.3.4 向量表

向量表包含堆栈指针的复位值以及所有向量处理程序的起始地址（也称为异常向量）。图 25-68 显示了异常向量在向量表中的放置顺序。每个向量的最低有效位必须为 1，表明异常处理程序都是用 Thumb 代码编写的。

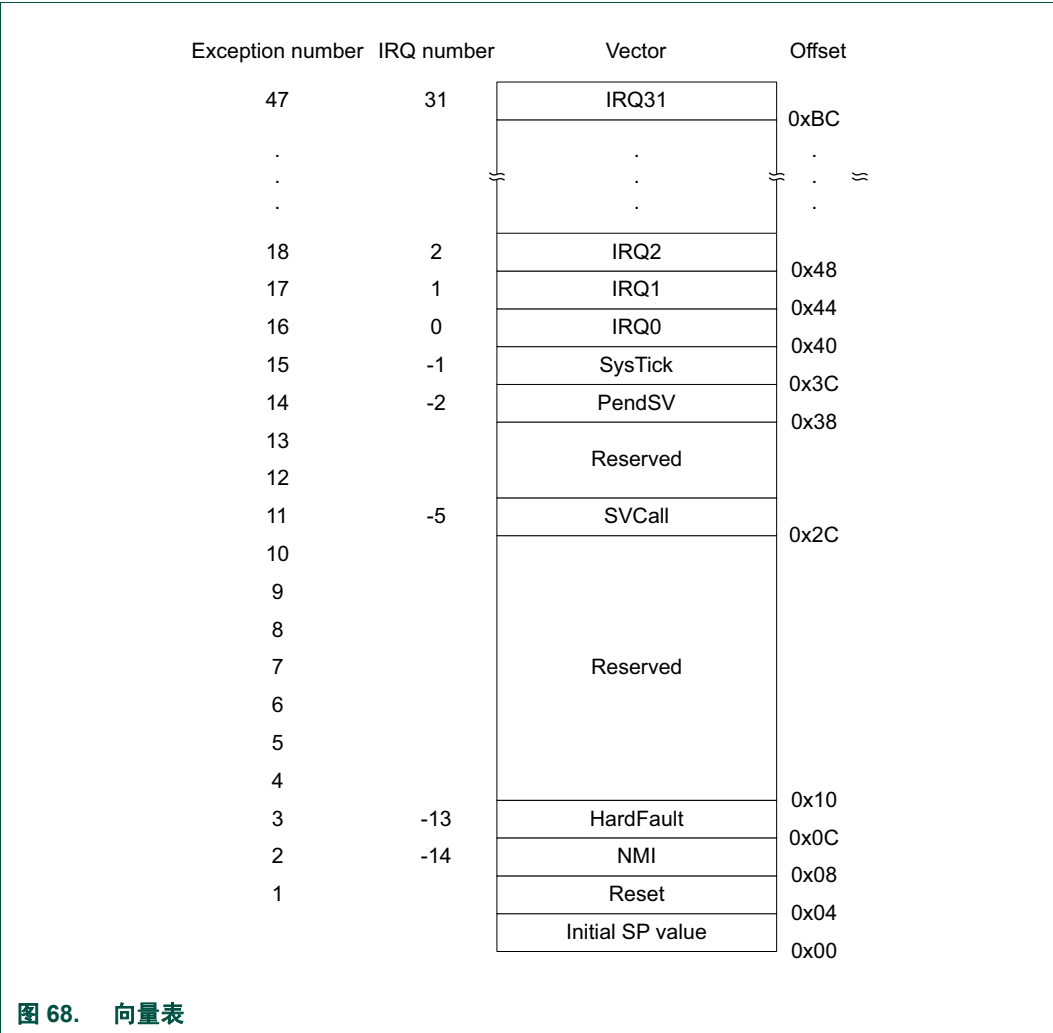


图 68. 向量表

向量表的地址固定为 0x00000000。

25.3.3.5 异常优先级

如表 25-363 所示，每个异常都有对应的优先级：

- 优先级值越小表示优先级越高
- 除复位、HardFault 和 NMI 之外，所有异常的优先级都是可配置的。

如果软件不配置任何优先级，那么，所有优先级可配置的异常的优先级就都为 0。有关配置异常优先级的信息，请见

- [章节 25-25.5.3.7](#)
- [章节 25-25.5.2.6](#)

优先级值的配置范围为 0 – 192，各值以 64 为步长。复位、HardFault 和 NMI 这些有固定负优先级值的异常，其优先级高于任何其他异常。

给 IRQ[0] 分配一个高优先级值、给 IRQ[1] 分配一个低优先级值就意味着 IRQ[1] 的优先级高于 IRQ[0]。如果 IRQ[1] 和 IRQ[0] 都有效，先处理 IRQ[1]。

如果多个挂起的异常具有相同的优先级，异常编号越小的挂起异常优先处理。例如，如果 IRQ[0] 和 IRQ[1] 均挂起，并且两者的优先级相同，那么先处理 IRQ[0]。

当处理器正在执行一个异常处理程序时，如果出现一个更高优先级的异常，那么这个异常就被抢占。如果出现的异常的优先级和正在处理的异常的优先级相同，这个异常就不会被抢占，与异常的编号大小无关。但是，新中断的状态就变为挂起。

25.3.3.6 异常的进入和返回

描述异常处理时使用了下列术语：

抢占 — 当处理器正在执行一个异常处理程序时，如果一个异常的优先级比正在处理的异常的优先级更高，那么低优先级的异常就被抢占。

当一个异常抢占另一个异常时，这些异常就被称为嵌套异常。更多信息请见[章节 25-25.3.3.6.1](#)。

返回 — 当异常处理程序结束，并且满足以下条件时，异常就返回：

- 没有优先级足够高的挂起异常要处理
- 已结束的异常处理程序没有在处理一个迟来的异常。

处理器弹出堆栈，处理器状态恢复到中断出现之前的状态，更多信息请见[章节 25-25.3.3.6.2](#)。

末尾连锁 — 这个机制加速了异常的处理。当一个异常处理程序结束时，如果一个挂起的异常满足异常进入的要求，就跳过堆栈弹出，控制权移交给新的异常处理程序。

迟来 — 这个机制加速了抢占的处理。如果一个高优先级的异常在前一个异常正在保存状态的过程中出现，处理器就转去处理更高优先级的异常，开始提取这个异常的向量。状态保存不受迟来异常的影响，因为两个异常保存的状态相同。从迟来异常的异常处理程序返回时，要遵守正常的末尾连锁规则。

25.3.3.6.1 异常的进入

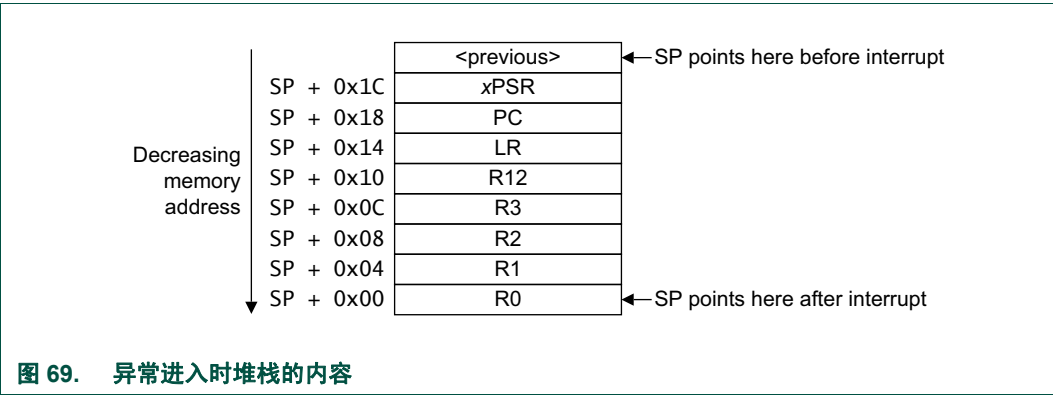
当有一个优先级足够高的挂起异常存在，并且满足下面的任何一个条件，就进入异常处理：

- 处理器处于线程模式
- 新异常的优先级高于正在处理的异常，这时，新异常就抢占了正在处理的异常。

当一个异常抢占另一个异常时，异常就被嵌套。

优先级足够高的意思是该异常的优先级比屏蔽寄存器中所限制的任何一个异常组的优先级都要高，见[章节 25-25.3.1.3.6](#)。优先级比这个异常要低的异常被挂起，但不被处理器处理。

当处理器处理异常时，除非异常是一个末尾连锁异常或迟来的异常，否则，处理器都把信息压入到当前的堆栈中。这个操作被称为**入栈**，8 个数据字的结构被称为**栈帧**。栈帧包含以下信息：



入栈后，堆栈指针立刻指向栈帧的最低地址单元。栈帧按照双字地址对齐。

栈帧包含返回地址。这是被中止的程序中下条指令的地址。这个值在异常返回时返还给 PC，使被中止的程序恢复执行。

处理器执行一次向量提取，从向量表中读出异常处理程序的起始地址。当入栈结束时，处理器开始执行异常处理程序。同时，处理器向 LR 写入一个 EXC_RETURN 值。这个值指示了栈帧对应哪个堆栈指针以及在异常出现之前处理器处于什么工作模式。

如果在异常进入的过程中没有更高优先级的异常出现，处理器就开始执行异常处理程序，并自动将相应的挂起中断的状态变为有效。

如果在异常进入的过程中有另一个优先级更高的异常出现，处理器就开始执行这个高优先级异常的异常处理程序，不改变前一个异常的挂起状态。这是一个迟来异常的情况。

25.3.3.6.2 异常返回

当处理器处于处理程序模式并且执行下面其中一条指令尝试将 PC 设为 EXC_RETURN 值时，出现异常返回：

- POP 指令，用来加载 PC
- BX 指令，用来使用任意的寄存器。

在异常进入时处理器将一个 EXC_RETURN 值保存到 LR 中。异常机制依靠这个值来检测处理器何时执行完一个异常处理程序。EXC_RETURN 值的位 [31:4] 为 0xFFFFFFFF。当处理器将一个相应的这种形式的值加载到 PC 时，它将检测到这个操作并不是一个正常的分支操作，而是异常已经结束。因此，处理器启动异常返回序列。EXC_RETURN 的位 [3:0] 指出了所需的返回堆栈和处理器模式，如表 25-364 所示。

表 364. 异常返回的行为

EXC_RETURN	描述
0xFFFFFFFF1	返回到处理机模式。 异常返回获得主堆栈的状态。 返回后执行使用 MSP。
0xFFFFFFFF9	返回到线程模式。 异常返回获得 MSP 的状态。 返回后执行使用 MSP。
0xFFFFFFFFD	返回到线程模式。 异常返回获得 PSP 的状态。 返回后执行使用 PSP。
所有其他值	保留。

25.3.4 故障处理

故障是异常的一个子集，见[章节 25-25.3.3](#)。所有的故障都导致 HardFault 异常被处理，或者，如果故障在 NMI 或 HardFault 处理程序中出现，会导致锁定。发生以下情况会导致出现故障：

- 在一个优先级等于或高于 SVCall 的地方执行 svc 指令
- 在没有调试器的情况下执行 BKPT 指令
- 在加载或存储时出现一个系统产生的总线错误
- 执行一个 XN 存储器地址中的指令
- 从系统产生了一个总线故障的地址单元中执行指令
- 在提取向量时出现了一个系统产生的总线错误
- 执行一个未定义的指令
- 由于 T 位之前被清零而导致不再处于 Thumb 状态的情况下执行一条指令
- 尝试对一个不对齐的地址执行加载或存储操作

只有复位和 NMI 可以抢占优先级固定的 HardFault 处理程序。HardFault 可以抢占除复位、NMI 或其他硬故障之外的任何异常。

25.3.4.1 锁定

如果在执行 NMI 或 HardFault 处理程序时出现故障，或者，在一个使用 MSP 的异常返回时出栈的却是 PSR 的时候系统产生一个总线错误，处理器进入锁定状态。当处理器处于锁定状态时，它不执行任何指令。处理器保持处于锁定状态，直到下面任何一种情况出现：

- 出现复位
- 调试器将锁定状态终止
- 出现一个 NMI，以及当前的锁定处于 HardFault 处理程序中。

如果锁定状态出现在 NMI 处理程序中，后面的 NMI 就无法使处理器离开锁定状态。

25.3.5 电源管理

Cortex-M0 处理器的睡眠模式可以降低功耗，睡眠模式包含 2 种：

- 睡眠模式：停止处理器时钟
- 深度睡眠模式：详情请参见 < 待定 >。

SCR 的 SLEEPDEEP 位选择使用哪种睡眠模式，见[章节 25–25.5.3.5](#)。

本节描述了进入睡眠模式的机制和将器件从睡眠模式唤醒的条件。

25.3.5.1 进入睡眠模式

本节描述了软件可以用来使处理器进入睡眠模式的一种机制。

系统可以产生伪唤醒事件，例如，一个调试操作唤醒处理器。因此，软件必须能够在这样的事件之后使处理器重新回到睡眠模式。程序中可以有空闲循环使得处理器回到睡眠模式。

25.3.5.1.1 等待中断

等待中断指令 (WFI) 使器件立刻进入睡眠模式。当执行一个 WFI 指令时，处理器停止执行指令，进入睡眠模式。更多信息请见[章节 25–25.4.7.12](#)。

25.3.5.1.2 等待事件

注：WFE 指令不能在 LPC11U1x 上使用。

等待事件指令 (WFE) 根据一个一位的事件寄存器的值来进入睡眠模式。处理器执行一个 WFE 指令时检查事件寄存器的值：

0 — 处理器停止执行指令，进入睡眠模式

1 — 处理器将寄存器的值设为 0，并继续执行指令，不进入睡眠模式。

更多信息请见[章节 25–25.4.7.11](#)。

如果事件寄存器为 1，表示处理器在执行 WFE 指令时不必进入睡眠模式。通常的原因是出现了一个外部事件，或者系统中的另一个处理器已经执行了 SEV 指令，见[章节 25–25.4.7.9](#)。软件不能直接访问这个寄存器。

25.3.5.1.3 Sleep-on-exit

如果 SCR 的 SLEEPONEXIT 位被设为 1，当处理器完成一个异常处理程序的执行并返回到线程模式时，处理器立刻进入睡眠模式。如果应用只要求处理器在中断出现时运行，就可以使用这种机制。

25.3.5.2 从睡眠模式唤醒

处理器的唤醒条件取决于使处理器进入睡眠模式所采用的机制。

25.3.5.2.1 从 WFI 或 sleep-on-exit 唤醒

通常，只有当检测到一个优先级足够高的异常导致进入异常处理时，处理器才唤醒。

某些嵌入式系统在处理器唤醒之后可能必须先执行系统恢复任务，然后再执行中断处理程序。通过将 PRIMASK 位设为 1 来实现这个操作。如果到来的中断被使能，并且优先级高于当前的异常优先级，处理器就唤醒，但不执行中断处理程序，直至处理器将 PRIMASK 设为 0。有关 PRIMASK 的更多信息，请见[章节 25–25.3.1.3.6](#)。

25.3.5.2.2 从 WFE 唤醒

如果出现以下情况，处理器就唤醒：

- 处理器检测到一个优先级足够高的异常导致进入异常处理
- 在一个多处理器的系统中，系统中的另一个处理器执行了 SEV 指令。

另外，如果 SCR 的 SEVONPEND 位被设为 1，那么任何新的挂起中断都能触发一个事件和唤醒处理器，即使这个中断被禁能，或者这个中断的优先级不够高而导致无法进入异常处理。有关 SCR 的更多信息，请见[章节 25–25.5.3.5](#)。

25.3.5.3 电源管理编程提示

ISO/IEC C 不能直接产生 WFI、WFE 和 SEV 指令。CMSIS 为这些指令提供了以下内在函数：

```
void __WFE(void) // 等待事件

void __WFE(void) // 等待中断

void __SEV(void) // 发送事件
```

25.4 指令集

25.4.1 指令集汇总

处理器执行特定版本的指令集。[表 365](#) 列出了所支持的指令。

注：于[表 365](#)

- 尖括号 <> 括着操作数的备用格式
- 大括号 {} 括着可选的操作数和助记符部分
- 操作数列所列出的操作数不完全。

有关指令和操作数的信息，详见指令描述。

表 365. Cortex-M0 指令

助记符	操作数	简述	标志	参考资料
ADCS	{Rd,} Rn, Rm	进位加法	N,Z,C,V	章节 25–25.4.5.1
ADD{S}	{Rd,} Rn, <Rm #imm>	加法	N,Z,C,V	章节 25–25.4.5.1
ADR	Rd, label	将基于 PC 相对偏移的地址读到寄存器	-	章节 25–25.4.4.1
ANDS	{Rd,} Rn, Rm	位与操作	N,Z	章节 25–25.4.5.1
ASRS	{Rd,} Rm, <Rs #imm>	算术右移	N,Z,C	章节 25–25.4.5.3
B{cc}	label	跳转 { 有条件 }	-	章节 25–25.4.6.1
BICS	{Rd,} Rn, Rm	位清除	N,Z	章节 25–25.4.5.2
BKPT	#imm	断点	-	章节 25–25.4.7.1
BL	label	带链接的跳转	-	章节 25–25.4.6.1
BLX	Rm	带链接的间接跳转	-	章节 25–25.4.6.1
BX	Rm	间接跳转	-	章节 25–25.4.6.1

表 365. Cortex-M0 指令

助记符	操作数	简述	标志	参考资料
CMN	<i>Rn, Rm</i>	比较负值	N,Z,C,V	章节 25–25.4.5.4
CMP	<i>Rn, <Rm #imm></i>	比较	N,Z,C,V	章节 25–25.4.5.4
CPSID	<i>i</i>	更改处理器状态, 禁能中断	-	章节 25–25.4.7.2
CPSIE	<i>i</i>	更改处理器状态, 使能中断	-	章节 25–25.4.7.2
DMB	-	数据内存屏障	-	章节 25–25.4.7.3
DSB	-	数据同步屏障	-	章节 25–25.4.7.4
EORS	<i>{Rd,} Rn, Rm</i>	异或	N,Z	章节 25–25.4.5.2
ISB	-	指令同步屏障	-	章节 25–25.4.7.5
LDM	<i>Rn{!}, reglist</i>	加载多个寄存器, 访问之后会递增地址	-	章节 25–25.4.4.5
LDR	<i>Rt, label</i>	从基于 PC 相对偏移的地址加载寄存器	-	章节 25–25.4.4
LDR	<i>Rt, [Rn, <Rm #imm>]</i>	用字加载寄存器	-	章节 25–25.4.4
LDRB	<i>Rt, [Rn, <Rm #imm>]</i>	用字节加载寄存器	-	章节 25–25.4.4
LDRH	<i>Rt, [Rn, <Rm #imm>]</i>	用半字加载寄存器	-	章节 25–25.4.4
LDRSB	<i>Rt, [Rn, <Rm #imm>]</i>	用有符号的字节加载寄存器	-	章节 25–25.4.4
LDRSH	<i>Rt, [Rn, <Rm #imm>]</i>	用有符号的半字加载寄存器	-	章节 25–25.4.4
LSLS	<i>{Rd,} Rn, <Rs #imm></i>	逻辑左移	N,Z,C	章节 25–25.4.5.3
U	<i>{Rd,} Rn, <Rs #imm></i>	逻辑右移	N,Z,C	章节 25–25.4.5.3
MOV{S}	<i>Rd, Rm</i>	传输	N,Z	章节 25–25.4.5.5
MRS	<i>Rd, spec_reg</i>	从特别寄存器传输到通用寄存器	-	章节 25–25.4.7.6
MSR	<i>spec_reg, Rm</i>	从通用寄存器传输到特别寄存器	N,Z,C,V	章节 25–25.4.7.7
MULS	<i>Rd, Rn, Rm</i>	乘法, 32 位结果值	N,Z	章节 25–25.4.5.6
MVNS	<i>Rd, Rm</i>	位非	N,Z	章节 25–25.4.5.5
NOP	-	无操作	-	章节 25–25.4.7.8
ORRS	<i>{Rd,} Rn, Rm</i>	逻辑或	N,Z	章节 25–25.4.5.2
POP	<i>reglist</i>	出栈, 将堆栈的内容放入寄存器	-	章节 25–25.4.4.6
PUSH	<i>reglist</i>	压栈, 将寄存器的内容压入堆栈	-	章节 25–25.4.4.6
REV	<i>Rd, Rm</i>	反转字里面的字节顺序	-	章节 25–25.4.5.7
REV16	<i>Rd, Rm</i>	反转打包半字里面的字节顺序	-	章节 25–25.4.5.7
REVSH	<i>Rd, Rm</i>	反转有符号半字里面的字节顺序	-	章节 25–25.4.5.7
RORS	<i>{Rd,} Rn, Rs</i>	循环右移	N,Z,C	章节 25–25.4.5.3
RSBS	<i>{Rd,} Rn, #0</i>	反向减法	N,Z,C,V	章节 25–25.4.5.1
SBCS	<i>{Rd,} Rn, Rm</i>	进位减法	N,Z,C,V	章节 25–25.4.5.1
SEV	-	发送事件	-	章节 25–25.4.7.9
STM	<i>Rn!, reglist</i>	存储多个寄存器, 在访问后地址递增	-	章节 25–25.4.4.5
STR	<i>Rt, [Rn, <Rm #imm>]</i>	将寄存器作为字来存储	-	章节 25–25.4.4
STRB	<i>Rt, [Rn, <Rm #imm>]</i>	将寄存器作为字节来存储	-	章节 25–25.4.4
STRH	<i>Rt, [Rn, <Rm #imm>]</i>	将寄存器作为半字来存储	-	章节 25–25.4.4
SUB{S}	<i>{Rd,} Rn, <Rm #imm></i>	减法	N,Z,C,V	章节 25–25.4.5.1

表 365. Cortex-M0 指令

助记符	操作数	简述	标志	参考资料
SVC	<i>#imm</i>	超级用户调用	-	章节 25–25.4.7.10
SXTB	<i>Rd, Rm</i>	符号扩展字节	-	章节 25–25.4.5.8
SXTH	<i>Rd, Rm</i>	符号扩展半字	-	章节 25–25.4.5.8
TST	<i>Rn, Rm</i>	基于逻辑与的测试	N,Z	章节 25–25.4.5.9
UXTB	<i>Rd, Rm</i>	0 扩展字节	-	章节 25–25.4.5.8
UXTH	<i>Rd, Rm</i>	0 扩展半字	-	章节 25–25.4.5.8
WFE	-	等待事件	-	章节 25–25.4.7.11
WFI	-	等待中断	-	章节 25–25.4.7.12

25.4.2 内部函数

ISO/IEC C 代码不能直接访问某些 Cortex-M0 指令。本章节对可以产生这些指令的内部函数进行了描述，内部函数可由 CMSIS 或有可能由 C 编译器提供。若 C 编译器不支持相关的内部函数，则用户可能需要使用内联汇编程序来访问相关的指令。

CMSIS 提供下列内部函数来产生 ISO/IEC C 代码不能直接访问的指令：

表 366. 产生某些 Cortex-M0 指令的 CMSIS 内部函数

指令	CMSIS 内部函数
CPSIE i	void __enable_irq(void)
CPSID i	void __disable_irq(void)
ISB	void __ISB(void)
DSB	void __DSB(void)
DMB	void __DMB(void)
NOP	void __NOP(void)
REV	uint32_t __REV(uint32_t int value)
REV16	uint32_t __REV16(uint32_t int value)
REVSH	uint32_t __REVSH(uint32_t int value)
SEV	void __SEV(void)
WFE	void __WFE(void)
WFI	void __WFI(void)

CMSIS 还提供使用 MRS 和 MSR 指令来访问特别寄存器的函数：

表 367. 访问特别寄存器的 CMSIS 内部函数

特别寄存器	访问类型	CMSIS 函数
PRIMASK	读	uint32_t __get_PRIMASK (void)
	写	void __set_PRIMASK (uint32_t value)
CONTROL	读	uint32_t __get_CONTROL (void)
	写	void __set_CONTROL (uint32_t value)
MSP	读	uint32_t __get_MSP (void)
	写	void __set_MSP (uint32_t TopOfMainStack)
PSP	读	uint32_t __get_PSP (void)
	写	void __set_PSP (uint32_t TopOfProcStack)

25.4.3 About the instruction descriptions

下列小节对如何使用指令进行了更为详细的描述：

- [章节 25.4.3.1 “操作数”](#)
- [章节 25.4.3.2 “使用 PC 或 SP 时的限制”](#)
- [章节 25.4.3.3 “移位操作”](#)
- [章节 25.4.3.4 “地址对齐”](#)
- [章节 25.4.3.5 “相对 PC 的表达式”](#)
- [章节 25.4.3.6 “条件执行”](#)

25.4.3.1 操作数

指令操作数可以是 ARM 寄存器、常量或其他的指令特定参数。指令在操作数上操作，并经常将结果存放在目标寄存器中。当指令中存在目标寄存器时，它通常会在其他操作数之前被指定。

25.4.3.2 使用 PC 或 SP 时的限制

对于用于操作数或目标寄存器的程序计数器 (PC) 或堆栈指针，许多指令都不能使用它们，或者存在着用户能否使用它们的限制。更多信息，详见指令的描述。

注：当使用 BX、BLX 或 POP 指令来更新 PC 时，为正确执行程序，任何地址的位 0 都必须为 1。这是因为该位指示目标指令集，且 Cortex-M0 处理器只支持 Thumb 指令。当 BL 或 BLX 指令将位 0 的值写入 LR 时，值 1 会被自动分配。

25.4.3.3 移位操作

寄存器移位操作通过特定的位数（**移位长度**）来实现寄存器位的左右移位操作。寄存器移位可以由指令 ASR、LSR、LSL 和 ROR 直接执行，且结果会被写入到目标寄存器中。允许的移位长度由移位类型和指令决定，请见各个指令的描述。若移位长度为 0，则不发生移位操作。寄存器移位操作会更新进位标志，当移位长度被指定为 0 时除外。下列各子节描述了各种移位操作以及它们是如何影响进位标志。在这些描述中，*Rm* 是包含着移位值的寄存器，*n* 是移位长度。

25.4.3.3.1 ASR

算术右移 *n* 位的操作是将 *Rm* 寄存器左边的 $32-n$ 个位向右移动 *n* 位，结果是寄存器右边有 $32-n$ 个位，然后再将寄存器位 [31] 的原始值复制到结果寄存器左边的 *n* 个位中。见图 25-70。

用户可以使用 ASR 对寄存器 *Rm* 的符号值进行 2^n 除法操作，得到的结果为负无穷大。

当指令为 ASRS 时，进位标志会被更新为最后移出的位值，即寄存器 *Rm* 的位 [*n*-1]。

注：

- 如果 *n* 为 32 或大于 32，那么结果中的所有位都会被置为 *Rm* 中位 [31] 的值。
- 如果 *n* 为 32 或大于 32，那么进位标志被更新为 *Rm* 位 [31] 的值。



图 70. ASR #3

25.4.3.3.2 LSR

逻辑右移 n 位的操作是将 Rm 寄存器左边的 $32-n$ 个位向右移动 n 位，结果是寄存器右边有 $32-n$ 个位，然后再将结果寄存器左边的 n 个位设为 0。见图 71。

用户可以使用 LSR 操作来对寄存器 Rm 的值进行 2^n 除法操作，如果值为无符号的整数。

当指令为 LSRS 时，进位标志会被更新为最后移出的位值，即寄存器 Rm 的位 $[n-1]$ 。

注：

- 如果 n 为 32 或大于 32，那么结果中的所有位都会被清除为 0。
- 如果 n 为 33 或大于 33，那么进位标志被更新为 0。

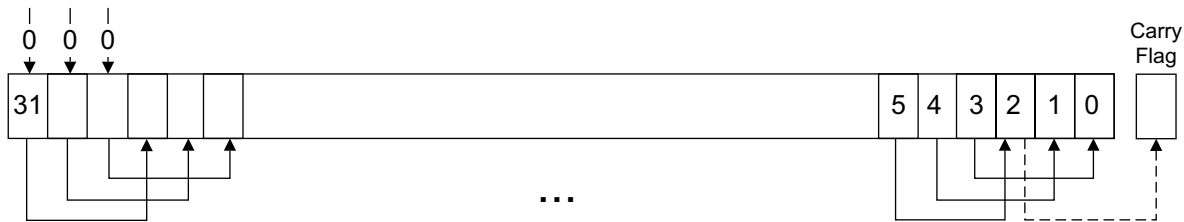


图 71. LSR #3

25.4.3.3.3 LSL

逻辑左移 n 位的操作是将 Rm 寄存器右边的 $32-n$ 个位向左移动 n 位，结果是寄存器左边有 $32-n$ 个位，然后再将结果寄存器右边的 n 个位设为 0。见图 72。

用户可以使用 LSL 操作来对寄存器 Rm 的值与 2^n 进行乘法操作，如果值为无符号的整数或是有符号 2 的补码整数值。溢出会在无警告的提示下发生。

当指令为 LSLS 时，进位标志会被更新为最后移出的位值，即寄存器 Rm 的位 $[32-n]$ 。当使用 LSL #0 时，这些指令不会影响进位标志。

注：

- 如果 n 为 32 或大于 32，那么结果中的所有位都会被清除为 0。
- 如果 n 为 33 或大于 33，那么进位标志被更新为 0。

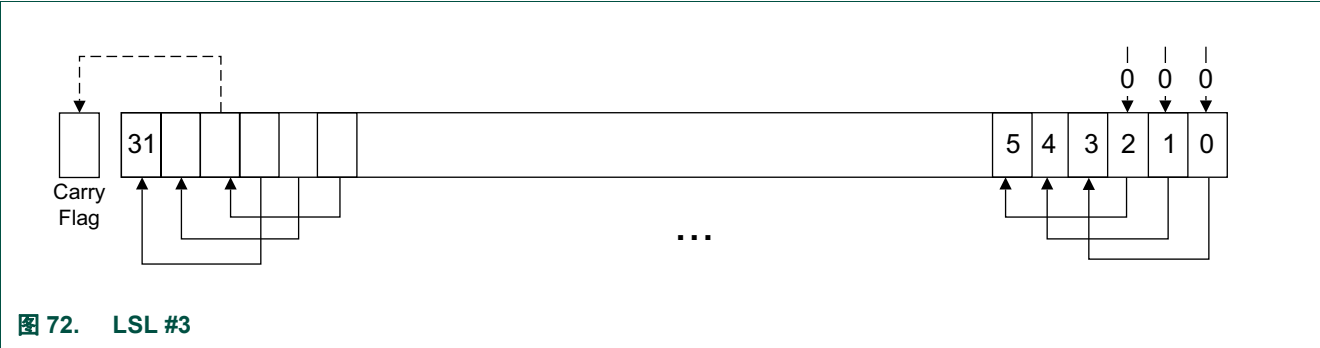


图 72. LSL #3

25.4.3.3.4 ROR

循环右移 n 位的操作是将 Rm 寄存器左边的 $32-n$ 个位向右移动 n 位，结果是寄存器右边有 $32-n$ 个位，然后再将寄存器右边的 n 个位移动到结果寄存器的左边的 n 个位中。见图 25-73。

当指令为 RORS 时，进位标志会被更新为最后循环出的位值，即寄存器 Rm 的位 $[n-1]$ 。

注：

- 如果 n 为 32，那么结果中的所有位值都相同于 Rm 中的值，且如果进位标志被更新，则会被更新为 Rm 的位 $[31]$ 的值。
- 移位长度
 n 大于 32 的

ROR

与移位长度为 $n-32$ 的 ROR 操作得到的结果相同。

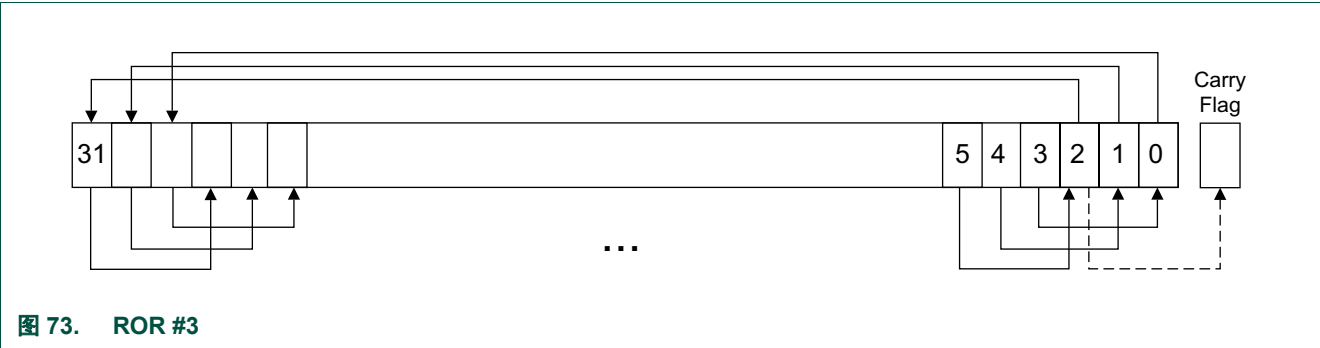


图 73. ROR #3

25.4.3.4 地址对齐

对齐访问是这样的一个操作：字对齐地址是用于字或多字访问，或者半字对齐地址是用于半字访问。字节访问通常是对齐访问的。

Cortex-M0 处理器不支持非对齐地址的访问。任何尝试执行一个非对齐的存储器访问操作都会导致 HardFault 异常。

25.4.3.5 相对 PC 的表达式

相对 PC 的表达式或**标签**是一个代表着指令或文字数据的地址的符号。在指令中它被表示为 PC 值加上或减去一个数字偏移量。汇编程序从标签和当前指令的地址中计算出所要求的偏移量。如果偏移量太大，则汇编程序会产生一个错误。

注：

- 对于大多数指令，PC 的值就是当前指令的地址加上 4 个字节。
- 汇编程序可能允许用其他语法来表示 PC 相对表达式，如标签加上或减去一个数值，或者用 [PC, #imm] 格式表示。

25.4.3.6 条件执行

大多数数据处理指令依据操作的结果在应用程序状态寄存器 (APSR) 中更新条件标志，见[章节 25.3.1.3.5](#)。某些指令更新所有标志，而某些指令则仅更新子集。如果标志不被更新，则原始值被保留。有关指令对标志的影响，请见指令的描述。

用户可以根据另一个指令中设置的条件标志按以下方式执行条件性的跳转指令：

- 在指令更新标志后立即执行
- 在经过任意数量的不会更新标志的间隔数指令后执行

在 Cortex-M0 处理器上，通过使用条件性的跳转指令，就可以实现条件性的执行操作。

本小节描述了以下内容：

- [章节 25-25.4.3.6.1“条件标志”](#)
- [章节 25-25.4.3.6.2“条件代码后缀”](#)

25.4.3.6.1 条件标志

APSR 包含下列条件标志：

- N** — 当操作的结果为负值时置为 1，否则清除为 0。
- Z** — 当操作的结果为 0 时置为 1，否则清除为 0。
- C** — 当操作的结果导致要进位时置为 1，否则清除为 0。
- V** — 当操作引发溢出时置为 1，否则清除为 0。

有关 APSR 的更多信息，请见[章节 25-25.3.1.3.5](#)。

当出现下列情况时，会发生进位操作：

- 如果加法的结果大于或等于 2^{32}
- 如果减法的结果为正或等于 0
- 由于移位指令或循环指令而发生的进位操作。

当位 [31] 中结果的符号值不与在无穷精度中所执行操作的结果符号值匹配时，溢出发生，例如：

- 如果二个负值相加得出一个正值
- 如果二个正值相加得出一个负值
- 如果从一个负值减去一个正值得到一个正值
- 如果从一个正值减去一个负值得到一个负值。

对于 CMP，比较操作与减法操作相同，对于 CMN，则加法操作相同，结果值会被丢弃除外。更多信息，详见指令的描述。

25.4.3.6.2 条件代码后缀

条件性跳转在语法描述显示为 B{cond}。只有 APSR 的条件代码标志符合指定的条件时，才能执行带有条件代码的跳转指令，否则要忽略跳转指令。显示了使用的条件代码。

表 368 同时还显示了条件代码后缀和 N、Z、C 和 V 标志之间的联系。

表 368. 条件代码后缀

后缀	标志	意义
EQ	Z = 1	等效，最后标志设置结果为 0
NE	Z = 0	不等效，最后标志设置结果为非 0
CS 或 HS	C = 1	更高或相同，无符号
CC 或 LO	C = 0	更低，无符号
MI	N = 1	负数
PL	N = 0	正数或 0
VS	V = 1	溢出
VC	V = 0	无溢出
HI	C = 1 和 Z = 0	更高，无符号
LS	C = 0 或 Z = 1	更低或相同，无符号
GE	N = V	大于或等效，有符号
LT	N != V	少于，有符号
GT	Z = 0 和 N = V	大于，有符号
LE	Z = 1 和 N != V	少于或等于，有符号
AL	可以为任意值	总是。当没有指定后缀时，这是默认的操作。

25.4.4 存储器访问指令

表 369 所示为存储器访问指令：

表 369. 访问指令

助记符	简述	见
LDR{type}	使用寄存器偏移量来加载寄存器	章节 25–25.4.4.3
LDR	从基于 PC 相对偏移的地址加载寄存器	章节 25–25.4.4.4
POP	出栈，将堆栈的内容放入寄存器	章节 25–25.4.4.6
PUSH	压栈，将寄存器的内容压入堆栈	章节 25–25.4.4.6
STM	存储多个寄存器	章节 25–25.4.4.5
STR{type}	使用立即数偏移量来存储寄存器	章节 25–25.4.4.2
STR{type}	使用寄存器偏移量来存储寄存器	章节 25–25.4.4.3

25.4.4.1 ADR

产生一个 PC 相对地址。

25.4.4.1.1 语法

ADR Rd, label

其中：

Rd 是目标寄存器。

label 是 PC-相对表达式。见[章节 25–25.4.3.5](#)。

25.4.4.1.2 操作

ADR 通过将立即数值加入到 PC 中来产生一个地址，并将得到的地址结果写入到目标寄存器中。

ADR 产生区域独立代码非常便利，因为地址是 PC 相对地址。

如果用户使用 ADR 来产生 BX 或 BLX 指令的目标地址，为了能正确执行程序，必须要保证将产生的地址的位 [0] 设置为 1。

25.4.4.1.3 限制

在该指令中，*Rd* 必须指定 R0-R7。地址数据值必须是字对齐，且不能超出当前 PC 的 1020 字节。

25.4.4.1.4 条件标志

该指令不会改变标志。

25.4.4.1.5 示例

```
ADR    R1, TextMessage    ; 将标签为
                                ; TextMessage 单元上的地址值写入到 R1 中

ADR    R3, [PC,#996]      ; 将 R3 的值设为 PC + 996。
```

25.4.4.2 LDR 和 STR，立即数偏移量

具有立即数偏移量的加载和存储。

25.4.4.2.1 语法

LDR *Rt*, [<*Rn* | SP> {, #*imm*}]

LDR<B|H> *Rt*, [*Rn* {, #*imm*}]

STR *Rt*, [<*Rn* | SP>, {, #*imm*}]

STR<B|H> *Rt*, [*Rn* {, #*imm*}]

其中：

Rt 是加载或存储的寄存器。

Rn 是基于存储器地址上的寄存器。

imm 是 *Rn* 的偏移量。如果 *imm* 被省略，则假设它为 0。

25.4.4.2.2 操作

LDR、LDRB 和 LDRH 指令将存储器中的字、字节或半字数据值加载到 *Rt* 指定的寄存器中。在将数据写入 *Rt* 指定的寄存器之前，长度少于字的数据要用 0 扩充到 32 位的长度。

STR、STRB 和 STRH 指令将 *Rt* 寄存器指定的单个寄存器中所包含的字、最低位字节或低半字存放到存储器中。从加载的存储器地址或用于存放的存储器地址是 *Rn* 或 SP 所指定的寄存器的值与立即数 *imm* 的和。

25.4.4.2.3 限制

在这些指令中：

- *Rt* 和 *Rn* 必须只能指定 R0-R7。
- *imm* 的值必须要符合下列要求：
 - 0 到 1020 之间，对于 LDR 和 STR 操作，
 - 在将 SP 用作基址寄存器时，其值必须是 4 的整数倍
 - 0 到 124 之间，对于 LDR 和 STR 操作，
 - 在将 R0-R7 用作基址寄存器时，其值必须是 4 的整数倍
 - 0 到 62 之间，对于 LDRH 和 STRH 操作，其值必须是 2 的整数倍
 - 0 到 31 之间，对于 LDRB 和 STRB。
- 计算出的地址必须能够被事务中的字节数整除，见[章节 25-25.4.3.4](#)。

25.4.4.2.4 条件标志

这些指令不会改变标志。

25.4.4.2.5 示例

```
LDR    R4, [R7           ] ; 将 R7 的地址载入到 R4。  
STR    R2, [R0,#const-struct] ; const-struct 是评估  
                                     ; 处于 0-102021 范围内的常量的表达式。
```

25.4.4.3 LDR 和 STR，寄存器偏移量

带寄存器偏移量的加载和存储。

25.4.4.3.1 语法

```
LDR Rt, [Rn, Rm]  
  
LDR<B|H> Rt, [Rn, Rm]  
  
LDR<SB|SH> Rt, [Rn, Rm]  
  
STR Rt, [Rn, Rm]  
  
STR<B|H> Rt, [Rn, Rm]
```

其中：

- Rt* 是加载或存储的寄存器。
- Rn* 是存储器地址所基于的寄存器。
- Rm* 是含有用作偏移量的值的寄存器。

25.4.4.3.2 操作

LDR、LDRB、U、LDRSB 和 LDRSH 将存储器中的字、0 扩展字节、0 扩展半字、符号扩展字节或符号扩展半字值加载到 *Rt* 指定的寄存器中。

STR、STRB 和 STRH 指令将 *Rt* 寄存器指定的单个寄存器中所包含的字，最低位字节或低半字存放到存储器中。

从中加载的存储器地址或用于存放的存储器地址是 *Rn* 和 *Rm* 所指定的寄存器中的值之和。

25.4.4.3.3 限制

在这些指令中：

- *Rt*、*Rn* 和 *Rm* 必须只能指定 R0-R7。
- 计算出的地址必须能够被加载或存储的字节数整除，见[章节 25–25.4.3.4](#)。

25.4.4.3.4 条件标志

这些指令不会改变标志。

25.4.4.3.5 示例

```
STR    R0, [R5, R1]    ; 将 R0 的值存储到
                        ; R5 加 R1 得出的地址中

LDRSH  R1, [R2, R3]    ; 从 (R2 + R3) 所指定的存储器地址中加载半字数据，
                        ; 符号扩展到 32 位并

                        ; 将其写入到 R1 中。
```

25.4.4.4 LDR，PC-相对

从存储器中加载寄存器（文字数据）。

25.4.4.4.1 语法

```
LDR Rt, label
```

其中：

- Rt* 是加载的寄存器。
- label* 是 PC-相对表达式。见[章节 25–25.4.3.5](#)。

25.4.4.4.2 操作

将 *label* 所指定的存储器中的字加载到 *Rt* 所指定的寄存器中。

25.4.4.4.3 限制

在这些指令中，*label* 的大小必须位于当前 PC 的 1020 字节范围之内，且是字对齐的。

25.4.4.4.4 条件标志

这些指令不会改变标志。

25.4.4.4.5 示例

```
LDR    R0, LookUpTable ; 将标签为 LookUpTable 的地址中的字数据
                        ; 加载到 R0 中。

LDR    R3, [PC, #100]   ; 将 (PC + 100) 上的存储器字加载到 R3 中。
```

25.4.4.5 LDM 和 STM

加载和存储多个寄存器。

25.4.4.5.1 语法

LDM *Rn*{!}, *reglist*

STM *Rn*!, *reglist*

其中：

Rn 是存储器地址所基于的寄存器。

! 回写后缀。

reglist 是被加载或存储的一个或多个寄存器的列表，用大括号括住。它可包含寄存器范围。若它包含多于一个的寄存器或寄存器范围，必须要将其用逗号隔开，见[章节 25–25.4.4.5.5](#)。

对于 LDM，LDMIA 和 LDMFD 相近。LDMIA 为每次访问后都会递增的基址寄存器。LDMFD 用法是将数据从满的递减堆栈中移出。

对于 STM，STMIA 和 STMEA 相近。STMIA 为每次访问后都会递增的基址寄存器。STMEA 用法是将数据压入空的递增堆栈中。

25.4.4.5.2 操作

LDM 指令将基于 *Rn* 上的存储器地址的字值加载到 *reglist* 的寄存器中。

STM 指令将 *reglist* 中的寄存器的字值存放到基于 *Rn* 的存储器地址中。

用于访问的存储器地址为 4 字节间隔，其范围为 *Rn* 所指定的寄存器的值至 $Rn + 4 * (n-1)$ 所指定的寄存器的值，这里的 *n* 是 *reglist* 中的寄存器数量。访问的顺序是按照寄存器的编号从低到高发生，最低编号的寄存器使用最低的存储器地址，最高编号的寄存器使用最高的存储器地址。如果写回后缀被指定，则 $Rn + 4 * n$ 所指定的寄存器的值会被写回到 *Rn* 所指定的寄存器中。

25.4.4.5.3 限制

在这些指令中：

- *reglist* 和 *Rn* 限制为 R0-R7。
- 必须始终使用写回后缀，除非指令是 LDM 指令，在 LDM 里，*reglist* 也含有 *Rn*，在这种情况下，不得使用写回后缀。
- *Rn* 所指定的寄存器的值必须是字对齐的。更多信息请见[章节 25–25.4.3.4](#)。
- 对于 STM，如果 *reglist* 中存在着 *Rn*，那么 *Rn* 必须是列表中的第一个寄存器。

25.4.4.5.4 条件标志

这些指令不会改变标志。

25.4.4.5.5 示例

```
LDM      R0, {R0, R3, R4}      ; LDMIA 相近于 LDM
STMIA    R1!, {R2-R4, R6}
```

25.4.4.5.6 错误示例

```
STM      R5!, {R4, R5, R6} ; 存放"/R5 的值是不可预测的
LDM      R2, { }           ; 在列表中至少要存在着一个寄存器
```

25.4.4.6 PUSH 和 POP

将寄存器压入满递减堆栈和将满递减堆栈中的内容移入寄存器。

25.4.4.6.1 语法

PUSH *reglist*

POP *reglist*

其中：

reglist 是非空的寄存器列表，用大括号括着。它可包含寄存器范围。若它包含多于一个的寄存器或寄存器范围，必须要将其用逗号隔开。

25.4.4.6.2 操作

PUSH 将寄存器存放到堆栈中，最低编号的寄存器使用低存储器地址，最高编号的寄存器使用高存储器地址。

POP 将堆栈中的内容加载到寄存器中，最低编号的寄存器使用最低存储器地址，最高编号的寄存器使用最高存储器地址。

PUSH 将 SP 寄存器的值减去 4 所得的值用作最高存储器地址，

POP 将 SP 寄存器的值用作最低的存储器地址来执行满递减堆栈操作。当操作完成时，

PUSH 会更新 SP 寄存器来指向最低存储值的单元，

而 POP 则会更新 SP 寄存器来指向高于所加载的最高单元的单元。

如果 POP 在它的 *reglist* 中包含了 PC，则当 POP 指令完成时，会在该单元上执行一个跳转操作。为 PC 所读出的 Bit[0] 值用来更新 APSR T- 位。该位必须为 1，以确保能正确执行程序。

25.4.4.6.3 限制

在这些指令中：

- *reglist* 必须只为 R0-R7。
- 对于 PUSH 和 POP，异常情况分别是 LR 和 PC。

25.4.4.6.4 条件标志

这些指令不会改变标志。

25.4.4.6.5 示例

```
PUSH    {R0,R4-R7}      ; 将 R0、R4、R5、R6、R7 压入堆栈
PUSH    {R2,LR}          ; 将 R2 和链接寄存器压入堆栈
POP     {R0,R6,PC}       ; 令 r0、r6 和 PC 出栈，然后跳转到
                        ; 新的 PC 值。
```

25.4.5 通用数据处理指令

[表 370](#) 显示了数据处理指令：

表 370. 数据处理指令

助记符	简述	见
ADCS	进位加法	章节 25–25.4.5.1
ADD{S}	加法	章节 25–25.4.5.1
ANDS	逻辑与	章节 25–25.4.5.2
ASRS	算术右移	章节 25–25.4.5.3
BICS	位清除	章节 25–25.4.5.2
CMN	比较负值	章节 25–25.4.5.4
CMP	比较	章节 25–25.4.5.4
EORS	异或	章节 25–25.4.5.2
LSLS	逻辑左移	章节 25–25.4.5.3
LSRS	逻辑右移	章节 25–25.4.5.3
MOV{S}	传输	章节 25–25.4.5.5
MULS	乘法	章节 25–25.4.5.6
MVNS	取反传输	章节 25–25.4.5.5
ORRS	逻辑或	章节 25–25.4.5.2
REV	反转字里面的字节顺序	章节 25–25.4.5.7
REV16	反转每半字里面的字节顺序	章节 25–25.4.5.7
REVSH	反转低半字中的字节顺序，并进行符号扩展	章节 25–25.4.5.7
RORS	循环右移	章节 25–25.4.5.3
RSBS	反向减法	章节 25–25.4.5.1
SBCS	进位减法	章节 25–25.4.5.1
SUBS	减法	章节 25–25.4.5.1
SXTB	符号扩展字节	章节 25–25.4.5.8
SXTH	符号扩展半字	章节 25–25.4.5.8
UXTB	0 扩展字节	章节 25–25.4.5.8
UXTH	0 扩展半字	章节 25–25.4.5.8
TST	测试	章节 25–25.4.5.9

25.4.5.1 ADC、ADD、RSB 和 SUB

进位加法、加法、反向减法、进位减法、减法。

25.4.5.1.1 语法

ADCS {Rd,} Rn, Rm

ADD{S} {Rd,} Rn, <Rm|#imm>

RSBS {Rd,} Rn, Rm, #0

SBCS {Rd,} Rn, Rm

SUB{S} {Rd,} Rn,

<Rm|#imm>

其中：

S 会令 ADD 或 SUB 指令更新标志

Rd 指定结果寄存器

Rn 指定首个源寄存器

Rn 指定第二个源寄存器

imm 指定一个常量立即数值。

当省略了可选的 *Rd* 寄存器限定符时，会假定其值与 *Rn* 相同，例如，ADDS R1、R2 与 ADDS R1、R1、R2 相同。

25.4.5.1.2 操作

ADCS 指令将 *Rn* 中的值加到 *Rm* 的值中，如果进位标志被置位，则加多一个 1，并将结果存放在 *Rd* 所指定寄存器里，同时更新 N、Z、C 和 V 标志。

ADD 指令将 *Rn* 的值加到 *Rm* 的值或 *imm* 指定的立即数中，并将结果存放到 *Rd* 所指定的寄存器中。

ADDS 指令执行的操作与 ADD 相同，并还可以更新 N、Z、C 和 V 标志。

RSBS 指令是用 0 减去 *Rn* 中的值，得到一个负数，然后将结果值存放在 *Rd* 所指定的寄存器中，并更新 N、Z、C 和 V 标志。

SBCS 指令是用 *Rm* 的值减去 *Rm* 的值，如果进位标志置位，则减去一个 1。指令会将结果值存放到 *Rd* 所指定的寄存器中，并更新 N、Z、C 和 V 标志。

SUB 指令会减去 *Rm* 的值或 *imm* 所指定的立即数。指令把结果值存放到 *Rd* 所指定的寄存器中。

SUBS 指令执行的操作与 SUB 相同，同时它还可以更新 N、Z、C 和 V 标志。

有关如何使用 ADC 和 SBC 来综合处理多字算术，请见[章节 25–25.4.5.1.4](#)。

另请参见[章节 25–25.4.4.1](#)。

25.4.5.1.3 限制

[表 371](#) 列出了寄存器指示符的合法组合和每一个指令可以使用的立即数。

表 371. ADC、ADD、RSB、SBC 和 SUB 操作数限制

指令	Rd	Rn	Rm	imm	限制
ADCS	R0-R7	R0-R7	R0-R7	-	<i>Rd</i> 和 <i>Rn</i> 必须指定相同的寄存器。
ADD	R0-R15	R0-R15	R0-PC	-	<i>Rd</i> 和 <i>Rn</i> 必须指定相同的寄存器。 <i>Rn</i> 和 <i>Rm</i> 必须不能同时指定 PC。
	R0-R7	SP 或 PC	-	0-1020	立即数必须为 4 的整数倍。
	SP	SP	-	0-508	立即数必须为 4 的整数倍。
ADDS	R0-R7	R0-R7	-	0-7	-
	R0-R7	R0-R7	-	0-255	<i>Rd</i> 和 <i>Rn</i> 必须指定相同的寄存器。
	R0-R7	R0-R7	R0-R7	-	-
RSBS	R0-R7	R0-R7	-	-	-
SBCS	R0-R7	R0-R7	R0-R7	-	<i>Rd</i> 和 <i>Rn</i> 必须指定相同的寄存器。
SUB	SP	SP	-	0-508	立即数必须为 4 的整数倍。

表 371. ADC、ADD、RSB、SBC 和 SUB 操作数限制

指令	Rd	Rn	Rm	imm	限制
SUBS	R0-R7	R0-R7	-	0-7	-
	R0-R7	R0-R7	-	0-255	Rd 和 Rn 必须指定相同的寄存器。
	R0-R7	R0-R7	R0-R7	-	-

25.4.5.1.4 示例

下例所示为二个指令将 R0 和 R1 所包含的 64 位整数值加到 R2 和 R3 所包含的另一个 64 位整数值中，并将结果存放到 R0 和 R1 中。

64 位加法：

```
ADDS    R0, R0, R2    ; 加上最低位的字
ADCS     R1, R1, R3    ; 加上最高位的字，带进位
```

多字的值无需要使用连续的寄存器。下面示例为指令会令 R4、R5 和 R6 所包含的 96 位整数值减去 R1、R2 和 R3 所包含的 96 位整数值。该例将结果值存放在 R4、R5 和 R6 中。

96 位减法：

```
SUBS    R4, R4, R1    ; 减去最低位字
SBCS    R5, R5, R2    ; 减去中间的字，带进位
SBCS    R6, R6, R3    ; 减去最高位字，带进位
```

下列所示的 RSBS 指令是用来执行单个寄存器 1 的补码的操作。

```
算术负值运算：    RSBS    R7, R7, #0    ; 用 0 减去 R7
```

25.4.5.2 AND、ORR、EOR 和 BIC

逻辑 AND、OR、异或和位清除。

25.4.5.2.1 语法

```
ANDS {Rd,} Rn, Rm
ORRS {Rd,} Rn, Rm
EORS {Rd,} Rn, Rm
BICS {Rd,} Rn, Rm
```

其中：

- Rd 是目标寄存器。
- Rn 是保存第一个操作数的寄存器，且还是与目标寄存器相同的寄存器。
- Rm 是第二个寄存器。

25.4.5.2.2 操作

AND、EOR 和 ORR 对 Rn 和 Rm 的值按位执行 AND、异或、或操作。

BIC 指令对 Rn 上的位执行 AND 操作，对 Rm 值上的相应位执行逻辑非操作。

条件代码标志会根据操作的结果被更新，见[章节 25-25.4.3.6.1](#)。

25.4.5.2.3 限制

在这些指令中， *Rd*、 *Rn* 和 *Rm* 必须只能指定 R0-R7。

25.4.5.2.4 条件标志

这些指令：

- 根据结果值来更新 N 和 Z 标志
- 不影响 C 或 V 标志。

25.4.5.2.5 示例

```
ANDS    R2, R2, R1
ORRS    R2, R2, R5
ANDS    R5, R5, R8
EORS    R7, R7, R6
BICS    R0, R0, R1
```

25.4.5.3 ASR、LSL、LSR 和 ROR

算术右移、逻辑左移、逻辑右移、循环右移。

25.4.5.3.1 语法

```
ASRS {Rd,} Rm, Rs
ASRS {Rd,} Rm, #imm
LSLS {Rd,} Rm, Rs
LSLS {Rd,} Rm, #imm
LSRS {Rd,} Rm, Rs
LSRS {Rd,} Rm, #imm
RORS {Rd,} Rm, Rs
```

其中：

Rd 是目标寄存器。如果 *Rd* 被省略，则假定它的值与 *Rm* 相同。
Rm 是保存要移位的值的寄存器。
Rs 是保存着移位长度（该长度要应用到 *Rm* 中的值）的寄存器。
imm 是移位长度。

移位长度要由指令来决定：

- ASR** — 移位长度为 1 到 32
- LSL** — 移位长度为 0 到 31
- LSR** — 移位长度为 1 到 32。

注：MOV_S *Rd*, *Rm* 是 LSLS *Rd*, *Rm*, #0 的假名。

25.4.5.3.2 操作

ASR、LSL、LSR 和 ROR 按照立即数 *imm* 所指定的或者 *Rs* 所指定寄存器的最低位字节值所指定的长度对 *Rm* 寄存器的位执行算术左移、逻辑左移、逻辑右移或循环右移。

关于不同的指令会产生什么样的结果，详情请见[章节 25–25.4.3.3](#)。

25.4.5.3.3 限制

在这些指令中，*Rd*、*Rm* 和 *Rs* 必须只能指定 R0-R7。对于非立即数指令，*Rd* 和 *Rm* 必须指定相同的寄存器。

25.4.5.3.4 条件标志

这些指令根据结果值来更新 N 和 Z 标志。

C 标志被更新为最后移出的位，移位长度为 0 时除外，见[章节 25–25.4.3.3](#)。V 标志不变。

25.4.5.3.5 示例

```
ASRS    R7, R5, #9    ; 算术右移 9 位
LSLS    R1, R2, #3    ; 逻辑左移 3 位，并更新标志
LSRS    R4, R5, #6    ; 逻辑右移 6 位
RORS    R4, R4, R6    ; 循环右移 R6 低字节中的值。
```

25.4.5.4 CMP 和 CMN

比较和比较负值。

25.4.5.4.1 语法

CMN *Rn*, *Rm*

CMP *Rn*, #*imm*

CMP *Rn*, *Rm*

其中：

Rn 是保存第一个操作数的寄存器。

Rm 是用于比较的寄存器。

imm 是用于比较的立即数值。

25.4.5.4.2 操作

这些指令将一个寄存器中的值与另一个寄存器中的值或立即数进行比较。指令会根据结果值来更新条件标志，但不会将结果写入寄存器。

CMP 指令将 *Rn* 的值减去 *Rm* 所指定的寄存器值或立即数 *imm*，并更新标志。这操作与 SUBS 指令相同，不同的是结果值会被丢弃。

CMN 指令将 *Rm* 的值加到 *Rn* 的值中，并更新标志。这操作与 ADDS 指令相同，不同的是结果值会被丢弃。

25.4.5.4.3 限制

对于：

- CMN

指令 *Rn* 和 *Rm* 必须只能指定 R0-R7。

- CMP 指令：
 - *Rn* 和 *Rm* 可以指定 R0-R14
 - 立即数的范围为 0-255。

25.4.5.4.4 条件标志

这些指令根据结果值来更新 N、Z、C 和 V 标志。

25.4.5.4.5 示例

```
CMP      R2, R9
CMN      R0, R2
```

25.4.5.5 MOV 和 MVN

传输和取反传输。

25.4.5.5.1 语法

MOV{S} *Rd*, *Rm*

MOVS *Rd*, #*imm*

MVNS *Rd*, *Rm*

其中：

S 是可选后缀。如果指定了 *S*，则会根据操作的结果值来更新条件代码标志，见[章节 25-25.4.3.6](#)。

Rd 是目标寄存器。

Rm 是寄存器。

imm 可以是 0-255 范围内的任何一个值。

25.4.5.5.2 操作

MOV 指令将 *Rm* 的值复制到 *Rd* 中。

MOVS 指令执行的操作与 MOV 指令相同，但是它会更新 N 和 Z 标志。

MVNS 指令采用 *Rm* 的值，对该值执行按位的逻辑取反操作，并将结果存放到 *Rd* 中。

25.4.5.5.3 限制

在这些指令中，*Rd* 和 *Rm* 必须只能指定 R0-R7。

当在 MOV 指令里 *Rd* 是 PC 时：

- 结果值的位 [0] 被丢弃。
- 在通过将结果值的位 [0] 强制为 0 来创造的地址上执行跳转操作。T- 位保持不变。

注：尽管可以将 MOV 用作跳转指令，但是为了软件的可移植性，AMR 强烈推荐使用 BX 或 BLX 指令来执行跳转操作。

25.4.5.5.4 条件标志

如果 **S** 被指定，则这些指令：

- 根据结果值来更新 **N** 和 **Z** 标志
- 不影响 **C** 或 **V** 标志。

25.4.5.5.5 示例

```
MOVS  R0, #0x000B    ; 将 0x000B 写入 R0, 更新标志
MOVVS  R1, #0x0       ; 将 0 写入 R1, 更新标志
MOV    R10, R12       ; 将 R12 的值写入 R10, 不更新标志
MOVVS  R3, #23        ; 将 23 写入 R3
MOV    R8, SP         ; 将堆栈指针的值写入 R8
MVNS   R2, R0         ; 将 R0 取反写入 R2 并更新标志
```

25.4.5.6 MULS

使用 32 位操作数的乘法，产生 32 位的结果值。

25.4.5.6.1 语法

MULS *Rd*, *Rn*, *Rm*

其中：

Rd 是目标寄存器。

Rn、*Rm* 是保存进行乘法操作值的寄存器。

25.4.5.6.2 操作

MUL 指令将 *Rn* 和 *Rm* 所指定的寄存器的值进行乘法操作，并将结果值的最低 32 位存放在 *Rd* 中。条件代码标志会根据操作的结果被更新，见[章节 25-25.4.3.6](#)。

该指令的结果并不是由操作数是有符号还是无符号来决定。

25.4.5.6.3 限制

在该指令中：

- *Rd*、*Rn* 和 *Rm* 必须只能指定 R0-R7
- *Rd* 必须要和 *Rm* 相同。

25.4.5.6.4 条件标志

该指令：

- 根据结果值来更新 **N** 和 **Z** 标志
- 不影响 **C** 或 **V** 标志。

25.4.5.6.5 示例

```
MULS   R0, R2, R0    ; 乘法操作，标志被更新，R0 = R0 x R2
```

25.4.5.7 REV、REV16 和 REVSH

反转字节。

25.4.5.7.1 语法

REV *Rd*, *Rn*

REV16 *Rd*, *Rn*

REVSH *Rd*, *Rn*

其中：

Rd 是目标寄存器。

Rn 是源寄存器。

25.4.5.7.2 操作

使用这些指令来改变数据的端点排序：

REV — 将 32 位大端数据转换成小端的数据或将 32 位小端的数据转换成大端数据。

REV16 — 将二个打包的 16 位大端数据转换成小端的数据或将二个打包的小端的数据转换成大端数据。

REVSH — 将 16 位有符号的大端数据转换成 32 位有符号小端数据或将 16 位有符号小端数据转换 32 位有符号大端数据。

25.4.5.7.3 限制

在这些指令中，*Rd* 和 *Rn* 必须只能指定 R0-R7。

25.4.5.7.4 条件标志

这些指令不会改变标志。

25.4.5.7.5 示例

```
REV    R3, R7 ; 反转 R7 值的字节顺序，并将其写入 R3
REV16  R0, R0 ; 反转 R0 中的每一个 16 位半字的字节顺序
REVSH  R0, R5 ; 反转有符号的半字
```

25.4.5.8 SXT 和 UXT

符号扩展和 0 扩展。

25.4.5.8.1 语法

SXTB *Rd*, *Rm*

SXTH *Rd*, *Rm*

UXTB *Rd*, *Rm*

UXTH *Rd*, *Rm*

其中：

Rd 是目标寄存器。

Rm 是寄存器，其保存的值会被扩展。

25.4.5.8.2 操作

这些指令从结果值中提取位：

- SXTB 提取位 [7:0] 并将值进行符号扩展到 32 位
- UXTB 提取位 [7:0] 并将值用 0 扩展到 32 位
- SXTH 提取 [15:0] 并将值进行符号扩展到 32 位
- UXTH 提取 [15:0] 并将值用 0 扩展到 32 位。

25.4.5.8.3 限制

在这些指令中，*Rd* 和 *Rm* 必须只能指定 R0-R7。

25.4.5.8.4 条件标志

这些指令不会影响标志。

25.4.5.8.5 示例

```
SXTB  R4, R6           ; 获取 R6 的低半字，
                        ; 然后将其进行符号扩展到 32 位，
                        ; 并将结果写入 R4。
UXTB  R3, R1           ; 获取 R10 最低位字节，并用 0
                        ; 扩展，最后将结果写入 R3
```

25.4.5.9 TST

测试位。

25.4.5.9.1 语法

TST *Rn*, *Rm*

其中：

Rn 是保存第一个操作数的寄存器。

Rm 是测试的寄存器。

25.4.5.9.2 操作

该指令将一个寄存器的值与另一个寄存器中的值进行测试。它会根据结果值来更新条件标志，但是不会将结果值写入寄存器。

TST 指令对 *Rn* 中的值和 *Rm* 中的值执行位与操作。这是与 ANDS 指令相同的操作，不同的是它会丢弃结果值。

为了测试 *Rn* 中的某个位是 0 还是 1，要使用 TST 指令，且寄存器的该位要设为 1，其他所有位被清除为 0。

25.4.5.9.3 限制

在这些指令中，*Rn* 和 *Rm* 必须只能指定 R0-R7。

25.4.5.9.4 条件标志

该指令：

- 根据结果值来更新 N 和 Z 标志
- 不影响 C 或 V 标志。

25.4.5.9.5 示例

TST R0, R1 ; 对 R0 值和 R1 值执行位与操作，
; 更新条件代码标志，但结果值会被丢弃

25.4.6 跳转和控制指令

表 372 所示为跳转和控制指令：

表 372. 跳转和控制指令

助记符	简述	见
B{cc}	跳转 { 有条件 }	章节 25–25.4.6.1
BL	带链接的跳转	章节 25–25.4.6.1
BLX	带链接的间接跳转	章节 25–25.4.6.1
BX	间接跳转	章节 25–25.4.6.1

25.4.6.1 B、BL、BX 和 BLX

跳转指令。

25.4.6.1.1 语法

B{cond} label

BL label

BX Rm

BLX Rm

其中：

- cond 是可选的条件代码，见[章节 25–25.4.3.6](#)。
- label 是 PC-相对表达式。见[章节 25–25.4.3.5](#)。
- Rm 是提供跳转地址的寄存器。

25.4.6.1.2 操作

所有这些指令都会对 label 所指示的地址或在 Rm 所指定的寄存器中包含地址上执行跳转操作。另外：

- BL 和 BLX 指令将下一个指令的地址写入 LR，链接寄存器 R14。
- 如果 Rm 的位 [0] 是 0，则 BX 和 BLX 指令会导致 HardFault 异常。

BL 和 BLX 指令还会将 LR 的位 [0] 设置为 1。这就确保了该值适合由后续 POP {PC} 或 BX 指令使用其来执行成功的返回跳转操作。

表 373 所示为适用于各种跳转指令的跳转范围。

表 373. 跳转范围

指令	跳转范围
B <i>label</i>	−2 KB 至 +2 KB
Bcond <i>label</i>	−256 字节至 +254 字节
BL <i>label</i>	−16 MB 至 +16 MB
BX <i>Rm</i>	寄存器中的任意值
BLX <i>Rm</i>	寄存器中的任意值

25.4.6.1.3 限制

在这些指令中：

- 不要在 BX 或 BLX 指令里使用 SP 或 PC。
- 对于 BX 和 BLX，为实现正确的执行操作，*Rm* 的位 [0] 必须为 1。位 [0] 用于更新 EPSR T- 位，并会被从目标地址上丢弃。

注：Bcond 是在 Cortex-M0 处理器上唯一的条件指令。

25.4.6.1.4 条件标志

这些指令不会改变标志。

25.4.6.1.5 示例

```
B      loopA ; 跳转到 loopA
BL     funC  ; 对函数 funC 进行带链接的跳转（调用），返回存放在
              ; LR 的地址
BX     LR    ; 从函数调用中返回
BLX    R0    ; 带链接的跳转，并从（调用）中更改为存放在
              ; R0 所存放的地址

BEQ     labelD ; 条件跳转到 labelD，如果最后的标志设置
              ; 指令设置 Z 标志，否则不执行跳转。
```

25.4.7 其他指令

表 374 所示为余下的 Cortex-M0 指令：

表 374. 其他指令

助记符	简述	见
BKPT	断点	章节 25–25.4.7.1
CPSID	更改处理器状态，禁能中断	章节 25–25.4.7.2
CPSIE	更改处理器状态，使能中断	章节 25–25.4.7.2
DMB	数据内存屏障	章节 25–25.4.7.3
DSB	数据同步屏障	章节 25–25.4.7.4
ISB	指令同步屏障	章节 25–25.4.7.5
MRS	从特别寄存器传输到寄存器	章节 25–25.4.7.6
MSR	从寄存器传输到特别寄存器	章节 25–25.4.7.7
NOP	无操作	章节 25–25.4.7.8

表 374. 其他指令

助记符	简述	见
SEV	发送事件	章节 25–25.4.7.9
SVC	超级用户调用	章节 25–25.4.7.10
WFE	等待事件	章节 25–25.4.7.11
WFI	等待中断	章节 25–25.4.7.12

25.4.7.1 BKPT

断点。

25.4.7.1.1 语法

BKPT #imm

其中：

imm 是 0-255 范围内的整数。

25.4.7.1.2 操作

BKPT 指令会令处理器进入调试状态。当指令到达特定的地址时，调试工具可以使用该指令来调查系统状态。处理器会忽略 imm。如有需要，调试器可以使用它来存放断点的其他信息。

如果在执行 BKPT 指令时调试器没有连接上，那么处理器还有可能会产生 HardFault 或进入锁定状态更多信息请见[章节 25–25.3.4.1](#)。

25.4.7.1.3 限制

该指令没有限制。

25.4.7.1.4 条件标志

该指令不会改变标志。

25.4.7.1.5 示例

BKPT #0 ; 立即数值设为 0x0 的断点。

25.4.7.2 CPS

更改处理器状态。

25.4.7.2.1 语法

CPSID i

CPSIE i

25.4.7.2.2 操作

CPS 更改 PRIMASK 特别寄存器值。通过设置 PRIMASK，CPSID 可禁能中断。而通过清除 PRIMASK，CPSIE 则可使能中断。关于这些寄存器的详细描述，请见[章节 25–25.3.1.3.6](#)。

25.4.7.2.3 限制

该指令没有限制。

25.4.7.2.4 条件标志

该指令不会更改条件标志。

25.4.7.2.5 示例

CPSID i ; 禁能所有的中断, NMI 除外 (设置 PRIMASK)

CPSIE i ; 使能中断 (清除 PRIMASK)

25.4.7.3 DMB

数据内存屏障。

25.4.7.3.1 语法

DMB

25.4.7.3.2 操作

DMB 用作数据内存屏障。它可确保会先检测到程序中位于 DMB 指令前的所有显式内存访问指令, 然后再检测到程序中位于 DMB 指令后的显式内存访问指令。DMB 不影响其他指令 (不访问内存的指令) 在处理器上的执行顺序。

25.4.7.3.3 限制

该指令没有限制。

25.4.7.3.4 条件标志

该指令不会改变标志。

25.4.7.3.5 示例

DMB ; 数据内存屏障

25.4.7.4 DSB

数据同步屏障。

25.4.7.4.1 语法

DSB

25.4.7.4.2 操作

DSB 用作特别数据同步内存屏障。只有当此指令执行完毕后, 才会执行程序位于此指令后的指令。位于此指令前的所有显式内存访问均完成时, DSB 指令才会完成。

25.4.7.4.3 限制

该指令没有限制。

25.4.7.4.4 条件标志

该指令不会改变标志。

25.4.7.4.5 示例

DSB ; 数据同步屏障

25.4.7.5 ISB

指令同步屏障。

25.4.7.5.1 语法

ISB

25.4.7.5.2 操作

ISB 用作指令同步屏障。它会刷新处理器的管道，因此在完成了 ISB 指令后，需要再次将 ISB 之后的所有指令从高速缓存或内存中提取出来。

25.4.7.5.3 限制

该指令没有限制。

25.4.7.5.4 条件标志

该指令不会改变标志。

25.4.7.5.5 示例

ISB ; 指令同步屏障

25.4.7.6 MRS

将特别寄存器的内容移动到通用寄存器中。

25.4.7.6.1 语法

MRS *Rd*, *spec_reg*

其中：

Rd 是通用目标寄存器。

spec_reg 是其中一个特别寄存器：APSR、IPSR、EPSR、IEPSR、IAPSR、EAPSR、PSR、MSP、PSP、PRIMASK 或 CONTROL。

25.4.7.6.2 操作

MRS 将特别寄存器的内容存放到通用寄存器中。MRS 指令可以结合 MR 指令来产生读 - 修改 - 写序列，这适用于在 PSR 中修改特别标志。

见 [章节 25–25.4.7.7](#)。

25.4.7.6.3 限制

在该指令中，*Rd* 必须不能是 SP 或 PC。

25.4.7.6.4 条件标志

该指令不会改变标志。

25.4.7.6.5 示例

MRS R0, PRIMASK ; 读取 PRIMASK 值并将其写入 R0

25.4.7.7 MSR

将通用寄存器的内容传移到指定的特别寄存器中。

25.4.7.7.1 语法

MSR *spec_reg*, *Rn*

其中：

Rd 是通用源寄存器。

spec_reg 是特别目的寄存器：APSR、IPSR、EPSR、IEPSR、IAPSR、EAPSR、PSR、MSP、PSP、PRIMASK 或 CONTROL。

25.4.7.7.2 操作

MSR 使用 *Rn* 所指定的寄存器的值来更新其中一个特别寄存器。

见[章节 25–25.4.7.6](#)。

25.4.7.7.3 限制

在该指令中，*Rn* 必须不能是 SP，且不能是 PC。

25.4.7.7.4 条件标志

该指令明确地根据 *Rn* 中的值来更新标志。

25.4.7.7.5 示例

MSR CONTROL, R1 ; 读取 R1 的值，并将其写入 CONTROL 寄存器

25.4.7.8 NOP

无操作。

25.4.7.8.1 语法

NOP

25.4.7.8.2 操作

NOP 执行的是无操作，且不能保证会占用指令时间。处理器可在它到达执行阶段之前将其从管道中移除。

使用 NOP 指令来进行填充，例如，在 64 位边界上放置后续指令。

25.4.7.8.3 限制

该指令没有限制。

25.4.7.8.4 条件标志

该指令不会改变标志。

25.4.7.8.5 示例

NOP ; 无操作

25.4.7.9 SEV

发送事件。

25.4.7.9.1 语法

SEV

25.4.7.9.2 操作

SEV将事件信号发送到一个多处理器系统内的所有处理器中。它还可设置局部事件寄存器，见[章节 25–25.3.5](#)。

另请参见[章节 25–25.4.7.11](#)。

25.4.7.9.3 限制

该指令没有限制。

25.4.7.9.4 条件标志

该指令不会改变标志。

25.4.7.9.5 示例

```
SEV ; 发送事件
```

25.4.7.10 SVC

超级用户调用。

25.4.7.10.1 语法

SVC #*imm*

其中：

imm 是 0-255 范围内的整数。

25.4.7.10.2 操作

SVC 指令会引发 SVC 异常。

处理器会忽略 *imm*。如果有需要，可以通过异常处理程序获取 *imm* 来决定要请求什么样的服务程序。

25.4.7.10.3 限制

该指令没有限制。

25.4.7.10.4 条件标志

该指令不会改变标志。

25.4.7.10.5 示例

```
SVC #0x32 ; 超级用户调用 (SVC 处理程序使用堆栈的 PC 来锁定立即数的位置，  
; 然后将其提取出来)
```

25.4.7.11 WFE

等待事件。

注：WFE 指令不能在 LPC11U1x 上使用。

25.4.7.11.1 语法

WFE

25.4.7.11.2 操作

如果事件寄存器为 0，则 WFE 挂起执行，直至发生下列其中的一个事件：

- 出现异常，除非异常屏蔽寄存器或当前优先级级别将其屏蔽
- 异常进入挂起状态，如果系统控制寄存器的 SEVONPEND 置位
- 存在调试进入请求，如果调试使能的话
- 外设或多处理器系统里另一个处理器通过使用 SEV 指令来发出事件信号。

如果事件寄存器为 1，则 WFE 将其清除为 0 并立即完成操作。

更多信息请见[章节 25–25.3.5](#)。

注：WFE 的目的只是用于节约功率。当写软件时，假定 WFE 作为 NOP 运行。

25.4.7.11.3 限制

该指令没有限制。

25.4.7.11.4 条件标志

该指令不会改变标志。

25.4.7.11.5 示例

WFE ; 等待事件

25.4.7.12 WFI

等待中断。

25.4.7.12.1 语法

WFI

25.4.7.12.2 操作

WFI

挂起执行，直至发生下列其中的一个事件：

- 异常
- 中断变为挂起状态，如果 PRIMASK 被清除，则该中断占用优先权
- 存在调试进入请求，无论调试是否使能。

注：WFI 的目的只是用于节约功率。当写软件时，假定 WFI 作为 NOP 运行。

25.4.7.12.3 限制

该指令没有限制。

25.4.7.12.4 条件标志

该指令不会改变标志。

25.4.7.12.5 示例

```
WFI ; 等待中断
```

25.5 外设

25.5.1 关于 ARM Cortex-M0

专用外设总线 (PPB) 的地址映射为：

表 375. 内核外设寄存器区

地址	内核外设	描述
0xE000E008-0xE000E00F	系统控制块	表 25-384
0xE000E010-0xE000E01F	系统定时器	表 25-393
0xE000E100-0xE000E4EF	嵌套向量中断控制器	表 25-376
0xE000ED00-0xE000ED3F	系统控制块	表 25-384
0xE000EF00-0xE000EF03	嵌套向量中断控制器	表 25-376

在寄存器描述中，寄存器的类型有以下几种：

- RW — 可读和可写。
- RO — 只读。
- WO — 只写。

25.5.2 嵌套向量中断控制器

本节描述嵌套向量中断控制器 (NVIC) 以及它使用的寄存器。NVIC 支持：

- 32 个中断。
- 每个中断的优先级可编程为 0-3 四种级别。级别越高对应的优先级越低。因此，级别 0 是最高的中断优先级。
- 中断信号的电平和脉冲检测。
- 中断末尾连锁。
- 一个外部不可屏蔽中断 (NMI)。

处理器在异常进入时自动使它的状态入栈，在异常退出时自动使它的状态出栈，无需采用任何指令。这就实现了低延迟的异常处理。NVIC 的硬件寄存器有：

表 376. NVIC 寄存器汇总

地址	名称	类型	复位值	描述
0xE000E100	ISER	RW	0x00000000	章节 25–25.5.2.2
0xE000E180	ICER	RW	0x00000000	章节 25–25.5.2.3
0xE000E200	ISPR	RW	0x00000000	章节 25–25.5.2.4
0xE000E280	ICPR	RW	0x00000000	章节 25–25.5.2.5
0xE000E400 – 0xE000E41C	IPR0-7	RW	0x00000000	章节 25–25.5.2.6

25.5.2.1 使用 CMSIS 访问 Cortex-M0 NVIC 寄存器

CMSIS 函数允许在不同的 Cortex-M 系列处理器之间进行软件移植。

当利用 CMSIS 来访问 NVIC 寄存器时要用到以下函数：

表 377. CMSIS 访问 NVIC 函数

CMSIS 函数	描述
void NVIC_EnableIRQ(IRQn_Type IRQn) ^[1]	使能一个中断或异常。
void NVIC_DisableIRQ(IRQn_Type IRQn) ^[1]	禁能一个中断或异常。
void NVIC_SetPendingIRQ(IRQn_Type IRQn) ^[1]	将中断或异常的挂起状态设为 1。
void NVIC_ClearPendingIRQ(IRQn_Type IRQn) ^[1]	将中断或异常的挂起状态清零。
uint32_t NVIC_GetPendingIRQ(IRQn_Type IRQn) ^[1]	读取中断或异常的挂起状态。 如果挂起状态被设为 1，这个函数就返回非零值。
void NVIC_SetPriority(IRQn_Type IRQn, uint32_t priority) ^[1]	将一个优先级可配置的中断或异常的优先级设置为级别 1。
uint32_t NVIC_GetPriority(IRQn_Type IRQn) ^[1]	读取一个优先级可配置的中断或异常的优先级。这个函数返回当前的优先级级别。

[1] 输入参数 IRQn 是 IRQ 编号，更多信息请见[表 363](#)。

25.5.2.2 中断设置 - 使能寄存器

ISER 使能中断，并显示哪些中断被使能。有关寄存器属性请见[表 376](#) 中的寄存器汇总。

寄存器的位分配如下所示：

表 378. ISER 的位分配

位	名称	功能
[31:0]	SETENA	中断设置 - 使能位。 写： 0 = 无影响 1 = 使能中断。 读： 0 = 中断被禁能 1 = 中断被使能。

如果一个挂起中断被使能，NVIC 就根据它的优先级来激活该中断。如果一个中断未被使能，使该中断的中断信号有效可将中断的状态变成挂起，但是，不管这个中断的优先级如何，NVIC 都不会激活该中断。

25.5.2.3 中断清除 - 使能寄存器

ICER 禁能中断，并显示哪些中断被使能。有关寄存器属性请见[表 25–376](#) 中的寄存器汇总。

寄存器的位分配如下所示：

表 379. ICER 的位分配

位	名称	功能
[31:0]	CLRENA	中断清除 - 使能位。 写： 0 = 无影响 1 = 禁能中断。 读： 0 = 中断被禁能 1 = 中断被使能。

25.5.2.4 中断设置 - 挂起寄存器

ISPR 强制中断进入挂起状态，并显示哪些中断正在挂起。有关寄存器属性请见[表 25-376](#)中的寄存器汇总。

寄存器的位分配如下所示：

表 380. ISPR 的位分配

位	名称	功能
[31:0]	SETPEND	中断设置 - 挂起位。 写： 0 = 无影响 1 = 将中断状态变为挂起。 读： 0 = 中断没有挂起 1 = 中断正在挂起。

注：向 ISPR 位写 1 相当于下面两种情况：

- 正在挂起的中断不会有任何影响
- 被禁能的中断会将中断的状态设置成挂起。

25.5.2.5 中断清除 - 挂起寄存器

ICPR 使中断离开挂起状态，并显示哪些中断正在挂起。有关寄存器属性请见[表 25-376](#)中的寄存器汇总。

寄存器的位分配如下所示：

表 381. ICPR 的位分配

位	名称	功能
[31:0]	CLRPEND	中断清除 - 挂起位。 写： 0 = 无影响 1 = 清除中断的挂起状态。 读： 0 = 中断没有挂起 1 = 中断正在挂起。

注：向 ICPR 位写 1 不影响相应中断的有效状态。

25.5.2.6 中断优先级寄存器

IPR0-IPR7 寄存器为每个中断提供了一个两位的优先级域。这些寄存器只能字访问。关于它们的属性请见表 25-376 中的寄存器汇总。每个寄存器包含 4 个优先级域，如下所示：

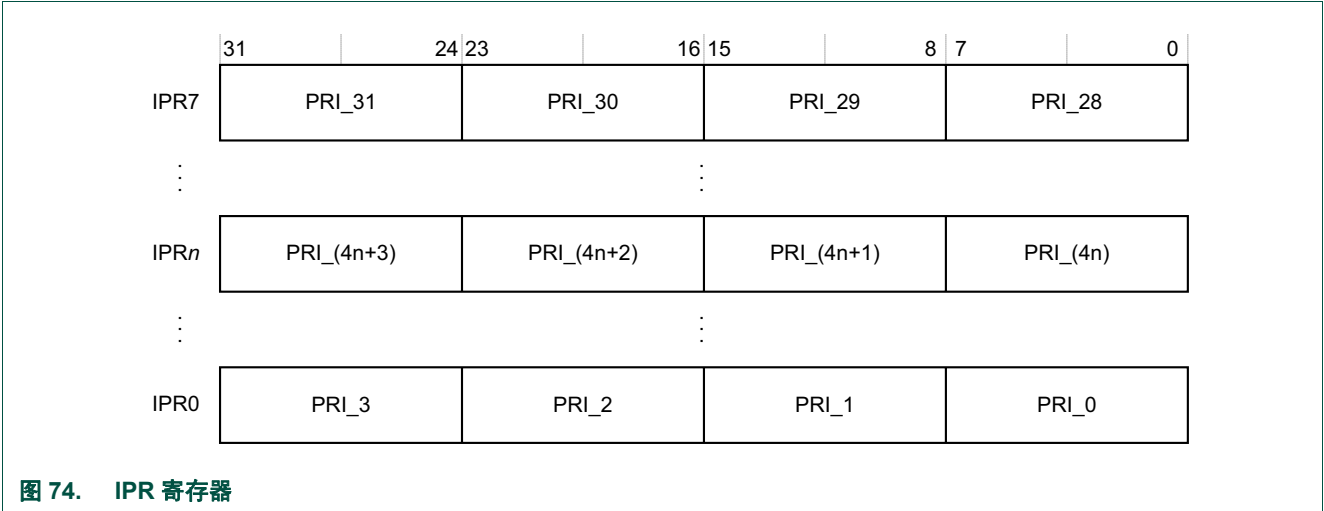


图 74. IPR 寄存器

表 382. IPR 的位分配

位	名称	功能
[31:24]	优先级，字节偏移量 3	每个优先级域保存一个优先级值 (0-3)。值越小，对应中断的优先级越高。处理器只使用每个域的位 [7:6]，位 [5:0] 读出的 0，写操作被忽略。
[23:16]	优先级，字节偏移量 2	
[15:8]	优先级，字节偏移量 1	
[7:0]	优先级，字节偏移量 0	

有关中断优先级阵列（提供了中断优先级的软件视角）访问的更多信息，请见[章节 25-25.5.2.1](#)。

使用下面的方法为中断 **M** 找出 IPR 编号和字节偏移量：

- 相应的 IPR 编号 **N**，通过等式 $N = M \text{ DIV } 4$ 得出
- 这个寄存器中所需优先级域的字节偏移量是 $M \text{ MOD } 4$ （**M** 除以 4 取余），在这里：
 - 字节偏移量 0 指的是寄存器位 [7:0]
 - 字节偏移量 1 指的是寄存器位 [15:8]
 - 字节偏移量 2 指的是寄存器位 [23:16]
 - 字节偏移量 3 指的是寄存器位 [31:24]。

25.5.2.7 电平有效的中断和脉冲中断

处理器支持电平有效的中断和脉冲中断。脉冲中断也被描述成边沿触发的中断。

一个电平有效的中断一直保持有效，直至外设将中断信号撤销。通常，发生这种情况的原因是 ISR 访问外设导致外设将中断请求清除。脉冲中断是在处理器时钟的上升沿同时采样得到的一个中断信号。为了确保 NVIC 检测到中断，外设必须使中断信号至少在一个时钟周期内保持有效，在这段时间内 NVIC 检测脉冲并锁存中断。

当处理器进入 ISP 时，它自动消除中断的挂起状态，见[章节 25.5.2.7.1](#)。对于一个电平有效的中断，如果在处理器从 ISR 返回之前中断信号未被撤销，中断就再次变成挂起，处理器必须再次执行 ISR。这就表示，外设可以一直使中断信号保持有效，直到它不再需要服务为止。

25.5.2.7.1 中断的硬件和软件控制

Cortex-M0 锁存所有的中断。外设中断会由于下面的其中一个原因而变为挂起：

- NVIC 检测到中断信号有效，而相应的中断无效
- NVIC 检测到中断信号的一个上升沿
- 软件向相应的中断设置 - 挂起寄存器位写入值，见[章节 25–25.5.2.4](#)。

挂起的中断一直保持挂起，直到出现以下其中一种情况：

- 处理器进入中断的 ISR。这就使中断的状态从挂起变为有效。而且：
 - 对于电平有效的中断，当处理器从 ISR 返回时，NVIC 采样中断信号。如果中断信号有效，中断的状态变回挂起，这可能使得处理器立刻再次进入 ISR。否则，中断的状态变为无效。
 - 对于脉冲中断，NVIC 继续监测中断信号，如果这个中断信号一直处于脉冲状态，中断的状态就变成挂起和有效。在这种情况下，当处理器从 ISR 返回时，中断的状态变为挂起，这可能使得处理器立刻重新进入 ISR。
 - 如果当处理器在处理 ISR 时中断信号的脉冲就不存在了，那么，当处理器从 ISR 返回时中断的状态变为无效。
- 软件向相应的中断清除 - 挂起寄存器位写入值。

对于电平有效的中断，如果中断信号仍然有效，中断的状态不改变。否则，中断的状态变为无效。

对于脉冲中断，中断的状态变为：

- 无效（如果中断之前的状态是挂起）
- 有效（如果中断之前的状态是有效和挂起）。

25.5.2.8 NVIC 的使用提示和技巧

保证软件正确使用对齐的寄存器访问。处理器不支持非对齐的 NVIC 寄存器访问。

中断即使被禁能也可以进入挂起状态。禁能一个中断只阻止处理器处理中断。

25.5.2.8.1 NVIC 编程提示

软件使用 CPSIE i 和指令来使能和禁能中断。CMSIS 为这些指令提供了以下内在函数：

```
void __disable_irq(void) // 禁能中断
```

```
void __enable_irq(void) // 使能中断
```

另外，CMSIS 提供了许多 NVIC 控制函数，包括：

表 383. CMSIS 的 NVIC 控制函数

CMSIS 中断控制函数	描述
void NVIC_EnableIRQ (IRQn_t IRQn)	使能 IRQn
void NVIC_DisableIRQ (IRQn_t IRQn)	禁能 IRQn
uint32_t NVIC_GetPendingIRQ (IRQn_t IRQn)	如果 IRQn 正在挂起，返回真 (1)
void NVIC_SetPendingIRQ (IRQn_t IRQn)	设置 IRQn 挂起状态
void NVIC_ClearPendingIRQ (IRQn_t IRQn)	清除 IRQn 挂起状态
void NVIC_SetPriority (IRQn_t IRQn, uint32_t priority)	设置 IRQn 的优先级
uint32_t NVIC_GetPriority (IRQn_t IRQn)	读取 IRQn 的优先级
void NVIC_SystemReset (void)	复位系统

输入参数 IRQn 是 IRQ 编号，更多信息请见表 25–363。有关这些函数的更多信息，请见 CMSIS 的资料。

25.5.3 系统控制块

系统控制块(SCB)提供了系统执行信息和系统控制，包括配置、控制和系统异常的报告 SCB 寄存器有：

表 384. SCB 寄存器汇总

地址	名称	类型	复位值	描述
0xE000ED00	CPUID	RO	0x410CC200	章节 25.5.3.2
0xE000ED04	ICSR	RW ^[1]	0x00000000	章节 25–25.5.3.3
0xE000ED0C	AIRCR	RW ^[1]	0xFA050000	章节 25–25.5.3.4
0xE000ED10	SCR	RW	0x00000000	章节 25–25.5.3.5
0xE000ED14	CCR	RO	0x00000204	章节 25–25.5.3.6
0xE000ED1C	SHPR2	RW	0x00000000	章节 25–25.5.3.7.1
0xE000ED20	SHPR3	RW	0x00000000	章节 25–25.5.3.7.2

[1] 更多信息请见寄存器描述。

25.5.3.1 Cortex-M0 SCB 寄存器的 CMSIS 映射

为了提高软件效率，CMSIS 简化了 SCB 寄存器的表现形式。在 CMSIS 中，阵列 SHP[1] 对应寄存器 SHPR2-SHPR3。

25.5.3.2 CPUID 寄存器

CPUID 寄存器包含处理器的部件号、版本和实现信息。有关其属性，请见中的寄存器汇总。寄存器的位分配如下所示：

表 385. CPUID 寄存器的位分配

位	名称	功能
[31:24]	Implementer	实现代码： 0x41 = ARM
[23:20]	Variant	更新编号，产品版本标识符 rnpm 中 r 的值： 0x0 = 版本 0

表 385. CPUID 寄存器的位分配

位	名称	功能
[19:16]	Constant	定义处理器架构的常量；读取的结果是： 0xC = ARMv6-M 架构
[15:4]	Partno	处理器的部件号： 0xC20 = Cortex-M0
[3:0]	Revision	修订编号，产品版本标识符 rnppn 中 p 的值： 0x0 = Patch 0

25.5.3.3 中断控制和状态寄存器

ICSR:

- 提供了：
 - 为不可屏蔽中断 (NMI) 异常提供了一个设置 - 挂起位
 - 为 PendSV 和 SysTick 异常提供了设置 - 挂起位和清除 - 挂起位
- 指明了：
 - 正在处理的异常的异常编号
 - 是否有被抢占的有效异常
 - 最高优先级挂起异常的异常编号
 - 是否有任何异常正在挂起。

有关 ICSR 的属性请见[表 25–384](#) 中的寄存器汇总。寄存器的位分配如下所示：

表 386. ICSR 的位分配

位	名称	类型	功能
[31]	NMIPENDSET	RW	NMI 设置 - 挂起位。 写： 0 = 无影响 1 = 将 NMI 异常的状态变为挂起。 读： 0 = NMI 异常未挂起 1 = NMI 异常正在挂起。 由于 NMI 是优先级最高的异常，因此，一般情况下，处理器一旦检测到向该位写 1 就立刻进入 NMI 异常处理程序。处理器进入处理程序后将该位清零。这就表示，只有当 NMI 信号在处理器正在执行 NMI 异常处理程序的过程中再次有效，通过异常处理程序读取这个位才返回 1。
[30:29]	-	-	保留。
[28]	PENDSVSET	RW	PendSV 设置 - 挂起位。 写： 0 = 无影响 1 = 将 PendSV 异常的状态变为挂起。 读： 0 = PendSV 异常未挂起 1 = PendSV 异常正在挂起。 向该位写 1 是将 PendSV 异常状态设为挂起的唯一方法。

表 386. ICSR 的位分配 (续)

位	名称	类型	功能
[27]	PENDSVCLR	WO	PendSV 清除 - 挂起位。 写： 0 = 无影响 1 = 撤销 PendSV 异常的挂起状态。
[26]	PENDSTSET	RW	SysTick 异常设置 - 挂起位。 写： 0 = 无影响 1 = 将 SysTick 异常的状态变为挂起。 读： 0 = SysTick 异常未挂起 1 = SysTick 异常正在挂起。
[25]	PENDSTCLR	WO	SysTick 异常清除 - 挂起位。 写： 0 = 无影响 1 = 撤销 SysTick 异常的挂起状态。 该位只可写。当对这个寄存器执行读操作时，该位读出的值不可知。
[24:23]	-	-	保留。
[22]	ISRPENDING	RO	除 NMI 和故障之外的中断的挂起标志： 0 = 中断未挂起 1 = 中断正在挂起。
[21:18]	-	-	保留。
[17:12]	VECTPENDING	RO	指示优先级最高的、正在挂起的并且使能的异常的异常编号： 0 = 没有正在挂起的异常 非零 = 优先级最高的、正在挂起的并且使能的异常的异常编号。
[11:6]	-	-	保留。
[5:0]	VECTACTIVE ^[1]	RO	包含有效的异常编号： 0 = 线程模式 非零 = 当前有效异常的异常编号 ^[1] 。 注： 这个值减去 16 得到 CMSIS IRQ 编号，编号标识出对应在中断清除 - 使能、设置 - 使能、清除 - 挂起、设置 - 挂起以及优先级寄存器中的位，请见表 25-358。

[1] 这个值与 IPSR 位 [5:0] 的值相同，请见表 25-358。

写 ICSR 时，如果执行下列操作，结果将不可知：

- 写 1 到 PENDSVSET 位和写 1 到 PENDSVCLR 位
- 写 1 到 PENDSTSET 位和写 1 到 PENDSTCLR 位。

25.5.3.4 应用中断和复位控制寄存器

AIRCR 提供了数据访问的字节顺序状态和系统的复位控制信息。有关其属性，请见表 25-384 和表 25-387 中的寄存器汇总。

如果要写这个寄存器，必须先向 VECTKEY 域写入 0x05FA，否则，处理器会将写操作忽略。
寄存器的位分配如下所示：

表 387. AIRCR 的位分配

位	名称	类型	功能
[31:16]	读：保留 写：VECTKEY	RW	寄存器码： 读出的值不可知 执行写操作时将 0x05FA 写入 VECTKEY，否则写操作被忽略。
[15]	ENDIANESS	RO	采用的数据字节存储顺序： 0 = 小端 1 = 大端。
[14:3]	-	-	保留
[2]	SYSRESETREQ	WO	系统复位请求： 0 = 无影响 1 = 请求一个系统级复位。 这个位读出为 0。
[1]	VECTCLRACTIVE	WO	保留供调试使用。这个位读出为 0。当写这个寄存器时，必须向这个位写 0，否则操作将不可预知。
[0]	-	-	保留。

25.5.3.5 系统控制寄存器

SCR 控制着低功耗状态的进入和退出特性。有关其属性，请见[表 25-384](#) 中的寄存器汇总。
寄存器的位分配如下所示：

表 388. SCR 的位分配

位	名称	功能
[31:5]	-	保留。
[4]	SEVONPEND	挂起时发送事件位： 0 = 只有使能的中断或事件能够唤醒处理器，不接受禁能中断的唤醒。 1 = 使能的事件和包括禁能中断在内的所有中断都能唤醒处理器。 当一个事件或中断进入挂起状态时，事件信号将处理器从 WFE 唤醒。 如果处理器并未在等待一个事件，事件被记录，影响下个 WFE。 处理器也可以在执行 SEV 指令时唤醒。
[3]	-	保留。
[2]	SLEEPDEEP	控制处理器是将睡眠模式还是深度睡眠模式作为低功耗模式： 0 = 睡眠 1 = 深度睡眠。
[1]	SLEEPONEXIT	指示当从处理程序模式返回到线程模式时 sleep-on-exit（退出时进入睡眠）： 0 = 处理器返回到线程模式时不进入睡眠。 1 = 处理器从 ISR 返回到线程模式时进入睡眠或深度睡眠。 将该位设为 1 允许一个中断驱动的应用程序避免返回到一个空的主应用程序。
[0]	-	保留。

25.5.3.6 配置和控制寄存器

CCR 是一个只读寄存器，指出了 Cortex-M0 处理器行为的一些情况。有关 CCR 属性，请见表 25-384 中的寄存器汇总。

寄存器的位分配如下所示：

表 389. CCR 的位分配

位	名称	功能
[31:10]	-	保留。
[9]	STKALIGN	该位读出总是为 0，指示进入异常时堆栈按 8 字节对齐。 进入异常时，处理器使用入栈的 PSR 的位 [9] 来指示栈对齐。从异常中返回时，处理器使用这个入栈的位来恢复正确的栈对齐。
[8:4]	-	保留。
[3]	UNALIGN_TRP	该位读出总是为 0，指示所有未对齐的访问产生一个 HardFault。
[2:0]	-	保留。

25.5.3.7 系统处理程序优先级寄存器

SHPR2-SHPR3 寄存器设置优先级可配置的异常处理程序的优先级级别 (0-3)。

SHPR2-SHPR3 是字可访问的。关于它们的属性请见表 25-384 中的寄存器汇总。

利用 CMSIS 访问系统异常的优先级级别要用到以下 CMSIS 函数：

- uint32_t NVIC_GetPriority(IRQn_Type IRQn)
- void NVIC_SetPriority(IRQn_Type IRQn, uint32_t priority)

输入参数 IRQn 是 IRQ 编号，更多信息请见表 25-363。

系统故障处理程序、优先级域以及每个处理程序的寄存器如下所示：

表 390. 系统故障处理程序优先级域

处理机模式	域	寄存器描述
SVCall	PRI_11	章节 25-25.5.3.7.1
PendSV	PRI_14	章节 25-25.5.3.7.2
SysTick	PRI_15	

每个 PRI_N 域 8 位宽，但处理器只使用每个域的位 [7:6]，位 [5:0] 读出为 0，写操作被忽略。

25.5.3.7.1 系统处理程序优先级寄存器 2

寄存器的位分配如下所示：

表 391. SHPR2 寄存器的位分配

位	名称	功能
[31:24]	PRI_11	系统处理程序 11 (SVCall) 的优先级
[23:0]	-	保留

25.5.3.7.2 系统处理程序优先级寄存器 3

寄存器的位分配如下所示：

表 392. SHPR3 寄存器的位分配

位	名称	功能
[31:24]	PRI_15	系统处理程序 15（SysTick 异常）的优先级
[23:16]	PRI_14	系统处理程序 14 (PendSV) 的优先级
[15:0]	-	保留

25.5.3.8 SCB 使用的提示和技巧

保证软件使用对齐的 32 位字大小传输来访问所有的 SCB 寄存器。

25.5.4 系统定时器， SysTick

当系统定时器被使能时，定时器从重装值开始递减计数到零，下个时钟周期再将 SYST_RVR 的值重新加载到定时器（一个回合），然后在后面的时钟周期下继续开始递减计数。向 SYST_RVR 写零会使计数器在下个回合禁能。当计数器跳变到零时， COUNTFLAG 状态位被设为 1。读 SYST_CSR 将 COUNTFLAG 位清零。

写 SYST_CVR 会将该寄存器和 COUNTFLAG 状态位都清零。这个写操作不触发 SysTick 异常逻辑。读取 SYST_CVR 寄存器返回的是读取操作执行当下的寄存器值。

注：当寄存器由于调试而被终止时，计数器不递减计数。

系统定时器寄存器有：

表 393. 系统定时器寄存器汇总

地址	名称	类型	复位值	描述
0xE000E010	SYST_CSR	RW	0x00000000	章节 25.5.4.1
0xE000E014	SYST_RVR	RW	不可知	章节 25–25.5.4.2
0xE000E018	SYST_CVR	RW	不可知	章节 25–25.5.4.3
0xE000E01C	SYST_CALIB	RO	0xC0000000 ^[1]	章节 25–25.5.4.4

[1] SysTick 的校准值。

25.5.4.1 SysTick 控制和状态寄存器

SYST_CSR 使能 SysTick 特性。有关其属性，请见中的寄存器汇总。寄存器的位分配如下所示：

表 394. SYST_CSR 的位分配

位	名称	功能
[31:17]	-	保留。
[16]	COUNTFLAG	如果自从上次读这个寄存器之后定时器计数到 0，该位就返回 1。
[15:3]	-	保留。

表 394. SYST_CSR 的位分配 (续)

位	名称	功能
[2]	CLKSOURCE	选择 SysTick 定时器的时钟源： 0 = 外部基准时钟 1 = 处理器时钟。
[1]	TICKINT	使能 SysTick 异常请求： 0 = 计数到零不提交 SysTick 异常请求 1 = 计数到零提交 SysTick 异常请求
[0]	ENABLE	使能计数器： 0 = 计数器被禁能 1 = 计数器被使能。

25.5.4.2 SysTick 重装值寄存器

SYST_RVR 设定了加载到 SYST_CVR 的起始值。有关其属性，请见表 25–393 中的寄存器汇总。寄存器的位分配如下所示：

表 395. SYST_RVR 的位分配

位	名称	功能
[31:24]	-	保留。
[23:0]	RELOAD	当计数器被使能且计数值到达 0 时加载到 SYST_CVR 的值，请见 章节 25.5.4.2.1 。

25.5.4.2.1 计算 RELOAD 值

RELOAD 值可以是 0x00000001-0x00FFFFFF 范围内的任何值。您可以将 RELOAD 的值设为 0，这不会产生任何影响，因为计数值从 1 变为 0 时 SysTick 异常请求和 COUNTFLAG 都被激活了。

如果要产生一个周期为 N 个处理器时钟周期的多次触发定时器，就可以将 RELOAD 值设为 N-1。例如，如果要求每隔 100 个时钟脉冲就触发一次 SysTick 中断，可将 RELOAD 设为 99。

25.5.4.3 SysTick 当前值寄存器

SYST_CVR 包含 SysTick 计数器的当前值。有关其属性，请见表 25–393 中的寄存器汇总。寄存器的位分配如下所示：

表 396. SYST_CVR 的位分配

位	名称	功能
[31:24]	-	保留。
[23:0]	CURRENT	读取时返回 SysTick 计数器的当前值。 向这个域写入任何值都会将该域清零，还会清零 SYST_CSR 的 COUNTFLAG 位。

25.5.4.4 SysTick 校准值寄存器

SYST_CALIB 寄存器指明了 SysTick 的校准特性。有关其属性，请见表 25–393 中的寄存器汇总。寄存器的位分配如下所示：

表 397. SYST_CALIB 寄存器的位分配

位	名称	功能
[31]	NOREF	该位读出为 0。该位指明不提供独立的基准时钟。
[30]	SKEW	该位读出为 0。由于 TENMS 不可知，因此，10ms 不精确计时的校准值不能确定。这会影响 SysTick 作为软件实时时钟的适用性。
[29:24]	-	保留。
[23:0]	TENMS	该位读出为 0。该域指明校准值不可知。

如果校准信息不可知，就通过处理器时钟或外部时钟的频率来计算所需的校准值。

25.5.4.5 SysTick 的使用提示和技巧

利用中断控制器时钟来更新 SysTick 计数器。

确保软件使用字访问来访问 SysTick 寄存器。

如果在复位时没有定义 SysTick 计数器的重装值和当前值，正确的 SysTick 计数器初始化序列如下：

- 1. 设置重装值。
- 2. 清除当前值。
- 3. 设置控制和状态寄存器。

26.1 缩写

表 398. 缩写

首字母缩略词	说明
ADC	数模转换器
AHB	高级高性能总线
APB	高级外设总线
BOD	掉电检测
CRC	循环冗余检查
CITT	Comité Consultatif International Téléphonique et Télégraphique
DMA	直接存储器访问
FIFO	先进先出
GPIO	通用输入 / 输出
I/O	输入 / 输出
IRC	内部电阻电容
PLL	锁相环路
SPI	串行外设接口
SSI	串行同步接口
TTL	晶体管 - 晶体管逻辑
UART	通用异步接收器 / 发送器

26.2 参考资料

- [1] **PCF8576D** — PCF8576D 数据手册
- [2] **ARM DUI 0497A** — Cortex-M0 设备一般用户指南
- [3] **ARM DDI 0432C** — Cortex-M0 修订版 r0p0 技术参考手册

26.3 法律信息

26.3.1 定义

草案 — 本文档仅为草案。其内容仍在内部审核中，尚未正式批准，可能会有进一步修改或补充。恩智浦半导体对本文所含信息的准确性或完整性不做任何说明或保证，并对因使用此类信息而带来的后果不承担任何责任。

26.3.2 免责声明

有限担保与责任 — 本文中的信息据信是准确和可靠的。但是，恩智浦半导体对此类信息的准确性或完整性不做任何明示或暗示的说明或保证，并对因使用此信息而带来的后果不承担任何责任。

在任何情况下，对于任何间接、意外、惩罚性、特殊或衍生性损害（包括但不限于利润损失、积蓄损失、业务中断、因拆卸或更换任何产品而产生的开支或返工费用），无论此等损害是否基于侵权行为（包括过失）、担保、违约或任何其他法理，恩智浦半导体均不承担任何责任。

对于因任何原因给客户带来的任何损害，恩智浦半导体对本文所述产品的总计责任和累积责任仅限于恩智浦 “商业销售条款和条件” 所规定的范围。

修改权利 — 恩智浦半导体保留对本文所发布的信息（包括但不限于规格和产品说明）随时进行修改的权利，恕不另行通知。本文件将取代并替换之前就此提供的所有信息。

适宜使用 — 恩智浦半导体产品并非设计、授权或担保适合用于生命保障、生命关键或安全关键系统或设备，亦非设计、授权或担保适合用于在恩智浦半导体产品失效或故障时可导致人员受伤、死亡或严重财产或环境损害的应用。恩智浦半导体对在此类设备或应用中加入和 / 或使用恩智浦半导体产品不承担任何责任，客户需自行承担因加入和 / 或使用恩智浦半导体产品而带来的风险。

应用 — 本文件所载任何产品的应用只用于例证目的。此类应用如不经进一步测试或修改用于特定用途，恩智浦半导体对其适用性不做任何说明或保证。

客户负责自行利用恩智浦半导体的产品进行设计和应用，对于应用或客户产品设计，恩智浦半导体无义务提供任何协助。客户须自行负责检验恩智浦半导体的产品是否适用于其规划的应用和产品，以及是否适用于其第三方客户的规划应用和使用。客户须提供适当的设计和操作安全保障措施，以降低与应用和产品相关的风险。

对于因客户的应用或产品中的任何缺陷或故障，或者客户的第三方客户的应用或使用导致的任何故障、损害、费用或问题，恩智浦半导体均不承担任何责任。客户负责对自己基于恩智浦半导体的产品的应用和产品进行所有必要测试，以避免这些应用和产品或者客户的第三方客户的应用或使用存在任何缺陷。恩智浦不承担与此相关的任何责任。

出口管制 — 本文件以及此处所描述的产品可能受出口法规的管制。出口可能需要事先经国家主管部门批准。

26.3.3 商标

注意：所有引用的品牌、产品名称、服务名称以及商标均为其各自所有者的财产。

I²C 总线 — 徽标是恩智浦的商标。

26.4 表

表 1.	订购信息.	5	表 31.	地址 0x4004 8104) 位描述.	25
表 2.	LPC122x 订购选项	5	表 32.	IOCONFIG 滤波器时钟分频寄存器 0 到 6 (IOCONFIGCLKDIV0 到 IOCONFIGCLKDIV6、 地址 4004 8014C 到 4004 80134) 位描述	25
表 3.	LPC122x 存储器配置	7	表 33.	BOD 控制寄存器 (BODCTRL、地址 0x4004 8150) 位描述	26
表 4.	中断源与中断向量控制器的连接.	9	表 34.	系统节拍定时器校准寄存器 (SYSTCKCAL、地 址 0x4004 8154) 位描述	26
表 5.	引脚摘要.	11	表 35.	AHB 矩阵主优先级寄存器 (AHBPRI0、地址 0x4004 8158) 位描述.	27
表 6.	寄存器简介：系统控制模块 (基址 0x4004 8000)	13	表 36.	IRQ 延迟寄存器 (IRQLATENCY、地址 0x4004 8170) 位描述	27
表 7.	寄存器简介：闪存配置 (基址 0x5006 0000)	15	表 37.	NMI 中断源配置寄存器 (INTNMI、地址 0x4004 8174) 位描述	28
表 8.	系统存储器重映射寄存器 (SYSMEMREMAP、 地址 0x4004 8000) 位描述	15	表 38.	启动逻辑边沿控制寄存器 0 (STARTAPRP0、地 址 0x4004 8200) 位描述	28
表 9.	外设复位控制寄存器 (PRESETCTRL、地址 0x4004 8004) 位描述	15	表 39.	启动逻辑信号使能寄存器 0 (STARTERP0、地址 0x4004 8204) 位描述.	29
表 10.	系统 PLL 控制寄存器 (SYSPLLCTRL、地址 0x4004 8008) 位描述	16	表 40.	启动逻辑复位寄存器 0 (STARTSRP0CLR、地 址 0x4004 8208) 位描述	30
表 11.	系统 PLL 状态寄存器 (SYSPLLSTAT、地址 0x4004 800C) 位描述.	17	表 41.	启动逻辑状态寄存器 0 (STARTSRP0、地址 0x4004 820C) 位描述	31
表 12.	系统振荡器控制寄存器 (SYSOSCCTRL、地址 0x4004 8020) 位描述	17	表 42.	外设中断的启动逻辑位连接	32
表 13.	看门狗振荡器控制寄存器 (WDTOSCCTRL、地 址 0x4004 8024) 位描述.	17	表 43.	启动逻辑边沿控制寄存器 1 (STARTAPRP1、地 址 0x4004 8210) 位描述	32
表 14.	内部共振晶体控制寄存器 (IRCCTRL、地址 0x4004 8028) 位描述	18	表 44.	启动逻辑信号使能寄存器 1 (STARTERP1、地址 0x4004 8214) 位描述.	33
表 15.	系统复位状态寄存器 (SYSRESSTAT、地址 0x4004 8030) 位描述	18	表 45.	启动逻辑复位寄存器 1 (STARTSRP1CLR、地 址 0x4004 8218) 位描述	33
表 16.	系统 PLL 时钟源选择寄存器 (SYSPLLCLKSEL、 地址 0x4004 8040) 位描述	19	表 46.	启动逻辑信号状态寄存器 1 (STARTSRP1、地址 0x4004 821C) 位描述	34
表 17.	系统 PLL 时钟源更新使能寄存器 (SYSPLLCLKUEN、地址 0x4004 8044) 位描述	19	表 47.	当 WDLOCKCLK = 0 (WD 振荡器未锁定) 时， PDSLEEPCFG 寄存器的允许值	34
表 18.	主时钟源选择寄存器 (MAINCLKSEL、地址 0x4004 8070) 位描述	20	表 48.	当 WDLOCKCLK = 1 (WD 振荡器锁定) 时， PDSLEEPCFG 寄存器的允许值	34
表 19.	主时钟源更新使能寄存器 (MAINCLKUEN、地址 0x4004 8074) 位描述	20	表 49.	深度睡眠配置寄存器 (PDSLEEPCFG、地址 0x4004 8230) 位描述.	35
表 20.	系统 AHB 时钟分频寄存器 (SYSAHBCLKDIV、 地址 0x4004 8078) 位描述	20	表 50.	唤醒配置寄存器 (PDAAWAKECFG、地址 0x4004 8234) 位描述	35
表 21.	系统 AHB 时钟控制寄存器 (SYSAHBCLKCTRL、地址 0x4004 8080) 位描 述.	20	表 51.	掉电配置寄存器 (PDRUNCFG、地址 0x4004 8238) 位描述	37
表 22.	SSP 时钟分频寄存器 (SSPCLKDIV、地址 0x4004 8094) 位描述	23	表 52.	器件 ID 寄存器 (DEVICE_ID、地址 0x4004 83F4) 位描述	38
表 23.	UART0 时钟分频寄存器 (UART0CLKDIV、地址 0x4004 8098) 位描述	23	表 53.	闪存配置寄存器 (FLASHCFG、地址 0x5006 0028) 位描述	38
表 24.	UART1 时钟分频寄存器 (UART1CLKDIV、地址 0x4004 809C) 位描述.	23	表 54.	PLL 频率参数	45
表 25.	RTC 时钟分频寄存器 (RTCCLKDIV、地址 0x4004 80A0) 位描述.	24	表 55.	PLL 配置示例	46
表 26.	CLKOUT 时钟源选择寄存器 (CLKOUTCLKSEL、地址 0x4004 80E0) 位描述	24	表 56.	寄存器简介：PMU (基址 0x4003 8000)	47
表 27.	CLKOUT 时钟源更新使能寄存器 (CLKOUTUEN、地址 0x4004 80E4) 位描述	24	表 57.	电源控制寄存器 (PCON、地址 0x4003 8000) 位 描述.	47
表 28.	CLKOUT 时钟分频寄存器 (CLKOUTDIV、地址 0x4004 80E8) 位描述.	25	表 58.	通用寄存器 0 至 3 (GPREG0 - GPREG3、地址 0x4003 8004 至 0x4003 8010) 位描述.	48
表 29.	POR 捕获 PIO 状态寄存器 0 (PIOPORCAP0、 地址 0x4004 8100) 位描述	25	表 59.	系统配置寄存器 (SYSCFG、地址 0x4003 8014) 位描述.	48
表 30.	POR 捕获 PIO 状态寄存器 1 (PIOPORCAP1、 地址 0x4004 8104) 位描述.	25	表 60.	可用的 GPIO 引脚 /IOCON 寄存器	49
				寄存器简介：I/O 配置模块	

	(基址 0x4004 4000)	50	表 90.	PIO2_2 寄存器 (PIO2_2, 地址 0x4004 4078)	
表 61.	IOCON 寄存器位分配 (I ² C 引脚除外)	52		位描述	82
表 62.	IOCON 寄存器位分配 (I ² C 总线引脚 PIO0_10 和 PIO0_11)	54	表 91.	PIO2_3 寄存器 (PIO2_3, 地址 0x4004 407C)	
表 63.	PIO0_19 寄存器 (PIO0_19, 地址 0x4004 4008)			位描述	83
	位描述	55	表 92.	PIO2_4 寄存器 (PIO2_4, 地址 0x4004 4080)	
表 64.	PIO0_20 寄存器 (PIO0_20, 地址 0x4004 400C)			位描述	84
	位描述	56	表 93.	PIO2_5 寄存器 (PIO2_5, 地址 0x4004 4084)	
表 65.	PIO0_21 寄存器 (PIO0_21, 地址 0x4004 4010)			位描述	85
	位描述	57	表 94.	PIO2_6 寄存器 (PIO2_6, 地址 0x4004 4088)	
表 66.	PIO0_22 寄存器 (PIO0_22, 地址 0x4004 4014)			位描述	86
	位描述	58	表 95.	PIO2_7 寄存器 (PIO2_7, 地址 0x4004 407C)	
表 67.	PIO0_23 寄存器 (PIO0_23, 地址 0x4004 4018)			位描述	87
	位描述	59	表 96.	PIO0_10 寄存器 (PIO0_10, 地址 0x4004 4090)	
表 68.	PIO0_24 寄存器 (PIO0_24, 地址 0x4004 401C)			位描述	88
	位描述	60	表 97.	PIO0_11 寄存器 (PIO0_11, 地址 0x4004 4094)	
表 69.	SWDIO_PIO0_25 寄存器 (SWDIO_PIO0_25, 地址 0x4004 4020)			位描述	89
	位描述	61	表 98.	PIO0_12 寄存器 (PIO0_12, 地址 0x4004 4098)	
表 70.	SWCLK_PIO0_26 寄存器 (SWCLK_PIO0_26, 地址 0x4004 4024)			位描述	90
	位描述	62	表 99.	RESET_PIO0_13 寄存器 (RESET_PIO0_13, 地址 0x4004 409C)	
表 71.	PIO0_27 寄存器 (PIO0_27, 地址 0x4004 4028)			位描述	91
	位描述	63	表 100.	PIO0_14 寄存器 (PIO0_14, 地址 0x4004 40A0)	
表 72.	PIO2_12 寄存器 (PIO2_12, 地址 0x4004 402C)			位描述	92
	位描述	64	表 101.	PIO0_15 寄存器 (PIO0_15, 地址 0x4004 40A4)	
表 73.	PIO2_13 寄存器 (PIO2_13, 地址 0x4004 4030)			位描述	93
	位描述	65	表 102.	PIO0_16 寄存器 (PIO0_16, 地址 0x4004 40A8)	
表 74.	PIO2_14 寄存器 (PIO2_14, 地址 0x4004 4034)			位描述	94
	位描述	66	表 103.	PIO0_17 寄存器 (PIO0_17, 地址 0x4004 40AC)	
表 75.	PIO2_15 寄存器 (PIO2_15, 地址 0x4004 4038)			位描述	95
	位描述	67	表 104.	PIO0_18 寄存器 (PIO0_18, 地址 0x4004 40B0)	
表 76.	PIO0_28 寄存器 (PIO0_28, 地址 0x4004 403C)			位描述	96
	位描述	68	表 105.	R_PIO0_30 寄存器 (R_PIO0_30, 地址 0x4004 40B4)	
表 77.	PIO0_29 寄存器 (PIO0_29, 地址 0x4004 4040)			位描述	97
	位描述	69	表 106.	PIO0_31 寄存器 (PIO0_31, 地址 0x4004 40B8)	
表 78.	PIO0_0 寄存器 (PIO0_0, 地址 0x4004 4044)			位描述	98
	位描述	70	表 107.	R_PIO1_0 寄存器 (R_PIO1_0, 地址 0x4004 40BC)	
表 79.	PIO0_1 寄存器 (PIO0_1, 地址 0x4004 4048)			位描述	99
	位描述	71	表 108.	R_PIO1_1 寄存器 (R_PIO1_1, 地址 0x4004 40C0)	
表 80.	PIO0_2 寄存器 (PIO0_2, 地址 0x4004 404C)			位描述	100
	位描述	72	表 109.	PIO1_2 寄存器 (PIO1_2, 地址 0x4004 40C4)	
表 81.	PIO0_3 寄存器 (PIO0_3, 地址 0x4004 4054)			位描述	101
	位描述	73	表 110.	PIO1_3 寄存器 (PIO1_3, 地址 0x4004 40C8)	
表 82.	PIO0_4 寄存器 (PIO0_4, 地址 0x4004 4058)			位描述	102
	位描述	74	表 111.	PIO1_4 寄存器 (PIO1_4, 地址 0x4004 40CC)	
表 83.	PIO0_5 寄存器 (PIO0_5, 地址 0x4004 405C)			位描述	103
	位描述	75	表 112.	PIO1_5 寄存器 (PIO1_5, 地址 0x4004 40D0)	
表 84.	PIO0_6 寄存器 (PIO0_6, 地址 0x4004 4060)			位描述	104
	位描述	76	表 113.	PIO1_6 寄存器 (PIO1_6, 地址 0x4004 40D4)	
表 85.	PIO0_7 寄存器 (PIO0_7, 地址 0x4004 4064)			位描述	105
	位描述	77	表 114.	PIO2_8 寄存器 (PIO2_8, 地址 0x4004 40E0)	
表 86.	PIO0_8 寄存器 (PIO0_8, 地址 0x4004 4068)			位描述	106
	位描述	78	表 115.	PIO2_9 寄存器 (PIO2_9, 地址 0x4004 40E4)	
表 87.	PIO0_9 寄存器 (PIO0_9, 地址 0x4004 406C)			位描述	107
	位描述	79	表 116.	PIO2_10 寄存器 (PIO2_10, 地址 0x4004 40E8)	
表 88.	PIO2_0 寄存器 (PIO2_0, 地址 0x4004 4070)			位描述	108
	位描述	80	表 117.	PIO2_11 寄存器 (PIO2_11, 地址 0x4004 40EC)	
表 89.	PIO2_1 寄存器 (PIO2_1, 地址 0x4004 4074)			位描述	109
	位描述	81	表 118.	LPC122x 引脚说明	111
			表 119.	LPC12D27 LQFP100 引脚说明	117

表 120.	引脚复用	123	表 144.	UART 中断识别寄存器 (IIR - 地址 0x4004 8008, 只读) 位描述	134
表 121.	可用的 GPIO 引脚 / 端口	125	表 145.	UART 中断处理	136
表 122.	寄存器简介: GPIO (基址端口 0: 0x5000 0000; 端口 1: 0x5001 0000; 端口 2: 0x5002 0000)	126	表 146.	UART FIFO 控制寄存器 (FCR - 地址 0x4000 8008, 只写) 位描述	136
表 123.	GPIO 屏蔽寄存器 (MASK - 地址 0x5000 0000 (GPIO0)、0x5001 0000 (GPIO1)、0x5002 0000 (GPIO2)) 的位描述	126	表 147.	UART 线路控制寄存器 (LCR - 地址 0x4000 800C) 位描述	137
表 124.	GPIO 引脚值寄存器 (PIN - 地址 0x5000 0004 (GPIO0)、0x5001 0004 (GPIO1); 0x5002 0004 (GPIO2)) 的位描述	127	表 148.	UART 调制解调器控制寄存器 (MCR - 地址 0x4000 8010) 位描述	138
表 125.	GPIO 引脚输出寄存器 (OUT - 地址 0x5000 0008 (GPIO0)、0x5001 0008 (GPIO1)、0x5002 0008 (GPIO2)) 的位描述	127	表 149.	调制解调器状态中断生成	140
表 126.	GPIO 引脚输出设置寄存器 (SET - 地址 0x5000 000C (GPIO0)、0x5001 000C (GPIO1)、0x5002 000C (GPIO2)) 的位描述	127	表 150.	UART 线路状态寄存器 (LSR - 地址 0x4000 8014, 只读) 位描述	140
表 127.	GPIO 引脚输出清除寄存器 (CLR - 地址 0x5000 0010 (GPIO0)、0x5000 1010 (GPIO1)、0x5002 0010 (GPIO2)) 的位描述	128	表 151.	UART 调制解调器状态寄存器 (MSR - 地址 0x4000 8018) 位描述	142
表 128.	GPIO NOT 寄存器 (NOT - 地址 0x5000 0014 (GPIO0)、0x5001 0014 (GPIO1)、0x5002 0014 (GPIO2)) 的位描述	128	表 152.	UART 高速暂时寄存器 (SCR - 地址 0x4000 801C) 位描述	142
表 129.	GPIO 数据方向寄存器 (DIR - 地址 0x5000 0020 (GPIO0)、0x5001 0020 (GPIO1)、0x5002 0020 (GPIO2)) 的位描述	128	表 153.	自动波特率控制寄存器 (ACR - 地址 0x4000 8020) 位描述	143
表 130.	GPIO 中断感应寄存器 (IS - 地址 0x5000 0024 (GPIO0)、0x5001 0024 (GPIO1)、0x5002 0024 (GPIO2)) 的位描述	128	表 154.	UART 小数分频寄存器 (FDR - 地址 0x4000 8028) 位描述	145
表 131.	GPIO 中断双边沿感应寄存器 (IBE - 地址 0x5000 0028 (GPIO0)、0x5001 0028 (GPIO1)、0x5002 0028 (GPIO2)) 的位描述	129	表 155.	小数分频器设置查找表	148
表 133.	GPIO 中断屏蔽寄存器 (IE - 地址 0x5000 0030、0x5001 0030 (GPIO1)、0x5002 0030 (GPIO2)) 的位描述	129	表 156.	UART 发送使能寄存器 (TER - 地址 0x4000 8030) 位描述	149
表 134.	GPIO 原始中断状态寄存器 (RIS - 地址 0x5000 0034 (GPIO0)、0x5001 0034 (GPIO1)、0x5002 0034 (GPIO2)) 的位描述	129	表 157.	UART RS485 控制寄存器 (RS485CTRL - 地址 0x4000 804C) 位描述	149
表 135.	GPIO 屏蔽中断状态寄存器 (MIS - 地址 0x5000 0038 (GPIO0)、0x5001 0038 (GPIO1)、0x5002 0038 (GPIO2)) 的位描述	130	表 158.	UART RS-485 地址匹配寄存器 (RS485ADRMATCH - 地址 0x4000 8050) 位描述	150
表 136.	GPIO 中断清除寄存器 (IC - 地址 0x5000 003C、0x5001 003C (GPIO1)、0x5002 003C (GPIO2)) 的位描述	130	表 159.	UART RS-485 延迟值寄存器 (RS485DLY - 地址 0x4000 8054) 位描述	150
表 137.	UART0 引脚描述	131	表 160.	UART FIFO 电平寄存器 (FIFOLVL - 地址 0x4000 8058, 只读) 位描述	152
表 138.	寄存器简介: UART0 (基址: 0x4000 8000)	132	表 161.	UART1 引脚说明	154
表 139.	UART 接收器缓冲寄存器 (RBR - 地址 0x4000 8000, 当 DLAB = 0 时, 只读) 位描述	133	表 162.	寄存器简介: UART0 (基址: 0x4000 C000)	155
表 140.	UART 发送器保持寄存器 (THR - 地址 0x4000 8000, 当 DLAB = 0 时, 只写) 位描述	133	表 163.	UART 接收器缓冲寄存器 (RBR - 地址 0x4000 C000, 当 DLAB = 0 时, 只读) 位描述	156
表 141.	UART 除数锁存器 LSB 寄存器 (DLL - 地址 0x4000 8000, 当 DLAB = 1 时) 位描述	133	表 164.	UART 发送器保持寄存器 (THR - 地址 0x4000 C000, 当 DLAB = 0 时, 只写) 位描述	156
表 142.	UART 除数锁存器 MSB 寄存器 (DLM - 地址 0x4000 8004, 当 DLAB = 1 时) 位描述	133	表 165.	UART 除数锁存器 LSB 寄存器 (DLL - 地址 0x4000 C000, 当 DLAB = 1 时) 位描述	156
表 143.	UART 中断使能寄存器 (IER - 地址 0x4000 8004, 当 DLAB = 0 时) 位描述	134	表 166.	UART 除数锁存器 MSB 寄存器 (DLM - 地址 0x4000 C004, 当 DLAB = 1 时) 位描述	156
			表 167.	UART 中断使能寄存器 (IER - 地址 0x4000 C004, 当 DLAB = 0 时) 位描述	157
			表 168.	UART 中断识别寄存器 (IIR - 地址 0x4004 C008, 只读) 位描述	158
			表 169.	UART 中断处理	159
			表 170.	UART FIFO 控制寄存器 (FCR - 地址 0x4000 C008, 只写) 位描述	159
			表 171.	UART 线路控制寄存器 (LCR - 地址 0x4000 C00C) 位描述	160
			表 172.	UART 线路状态寄存器 (LSR - 地址 0x4000 C014, 只读) 位描述	161
			表 173.	UART 高速暂时寄存器 (SCR - 地址 0x4000 C01C) 位描述	162
			表 174.	自动波特率控制寄存器 (ACR - 地址	

0x4000 C020) 位描述.....	163	0x4004 0010) 位描述.....	214
表 175. UART IrDA 控制寄存器 (ICR - 0x4000 C024) 位描述.....	165	表 212. SSP 中断屏蔽设置 / 清除寄存器 (IMSC - 地址 0x4004 0014) 位描述.....	215
表 176. IrDA 脉冲宽度.....	166	表 213. SSP 原始中断状态寄存器 (RIS - 地址 0x4004 0018) 位描述.....	215
表 177. UART 小数分频寄存器 (FDR - 地址 0x4000 C028) 位描述.....	167	表 214. SSP 屏蔽中断状态寄存器 (MIS - 地址 0x4004 001C) 位描述.....	215
表 178. 小数分频器设置查找表.....	169	表 215. SSP 中断清除寄存器 (ICR - 地址 0x4004 0020) 位描述.....	216
表 179. UART 发送使能寄存器 (TER - 地址 0x4000 C030) 位描述.....	170	表 216. SSP DMA 控制寄存器 (DMACR - 地址 0x4004 0024) 位描述.....	216
表 180. UART FIFO 电平寄存器 (FIFOLVL - 地址 0x4000 C058, 只读) 位描述.....	170	表 217. 计数器 / 定时器引脚描述.....	225
表 181. I ² C 总线引脚描述.....	173	表 218. 比较器到 16 位计数器 / 定时器的连接.....	225
表 182. 寄存器简介: I ² C (基址 0x4000 0000).....	174	表 219. 寄存器简介: 16 位计数器 / 定时器 0 CT16B0 (基址 0x4001 0000).....	226
表 183. I ² C 控制置位寄存器 (CONSET - 地址 0x4000 0000) 位描述.....	174	表 220. 寄存器简介: 16 位计数器 / 定时器 1 CT16B1 (基址 0x4001 4000).....	227
表 184. I ² C 状态寄存器 (STAT - 0x4000 0004) 位描述.....	176	表 221. 中断寄存器 (IR, 地址 0x4001 0000 (CT16B0) 和 0x4001 4000 (CT16B1)) 位描述.....	228
表 185. I ² C 数据寄存器 (DAT - 0x4000 0008) 位描述.....	176	表 222. 定时器控制寄存器 (TCR, 地址 0x4001 0004 (CT16B0) 和 0x4001 4004 (CT16B1)) 位描述.....	228
表 186. I ² C 从属地址寄存器 0 (ADR0 - 0x4000 000C) 位描述.....	176	表 223. 定时器计数器寄存器 (TC, 地址 0x4001 0008 (CT16B0) 和 0x4001 4008 (CT16B1)) 位描述.....	228
表 187. I ² C SCL 高电平占空比寄存器 (SCLH - 地址 0x4000 0010) 位描述.....	177	表 224. 预分频寄存器 (PR, 地址 0x4001 000C (CT16B0) 和 0x4001 400C (CT16B1)) 位描述.....	229
表 188. I ² C SCL 低电平占空比寄存器 (SCLL - 0x4000 0014) 位描述.....	177	表 225. 预分频计数器寄存器 (PC, 地址 0x4001 0010 (CT16B0) 和 0x4001 4010 (CT16B1)) 位描述.....	229
表 189. 用于选择 I ² C_PCLK 值的 I2SCLL + I2SCLH 值.....	177	表 226. 匹配控制寄存器 (MCR, 地址 0x4001 0014 (CT16B0) 和 0x4001 4014 (CT16B1)) 位描述.....	229
表 190. I ² C 控制清除寄存器 (CONCLR - 0x4000 0018) 位描述.....	178	表 227. 匹配寄存器 (MR0 到 3, 地址 0x4001 0018 到 24 (CT16B0) 以及 0x4001 4018 到 24 (CT16B1)) 位描述.....	230
表 191. I ² C 监控模式控制寄存器 (MMCTRL - 0x4000 001C) 位描述.....	178	表 228. 捕获控制寄存器 (CCR, 地址 0x4001 0028 (CT16B0) 和 0x4001 4028 (CT16B1)) 位描述.....	231
表 192. I ² C 从属地址寄存器 (ADR1 - 0x4000 0020、ADR2 - 0x4000 0024、ADR3 - 0x4000 0028) 位描述.....	179	表 229. 捕获寄存器 (CR0 到 3, 地址 0x4001 002C 到 38 (CT16B0) 以及 0x4001 402C 到 38 (CT16B1)) 位描述.....	232
表 193. I ² C 数据缓冲寄存器 (DATA_BUFFER - 0x4000 002C) 位描述.....	180	表 230. 外部匹配寄存器 (EMR, 地址 0x4001 003C (CT16B0) 和 0x4001 403C (CT16B1)) 位描述.....	232
表 194. I ² C 屏蔽寄存器 (MASK0 - 0x4000 0030, MASK1 - 0x4000 0034, MASK2 - 0x4000 0038, MASK3 - 0x4000 003C) 位描述.....	180	表 231. 外部匹配控制.....	233
表 195. 用于配置主机模式的 CONSET.....	181	表 232. 计数控制寄存器 (CTCR, 地址 0x4001 0070 (CT16B0) 和 0x4001 4070 (CT16B1)) 位描述.....	234
表 196. 用于配置从机模式的 CONSET.....	182	表 233. PWM 控制寄存器 (PWMC, 地址 0x4001 0074 和 0x4001 4074 (CT16B1)) 位描述.....	235
表 197. 用于描述 I ² C 操作的缩写.....	188	表 234. 计数器 / 定时器引脚描述.....	239
表 198. 用于初始化主发送模式的 CONSET.....	189	表 235. 寄存器简介: 32 位计数器 / 定时器 0 CT32B0 (基址 0x4001 8000).....	239
表 199. 用于初始化从接收模式的 CONSET.....	193	表 236. 寄存器简介: 32 位计数器 / 定时器 1 CT32B1 (基址 0x4001 C000).....	240
表 200. 主发送模式.....	196	表 237. 中断寄存器 (IR, 地址 0x4001 8000 (CT32B0); 和 IR, 地址 0x4001 C000) 位描述.....	241
表 201. 主接收模式.....	197	表 238. 定时器控制寄存器 (TCR, 地址 0x4001 8004 (CT32B0) 和 0x4001 C004 (CT32B1)) 位描述.....	242
表 202. 从接收模式.....	198	表 239. 定时器计数器寄存器 (TC, 地址 0x4001 8008 (CT32B0) 和 0x4001 C008 (CT32B1)) 位描述.....	242
表 203. 从发送模式.....	199	表 241. 预分频寄存器 (PC, 地址 0x4001 8010 (CT32B0) 和 0x4001 C010 (CT32B1)) 位描述.....	242
表 204. 其它状态.....	200	表 242. 匹配控制寄存器 (MCR, 地址 0x4001 8014	
表 205. SSP 引脚说明.....	211		
表 206. 寄存器简介: SSP (基址 0x4004 0000).....	211		
表 207. SSP 控制寄存器 0 (CR0 - 地址 0x4004 0000) 位描述.....	212		
表 208. SSP 控制寄存器 1 (CR1 - 地址 0x4004 0004) 位描述.....	213		
表 209. SSP 数据寄存器 (DR - 地址 0x4004 0008) 位描述.....	214		
表 210. SSP 状态寄存器 (SR - 地址 0x4004 000C) 位描述.....	214		
表 211. SSP 时钟预分频寄存器 (CPSR - 地址			

表 243.	(CT32B0) 和 0x4001 C014 (CT32B1)) 位描述	243	表 272.	看门狗定时器警告中断寄存器 (WARNINT - 0x4000 4014) 位描述	267
表 244.	匹配寄存器 (MR0 到 3, 地址 0x4001 8018 到 24 (CT32B0) 以及 0x4001 C018 到 24 (CT32B1)) 位描述	244	表 273.	看门狗定时器窗口寄存器 (WINDOW - 0x4000 4018) 位描述	268
表 245.	捕获控制寄存器 (CCR, 地址 0x4001 8028 (CT32B0) 和 0x4001 C028 (CT32B1)) 位描述	244	表 274.	比较器引脚描述	271
表 246.	捕获寄存器 (CR0 到 3, 地址 0x4001 802C 到 38 (CT32B0) 以及 0x4001 C02C 到 38 (CT32B1)) 位描述	245	表 275.	寄存器简介: 比较器 (基址 0x4005 4000)	271
表 247.	外部匹配寄存器 (EMR, 地址 0x4001 803C (CT32B0) 和 0x4001 C03C (CT32B1)) 位描述	245	表 276.	比较器控制寄存器 (CMP, 地址 0x4005 4000) 位描述	271
表 248.	外部匹配控制	246	表 277.	电压阶梯寄存器 (VLAD, 地址 0x4005 4004) 位描述	273
表 249.	计数控制寄存器 (CTCR, 地址 0x4001 8070 (CT32B0) 和 0x4001 C070 (CT32B1)) 位描述	247	表 278.	ADC 引脚说明	277
表 250.	PWM 控制寄存器 (PWMC, 0x4001 8074 (CT32B0) 和 0x4001 C074 (CT32B1)) 位描述	248	表 279.	寄存器简介: ADC (基址 0x4002 0000)	278
表 251.	寄存器简介: SysTick 定时器 (基址 0xE000 E000)	253	表 280.	A/D 控制寄存器 (CR - 地址 0x4002 0000) 位描述	278
表 252.	SysTick 定时器控制和状态寄存器 (SYST_CSR - 0xE000 E010) 位描述	253	表 281.	A/D 全局数据寄存器 (GDR - 地址 0x4002 0004) 位描述	279
表 253.	系统定时器重载值寄存器 (SYST_RVR - 0xE000 E014) 位描述	254	表 282.	A/D 中断使能寄存器 (INTEN - 地址 0x4002 000C) 位描述	280
表 254.	系统定时器当前值寄存器 (SYST_CVR - 0xE000 E018) 位描述	254	表 283.	A/D 数据寄存器 (DR0 到 DR7 - 地址 0x4002 0010 到 0x4002 002C) 位描述	280
表 255.	系统定时器校准值寄存器 (SYST_CALIB - 0xE000 E01C) 位描述	254	表 284.	A/D 状态寄存器 (STAT - 地址 0x4002 0030) 位描述	280
表 256.	RTC 时钟选项	256	表 285.	A/D 微调寄存器 (TRM - 地址 0x4002 4034) 位描述	281
表 257.	寄存器简介: RTC (基址 0x4005 0000)	258	表 286.	LPC122x 的 Flash 配置	282
表 258.	RTC 数据寄存器 (DR - 地址 0x4005 0000) 位描述	258	表 287.	LPC122x 的 Flash 配置	284
表 259.	RTC 匹配寄存器 (MR - 地址 0x4005 0004) 位描述	258	表 288.	代码读保护选项	289
表 260.	RTC 加载寄存器 (LR - 地址 0x4005 0008) 位描述	258	表 289.	代码读保护硬件 / 软件的相互作用	289
表 261.	RTC 控制寄存器 (CR - 地址 0x4005 000C) 位描述	259	表 290.	不同 CRP 等级下允许使用的 ISP 命令	290
表 262.	RTC 中断屏蔽寄存器 (ICSC - 地址 0x4005 0010) 位描述	259	表 291.	ISP 命令总结	290
表 263.	RTC 中断状态寄存器 (RIS - 地址 0x4005 0014) 位描述	259	表 292.	ISP 解锁命令	291
表 264.	RTC 屏蔽中断状态寄存器 (MIS - 地址 0x4005 0018) 位描述	259	表 293.	ISP 设置波特率命令	291
表 265.	RTC 中断清除寄存器 (ICR - 地址 0x4005 001C) 位描述	260	表 294.	ISP 回应命令	291
表 266.	寄存器简介: 看门狗定时器 (基址 0x4000 4000)	264	表 295.	ISP 写 RAM 命令	292
表 267.	看门狗模式寄存器 (MOD - 0x4000 4000) 位描述	264	表 296.	ISP 读存储器命令	292
表 268.	看门狗工作模式选择	266	表 297.	ISP 准备写操作命令的扇区	293
表 269.	看门狗定时器常量寄存器 (TC - 0x4000 4004) 位描述	266	表 298.	ISP 复制命令	293
表 270.	看门狗喂狗寄存器 (FEED - 0x4000 4008) 位描述	266	表 299.	ISP 运行命令	294
表 271.	看门狗定时器值寄存器 (TV - 0x4000 400C) 位描述	267	表 300.	ISP 擦除扇区命令	294
	看门狗定时器时钟源选择寄存器 (CLKSEL - 地址 0x4000 4010) 位描述	267	表 301.	ISP 扇区查空命令	294
			表 302.	ISP 读器件标识命令	295
			表 303.	LPC122x 器件标识号	295
			表 304.	ISP 读 Boot 代码版本号命令	295
			表 305.	ISP 比较命令	296
			表 306.	读 UID 命令	296
			表 307.	ISP 返回代码总览	296
			表 308.	IAP 命令总览	298
			表 309.	IAP 准备写操作命令的扇区	299
			表 310.	IAP 将 RAM 内容复制到 Flash 命令	300
			表 311.	IAP 擦除扇区命令	300
			表 312.	IAP 扇区查空命令	301
			表 313.	IAP 读器件标识命令	301
			表 314.	IAP 读 Boot 代码版本号命令	301
			表 315.	IAP 比较命令	302
			表 316.	IAP 重新调用 ISP	302
			表 317.	IAP 读 UID 命令	302
			表 318.	IAP 擦除页命令	303
			表 319.	IAP 擦除信息页命令	303

表 320.	IAP 状态代码小结	303	表 345.	src_data_end_ptr 的位分配	321
表 321.	DMA 连接	307	表 346.	dst_data_end_ptr 的位分配	322
表 322.	寄存器简介：微 DMA（基址 0x4004 C000）	308	表 347.	channel_cfg 的位分配	322
表 323.	DMA 状态寄存器（DMA_STATUS，地址 0x4004 C000）位描述	309	表 348.	寄存器简介：CRC 引擎 （基址 0x5007 0000）	326
表 324.	DMA 配置寄存器（DMA_CFG，地址 0x4004 C004）位描述	309	表 349.	CRC 模式寄存器（MODE，地址 0x5007 0000） 位描述	326
表 325.	通道控制基址指针寄存器（CTRL_BASE_PTR， 地址 0x4004 C008）位描述	310	表 350.	CRC 种子寄存器（SEED，地址 0x5007 0004） 位描述	327
表 326.	通道备用控制基址指针寄存器 （ATL_CTRL_BASE_PTR，地址 0x4004 C00C） 位描述	310	表 351.	CRC 校验和寄存器（SUM，地址 0x5007 0008） 位描述	327
表 327.	通道应请求等待状态寄存器 （DMA_WAITONREQ_STATUS，地址 0x4004 C010）位描述	310	表 352.	CRC 数据寄存器（WR_DATA，地址 0x5007 0008）位描述	327
表 328.	通道软件请求寄存器（CHNL_SW_REQUEST， 地址 0x4004 C014）位描述	311	表 353.	串行调试接口引脚说明	331
表 329.	通道使用猝发设置寄存器 （CHNL_USEBURST_SET，地址 0x4004 C018） 位描述	311	表 354.	处理器模式和堆栈使用选项小结	335
表 330.	通道使用猝发清除寄存器 （CHNL_USEBURST_CLR，地址 0x4004 C01C） 位描述	311	表 355.	内核寄存器组小结	336
表 331.	通道请求屏蔽设置寄存器 （CHNL_REQ_MASK_SET，地址 0x4004 C020） 位描述	312	表 356.	PSR 寄存器组合	337
表 332.	通道请求屏蔽清除寄存器 （CHNL_REQ_MASK_CLR，地址 0x4004 C024） 位描述	312	表 357.	APSR 位分配	338
表 333.	通道使能设置寄存器（CHNL_ENABLE_SET，地 址 0x4004 C028）位描述	312	表 358.	IPSR 的位分配	338
表 334.	通道使能清除寄存器（CHNL_ENABLE_CLR，地 址 0x4004 C02C）位描述	313	表 359.	EPSR 位分配	338
表 335.	通道主用 - 备用设置寄存器 （CHNL_PRI_ALT_SET，地址 0x4004 C030）位 描述	313	表 360.	PRIMASK 寄存器的位分配	339
表 336.	通道主用 - 备用清除寄存器 （CHNL_PRI_ALT_CLR，地址 0x4004 C034）位 描述	314	表 361.	CONTROL 寄存器的位分配	339
表 337.	通道优先级设置寄存器 （CHNL_PRIORITY_SET，地址 0x4004 C038） 位描述	314	表 362.	存储器访问行为	343
表 338.	通道优先级清除寄存器 （CHNL_PRIORITY_CLR，地址 0x4004 C03C） 位描述	315	表 363.	各种异常类型的特性	345
表 339.	总线错误清除寄存器（ERR_CLR，地址 0x4004 C04C）位描述	315	表 364.	异常返回的行为	349
表 340.	通道 DMA 中断状态寄存器 （CHNL_IRQ_STATUS，地址 0x4004 C080）位 描述	315	表 365.	Cortex-M0 指令	351
表 341.	DMA 错误中断使能寄存器 （IRQ_ERR_ENABLE，地址 0x4004 C084）位描 述	316	表 366.	产生某些 Cortex-M0 指令的 CMSIS 内部函数	353
表 342.	通道 DMA 中断使能寄存器 （CHNL_IRQ_ENABLE，地址 0x4004 C088）位 描述	316	表 367.	访问特别寄存器的 CMSIS 内部函数	353
表 343.	DMA 控制信号	316	表 368.	条件代码后缀	358
表 344.	DMA 通道优先级	317	表 369.	访问指令	358
表 345.	src_data_end_ptr 的位分配	321	表 370.	数据处理指令	364
表 346.	dst_data_end_ptr 的位分配	322	表 371.	ADC、ADD、RSB、SBC 和 SUB 操作数限制	365
表 347.	channel_cfg 的位分配	322	表 372.	跳转和控制指令	373
表 348.	寄存器简介：CRC 引擎 （基址 0x5007 0000）	326	表 373.	跳转范围	374
表 349.	CRC 模式寄存器（MODE，地址 0x5007 0000） 位描述	326	表 374.	其他指令	374
表 350.	CRC 种子寄存器（SEED，地址 0x5007 0004） 位描述	327	表 375.	内核外设寄存器区	381
表 351.	CRC 校验和寄存器（SUM，地址 0x5007 0008） 位描述	327	表 376.	NVIC 寄存器汇总	382
表 352.	CRC 数据寄存器（WR_DATA，地址 0x5007 0008）位描述	327	表 377.	CMSIS 访问 NVIC 函数	382
表 353.	串行调试接口引脚说明	331	表 378.	ISER 的位分配	382
表 354.	处理器模式和堆栈使用选项小结	335	表 379.	ICER 的位分配	383
表 355.	内核寄存器组小结	336	表 380.	ISPR 的位分配	383
表 356.	PSR 寄存器组合	337	表 381.	ICPR 的位分配	383
表 357.	APSR 位分配	338	表 382.	IPR 的位分配	384
表 358.	IPSR 的位分配	338	表 383.	CMSIS 的 NVIC 控制函数	386
表 359.	EPSR 位分配	338	表 384.	SCB 寄存器汇总	386
表 360.	PRIMASK 寄存器的位分配	339	表 385.	CPUID 寄存器的位分配	386
表 361.	CONTROL 寄存器的位分配	339	表 386.	ICSR 的位分配	387
表 362.	存储器访问行为	343	表 387.	AIRCR 的位分配	389
表 363.	各种异常类型的特性	345	表 388.	SCR 的位分配	389
表 364.	异常返回的行为	349	表 389.	CCR 的位分配	390
表 365.	Cortex-M0 指令	351	表 390.	系统故障处理程序优先级域	390
表 366.	产生某些 Cortex-M0 指令的 CMSIS 内部函数	353	表 391.	SHPR2 寄存器的位分配	390
表 367.	访问特别寄存器的 CMSIS 内部函数	353	表 392.	SHPR3 寄存器的位分配	391
表 368.	条件代码后缀	358	表 393.	系统定时器寄存器汇总	391
表 369.	访问指令	358	表 394.	SYST_CSR 的位分配	391
表 370.	数据处理指令	364	表 395.	SYST_RVR 的位分配	392
表 371.	ADC、ADD、RSB、SBC 和 SUB 操作数限制	365	表 396.	SYST_CVR 的位分配	392

表 397. SYST_CALIB 寄存器的位分配 393

表 398. 缩写. 394

26.5 图

图 1.	LPC122x 框图	6	图 46.	RTC 与 PMU 和 SYSCON 模块的连接	257
图 2.	LPC122x 存储器映射	8	图 47.	RTC DR 寄存器复位后读取	261
图 3.	LPC122x CGU 框图	12	图 48.	看门狗功能框图	268
图 4.	系统 PLL 功能框图	44	图 49.	启用窗口模式时提前看门狗喂狗	269
图 5.	自动 RTS 功能时序	139	图 50.	启用窗口模式时正确的看门狗喂狗	269
图 6.	自动 CTS 功能时序	140	图 51.	看门狗警告中断	269
图 7.	自动波特率 a) 模式 0 和 b) 模式 1 的波形图	145	图 52.	比较器功能框图	274
图 8.	设置 UART 分频器的算法	147	图 53.	比较器输入	275
图 9.	UART 功能框图	153	图 54.	Flash 的存储器映射与 Boot ROM 存储器区域	284
图 10.	自动波特率 a) 模式 0 和 b) 模式 1 的波形图	165	图 55.	Boot 处理流程图	286
图 11.	设置 UART 分频器的算法	168	图 56.	IAP 参数传递	299
图 12.	UART1 功能框图	171	图 57.	微 DMA 控制器功能框图	306
图 13.	I ² C 总线配置	173	图 58.	DMA 乒乓周期	320
图 14.	主发送模式下的格式	181	图 59.	微 DMA 通道控制数据结构 (21 个通道) 的存储器映射	321
图 15.	主接收模式下的格式	182	图 60.	CRC 功能框图	326
图 16.	发送重复起始位后主接收切换到主发送	182	图 61.	将 SWD 引脚连接到标准的 SWD 连接器	332
图 17.	从接收模式下的格式	183	图 62.	Cortex-M0 的具体实现	333
图 18.	从发送模式下的格式	183	图 63.	处理器内核寄存器组	336
图 19.	I ² C 串行接口功能框图	184	图 64.	APSR、IPSR、EPSR 寄存器的位分配	337
图 20.	仲裁过程	186	图 65.	Cortex-M0 存储器映射	341
图 21.	串行时钟同步	186	图 66.	存储器秩序限制	342
图 22.	主发送模式下的格式和状态	190	图 67.	小端格式	344
图 23.	主接收模式下的格式和状态	192	图 68.	向量表	346
图 24.	从接收模式下的格式和状态	194	图 69.	异常进入时堆栈的内容	348
图 25.	从发送模式下的格式和状态	195	图 70.	ASR #3	355
图 26.	两个主机同时启动重复起始条件	200	图 71.	LSR #3	355
图 27.	强制访问忙 I ² C 总线	201	图 72.	LSL #3	356
图 28.	从因 SDA 低电平而导致总线受阻的状态中恢复	202	图 73.	ROR #3	356
图 29.	德州仪器同步串行帧格式: a) 单帧和 b) 连续 / 背靠背 2 帧传输	217	图 74.	IPR 寄存器	384
图 30.	CPOL=0 且 CPHA=0 时的 SPI 帧格式 (a) 单帧和 b) 连续帧传输)	218			
图 31.	CPOL=0 且 CPHA=1 时的 SPI 帧格式	219			
图 32.	CPOL = 1 且 CPHA = 0 时的 SPI 帧格式 (a) 单帧和 b) 连续帧传输)	220			
图 33.	CPOL = 1 且 CPHA = 1 时的 SPI 帧格式	221			
图 34.	Microwire 帧格式 (单帧传输)	221			
图 35.	Microwire 帧格式 (连续帧传输)	222			
图 36.	Microwire 帧格式建立及保持时间	223			
图 37.	采样 PWM 波形, 其中 PWM 周期长度为 100 (MR3 选择), MAT3:0 被 PWCON 寄存器使能为 PWM 输出	236			
图 38.	定时器周期, 其中 PR = 2、MRx = 6, 并且启用了在匹配时同时执行中断和复位操作	236			
图 39.	定时器周期, 其中 PR = 2、MRx = 6, 并且启用了在匹配时同时执行中断和停止操作	236			
图 40.	16 位计数器 / 定时器功能框图	237			
图 41.	采样 PWM 波形, 其中 PWM 周期长度为 100 (MR3 选择), MAT3:0 被 PWCON 寄存器使能为 PWM 输出	249			
图 42.	定时器周期, 其中 PR = 2、MRx = 6, 并且启用了在匹配时同时执行中断和复位操作	250			
图 43.	定时器周期, 其中 PR = 2、MRx = 6, 并且启用了在匹配时同时执行中断和停止操作	250			
图 44.	32 位计数器 / 定时器功能框图	251			
图 45.	系统节拍定时器功能框图	252			

26.6 内容

第 1 章：LPC122x 简介信息

1.1	简介	3	1.3.1	器件选择一览	5
1.2	功能特点	3	1.4	框图	6
1.3	订购信息	5			

第 2 章：LPC122x 存储器映射

2.1	本章导读	7	2.3	存储器分配	8
2.2	简介	7			

第 3 章：LPC122x 可嵌套中断向量控制器 (NVIC)

3.1	本章导读	9	3.3	描述	9
3.2	特性	9	3.4	中断源	9

第 4 章：LPC122x 系统控件 (SYSCON)

4.1	本章导读	11	4.5.33	启动逻辑状态寄存器 0	31
4.2	简介	11	4.5.34	启动逻辑边沿控制寄存器 1	32
4.3	引脚描述	11	4.5.35	启动逻辑信号使能寄存器 1	33
4.4	简介	11	4.5.36	启动逻辑复位寄存器 1	33
4.5	寄存器描述	12	4.5.37	启动逻辑状态寄存器 1	34
4.5.1	系统存储器重映射寄存器	15	4.5.38	深度睡眠模式配置寄存器	34
4.5.2	外设复位控制寄存器	15	4.5.39	唤醒配置寄存器	35
4.5.3	系统 PLL 控制寄存器	16	4.5.40	掉电配置寄存器	36
4.5.4	系统 PLL 状态寄存器	16	4.5.41	器件 ID 寄存器	37
4.5.5	系统振荡器控制寄存器	17	4.5.42	闪存配置寄存器	38
4.5.6	看门狗振荡器控制寄存器	17	4.6	复位	38
4.5.7	内部共振晶体控制寄存器	18	4.7	电源管理	39
4.5.8	系统复位状态寄存器	18	4.7.1	工作模式	39
4.5.9	系统 PLL 时钟源选择寄存器	19	4.7.1.1	工作模式下的电源配置	39
4.5.10	系统 PLL 时钟源更新使能寄存器	19	4.7.2	睡眠模式	39
4.5.11	主时钟源选择寄存器	19	4.7.2.1	睡眠模式下的电源配置	39
4.5.12	主时钟源更新使能寄存器	20	4.7.2.2	对睡眠模式进行编程	40
4.5.13	系统 AHB 时钟分频寄存器	20	4.7.2.3	从睡眠模式唤醒	40
4.5.14	系统 AHB 时钟控制寄存器	20	4.7.3	深度睡眠模式	40
4.5.15	SSP 时钟分频寄存器	23	4.7.3.1	深度睡眠模式下的电源配置	40
4.5.16	UART0 时钟分频寄存器	23	4.7.3.2	对深度睡眠模式进行编程	40
4.5.17	UART1 时钟分频寄存器	23	4.7.3.3	从深度睡眠模式唤醒	41
4.5.18	RTC 时钟分频寄存器	24	4.7.4	深度掉电模式	41
4.5.19	CLKOUT 时钟源选择寄存器	24	4.7.4.1	深度掉电模式下的电源配置	41
4.5.20	CLKOUT 时钟源更新使能寄存器	24	4.7.4.2	对深度掉电模式进行编程	41
4.5.21	CLKOUT 时钟分频寄存器	25	4.7.4.3	从深度掉电模式唤醒	42
4.5.22	POR 捕获 PIO 状态寄存器 0	25	4.8	深度睡眠模式详细信息	42
4.5.23	POR 捕获 PIO 状态寄存器 1	25	4.8.1	IRC 振荡器	42
4.5.24	IOCONFIG 时钟分频寄存器 0 到 6	25	4.8.2	使用外部引脚从深度睡眠模式唤醒 (启动逻辑 0)	42
4.5.25	BOD 控制寄存器	26	4.8.3	使用 RTC 从深度睡眠模式唤醒 (启动逻辑 1)	42
4.5.26	系统节拍计数器校准寄存器	26	4.9	深度掉电模式详细信息	43
4.5.27	AHB 矩阵主优先级寄存器	27	4.9.1	使用 WAKEUP 引脚从深度掉电模式唤醒	43
4.5.28	中断请求 (IRQ) 延迟寄存器	27	4.9.2	使用 RTC 从深度掉电模式唤醒	43
4.5.29	NMI 中断源配置寄存器	28	4.10	系统 PLL 的功能描述	44
4.5.30	启动逻辑边沿控制寄存器 0	28	4.10.1	锁定检测器	44
4.5.31	启动逻辑信号使能寄存器 0	29			
4.5.32	启动逻辑复位寄存器 0	30			

4.10.2	掉电控制	44	4.10.4	频率选择	45
4.10.3	分频器比率编程	45	4.10.4.1	正常模式	45

第 5 章：LPC122x 电源管理单元 (PMU)

5.1	本章导读	47	5.3.1	电源控制寄存器	47
5.2	简介	47	5.3.2	通用寄存器 0 至 3	48
5.3	寄存器描述	47	5.3.3	系统配置寄存器	48

第 6 章：LPC122x I/O 配置 (IOCONFIG)

6.1	本章导读	49	6.4.24	PIO0_7 寄存器	77
6.2	特性	49	6.4.25	PIO0_8 寄存器	78
6.3	简介	49	6.4.26	PIO0_9 寄存器	79
6.3.1	引脚功能	49	6.4.27	PIO2_0 寄存器	80
6.3.2	引脚模式	49	6.4.28	PIO2_1 寄存器	81
6.3.3	引脚驱动	49	6.4.29	PIO2_2 寄存器	82
6.3.4	开漏模式	50	6.4.30	PIO2_3 寄存器	83
6.3.5	A/D 模式	50	6.4.31	PIO2_4 寄存器	84
6.3.6	I ² C 总线模式	50	6.4.32	PIO2_5 寄存器	85
6.3.7	可编程干扰滤波器	50	6.4.33	PIO2_6 寄存器	86
6.4	寄存器描述	50	6.4.34	PIO2_7 寄存器	87
6.4.1	引脚配置寄存器	52	6.4.35	PIO0_10 寄存器	88
6.4.2	PIO0_19 寄存器	55	6.4.36	PIO0_11 寄存器	89
6.4.3	PIO0_20 寄存器	56	6.4.37	PIO0_12 寄存器	90
6.4.4	PIO0_21 寄存器	57	6.4.38	RESET_PIO0_13 寄存器	91
6.4.5	PIO0_22 寄存器	58	6.4.39	PIO0_14 寄存器	92
6.4.6	PIO0_23 寄存器	59	6.4.40	PIO0_15 寄存器	93
6.4.7	PIO0_24 寄存器	60	6.4.41	PIO0_16 寄存器	94
6.4.8	SWDIO_PIO0_25 寄存器	61	6.4.42	PIO0_17 寄存器	95
6.4.9	SWCLK_PIO0_26 寄存器	62	6.4.43	PIO0_18 寄存器	96
6.4.10	PIO0_27 寄存器	63	6.4.44	R_PIO0_30 寄存器	97
6.4.11	PIO2_12 寄存器	64	6.4.45	R_PIO0_31 寄存器	98
6.4.12	PIO2_13 寄存器	65	6.4.46	R_PIO1_0 寄存器	99
6.4.13	PIO2_14 寄存器	66	6.4.47	R_PIO1_1 寄存器	100
6.4.14	PIO2_15 寄存器	67	6.4.48	PIO1_2 寄存器	101
6.4.15	PIO0_28 寄存器	68	6.4.49	PIO1_3 寄存器	102
6.4.16	PIO0_29 寄存器	69	6.4.50	PIO1_4 寄存器	103
6.4.17	PIO0_0 寄存器	70	6.4.51	PIO1_5 寄存器	104
6.4.18	PIO0_1 寄存器	71	6.4.52	PIO1_6 寄存器	105
6.4.19	PIO0_2 寄存器	72	6.4.53	PIO2_8 寄存器	106
6.4.20	PIO0_3 寄存器	73	6.4.54	PIO2_9 寄存器	107
6.4.21	PIO0_4 寄存器	74	6.4.55	PIO2_10 寄存器	108
6.4.22	PIO0_5 寄存器	75	6.4.56	PIO2_11 寄存器	109
6.4.23	PIO0_6 寄存器	76			

第 7 章：LPC122x 引脚配置

7.1	简介	111	7.3	引脚复用	123
7.2	引脚说明	111			

第 8 章：LPC122x 通用 I/O (GPIO)

8.1	本章导读	125	8.3.2	GPIO 引脚值寄存器	126
8.2	简介	125	8.3.3	GPIO 引脚输出寄存器	127
8.2.1	特性	125	8.3.4	GPIO 引脚输出设置寄存器	127
8.3	寄存器描述	125	8.3.5	GPIO 引脚输出清除寄存器	128
8.3.1	GPIO 屏蔽寄存器	126	8.3.6	GPIO NOT 寄存器	128
			8.3.7	GPIO 数据方向寄存器	128

8.3.8	GPIO 中断感应寄存器	128	8.3.12	GPIO 原始中断状态寄存器	129
8.3.9	GPIO 中断双边沿感应寄存器	129	8.3.13	GPIO 屏蔽中断状态寄存器	130
8.3.10	GPIO 中断事件寄存器	129	8.3.14	GPIO 中断清除寄存器	130
8.3.11	GPIO 中断屏蔽寄存器	129			

第 9 章：具备调制解调器控制功能的 LPC122x UART0

9.1	本章导读	131	9.5.10	UART 调制解调器状态寄存器	141
9.2	基本配置	131	9.5.11	UART 高速暂时寄存器	142
9.3	特性	131	9.5.12	UART 自动波特率控制寄存器	142
9.4	引脚描述	131	9.5.12.1	自动波特率	143
9.5	寄存器描述	132	9.5.12.2	自动波特率模式	144
9.5.1	UART 接收器缓冲寄存器（当 DLAB = 0 时，只读）	133	9.5.13	UART 小数分频寄存器	145
9.5.2	UART 发送器保持寄存器（当 DLAB = 0 时，只写）	133	9.5.13.1	波特率计算	146
9.5.3	UART 除数锁存器 LSB 和 MSB 寄存器（当 DLAB = 1 时）	133	9.5.13.1.1	示例 1: UART_PCLK = 14.7456 MHz, BR = 9600	148
9.5.4	UART 中断使能寄存器（当 DLAB = 0 时）	134	9.5.13.1.2	示例 2: UART_PCLK = 12 MHz, BR = 115200	148
9.5.5	UART 中断识别寄存器	134	9.5.14	UART 发送使能寄存器 (TER - 0x4000 8030)	148
9.5.6	UART FIFO 控制寄存器	136	9.5.15	UART RS485 控制寄存器	149
9.5.6.1	DMA 操作	137	9.5.16	UART RS-485 地址匹配寄存器	150
9.5.6.1.1	UART 接收器 DMA	137	9.5.17	UART1 RS-485 延迟值寄存器	150
9.5.6.1.2	UART 发送器 DMA	137	9.5.18	RS-485/EIA-485 模式的操作	150
9.5.7	UART 线路控制寄存器	137	9.5.18.1	RS-485/EIA-485 普通多点模式 (NMM)	150
9.5.8	UART 调制解调器控制寄存器	138	9.5.18.2	RS-485/EIA-485 自动地址检测 (AAD) 模式	151
9.5.8.1	自动流控制	139	9.5.18.3	RS-485/EIA-485 自动方向控制	151
9.5.8.1.1	自动 RTS	139	9.5.18.4	RS485/EIA-485 驱动器延迟时间	151
9.5.8.1.2	自动 CTS	139	9.5.18.5	RS485/EIA-485 输出反转	151
9.5.9	UART 线路状态寄存器	140	9.5.19	UART FIFO 电平寄存器	152
			9.6	架构	152

第 10 章：LPC122x 通用异步收发器 1(UART1)

10.1	本章导读	154	10.5.7	UART 线路控制寄存器	160
10.2	基本配置	154	10.5.8	UART 线路状态寄存器	161
10.3	特性	154	10.5.9	UART 高速暂时寄存器	162
10.4	引脚说明	154	10.5.10	UART 自动波特率控制寄存器	162
10.5	寄存器描述	154	10.5.10.1	自动波特率	163
10.5.1	UART 接收器缓冲寄存器（当 DLAB = 0 时，只读）	156	10.5.10.2	自动波特率模式	164
10.5.2	UART 发送器保持寄存器（当 DLAB = 0 时，只写）	156	10.5.11	UART IrDA 控制寄存器	165
10.5.3	UART 除数锁存器 LSB 和 MSB 寄存器（当 DLAB = 1 时）	156	10.5.12	UART 小数分频寄存器	166
10.5.4	UART 中断使能寄存器（当 DLAB = 0 时）	157	10.5.12.1	波特率计算	167
10.5.5	UART 中断识别寄存器	157	10.5.12.1.1	示例 1: UART_PCLK = 14.7456 MHz, BR = 9600	169
10.5.6	UART FIFO 控制寄存器	159	10.5.12.1.2	示例 2: UART_PCLK = 12 MHz, BR = 115200	169
10.5.6.1	DMA 操作	160	10.5.13	UART 发送使能寄存器	169
10.5.6.1.1	UART 接收器 DMA	160	10.5.14	UART FIFO 电平寄存器	170
10.5.6.1.2	UART 发送器 DMA	160	10.6	架构	170

第 11 章：LPC122x I2C 总线控制器

11.1	本章导读	172	11.5	描述	172
11.2	基本配置	172	11.5.1	I ² C 超快速模式	173
11.3	特性	172	11.6	引脚描述	173
11.4	应用	172	11.7	寄存器描述	174

11.7.1	I ² C 控制置位寄存器 (CONSET - 0x4000 0000)	174	11.10.7.2	仲裁丢失后的数据传输	200
11.7.2	I ² C 状态寄存器 (STAT - 0x4000 0004)	176	11.10.7.3	强制访问 I ² C 总线	201
11.7.3	I ² C 数据寄存器 (DAT - 0x4000 0008)	176	11.10.7.4	I ² C 总线的操作被 SCL 或 SDA 低电平妨碍	201
11.7.4	I ² C 从属地址寄存器 0 (ADR0- 0x4000 000C)	176	11.10.7.5	总线错误	202
11.7.5	I ² C SCL 高电平占空比寄存器和低电平占空比寄存器 (SCLH - 0x4000 0010 和 I2SCLL- 0x4000 0014)	177	11.10.8	I ² C 状态服务程序	202
11.7.5.1	选择适当的 I ² C 数据速率和占空比	177	11.10.8.1	初始化	202
11.7.6	I ² C 控制清零寄存器 (CONCLR - 0x4000 0018)	177	11.10.8.2	I ² C 中断服务	202
11.7.7	I ² C 监控模式控制寄存器	178	11.10.8.3	状态服务程序	203
11.7.7.1	监控模式下的中断	179	11.10.8.4	配合实际应用的状态服务	203
11.7.7.2	监控模式下仲裁丢失	179	11.11 软件示例	203	
11.7.8	I ² C 从属地址寄存器 (ADR[1, 2, 3]- 0x4000 00[20, 24, 28])	179	11.11.1	初始化程序	203
11.7.9	I ² C 数据缓冲寄存器 (DATA_BUFFER - 0x4000 002C)	180	11.11.2	启动主机发送功能	203
11.7.10	I ² C 屏蔽寄存器 (MASK[0, 1, 2, 3]- 0x4000 00[30, 34, 38, 3C])	180	11.11.3	启动主机接收功能	203
11.8 I²C 操作模式	181	11.11.4	I ² C 中断程序	204	
11.8.1	主发送模式	181	11.11.5	无指定模式的状态	204
11.8.2	主接收模式	181	11.11.5.1	状态: 0x00	204
11.8.3	从接收模式	182	11.11.5.2	主机状态	204
11.8.4	从发送模式	183	11.11.5.3	状态: 0x08	204
11.9 I²C 执行和操作	183	11.11.5.4	状态: 0x10	204	
11.9.1	输入滤波器和输出级	184	11.11.6	主发送状态	205
11.9.2	地址寄存器 ADR0 ~ ADR3	185	11.11.6.1	状态: 0x18	205
11.9.3	地址屏蔽寄存器, MASK0 ~ MASK3	185	11.11.6.2	状态: 0x20	205
11.9.4	比较器	185	11.11.6.3	状态: 0x28	205
11.9.5	移位寄存器, DAT	185	11.11.6.4	状态: 0x30	205
11.9.6	仲裁及同步逻辑	185	11.11.6.5	状态: 0x38	206
11.9.7	串行时钟发生器	186	11.11.7	主接收状态	206
11.9.8	时序和控制	186	11.11.7.1	状态: 0x40	206
11.9.9	控制寄存器 ICONSET 和 CONCLR	187	11.11.7.2	状态: 0x48	206
11.9.10	状态解码器和状态寄存器	187	11.11.7.3	状态: 0x50	206
11.10 I²C 操作模式详解	188	11.11.7.4	状态: 0x58	206	
11.10.1	主发送模式	189	11.11.8	从接收状态	207
11.10.2	主接收模式	191	11.11.8.1	状态: 0x60	207
11.10.3	从接收模式	193	11.11.8.2	状态: 0x68	207
11.10.4	从发送模式	195	11.11.8.3	状态: 0x70	207
11.10.5	详细状态表	196	11.11.8.4	状态: 0x78	207
11.10.6	其它状态	199	11.11.8.5	状态: 0x80	208
11.10.6.1	STAT = 0xF8	199	11.11.8.6	状态: 0x88	208
11.10.6.2	STAT = 0x00	199	11.11.8.7	状态: 0x90	208
11.10.7	某些特殊情况	200	11.11.8.8	状态: 0x98	208
11.10.7.1	两个主机同时启动重复起始条件	200	11.11.8.9	状态: 0xA0	208
			11.11.9	从发送状态	209
			11.11.9.1	状态: 0xA8	209
			11.11.9.2	状态: 0xB0	209
			11.11.9.3	状态: 0xB8	209
			11.11.9.4	状态: 0xC0	209
			11.11.9.5	状态: 0xC8	209

第 12 章：LPC122x SSP 控制器

12.1	本章导读	210	12.6.2	SSP 控制寄存器 1	213
12.2	基本配置	210	12.6.3	SSP 数据寄存器	214
12.3	特性	210	12.6.4	SSP 状态寄存器	214
12.4	描述	210	12.6.5	SSP 时钟预分频寄存器	214
12.5	引脚说明	211	12.6.6	SSP 中断屏蔽设置 / 清除寄存器 (IMSC - 0x4004 0014)	215
12.6	寄存器描述	211	12.6.7	SSP 原始中断状态寄存器	215
12.6.1	SSP 控制寄存器 0	212	12.6.8	SSP 屏蔽中断状态寄存器	215

12.6.9	SSP 中断清除寄存器	216	12.7.2.3	CPOL=0, CPHA=1 时的 SPI 格式	218
12.6.10	SSP DMA 控制寄存器	216	12.7.2.4	CPOL = 1 且 CPHA = 0 时的 SPI 格式	219
12.7	功能说明	216	12.7.2.5	CPOL = 1 且 CPHA = 1 时的 SPI 格式	220
12.7.1	德州仪器同步串行帧格式	216	12.7.3	半导体 Microwire 帧格式	221
12.7.2	SPI 帧格式	217	12.7.3.1	Microwire 模式下 CS 相对于 SK 的建立和保持时间要求	222
12.7.2.1	时钟极性 (CPOL) 及相位 (CPHA) 控制	217			
12.7.2.2	CPOL=0, CPHA=0 时的 SPI 格式	217			

第 13 章：LPC122x 16 位计数器 / 定时器 0/1 (CT16B0/1)

13.1	本章导读	224	13.7.5	预分频计数器寄存器	229
13.2	基本配置	224	13.7.6	匹配控制寄存器	229
13.3	特性	224	13.7.7	匹配寄存器	230
13.4	应用程序	224	13.7.8	捕获控制寄存器	230
13.5	描述	225	13.7.9	捕获寄存器	232
13.6	引脚描述	225	13.7.10	外部匹配寄存器	232
13.7	寄存器描述	225	13.7.10.1	DMA 操作	233
13.7.1	中断寄存器	228	13.7.11	计数控制寄存器	233
13.7.2	定时器控制寄存器	228	13.7.12	PWM 控制寄存器	234
13.7.3	定时器计数器	228	13.7.13	单边沿控制的 PWM 输出规则	235
13.7.4	预分频寄存器	229	13.8	定时器操作示例	236
			13.9	架构	237

第 14 章：LPC122x 32 位计数器 / 定时器 0/1 (CT32B0/1)

14.1	本章导读	238	14.7.5	预分频计数器寄存器	242
14.2	基本配置	238	14.7.6	匹配控制寄存器	243
14.3	特性	238	14.7.7	匹配寄存器	244
14.4	应用程序	238	14.7.8	捕获控制寄存器 (CCR 和 CCR)	244
14.5	简介	239	14.7.9	捕获寄存器	245
14.6	引脚描述	239	14.7.10	外部匹配寄存器	245
14.7	寄存器描述	239	14.7.10.1	DMA 操作	247
14.7.1	中断寄存器	241	14.7.11	计数控制寄存器	247
14.7.2	定时器控制寄存器	242	14.7.12	PWM 控制寄存器	248
14.7.3	定时器计数器寄存器	242	14.7.13	单边沿控制的 PWM 输出规则	249
14.7.4	预分频寄存器	242	14.8	定时器操作示例	249
			14.9	架构	250

第 15 章：LPC122x 系统节拍 (SysTick) 定时器

15.1	本章导读	252	15.5.2	系统定时器重载值寄存器	253
15.2	基本配置	252	15.5.3	系统定时器当前值寄存器	254
15.3	特性	252	15.5.4	系统定时器校准值寄存器 (SYST_CALIB - 0xE000E01C)	254
15.4	简介	252	15.6	功能说明	254
15.5	寄存器描述	253	15.7	定时器计算示例	254
15.5.1	系统定时器控制和状态寄存器	253			

第 16 章：LPC122x 实时时钟 (RTC)

16.1	本章导读	256	16.5.4	RTC 控制寄存器	259
16.2	基本配置	256	16.5.5	RTC 中断控制设置 / 清除寄存器	259
16.3	特性	256	16.5.6	RTC 中断状态寄存器	259
16.4	简介	257	16.5.7	RTC 屏蔽中断状态寄存器	259
16.5	寄存器描述	258	16.5.8	RTC 中断清除寄存器	260
16.5.1	RTC 数据寄存器	258	16.6	功能说明	260
16.5.2	RTC 匹配寄存器	258	16.6.1	RTC 启动行为	260
16.5.3	RTC 加载寄存器	258	16.6.2	RTC 匹配中断	261
			16.6.3	在深度睡眠或深度掉电模式下使用 RTC	261

第 17 章：LPC122x 窗口看门狗定时器 (WWDT)

17.1	本章导读	262	17.7.2	看门狗定时器常量寄存器	266
17.2	基本配置	262	17.7.3	看门狗喂狗寄存器	266
17.3	特性	262	17.7.4	看门狗定时器值寄存器	266
17.4	描述	262	17.7.5	看门狗定时器时钟源选择寄存器	267
17.5	时钟和电源控制	263	17.7.6	看门狗定时器警告中断寄存器	267
17.6	看门狗锁定功能	263	17.7.7	看门狗定时器窗口寄存器	268
17.7	寄存器描述	264	17.8	功能框图	268
17.7.1	看门狗模式寄存器	264	17.9	看门狗定时示例	268

第 18 章：LPC122x 比较器

18.1	本章导读	270	18.6.2	电压阶梯寄存器	273
18.2	基本配置	270	18.7	功能说明	274
18.3	特性	270	18.7.1	输入多路复用器	274
18.4	简介	270	18.7.2	基准电压	275
18.5	引脚描述	271	18.7.3	张弛振荡器	275
18.6	寄存器描述	271	18.7.4	中断	276
18.6.1	比较器控制寄存器	271	18.7.5	比较器输出	276

第 19 章：LPC122x ADC

19.1	本章导读	277	19.6.3	A/D 中断使能寄存器	279
19.2	基本配置	277	19.6.4	A/D 数据寄存器	280
19.3	特性	277	19.6.5	A/D 状态寄存器	280
19.4	简介	277	19.6.6	A/D 微调寄存器	281
19.5	引脚描述	277	19.7	操作	281
19.6	寄存器描述	278	19.7.1	硬件触发转换	281
19.6.1	A/D 控制寄存器	278	19.7.2	中断	281
19.6.2	A/D 全局数据寄存器	279	19.7.3	DMA 控制	281

第 20 章：LPC122x Flash ISP/IAP

20.1	本章导读	282	20.5.5.9	IAP 命令处理程序使用的 RAM	288
20.2	特性	282	20.6	代码读保护 (CRP)	288
20.3	bootloader	282	20.6.1	ISP 入口保护	290
20.4	应用程序	282	20.7	ISP 命令	290
20.5	描述	283	20.7.1	解锁 < 解锁代码 >	291
20.5.1	复位后的存储器映射	283	20.7.2	设置波特率 < 波特率 > < 停止位 >	291
20.5.2	Flash 扇区	284	20.7.3	回应 < 设定 >	291
20.5.2.1	信息块	285	20.7.4	写 RAM < 起始地址 > < 字节数 >	292
20.5.2.2	Flash 模块的擦除操作	285	20.7.5	读存储器 < 地址 > < 字节数 >	292
20.5.2.3	Flash 模块的写操作	285	20.7.6	准备写操作的扇区 < 起始扇区号 >	
20.5.3	Boot 处理流程图	286		< 结束扇区号 >	293
20.5.4	有效用户代码的判定标准	286	20.7.7	将 RAM 内容复制到 Flash < Flash 地址 >	
20.5.5	通信协议	287		< RAM 地址 > < 字节数 >	293
20.5.5.1	ISP 命令格式	287	20.7.8	运行 < 地址 > < 模式 >	294
20.5.5.2	ISP 响应格式	287	20.7.9	擦除扇区 < 起始扇区号 > < 结束扇区号 >	294
20.5.5.3	ISP 数据格式	287	20.7.10	扇区查空 < 起始扇区号 > < 结束扇区号 >	294
20.5.5.4	ISP 流量控制	287	20.7.11	读器件标识号	295
20.5.5.5	ISP 命令中止	287	20.7.12	读 Boot 代码版本号	295
20.5.5.6	ISP 过程中的中断	288	20.7.13	比较 < 地址 1 > < 地址 2 > < 字节数 >	296
20.5.5.7	IAP 过程中的中断	288	20.7.14	读 UID	296
20.5.5.8	ISP 命令处理程序使用的 RAM	288	20.7.15	ISP 返回代码	296
			20.8	IAP 命令	297

20.8.1	准备写操作的扇区	299	20.8.8	重新调用 ISP	302
20.8.2	将 RAM 内容复制到 Flash	300	20.8.9	读 UID	302
20.8.3	擦除扇区	300	20.8.10	擦除页	303
20.8.4	扇区查空	301	20.8.11	擦除信息页	303
20.8.5	读器件标识号	301	20.8.12	IAP 状态代码	303
20.8.6	读 Boot 代码版本号	301	20.9	串行线调试 (SWD)Flash 编程接口	304
20.8.7	比较 < 地址 1> < 地址 2> < 字节数 >	302			

第 21 章：LPC122x 通用微 DMA 控制器

21.1	本章导读	305	21.6.13	通道主 - 备用设置寄存器	313
21.2	简介	305	21.6.14	通道主 - 备用清除寄存器	314
21.3	特性	305	21.6.15	通道优先级设置寄存器	314
21.4	描述	305	21.6.16	通道优先级清除寄存器	314
21.4.1	微 DMA 控制器可访问的存储器区域	306	21.6.17	总线错误清除寄存器	315
21.4.2	DMA 系统连接	307	21.6.18	通道 DMA 中断状态寄存器	315
21.5	计时和功耗控制	307	21.6.19	DMA 错误中断使能寄存器	315
21.6	寄存器描述	307	21.6.20	通道 DMA 中断使能寄存器	316
21.6.1	DMA 状态寄存器	309	21.7	功能说明	316
21.6.2	DMA 配置寄存器	309	21.7.1	DMA 控制信号	316
21.6.3	通道控制基址指针寄存器	310	21.7.2	DMA 仲裁	317
21.6.4	通道备用控制基址指针寄存器	310	21.7.3	DMA 优先级	317
21.6.5	通道应请求等待状态寄存器	310	21.7.4	DMA 周期类型	318
21.6.6	通道软件请求寄存器	310	21.7.4.1	无效周期	318
21.6.7	通道使用猝发设置寄存器	311	21.7.4.2	基本周期	318
21.6.8	通道使用猝发清除寄存器	311	21.7.4.3	自动请求周期	318
21.6.9	通道请求屏蔽设置寄存器	312	21.7.4.4	乒乓周期	319
21.6.10	通道请求屏蔽清除寄存器	312	21.7.5	DMA 控制	320
21.6.11	通道使能设置寄存器	312	21.7.5.1	源数据末端指针	321
21.6.12	通道使能清除寄存器	313	21.7.5.2	目标数据末端指针	321
			21.7.5.3	控制数据配置	322

第 22 章：LPC122x CRC 引擎

22.1	本章导读	325	22.5.1	CRC 模式寄存器	326
22.2	简介	325	22.5.2	CRC 种子寄存器	327
22.3	特性	325	22.5.3	CRC 校验和寄存器	327
22.4	描述	326	22.5.4	CRC 数据寄存器	327
22.5	寄存器描述	326	22.6	功能说明	327

第 23 章：LPC122x 整数除法程序

23.1	本章导读	329	23.4	示例	330
23.2	特性	329	23.4.1	初始化	330
23.3	描述	329	23.4.2	有符号除法	330
			23.4.3	带余数的无符号除法	330

第 24 章：LPC122x 串行调试接口 (SWD)

24.1	本章导读	331	24.4	引脚说明	331
24.2	特性	331	24.5	调试注意事项	331
24.3	简介	331	24.5.1	调试限制	331
			24.5.2	调试连接	331

第 25 章：LPC122x ARM Cortex-M0 参考资料

25.1	简介	333	25.2.2	集成的可配置调试	334
25.2	关于 Cortex-M0 处理器和内核外设	333	25.2.3	Cortex-M0 处理器的特性小结	334
25.2.1	系统级接口	334	25.2.4	Cortex-M0 内核外设	334

25.3	处理器	335	25.4	存储器访问指令	358
25.3.1	编程模型	335	25.4.4.1	ADR	358
25.3.1.1	处理器模式	335	25.4.4.1.1	语法	358
25.3.1.2	堆栈	335	25.4.4.1.2	操作	359
25.3.1.3	内核寄存器	335	25.4.4.1.3	限制	359
25.3.1.3.1	通用寄存器	336	25.4.4.1.4	条件标志	359
25.3.1.3.2	堆栈指针	336	25.4.4.1.5	示例	359
25.3.1.3.3	链接寄存器	337	25.4.4.2	LDR 和 STR, 立即数偏移量	359
25.3.1.3.4	程序计数器	337	25.4.4.2.1	语法	359
25.3.1.3.5	程序状态寄存器	337	25.4.4.2.2	操作	359
25.3.1.3.6	异常屏蔽寄存器	339	25.4.4.2.3	限制	360
25.3.1.3.7	CONTROL 寄存器	339	25.4.4.2.4	条件标志	360
25.3.1.4	异常和中断	340	25.4.4.2.5	示例	360
25.3.1.5	数据类型	340	25.4.4.3	LDR 和 STR, 寄存器偏移量	360
25.3.1.6	Cortex 微控制器软件接口标准	340	25.4.4.3.1	语法	360
25.3.2	存储器模型	341	25.4.4.3.2	操作	360
25.3.2.1	存储区、类型和属性	341	25.4.4.3.3	限制	361
25.3.2.2	存储器系统的存储器访问秩序	342	25.4.4.3.4	条件标志	361
25.3.2.3	存储器访问行为	342	25.4.4.3.5	示例	361
25.3.2.4	软件的存储器访问秩序	343	25.4.4.4	LDR, PC-相对	361
25.3.2.5	存储器的字节存储顺序	344	25.4.4.4.1	语法	361
25.3.2.5.1	小端格式	344	25.4.4.4.2	操作	361
25.3.3	异常模型	344	25.4.4.4.3	限制	361
25.3.3.1	异常状态	344	25.4.4.4.4	条件标志	361
25.3.3.2	异常类型	344	25.4.4.4.5	示例	361
25.3.3.3	异常处理程序	345	25.4.4.5	LDM 和 STM	361
25.3.3.4	向量表	346	25.4.4.5.1	语法	362
25.3.3.5	异常优先级	346	25.4.4.5.2	操作	362
25.3.3.6	异常的进入和返回	347	25.4.4.5.3	限制	362
25.3.3.6.1	异常的进入	347	25.4.4.5.4	条件标志	362
25.3.3.6.2	异常返回	348	25.4.4.5.5	示例	362
25.3.4	故障处理	349	25.4.4.5.6	错误示例	362
25.3.4.1	锁定	349	25.4.4.6	PUSH 和 POP	363
25.3.5	电源管理	350	25.4.4.6.1	语法	363
25.3.5.1	进入睡眠模式	350	25.4.4.6.2	操作	363
25.3.5.1.1	等待中断	350	25.4.4.6.3	限制	363
25.3.5.1.2	等待事件	350	25.4.4.6.4	条件标志	363
25.3.5.1.3	Sleep-on-exit	350	25.4.4.6.5	示例	363
25.3.5.2	从睡眠模式唤醒	350	25.4.5	通用数据处理指令	363
25.3.5.2.1	从 WFI 或 sleep-on-exit 唤醒	350	25.4.5.1	ADC、ADD、RSB 和 SUB	364
25.3.5.2.2	从 WFE 唤醒	351	25.4.5.1.1	语法	364
25.3.5.3	电源管理编程提示	351	25.4.5.1.2	操作	365
25.4	指令集	351	25.4.5.1.3	限制	365
25.4.1	指令集汇总	351	25.4.5.1.4	示例	366
25.4.2	内部函数	353	25.4.5.2	AND、ORR、EOR 和 BIC	366
25.4.3	About the instruction descriptions	354	25.4.5.2.1	语法	366
25.4.3.1	操作数	354	25.4.5.2.2	操作	366
25.4.3.2	使用 PC 或 SP 时的限制	354	25.4.5.2.3	限制	367
25.4.3.3	移位操作	354	25.4.5.2.4	条件标志	367
25.4.3.3.1	ASR	354	25.4.5.2.5	示例	367
25.4.3.3.2	LSR	355	25.4.5.3	ASR、LSL、LSR 和 ROR	367
25.4.3.3.3	LSL	355	25.4.5.3.1	语法	367
25.4.3.3.4	ROR	356	25.4.5.3.2	操作	368
25.4.3.4	地址对齐	356	25.4.5.3.3	限制	368
25.4.3.5	相对 PC 的表达式	356	25.4.5.3.4	条件标志	368
25.4.3.6	条件执行	357	25.4.5.3.5	示例	368
25.4.3.6.1	条件标志	357	25.4.5.4	CMP 和 CMN	368
25.4.3.6.2	条件代码后缀	358	25.4.5.4.1	语法	368

25.4.5.4.2 操作	368	25.4.7.3.4 条件标志	376
25.4.5.4.3 限制	368	25.4.7.3.5 示例	376
25.4.5.4.4 条件标志	369	25.4.7.4 DSB	376
25.4.5.4.5 示例	369	25.4.7.4.1 语法	376
25.4.5.5 MOV 和 MVN	369	25.4.7.4.2 操作	376
25.4.5.5.1 语法	369	25.4.7.4.3 限制	376
25.4.5.5.2 操作	369	25.4.7.4.4 条件标志	376
25.4.5.5.3 限制	369	25.4.7.4.5 示例	376
25.4.5.5.4 条件标志	370	25.4.7.5 ISB	377
25.4.5.5.5 示例	370	25.4.7.5.1 语法	377
25.4.5.6 MULS	370	25.4.7.5.2 操作	377
25.4.5.6.1 语法	370	25.4.7.5.3 限制	377
25.4.5.6.2 操作	370	25.4.7.5.4 条件标志	377
25.4.5.6.3 限制	370	25.4.7.5.5 示例	377
25.4.5.6.4 条件标志	370	25.4.7.6 MRS	377
25.4.5.6.5 示例	370	25.4.7.6.1 语法	377
25.4.5.7 REV、REV16 和 REVSH	371	25.4.7.6.2 操作	377
25.4.5.7.1 语法	371	25.4.7.6.3 限制	377
25.4.5.7.2 操作	371	25.4.7.6.4 条件标志	377
25.4.5.7.3 限制	371	25.4.7.6.5 示例	377
25.4.5.7.4 条件标志	371	25.4.7.7 MSR	378
25.4.5.7.5 示例	371	25.4.7.7.1 语法	378
25.4.5.8 SXT 和 UXT	371	25.4.7.7.2 操作	378
25.4.5.8.1 语法	371	25.4.7.7.3 限制	378
25.4.5.8.2 操作	372	25.4.7.7.4 条件标志	378
25.4.5.8.3 限制	372	25.4.7.7.5 示例	378
25.4.5.8.4 条件标志	372	25.4.7.8 NOP	378
25.4.5.8.5 示例	372	25.4.7.8.1 语法	378
25.4.5.9 TST	372	25.4.7.8.2 操作	378
25.4.5.9.1 语法	372	25.4.7.8.3 限制	378
25.4.5.9.2 操作	372	25.4.7.8.4 条件标志	378
25.4.5.9.3 限制	372	25.4.7.8.5 示例	378
25.4.5.9.4 条件标志	373	25.4.7.9 SEV	379
25.4.5.9.5 示例	373	25.4.7.9.1 语法	379
25.4.6 跳转和控制指令	373	25.4.7.9.2 操作	379
25.4.6.1 B、BL、BX 和 BLX	373	25.4.7.9.3 限制	379
25.4.6.1.1 语法	373	25.4.7.9.4 条件标志	379
25.4.6.1.2 操作	373	25.4.7.9.5 示例	379
25.4.6.1.3 限制	374	25.4.7.10 SVC	379
25.4.6.1.4 条件标志	374	25.4.7.10.1 语法	379
25.4.6.1.5 示例	374	25.4.7.10.2 操作	379
25.4.7 其他指令	374	25.4.7.10.3 限制	379
25.4.7.1 BKPT	375	25.4.7.10.4 条件标志	379
25.4.7.1.1 语法	375	25.4.7.10.5 示例	379
25.4.7.1.2 操作	375	25.4.7.11 WFE	380
25.4.7.1.3 限制	375	25.4.7.11.1 语法	380
25.4.7.1.4 条件标志	375	25.4.7.11.2 操作	380
25.4.7.1.5 示例	375	25.4.7.11.3 限制	380
25.4.7.2 CPS	375	25.4.7.11.4 条件标志	380
25.4.7.2.1 语法	375	25.4.7.11.5 示例	380
25.4.7.2.2 操作	375	25.4.7.12 WFI	380
25.4.7.2.3 限制	375	25.4.7.12.1 语法	380
25.4.7.2.4 条件标志	376	25.4.7.12.2 操作	380
25.4.7.2.5 示例	376	25.4.7.12.3 限制	381
25.4.7.3 DMB	376	25.4.7.12.4 条件标志	381
25.4.7.3.1 语法	376	25.4.7.12.5 示例	381
25.4.7.3.2 操作	376	25.5 外设	381
25.4.7.3.3 限制	376	25.5.1 关于 ARM Cortex-M0	381

25.5.2	嵌套向量中断控制器	381	25.5.3.4	应用中断和复位控制寄存器	388
25.5.2.1	使用 CMSIS 访问 Cortex-M0 NVIC 寄存器	382	25.5.3.5	系统控制寄存器	389
25.5.2.2	中断设置 - 使能寄存器	382	25.5.3.6	配置和控制寄存器	390
25.5.2.3	中断清除 - 使能寄存器	382	25.5.3.7	系统处理程序优先级寄存器	390
25.5.2.4	中断设置 - 挂起寄存器	383	25.5.3.7.1	系统处理程序优先级寄存器 2	390
25.5.2.5	中断清除 - 挂起寄存器	383	25.5.3.7.2	系统处理程序优先级寄存器 3	390
25.5.2.6	中断优先级寄存器	384	25.5.3.8	SCB 使用的提示和技巧	391
25.5.2.7	电平有效的中断和脉冲中断	384	25.5.4	系统定时器, SysTick	391
25.5.2.7.1	中断的硬件和软件控制	385	25.5.4.1	SysTick 控制和状态寄存器	391
25.5.2.8	NVIC 的使用提示和技巧	385	25.5.4.2	SysTick 重装值寄存器	392
25.5.2.8.1	NVIC 编程提示	385	25.5.4.2.1	计算 RELOAD 值	392
25.5.3	系统控制块	386	25.5.4.3	SysTick 当前值寄存器	392
25.5.3.1	Cortex-M0 SCB 寄存器的 CMSIS 映射	386	25.5.4.4	SysTick 校准值寄存器	392
25.5.3.2	CPUID 寄存器	386	25.5.4.5	SysTick 的使用提示和技巧	393
25.5.3.3	中断控制和状态寄存器	387			

第 26 章：增补信息

26.1	缩写	394	26.3.3	商标	395
26.2	参考资料	394	26.4	表	396
26.3	法律信息	395	26.5	图	402
26.3.1	定义	395	26.6	内容	403
26.3.2	免责声明	395			

注意：关于本文及相关产品的重要说明详见“法律信息”一节。